AUTONOMOUS LANDING OF A VTOL UAV ON A SHIP DECK MOVING ON STANDING WAVY SEA USING CAMERA AND LASER RANGE FINDER

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

ALİ MANCAR

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
ELECTRICAL AND ELECTRONICS ENGINEERING

JANUARY 2024

Approval of the thesis:

AUTONOMOUS LANDING OF A VTOL UAV ON A SHIP DECK MOVING ON STANDING WAVY SEA USING CAMERA AND LASER RANGE FINDER

submitted by **ALİ MANCAR** in partial fulfillment of the requirements for the degree of **Master of Science** i**n Electrical and Electronics Engineering, Middle East Technical University** by,

Prof. Dr. Halil Kalıpçılar
Dean, **Graduate School of Natural and Applied Sciences**

Prof. Dr. İlkay Ulusoy
Head of Department, **Electrical and Electronics Engineering**

Prof. Dr. Mehmet Kemal Leblebicioğlu
Supervisor, **Electrical and Electronics Engineering, METU**

Examining Committee Members:

Prof. Dr. Ozan Tekinalp
Aerospace Engineering Dept., METU

Prof. Dr. Mehmet Kemal Leblebicioğlu
Electrical and Electronics Engineering Dept., METU

Prof. Dr. Asım Egemen Yılmaz
Electrical and Electronics Engineering Dept., Ankara Uni.

Date: 24.01.2024

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last name: Ali Mancar

Signature:

ABSTRACT

AUTONOMOUS LANDING OF A VTOL UAV ON A SHIP DECK MOVING ON STANDING WAVY SEA USING CAMERA AND LASER RANGE FINDER

Mancar, Ali
Master of Science, Electrical and Electronics Engineering
Supervisor: Prof. Dr. Mehmet Kemal LEBLEBİCİOĞLU

January 2024, 90 pages

This thesis investigates the development of an autonomous landing system for Vertical Takeoff and Landing (VTOL) Unmanned Aerial Vehicles (UAV) on ship decks in wavy sea conditions. This system aligns the UAV's movements with the ship's motion. It focuses on methodologies for precise and smooth landing using low computational cost image processing of AprilTag for landing point detection, and a descent rate control algorithm utilizing a laser rangefinder. Proportional – Integral – Derivative (PID) controllers are employed for horizontal position and vertical velocity control. The results demonstrate successful, precise, and smooth UAV landings on moving and oscillating ship decks, enhancing autonomous UAV operations in maritime environments. Simulation results are verified by some physical experiments.

Keywords: Vertical Takeoff and Landing Unmanned Aerial Vehicle (VTOL UAV), Quadplane, Autonomous Ship Landing, Proportional – Integral – Derivative (PID), AprilTag

ÖZ

KAMERAYLA VE LAZER MESAFE ÖLÇERLE BIR VTOL İHA'NIN SABİT DALGALI DENİZDE HAREKET EDEN BİR GEMİ GÜVERTESİNE OTONOM İNİŞİ

Mancar, Ali
Yüksek Lisans, Elektrik ve Elektronik Mühendisliği
Tez Yöneticisi: Prof. Dr. Mehmet Kemal LEBLEBİCİOĞLU

Ocak 2024, 90 sayfa

Bu tez, sabit dalgalı deniz koşullarında gemi güvertelerinde Dikey İniş Kalkışlı İnsansız Hava Araçları (DİHA) için otonom bir iniş sisteminin geliştirilmesini araştırmaktadır. Bu sistem, DİHA'nın hareketlerini geminin hareketiyle uyumlu hale getirir. Çalışmada düşük hesaplama maliyetli görüntü işleme ve AprilTag kullanarak iniş noktası tespiti ve lazer mesafe bulucu kullanan bir iniş hızı kontrol algoritması üzerine odaklanılmaktadır. PID kontrolcüleri yatay konum ve dikey hız kontrolü için kullanılmaktadır. Sonuçlar, hareketli ve salınan gemi güvertelerinde başarılı, hassas ve sarsıntısız DİHA inişlerini göstermektedir, hareketli deniz ortamlarında otonom İHA operasyonlarına imkan sağlamaktadır. Benzetim sonuçları bazı fiziksel deneylerle de doğrulanmıştır.

