VERTICAL REPLENISHMENT BY UNMANNED AERIAL VEHICLES

LIU PEIDONG

(*B.Eng.(Hons.), NUS*)

A THESIS SUBMITTED

FOR THE DEGREE OF MASTER OF ENGINEERING

ELECTRICAL AND COMPUTER ENGINEERING

NATIONAL UNIVERSITY OF SINGAPORE

2015

Declaration

I hereby declare that the thesis is my original work and it has been written by me in its entirety. I have duly acknowledged all the sources of information which have been used in the thesis.

This thesis has also not been submitted for any degree in any university previously.

[illegible]

Liu Peidong
18st May 2015

Acknowledgments

When looking back on the past three years in the Unmanned Aerial Vehicle (UAV) Research Group, National University of Singapore (NUS), I am surprised see that I have grown up in many ways. I would like to thank everyone who has helped me and taught me along the way of my growth.

First of all, I would like to express my deep and sincere gratitude to my supervisor, Professor Ben M. Chen for his guidance, encouragement, and patience during my studies at NUS. He taught me, instructed me and inspired me not only in the academic studies but also in the daily lives as well.

Special thanks are given to our NUS UAV research group. I will never forget the days when working with my team mates day and night, especially when we were preparing for competitions. Particularly, I would like to thank Dr. Peng Kemao, Dr. Wang Biao, Dr. Cai Guowei and Dr. Lin Feng for their valuable technical suggestions. I am also grateful for the generous help and accompanies from Dr. Dong Xiangxu, Dr. Zhao Shiyu, Dr. Wang Fei, Dr. Phang Sweeking, Dr. Cui Jinqiang, Dr. Kevin, Ang Zongyao, Mr. Li Kun, Mr. Bai Limiao, Mr. Lai Shupeng, Mr. Pang Tao, Mr. Ke Yijie, Mr. Wang Kangli, Miss Lin Jing, Miss Deng Di, Miss Li Xiang, Mr. Yang Zhaolin, Mr. Bi Yingcai, Mr. Li Jiaxin, Mr. Shan Mo, Mr. Qin Hailong and Mr. Liu Wenqi. I will never forget the days and nights when we fight for the champions and play basketball together. It has been a wonderful time with all of you.

Finally, I am grateful to my parents, my girlfriend, Miss Wang Siqi, and my younger sister. Without their understanding and wholehearted support, it would be impossible for me to finish my studies.

Contents

Summary

This master thesis presents the development of an unmanned helicopter in hardware design as well as algorithm developments for vertical replenishment. It consists of eight chapters. The introduction and conclusion are addressed in the first chapter and last chapter, respectively. From Chapter 2-6, each chapter describes the development of a single functional module. Chapter 7 presents the methods used for integrating all these modules together to form a fully functional system for the vertical replenishment.

This thesis starts with the development and configurations of the hardware platform in Chapter 2. As one of the foundations for upper layer algorithm developments and implementations, the hardware platform is constructed in a systematic way. The chapter covers the methods used for bare helicopter modification, sensor selections, on-board computer selections and system integrations etc..

Chapter 3 addresses the dynamic modeling of the constructed platform, which is the foundation for the automatic flight controller design. The nonlinear dynamic model will be presented based on the Newton-Euler formulation and the aerodynamics of the helicopter. In order to employ advanced modern control techniques, a linear state-space model structure is derived. The unknown variables of the model are further identified and validated with real flight data.

Based on the obtained linear dynamic model in Chapter 3, a two layer flight controller is developed in Chapter 4. The controller consists of an inner-loop controller and an outer-loop controller. The inner-loop controller is used to stabilize the attitude of the helicopter and is designed with H_∞ control technique. The outer-loop controller is used for the translational movements of the helicopter and is designed with the so-called robust and perfect tracking (RPT) control method. Real flight experiment results are presented to evaluate the performance of the controller.

Measurements are essential and important for automatic flight control systems. Chapter 5 addresses the state estimation methods developed for precision height measurement and cargo localization based on 2D laser scanner and camera, respectively. Furthermore, it presents the algorithm used for cargo detections through a monocular camera. Experiments are conducted to evaluate the performance of the state estimation algorithms. The results show that the state estimations are satisfactory for our requirements.

In chapter 6, algorithms for trajectory generation are presented. The algorithm can smooth the flight trajectories if given the velocity, acceleration constraints as well as the distance need to fly. For example, if the helicopter is commanded to fly towards 5 m along the x-axis, the trajectory generator will interpret it to 50 Hz set-points commands for the flight controller to execute. It is an important module for the helicopter to finish the vertical replenishment task.

Lastly, chapter 7 integrates all the above modules together to form a functional system for vertical replenishment. The system is divided into five layers, each layer contains one or more the above mentioned modules. The interactions among these layers are well defined so that they can behave orderly. Flight experiments to delivery cargos from one ship to another are conducted and the experiment results show that the developed system is capable for the vertical replenishment.

List of Figures

Chapter 1

Introduction

1.1 Motivations

Helicopters are extensively used for vertical replenishment (VERTREP), which is a type of underway replenishment for seaborne vessels. In this application, a helicopter lifts the cargo from the supplying ship and delivers it to the receiving ship as shown in Fig. 1.1. The major advantage of VERTREP is that the operation does not require physical contact of the vessels, which reduces the chance of collision. The operation is usually done by manned helicopters, which possesses potential risks for both pilots and ground crews, especially in extreme weather conditions. A catastrophic accident happened in 1998, where a CH-46 SeaKnight helicopter crashed into the Mediterranean Sea while conducting a VERTREP operation with a Spruance-class destroyer, resulting in death of pilots and crewmen.

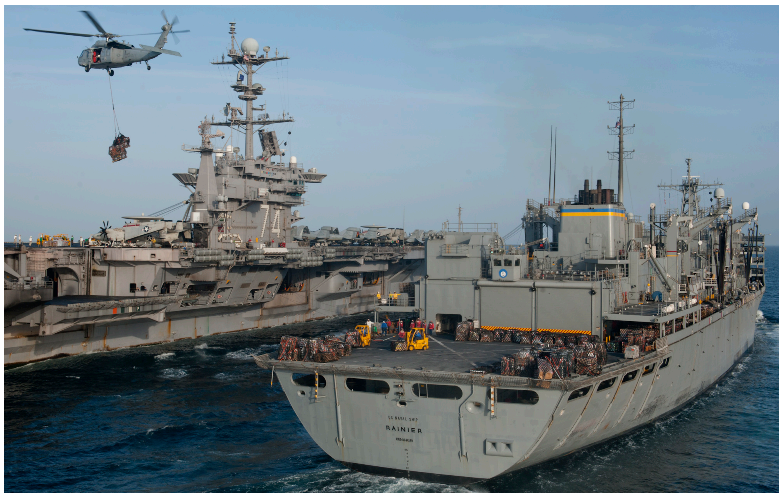


Figure 1.1: Vertical replenishment by U.S. Navy (Use of released U.S. Navy imagery does not constitute product or organizational endorsement of any kind by the U.S. Navy.)

The recent advancement of unmanned aerial vehicles (UAVs) however has opened the possibility of using unmanned rotor-crafts for this kind of cargo transportation tasks, which can reduce both risk and cost to a large extent. In this thesis, we will develop a fully autonomous helicopter to tackle this problem.

1.2 Challenges and aims of this thesis

To accomplish the vertical replenishment tasks by an unmanned helicopter, there are several challenges need to address.

1. **Precision**, to deliver the cargos from one ship to another automatically, the unmanned helicopter has to grab and unload the cargos precisely. Without the help of highly accurate measurement devices, such as differential GPS (which can provide *cm*-level localization accuracy), it is difficult to achieve;

2. **Disturbances**, the movements of the ships, windy weather and the loading of a cargo to the helicopter will usually bring in disturbances to the flight controller of the unmanned helicopter; It further affects the control performances of the helicopter;

3. **Uncertainties**, a fully automatic unmanned helicopter is required to finish the vertical replenishment tasks without any (or very little) human interventions after take-off; There are many unexpected situations may occur, which may result in mission failures if the system is not properly designed;

Thus, in this thesis, we are trying to solve the above mentioned challenges for the vertical replenishment tasks by an unmanned helicopter. The helicopter is controlled by fusing measurements from different kinds of sensors, such as inertial measurement unit, GPS, 2D laser scanner and camera. The precision challenge is expected to be overcome by using all these sensor measurements for the helicopter flight control, navigation and guidance. A robust H_∞ optimal controller is going to be developed to address the external disturbances challenge. It has been proved that the H_∞ controller can minimize the effect from the external disturbances to the controlled output. A flowchart of procedures is to be developed in this thesis to address the uncertainty challenge. It will be used to handle unexpected events as many as possible.

1.3 Related works

Extensive researches have been carried out on unmanned helicopters for the last two decades, focusing on the automatic flight control, navigation and guidance [28].

Variety of flight control methods have been proposed and implemented for unmanned helicopters. The most common used control methods belong to the linear state-space control technique. For example, in [6], a linear quadratic regulator (LQR) is designed for the inner-loop of a Yamaha RMAX robotic helicopter from Carnegie Mellon University; in [41] a linear quadratic gaussian (LQG) controller is designed for a mini quadrotor; in [13] and [12], a robust H_∞ controller is designed for the inner-loop of a raptor-90 helicopter. Some nonlinear controllers have also been designed and implemented in the literature. For example, in [43], a composite nonlinear feedback (CNF) controller is designed for a raptor-90 helicopter; a nonlinear feedback linearization based controller was also designed and implemented for a quadrotor in [38]. In the literature, fuzzy logic has also been used for the flight controller design. For example, in [23], a fuzzy logic based controller is designed and implemented for a Joker-Maxi II helicopter. Furthermore, the apprenticeship learning method has also been used to control helicopters for aerobatics flights [1].

Several navigation and guidance systems have been implemented for unmanned helicopters, such as a vision-aided navigation system is presented in [52]; an autonomous landing system for a miniature aerial vehicle is illustrated in [5]; A bearing only measurements based formation flight method and an onboard software system for formation flight are proposed and implemented in [55] and [19].

Limited works have demonstrated the applications of unmanned helicopters for vertical replenishment or cargo transportation. To the best of our knowledge, there are only few (semi-)autonomous slung load systems using unmanned helicopters reported in the literature. In [46], a K-MAX helicopter is modified for autonomous operation and used for slung load transportation in Afghanistan by the United States army. A helicopter designed to solve the general slung load transportation problem with long ropes is presented in [7]. A group of researchers have also proposed the estimations of load position and velocity in such system [9]. Besides, some researchers have also investigated the ability of cargo transporting with the collaboration of multiple UAVs [36] [30] [8]. With this cooperative structure, the size and cost of each individual UAV can be reduced. All these systems involve human intervention in the loop. The ground operators are required to pick up and fasten the cargoes for the unmanned helicopters. In [35], a team of quadrotor have been used to transport and construct the cubic structures fully autonomously. However, these quadrotors are aided by a motion capture system in an indoor environment, which can localize the quadrotors and the cargoes precisely (in *mm*-level). In [11], the AirMule UAV from UrbanAero, is developed to transport up to 500 kg of cargo to places as far as 50 km away and has been used to transport cargo in Israel for military purposes. The cargo transportation problem can also be solved by a rigid claw mechanism such as those appeared in [42,47].

When solving this UAV cargo transportation problem, most of the existing works assume that the loading and unloading positions are accurately known or the human operators can help them find the cargo. This assumption is reasonable in a few occasions where the environment is fully in control, but may not be valid for the more general cases. To expand the horizon of applications a small-scale UAV can do, an intelligent navigation and guidance system which can provide high-quality measurements and guidance information for UAV automatic flight control needs to be developed. One elegant solution is to integrate a computer vision sub-system for target searching and tracking. In fact, vision-based target detection and localization have been investigated intensively. Some of them rely on visual targets with special shapes and features, such as [40] in which range estimation has been carried out based on specific geometric features including points, lines and curves. Others target on more general objects such as a helipad [45], a mobile ground vehicle [18, 34] or another UAV [50]. In addition, there is also a trend in integrating visual information in feedback control for mobile robot autonomous grasping and manipulation [31].

1.4 Contributions and outlines of the thesis

In this thesis, we propose and implement a comprehensive system for vertical replenishment by an unmanned helicopter which incorporates a small-size single-rotor helicopter with onboard sensors and processors, an innovative cargo grabbing mechanism, a set of UAV autonomous guidance, navigation and control (GNC) algorithms, and a cargo searching and localization vision system.

- The developed system has shown the capability for cargo precision grabbing, cargo delivery and cargo unloading. The precision challenge has been overcome by using vision-based guidance and laser-based precision height control for the helicopter;

- The developed system has shown the capability to fly stably and accomplish the tasks under external disturbances, such as movements of ships and windy weather; It is resulted from the development of a robust H_∞ optimal controller;
- The developed system has also shown the capability to accomplish the vertical replenishment tasks under uncertainties through the design of a flowchart of procedures; There are also some insufficiencies shown for the developed system, such as the helicopter cannot take action accordingly when it dropped off a cargo unexpectedly during the experiments;

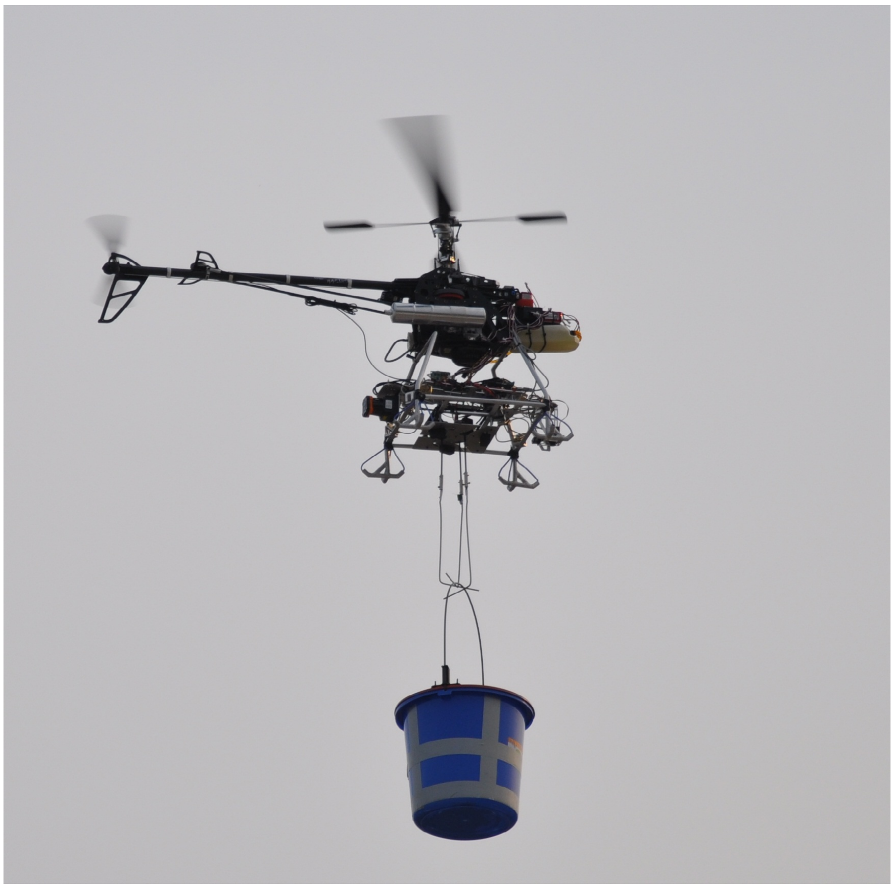


Figure 1.2: NUS^{2}TLion developed by NUS UAV Group

The developed UAV system, named NUS^{2}T-Lion, has taken part in the 2nd AVIC Cup – International UAV Innovation Grand Prix (UAVGP), which was held in Beijing in September 2013. In this competition, the rotary-wing UAVs from various participating teams are required to automatically transport cargos between two parallel moving ships. The cargos are in the form of buckets with handles and they are initially placed within colored circles drawn on the surface of the first ship. Circles with a different color are drawn on the other ship, indicating the unloading positions. The ships are simulated by ground platforms moving on railways. It is set-up to simulate the vertical replenishment. During the competition, we are the only team that finished the competition requirements. Our developed helicopter has successfully transported the cargos from one ship to another automatically. It further shows our contributions to this problem. Fig. 1.2 shows a snap shot of NUS^{2}T-Lion carrying the the cargo bucket in this Grand Prix.

The outlines of this thesis is as follows. The thesis contains two main parts, i.e., the developments of individual functional blocks of the system (from Chapter 2 to Chapter 6) and the integrations of the these blocks (Chapter 7).

In detail, Chapter 2 will talk about the design and integration of the UAV hardware system. Chapter 3 will present the aerodynamics of traditional helicopters and derive a linear state-space model of our developed helicopter platform for future automatic controller design. Chapter 4 will present the methods used to design an automatic flight controller for our helicopter. Chapter 5 will present the methods used to estimate the flight status of the helicopter, such as the position, velocity, height etc., for helicopter automatic control, navigation, and guidance. In Chapter 6, a trajectory generator is going to be developed. It is useful to interpret discrete events to acceptable set-point commands for the flight controller to execute. Chapter 7 will present the methods used for integrating all the developed modules together as a functional system for vertical replenishment. Finally, concluding remarks are made in Chapter 8.

Chapter 2

Hardware Configurations

2.1 Introduction

In robotic community, hardware platform is the foundation of the research. There are many different kinds of robot platforms have been built in the literature, varying from under-water robots, surface robots, ground robots to aerial robots. Fig. 2.1 shows two aerial robots developed by the Unmanned Aerial Vehicle (UAV) research group in NUS. The quality of the platform's construction usually dominates the automatic controller design of the overall robotic system. It further affects the deployments of the other high-level algorithms on the robots, such as environment perceptions as well as decision makings. A good selection of the sensors and actuators as well as platform structure designs are the keys to good platform constructions. In control perspective, the sensors' placement position, actuators' placement position and the platform structures will affect the state-space model (i.e., A, B and C matrices) of the robots directly. Interested readers are recommended to refer [17] for theoretical guidelines of platform constructing.

In this chapter, a systematic approach of constructing an aerial robot hardware platform is to be presented. The robot, named as NUS^{2}TLion, is used as a test-bed for implementing the automatic control algorithms as well as other high-level intelligent mission algorithms, such as the algorithm to tackle the vertical replenishment problem in this thesis. The outline of this chapter is as follows: an overview of the hardware system is provided first; the approaches for platform selection, sensor selections on-board computing equipments selections etc. are then given in the following sections; finally, the method for system integrations is presented.

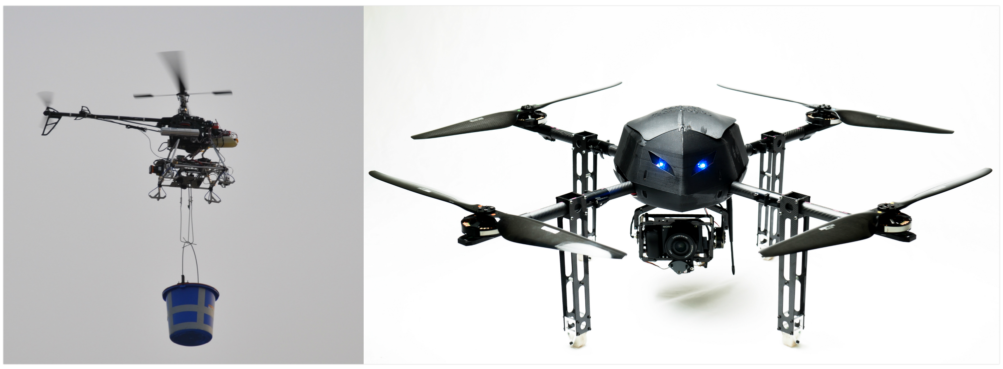


Figure 2.1: Aerial robots developed by NUS Unmanned Aerial Vehicle Research Group

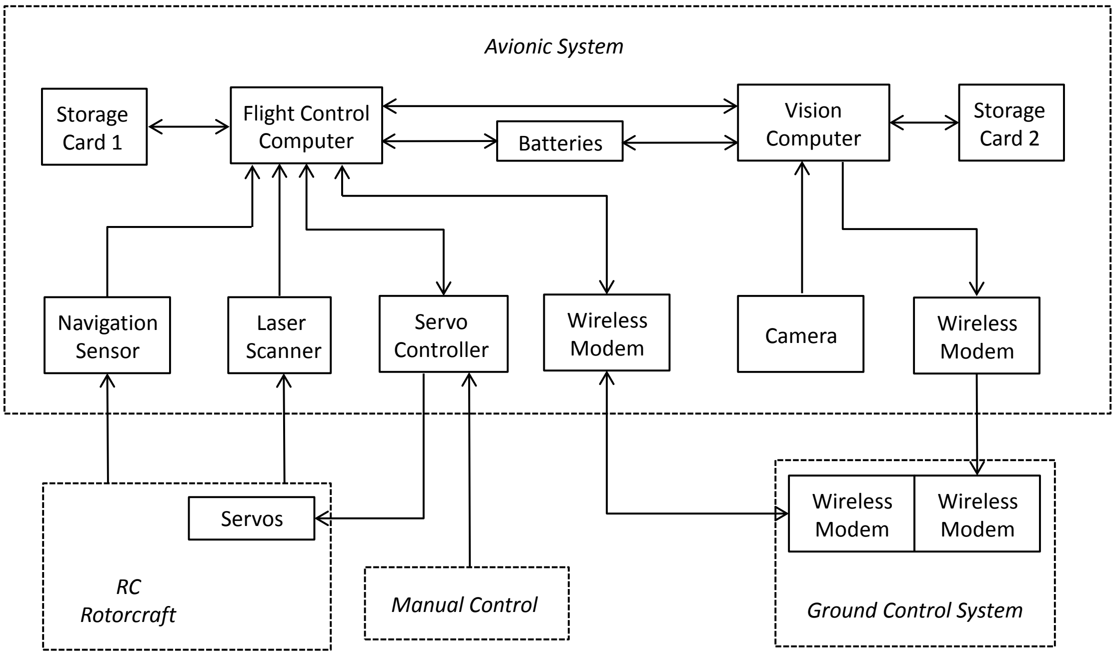


Figure 2.2: Hardware configuration of NUS^{2}T-Lion rotorcraft system

2.2 Overview of the hardware system

The hardware configuration of NUS^{2}T-Lion follows the rotor-craft UAV structure proposed in [12]. As illustrated in Fig. 2.2 in which each block represents an individual hardware device, the whole system is constituted by four main parts, namely a bare rotor-craft platform, onboard avionic system, a manual control system and a ground control system (GCS). While the manual control system and the GCS are quite standard for all kinds of UAV systems, the choices of the bare rotorcraft platform and its onboard avionic system are usually application dependent. For this case, they should be selected and integrated specifically for the UAV cargo transportation task. It is believed that by designing the hardware configuration effectively, difficulties for the later software algorithm development can be minimized.

2.3 Bare rotorcraft platform

The Thunder Tiger Raptor 90 SE Nitro radio-controlled (RC) helicopter is adopted as the bare rotor-craft platform in this work. It is a hobby-level single rotor helicopter originally designed for acrobatic flights. As compared with other commercial off-the-shelf (COTS) RC rotor-crafts such as Turbulence D3 and Observer Twin, Raptor 90 SE provides a reliable structural design and equivalent flight performance, at approximately half the price.

However, with the original Raptor 90's nitro engine and nitro fuel tank, the endurance of the UAV can barely reach 8 minutes with full load avionics. This is not sufficient for practical applications. To overcome this limitation, the original nitro engine is replaced by a gasoline counterpart, which is a product from Zenoah with model number G270RC. With the more efficient gasoline engine, a full-tank Raptor 90 can fly up to 30 minutes. This greatly widens the range of potential applications this UAV can do and it is especially beneficial to the cargo transportation task.

Unfortunately, this endurance improvement comes with two trade-offs. First, the vibration of the whole platform intensifies due to the gasoline engine. Second, the ignition magnet inside Zenoah G270RC is so large that its magnetic field can badly affect the onboard sensors. To overcome the vibration issue, wire rope isolators are used to protect the onboard avionics and filter out unwanted high frequency noises. The solution will be discussed in Section 2.6. For the problem of magnetic interference, the final solution is to replace the electro-magnetic ignition system inside the engine with a pure electric ignition system. With this modification, the onboard sensors, especially the magneto-meter, all work in the way they originally should.

2.4 Mechanical manipulator

To cope with the cargo transportation task, there must be a loading mechanism integrated into the helicopter platform. By comparing the solution of a rigid claw-like grabbing mechanism and a long flexible rope hooking mechanism, the former is more precise in picking up the cargos, while the latter can avoid descending the UAV too low to the ship surface where the aerodynamic ground effect becomes significant.

In this work, an innovative design incorporating advantages from both sides has been proposed. The solution is a claw-like grabbing mechanism with very long arms (see Fig. 2.3). With this design, the UAV can keep a safe distance to the ship surface, and at the same time, grab

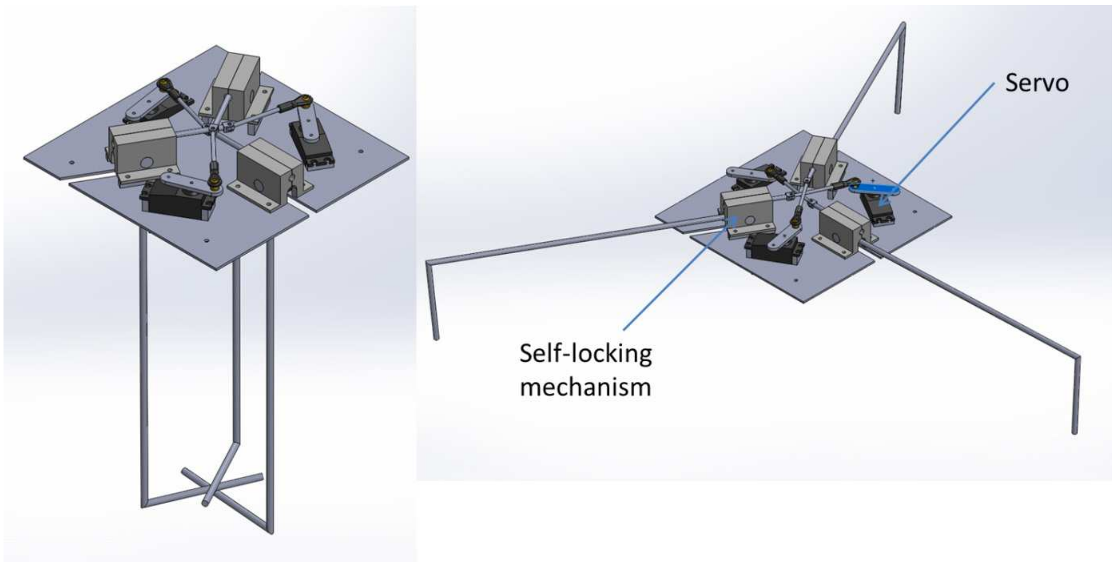


Figure 2.3: Grabbing mechanism in closed and open configurations

and release the cargo in a precise and reliable way. Another highlight of this design is its omni-directional feature, meaning no matter in which direction the cargo handle is oriented, it is not necessary for the UAV to adjust its heading to align accordingly. This saves time and minimizes unnecessary UAV maneuvers.

In addition, this design features a self-locking mechanism commonly used in landing gears of hobby-grade fixed-wing planes. The mechanism is enclosed in the rectangular boxes as shown in Fig. 2.3 with each box supports one arm and is powered by one servo motor. When the claw fully opens or closes, there is a slider inside the box to lock the position of the servo motor. In this way, the servo motors consume zero power while carrying a heavy cargo.

A load sensing mechanism which can differentiate a successful cargo loading from a failure is also installed. This mechanism acts as a safeguard in cases where the UAV makes a grasping action but the targeted cargo is not loaded successfully. By knowing that the cargo loading is unsuccessful, the UAV can descend and try grasping the cargo again. The detailed design is shown in Fig. 2.4, where four limit switches, which send out electrical signals when pushed down, are installed on the customized landing skid. The baseplate of the claw is rigidly attached to a hollow rectangular plate on its top. The rectangular plate is then resting on the cross-over beams of the landing skid via four springs. When the claw is loaded, the rectangular plate compresses the spring and trigger one or more of the limit switches. When the claw is unloaded, the springs push up the rectangular plate to release the limit switches.

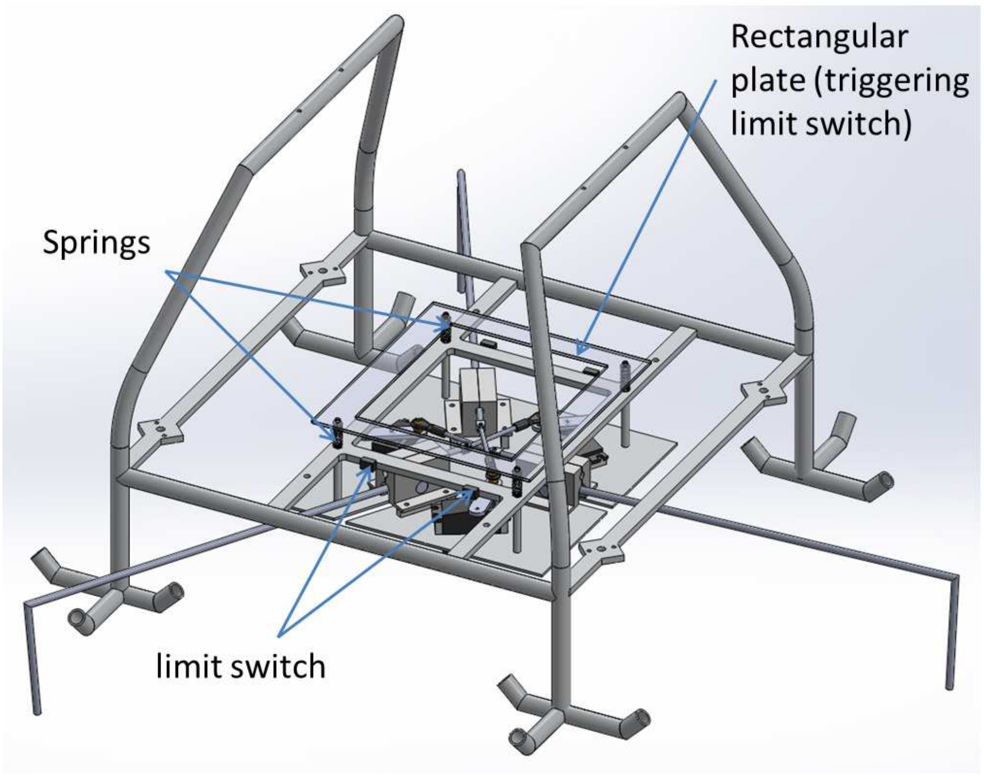


Figure 2.4: Landing gear with bucket grabbing and load sensing functions

2.5 Avionic system

To realize fully autonomous flight, onboard avionic system with sensors, processors and other electronic boards has to be designed. All components used on NUS^{2}T-Lion are the carefully chosen COTS products up to date. Fig. 2.5 gives a complete view of the onboard system with the key components indicated. The details and usage of these components are explained as follows.

2.5.1 Onboard sensors

The SBG IG-500N GPS/INS (GPS aided inertial navigation system) unit is chosen as the fundamental navigation sensor for NUS^{2}T-Lion. SBG IG-500N is one of the world's smallest GPS enhanced attitude and heading reference system (AHRS) embedded with an extended Kalman filter (EKF). It includes a micro-electromechanical systems (MEMS) based IMU, a GPS receiver and a barometer. It is able to provide precise and drift-free 3D orientation and position even during aggressive maneuvers, updated at 100 Hz. With its presence, the UAV's attitude, velocity and position can be consistently obtained, despite the fact that the position measurement from

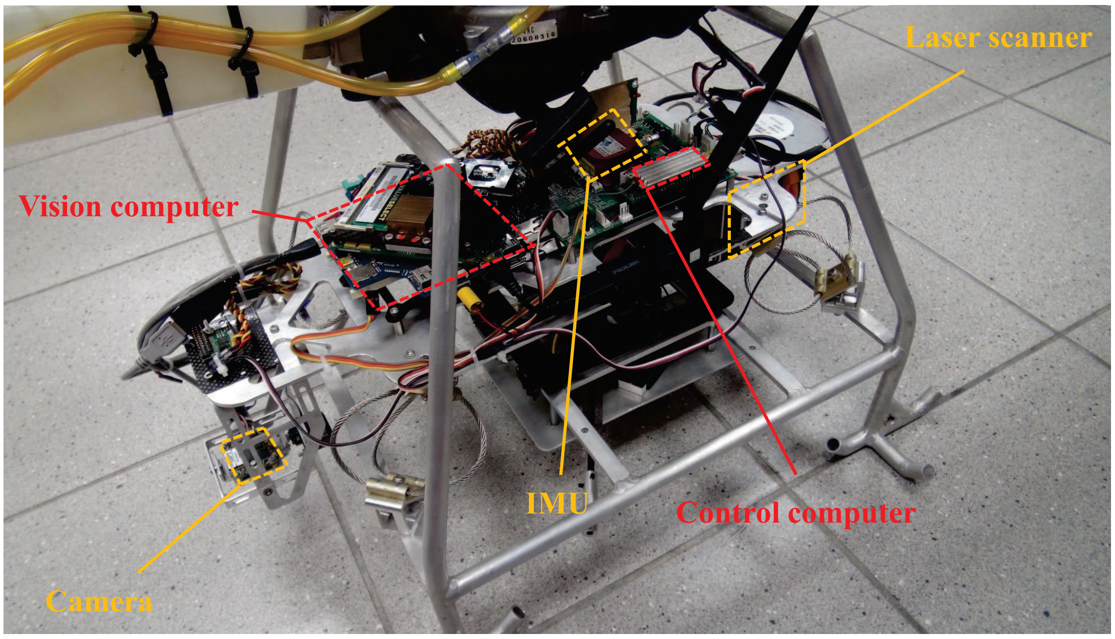


Figure 2.5: Onboard avionic system of NUS^{2}T-Lion

Table 2.1: Main specifications of IG-500N.

Specifications	IG-500N
Attitude range	360° in three axes
Attitude accuracy	±0.5° (pitch, roll), ±1° (heading)
Accelerometer range	±5 g
Gyroscope range	±300°
Magnetometer range	±1.2 Gauss
GPS accuracy in CEP	2.5 m (horizontal), 5 m (vertical)
Output rate (Hz)	{1, 25, 50, 75, 100} selectable
Dimensions	36 × 49 × 22 mm
Weight	46 g (with aluminum enclosure)
Power consumption	550 mW @ 5.0V

IG-500N alone is not accurate enough for the precise cargo loading and unloading task.

Its key specifications are summarized in Table 2.1.