Anahtar Kelimeler: Dikey İniş Kalkışlı İnsansız Hava Aracı (DİHA), Gemi Güvertesine Otonom İniş, PID, AprilTag

To my beloved wife & daughter,

your smiles inspired every word.

ACKNOWLEDGMENTS

I would like to announce my deepest gratitude to my supervisor Prof. Dr. Mehmet Kemal Leblebicioğlu for his valuable guidance, advice, and support throughout the research.

I am also grateful to my thesis committee members Prof. Dr. Ozan Tekinalp and Prof. Dr. Asım Egemen Yılmaz for their criticism and advice.

I wish to express my endless thanks to my whole family, especially to my wife Dr. Melis Kesik Mancar for her endless support, kindness and love, my daughter for being sunshine of my life, my parents Aysun Mancar and K. İrfan Mancar for their lifetime supports, my brothers Mert Mancar and Halit Mancar for growing up together.

I am deeply grateful to Dr. Ferit Çakıcı, my mentor, for technical and all other support.

The technical assistance of Berkay Ayhan, my colleague, is gratefully acknowledged.

Finally, I would like to extend my heartfelt thanks to Alper Tosun, my manager, for invaluable support. His expertise, insights, and encouragement have been instrumental in shaping my research and have significantly contributed to my personal and professional growth.

TABLE OF CONTENTS

ABSTRACT v

ÖZ vi

ACKNOWLEDGMENTS viii

TABLE OF CONTENTS ix

LIST OF FIGURES xii

LIST OF ABBREVIATIONS xv

1 INTRODUCTION 1

1.1 Motivation 1

1.2 Problem Definition and Goals 2

1.3 Outline of the Thesis 3

2 LITERATURE REVIEW 5

2.1 VTOL Fixed Wing Planes (Quadplanes) 5

2.2 Vision Based Localization 8

2.3 Quadplane Ship Landing 13

3 METHODOLOGY 17

3.1 System Architecture 17

3.1.1 VTOL Fixed Wing UAV Platform (Quadplane) 18

3.1.2 Flight Controller Unit (FCU) 20

3.1.3 Companion Computer 22

3.1.4 Sensors 23

3.1.5 Beacon 26
3.1.6 Communication Links 26
3.1.7 Ground Control Station (GCS) 26
3.2 Control Algorithms 28
3.2.1 Quadplane Ship Landing 29
3.2.2 Vision Based Position Control 31
3.2.3 Descent Rate Control 38
4 SIMULATIONS 43
4.1 Simulation Environment 43
4.1.1 Gazebo Simulator 44
4.1.2 ArduPilot Software in the Loop (SITL) 51
4.2 Control Algorithms 53
4.2.1 Horizontal Position Control 53
4.2.2 Descent Rate Control 54
4.3 Results 54
5 EXPERIMENTS 67
5.1 Experiment Setup 67
5.1.1 Quadplane 67
5.1.2 Ship (Remote Controlled Electric Ride on Truck) 71
5.1.3 GCS 73
5.2 Flights 74
5.2.1 First Flight 74
5.2.2 Second Flight 78
6 CONCLUSION AND FUTURE WORK 81

6.1 Conclusion 81

6.2 Future Work 82

REFERENCES 83

LIST OF FIGURES

FIGURES

Figure 2.1. Control elements of VTOL–FW UAV 6
Figure 2.2. Flight envelope of VTOL–FW UAV platform 7
Figure 2.3. Simplified dataflow of the vision algorithm 9
Figure 2.4. AprilTag based visual localization algorithm 11
Figure 2.5. AprilTag detection 12
Figure 3.1. System architecture 17
Figure 3.2. A quadplane system architecture 19
Figure 3.3. Position and attitude control of a quadplane 20
Figure 3.4. Pixhawk cube orange 21
Figure 3.5. MavProxy 27
Figure 3.6. Landing on a ship deck 30
Figure 3.7. Landing algorithm 31
Figure 3.8. Custom cascaded AprilTag 32
Figure 3.9. Vision based position control algorithm 33
Figure 3.10. AprilTag detection and pose estimation algorithm 34
Figure 3.11. World coordinate system and camera coordinate system 35
Figure 3.12. Rotation of the coordinate system 36
Figure 3.13. PID controller 37
Figure 3.14. Descent rate control algorithm 39
Figure 3.15. Schematic of the closed-loop control system 42
Figure 4.1. Quadplane ship landing simulation architecture 44
Figure 4.2. Gazebo waves control plugin panel 46
Figure 4.3. Havyard vessel model on Gazebo 49
Figure 4.4. Alti transition quadplane model on Gazebo 50
Figure 4.5. First stage of the simulation 55
Figure 4.6. Second stage of the simulation 56

Figure 4.7. Quadplane takeoff from the ship deck 57
Figure 4.8. Quadplane horizontal position correction 58
Figure 4.9. Quadplane landing to the ship deck 59
Figure 4.10. Landing completed success 59
Figure 4.11. Altitudes of the quadplane and ship from the initial simulation 61
Figure 4.12. Z axis acceleration from the initial simulation 61
Figure 4.13. Altitudes of the quadplane and ship from the final simulation 63
Figure 4.14. Z axis acceleration from the final simulation 63
Figure 4.15. Quad mode throttle percentage of the quadplane with altitudes from the initial simulation 64
Figure 4.16. Quad mode throttle percentage of the quadplane with altitudes from the final simulation 65
Figure 5.1. Quadplane conversion of a model airplane (Volantex FPV Raptor V2) 67
Figure 5.2. Quad-propulsion system motor and ESC 68
Figure 5.3. Autopilot hardware (Pixhawk cube orange) integration 69
Figure 5.4. Pitot tubes, airspeed sensors and GPS receiver installed on the quadplane 70
Figure 5.5. 3DR 433MHz radio telemetry with antenna 70
Figure 5.6. Camera, rangefinder and companion computer installations 71
Figure 5.7. RC truck for the ship role 72
Figure 5.8. Proposed AprilTag and beacon (Autopilot and GPS receiver) installation 72
Figure 5.9. GCS setup with telemetry modules 73
Figure 5.10. Camera view from the quadplane 74
Figure 5.11. Camera view of the AprilTag tag detection from the quadplane at 30m height 75
Figure 5.12. Camera view of the outer AprilTag detection from the quadplane at 20m height 76

Figure 5.13. Camera view of the outer and inner AprilTag detection from the quadplane at 10m height 77
Figure 5.14. Camera view of the inner AprilTag detection from the quadplane at 1.3m height 78
Figure 5.15. Quadplane autonomous takeoff and landing to a moving platform 79
Figure 6.1. Tulpars hybrid quadplane 82

LIST OF ABBREVIATIONS

ABBREVIATIONS

VTOL	(Vertical Takeoff and Landing)
UAV	(Unmanned Aerial Vehicle)
FCU	(Flight Controller Unit)
GPS	(Global Positioning System)
IMU	(Inertial Measurement Unit)
IoT	(Internet of Things)
GCS	(Ground Control Station)
PID	(Proportional – Integral – Derivative)
SITL	(Software in the Loop)
FFT	(Fast Fourier Transform)
URDF	(Unified Robot Description Format)
SDF	(Simulation Description Format)
CGAL	(Computational Geometry Algorithms Library)
UUV	(Unmanned Underwater Vehicle)
VMRC	(VMware Remote Console)
GUI	(Graphical User Interface)
FOV	(Field of View)
MSL	(Mean Sea Level)
AGL	(Above Ground Level)