The second main sensor used onboard of NUS^{2}T-Lion is the mvBlueFOX camera from Matrix Vision. It is a compact industrial CMOS camera, compatible to any computers with USB ports. A superior image quality makes it suitable for both indoor and outdoor applications. In addition, it incorporates field-programmable gate array (FPGA), which reduces the computer load to the minimum during image pre-processing. The standard Hi-Speed USB interface guarantees an easy integration without any additional interface board. In this specific cargo transportation application, it is the main guidance sensor for locating the cargos and their unloading points.

For cargo transportation applications, height measurement from GPS/INS or barometer may not be accurate enough for the UAV to pick up or drop the cargo appropriately. The UAV may even crash onto the surface of the cargo platform because of inaccurate height measurement, resulting in catastrophic consequences. While vision sensor or 1-D laser range finder may accomplish the task, the former can only be relied on when the visual target is within the field of view and the latter cannot handle ground surfaces with scattered obstacles. To make the height measurement accurate and consistent, a scanning laser range finder is the best choice. The laser scanner codenamed URG-30LX from Hokuyo is installed in the system. It has a maximum range of 30 m with fine resolution of 5 mm and it can scan its frontal 270° fan-shaped area with a resolution of 0.25°.

2.5.2 Onboard computers

There are two onboard computers in the avionic system; one for the implementation of guidance, navigation and control algorithms, and the other more powerful one dedicated for vision processing. With this dual-computer structure, the vision algorithm can be implemented and tested separately at the development stage and it is very convenient to upgrade to a more powerful vision computer in future without modifying the control hardware and software system. It also improves the reliability of the overall system since this structure ensures control stability even when the vision computer malfunctions or encounters run-time errors. It happens more frequently on the vision computer compared to the control counterpart because the vision algorithm usually involves more sophisticated calculations and logics. If it ever happens, the UAV should still fly safely with the control computer alone and there will be enough time for human pilot to take over and land the UAV safely.

For the onboard control computer, it collects measurement data from various sensors, performs sensor filtering and fusion, executes flight control law, and outputs control signals to carry out the desired control actions. In addition, it is also responsible for communicating with the GCS as well as data logging. Beging a light-weight yet powerful embedded computer for real-time tasks, the Gumstix Overo Fire embedded computer is selected for this purpose. It has a main processor running at 720 MHz and a DSP coprocessor. The main processor is an OMAP3530 ARM chip from Texas Instruments and it is one of the fastest low-power embedded processor as of writing. Moreover, it has built-in Wi-Fi module which saves the weight of an additional communication device.

For the onboard vision computer, it is mainly for implementing image processing algorithms, including color segmentation, object identification, object tracking and localization. Image processing tasks are usually computationally intensive and hence require powerful processors to run the algorithms in real time. We have chosen the Mastermind computer from Ascending Technologies. It has an Intel Core i7 processor but is still small and light enough to be carried by NUS^{2}T-Lion. It also has abundant communication ports to interact with peripheral devices like USB cameras and Wi-Fi devices. One UART port is used to communicate with the flight control computer.

2.5.3 Servo controller

An 8-channel pulse-width modulation (PWM) servo controller, UAV100 from Pontech, is used to enable servo control by either an onboard computer via serial port (automatic mode) or output from an RC receiver (manual mode). The switching between the two modes depends on the state of an auxiliary channel from the RC transmitter. While the UAV maneuvers autonomously in the air, it is desirable to have a failsafe feature to allow the ground pilot to take over control during emergencies. Besides, this servo controller has the function of outputting quantitative servo values. This makes collecting manual or autonomous control data possible and it is a necessary requirement for UAV dynamic modeling and system identification.

2.5.4 Avionic hub

A customized printed circuit board (PCB) called LionHub (see Fig. 2.6) is developed as an expansion board to host various hardware devices. It is an improved version of a similar board introduced in [33]. The aforementioned IG-500N navigation sensor, the Gumstix Overo Fire computer, and the UAV100 servo control board can be physically installed on the slots of this PCB hub and connected to the onboard power regulator and other essential components. Besides the mounting slots, extra mounting holes on LionHub are used to lock the installed modules to resist the vibration and shock generated in flight and landing. With the introduction of LionHub, manual wire wrap is minimized to improve the reliability and quality of the system. A serial RS-232 to TTL level voltage converter is included in LionHub to connect the output of IG-500N to the UART port of Gumstix. Furthermore, to power up all the avionics, linear regulators designed in the avionic hub to convert a power input from a 4-cell Lithium-Polymer (LiPo) battery to 12 V and 5 V outputs with sufficient current delivering. The 12 V output port powers

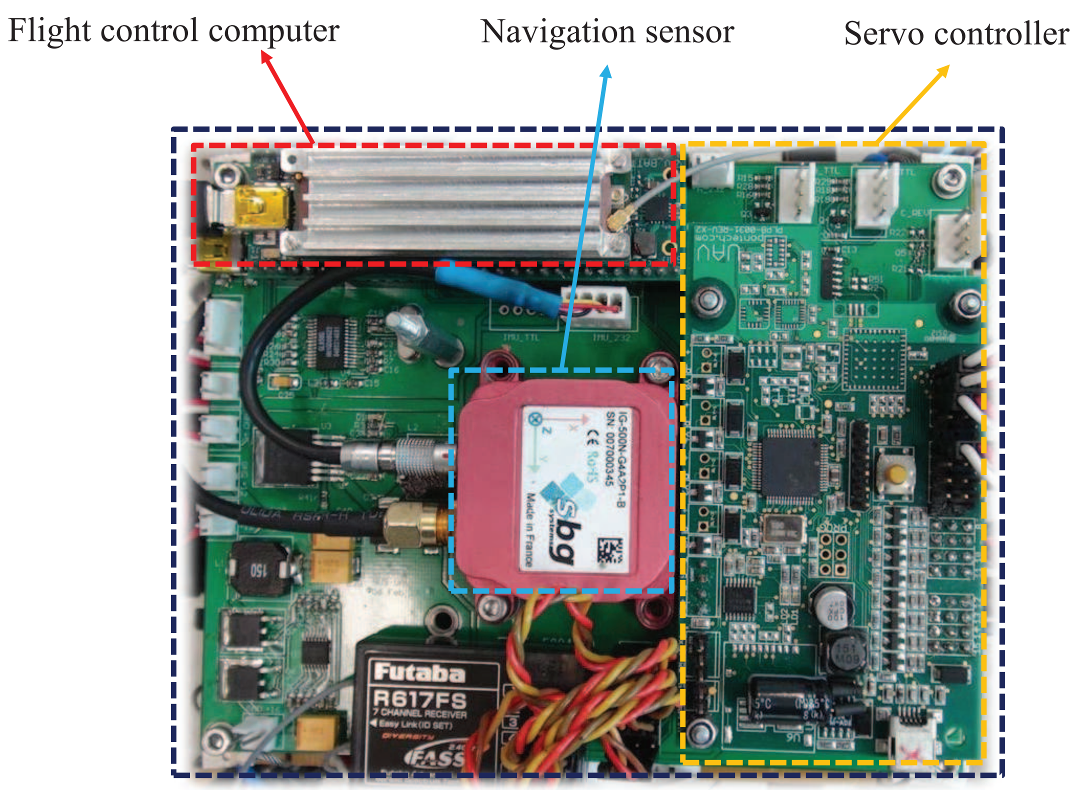


Figure 2.6: Control hub with all hardware components attached

the Mastermind computer and the Lidar sensor, while the 5 V output port powers the Gumstix computer and other electronic boards.

2.6 System integration

After selecting and configuring the individual mechanical and avionic components, all these hardware parts need to be assembled to form a coherent UAV platform. To accomplish this task, special attention needs to be paid in the layout design of the overall onboard system and anti-vibration consideration.

2.6.1 Layout design

The first priority is to place the navigation sensor as close to the center of gravity (CG) of the whole UAV platform as possible to minimize the so-called lever effect, which causes bias to acceleration measurement when the UAV platform performs rotational motion. Note that all the other electronic boards on the LionHub will also be located near to the CG position because they

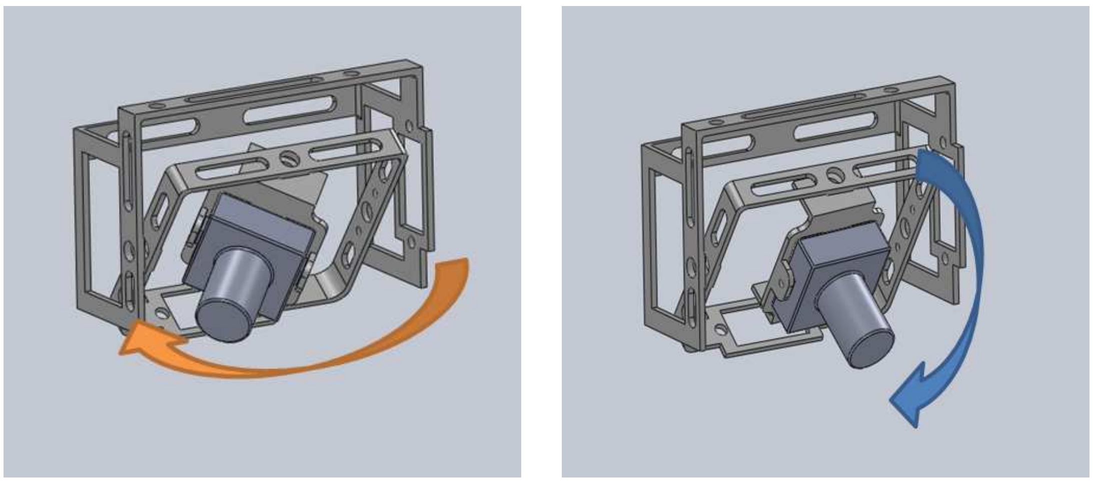


Figure 2.7: Camera pan-tilt mechanism

are rigidly linked to the IMU. Usually there is no problem to align the IMU so that its planar *x*- and *y*-axis position coincide with the UAV CG. However, since a minimum space between the helicopter belly and the onboard system is needed for bumping avoidance, compromise needs to be made in the vertical *z*-axis and software compensation can be implemented to minimize the measurement error caused by this vertical offset. In order to have better signal reception, the GPS antenna is placed on the horizontal fin of the helicopter tail. Again, its 3D position offset to the IMU needs to be compensated.

The next priority goes to the camera sensor. By considering the fact that the UAV usually flies forward to search for targets and hovers right above the cargo for loading and unloading, the best position to place the camera is at the nose of the helicopter. In addition, a controlled pan-tilt gimbal (see Fig. 2.7) is designed to host the camera sensor so that it always looks vertically downwards despite the UAV rolling and pitching motions. Taking advantage of the camera's wide viewing angle, even when the UAV descends to the lowest altitude for cargo grabbing, the camera can still see the cargo which should be right under the UAV CG.

In order to retain CG balancing, the cargo loading mechanism needs to be installed precisely under the UAV CG. In this way, the UAV roll and pitch dynamics will not change too much after the cargo is loaded, thus the same set of robust control law can be used. This design also makes sure that controlling the UAV CG to the correct planar position is equivalent to controlling the cargo loading mechanism to the correct position so that a precise grabbing action can take place.

The placement of the remaining onboard components are less restricted. The overall CG balancing can be achieved by adjusting their mounting positions. For our case, the laser scanner

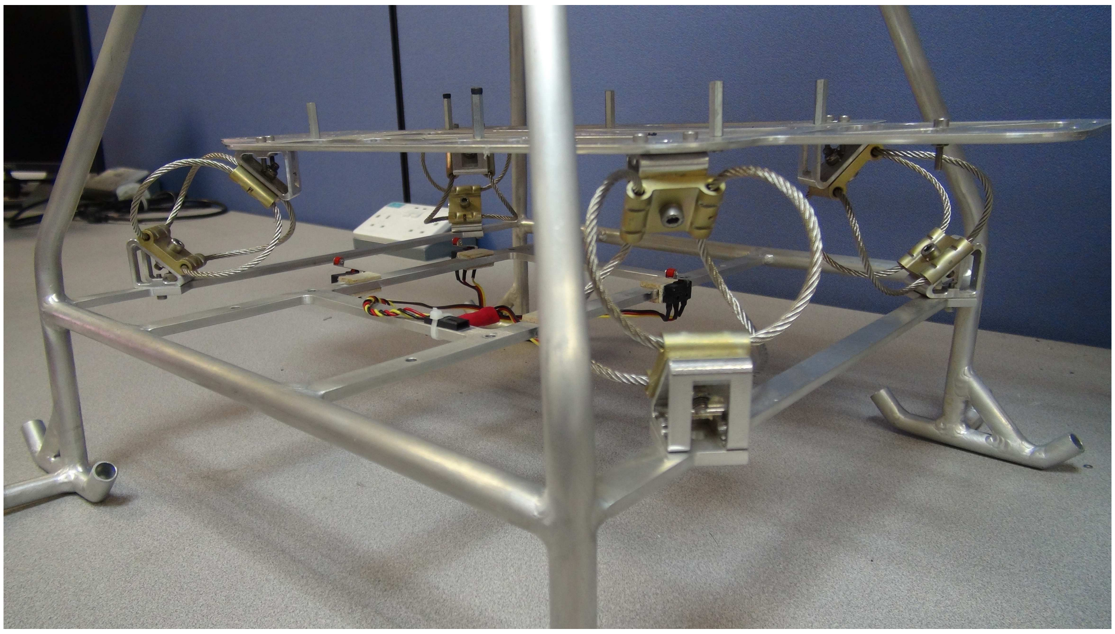


Figure 2.8: Anti-vibration using wire rope isolators

is positioned at the back end of the onboard system, scanning downwards. The vision computer is put at the frontal part to counter-balance the laser scanner and to make wiring to the camera sensor shorter. The battery is slotted at a bottom middle position so that it adds on minimal moment of inertia to the whole UAV platform.

With the above layout design, the distribution of mass is balanced, the control challenge caused by the cargo loading is minimized, and all sensors are working properly. An aluminium plate is used to mount all the onboard components and it sits on four wire rope isolators (see Fig. 2.8) which helps to solve the mechanical vibration problem. The final integrated unmanned helicopter is shown in Fig .2.9.

2.6.2 Anti-vibration design

Anti-vibration for the onboard avionics is one of the most important considerations in hardware design. It can improve the overall performance of the UAV system significantly by reducing wear and tear of the mechanical and electrical connectors and attenuating unwanted noises at high frequencies. Indeed, the replacing of nitro engine with a gasoline engine amplifies the vibration issue. The main vibration sources on NUS^{2}T-Lion are from its main rotors and the engine. From a frequency analysis of the in-flight acceleration data logged while hovering (see Fig. 2.10), one can see that the most significant high-frequency vibration occurs at 22 Hz.

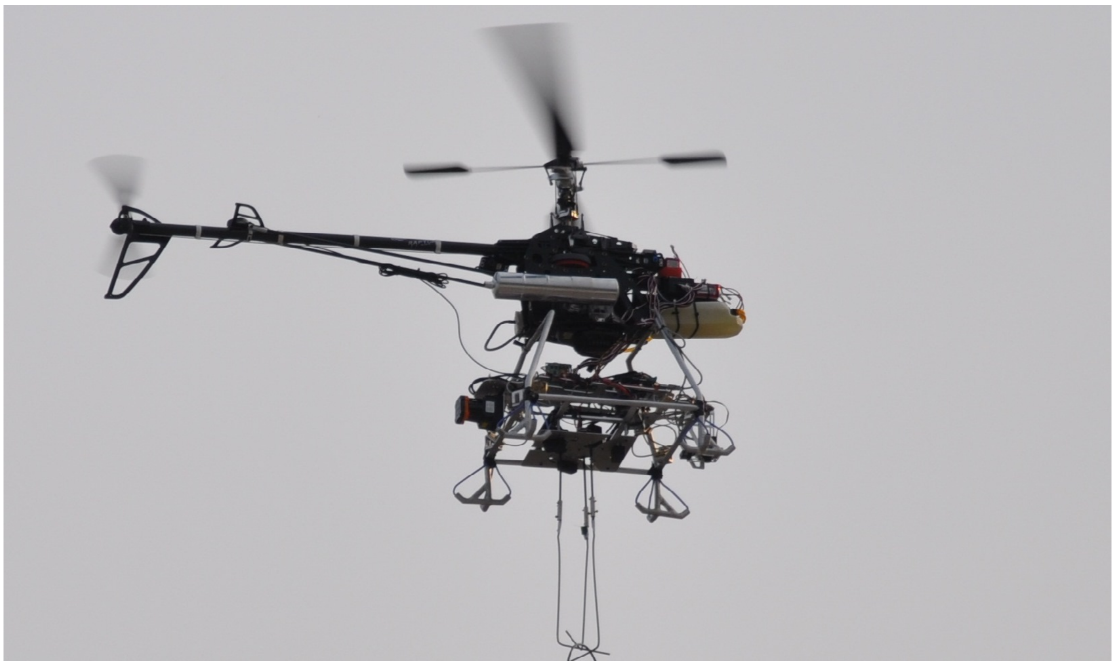


Figure 2.9: Unmanned Helicopter: NUS^{2}TLion

To attenuate noise at this specific frequency, the CR4-400 compact wire rope isolator from Enidine is used. According to the CR series manual provided by Enidine, the best stiffness for the chosen isolator, K_v can be calculated as

$$K_v = W_s(2\pi f_i/3)^2/g, \tag{2.1}$$

where W_s is the static load on every isolator, f_i is the input excitation frequency needs to be attenuated, and g is the gravitational constant. For our case, about 2 kg of onboard load is shared by four isolators, which gives $W_s = 4.9$. By substituting also $f_i = 22$ and $g = 9.781$ into (2.1), K_v can be calculated as 1.06 kN/m which is best matched by the vibration stiffness value obtained by CR4-400 mounted in a '45° Compression/Roll' mode. There are also the 'Pure Compression' and 'Shear/Roll' mounting methods, but the '45° Compression/Roll' mode is the best for attenuating vibration in all three axes. After the installation of wire rope isolators, Fig. 2.11 shows the improved performance of acceleration measurement. As compared to the original graph, the higher frequency noises have been reduced by 10 times or more.

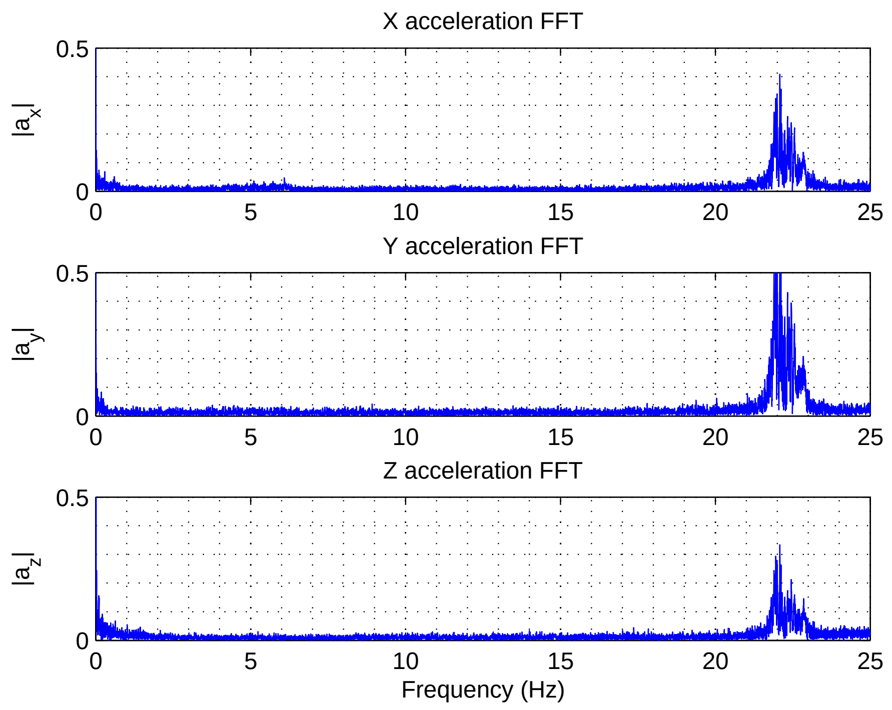


Figure 2.10: Frequency analysis of acceleration without isolators

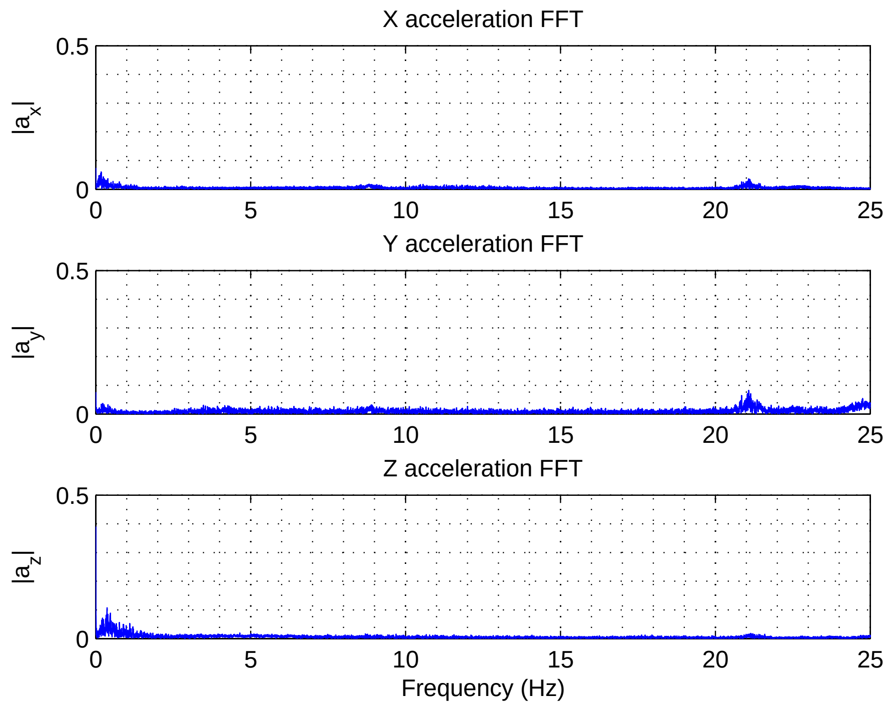


Figure 2.11: Frequency analysis of acceleration with isolators

2.7 Conclusion

In this chapter, systematic procedures are presented for the hardware configurations of the unmanned helicopter. Considering the particular application requirement, i.e., long endurance and large payload, a gasoline engine powered bare helicopter is chosen. Proper measurement devices, such as IMU/AHRS, laser scanner and camera are selected to give satisfactory measurements for future automatic navigation and guidance. Detailed mechanical design for object manipulating is then presented and vibration issues induced by the gasoline engine is also addressed. Finally, the system integration procedures are illustrated.

Chapter 3

Modeling of the Helicopter Platform

3.1 Introduction

The dynamic modeling lays foundations for automatic controller design. An accurate model is essential for using advanced modern control techniques to stabilize the plant. Thus, we present the procedures to derive the linear state-space model of the helicopter in this chapter. Many works related to flight dynamics modeling of miniature rotorcraft have been conducted since the early 1990s and some successful results have been achieved based on either the first-principles modeling approach or the system identification method [12].

In this chapter, the modeling procedures follow the methods proposed in [12] and [39]. The nonlinear dynamic model will be derived based on the Newton-Euler formulation and aerodynamics of the helicopter. In order to employ advanced modern control techniques, such as H_∞ method, a linear state-space model is derived with 23 unknown variables based on the nonlinear dynamic model at near hovering condition. These unknown variables are further identified based on frequency domain identification method using a commercial software. To validate the accuracy of the identified model, comparisons between real flight data and outputs obtained through the dynamic model will be conducted.

3.2 Frames and notations

The helicopter is considered to be a rigid body with 6 degrees of freedom (DoF), free to move in three translational directions and to rotate about all three axes simultaneously. Basically, there are three different right-handed helicopter reference-frame are defined throughout the helicopter dynamic modeling, i.e., the local North-East-Down (NED) coordinate frame, the vehicle body carried NED coordinate frame and the helicopter body coordinate frame.

The local NED frame is defined for the use of the Newtonian mechanics in the helicopter modeling, flight control and navigation. Its origin and axes are defined as following:

1. The origin (denoted by O_n) is arbitrarily fixed to a point on the earth's surface.
2. The X-axis (denoted by X_n) points towards the geodetic north.
3. The Y-axis (denoted by Y_n) points towards the geodetic east.
4. The Z-axis (denoted by Z_n) points downwards along the ellipsoid normal.

Coordinate vectors expressed in the local NED frame are denoted with a subscript "n". More specifically, the position vector, P_n, the velocity vector, V_n, and the acceleration vector, a_n, of the NED coordinate system are adopted and are respectively defined as

$$P_n = \begin{bmatrix} x_n \\ y_n \\ z_n \end{bmatrix} \quad V_n = \begin{bmatrix} u_n \\ v_n \\ w_n \end{bmatrix} \quad \text{and} \quad a_n = \begin{bmatrix} a_{x,n} \\ a_{y,n} \\ a_{z,n} \end{bmatrix}$$

The definition of vehicle body carried frame is similar to that of the local NED frame, except the origin is located at the center of gravity of the helicopter. Coordinate vectors expressed in the vehicle-carried NED frame are denoted with a subscript "nv".

The body coordinate system is vehicle-carried and directly defined on the body of the flying vehicle. Its origin and axes are given as following

1. The origin (denoted by O_b) is located at the center of gravity of the flying vehicle.
2. The X-axis (denoted by X_b) points forward along the helicopter longitudinal direction (i.e., through the nose).
3. The Y-axis (denoted by Y_b) points to the right along the lateral direction when seen from the above.
4. The Z-axis (denoted by Z_b) points downwards and perpendicular to the other axes (i.e., form a right-hand coordinate frame).

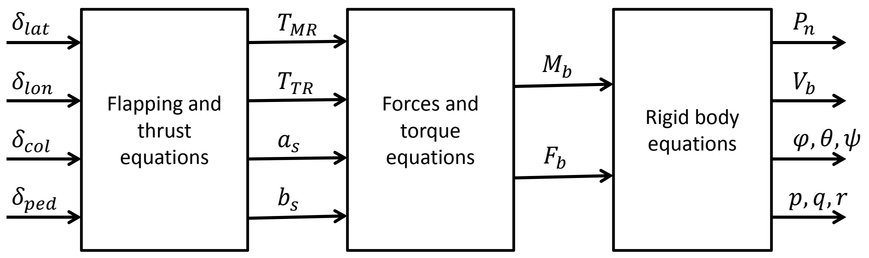


Figure 3.1: Structure of the flight dynamics model

The notations u, v and w are used to denote the translatory velocities of the helicopter, relative to local NED frame, expressed in body frame. a_x, a_y and a_z denote the measured accelerations relative to local NED frame, expressed in body frame.

The attitude and rotary movements of the helicopter are described by a number of variables. The angular velocities, p, q and r, denote the roll, pitch and yaw motions relative to the body frame, respectively. The Euler-angles, ϕ, θ and ψ, define the angles between the body frame and the body-carried NED frame after a roll, pitch and yaw movement, respectively.

The inputs to the helicopter actuators are denoted as δ_{lat}, δ_{lon}, δ_{col} and δ_{ped}. The aileron servo input δ_{lat} affects the roll channel movement (i.e., p and v). The elevator servo input δ_{lon} affects the pitch channel movement (i.e., q and u). The collective pitch servo input δ_{col} affects the heave channel movement (i.e., w). Lastly, the rudder servo input δ_{ped} affects the yaw channel movement (i.e., r).

The movements of the helicopter are mainly driven by the forces and moments generated by the main rotor and tail rotor. By altering the collective pitch angle through δ_{col}, the magnitude of the thrust (T_{MR}) generated by main rotor is controlled. When altering the orientation of the thrust vector through δ_{lat} and δ_{lon}, the plane spanned by the main rotor is tilted, and defines a new plane denoted as the "tip path plane" (TPP). The angles a_s and b_s are used to denote the longitudinal and lateral flapping angles of the TPP, respectively. The notation T_{TR} denotes the thrust generated by the tail rotor.

3.3 Aerodynamics modeling of the helicopter

The aerodynamics of the helicopter have been studied extensively in the literature (see [44], [32], etc). Fig. 3.1 shows the structure of the flight dynamics model.

3.3.1 Rigid body dynamics

The motion of the helicopter is described in the “rigid body equations” box. The Newton-Euler formulation is used here to model the helicopter’s movements as the helicopter is considered as a rigid body.

Define

$$\omega = \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{3.1}$$

then by the Newton-Euler formulation, the differential equations of the movements can be obtained as follows,

$$\begin{aligned} M_b &= J \cdot \dot{\omega} + \omega \times (J \cdot \omega) \\ F_b &= m\dot{V}_b + m\omega \times V_b \end{aligned} \quad , \tag{3.2}$$

where M_b is the sum of external torques and F_b is the sum of external forces, J is the moment of inertial matrix of the helicopter and is defined as

$$J = \begin{bmatrix} J_{xx} & 0 & 0 \\ 0 & J_{yy} & 0 \\ 0 & 0 & J_{zz} \end{bmatrix}, \tag{3.3}$$

V_b is the helicopter velocity relative to the inertial frame, expressed in body frame, and is defined as

$$V_b = \begin{bmatrix} u \\ v \\ w \end{bmatrix}. \tag{3.4}$$

Equation 3.2 can be further transformed and expanded as

$$\dot{\omega} = \begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} \frac{(J_{yy}-J_{zz})\cdot q\cdot r+L}{J_{xx}} \\ -\frac{(J_{xx}-J_{zz})\cdot p\cdot r-M}{J_{yy}} \\ \frac{(J_{xx}-J_{yy})\cdot p\cdot q+N}{J_{zz}} \end{bmatrix} \tag{3.5}$$

$$\dot{V}_b = \begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \end{bmatrix} = \begin{bmatrix} \frac{f_{x,b}}{m} + v\cdot r - w\cdot q \\ \frac{f_{y,b}}{m} - u\cdot r + w\cdot p \\ \frac{f_{z,b}}{m} + u\cdot q - v\cdot p \end{bmatrix}, \tag{3.6}$$

where

$$F_b = \begin{bmatrix} f_{x,b} \\ f_{y,b} \\ f_{z,b} \end{bmatrix}, \qquad M_b = \begin{bmatrix} L \\ M \\ N \end{bmatrix}. \tag{3.7}$$

The equation describing the relationship between euler angles and the angular rates is described by [12]

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & sin(\phi)tan(\theta) & cos(\phi)tan(\theta) \\ 0 & cos(\phi) & -sin(\phi) \\ 0 & \frac{sin(\phi)}{cos(\theta)} & \frac{cos(\phi)}{cos(\theta)} \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{3.8}$$

Equations 3.5, 3.6 and 3.8 describe the motion of the rigid body.

3.3.2 Force and torque equations

This section describes the forces and torques acting on the helicopter. The forces and torques are mainly generated by the main rotor and tail robot as shown in Fig. 3.1. The forces acting on the helicopter are decomposed into two parts, one part drives the translatory movements of the helicopter and another part is transformed to torques which cause the rotary movements of the helicopter.

Forces

The external forces acting on the helicopter are mainly from the main rotor thrust T_{MR}, tail rotor thrust T_{TR} and the gravitational force F_{mg}. Decomposing the forces in body frame, the total force can be expressed as

$$F_b = \begin{bmatrix} T_{MR,x} + T_{TR,x} + F_{mg,x} \\ T_{MR,y} + T_{TR,y} + F_{mg,y} \\ T_{MR,z} + T_{TR,z} + F_{mg,z} \end{bmatrix} = \begin{bmatrix} -T_{MR}sin(a_s) + 0 - mg \cdot sin(\theta) \\ T_{MR}sin(b_s) + T_{TR} + mg \cdot sin(\phi)\cos(\theta) \\ -T_{MR}cos(a_s)cos(b_s) + 0 + mg \cdot cos(\phi)cos(\theta) \end{bmatrix}. \tag{3.9}$$

Torques

The total torque acting on the helicopter are mainly caused by the main rotor T_{MR} and tail rotor T_{TR}. Assume the main rotor thrust T_{MR} is $[l_m\ y_m\ h_m]_b^T$ away from the center of gravity, the tail rotor thrust T_{TR} is $[-l_t\ 0\ h_t]_b^T$ away from the center of gravity, thus the torques equation can be obtained as

$$M_b = \begin{bmatrix} l_m \\ y_m \\ h_m \end{bmatrix} \times \begin{bmatrix} T_{MR,x} \\ T_{MR,y} \\ T_{MR,z} \end{bmatrix} + \begin{bmatrix} -l_t \\ 0 \\ h_t \end{bmatrix} \times \begin{bmatrix} T_{TR,x} \\ T_{TR,y} \\ T_{TR,z} \end{bmatrix}. \tag{3.10}$$

The force equation and torque equation shown in Equation 3.9 and Equation 3.10 are simplified representations. There are also other sources of forces and moments, such as the force and moments generated by fuselage, horizontal fin, vertical fin etc.. For complete forces equations, interested readers are suggested to refer [12] for more details.

3.3.3 Flapping and thrust equations

Main rotor thrust

The main rotor is the source of lift. The thrust generated by the main rotor can be described by the following equations [25]

$$T_{MR} = \frac{\rho \Omega_{MR} R_{MR}^2 C_{l\alpha,MR} b_{MR} c_{MR}}{4} (w_{bl,MR} - v_{i,MR}) \tag{3.11}$$

and

$$v_{i,MR}^2 = \sqrt{(\frac{\hat{v}_{MR}^2}{2})^2 + (\frac{T_{MR}}{2\rho\pi R_{MR}^2})^2} - \frac{\hat{v}_{MR}^2}{2}, \tag{3.12}$$

where

$$\hat{v}_{MR}^2 = u^2 + v^2 + w_{r,MR}(w_{r,MR} - 2v_{i,MR}), \tag{3.13}$$

$$w_{r,MR} = w + a_s u - b_s v, \tag{3.14}$$

$$w_{bl,MR} = w_{r,MR} + \frac{2}{3}\Omega_{MR} R_{MR} \theta_{col}, \tag{3.15}$$

$$\theta_{col} = K_{col}\delta_{col} + \theta_{col,0}, \tag{3.16}$$

and ρ is the air density, Ω_{MR} is the rotation speed of the main rotor, R_{MR} is the radius of the main rotor disc, $C_{l\alpha,MR}$ is the lift curve slope of the main rotor blade, b_{MR} is the blade number,

c_{MR} is the chord length of the main rotor blade, $\omega_{bl,MR}$ is the net vertical velocity relative to the main rotor blade, $\hat{v}^2_{MR}$ is an intermediate variable in the main rotor thrust calculation, $\omega_{r,MR}$ is the net vertical velocity through the main rotor disc, and θ_{col} is the collective pitch angle of the main rotor blade.

Tail rotor thrust

The tail rotor generates a thrust to counter the fuselage torque arising from the rotation of the main rotor. Similar to the main rotor, the tail rotor thrust T_{TR} can be expressed as

$$T_{TR} = \frac{\rho \Omega_{TR} R^2_{TR} C_{l\alpha,TR} b_{TR} c_{TR}}{4} (w_{bl,TR} - v_{i,TR}) \tag{3.17}$$

and

$$v^2_{i,TR} = \sqrt{(\frac{\hat{v}^2_{TR}}{2})^2 + (\frac{T_{TR}}{2\rho\pi R^2_{TR}})^2} - \frac{\hat{v}^2_{TR}}{2}, \tag{3.18}$$

where

$$\hat{v}^2_{TR} = (w + qD_{TR})^2 + u^2 + w_{r,TR}(w_{r,TR} - 2v_{i,TR}), \tag{3.19}$$

$$w_{r,TR} = v - rD_{TR} + pH_{TR}, \tag{3.20}$$

$$w_{bl,TR} = w_{r,TR} + \frac{2}{3}\Omega_{TR} R_{TR} \theta_{ped}, \tag{3.21}$$

$$\theta_{ped} = K_{ped}\bar{\delta}_{ped} + \theta_{ped,0}, \tag{3.22}$$

and Ω_{TR} is the rotation speed of the tail rotor, R_{TR} is the radius of the tail rotor disc, $C_{l\alpha,TR}$ is the lift curve slope of the tail rotor blade, b_{TR} is the tail rotor blade number, c_{TR} is the chord length of the tail rotor blade, $w_{bl,TR}$ is the net vertical velocity relative to the tail rotor disc, $\hat{v}^2_{TR}$ is an intermediate variable in the recursive calculation, D_{TR} is the tail rotor hub location behind the CG of the helicopter, $w_{r,TR}$ is the net vertical velocity through the tail rotor disc, H_{TR} is the tail rotor hub location above the CG of the helicopter, and θ_{ped} is the collective pitch angle of the tail rotor blade. $\bar{\delta}_{ped}$ is an intermediate state of the servo input of yaw channel due to the existence of the yaw rate feedback controller.

Flapping dynamic equations

Since the relative small size of the tail rotor, its flapping dynamic is neglected. The complete main rotor flapping dynamics is given by [12]

$$\dot{a}_s = -\frac{\tau_{MR} + K_{sb}\tau_{sb}}{\tau_{MR} + \tau_{sb}} q - \frac{1}{\tau_{MR} + \tau_{sb}} a_s + \frac{\tau_{MR} A_{b_s}}{\tau_{MR} + \tau_{sb}} b_s + \frac{A_{lon} + K_{sb} C_{lon}}{\tau_{MR} + \tau_{sb}} \delta_{lon} + A_{lat} \delta_{lat}, \quad (3.23)$$

$$\dot{b}_s = -\frac{\tau_{MR} + K_{sb}\tau_{sb}}{\tau_{MR} + \tau_{sb}} p + \frac{\tau_{MR} B_{a_s}}{\tau_{MR} + \tau_{sb}} a_s - \frac{1}{\tau_{MR} + \tau_{sb}} b_s + \frac{B_{lat} + K_{sb} D_{lat}}{\tau_{MR} + \tau_{sb}} \delta_{lat} + B_{lon} \delta_{lon}, \quad (3.24)$$

where τ_{MR} is the time constant of the main rotor flapping motion, K_{sb} is the ratio of main rotor blade cyclic pitch to stabilizer bar flapping, θ_{cyc,a_s} and θ_{cyc,b_s} are the longitudinal and lateral cyclic pitch of the main rotor blade, A_{lon} is the ratio of θ_{cyc,a_s} to δ_{lon}, A_{lat} is the ratio of θ_{cyc,a_s} to δ_{lat}, B_{lon} is the ratio of θ_{cyc,b_s} to δ_{lon}, B_{lat} is the ratio of θ_{cyc,b_s} to δ_{lat}, A_{b_s} and B_{a_s} are the coupling effect between longitudinal and lateral flapping motions.

3.4 Linear state-space model structure determination

In order to use modern advanced control techniques for the helicopter, a proper and accurate linear state-space model should be obtained. One of the most important tasks for obtaining the state-space model is to determine the model structure based on the above mentioned non-linear model. The model structure used in this thesis is adopted directly from [12] and [39], which can be derived through the analysis of the nonlinear model.

3.4.1 Lateral and longitudinal fuselage dynamic equations

From the rigid body dynamic equations derived in Equation 3.5 and Equation 3.6, we can get the four linear equations for the lateral and longitudinal linear and angular fuselage motions:

$$\begin{aligned} \Delta\dot{u} &= (-w_0 q + v_0 r) - g \cdot \Delta\theta + X_u \cdot \Delta u + X_a \cdot \Delta a_s \\ \Delta\dot{v} &= (-u_0 r + w_0 p) + g \cdot \Delta\phi + Y_v \cdot \Delta v + Y_b \cdot \Delta b_s \\ \Delta\dot{p} &= L_u \cdot \Delta u + L_v \cdot \Delta v + L_b \cdot \Delta b_s \\ \Delta\dot{q} &= M_u \cdot \Delta u + M_v \cdot \Delta v + M_a \cdot \Delta a_s \end{aligned} . \quad (3.25)$$

The external aerodynamic and gravitational forces and moments are formulated in terms of stability derivatives. For example, the rotor forces are expressed through the rotor derivatives X_a,

Y_b, and the rotor moments through the flapping spring-derivatives L_b, M_a. General aerodynamics effects are expressed by speed derivatives such as X_u, Y_v, L_u, L_v, M_u and M_v. The centrifugal terms in the linear motion equations, which are functions of the trim condition (u_0, v_0, w_0), are relevant only in cruise flight.

3.4.2 Rotor flapping dynamics

The linear flapping dynamic model is obtained through Equation 3.23 and Equation 3.24 as

$$\begin{aligned} \tau_f \cdot \Delta\dot{a}_s &= -\Delta a_s - \tau_f \cdot \Delta q + A_b \cdot \Delta b_s + A_{lat} \cdot \Delta\delta_{lat} + A_{lon} \cdot \Delta\delta_{lon} \\ \tau_f \cdot \Delta\dot{b}_s &= -\Delta b_s - \tau_f \cdot \Delta p + B_a \cdot \Delta a_s + B_{lat} \cdot \Delta\delta_{lat} + B_{lon} \cdot \Delta\delta_{lon} \end{aligned}, \tag{3.26}$$

where B_{lat}, B_{lon} and A_{lat}, A_{lon} are the input derivatives, τ_f is the main rotor time constant, which is a function of the main blade lock number and rotor speed. B_a and A_b account for the cross-coupling effects occurring at the level of the rotor itself.

3.4.3 Heave dynamics

The heave dynamics can be linearized through Equation 3.6 as

$$\Delta\dot{w} = (-v_0 p + u_0 q) + Z_w \cdot \Delta w + Z_{col} \cdot \Delta\delta_{col}, \tag{3.27}$$

where Z_w, Z_{col} are corresponding derivatives of w and heave input δ_{col}; (u_0, v_0) are the trim conditions.

3.4.4 Yaw dynamics

Due to the artificial yaw rate gyro controller of the helicopter, the yaw channel dynamics model is a bit more complicated. The final corresponding differential equations used in the state-space model is

$$\begin{aligned} \Delta\dot{r} &= N_r \Delta r + N_{ped,int} \Delta\delta_{ped,int} + N_{ped} \Delta\delta_{ped} \\ \Delta\dot{\delta}_{ped,int} &= K_{ped} \Delta\delta_{ped} - \Delta r \end{aligned} \tag{3.28}$$

3.4.5 Complete state-space model structure of the helicopter

Combining Equation 3.25, Equation 3.26, Equation 3.27 and Equation 3.28, we can get the complete state space model structure of the helicopter as

$$\dot{\mathbf{x}} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & L_b & 0 & L_u & L_v & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & M_a & 0 & 0 & M_u & M_v & 0 \\ 0 & 0 & 0 & 0 & 0 & N_r & 0 & 0 & N_{ped,int} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & -\frac{1}{\tau_f} & A_{bs} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & B_{as} & -\frac{1}{\tau_f} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -g & 0 & 0 & 0 & 0 & X_a & 0 & 0 & X_u & 0 & 0 \\ g & 0 & 0 & 0 & 0 & 0 & 0 & Y_b & 0 & 0 & Y_v & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & Z_w \end{bmatrix} \mathbf{x} + \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & N_{ped} & 0 \\ A_{lat} & A_{lon} & 0 & 0 \\ B_{lat} & B_{lon} & 0 & 0 \\ 0 & 0 & K_{ped} & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & Z_{col} \end{bmatrix} \mathbf{u}, \tag{3.29}$$

where $\mathbf{x} = [\phi \quad \theta \quad \psi \quad p \quad q \quad r \quad a_s \quad b_s \quad \delta_{ped,int} \quad u \quad v \quad w]^T$ and $\mathbf{u} = [\delta_{lat} \quad \delta_{lon} \quad \delta_{ped} \quad \delta_{col}]^T$.

3.5 Linear model identification

The state space model obtained in Equation 3.29 contains 23 unknown variables. In order to represent the accurate dynamic behaviors of the helicopter, the numerical values of these unknown variables need to be identified. The parameter identification procedures can be conducted in either time domain or frequency domain. In this thesis, we use the identification technique in frequency domain.

The steps involved in the identification process usually are [39]

1. *Collection of flight data.* The flight data is collected during special flight experiments using frequency sweeps.

2. *Frequency response calculation.* The frequency response for each input-output pair is computed using a Chirp-Z transform. At the same time, the coherence function for each frequency response is calculated.

3. *Multivariable frequency domain anaylysis.* The single input-output frequency responses

are conditioned by removing the effects from the secondary inputs. The partial coherence measures are computed.

4. *Window combination*. The accuracy of the low and high frequency ends of the frequency responses is improved through optimal combination of frequency response generated using different window lengths.

3.5.1 Flight data collections

High quality flight data is essential to a successful identification. The principal concerns are the accuracy of the state estimates (it is better to collect the data in no-wind environment conditions), the information content of the flight data, and the compatibility of the flight data with the postulate of linear dynamics used for the modeling.

The responses of the system to low frequency excitations are important for the identification of the speed derivatives (0.1 rad/s) and the responses to high frequency excitations are important for the identification of the coupled rotor/fuselage dynamics ($8-14$ rad/s). To guarantee that the flight data captures the dominant flight-dynamic effects, a frequency-sweep technique is used for the flight testing.

The sweep inputs to the helicopter should cover all the effective frequencies from low frequency to high frequency as much as possible. It is also better to reduce the combinations of the channel inputs as much as possible, i.e., when exciting the lateral channel, the other channels should have no (ideal) pilot inputs. Fig. 3.2 shows one set of the collected frequency sweep data used for the model identification. The interested readers are suggested to read [39] for detailed guidelines.

Furthermore, the inputs (i.e, δ_{lat}, δ_{lon}, δ_{col} and δ_{ped}) to the helicopter are given by the pilot in this thesis. It will usually lose information at certain interested frequencies since it requires good control skills of the pilot to perturb the transmitter inputs. This drawback from the pilot inputs will usually degrade the quality of the collected flight data. If the readers are interested in the modeling of the helicopter, we recommend and propose the readers a new data collection procedure to improve the data quality. The basic idea is to use computer to generate the perturbation inputs (the magnitudes should be small compared to normal pilot inputs) for the helicopter in order to cover interested frequencies as much as possible . The pilot's inputs are augmented to these computer generated perturbation inputs to safeguard and stabilize the

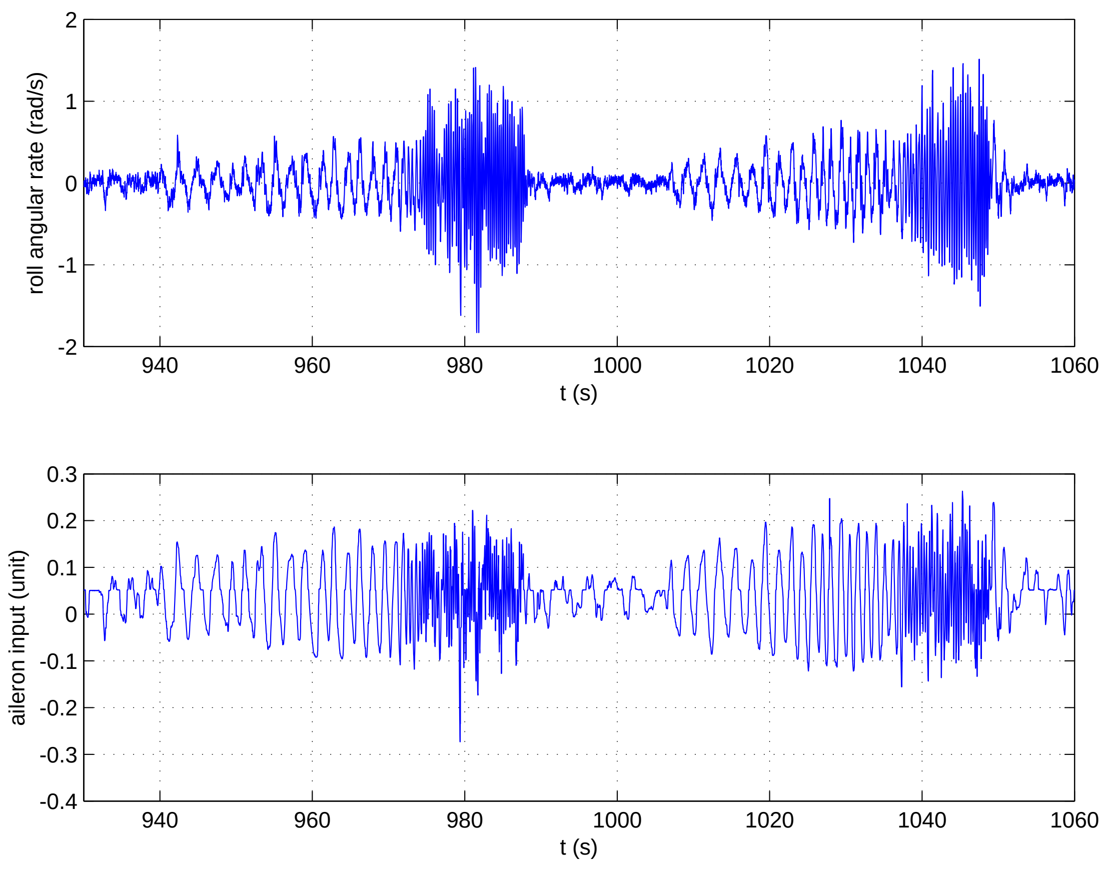


Figure 3.2: Data collected from frequency sweep technique

helicopter. In this way, the collected flight data should have high qualities which covers most of the interested frequencies.

3.5.2 Parameter identifications

In identifying the model parameters of NUS^{2}T-Lion, a MATLAB-based software, CIFER (Comprehensive Identification from FrEquency Response) was used to complete the frequency response calculation, multivariable frequency domain analysis and the window combination procedures. CIFER is developed by the NASA Ames Research Center for military-based rotorcraft systems. It searches for optimum model parameters by comparing the frequency domain responses from the proposed model to the actual flight data. The detailed procedures will not be repeated here. Interested readers are suggested to read the documentation of the CIFER software.

The most critical model-data fitting results showing the main channel responses are briefly shown here. In Fig. 3.3–3.5, the solid lines and the dashed lines show the frequency responses of the in-flight data and the fitted model respectively. The "coherence" shown in the three figures

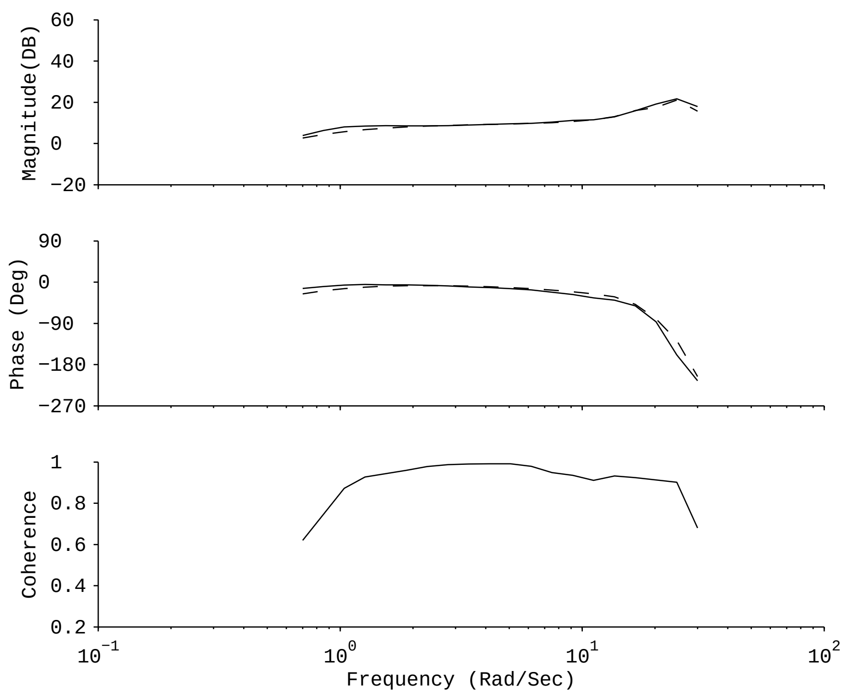


Figure 3.3: Frequency-domain model fitting: δ_{lat} to p

means the degree of matching between the fitted model and the real in-flight data. The matching is defined better for higher coherence values (1 is the highest and 0 is the lowest value). It can be seen that the model fits the flight test data very well, indicating a good fidelity of the derived parameters. After several iterations, the complete state-space model is identified as

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u}, \tag{3.30}$$

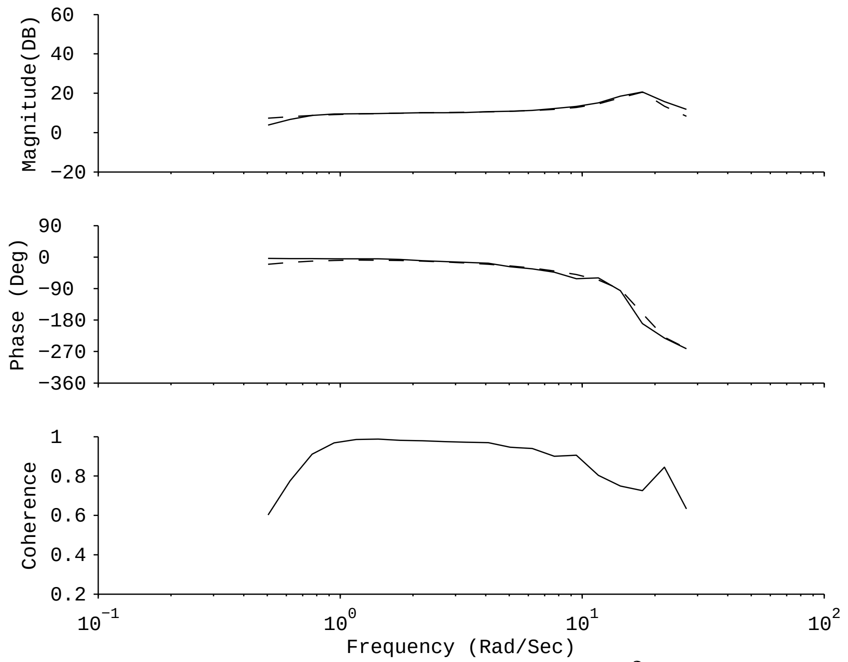


Figure 3.4: Frequency-domain model fitting: δ_{lon} to q

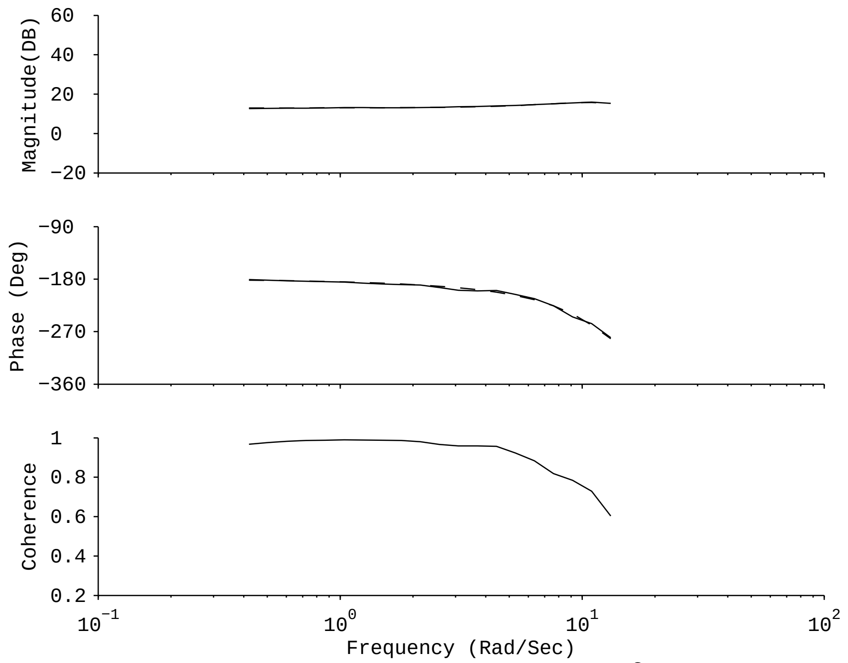


Figure 3.5: Frequency-domain model fitting: δ_{rud} to r

where

$$\mathbf{A} = \begin{bmatrix}
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 620.52 & 0 & -2.31 & 2.36 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 327.64 & 0 & 0 & 0 & -2.08 & 0 \\
0 & 0 & 0 & 0 & 0 & -13.48 & 0 & 0 & 165.64 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & -1 & 0 & -5.4048 & 6.4490 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & -1 & 0 & 0 & -3.7160 & -5.4048 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & -9.78 & 0 & 0 & 0 & 0 & -9.75 & 0 & 0 & -0.38 & 0 & 0 \\
9.78 & 0 & 0 & 0 & 0 & 0 & 0 & -61.72 & 0 & 0 & -0.86 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -0.57
\end{bmatrix}$$

$$\mathbf{B} = \begin{bmatrix}
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & -54.6861 & 0 \\
2.9753 & -0.3004 & 0 & 0 \\
0.7802 & 3.2295 & 0 & 0 \\
0 & 0 & -4.4634 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 19.5243
\end{bmatrix}$$

3.6 Linear model verification

A comprehensive evaluation on the quality of the obtained flight dynamic model is carried out. The basic idea is to collect another set of frequency sweep data, which will not used for the model identification, to validate the coherence of the model outputs and real flight data.

Fig. 3.6 shows the simulink block diagram used for obtaining the model outputs. The inputs to the simulink block diagram are the collected real flight input data to the helicopter from the pilot. The model (i.e., matrix A and B) are the obtained model in the above section. The outputs (i.e., States) are recorded and compared with the recorded real flight output data. The states chosen to be compared are the angular velocities, i.e., p, q, r. The reason is that the derivatives of euler angles is simplified to be the angular velocities. The euler angle outputs from the model is the integrations of angular rates. Small error accumulations in angular rates will excite the values of euler angles, we will not choose it. The derivatives of translational velocities are functions of euler angles, thus we will also not compare the outputs of translational velocities. Since the angular rates captures the most important characteristics of the helicopter dynamics and have less such problems, thus we choose to compare these variables between the model outputs and the real flight data.

The comparisons are plotted in Fig. 3.7. The flight data and the model outputs are well matched. Furthermore, it is almost matched between the flight data and the model outputs in the low-frequency regions. The simplifications of the coupling effects among different channels in the above linear state-space model lead the match not well in high frequency region. These model uncertainties can be compensated by design a proper feedback controller.

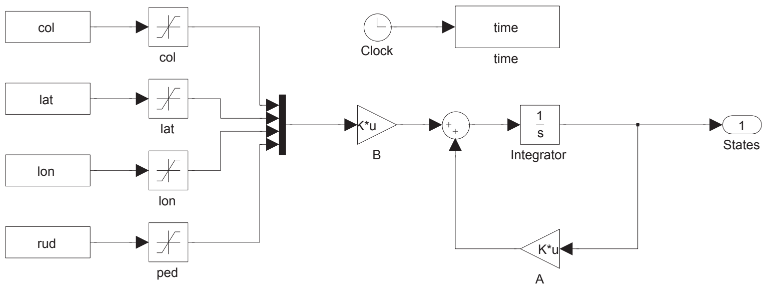


Figure 3.6: Linear model verification simulink block diagram

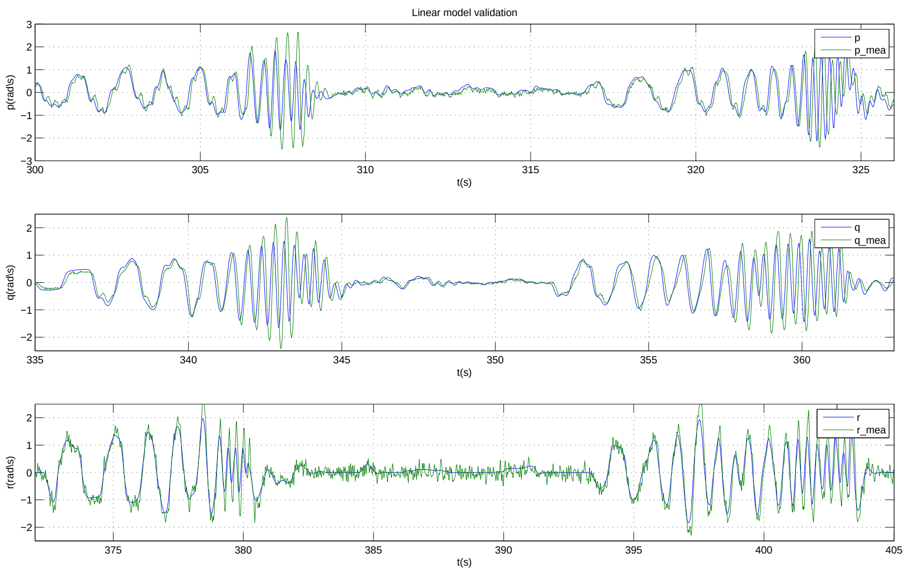


Figure 3.7: Linear model verification

3.7 Conclusion

In this section, the dynamic modeling of the helicopter is presented. A simplified nonlinear model of the helicopter is derived based on Newton-Euler formulation and the aerodynamics of the helicopter. In order to employ advanced modern control techniques to stabilize the helicopter, a linear state-space model structure is derived with 23 unknown variables. With high quality collected flight data based on frequency-sweep method, these variables are identified using frequency domain identification method. The identified linear model is further validated to be accurate for controller design.

Chapter 4

Controller Design

4.1 Introduction

For all UAV related applications, stability of the controlled platform is the most fundamental problem that needs to be solved first. Otherwise, there is no foundation for high-level navigation and guidance algorithms to be built upon. In this chapter, we decompose the UAV control problem into two layers, namely the attitude stabilization layer and the position tracking layer. The former involves the design of an inner-loop control law which makes sure the UAV roll, pitch and yaw dynamics are robustly stable. The latter position tracking layer involves the design of an outer-loop control law which enables the UAV to track any smooth 3D trajectory references in a responsive and precise way.

In this chapter, the background knowledge about robust H_∞ control technique and robust and perfect tracking (RPT) control technique will be briefly addressed first. The control structure formulation, inner-loop controller design based on H_∞ method, outer-loop controller design based on RPT method, and the inner-loop command generator which connects the two layers in a reasonable way will then be presented.

4.2 Background materials

4.2.1 H_∞ control technique

Given a continuous-time linear time-invariant system described as,

$$\Sigma : \begin{cases} \dot{x} = Ax + Bu + Ew \\ y = C_1 x + 0u + D_1 w \\ z = C_2 x + D_2 u + 0w \end{cases} \tag{4.1}$$

the standard H_∞ control problem is to find an internal stabilizing proper measurement feedback controller

$$\Sigma_c : \begin{cases} \dot{v} = A_{cmp} v + B_{cmp} y \\ u = C_{cmp} v + D_{cmp} y \end{cases} \tag{4.2}$$

such that the resulting closed-loop system is internally stable and the H_∞-norm of the overall closed-loop transfer matrix function from w to z, i.e., $T_{zw}(s)$, is minimized. The H_∞-norm of a stable continuous-time transfer matrix, e.g., $T_{zw}(s)$, is defined as

$$||T_{zw}||_\infty = \sup \bar{\sigma}[T_{zw}(j\omega)], \quad \forall \omega \in [0, \infty). \tag{4.3}$$

It is clear that the H_∞-norm of $T_{zw}(s)$ corresponds to the worst case gain from the input w to the output z. For future use, we define the infimum of H_∞ optimization, i.e., the infimum of the H_∞-norm of the closed-loop transfer matrix $T_{zw}(s)$ over all stabilizing proper controllers, as

$$\gamma_\infty^* = \inf\{||T_{zw}(\Sigma \times \Sigma_c)||_\infty\}, \forall \Sigma_c \text{ internally stabilizes } \Sigma. \tag{4.4}$$

The H_∞ control problem is said to be regular if the following conditions are satisfied,

1. D_2 is of maximal column rank, i.e., D_2 is a tall and full rank matrix;
2. The subsystem (A, B, C_2, D_2) has no invariant zeros on the imaginary axis;
3. D_1 is of maximal row rank, i.e., D_1 is a fat and full rank matrix;
4. The subsystem (A, E, C_1, D_1) has no invariant zeros on the imaginary axis;

It is said to be singular if it is not regular, i.e., at least one of the above 4 conditions is not satisfied.

H_∞ state feedback problems: the regular case

- Problem definition:

  The state feedback H_∞ control problems are referred to the problems in which all the states of the given plant Σ are available for feedback. That is the given system is

$$\Sigma : \begin{cases} \dot{x} = Ax + Bu + Ew \\ y = x \\ z = C_2 x + D_2 u \end{cases} \tag{4.5}$$

  where $(A,\, B)$ is stabilizable, D_2 is of maximal column rank and $(A,\, B,\, C_2,\, D_2)$ has no invariant zeros on the imaginary axis. In the state feedback case, we are looking for a static control law

$$u = Fx \tag{4.6}$$

  such that the H_∞-norm of the closed-loop system is minimized.

- Solution:

  Given $\gamma > \gamma_\infty^*$, solve the following algebraic Riccati equation (H_∞-ARE)

$$A^T P + PA + C_2^T C_2 + \frac{PEE^T P}{\gamma^2} - (PB + C_2^2 D_2)(D_2^T D_2)^{-1}(D_2^T C_2 + B^T P) = 0 \tag{4.7}$$

  for a unique positive semi-definite stabilizing solution $P \geq 0$. The H_∞ γ-suboptimal state feedback law is then given by

$$u = Fx = -(D_2^T D_2)^{-1}(D_2^T C_2 + B^T P)x \tag{4.8}$$

  The resulting closed-loop system $T_{zw}(s)$ has the following property: $\|T_{zw}\|_\infty < \gamma$.

H_∞ state feedback problems: the singular case

Consider the following system again,

$$\Sigma : \begin{cases} \dot{x} = Ax + Bu + Ew \\ y = x \\ z = C_2 x + D_2 u \end{cases} \tag{4.9}$$

where $(A,\, B)$ is stabilizable, D_2 is not necessarily of maximal rank and $(A,\, B,\, C_2,\, D_2)$ might have invariant zeros on the imaginary axis. Solution to this kind of problems can be done using the so-called perturbation approach. Define a new controlled output

$$\tilde{z} = \begin{bmatrix} z \\ \varepsilon x \\ \varepsilon u \end{bmatrix} = \begin{bmatrix} C_2 \\ \varepsilon I \\ 0 \end{bmatrix} x + \begin{bmatrix} D_2 \\ 0 \\ \varepsilon I \end{bmatrix} u \tag{4.10}$$

Clearly, $z \propto \tilde{z}$, if $\varepsilon = 0$. Now the perturbed system becomes

$$\tilde{\Sigma} : \begin{cases} \dot{x} = Ax + Bu + Ew \\ y = x \\ \tilde{z} = \tilde{C}_2 x + \tilde{D}_2 u \end{cases} . \tag{4.11}$$

Obviously, $\tilde{D}_2$ is of maximal column rank and $(A,\, B,\, \tilde{C}_2,\, \tilde{D}_2)$ is free of invariant zeros for any $\varepsilon > 0$. Thus, $\tilde{\Sigma}$ satisfies the conditions of the regular state feedback case, and hence we can apply the procedures for regular cases to the perturbed system to find the H_∞ control laws.

H_∞ output feedback problems: the regular case

Recall the system with measurement feedback as described in Equation 4.1, where $(A,\, B)$ is stabilizable and $(A,\, C_1)$ is detectable. Also, it satisfies the following regularity assumptions:

1. D_2 is of maximal column rank, i.e., D_2 is a tall and full rank matrix
2. The subsystem $(A,\, B,\, C_2,\, D_2)$ has no invariant zeros on the imaginary axis
3. D_1 is of maximal row rank, i.e., D_1 is a fat and full rank matrix
4. The subsystem $(A,\, E,\, C_1,\, D_1)$ has no invariant zeros on the imaginary axis.

Given a $\gamma > \gamma_\infty^*$, the optimal controller can be found by solving the following algebraic Riccati equation (H_∞-ARE)

$$A^T P + PA + C_2^T C_2 + \frac{PEE^T P}{\gamma^2} - (PB + C_2^2 D_2)(D_2^T D_2)^{-1}(D_2^T C_2 + B^T P) = 0 \tag{4.12}$$

for a positive semi-definite stabilizing solution $P \geq 0$, and the following ARE

$$QA^T + AQ + EE^T + \frac{QC_2^T C_2 Q}{\gamma^2} - (QC_1^T + ED_1^T)(D_1 D_1^T)^{-1}(D_1 E_T + C_1 Q) = 0 \tag{4.13}$$

for a positive semi-definite stabilizing solution $Q \geq 0$. In fact, these P and Q satisfy the so-called coupling condition: $\rho(PQ) < \gamma^2$. The H_∞ γ-suboptimal output feedback law is then given by

$$\Sigma_c : \begin{cases} \dot{v} = A_c v + B_c y \\ u = C_c v \end{cases} . \tag{4.14}$$

where $B_c = -(I - \gamma^{-2}QP)^{-1}K$, $C_c = F$, $A_c = A + \gamma^{-2}EE^T P + BF + (I - \gamma^{-2}QP)^{-1}K(C_1 + \gamma^{-2}D_1 E^T P)$ and where $F = -(D_2^T D_2)^{-1}(D_2^T C_2 + B^T P)$, $K = -(QC_1^T + ED_1^T)(D_1 D_1^T)^{-1}$.

The singular output feedback problems can be solved similar to the singular full state feedback problems by augmenting a small perturbation into E and D_1. The readers are suggested to read [14] for more details about H_∞ control.