RTL	(Return to Launch)
QRTL	(Quadplane Return to Launch)
ESC	(Electronic Speed Controller)
PWM	(Pulse Width Modulation)
RC	(Remote Control)

CHAPTER 1

INTRODUCTION

1.1 Motivation

Unmanned Aerial Vehicles (UAVs) have witnessed significant advancements in recent years [1, 2] becoming indispensable tools across various industries, from surveillance and monitoring to search and rescue missions. VTOL UAVs offer unique advantages in terms of flexibility and adaptability to various environments [3]. One critical application of VTOL UAVs is the autonomous landing on moving platforms [4, 5], such as trucks and ship decks, which poses a considerable challenge due to the inherent dynamics of both the UAV and the moving platform.

The maritime domain plays a pivotal role [6, 7] in global trade, security, and exploration. Autonomous UAVs capable of landing on moving ship decks can revolutionize maritime operations, enabling tasks ranging from surveillance and reconnaissance to delivering critical payloads in challenging sea conditions. The successful implementation of autonomous landing systems on moving ships requires a sophisticated integration of computer vision, control systems, and precise navigation technologies.

This research is motivated by the pressing need to enhance the capabilities of VTOL UAVs, making them adept at autonomously landing on ship decks that are not only in motion but also subjected to wavy sea conditions. Conventional landing systems struggle to cope with the dynamic and unpredictable nature of such environments, necessitating a novel approach that combines cutting-edge technologies, including cameras and laser range finders, to achieve precision and reliability.

As we embark on this research journey, the goal is not only to address the technical challenges associated with autonomous landings on moving ship decks but also to

pave the way for future innovations in robotics and autonomous systems. By pushing the boundaries of what is currently achievable, this thesis seeks to contribute valuable insights and solutions to the broader field of robotics, ultimately shaping the landscape of autonomous operations in dynamic and unpredictable environments.

In summary, the motivation behind this thesis lies in the revolutionary impact of developing an autonomous precise landing system for VTOL UAVs on moving ship decks in standing wavy seas, with the aim of advancing maritime operations and pushing the boundaries of autonomous robotics in challenging real-world scenarios.

1.2 Problem Definition and Goals

The main problem addressed in this thesis is the development of an autonomous precise landing system for VTOL UAVs on moving ship decks under the influence of wavy sea conditions. This problem is characterized by the dynamic nature of both the landing platform and the sea, posing challenges to conventional landing systems. Both ship and UAV uses GPS for positioning which may have several meters error. Also, the communication link between the vehicles (ship and UAV) and the Ground Control Station (GCS) may experience delays, affecting the real-time responsiveness of the UAV in following the ship. Also, the heaving motion of the ship deck due to waves is the biggest problem to accomplish a smooth landing of UAV.

The primary goals are to achieve precise landing point location and constant descent rate according to the heaving motion of the ship in autonomous landings, thereby enhancing the operational capabilities of VTOL UAVs in maritime environments.

1.3 Outline of the Thesis

The thesis explores the intricacies of developing an autonomous landing system for VTOL UAVs navigating moving ship decks among standing wavy sea conditions. Chapter 2, the literature survey, provides a comprehensive overview of existing autonomous landing systems, underlining their relevance to maritime applications and identifying challenges in dynamic environments.

Moving forward, Chapter 3, Methodology, is structured to address the identified challenges. Section 3.1, System Architecture, establishes the foundational concepts, technologies, and theoretical frameworks crucial for developing the system. Section 3.2 details the Vision Based Position Control Algorithm and Descent Rate Control Algorithm, respectively, outlining the algorithms' structures, components, and functionalities to ensure precision and reliability during autonomous landings.