4.2.2 Robust and perfect tracking (RPT) control technique

Consider the following continuous-time system:

$$\Sigma : \begin{cases} \dot{x} = Ax + Bu + Ew, x(0) = x_0 \\ y = C_1 x + D_1 w \\ z = C_2 x + D_2 u + D_{22} w \end{cases} . \tag{4.15}$$

where $x \in \mathbb{R}^n$ is the state, $u \in \mathbb{R}^m$ is the control input, $w \in \mathbb{R}^q$ is the external disturbance, $y \in \mathbb{R}^q$ is the measurement output, and $z \in \mathbb{R}^l$ is the output to be controlled. Given the external disturbance $w \in L_p$, $p \in [1, \infty)$, and any reference signal vector $r \in R^l$ with r, $\dot{r}$, ..., $r^{(k-1)}$, $k \geq 1$, being available, and $r^{(k)}$ being either a vector of delta functions or in L_p, the RPT problem for the system in 4.15 is to find a parameterized dynamic measurement control law of the following form:

$$\begin{cases} \dot{v} = A_{cmp}(\varepsilon)v + B_{cmp}y + G_0(\varepsilon)r + ... + G_{k-1}(\varepsilon)r^{(k-1)} \\ u = C_{cmp}v + D_{cmp}y + H_0(\varepsilon)r + ... + H_{k-1}(\varepsilon)r^{(k-1)} \end{cases} \tag{4.16}$$

such that when the controller 4.16 is applied to the system of 4.15, we have the following

1. There exists an $\varepsilon^* > 0$ such that the resulting closed-loop system with $r = 0$ and $w = 0$ is

asymptotically stable for all $\varepsilon \in (0, \varepsilon^*]$.

2. Let $z(t,\varepsilon)$ be the closed-loop controlled output response and let $e(t,\varepsilon)$ be the resulting tracking error, i.e., $e(t,\varepsilon) = z(t,\varepsilon) - r(t)$. Then, for any initial condition of the state, $x_0 \in R^n$,
$$\|e\|_p = \left(\int_0^\infty |e(t)|^p dt\right)^{\frac{1}{p}} \to 0, \quad \text{as} \quad \varepsilon \to 0. \tag{4.17}$$

We introduce in the above formulation some additional information besides the reference signal r, i.e., $\dot{r}$, $\ddot{r}$, ..., $r^{(k-1)}$, as additional controlled inputs. Note that, in general, these additional signals can easily be generated without any extra costs. For example, if $r(t) = t^2$, then one can easily obtain its first-order derivatives $\dot{r}(t) = 2t$ and its second-order derivative $\ddot{r} = 2$. In flight control systems, taking r as a position reference, generally, its associated velocity, $\dot{r}$, and acceleration $\ddot{r}$, are readily available. These $\dot{r}$ and $\ddot{r}$ can be used to improve the overall tracking performance. We also note that the above formulation covers all possible reference signals that have the form $r(t) = t^k$, $0 \le k < \infty$. Thus, it can be applied to track approximately those reference signals that have a Taylor series expansion at $t = 0$. Yhis can be done by truncating the higher-order terms of the Taylor series of the given signal.

It is shown that the RPT problem for the system in 4.15 is solvable if and only if the following conditions hold:

1. (A, B) is stabilizable and (A, C_1) is detectable.

2. $D_{22} + D_2 S D_1 = 0$, where $S = -(D_2^T D_2)^\dagger D_2^T D_{22} D_1^T (D_1 D_1^T)^\dagger$.

3. (A, B, C_2, D_2) is right invertible and minimum phase.

4. $Ker(C_2 + D_2 S C_1) \supset C_1^{-1} Im(D_1)$.

Here, we note that $X^\dagger$ denotes the Moore-Penrose (pseudo) inverse of a constant matrix X, Im(X) and Ker(X) are respectively the range and null spaces of X, and lastly, $C^{-1}\chi = x | Cx \in \chi$, where χ is a subspace and C is a constant matrix. We also note that for the case when $D_1 = 0$, then the above solvability conditions can be simplified as follows:

1. (A, B) is stabilizable and (A, C_1) is detectable.

2. $D_{22} = 0$.

3. (A, B, C_2, D_2) is right invertible and of minimum phase.

4. $Ker(C_2) \supset Ker(C_1)$.

The last condition is automatically satisfied if the controlled output z of the given system is part of its measurement output y.

We assume throughout the rest of this section that the above conditions are satisfied , and we move on to construct solutions to the RPT problem. Since the outer-loop controller of the helicopter is to be designed with full state feedback, we focus on the state feedback RPT problem in the remaining section.

When all states of the plant are measured for feedback, the problem can be solved by a static control law. We construct a parameterized state feedback control law,

$$u = F(\varepsilon)x + H_0(\varepsilon)r + ... + H_{k-1}(\varepsilon)r^{(k-1)}, \tag{4.18}$$

that solves the RPT problem for the system in 4.15. It is simple to note that we can rewrite the given reference in the following form:

$$\frac{d}{dt}\begin{bmatrix} r \\ \vdots \\ r^{(k-2)} \\ r^{(k-1)} \end{bmatrix} = \begin{bmatrix} 0 & I_l & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & I_l \\ 0 & 0 & \cdots & 0 \end{bmatrix} \begin{bmatrix} r \\ \vdots \\ r^{(k-2)} \\ r^{(k-1)} \end{bmatrix} + \begin{bmatrix} 0 \\ \vdots \\ 0 \\ I_l \end{bmatrix} r^{(k)}. \tag{4.19}$$

Combining 4.19 with the given system, we obtain the following augmented system:

$$\Sigma_{AUG} : \begin{cases} \dot{x} = Ax + Bu + Ew, x(0) = x_0 \\ y = C_1 x \\ e = C_2 x + D_2 u \end{cases} . \tag{4.20}$$

where

$$x = \begin{bmatrix} r \\ \vdots \\ r^{(k-2)} \\ r^{(k-1)} \\ x \end{bmatrix}, \quad w = \begin{bmatrix} w \\ r^{(k)} \end{bmatrix}, \tag{4.21}$$

$$A = \begin{bmatrix} 0 & I_l & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & \cdots & I_l & 0 \\ 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & \cdots & 0 & A \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 0 \\ B \end{bmatrix}, \quad E = \begin{bmatrix} 0 & 0 \\ \vdots & \vdots \\ 0 & 0 \\ 0 & I_l \\ E & 0 \end{bmatrix} \tag{4.22}$$

and

$$C_2 = \begin{bmatrix} -I_l & 0 & 0 & \cdots & 0 & C_2 \end{bmatrix}, \quad D_2 = D_2. \tag{4.23}$$

It is then straightforward to show that the subsystem from u to e in the augmented system of 4.20, i.e., the quadruple (A,B,C_2,D_2), is right invertible and has the same infinite zeros structures as that of the original (A,B,C_2,D_2). Furthermore, its invariant zeros contain those of (A,B,C_2,D_2) and $l \times k$ extra ones at $s = 0$.

Next, we define

$$\tilde{C}_2 = \begin{bmatrix} C_2 \\ \varepsilon I_{kl+n} \\ 0 \end{bmatrix}, \quad \tilde{D}_2 = \begin{bmatrix} D_2 \\ 0 \\ \varepsilon I_m \end{bmatrix}, \tag{4.24}$$

$$\tilde{A} = \begin{bmatrix} \tilde{A}_0 & 0 \\ 0 & A \end{bmatrix}, \quad \tilde{A}_0 = -\varepsilon_0 I_{kl} + \begin{bmatrix} 0 & I_l & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & I_l \\ 0 & 0 & \cdots & 0 \end{bmatrix} \tag{4.25}$$

where ε_0 is a sufficiently small scalar, and solve the following Riccati equation:

$$P\tilde{A} + \tilde{A}^T P + \tilde{C}_2^T \tilde{C}_2 - (PB + \tilde{C}_2^T \tilde{D}_2)^{-1}(PB + \tilde{C}_2^T \tilde{D}_2)^T = 0 \tag{4.26}$$

for a positive-definite solution $P > 0$. The required state feedback gain matrix that solves the RPT problem for the given system is then given by

$$\tilde{F}(\varepsilon) = -(\tilde{D}_2^T \tilde{D}_2)^{-1}(PB + \tilde{C}_2^T \tilde{D}_2)^T = [H_0(\varepsilon) \cdots H_{k-1}(\varepsilon) F(\varepsilon)], \tag{4.27}$$

where $H_i(\varepsilon) \in \mathbb{R}^{m \times l}$ and $F(\varepsilon) \in \mathbb{R}^{m \times n}$.

Finally, we note that solutions to the Riccati equation might have severe numerical problems as ε becomes smaller and smaller. Alternatively, one can solve the RPT control problem using

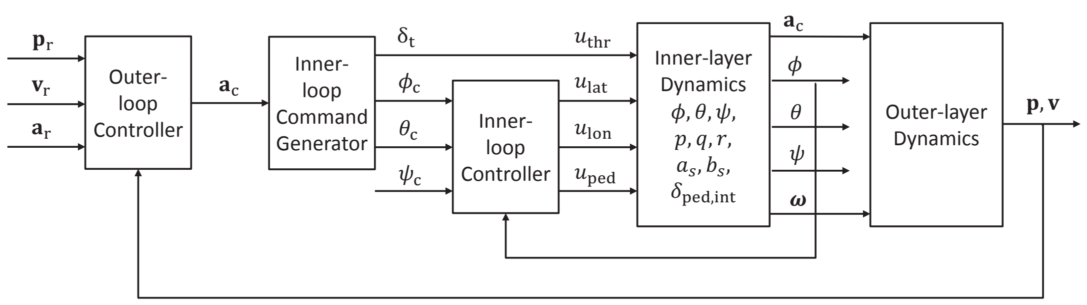


Figure 4.1: Control structure of NUS^{2}T-Lion

a structural decomposition approach, which can be found in [14].

4.3 Control structure

In control engineering, the divide-and-conquer strategy is usually used when a relatively complex system needs to be handled. In flight control engineering, a natural decomposition of the full-order dynamic model of a helicopter is based on motion types, i.e. rotational motion and translational motion. In general, the dynamics of rotational motion is much faster than that of the translational motion, which makes them severable in the frequency domain. Hence, the overall control system can be formulated in a dual-loop structure, so that the inner-loop and outer-loop controllers can be designed separately. Moreover, the linearized model of the single rotor helicopter system is found to be of non-minimum phase if the two motion dynamics are combined together. This non-minimum phase characteristics will highly complicate the control problem and needs to be avoided.

For the inner loop, the controlled object covers the rotational motion of the helicopter body, the flapping motion of rotor blades and the stabilizer bar, as well as the dynamics embedded within the head-lock gyro. The main task of the inner-loop controller is to stabilize the attitude and heading of the helicopter in all flight conditions. In our implementation, the H_∞ control method is used to minimize the disturbance from wind gusts. For the outer loop, the controlled object covers only the translational motion. The main task is to steer the helicopter flying with reference to a series of given locations. A robust and perfect tracking (RPT) approach is implemented to emphasize the time factor. Fig. 4.1 gives an overview of the dual-loop control structure.

4.4 Inner-loop control design

Although the full model of NUS^{2}T-Lion is highly complicated and nonlinear, it is verified by simulation that its inner dynamics, after linearization, is more or less invariant under different non-acrobatic flight conditions. Hence, it is reasonable to design a feedback control law based on the linearized model of the inner-layer dynamics, while using the nonlinear model for verification purposes only. Besides, it is noted that NUS^{2}T-Lion falls into the category of small-scale UAV helicopter which is quite vulnerable to environmental disturbances such as wind gusts. Hence, the H_∞ control method, which is specifically developed to minimize output error caused by external disturbances, naturally becomes the best choice. The linearized inner-dynamics model of NUS^{2}T-Lion can be represented by a 9th order state space form as shown below:

$$\begin{cases} \dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u} + \mathbf{E}\mathbf{w} \\ \mathbf{y} = \mathbf{C}_1\mathbf{x} + \mathbf{D}_1\mathbf{w} \\ \mathbf{h} = \mathbf{C}_2\mathbf{x} + \mathbf{D}_2\mathbf{u} \end{cases}, \tag{4.28}$$

where $\mathbf{x}$ is the state, $\mathbf{y}$ is the measurement output, $\mathbf{h}$ is the controlled output, $\mathbf{u}$ is the input, and $\mathbf{w}$ is the wind disturbance. More specifically,

$$\mathbf{x} = \begin{bmatrix} \phi & \theta & \psi & p & q & r & a_s & b_s & \delta_{\text{ped,int}} \end{bmatrix}^{\text{T}}, \tag{4.29}$$

$$\mathbf{u} = \begin{bmatrix} \delta_{\text{lat}} & \delta_{\text{lon}} & \delta_{\text{ped}} \end{bmatrix}^{\text{T}}, \tag{4.30}$$

$$\mathbf{w} = \begin{bmatrix} u_{\text{wind}} & v_{\text{wind}} & w_{\text{wind}} \end{bmatrix}^{\text{T}}, \tag{4.31}$$

where

$$\mathbf{A} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 620.52 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 327.64 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -13.48 & 0 & 0 & 165.64 \\ 0 & 0 & 0 & 0 & -1 & 0 & -5.4048 & 6.4490 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & -3.7160 & -5.4048 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \end{bmatrix} \tag{4.32}$$

$$\mathbf{B} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & -54.6861 \\ 2.9753 & -0.3004 & 0 \\ 0.7802 & 3.2295 & 0 \\ 0 & 0 & -4.4634 \end{bmatrix} \tag{4.33}$$

The disturbance matrix E can be obtained by linearizing the whole flight dynamics of the unmanned system with an injection of V_{wind} as a disturbance input to its respective channels. The exact values of the matrix are referred from [12] and it is described as

$$\mathbf{E} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ -0.0001 & 0.1756 & -0.0395 \\ 0 & 0.0003 & 0.0338 \\ -0.0002 & -0.3396 & 0.6424 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}. \tag{4.34}$$

As the onboard IMU can provide measurements of the first six state variables, C_1 can be formed accordingly and and D_1 can be left as a zero matrix.

$$\mathbf{C}_1 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}, \quad \mathbf{D}_1 = \begin{bmatrix} 0_{6\times3} \end{bmatrix} \tag{4.35}$$

C_2 and D_2 constitute weighting parameters specifying the control objective, and usually need to be tuned for practical implementation. For this case, they are in the following form, which considers the first six state variables and the three control inputs.

$$\mathbf{C}_2 = \left[\begin{array}{ccccccccc} c_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & c_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & c_3 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & c_4 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & c_5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & c_6 & 0 & 0 & 0 \\ \hline \multicolumn{9}{c}{\mathbf{0}_{3\times 9}} \end{array}\right], \quad \mathbf{D}_2 = \left[\begin{array}{ccc} \multicolumn{3}{c}{\mathbf{0}_{6\times 3}} \\ \hline d_1 & 0 & 0 \\ 0 & d_2 & 0 \\ 0 & 0 & d_3 \end{array}\right]. \tag{4.36}$$

The H_∞ control problem is to find an internally stabilizing proper measurement feedback control law,

$$\begin{cases} \dot{\mathbf{v}} = \mathbf{A}_{\text{cmp}}\mathbf{v} + \mathbf{B}_{\text{cmp}}\mathbf{y} \\ \mathbf{u} = \mathbf{C}_{\text{cmp}}\mathbf{v} + \mathbf{D}_{\text{cmp}}\mathbf{y} \end{cases}, \tag{4.37}$$

such that the H_∞-norm of the overall closed-loop transfer matrix function from $\mathbf{w}$ to $\mathbf{h}$ is minimized. According to [14], the minimum H_∞-norm, γ^*, can be exactly computed using some numerical algorithms. However, it is almost impossible to find a control law with finite gain to achieve this particular optimal performance. Usually, an H_∞ suboptimal controller is designed, resulting in a suboptimal H_∞-norm smaller than γ, where $\gamma > \gamma^*$. It is also proved in [14] that when the subsystem $(\mathbf{A}, \mathbf{E}, \mathbf{C}_1, \mathbf{D}_1)$ is left invertible and of minimum phase, which is exactly the case for NUS^{2}T-Lion, the optimal achievable H_∞ control performance under the state feedback and the measurement feedback are identical. In other words, it is appropriate to design the state feedback control law and the observer separately for the inner loop of NUS^{2}T-Lion. Moreover, only a reduced-order observer is needed to estimate the three unmeasurable state variables, i.e., a_s, b_s, $\delta_{\text{ped,int}}$.

To design an H_∞ reduced-order output feedback control law for the inner-dynamics of

NUS^{2}T-Lion, the original system (4.28) can be rewritten as follows:

$$\begin{cases} \begin{pmatrix} \dot{\mathbf{x}}_1 \\ \dot{\mathbf{x}}_2 \end{pmatrix} = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{pmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{pmatrix} + \begin{bmatrix} B_1 \\ B_2 \end{bmatrix} \mathbf{u} + \begin{bmatrix} E_1 \\ E_2 \end{bmatrix} \mathbf{w} \\ \begin{pmatrix} \mathbf{y}_0 \\ \mathbf{y}_1 \end{pmatrix} = \begin{bmatrix} 0 & C_{1,02} \\ I & 0 \end{bmatrix} \begin{pmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{pmatrix} + \begin{bmatrix} D_{1,0} \\ 0 \end{bmatrix} \mathbf{w} \\ \mathbf{h} = \begin{bmatrix} C_{2,1} & C_{2,2} \end{bmatrix} \begin{pmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{pmatrix} + D_2 \mathbf{u} \end{cases} . \tag{4.38}$$

In this form, the original state $\mathbf{x}$ is partitioned into a measurable state $\mathbf{x}_1$ and an unmeasurable state $\mathbf{x}_2$; $\mathbf{y}$ is partitioned into $\mathbf{y}_0$ and $\mathbf{y}_1$ with $\mathbf{y}_1 \equiv \mathbf{x}_1$. If we define an auxiliary subsystem characterized by a matrix quadruple (A_R, E_R, C_R, D_R), where

$$(A_R, E_R, C_R, D_R) = \left(A_{22}, E_2, \begin{bmatrix} C_{1,02} \\ A_{12} \end{bmatrix}, \begin{bmatrix} D_{1,0} \\ E_1 \end{bmatrix} \right), \tag{4.39}$$

then the following procedures can be followed to design the reduced-order output feedback H_∞ controller:

Step 1: Construction of state feedback gain matrix

Define an auxiliary system

$$\begin{cases} \dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u} + \mathbf{E}\mathbf{w} \\ \mathbf{y} = \mathbf{x} \\ \mathbf{h} = \mathbf{C}_2\mathbf{x} + \mathbf{D}_2\mathbf{u} \end{cases} . \tag{4.40}$$

Select $\gamma > \gamma^*$, compute its corresponding H_∞ γ-suboptimal state feedback gain matrix F.

Step 2: Construction of observer gain matrix

Define another auxiliary system

$$\begin{cases} \dot{\mathbf{x}} = \mathbf{A}_\mathrm{R}^\mathrm{T}\mathbf{x} + \mathbf{C}_\mathrm{R}^\mathrm{T}\mathbf{u} + \mathbf{C}_{2,2}^\mathrm{T}\mathbf{w} \\ \mathbf{y} = \mathbf{x} \\ \mathbf{h} = \mathbf{E}_\mathrm{R}^\mathrm{T}\mathbf{x} + \mathbf{D}_\mathrm{R}^\mathrm{T}\mathbf{u} \end{cases} . \tag{4.41}$$

Select a sufficiently small $\gamma > 0$, compute its corresponding H_∞ γ-suboptimal state feedback gain matrix F_R and then let $K_R = F_R^{\mathrm{T}}$.

Step 3: Construction of output feedback controller

Partition F and K_R as

$$\mathbf{F} = [F_1 \;\; F_2], \qquad \mathbf{K}_R = [K_{\mathrm{R0}} \;\; K_{\mathrm{R1}}], \tag{4.42}$$

in conformity with the partitioning of **x** and **y** respectively. Now define

$$\mathbf{G}_{\mathrm{R}} = [-K_{\mathrm{R0}}, \; A_{21} + K_{\mathrm{R1}}A_{11} - (A_{\mathrm{R}} + K_{\mathrm{R}}C_{\mathrm{R}})K_{\mathrm{R1}}],$$

then the reduced-order output feedback controller is given by (4.37), where

$$\begin{aligned}
\mathbf{A}_{\mathrm{cmp}} &= A_{\mathrm{R}} + B_2F_2 + K_{\mathrm{R}}C_{\mathrm{R}} + K_{\mathrm{R1}}B_1F_2,\\
\mathbf{B}_{\mathrm{cmp}} &= G_{\mathrm{R}} + (B_2 + K_{\mathrm{R1}}B_1)[0, \; F_1 - F_2K_{\mathrm{R1}}],\\
\mathbf{C}_{\mathrm{cmp}} &= F_2,\\
\mathbf{D}_{\mathrm{cmp}} &= [0, \; F_1 - F_2K_{\mathrm{R1}}].
\end{aligned}$$

Based on the above procedures while choosing an appropriate set of C_2, D_2, a H_∞ reduced-order output feedback controller can be determined. After several rounds of tunings, the final values for the weighting parameters in C_2 and D_2 are chosen to be

$$c_1 = 13, \quad c_2 = 12, \quad c_3 = 1, \quad c_4 = 1, \quad c_5 = 1, \quad c_6 = 6, \tag{4.43}$$

and

$$d_1 = 13, \quad d_2 = 12, \quad d_3 = 30. \tag{4.44}$$

The corresponding $\gamma* = 0.2057$ and we choose $\gamma = 0.21$, which results in the following γ-suboptimal state feedback gain matrix,

$$\mathbf{F} = \begin{bmatrix} -0.9952 & -0.1177 & 0.0017 & -0.0271 & 0.0098 & 0.0143 & -1.8795 & -0.5324 & 0.0457 \\ 0.1386 & -0.9927 & -0.0005 & -0.0056 & -0.0467 & -0.0043 & 0.0253 & -1.8175 & -0.0503 \\ -0.0186 & 0.0096 & 0.0526 & -0.0006 & 0.0026 & 0.2379 & -0.0925 & -0.0216 & 1.3287 \end{bmatrix} \tag{4.45}$$

The corresponding feed-forward matrix is calculated as

$$\mathbf{G} = \begin{bmatrix} 0.9952 & 0.1177 & -0.0017 \\ -0.1386 & 0.9927 & 0.0005 \\ 0.0186 & -0.0096 & -0.0526 \end{bmatrix}.$$

The last three unmeasurable state variables, denoted by $\hat{\mathbf{x}}$, can be estimated by an observer as follows:

$$\dot{\hat{\mathbf{x}}} = \bar{\mathbf{F}}\hat{\mathbf{x}} + \bar{\mathbf{G}}\mathbf{y} + \bar{\mathbf{H}}\mathbf{u} \tag{4.46}$$

where

$$\bar{\mathbf{F}} = \begin{bmatrix} -0.9952 & -0.1177 & 0 \\ 0.1386 & -0.9927 & 0 \\ -0.0186 & 0 & -28 \end{bmatrix}, \tag{4.47}$$

$$\bar{\mathbf{G}} = \begin{bmatrix} 0 & 0 & 0 & -9.309 & 0.24040 \\ 0 & 0 & 0 & -1.225 & -5.1060 \\ 0 & 0 & 0 & 0 & 0-3.452 \end{bmatrix}, \tag{4.48}$$

$$\bar{\mathbf{H}} = \begin{bmatrix} 2.975 & -0.3 & 0 \\ 0.780 & 3.23 & 0 \\ 0 & 0 & 4.78 \end{bmatrix}. \tag{4.49}$$

With this set of gain matrices, the inner-loop system is stable with a bandwidth of 2.95 rad/s for the roll angle dynamics, 2.8 rad/s for the pitch angle dynamics and 2.17 rad/s for the yaw angle dynamics.

4.5 Outer-loop control design

For the outer-loop, an RPT controller is designed to let the UAV track any 3D trajectories precisely. The controller structure and design techniques are adopted from [16]. By perfect tracking, it means the ability of the controlled system to track a given reference with arbitrarily fast settling time subjected to disturbances and initial conditions. Considering the following linear time invariant system

$$\Sigma : \begin{cases} \dot{\mathbf{x}} = A\mathbf{x} + B\mathbf{u} + E\mathbf{w} \\ \mathbf{y} = C_1\mathbf{x} + D_1\mathbf{w} \\ \mathbf{h} = C_2\mathbf{x} + D_2\mathbf{u} + D_{22}\mathbf{w} \end{cases}, \tag{4.50}$$

with $\mathbf{x}, \mathbf{u}, \mathbf{w}, \mathbf{y}, \mathbf{h}$ being the state, control input, disturbance, measurement and controlled output respectively, the task of an RPT controller is to formulate a dynamic measurement control law in the form of

$$\dot{\mathbf{v}} = A_c(\varepsilon)\mathbf{v} + B_c(\varepsilon)\mathbf{y} + G_0(\varepsilon)r + ... + G_{\kappa-1}(\varepsilon)r^{\kappa-1},$$
$$\mathbf{u} = C_c(\varepsilon)\mathbf{v} + D_c(\varepsilon)\mathbf{y} + H_0(\varepsilon)r + ... + H_{\kappa-1}(\varepsilon)r^{\kappa-1},$$

so that when a proper $\varepsilon^* > 0$ is chosen,

- The resulted closed-loop system is asymptotically stable subjected to zero reference.

- If $e(t, \varepsilon)$ is the tracking error, then for any initial condition $\mathbf{x}_0$, there exists:

$$\|e\|_p = (\textstyle\int_0^\infty |e(t)^p|dt)^{1/p} \to 0, \quad as \quad \varepsilon \to 0.$$

For non-zero references, their derivatives are used to generate additional control inputs. Thus, any reference of the form of $r(t) = p_1t^k + p_2t^{k-1} + ... + p_{k+1}$ are covered in the RPT formulation. Furthermore, any references that have a Taylor series expansion at $t = 0$ can also be tracked using the RPT controller.

Similar to the case introduced in [38], the outer dynamics of NUS^{2}T-Lion is differentially flat. That means all its state variables and inputs can be expressed in terms of algebraic functions of flat outputs and their derivatives. A proper choice of flat outputs could be

$$\sigma = \begin{bmatrix} x & y & z & \psi \end{bmatrix}^{\mathrm{T}}. \tag{4.51}$$

It can also be observed that the first three outputs, x, y, z, are totally independent. In other words, we can consider the UAV as a mass point with constrained velocity, acceleration, jerk, and so on. in the individual axes of the 3D global frame when designing its outer-loop control law and generating the position references. Hence, a stand-alone RPT controller based on a double integrator model in each axis can be designed to track the corresponding reference in that particular axis. For each axis, the nominal system can be written as

$$\begin{cases} \dot{\mathbf{x}}_\mathrm{n} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \mathbf{x}_\mathrm{n} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \mathbf{u}_\mathrm{n} \\ \mathbf{y}_\mathrm{n} = \mathbf{x}_\mathrm{n} \end{cases} . \tag{4.52}$$

To achieve better tracking performance, it is common to include an integrator to ensure zero steady state error subjected to step inputs. Thus, the RPT controller proposed here is with integral action. This requires an augmented system to be formulated as

$$\begin{cases} \dot{\mathbf{x}}_\mathrm{o} = \begin{bmatrix} 0 & -1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \mathbf{x}_\mathrm{o} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \mathbf{u}_\mathrm{o} \\ \mathbf{y}_\mathrm{o} = \mathbf{x}_\mathrm{o} \\ \mathbf{h}_\mathrm{o} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \mathbf{x}_\mathrm{o} \end{cases} , \tag{4.53}$$

where $\mathbf{x}_\mathrm{o} = \begin{bmatrix} \int (p_e) & p_r & v_r & a_r & p & v \end{bmatrix}^\mathrm{T}$ with p_r, v_r, a_r as the position, velocity and acceleration references, p, v as the actual position and velocity and $p_e = r_p - p$ as the tracking error of position. By following the procedures in [14], a linear feedback control law of the form below can be acquired,

$$u_\mathrm{o} = F_\mathrm{o} \mathbf{x}_\mathrm{o}, \tag{4.54}$$

where

$$F_\mathrm{o} = \Big[\begin{matrix} \frac{k_\mathrm{i}\omega_n^2}{\varepsilon^3} & \frac{\omega_n^2 + 2\zeta\omega_n k_\mathrm{i}}{\varepsilon^2} & \frac{2\zeta\omega_n + k_\mathrm{i}}{\varepsilon} \\ 1 & -\frac{\omega_n^2 + 2\zeta\omega_n k_\mathrm{i}}{\varepsilon^2} & -\frac{2\zeta\omega_n + k_\mathrm{i}}{\varepsilon} \end{matrix} \Big] . \tag{4.55}$$

ε is a design parameter to adjust the settling time of the closed-loop system. $\omega_n, \zeta, k_\mathrm{i}$ are the parameters that determines the desired pole locations of the infinite zero structure of (4.53) through

$$p_\mathrm{i}(s) = (s + k_\mathrm{i})(s^2 + 2\zeta\omega_n s + \omega_n^2). \tag{4.56}$$

Theoretically, when the design parameter ε is small enough, the RPT controller gives arbitrarily fast response. However, in real life, due to the constraints of the UAV physical dynamics and its inner-loop bandwidth it is safer to limit the bandwidth of the outer loop to be one fifth to one third of the controlled inner-loop system. For the case of NUS^{2}T-Lion case, the following design parameters are used for different axes:

$$x, y : \begin{cases} \varepsilon &= 1 \\ \omega_n &= 0.707 \\ \zeta &= 0.707 \\ k_\mathrm{i} &= 0.25 \end{cases} \qquad z : \begin{cases} \varepsilon &= 1 \\ \omega_n &= 0.99 \\ \zeta &= 0.707 \\ k_\mathrm{i} &= 0.29 \end{cases}$$

4.6 Inner-loop command generator

We have designed the inner-loop and the outer-loop controllers separately to avoid the non-minimum phase problem and to relieve task complexity. As the inner-loop dynamics is designed much faster than that of the outer loop, it can be treated as a non-dynamic static gain matrix when viewed from outside. However, the output from the outer-loop controller in physical meaning is the desired accelerations in the global frame, $\mathbf{a}_\mathrm{n,c}$, while the inner-loop controller is looking for attitude references (ϕ_c, θ_c, ψ_c). Obviously, a global-to-body rotation followed by a command conversion is needed. Moreover, the body-axis acceleration $\mathbf{a}_\mathrm{b,c}$ does not mean anything to the heading direction reference ψ_c. Therefore, unlike the other two attitude angle references (ϕ_c, θ_c), ψ_c is not involved in this conversion, but generated independently. In addition, the acceleration reference in the UAV body z-axis directly links to the needed collective control input δ_col, which is not manipulated by the inner-loop at all. Based on the above ideas, if $\mathbf{G}_\mathrm{a}$ is the steady-state gain matrix from the inputs (δ_col, ϕ_c, θ_c) to the UAV body-frame accelerations, then we can get an approximated conversion matrix G_c as its inverse. So,

$$\begin{pmatrix} \delta_{\text{col}} & \phi_{\text{c}} & \theta_{\text{c}} \end{pmatrix}^{\text{T}} = \mathbf{G}_{\text{c}} \mathbf{a}_{\text{b,c}} = \mathbf{G}_{\text{a}}^{-1} \mathbf{a}_{\text{b,c}}. \tag{4.57}$$

Note that $\mathbf{G}_{\text{a}}$ must be non-singular. Otherwise, it means $\mathbf{a}_{\text{b}}$ cannot be manipulated by the control inputs u_{col}, u_{lat}, u_{lon}. For the case of NUS^{2}T-Lion,

$$\mathbf{G}_{\text{c}} = \begin{bmatrix} 0 & 0 & 0.0523 \\ 0 & 0.1022 & 0 \\ -0.1022 & 0 & 0 \end{bmatrix}. \tag{4.58}$$

The reference command conversion matrix G_c here is simplified with only constant DC gains, it is necessary to be expressed with other dynamics if large-envelop maneuver flights are required.

4.7 Control performance evaluations

Automatic hovering flight performance is important to evaluate the quality of the controller design. In this section, an automatic hovering flight is conducted. The procedures can be briefly described as: the safety pilot flies the helicopter to a specified proper location (e.g., at a height 7 m) manually; the ground command operator then sends a command to ask the helicopter to do automatic hovering; the safety pilot switches the control from manual to automatic.

Fig. 4.2 shows the experimental results from an experiment. It can be seen that the pilot switched the control from manual to automatic at about $t = 90$ s. There is step change with the position references then. From the plot, it can be seen that the peek-to-peek values of the position control errors are within 1 m or even less. The peek-to-peek yaw angle control error is within 2 degrees if zoom in the plot. The control performance is considered good with a GPS measurement unit to provide the position measurements (i.e., the CEP error of the position measurement from GPS is about 2.5 m as shown in Table 2.1).

4.8 Conclusion

Both the hardware platform construction and the automatic controller design lay the foundations of robotic research. In this chapter, the methods used to design a comprehensive controller for helicopter is presented. The structure of the controller consists of two layers, the inner-loop layer and the outer-loop layer. The inner-loop layer is used to stabilize the attitude of the helicopter and the outer-loop layer is used for controlling the translational movements. Due to the complex dynamics of the inner-loop of the helicopter, an advanced robust H_∞ controller is designed for the inner-loop. For the simplicity of accepting high-level commands, such as position, velocity and acceleration references from the trajectory generator, a so-called robust and perfect tracking controller is designed for the outer-loop. Experimental results from real flight tests show that the performance of the designed automatic controller is satisfactory for our vertical replenishment task.