Chapter 4, Simulations, showcases the outcomes of testing and validation, presenting a comprehensive analysis of the system's performance under diverse maritime conditions and providing insights through comparative assessments. Chapter 5, Conclusion and Future Work, summarizes the achieved milestones, underscores the research's significance in advancing maritime operations, and outlines avenues for future research.

In summary, the outlined research seeks to contribute valuable insights and solutions to the broader field of robotics, pushing the boundaries of what is currently achievable in autonomous operations on moving ship decks in wavy seas.

CHAPTER 2

LITERATURE REVIEW

To achieve precise autonomous landing on a moving ship deck with a quadplane, the literature survey is conducted on three main topics.

2.1 VTOL Fixed Wing Planes (Quadplanes)

A quadplane is a type of hybrid UAV that combines features of both quadcopters and fixed wing airplanes. This hybrid design allows the vehicle to take off and land vertically like a quadcopter, but it can also transition to horizontal flight like a traditional airplane.

The quadplane as described in [8] typically has four rotors, arranged in a quadcopter configuration, for vertical takeoff and landing. Once airborne, it transitions to forward flight by engaging fixed wing propulsion or tilting its rotors or wings to achieve the desired orientation for fixed wing flight. This hybrid design provides the advantages of VTOL capabilities while also allowing for efficient and fast forward flight.

Different manufacturers and research groups may have their own designs and configurations for quadplanes, but the basic principle of combining quadcopter and fixed wing capabilities remains consistent across various models. Çakıcı et al [8] focuses on the design and analysis of a VTOL–FW UAV, integrating multirotor and fixed-wing structures for enhanced flight capabilities. It employs aerodynamic modeling, including post-stall conditions, and utilizes linear analysis for stability and controllability in trim conditions. The control strategy includes mode-specific controllers and a transition algorithm for seamless switching between VTOL and FW modes, aiming to expand the operational flight envelope and improve agility and

survivability. Simulations and flight tests confirm the UAV's dual-mode flight characteristics and additional advantages.