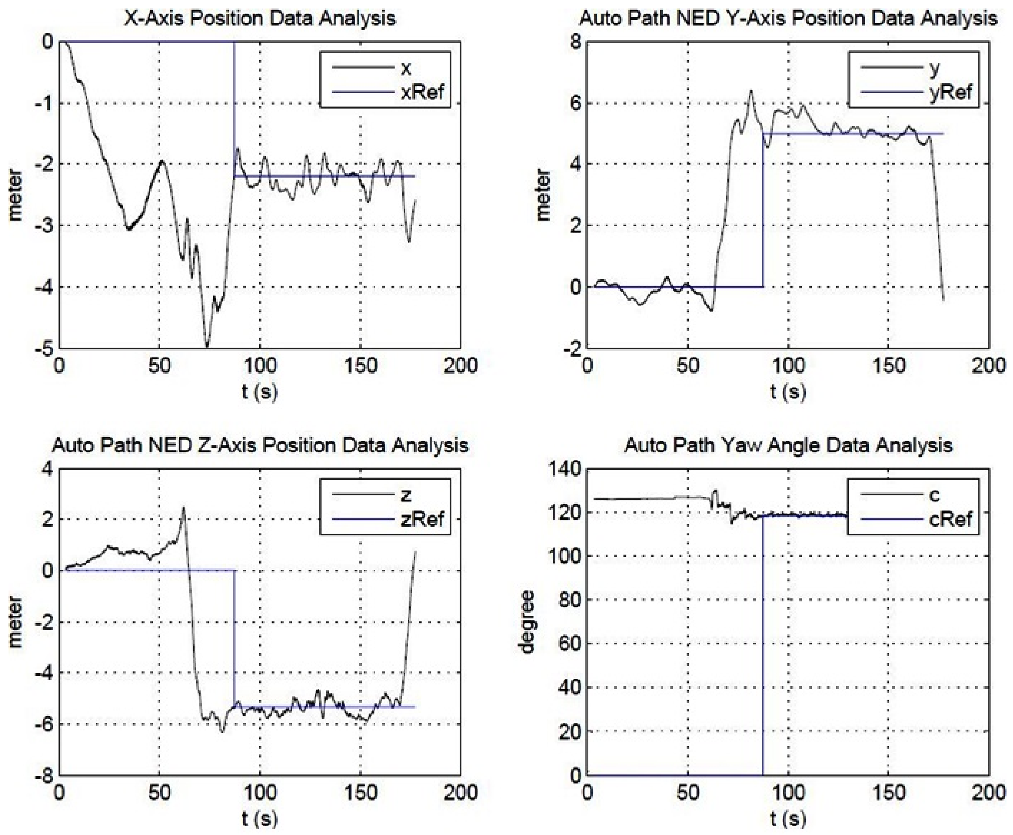


Figure 4.2: Automatic hovering performance of NUS^{2}T-Lion

Chapter 5

State Estimations

5.1 Introduction

Measurements are essential and important for automatic feedback control systems. Poor measurements usually require proper compensations of the controller, which is difficult to achieve and to result in good overall closed-loop performances.

Common measurement devices are the inertial measurement unit (IMU), Global Positioning System (GPS), 2D laser scanner, 3D Lidar, radar, RGB camera and RGBD camera. Due to the large size and heavy weight, 3D Lidar and radar are usually not suitable for small unmanned aerial vehicles. Due to the inheriting characteristics of RGBD cameras, which use the infrared-light to generate images, they are usually not suitable for outdoor usages. Thus, in this application, 3D Lidar, radar and the RGBD camera will not be chosen due to the small size of our helicopter and the helicopter will operate in outdoor environments.

In this chapter, state estimation techniques based on sensors selected in our hardware configurations will be presented. The fusion technique used for IMU and GPS devices will be firstly presented. They provide global positions, velocities, accelerations, euler angles and angular rates' measurements. However, the precision of the position measurements still cannot satisfy the requirements for precise cargo grabbing and unloading tasks. Thus, A RGB camera is selected to detect the cargos and get the estimations of the relative distance between our helicopter and cargo for precision guidance. To get accurate height measurement, a 2D laser scanner is selected and its measurement is fused together with acceleration.

5.2 Linear kalman filter

Consider an linear-time-invariant (LTI) system characterized by

$$\begin{cases} \dot{x} = Ax + Bu + v(t) & v \text{ is the input noise} \\ y = Cx + w(t) & w \text{ is the measurement noise} \end{cases} \tag{5.1}$$

Assume:

1. (A, C) is observable
2. $v(t)$ and $w(t)$ are independent white noises with the following properties

$$\begin{aligned} &E[v(t)] = 0, \quad E[v(t)v^T(\tau)] = Q\delta(t-\tau), \quad Q = Q^T \geq 0, \\ &E[w(t)] = 0, \quad E[w(t)w^T(\tau)] = R\delta(t-\tau), \quad R = R^T > 0, \end{aligned} .$$

3. $(A, Q^{\frac{1}{2}})$ is stabilizable (to guarantee closed-loop stability).

The problem of Kalman Filter is to design a state estimator to estimate the state $x(t)$ by $\hat{x}(t)$ such that the estimation error covariance is minimized, i.e., the following index is minimized

$$J_e = E[x(t) - \hat{x}(t)^T x(t) - \hat{x}(t)]$$

The solution can be found by constructing a system described by

$$\begin{cases} \dot{\hat{x}} = A\hat{x} + Bu + K_e(y - \hat{y}), \quad \hat{x}(0) \text{ is given} \\ \hat{y} = C\hat{x} \end{cases}$$

with the Kalman Filter gain K_e being given by

$$K_e = P_e C^T R^{-1}$$

where P_e is the positive definite solution of the following Riccati equation,

$$P_e A^T + AP_e - P_e C^T R^{-1} C P_e + Q = 0$$

Let $e = x - \hat{x}$, it can shown that such a Kalman Filter has the following properties

$$\lim_{t\to\infty} E[e(t)] = \lim_{t\to\infty} E[x(t) - \hat{x}(t)] = 0, \quad \lim_{t\to\infty} J_e = \lim_{t\to\infty} E[e^T(t)e(t)] = traceP_e$$

The above described Kalman Filter are standard Kalman Filter for linear-time-invariant systems. The equations are adopted from a control class lecture notes, which can be found in [15]. It will be used for the laser scanner height measurement fusion in later section.

5.3 Inertial measurement fusion with GPS

The inertial navigation system usually fuses the inertial measurements together with GPS measurements. The inertial measurement unit usually consists of the accelerometer (to measure the translational accelerations), gyroscope (to measure the angular velocities), and magnetometer (to measure the heading direction). Extended Kalman Filter is commonly used to design the fusion algorithm [12].

In this thesis, the used inertial navigation system is a commercial product, all the internal algorithms are hidden to end users. The readers are suggested to read [12] for the commonly used fusion algorithms.

Fig. 5.1 and Fig. 5.2 show the measurements for translational movements and angular movements of the used SBG IG500n device. The device is placed stationary. It can be seen that the peek-to-peek position measurement errors are within 3 m. The corresponding velocity errors are within 0.4 m/s and there are also constant biases for the x and y channel. The peek-to-peek euler angles measurements are within 1 degree. The corresponding angular velocities are within 0.01 rad/s. The experiment results described above further justify that the other sensors are needed for precision cargo grabbing and unloading, which usually requires the automatic control errors (position) being bounded within 0.5 m or even less.

Thus, we choose the 2D laser scanner for the height precision measurement. A RGB camera is chosen for precise pose estimations between cargo and helicopter, which is used for precision guidance. The state estimation algorithms used for both kind of sensors will be described in the remaining sections.

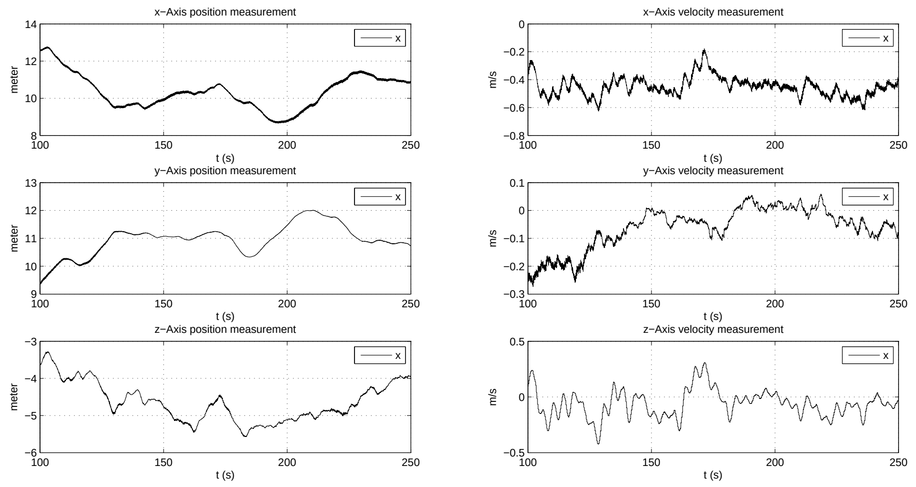


Figure 5.1: Translational movement measurements of SBG IG500n at stationary condition

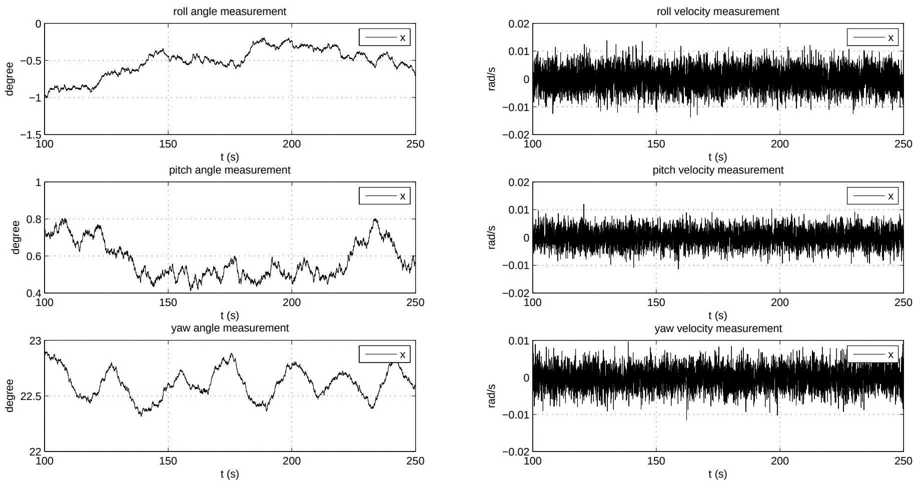


Figure 5.2: Angular movement measurements of SBG IG500n at stationary condition

5.4 Height measurement via laser scanner

As mentioned previously, a very accurate height measurement is needed for the cargo loading and unloading tasks. In fact, it is also the most helpful information for the UAV to carry out autonomous taking-off and landing. Motivated by this, a high-end scanning laser range finder is installed onboard of the UAV platform. The corresponding algorithm to calculate the UAV height via its range measurements is explained below.

For each frame of scanning, the laser sensor will output 1081 integer numbers representing the measured distances in millimeter from its starting point on the right to the end point on the

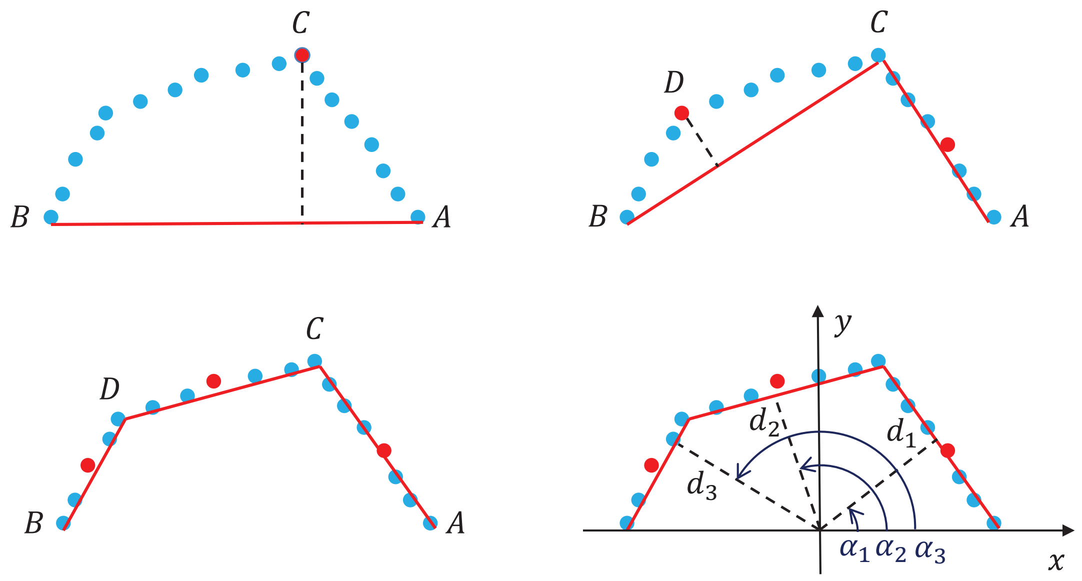


Figure 5.3: The *split-and-merge* algorithm for line extraction

left sequentially. Each distance data is associated with its own angle direction, thus the data can be seen as in polar coordinates. A simple transformation can be applied to the raw measurement data to convert it from polar coordinates (r_i, θ_i) to Cartesian coordinates (x_i, y_i) by

$$\begin{cases} x_i = r_i \cos\theta_i \\ y_i = r_i \sin\theta_i \end{cases}, \tag{5.2}$$

where $i \in \{1, 2, 3, \ldots, 1081\}$ is the index of the laser scanner measurements. Then, the *split-and-merge* algorithm 10 is applied to this array of 2D points so that they can be divided into clusters, with each cluster of points belonging to an individual line segment. The main steps of the *split-and-merge* algorithm is summarized below with Fig. 5.3 giving a graphical illustration.

- Connect the first point A and the last point B of the input data by a straight line.
- Find point C among all data points that has the longest perpendicular distance to line AB.
- If this longest distance is within a threshold, then a cluster is created containing points in between A and B.
- Else, the input points will be split into two subgroups, A-C and C-B. For each group, the *split-and-merge* algorithm will be applied recursively.

Clusters of points will be created thereafter. Least square line fitting algorithm can then be

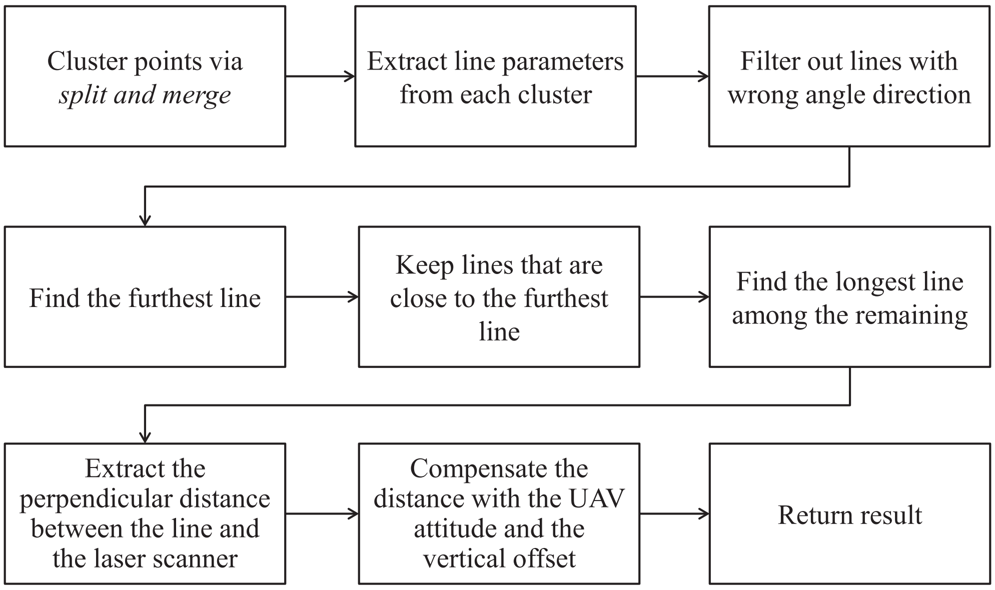

Cluster points via *split and merge*
Extract line parameters from each cluster
Filter out lines with wrong angle direction
Find the furthest line
Keep lines that are close to the furthest line
Find the longest line among the remaining
Extract the perpendicular distance between the line and the laser scanner
Compensate the distance with the UAV attitude and the vertical offset
Return result

Figure 5.4: Steps to compute height via laser scanner measurement

applied to points in each cluster to obtain the individual lines. Each line can be represented by two parameters, namely the line's normal direction α_k and its perpendicular distance to the center of laser scanner d_k. In the last sub-figure of Fig. 5.3, xy axes represent the laser scanner frame. Normal direction of the line is defined as the angle from the x-axis to the line normal, counterclockwise as positive. The next step is to filter out lines with dissimilar gradient as the ground plane. Since the obtained lines are still expressed in the laser scanner frame, their directions α_k should be compensated by the UAV roll angle ϕ and then compared with the normal line of the ground plane which is at $\pi/2$. Let

$$\Delta\alpha_k = \alpha_k - \phi - \pi/2. \tag{5.3}$$

If $|\Delta\alpha_i|$ is greater than a threshold, say 5 degrees, then the corresponding line is filtered out. The remaining lines are sorted by their perpendicular distances to the laser scanner and the furthest ones are kept. Among them, the longest line is believed to be the true ground. Finally, the perpendicular distance of this line to the laser scanner center is compensated with the UAV pitch angle θ and the offset between the laser scanner and the UAV CG, Δh, leaving the final height estimation to be

$$h = r\cos(\theta) - \Delta h. \tag{5.4}$$

Fig. 5.4 has shown the flow chart of the laser scanner based height calculation algorithm. By using this method, an accurate height measurement can be obtained as long as the laser scanner projects a portion of its laser beams onto the true ground. Hence, it even works for the case when the UAV flies over protruding objects on the ground or on the ship surface.

5.5 Height measurement fusion

Since there are two sources of height measurements, one from GPS/INS and the other from laser scanner, it is best to combine them so that the most reliable and accurate UAV state variables in the z-axis, i.e. $\boldsymbol{x}_\mathrm{h} = \begin{bmatrix} z & w_g & a_{z,g} & \delta_z \end{bmatrix}^\mathrm{T}$, can be obtained. Here, z is the UAV vertical height with respect to the ground surface, w_g and $a_{z,g}$ are the corresponding velocity and acceleration in this axis and δ_z is the offset between the GPS/INS height and the laser counterpart. This offset has to be considered because the two sensory systems have different zero references and it also accounts for the time-varying position bias of the GPS/INS sensor. Here, we also formulate the estimator by considering the physical dynamics of a single-axis mass point system as:

$$\begin{cases} \dot{\boldsymbol{x}}_\mathrm{h} = A_\mathrm{h}\boldsymbol{x}_\mathrm{h} + E_\mathrm{h}\boldsymbol{w}_\mathrm{h} \\ \boldsymbol{y}_\mathrm{h} = C_\mathrm{h}\boldsymbol{x}_\mathrm{h} + \boldsymbol{v}_\mathrm{h} \end{cases}, \tag{5.5}$$

where $\boldsymbol{x}_\mathrm{h} = \begin{bmatrix} z & w_g & a_{z,g} & \delta_z \end{bmatrix}^\mathrm{T}$

$$A_\mathrm{h} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \; E_\mathrm{h} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}, \; C_\mathrm{h} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}, \tag{5.6}$$

and $\boldsymbol{w}_\mathrm{h}$, $\boldsymbol{v}_\mathrm{h}$ are Gaussian noises with covariance matrices Q_h and R_h respectively. Q_h and R_h can be chosen by analyzing signal noise levels logged in UAV hovering flight test with the assumption that all measurements are Gaussian and independent of each other. In our implementation, they are set as:

$$Q_{\mathrm{h}} = \begin{bmatrix} 0.1^2 & 0 \\ 0 & 0.01^2 \end{bmatrix}, \ R_{\mathrm{h}} = \begin{bmatrix} 0.05^2 & 0 & 0 & 0 \\ 0 & 0.8^2 & 0 & 1 \\ 0 & 0 & 0.5^2 & 0 \\ 0 & 0 & 0 & 0.3^2 \end{bmatrix}. \tag{5.7}$$

By discretizing the system at a sampling rate of 50 Hz and applying Kalman filter, a reliable estimation of UAV height can be obtained. In implementing this filter, an additional technique is utilized to discard occasional outliers in the height measurement from the laser scanner. The idea is to check whether the received measurement is within a threshold multiply of the current process noise. If the discrepancy is too large, then the measurement at this particular instance is ignored. In [54], a similar technique is introduced and it is called the *innovation filter*. Figs. 5.5–5.6 show the height estimation result via data collected in one of the flight tests. It can be seen that the fused result has higher quality than the original height information from GPS/INS or laser scanner alone. The problem of slow drifting of GPS/INS (see Fig. 5.5) and a few small outliers from laser height measurement (see Fig. 5.6) are not affecting the fused result too much. At the same time, the estimated values of UAV vertical velocity and acceleration are also less noisy than their respective raw measurements from GPS/INS (see Fig. 5.7–5.8).

Fig. 5.5 and Fig. 5.6 show that the height measurement from laser scanner is sufficiently good for that set of experiment. However, to solve a practical problem, the worst situation should be considered. If the UAV is flying around the edge of the ship, the laser scanner may have two different measurements alternatively changing since it may consider the ground as the baseline or it may consider the ship deck as the baseline. This kind of measurement changing (jump) is bad to UAV's automatic flights. Therefore, a measurement fusion by the above mentioned Kalman filter is still required to handle this kind of "false measurements", since the UAV is actually not moving in the heave direction relative to the inertial frame.

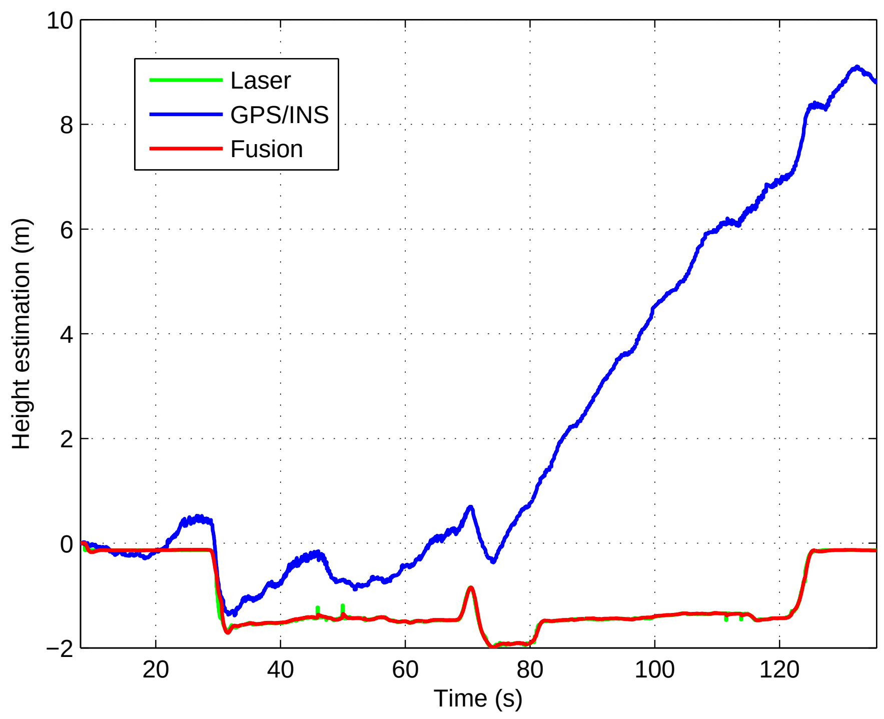


Figure 5.5: Result of height estimation by data fusion

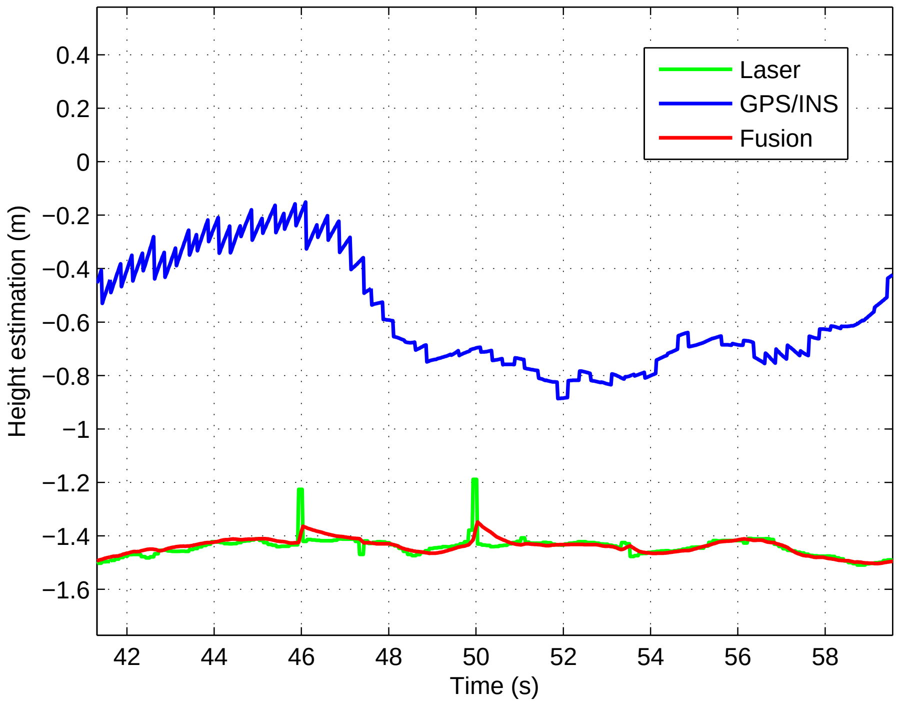


Figure 5.6: Result of height estimation by data fusion (zoomed in)

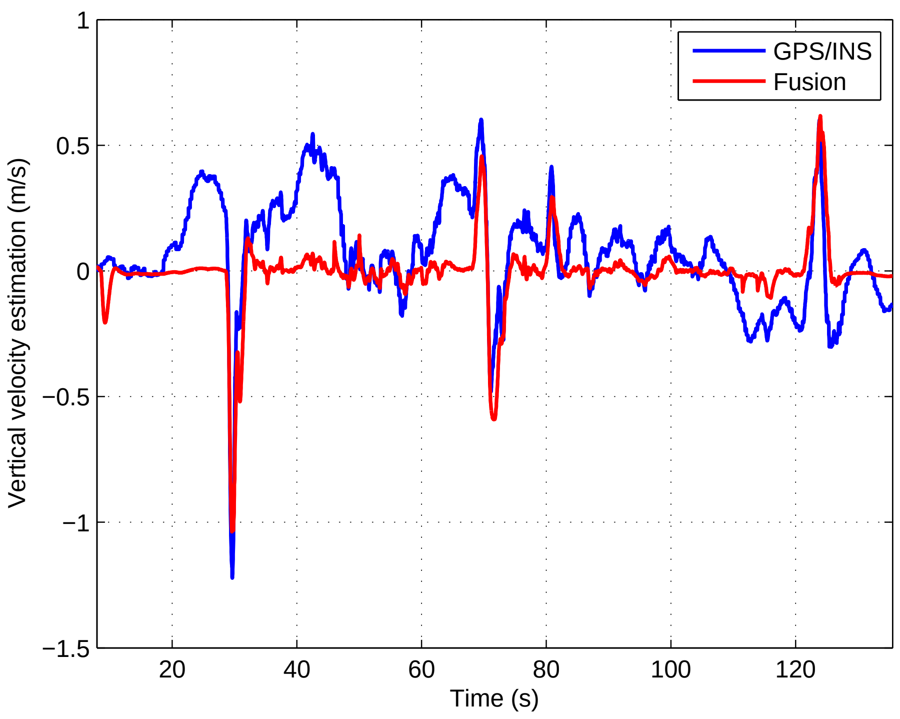


Figure 5.7: Result of vertical velocity estimation by data fusion

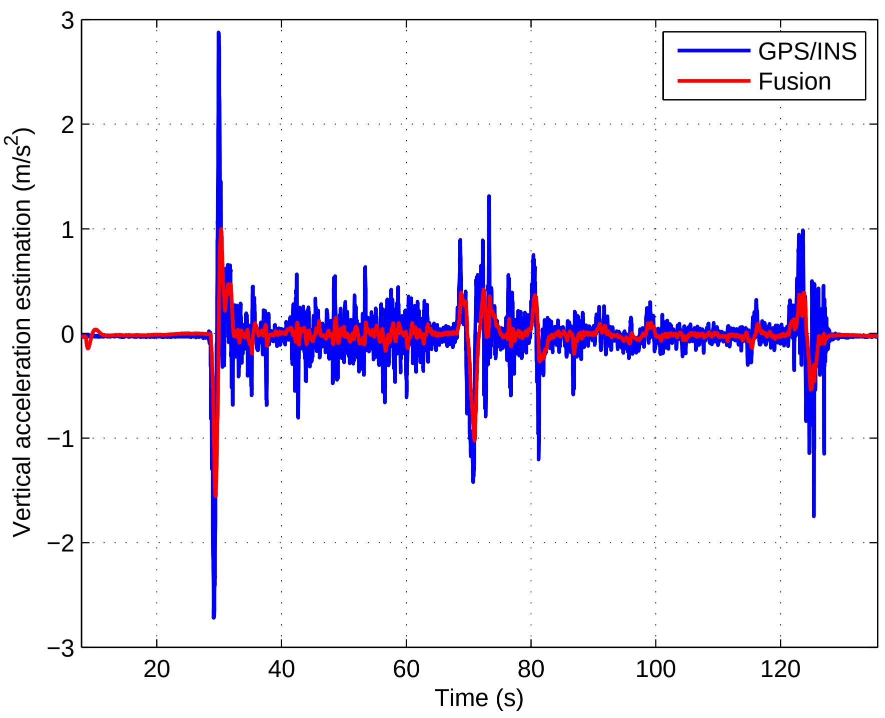


Figure 5.8: Result of vertical acceleration estimation by data fusion

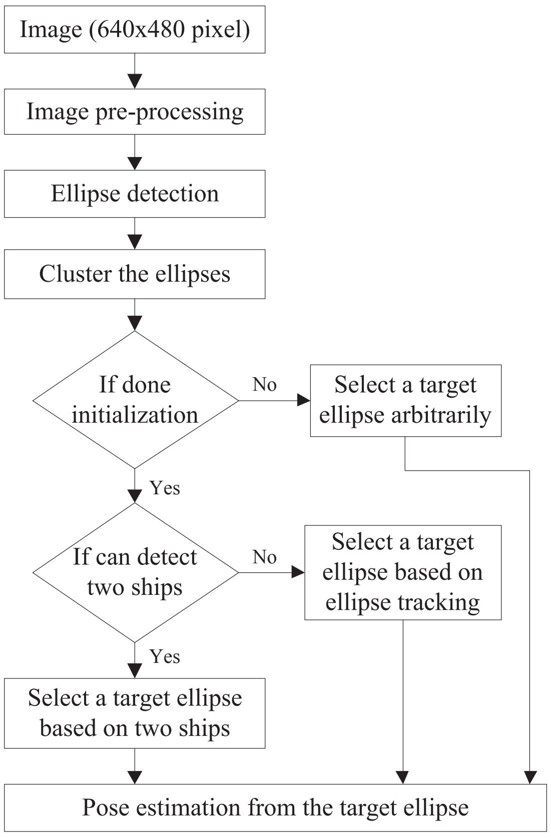


Image (640x480 pixel)
Image pre-processing
Ellipse detection
Cluster the ellipses
If done initialization
No
Select a target ellipse arbitrarily
Yes
If can detect two ships
No
Select a target ellipse based on ellipse tracking
Yes
Select a target ellipse based on two ships
Pose estimation from the target ellipse

Figure 5.9: Flow chart of the vision system

5.6 Vision-based target localization

Due to the inaccuracy of the inertial measurement system (IMU/GPS), a vision-based target detection and localization system is developed for precision bucket grabbing and unloading. The vision system will provide relative distance measurement between the helicopter and the cargoes. These information will be further used for automatic guidance to grab and unload cargoes. In the context of the vertical replenishment tasks, the cargoes to be transported are in the form of plastic buckets located inside circular patterns drawn on the ship surfaces. The first idea naturally comes into mind is to use image processing to detect circles and then do circle-based pose estimation. However, it should be noted that after camera projection, circles become ellipses in an image. Hence the main task of the vision system here is to first select the correct target ellipse in the captured image and then estimate the 3D pose of the actual circle on the ship surface. The flow chart of the vision system is given in Fig. 5.9. Before zooming into the detailed steps, three key algorithms in this vision system need to be highlighted. They are *ellipse detection*, *ellipse tracking* and *single-circle-based pose estimation*. These three algorithms are not restricted to the specific UAVGP competition tasks, but also applicable to many other UAV guidance tasks such as circle-based target following and circle-based landing.

- *Ellipse detection* has been investigated extensively in literature. Ellipse fitting, introduced in 2, 21 is chosen as the core ellipse detection algorithm in this work, because it is very efficient compared to hough transform based ellipse detection algorithms proposed in 4, 37. Unfortunately, ellipse fitting only fits the best ellipse for a given contour without questioning whether the contour is suitable to be seen as an ellipse in the first place. To complement its shortage, a three-step procedure, consisting of pre-processing, ellipse fitting and post-processing, is proposed and implemented for real-time and robust ellipse detection. The pre-processing is based on affine moment invariants (AMIs) 22, while the post-processing is based on the algebraic error between the contour and the fitted ellipse. The three-step procedure is not only robust against non-elliptical contours, but also can handle partial occlusion cases.

- *Ellipse tracking* is to continuously track a single ellipse after its detection has been initialized. In practical applications, multiple ellipses may be detected in an image but only one of them is to be targeted. There are two main challenges for ellipse tracking. First, the areas enclosed by the ellipses (the interested one and the others) are similar to each other in both shape and color. Thus, template matching based on color, shape or feature points may not be suitable for this task. Second, when implementing vision-based tracking algorithms on a flying UAV, the fast dynamical motion of the UAV may cause large displacement of the target ellipse between two consecutive images. In order to track the target ellipse smoothly, the frame rate of the image sequence must be high, which requires a very efficient implementation of the vision algorithm. To best solve these problems, an efficient image tracking method CAMShift 3 is chosen as the core of the tracking algorithm in this work. This algorithm can robustly track the target ellipse even when the scale, shape or color of the ellipse area are dynamically changing.

- *Single-circle-based pose estimation* is to calculate the 3D position of the target circle after its projected ellipse on the 2D image has been identified and tracked. Circle-based camera calibration and pose estimation have been studied in 20, 26, 27, 29, 51. However, these existing work mainly focused on the cases of concentric circles 20, 27, 29, but our aim is to do pose estimation via only one circle. Theoretically, it is impossible to estimate the pose of a single circle purely from its perspective projection 51. But from a practical point of view, the single-circle-based pose estimation problem can be solved by adopting a reasonable assumption that the image plane of the camera is parallel to the plane that contains the circle. This assumption is satisfied in our work because the onboard camera is installed on a pan-tilt mechanism which can ensure the image plane to be always parallel to the ground plane. By exploiting this assumption, the 3D position of the targeted circle can be estimated from its projection as a single ellipse in the image.

More detailed explanations and discussions about these vision algorithms are documented in another paper due to its own research significance in the vision society 53. We next explain the steps shown in Fig. 5.9.