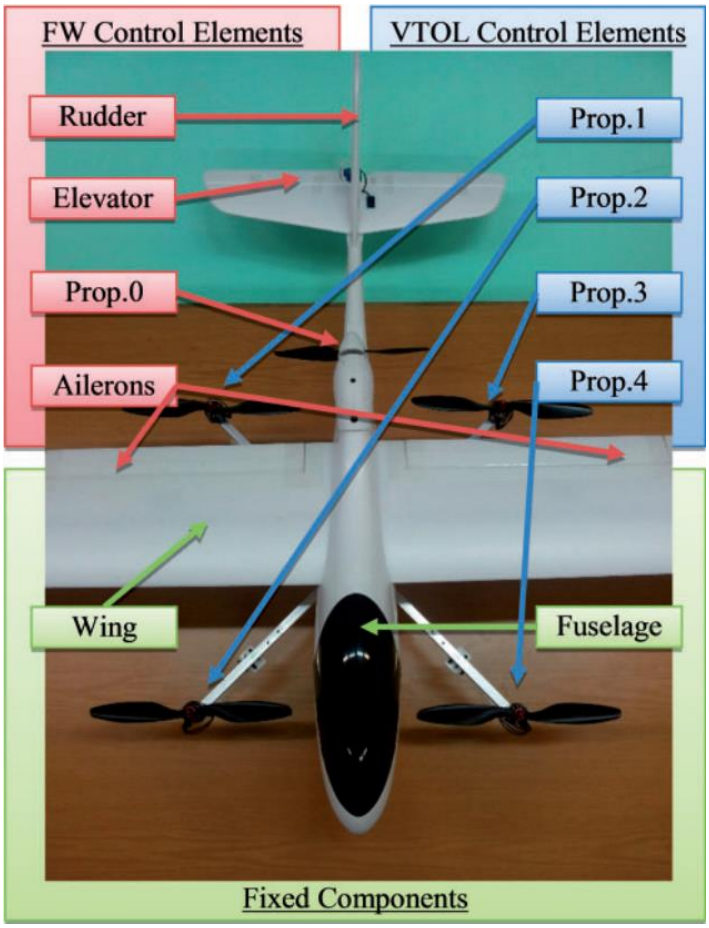

Figure 2.1. Control elements of VTOL–FW UAV

The aircraft's main components as shown in Figure 2.1, including the fuselage, fixed wing (FW) propulsion system, main wing, ailerons, rudder, and elevator, play crucial roles in its flight dynamics. The fuselage encounters drag opposite to airflow, counterbalanced by the FW propulsion system's thrust. The main wing generates lift to counteract gravity, while the ailerons, rudder, and elevator facilitate conventional airplane movements like roll, pitch, and yaw. The VTOL propulsion systems enable

lift and these movements through variable propeller speeds, akin to a multirotor's functionality.

The VTOL-FW UAV's flight mode is dictated by its vertical and horizontal velocities within an expanded flight envelope in Figure 2.2, covering both VTOL and FW regions. In low-velocity conditions, VTOL mode is engaged with corresponding control elements active. As horizontal velocity increases, the aircraft transitions to FW mode, activating FW control mechanisms. A control mixer manages the activation of these elements during the transition phase, effectively distributing commands to ensure a smooth switch between modes.

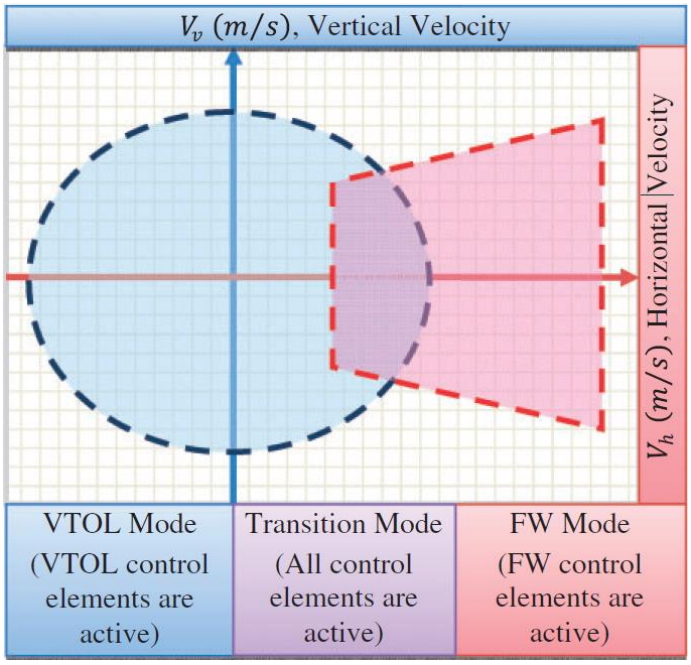

Figure 2.2. Flight envelope of VTOL–FW UAV platform

In another study, Gu et al [9] outlines the creation and validation of the aerodynamic performance, flight stability, and system-level autonomy of the designed dual-system VTOL UAV and its accompanying flight control system. The first experiment focuses on confirming the reliability of the transition controller, while

the second experiment assesses the stability of the flight controller during active pilot control. The third experiment evaluates the UAV's range, endurance, and its ability to execute autonomous flight, and the fourth experiment aims to establish the optimal flight speed.

Quadplanes are often used in applications where both VTOL capability and the ability to cover large distances efficiently are required. For example, they can be used in search and rescue [10-13], surveillance [14-16], mapping [17, 18] and other tasks [19, 20] where versatility in flight modes is essential.

Quadplanes can be effective in maritime operations such as surveillance, search and rescue [21, 22] or cargo. The VTOL feature of quadplanes eliminates the need for runways. The fixed wing flight capability provides longer flight times than multicopters and ensures faster flight to track moving trucks or ships.