- *Image* – Color images are captured at 5 Hz by the onboard camera. The image resolution is 640×480 pixels.
- *Image pre-processing* The purpose of this block is to detect all the possible contours that may be formed by the projected circles. It consists of four sub-steps, namely image undistortion, RGB to HSV conversion, color thresholding and contour detection. Since the color information of the circles has been given in the competition, the contours of the circles can be efficiently detected based on color thresholding in the HSV color space. It is also possible to detect the contours using other methods such as edge detection. However, they will consume more computational power.
- *Ellipse detection* – The perspective projections of the circles are ellipses. Based on the contours given in the previous step, the ones that correspond to ellipses should be detected.
- *Ellipse clustering* – The main aim for ellipse clustering is to decide whether the two ships are in the camera's field of view. If they are, there should be two clusters of ellipses. Since the four ellipses on each ship are of the same size and they are distributed evenly in a line, the size and position information can be used for ellipse clustering. If two clusters are obtained, then we can conclude that the two ships are in the field of view.
- *Initialization* – The UAV takes off at a location far from the ships and is guided to the two ships based on GPS. Once the vision system can detect two ships, an initialization procedure will be triggered. The purpose of the initialization procedure is to select a proper target circle according to the UAV's current task. The tracking algorithm is also initialized in this step.

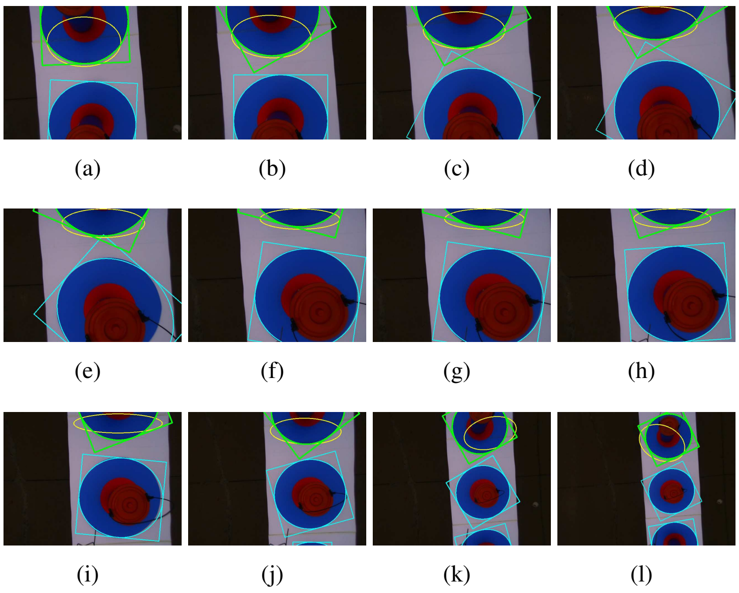


Figure 5.10: Onboard images with the ellipse detection and tracking result

- *Select a target ellipse arbitrarily* – This is a failsafe mechanism. After the UAV takes off, it will be guided to the ship area by GPS. However, the position measurement from GPS is not very precise. Hence, the UAV may not be guided to the exact position at which both ships are in the onboard camera's field of view. To handle this kind of situation, the vision system will return any detected ellipse until the *initialization* condition has been detected.

- *Select a target ellipse based on two ships* – If two ships are in the field of view, we will select the target ellipse based on the knowledge of the initial placements of the cargoes and the current task status. Suppose the buckets are initially placed on the right ship and they are required to be taken to the left one. If the current mission for the UAV is to grab a bucket, the vision system will select one circle that contains a bucket from the right ship. If the current mission for the UAV is to drop a bucket, the vision system will select one circle that does not contain a bucket from the left ship.

- *Select a target ellipse based on ellipse tracking* – In many of the cases, the two ships may not be in the field of view simultaneously. Then ellipse tracking algorithm is used to track a target ellipse over the image sequence.

- *Pose estimation from the target ellipse* – Once the target ellipse is selected in any of the ways mentioned above, the ellipse will be used to estimate the position of the circle center relative to the camera. The detailed method is explained in [53].

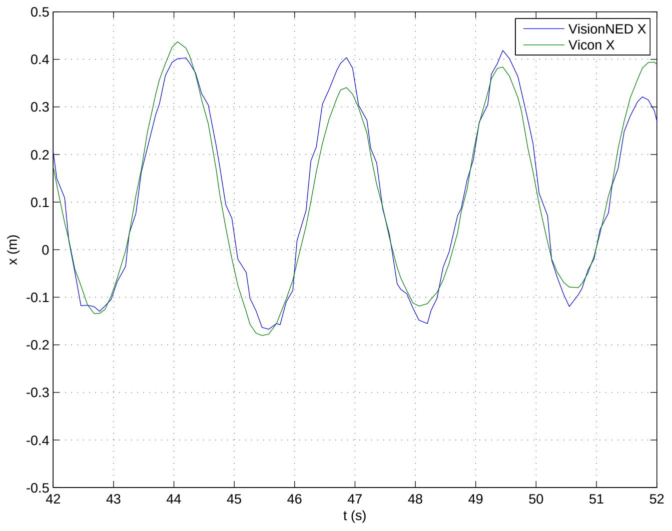
Fig. 5.10 shows a number of consecutive images taken by the onboard camera. All the detected ellipses have been drawn on each image. The green ellipse is the target ellipse tracked by the vision algorithm. The yellow ellipse is the area of interest returned by the CAMShift algorithm. It can be seen that all the ellipses have been successfully detected. The target ellipse is also tracked steadily even when its scale and shape keep varying.

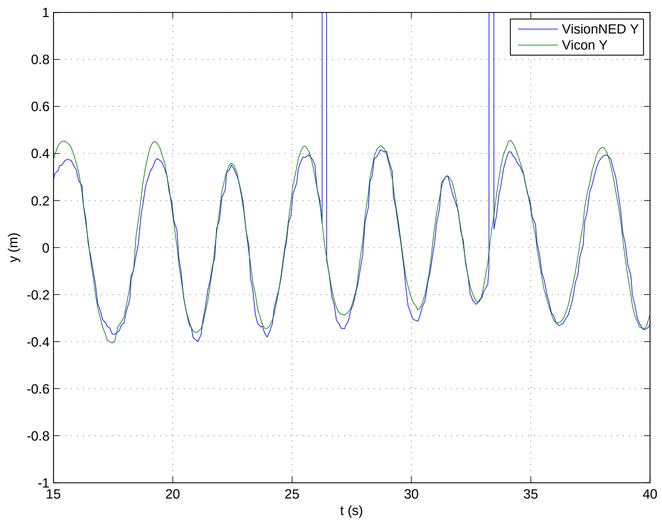
To verify the position estimation of the vision algorithm, experiments with a motion capture system (VICON) as the ground truth were conducted. The motion capture system can provide precis position measurements in *mm* level. It can be shown from Fig. 5.11 that the position estimation from the developed vision algorithm matched well with the measurement provided by VICON. The spikes shown in the figure are due to the blockage of camera when we did experiments. It shows that our developed vision algorithm is accurate.

5.7 Conclusion

In this chapter, we present the techniques used for estimating position, velocity and angular movements of our helicopter platform. High quality state estimation is necessary for precision automatic control.

Thus, a 2D laser scanner is chosen for the height precision measurement. A linear Kalman filter is also designed to fuse both the accelerations measurement and the GPS altitude measurement. Due to the inaccuracy of the IMU/GPS devices, a vision system is developed to provide relative precision horizontal (i.e., x and y axis) measurements. Ellipse detection and single-circle based pose estimation algorithms make up the overall vision system. Experiment results have shown the accuracies of robustness of the state estimation algorithms for both the laser scanner measurement and the camera measurement. The height measurements and the relative pose estimations are satisfactory for precise bucket grabbing and unloading.





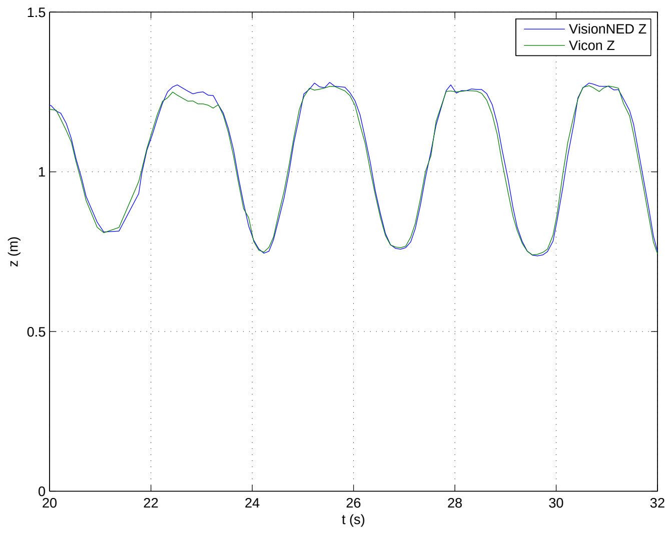


Figure 5.11: Comparison of measurements between vision algorithm and VICON

Chapter 6

Trajectory Generations

6.1 Introduction

Real-time trajectory planning and generation play an important role for robots to finish certain tasks in dynamic environments. In some applications, it is possible not to generate the trajectories in real-time, for example, waypoint flight for an unmanned aerial vehicle (UAV). The trajectories can be solved by transforming it to certain optimization problems and then computed through MATLAB offline. However, it is almost impractical to generate fixed trajectories offline for the helicopter to achieve the vertical replenishment task since the ships are moving. There are also many other uncertainties in the environments. Thus, the helicopter should have the capability to re-plan and re-generate the flight trajectories to overcome the effects resulted from these uncertainties.

One of the fundamental problems here is how to transform the command (e.g., fly ahead 3 m and fly left 5 m) from the flight planner in real-time to understandable command (e.g., 50 Hz set-point command, the flight controller is running at 50 Hz) for the low-level flight control module. The problem here is called the trajectory generation problem. The trajectory planning algorithm will be covered in later chapter through a flowchart of procedures.

Thus, in this chapter, we will present an algorithm, which can generate 50 Hz set-point command for the flight control module if given targeted position and maximum allowed velocities constraints in real-time.

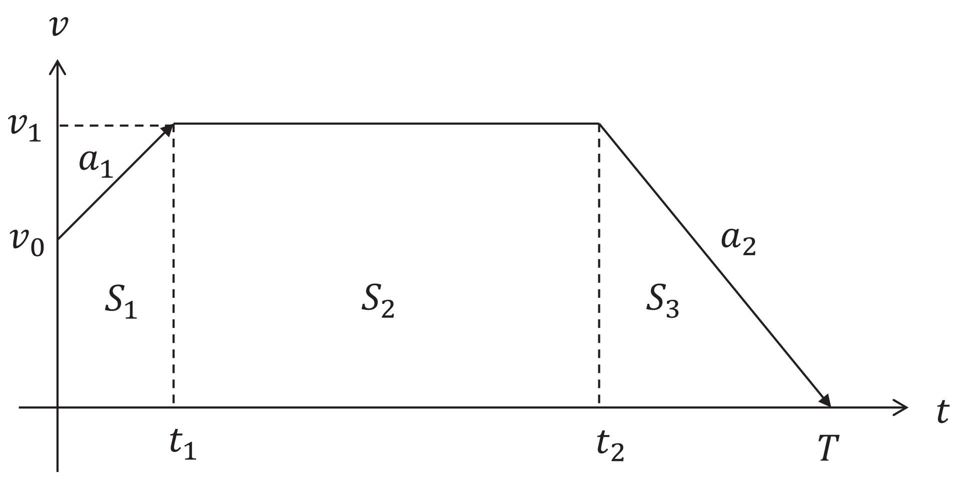


Figure 6.1: Trajectory planning with continuous velocity

6.2 Trajectory generation

The trajectory generation problem is being formulated to find the solution $\{t_1, t_2\}$ given the velocity profile as shown in Fig. 6.1. There are many other types of velocity profiles, Fig. 6.1 is one of them for illustration purpose. The basic philosophy for finding the solutions is that "increase T if V_1 exceeds the velocity constraint". The fundamental idea of the proposed trajectory planning algorithm has the following characteristics:

- The resulting position reference is continuous and smooth.
- The resulting velocity reference is continuous.
- The resulting acceleration reference is non-continuous and only have three discrete possibilities, $a_{\max}$, $-a_{\max}$ and 0.
- The trajectory can start from a non-zero velocity but always ends at zero velocity.
- The area under the velocity profile from time 0 to T integrates to the total displacement on each axis.

Viewing Fig. 6.1 in a geometrical way, one can get the following two relationships:

$$\begin{aligned} S &= S_1 + S_2 + S_3 \\ &= \frac{1}{2}(v_0 + v_0 + a_1 t_1)t_1 + (v_0 + a_1 t_1)(t_2 - t_1) \\ &\quad + \frac{1}{2}(v_0 + a_1 t_1)(T - t_2), \end{aligned} \tag{6.1}$$

$$v_1 = v_0 + a_1 t_1 = -a_2(T - t_2). \tag{6.2}$$

If v_0, a_1, a_2 and T are known, t_1 can be solved in a quadratic equation form:

$$At_1^2 + Bt_1 + C = 0, \tag{6.3}$$

where

$$\begin{cases} A &= a_1^2 - a_1 a_2, \\ B &= 2v_0 a_1 + 2a_1 a_2 T, \\ C &= v_0^2 + 2a_2 v_0 T - 2a_2 S. \end{cases} \tag{6.4}$$

Define $D = B^2 - 4AC$, then

$$\begin{cases} t_1 = -C/B & \text{if} \quad A = 0; \\ t_1 = \dfrac{-B - \sqrt{D}}{2A} & \text{if} \quad AC > 0 \ \& \ AB < 0 \ \& \ D > 0; \\ t_1 = \dfrac{-B + \sqrt{D}}{2A} & \text{if} \quad AC < 0 \ \& \ D > 0; \\ t_1 = -1 & \text{if} \quad \text{otherwise}, \end{cases} \tag{6.5}$$

and correspondingly,

$$t_2 = T + \frac{v_0 + a_1 t_1}{a_2}. \tag{6.6}$$

However, a_1, a_2 and T are not known exactly. To solve this problem, a recursive algorithm by listing all four cases of a_1-a_2 combination and continuously increasing T by 1-second step is proposed as Fig. 6.2. The iteration stops until a feasible solution occurs.

In actual implementations, this trajectory planning algorithm runs at 1 Hz only because it consumes high computational power with respect to an embedded computer. In addition, instead of inputting v_0 as the current velocity measurement, we use the current velocity reference, and instead of accumulating on the current position measurement for future position reference, we accumulate on the current position reference. This is to make sure that the velocity and position

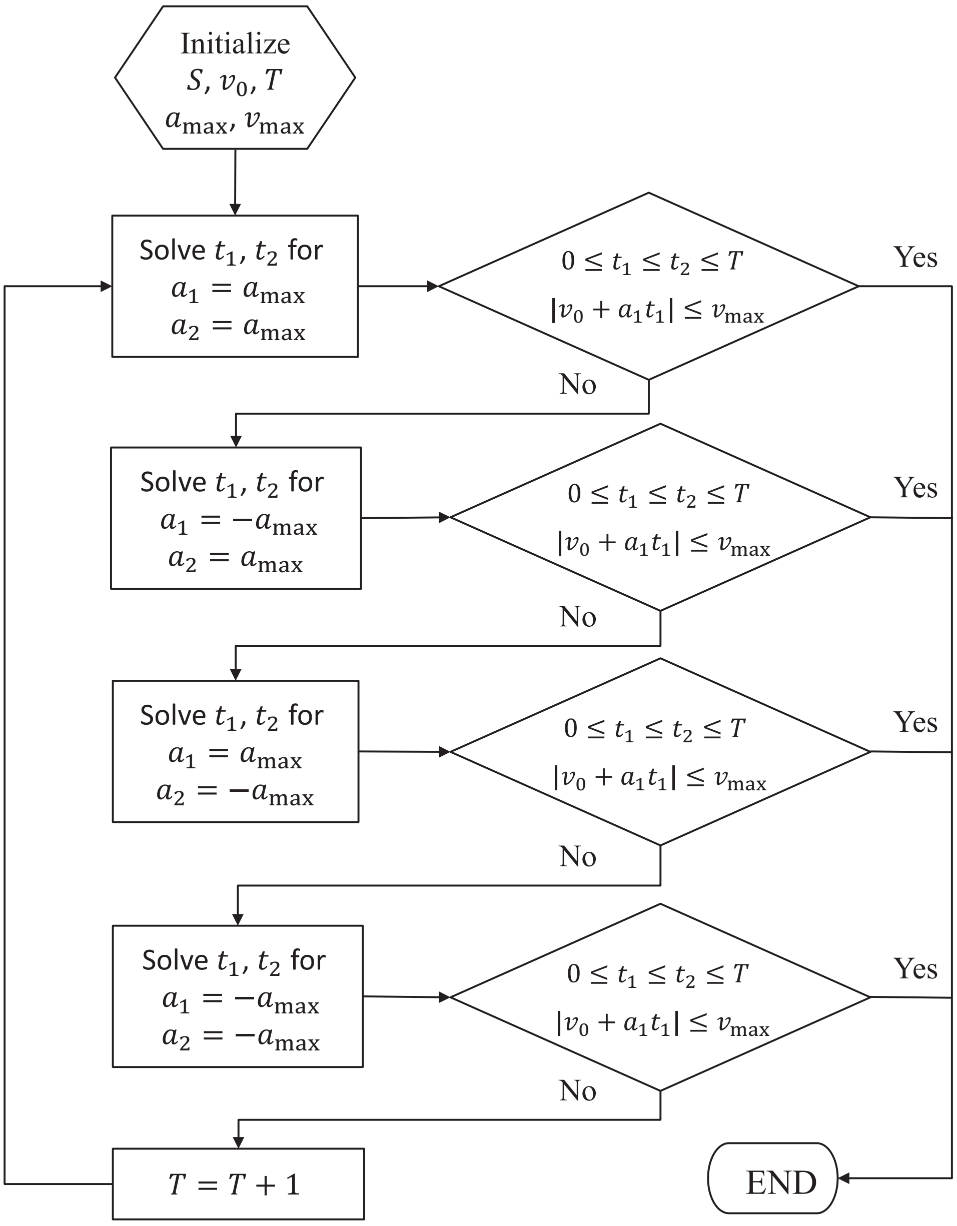


Figure 6.2: Flowchart of the trajectory planning algorithm

reference signals are always continuous. In fact, it is quite reasonable because at a slow rate of trajectory planning with strict velocity and acceleration constraints, the UAV's actual position, velocity and acceleration should be settled to more or less the same value of their corresponding reference signals at every instance the trajectory planning algorithm is called.

Furthermore, the algorithms illustrated above are only to find the solution for a single axis, say x-axis. However, it usually involves three dimensional movements for helicopters, i.e., in $x-y-z$ axis. One more constraint here is the total flight time for each axis, i.e., T_x, T_y and T_z should equal with each other.

The details of the algorithms for generating trajectories are shown in Algorithms 1, 2, 3 and 4. Algorithm 1 is the main routine for generating trajectories. The user may specify the distance to move, the initial velocity, the maximum speed and acceleration allowed as well as an "initial guess" for the total time of the trajectory for the required path. The routine calls the Algorithm 2 to find the correct trajectory for the single x and y axis, respectively. The above procedures are repeated until a feasible trajectory is found. The total time for the trajectory will be increased by a constant step Δt for each iteration such that the velocity constraint can be satisfied. Algorithm 3 is used to find the solution $\{t_1, t_2\}$ as shown in Fig. 6.1 given a specified input set $\{\Delta d, v_0, a_1, a_2, T\}$ for single axis. Algorithm 4 is used to check whether the resulted trajectory satisfies the specified velocity constraint in Algorithm 2.

Algorithm 1 Trajectory Generator

```
1: procedure REFERENCEGENERATOR($\Delta x, \Delta y, v_{x0}, v_{y0}, \overline{v_{max}}, \overline{a_{max}}, T_0$)    ▷ $t_{x1}, t_{y1} < 0$ is considered as incorrect solution
2:     $T \leftarrow T_0 - \Delta t$
3:     repeat
4:         $T \leftarrow T + \Delta t$
5:         $t_{x1}, t_{x2}, a_{x1}, a_{x2} \leftarrow$ SINGLEAXISTRAJGENERATOR($\Delta x, v_{x0}, \overline{v_{max}}, \overline{a_{max}}, T$)
6:         $t_{y1}, t_{y2}, a_{y1}, a_{y2} \leftarrow$ SINGLEAXISTRAJGENERATOR($\Delta y, v_{y0}, \overline{v_{max}}, \overline{a_{max}}, T$)
7:     until $t_{x1} \neq -1$ and $t_{y1} \neq -1$
8:     return $t_{x1}, t_{x2}, a_{x1}, a_{x2}, t_{y1}, t_{y2}, a_{y1}, a_{y2}, T$
9: end procedure
```

6.3 Trajectory generation evaluations

Fig. 6.3 shows one set of outputs given by the trajectory generator module. The trajectory is generated for moving 4 m in the positive x-direction; 3 m in the positive y-direction with an initial velocity at −0.3 m/s and −0.5 m/s in the x and y direction respectively. The norms of

Algorithm 2 Single-Axis Trajectory Generator

```
    procedure singleAxisTrajGenerator($\Delta d, v_0, \overline{v_{\max}}, a, T$)  ▷ This procedure solves the trajectory for a single axis
2:      $t_1 \leftarrow -1$
        $t_2 \leftarrow -1$
4:      $a_1 \leftarrow a$
        $a_2 \leftarrow a$
6:      $t_1, t_2 \leftarrow$ solver($\Delta d, v_0, a_1, a_2, T$)
        if isSolutionCorrect($t_1, t_2, v_0, a_1, T, \overline{v_{\max}}$) then
8:          return $t_1, t_2, a_1, a_2$
        end if
10:     $a_1 \leftarrow a$
        $a_2 \leftarrow (-1) \cdot a$
12:     $t_1, t_2 \leftarrow$ solver($\Delta d, v_0, a_1, a_2, T$)
        if isSolutionCorrect($t_1, t_2, v_0, a_1, T, \overline{v_{\max}}$) then
14:         return $t_1, t_2, a_1, a_2$
        end if
16:     $a_1 \leftarrow (-1) \cdot a$
        $a_2 \leftarrow a$
18:     $t_1, t_2 \leftarrow$ solver($\Delta d, v_0, a_1, a_2, T$)
        if isSolutionCorrect($t_1, t_2, v_0, a_1, T, \overline{v_{\max}}$) then
20:         return $t_1, t_2, a_1, a_2$
        end if
22:     $a_1 \leftarrow (-1) \cdot a$
        $a_2 \leftarrow (-1) \cdot a$
24:     $t_1, t_2 \leftarrow$ solver($\Delta d, v_0, a_1, a_2, T$)
        if isSolutionCorrect($t_1, t_2, v_0, a_1, T, \overline{v_{\max}}$) then
26:         return $t_1, t_2, a_1, a_2$
        end if
28:     return $t_1, t_2, a_1, a_2$
    end procedure
```

Algorithm 3 Solver

```
procedure SOLVER($\Delta d, v_0, a_1, a_2, T$)    ▷ $t_1, t_2 < 0$ is considered as incorrect solution
    $t_1 \leftarrow -1, t_2 \leftarrow -1$
                                                  ▷ Solve Quadratic Equation $ax^2 + bx + c = 0$
    $a \leftarrow a_1^2 - a_1 \cdot a_2$
    $b \leftarrow 2 \cdot a_1 \cdot v_0 + 2 \cdot a_1 \cdot a_2 \cdot T$
    $c \leftarrow v_0^2 + 2 \cdot a_2 \cdot v_0 \cdot T - 2 \cdot a_2 \cdot \Delta d$
    if $a == 0$ then
        $t_1 \leftarrow -c/b$ ; $v_{\max} \leftarrow v_0 + a_1 \cdot t_1$
        $t_2 \leftarrow (v_{\max} + a_2 \cdot T)/a_2$
    else if $(b^2 - 4 \cdot a \cdot c) \geq 0$ and $a \neq 0$ then
        $t_{s1} \leftarrow (-b + \sqrt[2]{b^2 - 4 \cdot a \cdot c})/(2 \cdot a)$
        $t_{s2} \leftarrow (-b - \sqrt[2]{b^2 - 4 \cdot a \cdot c})/(2 \cdot a)$
        if $t_{s1} < 0$ and $t_{s2} < 0$ then
            $t_1, t_2 \leftarrow -1$
            return $t_1, t_2$
        else if $t_{s1} \cdot t_{s2} < 0$ then
            $t_1 \leftarrow$ MAX$(t_{s1}, t_{s2})$ ; $v_{\max} \leftarrow v_0 + a_1 \cdot t_1$
            $t_2 \leftarrow (v_{\max} + a_2 \cdot T)/a_2$
        else
            $t_1 \leftarrow$ MIN$(t_{s1}, t_{s2})$ ; $v_{\max} \leftarrow v_0 + a_1 \cdot t_1$
            $t_2 \leftarrow (v_{\max} + a_2 \cdot T)/a_2$
        end if
        if MIN$(t_1, t_2) < 0$ then
            $t_1, t_2 \leftarrow -1$
            return $t_1, t_2$
        end if
    else
        $t_1, t_2 \leftarrow -1$
    end if
    return $t_1, t_2$
end procedure
```

Algorithm 4 Checks whether solution is correct

```
procedure ISSOLUTIONCORRECT($t_1, t_2, v_0, a_1, T, \overline{v_{\max}}$) ▷ This function checks whether the
solution is correct
    if $t_1 \geq 0$ and $t_1 \leq t_2$ and $t_2 \leq T$ and ABS$(v_0 + a_1 \cdot t_1) \leq \overline{v_{\max}}$ then
        return TRUE
    else
        return FALSE
    end if
end procedure
```

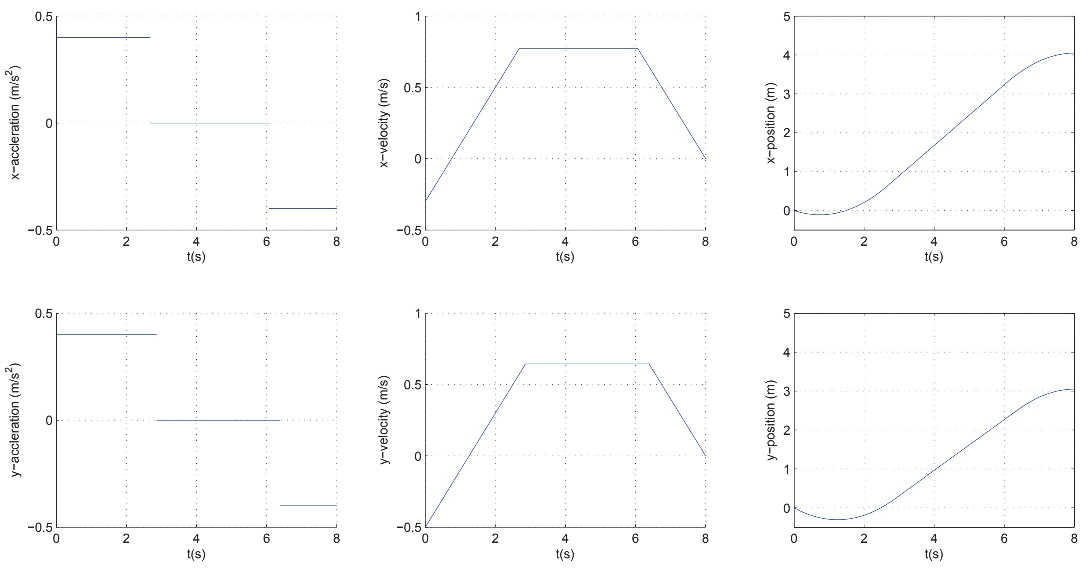


Figure 6.3: Plots of the result from the trajectory generator (Δx = 4 m, Δy= 3 m, $v_{\mathrm{x0}} = -0.3$ m/s, $v_{\mathrm{y0}} = -0.5$ m/s, $\overline{v_{\max}}$ = 2 m/s and $\overline{a_{\max}}$ = 0.4 m/s^2)

the velocity and acceleration are constrained within 2 m/s and 0.4 m/s^2 respectively. It can be seen from the plots that the trajectory generator module provides smooth trajectories. The drawback of the module is that the generated acceleration reference is not smooth, which might be unsatisfied for aggressive maneuvers. However, it is satisfactory for our current system, where the helicopter is assumed to maneuver at low speed in this work.

The algorithm can be further improved to provide smooth acceleration references for aggressive maneuvers. There are also some commercial softwares which can generate this kind of trajectories, such as the reflexxes motion libraries [24].

6.4 Conclusion

In this chapter, we present a real-time online trajectory generation algorithm for the helicopter. Due to the uncertainties involved in the vertical replenishment tasks, such a kind of trajectory generation module is necessary. The algorithm employs a simple search algorithm to find a feasible solution given the velocity constraints, acceleration constraints and distance to travel. Although the resulted acceleration is not smooth. the experiment results have shown that the resulted trajectories is satisfactory for our application, where the helicopter is moving in low speed.

Chapter 7

System Integrations

7.1 Introduction

In preceding chapters, we presented the detailed implementations of the necessary functional blocks of the unmanned helicopter for vertical replenishment application. These blocks are the hardware platform construction system, flight control system, state estimation and perception system, and the trajectory generation system. Both the hardware platform construction system and the flight control system lay the foundations for high-level robotic intelligences. To enable the helicopter with proper intelligence for precision cargo detection, grabbing and unloading capabilities, a laser and vision based environment perception system is developed. The system can estimate the relative 3D distance information between the helicopter and cargos precisely, which is useful for the guidance and navigation algorithm. The cargos can also be detected and identified by this system. The trajectory generation system is developed to make the helicopter transit from a state (position, velocity, etc.) to another smoothly.

To enable the full capability of the helicopter for vertical replenishment tasks, these functional blocks should be integrated together properly. Thus, in this chapter, we will present the methods for system integrations. In the following sections, the system overview will be presented. The detailed implementations of other functional blocks, such as flight planner, etc., which are used for integrating the above mentioned blocks will be explained in the remaining sections.

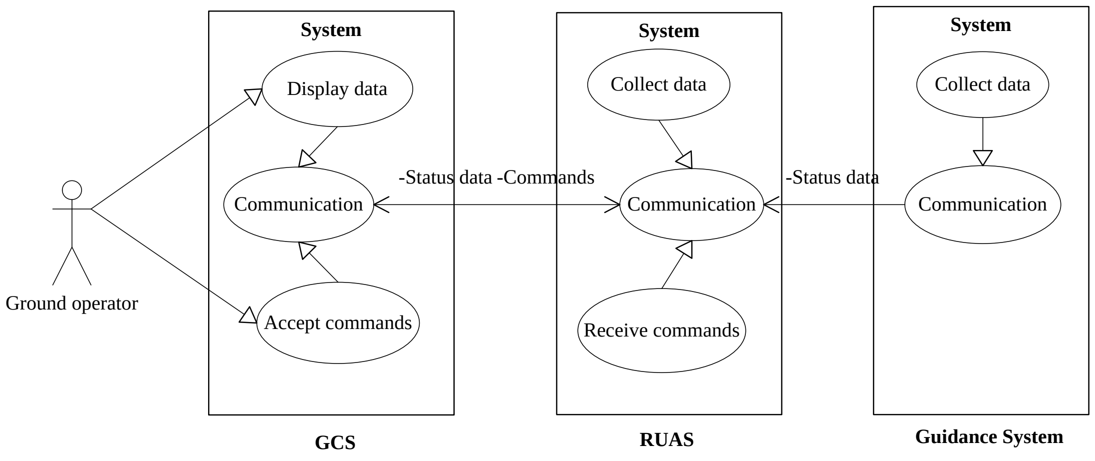


Figure 7.1: Overall data flow among software systems

7.2 System overview

As shown in Fig. 7.1, the whole system for vertical replenishment application is divided into three parts. The ground control system (GCS), the guidance system and the rotor-craft unmanned aerial system (RUAS). Together with the interactions between these systems, they make up the overall functional system to identify, grab, transport, unload the cargos from one ship to another automatically.

GCS is used as an interface between the human operator and the unmanned helicopter. It displays different kinds of information about the helicopter, such as the position, velocity, headings of the helicopter, etc. Meanwhile, the GCS is also used to transmit commands from the human operator to the unmanned helicopter.

It is possible that the cargos are not in the view of the helicopter vision system once the helicopter takes off from its home location. Thus, a GPS-based guidance system is developed to assist the unmanned helicopter to fly into the interested regions, where the moving ships are located, such that the vision system can detect where the cargos are. It provides status data of moving platforms to the unmanned helicopter, which includes the velocity, acceleration, position and moving direction of moving ships. The helicopter can be guided to the ships with these information known.

The unmanned helicopter plays the main role in the system. It accepts commands from the GCS and receives status data from the guidance system. It also streams down its own status data to the GCS for ground operators' monitoring. Different behaviors will be triggered automatically depends on received commands and perceived environment information. Different functional blocks together with the interactions among them make up the unmanned helicopter.

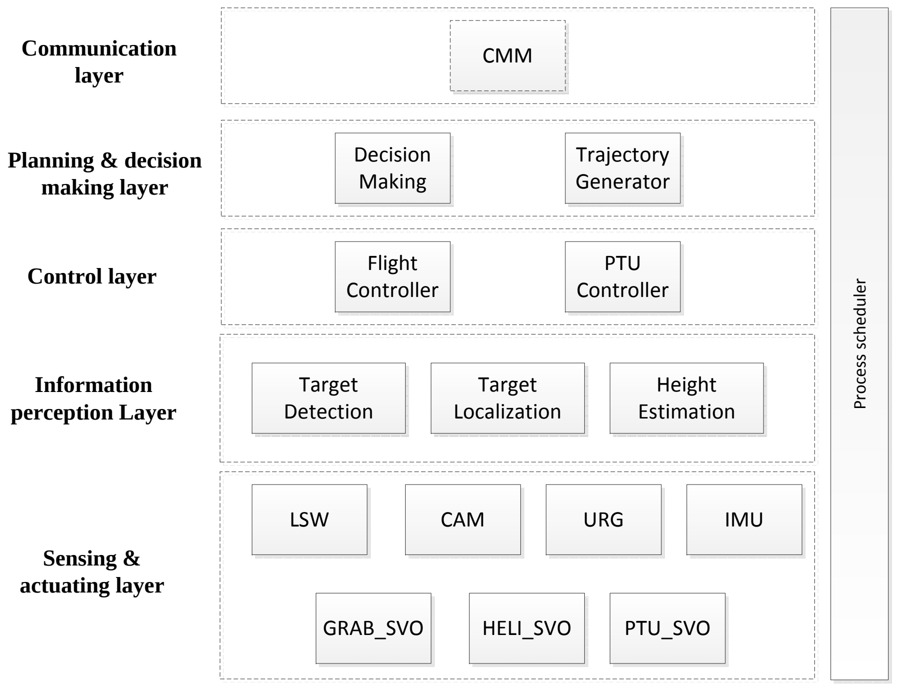


Figure 7.2: Functional blocks of the unmanned helicopter

In the remaining sections, we will focus on how to integrate these blocks together.

7.3 System integrations of the unmanned helicopter

To enable the helicopter with certain-level intelligences for cargo detection, precision grabbing and unloading, the functional blocks built for the unmanned helicopter should be properly integrated together. Interactions among the blocks are necessary. Thus, in this section, we will describe the interactions among them and how they interact with each other to make up a fully functional system for the vertical replenishment problem.

We divide the whole system into five main layers as shown in Fig. 7.2, which includes the sensing and actuating layer, the information perception layer, the control layer, the planning and decision making layer and the communication layer. Many of the layers have been described in previous chapters. All the functional blocks are implemented by an ARM processor and an onboard vision computer. The details about these layers will be explained in the following sections.

7.3.1 Sensing and actuating layer

The sensing and actuating layer is an abstraction layer of the hardware platform, which are used to sense the environment and to actuate the helicopter. It consists of seven main components,

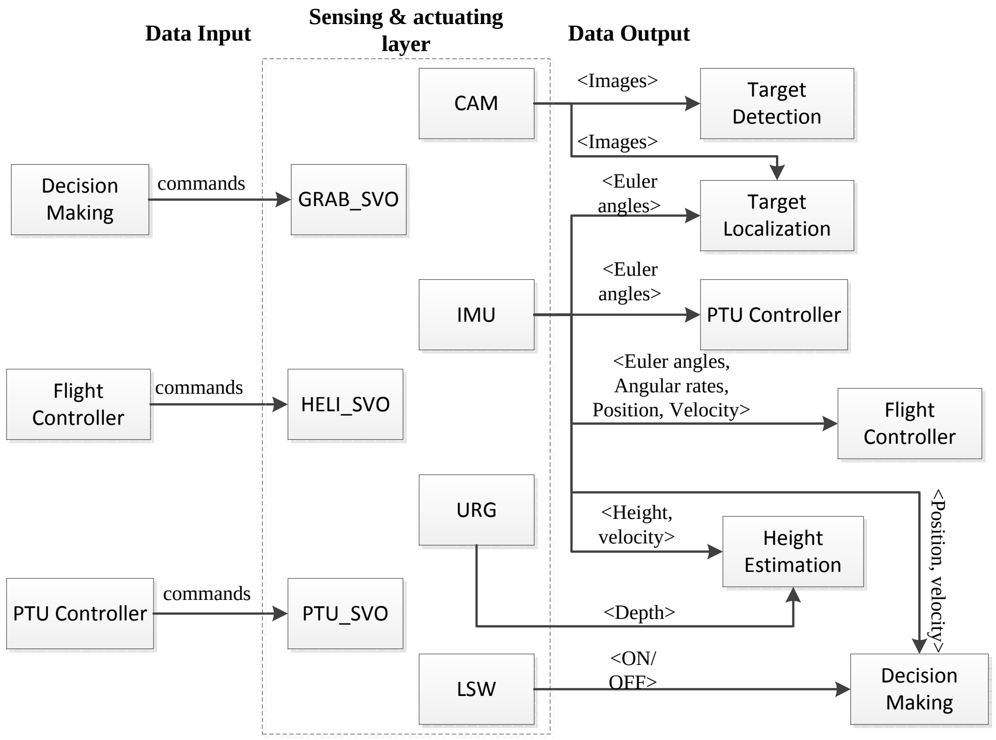


Figure 7.3: Data flow for sensing and actuating layer

which can be categorized into two types of hardware, i.e., the sensing type and the actuating type. The sensing type components consist of limit switch sensors (LSW), camera (CAM), 2D laser scanner (URG) and inertial measurement unit (IMU). The actuating type components consists of servos (GRAB_SERVO) for grabbing/releasing cargoes, servos (HELI_SVO) for actuating the helicopter's maneuvers and servos (PTU_SVO) for actuating the camera pan-tilt unit. The detailed configurations about these hardware have been given in Chapter 2. The detailed interactions of this layer with other layers are presented in Fig. 7.3.

The sensors will collect data and forward them to the upper layers for further processing. Two main layers, which will accept these information, are the flight control layer and the information perception layer. The flight control layer stabilize the helicopter through the euler angles, angular rates, position and translational velocity measurements (from IMU/GPS). The information perception layer further processes the measurements from camera and laser scanner to get the relative distances between helicopter and the cargos.

This layer also accepts commands from upper layers. Different types of actuators will be actuated in order to stabilize the helicopter, grab/unload the cargos, etc..

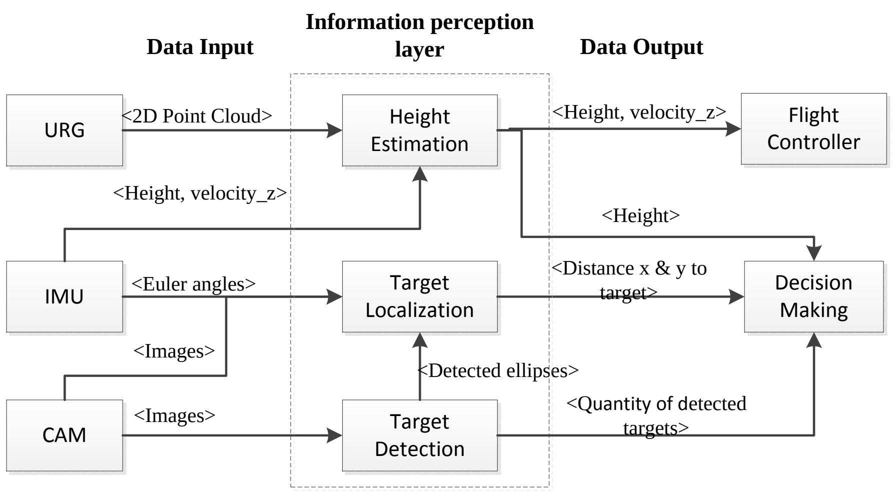


Figure 7.4: Data flow for information perception layer

7.3.2 Information perception layer

The information perception layer is used for further processing the raw sensory data from laser scanner and the camera. It consist of three main blocks. The height estimation block is used to estimate the relative height between the helicopter and the cargos, with laser scanner measurements. The height information is used for precision height automatic control of the helicopter.

The target (cargo) detection block is used to detect the cargos from raw image data of camera. The target (cargo) localization block is used to estimate the relative distance between the cargo and the helicopter once the cargo is identified from image data. These distance and detection information will be used for guidance by the decision making layer. The details are demonstrated in Fig. 7.4. Furthermore, the detailed implementations of these blocks have been explained in Chapter 5.

7.3.3 Control layer

The control layer consists of two blocks, i.e., the flight control block and the camera pan-tilt unit control block.

The implementations of the flight control block have been studied in Chapter 4. The functionality of the block is to stabilize the helicopter and enable the automatic waypoint-tracking function of the helicopter. It mainly interacts with the sensing layer for IMU/GPS measurements, perception layer for precision height measurement, guidance system through the communication layer for ships' status, and the decision making layer for waypoint references. The actuating layer (helicopter servos) will receive the outputs of this layer for proper servo actuat-

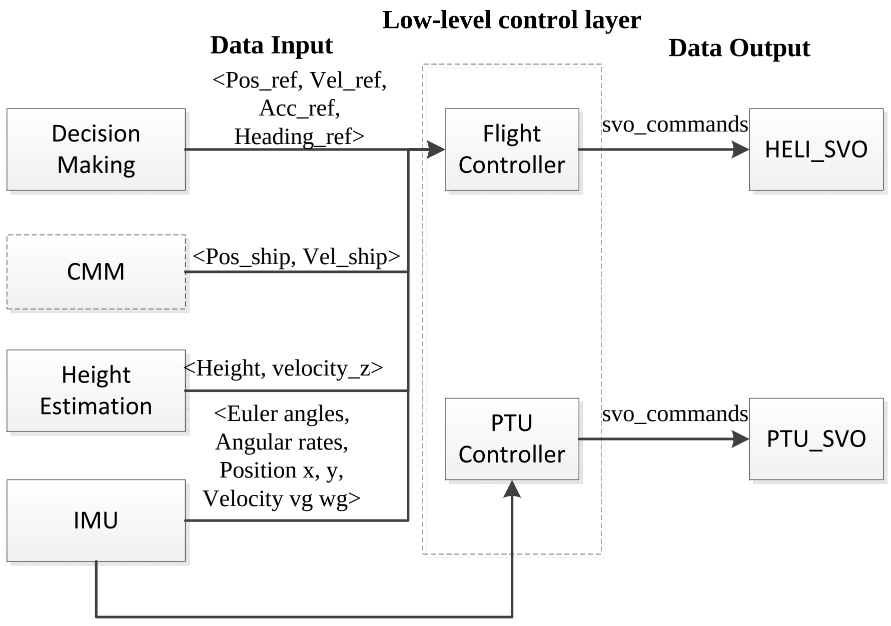


Figure 7.5: Data flow for low-level control layer

ing.

The camera pan-tilt unit controller is used to control the camera pan-tilt unit as a downward looking gimbal, which is useful for the vision algorithm's image acquisition. The camera is controlled to keep horizontal to the ships' surface all the time to full-fill an assumption of the vision algorithm. The detailed data flow for the control layer is defined in Fig. 7.5.

7.3.4 Planning and decision making layer

The planning and decision making layer is the flight planner. It consists of the decision making block and the trajectory generation block. The decision making block collects all necessary information from almost all lower-blocks to make decisions, such as when to execute return-home task, when to grab cargo, etc.. All these decisions are discrete events. In order to translate these events to useful commands for the control layer, a trajectory generator is developed as the interpreter. For example, if the decision making block ask the helicopter to take-off to 10 m high, the trajectory generator will will interpret this decision to 50 Hz set-point references for the flight controller to track. The details about the interactions of this layer with other layers are given in Fig. 7.6.

The implementations of the trajectory generator have been presented in Chapter 6. Thus, in the remaining sections, the implementations about the decision making block will be given in detail. The algorithm is implemented with a flowchart of procedures. It consists of the ship-

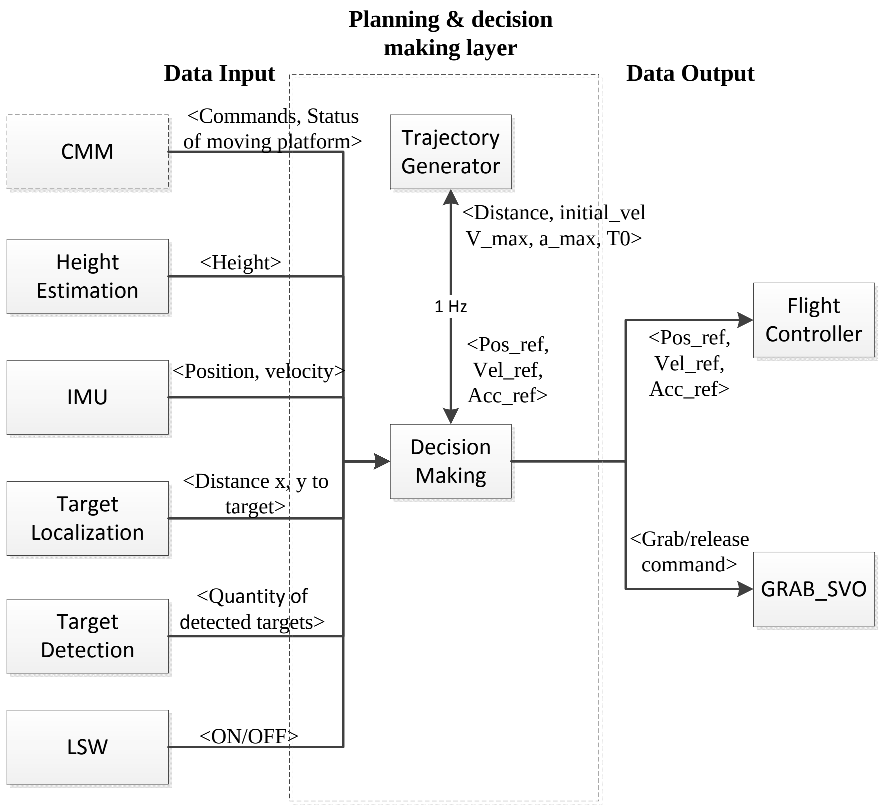

Planning & decision making layer
Data Input
Data Output
CMM
<Commands, Status of moving platform>
Trajectory Generator
<Distance, initial_vel V_max, a_max, T0>
Height Estimation
<Height>
1 Hz
<Pos_ref, Vel_ref, Acc_ref>
IMU
<Position, velocity>
Decision Making
Flight Controller
<Pos_ref, Vel_ref, Acc_ref>
Target Localization
<Distance x, y to target>
<Grab/release command>
GRAB_SVO
Target Detection
<Quantity of detected targets>
LSW
<ON/OFF>

Figure 7.6: Data flow for planning and decision making layer

frame navigation, vision guidance, real-time path planning, etc..

7.3.5 Communication layer

The communication layer behaves as a “bridge” between the helicopter and the external world. It accepts the high-level commands from ground operators through the GCS and feed-forward them to the decision making module. It also behaves as a “sensor”, which senses the dynamics about the moving platforms and provides these information to the flight control block and the decision making block. The flight status of the helicopter is also streamed down to the GCS for ground operators for status monitoring. The details about these data flow are demonstrated in Fig. 7.7.

7.4 Navigation

In order to synchronize with the motion of the ships, the controlled helicopter needs to know at every moment how the ships are moving. A simple and reliable solution is to install another GPS/INS sensor on the ship and send its information to the helicopter onboard system. By doing so, the helicopter can be controlled in a ship-referenced frame instead of the global frame. In

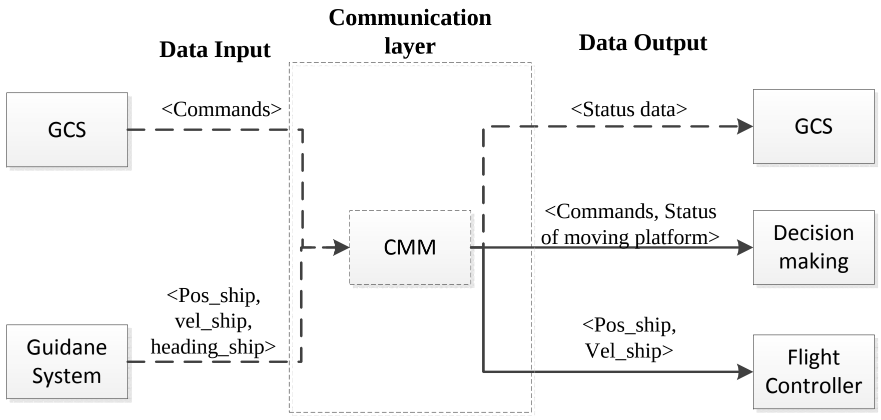


Figure 7.7: Data flow for communication layer

this ship-referenced frame, or called ship frame for simplicity, a zero steady-state position and velocity tracking error means the helicopter is controlled right above the ship with the same velocity as the ship.

However, it should be noted that this zero error is judged by the measurement difference of the two GPS/INS sensors. By considering their respective circular error probable, the measurement error sometimes goes beyond 6 m in all three axes. Obviously, it is not accurate enough for the cargo transportation task. In this section, the UAV z-axis measurement will be first complemented by fusing the information from a scanning laser range finder. As the height information extracted from the laser scanner is available and reliable for all time, it can be consistently fused in to improve the navigation accuracy in the z-axis. On the other hand, the x- and y-axis measurement errors are more difficult to be reduced in a navigation sense because there is no other sensor which can provide a consistent and accurate measurement in these two axes. However, after realizing that the best tracking performance is not really needed throughout the mission, but only required at the loading and unloading instances, they can be compensated in a guidance sense when the target enters the view angle of the onboard camera. This vision-based guidance will be discussed later in Section. Here, we only explain the ship-frame navigation.

Furthermore, the reason we choose to control the helicopter in the relative ship frame is that, during vertical replenishment, the ships may be stationary or in movement. To achieve the tasks, the unmanned helicopter needs to track the ships. To better handle both the stationary and moving cases, the origin of the coordinate frame is chosen on the moving ships. In this case, the coordinate frame used for the flight controller is independent of the translational movements of the ships relative to the earth. Thus, the stationary and moving cases can be treated equally since they are stationary or moving relative to the earth.

During the cargo delivering phase, it is complicated for the decision making block to command the helicopter where to fly if we use the local NED coordinate frame. For example, assume

- the heading direction of the ship is not pointing to the north (heading angle = 0 degree);
- the helicopter is controlled (outer-loop) in local-NED frame;
- the helicopter is currently located on ship A and the cargo is located on ship B;
- the perpendicular distance between ship A and B is 9 m;
- the helicopter needs to fly from ship A to ship B to grab the cargo;

In this case, the decision making module needs to express the 9 m distance to local-NED frame for the flight controller by using the heading of the ships. If the ship is continuously changing its direction, the frame conversion needs to be done continuously to avoid navigation error. It means the trajectory generation needs to be conducted more frequently instead of 1 Hz to avoid navigation error, which is not suitable for our trajectory generator. Thus, we use the ship-frame for the flight controller. Only one time trajectory generation is required, i.e., command the helicopter to fly 9 m in the y-direction of the ship frame and the heading changing of the ships can be handled in 50 Hz in the flight controller.

Thus, we choose the relative ship frame for our flight controller during the cargo delivering phase.

As measurements provided by GPS/INS sensors on the UAV and on the ship are defined in the same global frame and the motion of the ship involves no rotation, it is adequate to convert all position, velocity and acceleration measurements into the ship frame by simple subtraction. So,

$$\begin{cases} \mathbf{p} = \mathbf{R}_{s/g}(\mathbf{p}_{\text{uav}} - \mathbf{p}_{\text{ship}}) \\ \mathbf{v} = \mathbf{R}_{s/g}(\mathbf{v}_{\text{uav}} - \mathbf{v}_{\text{ship}}) \\ \mathbf{a} = \mathbf{R}_{s/g}(\mathbf{a}_{\text{uav}} - \mathbf{a}_{\text{ship}}) \end{cases} . \tag{7.1}$$

The outer-loop measurements and references are now both represented in the ship frame instead of the NED frame. Following this convention, it is also straight forward to convert the UAV heading angle to the ship frame as well. Hence,

$$\psi = \psi_{\text{uav}} - \psi_{\text{ship}}. \tag{7.2}$$

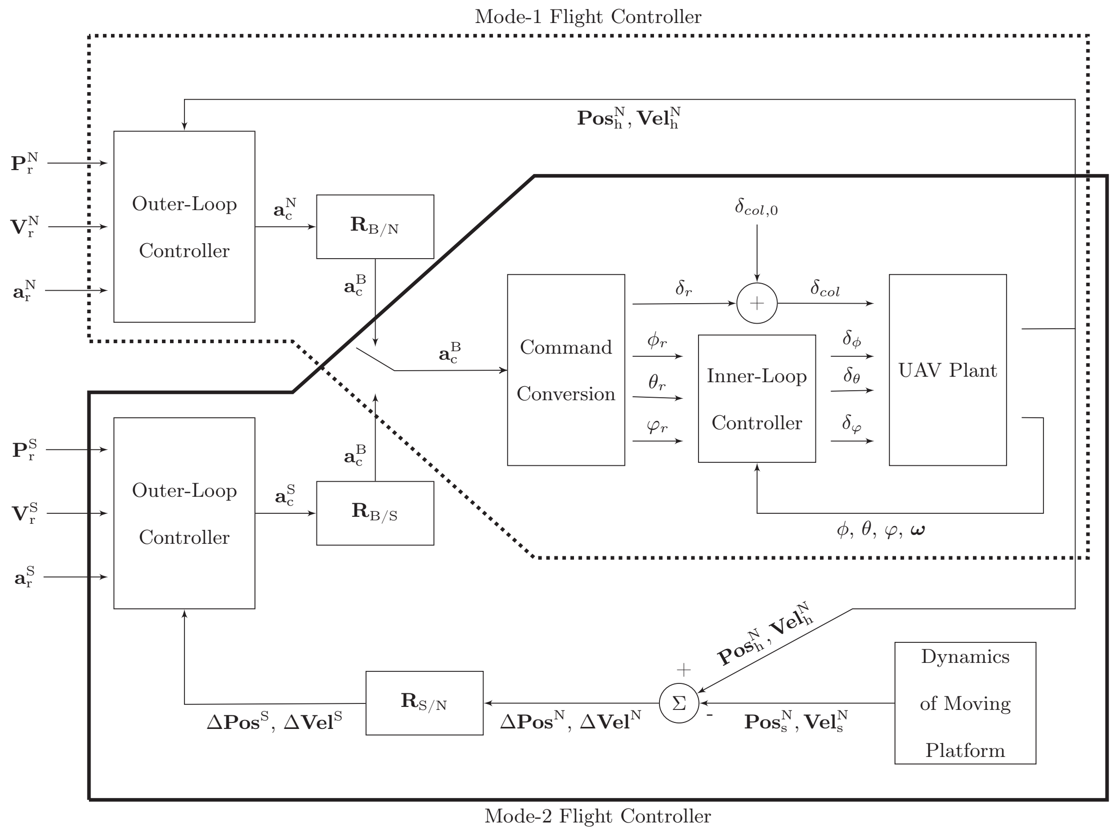


Figure 7.8: Dual frame flight controller architecture

By redefining ψ this way, the original rotational matrix $\mathbf{R}_{b/n}$ (rotation from the NED frame to the UAV body frame) can be substituted by $\mathbf{R}_{b/s}$, which is the rotation from the ship frame to the UAV body frame. Note that ϕ and θ in the rotational matrix are still the UAV roll and pitch angles as we assume that the ship has almost zero roll and pitch angles.

Fig. 7.8 illustrates the structure of the flight controller for navigation, which consists of the mode-1 controller and the mode-2 controller. It is almost the same as that discussed in Chapter 4. This dual frame flight controller consists of two main feedback loops, i.e., the inner-loop controller and the outer-loop controller. The inner loop controller, which is also called the attitude controller, is used to stabilize the attitude of the helicopter. The outer-loop controller is used to regulate the translational motion of the helicopter.

As shown in Fig. 7.8, the outer-loop controller of the mode-1 flight controller is designed in the local NED coordinate frame. The feedback measurements are the position and velocity information of the helicopter in local NED coordinate frame. This mode is used to navigate the helicopter for automatic taking-off, home returning and landing tasks as mentioned in the later decision making section. The shown mode-2 controller is the one we have discussed, the outer-loop of the controller is implemented in the relative ship heading frame. It is used for the

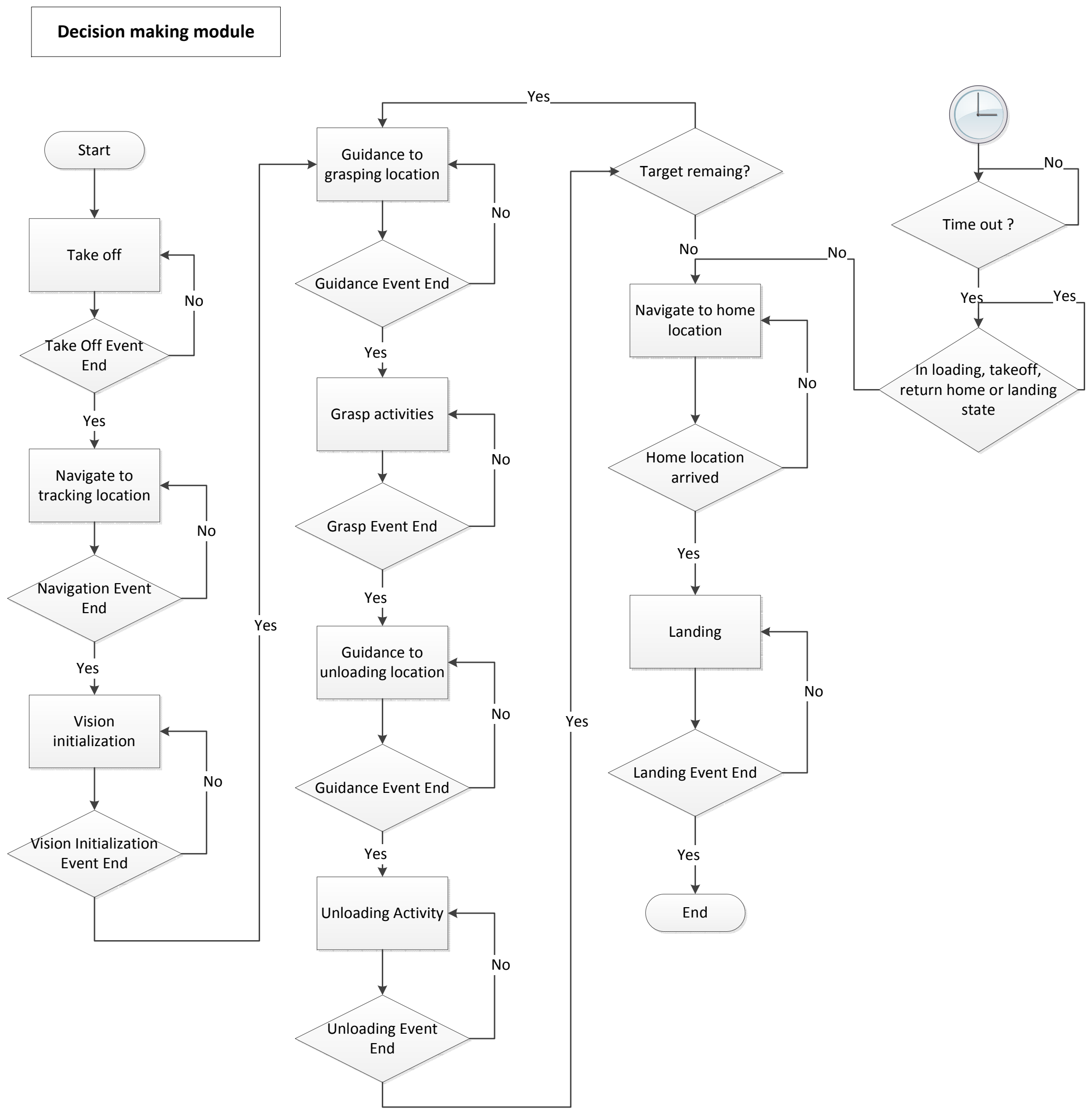


Figure 7.9: Decision making module

cargo delivery.

7.5 Guidance and decision makings

Fig. 7.9 shows a flowchart of the decision making module. The flowchart consists of five main sub-routines: taking-off routine, navigating to the moving platforms routine, vision initialization routine, task routine, returning home routine and landing routine. The mission may be time constrained, a timer interrupt routine is also implemented in the software. The software can trigger the returning home routine once the overall time is running out. The details of these routines will be discussed as following.

1. Prior to the taking-off stage, proper initialization will be performed for mission execution

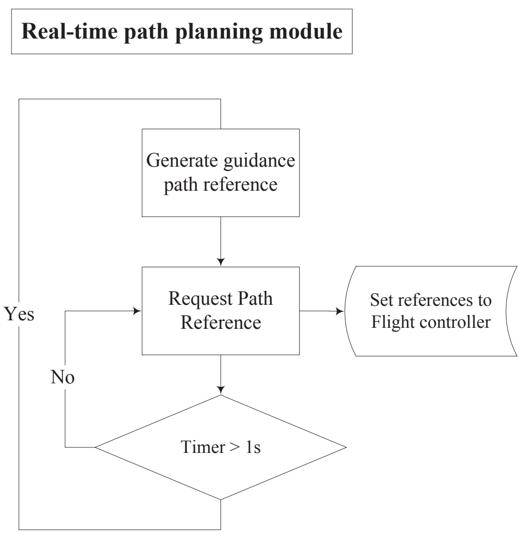

Real-time path planning module
Generate guidance path reference
Request Path Reference
Set references to Flight controller
Yes
No
Timer > 1s

Figure 7.10: Real-time path planning module

once the unmanned helicopter receives the “Execute the VERTREP” command from end users through GCS. The servo control signals for the throttle and collective pitch channels will then be increased to trim values such that the unmanned system can stay in a “pre-hover” state near the ground. The decision making module will then employ the trajectory generator to issue a “going up” trajectory to a height at 10 m (i.e., $\mathbf{Pos}_{\mathrm{r}}^{\mathrm{N}}$ = [0 0 −10] m) for the unmanned system. These trajectories will be issued to the mode-1 flight controller such that the unmanned system can reach at the final position. As the unmanned system arrives the required position (i.e., $\mathbf{Pos}_{\mathrm{h}}^{\mathrm{N}}$ = [0 0 −10] m), the software will trigger a “taking-off event end” signal such that the unmanned system can execute next scheduled routine, which is the navigating to the ship routine.

2. As the unmanned system completes the taking-off procedure, it will enter the navigating to the moving platform stage. The decision making module will first collect the position and velocity information of the moving platforms and the unmanned system respectively. Then it will calculate the relative initial distance vector, velocity difference vector between the moving platforms and the unmanned system. The decision making module will then ask the trajectory generator to generate the relative local-NED frame trajectories for the mode-2 flight controller. The mode-2 flight controller, which also considers the dynamics of the moving platforms in the feedback loop, will further navigate the unmanned system to catch up with the moving platforms.

3. As the unmanned system “catches up” with the moving platforms, the vision system will

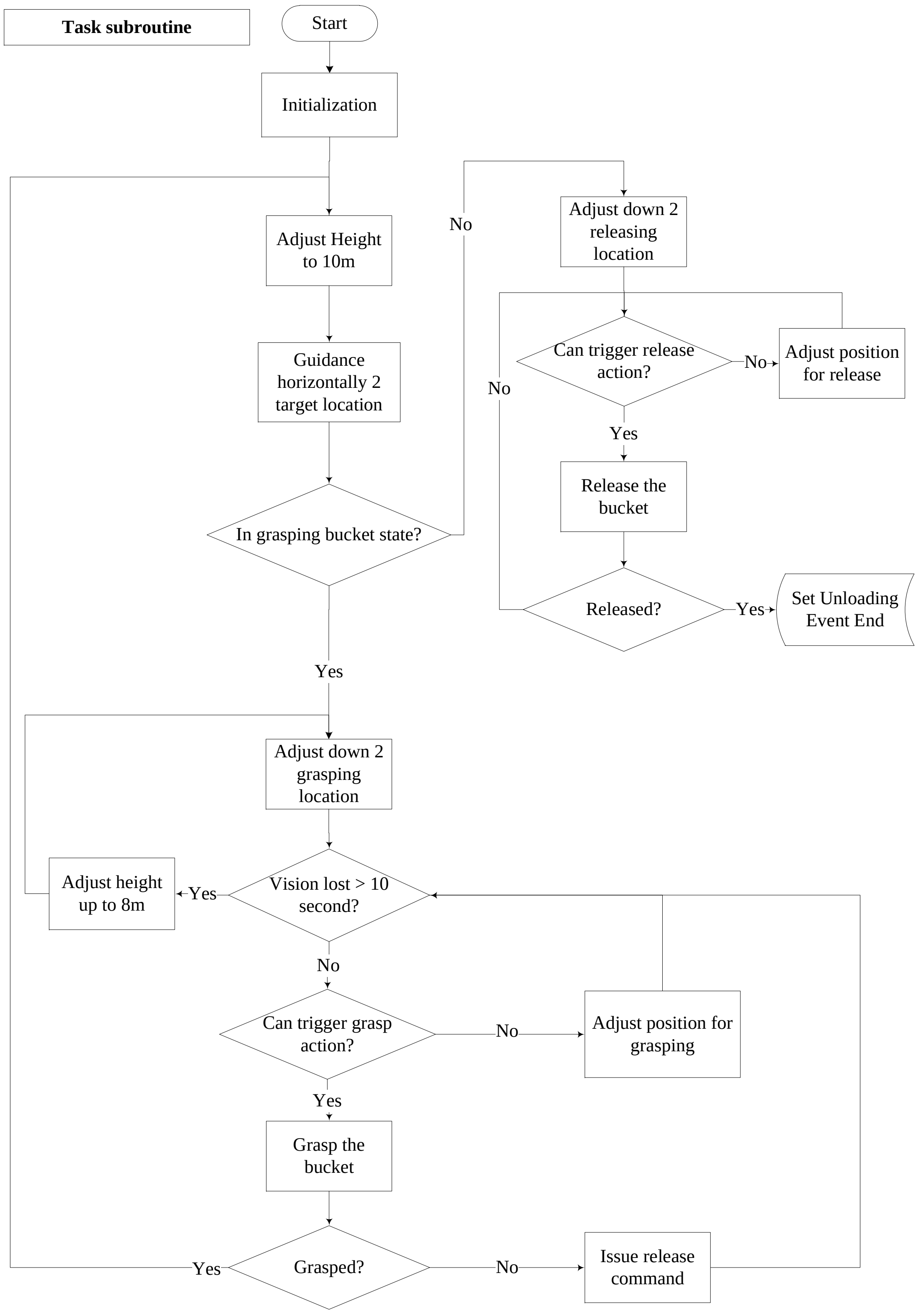

Task subroutine
Start
Initialization
Adjust Height to 10m
Guidance horizontally 2 target location
In grasping bucket state?
No
Adjust down 2 releasing location
Can trigger release action?
No
Adjust position for release
No
Yes
Release the bucket
Released?
Yes
Set Unloading Event End
Yes
Adjust down 2 grasping location
Vision lost > 10 second?
Yes
Adjust height up to 8m
No
Can trigger grasp action?
No
Adjust position for grasping
Yes
Grasp the bucket
Grasped?
Yes
No
Issue release command

Figure 7.11: Task routine

perform proper (vision) initialization. It will first check whether it can detect all the target circles. If the vision system can detect enough target circles (such that the vision system can distinguish the two moving platforms) successfully, the vision system will finish initialization and the decision making module will schedule the software entering the task routine. If the vision system cannot detect enough target circles, a real-time path planning mechanism as shown in Fig. 7.10 will be triggered. As long as the vision system cannot finish the initialization, it will continuously (in 5 Hz) provide the unmanned system a distance vector (i.e., relative distance vector between the unmanned system and the moving platform in ship heading coordinate frame). The unmanned system will then employ the trajectory generator to re-plan its flight trajectories per second. The trajectories are fed to the mode-2 flight controller every 20 ms (i.e., 50 Hz) such that the unmanned system will be guided to a location with good sight views (i.e., the vision system can detect enough target circles).

4. The task routine as shown in Fig. 7.11 is the most important part of the decision making module. It is responsible for commanding the unmanned system to finish the task, which is delivering the cargo from one ship to another ship autonomously. The overall logic consists of two parts, which are commanding the unmanned system to the target area for grabbing/releasing cargo and commanding the unmanned system to grab or release the cargo.

   In this part, the paper will focus on the logic flow about how to finish the task. The basic functionality about how to guide the unmanned system based on the information provided by the vision system will be briefly introduced. The movements of the unmanned system in this part are based on the real-time path planning module as shown in Fig. 7.10. The information perception layer will continuously (i.e., in 5 Hz) provide the distance vector (i.e., from the unmanned system to the target point) to the real-time path planning module such that the unmanned system can be guided to the target area as explained in the previous sections. For example, at $t = 1$ s, the information perception layer provides a distance vector, $\Delta\mathbf{d}^{\mathrm{S}} = [3\ 4\ 0]$ m, to the decision making module, the decision making module will employ the trajectory generator to generate a trajectory such that the unmanned system can move forward 3 m along the x-axis and 4 m along the y-axis from its current location. The unmanned system repeats the procedure every second by employing the real-time path planning module as shown in Fig. 7.10. This kind of guidance functionality provided by the vision system are presented during the task stage. The above mentioned mechanism will guide the unmanned system to the required target location, which employs the mode-2 flight controller to complete these tasks.

Fig. 7.11 shows that the unmanned system will first adjust itself to the height at 10 m such that the unmanned system can provide a good sight of view for the vision system. As the vision system provides a reliable guidance distance vector (between the unmanned system and the target, i.e., cargo or unloading area), the decision making module employs the real-time path planning module as mentioned above to guide the unmanned system to the target location. The position for the heave direction will be kept constant during this stage. The unmanned system will further adjust down to the grabbing or unloading height as it reaches the horizontal position of the target, after which the unmanned system will enter the “grabbing/releasing” phase.

During the “grabbing” phase, the decision making module will first check whether the vision system has lost the target for more than 10 s. The unmanned system will move up to a height position at 8 m if the vision system has lost the target, after which it will move down to the grabbing position again. The assumption here is that the vision system will “re-track” the target as the unmanned system moves up, since it will have a broader sight of view. If the vision system performs well (i.e., no tracked target lost), it will check whether the unmanned system has moved to the “action” region, which is defined as a 3-D window (i.e., the distance vector between the unmanned system and the target lies within $[\,-15\text{ cm} < x < 15\text{ cm}, -15\text{ cm} < y < 15\text{ cm}$ and $5\text{ cm} < z < 15\text{ cm}\,]$). Once the unmanned system enters this region, the decision making module will issue a “grabbing” command to the actuator (i.e., the grabber) to grab the cargo. After about 1 second of the actuation, the unmanned system will check whether the cargo has been grabbed successfully. It will adjust the height to 10 m if it senses the cargo has been successfully grabbed by checking the information provided by the limit switch sensors. However, if it senses the cargo has not been grabbed successfully, it will command the grabber to unlock and repeat the above grabbing procedure again.

For the “releasing” phase, the procedure is similar to the “grabbing” phase as shown in Fig. 7.11. The decision making module will issue a ”Set Unloading Event End” signal after the cargo has been unloaded successfully. It will then check whether there is any other remaining cargoes for delivering. The above procedures will be repeated if there is extra remaining cargoes. Otherwise, the returning-home routine will be activated.

5. After the unmanned system has finished its tasks or the overall planned time is running out, the returning home routine will be triggered. The decision making module takes the current state as the initial state and employs the trajectory generator to generate a trajectory back to the home location (i.e., $\mathbf{Pos}^{\mathrm{N}} = [0\ 0\ -10]$ m). The mode-1 flight controller is used to regulate the unmanned system to follow the trajectory towards its home location.

6. The landing routine will be triggered as the unmanned system arrives at its home location. The procedure for the landing routine is similar to the taking-off routine. The decision making module employs the mode-1 flight controller to navigate the unmanned system moving downwards with a constant speed at 0.5 m/s (if $\mathbf{Pos}^{\mathrm{N}}_{\mathrm{h}\,\mathrm{z}} < -5$ m) or 0.2 m/s (if $\mathbf{Pos}^{\mathrm{N}}_{\mathrm{h}\,\mathrm{z}} > -5$ m). Once the unmanned system reaches near the ground at about 8 cm, the engine of the unmanned system will be shutdown automatically to finish the landing procedure as well as the whole mission.

7. Besides the above mentioned routines, a timer interrupt routine is also implemented as shown in Fig. 7.9. The interrupt routine will be triggered once the overall planned time is running out. It is used to prevent the unmanned system staying at the task site too long once it cannot finish the scheduled tasks due to unexpected situations. The interrupt cannot be triggered in the taking-off, releasing the bucket, returning home and landing stages for safety purpose.

7.6 Experiment set-up and performance evaluations

7.6.1 Experiment set-up

The 2nd AVIC Cup – International UAV Innovation Grand Prix is organized by the Aviation Industry Corporation of China and Chinese Society of Aeronautics and Astronautics. The competition program consists of two main categories, i.e., the athletics grand prix category and the creativity grand prix category. The athletics grand prix category consists of the fixed-wing sub-category, the rotorcraft sub-category and the model aircraft sub-category.

The NUS UAV Research Team took participant in the rotorcraft athletics grand prix category. The developed system for vertical replenishment was successfully verified in the competition.

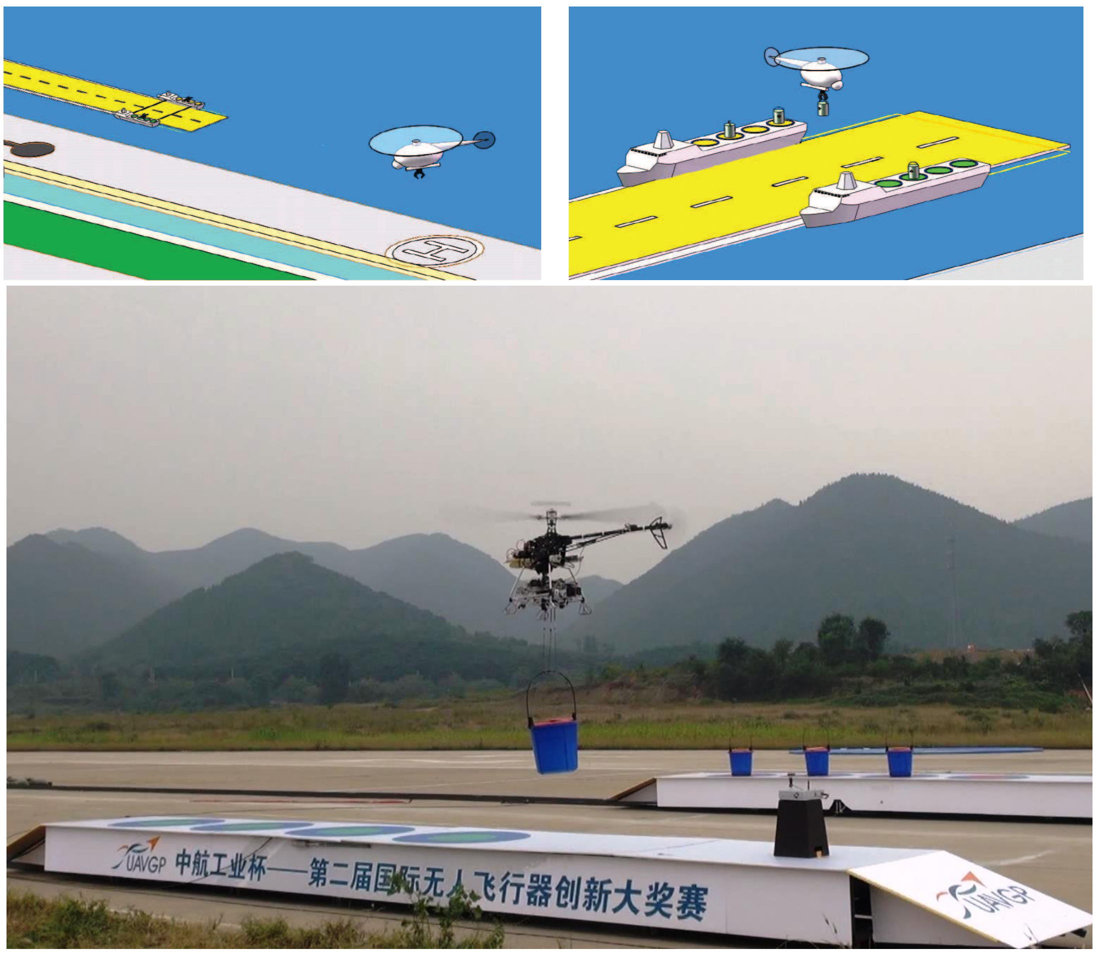


Figure 7.12: Competition field demonstration

Fig. 7.12 illustrates the details about the competition requirements. There are two platforms, say Ship A and Ship B, used to simulate the seaborne vessels. They move concurrently along an 80 meters' long track with a maximum speed at 1 m/s. The moving platforms turn back once they reach the end of the track. There are four circles with the diameter as 1 m on each platform. Four cargoes (buckets filled with 1.5 kg sand) are located inside the four circles respectively on Ship A. The unmanned helicopter needs to deliver the four cargoes to Ship B autonomously. The home location of the helicopter is about 50 m far from the nearest end of the track. The helicopter is required to handle all the required tasks, which are automatic taking-off, moving platforms tracking, automatic cargo identifying, grabbing, delivering and unloading, returning and automatic landing. No human intervention is allowed after the helicopter takes off automatically. The technical solutions will be scored according to the flight control performances, the number of transported cargoes, the cargo stacking precisions and the overall time consumption from taking-off to landing.

7.6.2 Performance evaluations

In preparation for the UAVGP competition, numerous flight tests have been carried out to verify the overall solution and to tune for the optimal performance. Figs. 7.14–7.16 show the position data logged in one of the flight tests. As the raw data is obtained by GPS/INS and then converted to the ship frame, it may not be the ground truth. However, it still shows the control performance in a general sense and indicates whether the UAV is doing the correct movement. In Fig. 7.14, the x position signal becomes larger progressively because the UAV is moving from the first bucket to the fourth bucket. It always comes back to a position around zero because the reference path is purposed defined in a way that the onboard camera has the best view of the two ships before every loading or unloading dive. In Fig. 7.15, the y position signal goes back and forth, indicating alternative movements between the two ships. In Fig. 7.16, it is clear to see all the diving motions of the UAV. The UAV will stay at a very low altitude with a variable time duration depends on how many loading or unloading trials have been performed until the final success one.

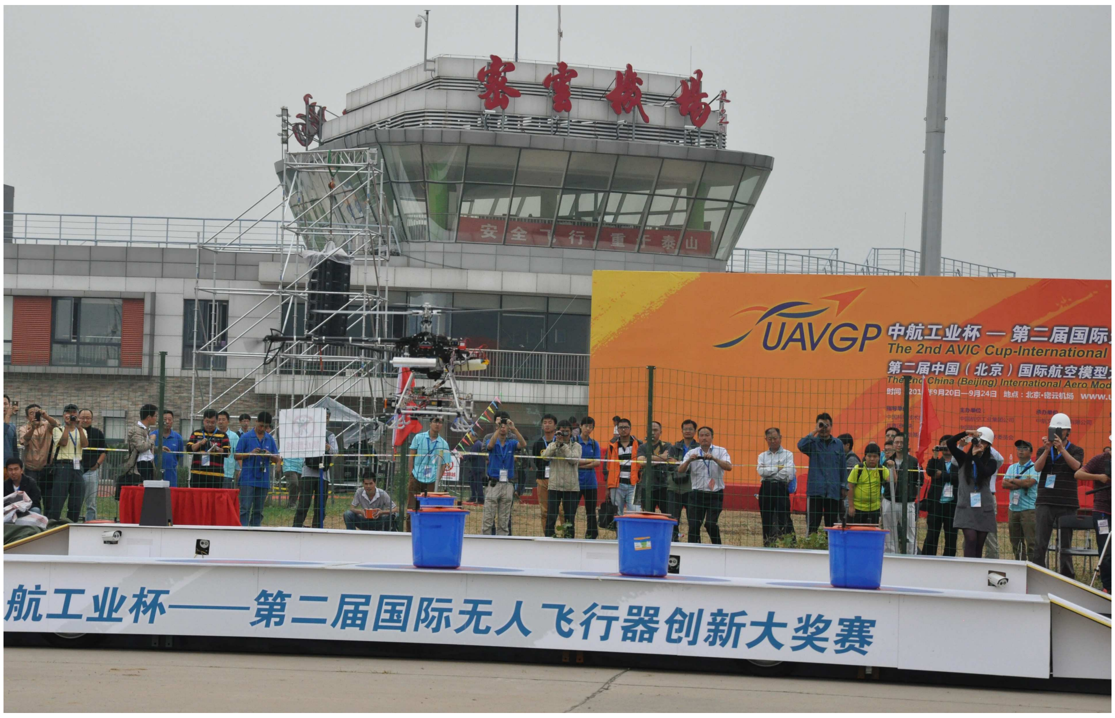


Figure 7.13: NUS^{2}T-Lion in the International UAV Innovation Grand Prix

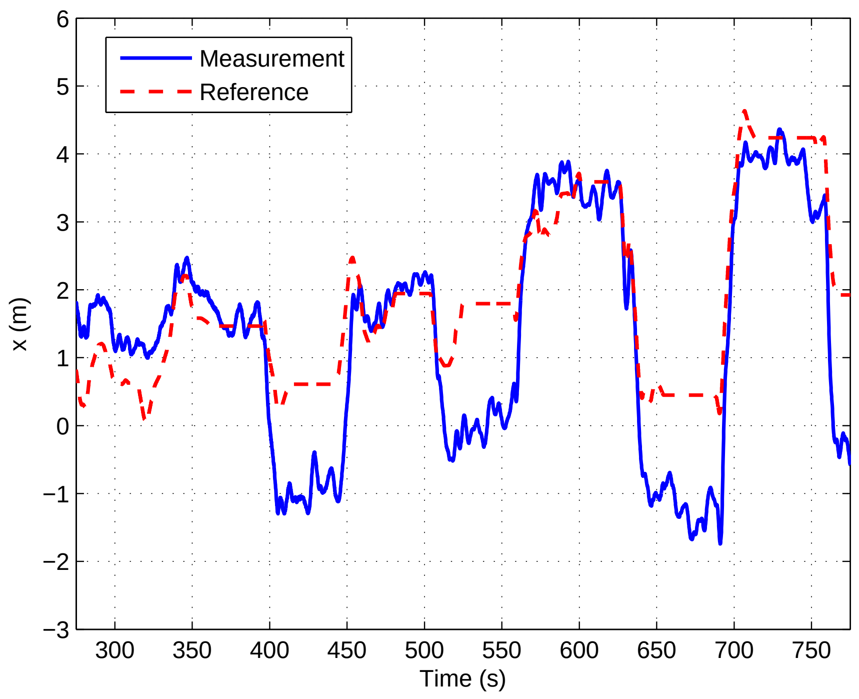


Figure 7.14: UAV position response in the ship-frame *x*-axis

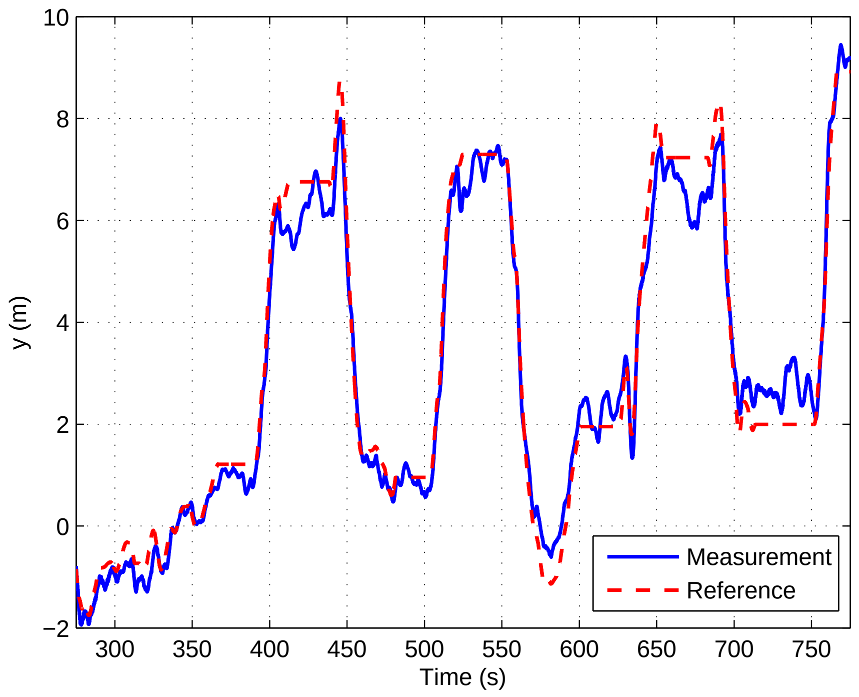


Figure 7.15: UAV position response in the ship-frame *y*-axis

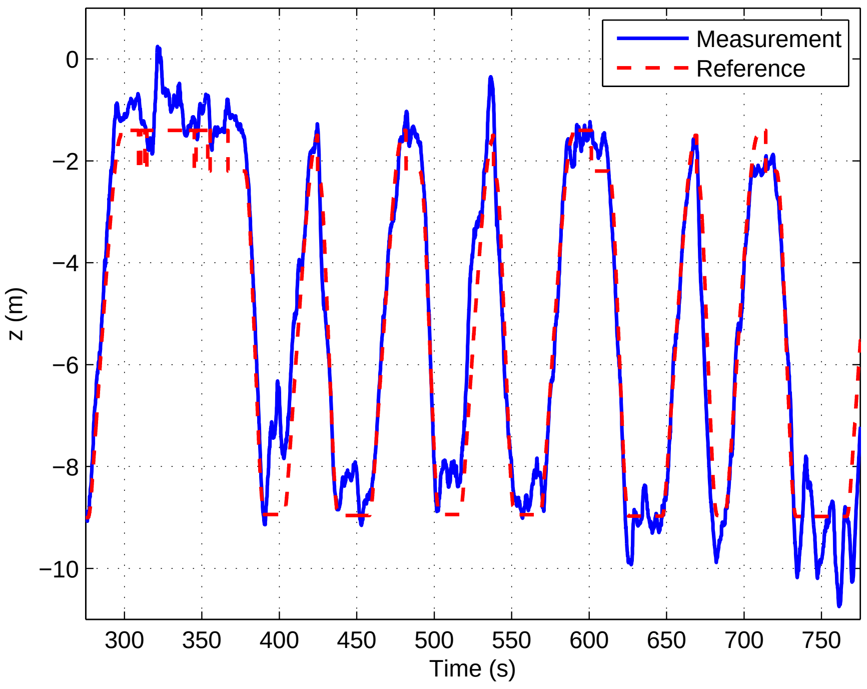


Figure 7.16: UAV position response in the NED-frame z-axis

With this kind of performance, NUS^{2}T-Lion has successfully accomplished the competition tasks in the UAVGP rotary-wing category. A final score of 1127.56 with 472.44 from the preliminary contest and 655.13 from the final has made the team second position in the overall Grand Prix. In fact, 655.13 is the highest score in the final round of the competition. It should be highlighted that unlike the preliminary contest, the final round of the competition requires the UAV to carry out the cargo transportation task with the 'ships' moving. This demands for better robustness and higher intelligence from the participants' UAV systems, and it is indeed the strongest point of the solution proposed in this thesis. Fig. 7.13 shows a snap shot of NUS^{2}T-Lion going to grab the second bucket in this competition. The full process has been video-recorded and uploaded to [48] and [49] for the English and Chinese versions respectively.

7.7 Conclusion

In this chapter, we present the methods used for the integrations of the functional blocks developed in preceding chapters. The whole system is divided into five layers, which include the hardware layer, state estimation and perception layer, control layer, decision making layer and the communication layer. The interactions among these layers as well as their contained blocks are presented. It is these interactions integrate the layers/blocks together to make up a fully functional system for vertical replenishment.

The decision making block is explained in detail in this section. It behaves as the central coordinator among the blocks. The ship-frame navigation, vision based guidance, real-time path planning are addressed in details.

To verify the functionalities and robustness of the system, the helicopter is brought to take participant an international competition. The competition performance of our developed system shows that the system is capable and robust for the vertical replenishment problem.

Chapter 8

Conclusion and Future Works

8.1 Conclusion

This thesis presents a systematic approach used to develop an unmanned helicopter for vertical replenishment.

One main contribution of this thesis is that we successfully solved the precision cargo grabbing problem. This problem is the key and most difficult part of the system. The localization accuracy of our used GPS device is around 2.5 m, which is far insufficient for the helicopter to grab the cargo precisely. Thus, in this thesis, a height estimation algorithm based on 2D laser scanner is developed for precision height control; a vision-guidance algorithm is also developed for cargo precise localization. By fusing the measurements from inertial measurement, GPS, laser scanner, and camera, the developed system can grab the cargo precisely and robustly.

Another main contribution is that the developed helicopter is capable to finish the vertical replenishment task fully autonomously (without any human intervention). It is an important characteristic for robots toward autonomy. The function is achieved through the implementation of a decision making module for the helicopter. The decision making module collects all the necessary information from other modules and make a event-based decision for the helicopter. A flowchart of procedures is used to implement this module. Thorough ground and flight experiments have been conducted to evaluate the system.

Some insufficiencies are also shown by this system. For example, the helicopter did not know how to react properly after he accidently dropped one cargo during the competition. The helicopter wasted lots of time waiting there for the cargo. It did not know that he should return home or search the cargo around the lost location.

8.2 Future works

The developed system is still a prototype. It is a long way to go towards the full autonomy of the unmanned helicopter for vertical replenishment. Thus, I summarize here some future works need to be done based on the knowledge I have.

1. The **hardware platform** can be further optimized to be smaller and lighter. For example, the size of the auto-pilot, the supporting plate, and the anti-vibration damper can be further reduced. The reduction of the size and weight of the avionics will increase the flight endurance and introduce extra payload for the helicopter.

2. The **mechanical manipulator** can be further optimized; During the competition, one of the grabbed cargo was dropped off unexpectedly due to the mechanical failure of the manipulator.

3. The **decision making module** of the system also needs further improvement as mentioned in previous section; Algorithms developed for artificial intelligence can be incorporated, such as automata theory, probabilistic reasoning, etc.

4. The **vision-based perception** algorithm of the system also needs further improvement; The current developed vision perception algorithm is very sensitive to sun light conditions; The threshold used for image segmentation needs human tuning before every flight, which is undesirable towards the full autonomy of the helicopter; A marker-less algorithm is expected to be developed in future.

Bibliography

[1] P. Abbeel, A. Coates, and A. Ng. Autonomous helicopter aerobatics through apprenticeship learning. *International Journal of Robotics Research*, 29(13):1608–1639, 2010.

[2] S. J. Ahn, W. Rauh, and H.-J. Warnecke. Least-squares orthogonal distances fitting of circle, sphere, ellipse, hyperbola, and parabola. *Pattern Recognition*, 34(12):2283–2303, 2001.

[3] J. G. Allen, R. Y. D. Xu, and J. S. Jin. Object tracking using CamShift algorithm and multiple quantized feature spaces. In *Proceedings of the Pan-Sydney area workshop on Visual information processing*, 2004.

[4] D. H. Ballard. Generalizing the hough transform to detect arbitrary shapes. *Pattern Recognition*, 13(2):111–122, 1981.

[5] D. B. Barber, S. R. Griffiths, T. W. Mclain, and R. W. Beard. Autonomous landing of miniature aerial vehicles. *AIAA Journal of Aerospace Computing, Information and Communication*, 4(5):770–784, 2007.

[6] M. Bergerman, O. Amidi, J. Miller, N. Vallidis, and T. Dudek. Cascaded position and heading control of a robotic helicopter. In *Proceedings of the 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2007.

[7] M. Bernard and K. Kondak. Generic slung load transportation system using small size helicopters. In *In Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3258–3264, 2009.

[8] M. Bernard, K. Kondak, I. Maza, and A. Ollero. Autonomous transportation and deployment with aerial robots for search and rescue missions. *Journal of Field Robotics*, 28(6):914–931, 2011.

[9] M. Bisgaard, A. la Cour-Harbo, and J. D. Bendtsen. Adaptive control system for autonomous helicopter slung load operations. *Control Engineering Practice*, 18(7):800–811, 2010.

[10] G. Borges. A split-and-merge segmentation algorithm for line extraction in 2-d range images. In *Proceedings of the International Conference on Pattern Recognition*, volume 1, 2000.

[11] A. Bujak, M. Smolarek, and A. Gebczyǹska. Applying military telematic solutions for logistics purposes. In *11th International Conference on Transport Systems Telematics*, pages 248–256, 2011.

[12] G. Cai, B. M. Chen, and T. H. Lee. *Unmanned Rotorcraft Systems*. Springer, New York, 2011.

[13] G. Cai, B. Wang, B. M. Chen, and T. H. Lee. Design and implementation of a flight control system for an unmanned rotorcraft using rpt control approach. *Asian Journal of Control*, 15(1):95–119, 2013.

[14] B. M. Chen. *Robust and H_∞ Control*. Springer, New York, 2000.

[15] B. M. Chen. Lecture notes on multivariable control systems. `http://vlab.ee.nus.edu.sg/bmchen/`, June 2015.

[16] B. M. Chen, T. H. Lee, and V. Venkataramanan. *Hard Disk Drive Servo Systems*. Advances in Industrial Control Series. Springer, New York, 2002.

[17] B. M. Chen, Z. L. Lin, and Y. Shamash. *Linear Systems Theory - A Structural Decomposition Approach*. Birkhiiuser Boston, 2004.

[18] V. N. Dobrokhodov, I. I. Kaminer, K. D. Jones, and R. Ghabcheloo. Vision-based tracking and motion estimation for moving targets using small UAVs. In *American Control Conference*, 2006.

[19] X. Dong, B. M. Chen, G. Cai, H. Lin, and T. H. Lee. A comprehensive real-time software system for flight coordination and cooperative control of multiple unmanned aerial vehicles. *International Journal of Robotics and Automation*, 26(1):49–63, 2011.

[20] D. Eberli, D. Scaramuzza, S. Weiss, and R. Siegwart. Vision based position control for MAVs using one single circular landmark. *Journal of Intelligent and Robotic Systems*, 61(1-4):495–512, 2011.

[21] A. Fitzgibbon, M. Pilu, and R. B. Fisher. Direct least square fitting of ellipses. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(5):476–480, 1999.

[22] J. Flusser and T. Suk. Pattern recognition by affine moment invariants. *Pattern Recognition*, 21(1):167–174, 1993.

[23] R. Garcia and K. Valavanis. The implementation of an autonomous helicopter testbed. *Journal of Intelligent and Robotics Systems*, 54(1-3):423–454, 2009.

[24] Google Inc. Reflexxes Motion Libraries. `http://www.reflexxes.ws/products/overview-and-download.html`, November 2013.

[25] R. Heffley and M. Mnich. Minimum-complexity helicopter simulation math model. Technical report, NASA Contractor Report, 1988.

[26] J. Heikkila and O. Silven. A four-step camera calibration procedure with implicit image correction. In *Proceedings of the 1997 IEEE Conference on Computer Vision and Pattern Recognition*, 1997.

[27] G. Jiang and L. Quan. Detection of concentric circles for camera calibration. In *Proceedings of the 10th IEEE International Conference on Computer Vision*, 2005.

[28] F. Kendoul. Survey of advances in guidance, navigation, and control of unmanned rotorcraft systems. *Journal of Field Robotics*, 29(2):315–378, 2012.

[29] J.-S. Kim, P. Gurdjos, and I.-S. Kweon. Geometric and algebraic constraints of projected concentric circles and their applications to camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(4):637–642, 2005.

[30] K. Kosuge and M. Sato. Transportation of a single object by multiple decentralized-controlled nonholonomic mobile robots. In *IEEE International Conference on Intelligent Robots and Systems*, pages 1681–1686, 1999.

[31] H. Lang, M. T. Khan, K.-K. Tan, and C. W. de Silva. Developments in visual servoing for mobile manipulation. *Unmanned Systems*, 1(1):143–162, 2013.

[32] J. G. Leishman. *Principles of Helicopter Aerodynamics*. Cambridge University Press, 2000.

[33] F. Lin, K. Z. Y. Ang, F. Wang, B. M. Chen, T. H. Lee, B. Yang, M. Dong, X. Dong, J. Cui, S. K. Phang, B. Wang, D. Luo, K. Peng, G. Cai, S. Zhao, M. Yin, and K. Li. Development of an unmanned coaxial rotorcraft for the DARPA UAVForge Challenge. *Unmanned Systems*, 01(02):17–34, 2013.

[34] F. Lin, X. Dong, B. M. Chen, K. Y. Lum, and T. H. Lee. A robust real-time embedded vision system on an unmanned rotorcraft for ground target following. *IEEE Transactions on Industrial Electronics*, 59(2):1038–1049, 2012.

[35] Q. Lindsey, D. Mellinger, and V. Kumar. Construction of cubic structures with quadrotor teams. In *Robotics Science and Systems*, 2011.

[36] I. Maza, K. Kondak, M. Bernard, and A. Ollero. Multi-UAV cooperation and control for load transportation and deployment. *Journal of Intelligent and Robotic Systems*, 57(1-4):417–449, 2010.

[37] R. A. McLaughlin. Randomized hough transform: Improved ellipse detection with comparison. *Pattern Recognition Letters*, 19(3-4):299–305, 1998.

[38] D. Mellinger and V. Kumar. Minimum snap trajectory generation and control for quadrotors. In *In Proceedings of the IEEE International Conference on Robotics and Automation*, Shanghai, China, 2011.

[39] B. Mettler. *Identification Modeling and Characteristics of Miniature Rotorcraft*. Kluwer Academic Publishers, 2003.

[40] B. K. G. Mrdjan Jankovic. Visually guided ranging from observations of points, lines, and curves via identifier based nonlinear observer. *System & Control Letter*, 25(1):63–73, 1995.

[41] K. Nonami, F. Kendoul, S. Suzuki, and W. Wang. *Autonomous flying robots: Unmanned aerial vehicles and micro air vehicles*. Tokyo: Springer-Verlag, 2010.

[42] M. Orsag, C. Korpela, and P. Oh. Modeling and control of mm-uav: Mobile manipulating unmanned aerial vehicle. *Journal of Intelligent & Robotic Systems*, 69(1-4):227–240, 2013.

[43] K. Peng, G. Cai, B. M. Chen, M. Dong, K. Y. Luma, and T. H. Lee. Design and implementation of an autonomous flight control law for a uav helicopter. *Automatica*, 45(10):2333–2338, 2009.

[44] R. Prouty. *Helicopter Performance, Stability and Control*. Krieger Publishing Company, 1986.

[45] S. Saripalli, J. F. Montgomery, and G. Sukhatme. Vision-based autonomous landing of an unmanned aerial vehicle. In *IEEE International Conference on Robotics and Automation*, 2002.

[46] C. R. Theodore, M. B. Tischler, and J. D. Colbourne. Rapid frequency-domain modeling methods for unmanned aerial vehicle flight control applications. *Journal of Aircraft*, 41(4):735–743, 2004.

[47] J. Thomas, J. Polin, K. Sreenath, and V. Kumar. Avian-inspired grasping for quadrotor micro uavs. In *IDETC/CIE*. ASME, 2013.

[48] Unmanned Aerial Vehicle Research Group, National University of Singapore. The video of the final round competition of UAVGP (Chinese version). `http://v.youku.com/v_show/id_XNjI5MTM2MDI0.html`, 2013.

[49] Unmanned Aerial Vehicle Research Group, National University of Singapore. The video of the final round competition of UAVGP (English version). `http://www.youtube.com/watch?v=Ex0jgtX2ZrY`, 2013.

[50] P. Vela, A. Bester, J. Malcolm, and A. Tannenbaum. Vision-based range regulation of a leader follwer formation. *IEEE Transaction on Control Systems Technology*, 17(2):442–448, 2009.

[51] G. Wang, J. Wu, and Z. Ji. Single view based pose estimation from circle or parallel lines. *Pattern Recognition Letters*, 29(7):977–985, 2008.

[52] A. D. Wu, E. N. Johnson, and A. A. Proctor. Vision-aided inertial navigation for flight control. *AIAA Journal of Aerospace Computing, Information and Communication*, 2(9):348–360, 2005.

[53] S. Zhao, Z. Hu, M. Yin, K. Ang, P. Liu, F. Wang, X. Dong, F. Lin, B. M. Chen, and T. H. Lee. A robust real-time vision system for autonomous cargo transfer by an unmanned helicopter. *Industrial Electronics, IEEE Transactions on*, 62(2):1210–1219, 2014.

[54] S. Zhao, F. Lin, K. Peng, B. M. Chen, and T. H. Lee. Homography-based vision-aided inertial navigation of uavs in unknown environments. In *AIAA Guidance, Navigation and Control Conference*, 2012.

[55] S. Zhao, F. Lin, K. Peng, B. M. Chen, and T. H. Lee. Distributed control of angle-constrained circular formation using bearing-only measurements. *Systems and Control Letters*, 63(1):12–24, 2014.

List of Publications

Journal Articles:

1. F. Wang, P. Liu, S. Zhao, B.M. Chen, T. H. Lee, S. K. Phang, S. Lai, T. Pang, et al., Development of an unmanned rotorcraft system for the International UAV Innovation Grand Prix, *Unmanned Systems*, Vol. 3, No. 1, pp. 63-87, 2015.

2. S. Zhao, Z. Hu, M. Yin, K. Z. Y. Ang, P. Liu, F. Wang, X. Dong, etc., A Robust Real-time Vision System for Autonomous Cargo Transfer by an Unmanned Helicopter, *IEEE Transactions on Industrial Electronics*, Vol. 62, No. 2, pp. 1210-1219, 2015.

Conference Papers:

1. K. Li, P. Liu, T. Pang, Z. Yang and B.M. Chen, Development of an Unmanned Aerial Vehicle for Rooftop Landing and Surveillance, International Conference on Unmanned Aircraft Systems, Denver, USA, 2015.

2. P. Liu, X. Dong, F. Wang and B.M. Chen, Development of a comprehensive software system for cargo transportation by unmanned helicopters, International Conference on Intelligent Unmanned Systems (ICIUS), Montreal, Canada, 2014.

3. F. Wang, P. Liu, S. Zhao, B.M. Chen, etc., Guidance, navigation and control of an unmanned helicopter for automatic cargo transportation, 33rd Chinese Control Conference (CCC), Nanjing, China, pp. 1013-1020, 2014.

4. S. Zhao, Z. Hu, M. Yin, K. Z. Y. Ang, P. Liu, F. Wang, X. Dong, etc., A robust vision system for a uav transporting cargoes between moving platforms, 33rd Chinese Control Conference (CCC), Nanjing, China, pp. 544-549, 2014.

5. J. Q. Cui, S. Lai, X. Dong, P. Liu, B.M. Chen, T. H. Lee, Autonomous navigation of UAV in forest, International Conference on Unmanned Aircraft Systems (ICUAS), Orlando, USA, pp. 726-733, 2014.

6. P. Liu, X. Dong, B.M. Chen, T. H. Lee, Development of an enhanced ground control system for unmanned aerial vehicles, Proceedings of the IASTED International Conference on Engineering and Applied Science, Colombo, Sri Lanka, pp. 136-143, 2013.