2.2 Vision Based Localization

The studies [23-27] delve into various aspects of vision-based autonomous landing systems for UAVs, each highlighting different focal points and methodologies. These studies might explore topics such as the integration of computer vision for precision landing, algorithmic advancements for better target recognition, and enhancements in UAV stability and adaptability during the landing phase. The collective research offers a broad spectrum of insights into improving UAV landing capabilities through advanced vision-based techniques.

Blachut and co-workers [23] explore a hardware-software system enabling a multirotor UAV's autonomous landing. It utilizes a vision-based control system with real-time processing of a 1280x720 video stream at 60 fps and a LiDAR sensor for altitude measurement. The system, tested on various sequences, showed a landing pad detection accuracy of 96%. The research highlights the effectiveness of using a reprogrammable heterogeneous computing system in UAVs for real-time data processing with low energy consumption.

This study shows that the camera and a Lidar sensor are useful for an UAV's autonomous landing. However, the landing point is stationary for this study. This thesis focuses on moving landing points and variable heights due to waves.

A simplified block diagram of the entire vision algorithm is depicted in Figure 2.3. The input image converted into gray scale at first. Then a standard Gaussian blur filtering is applied. Adaptive thresholding is used for the detection of the landing mark. Then a few simple context filtering operations such as erosion, median filtering, and dilation to remove small objects. The other steps are connected component labelling, detection of the rings, squares and rectangles and bounding box analysis to reduce the number of false detections. Finally position and orientation are calculated.

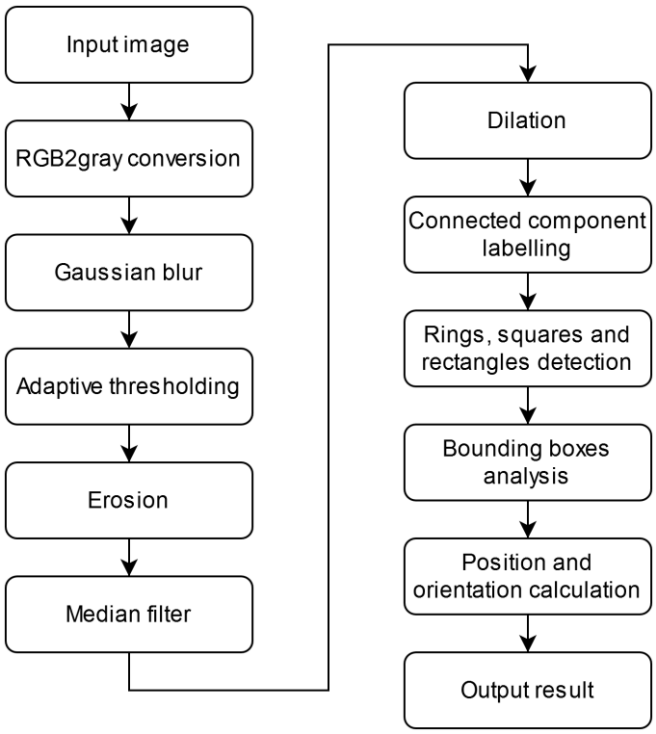

Figure 2.3. Simplified dataflow of the vision algorithm

Yu et al [28] presents an innovative approach for indoor localization of mobile robots. It integrates the Adaptive Monte Carlo Localization (AMCL) method with AprilTag detection for improved accuracy. The key methodology involves using low-cost image processing of AprilTags for precise localization and a laser rangefinder to refine the robot's pose. The approach enhances the AMCL framework by correcting errors in the odometry model with the absolute pose information from AprilTags. Field tests demonstrate the method's enhanced localization accuracy and stability compared to traditional AMCL, highlighting its effectiveness in indoor environments.

The algorithm, as shown in Figure 2.4, in this paper integrates global coordinate information provided by AprilTag into the odometry model. This integration enhances the accuracy of the localization system. Additionally, the algorithm restricts the prior distribution of particles within a permissible range through threshold comparison, further refining the localization process.

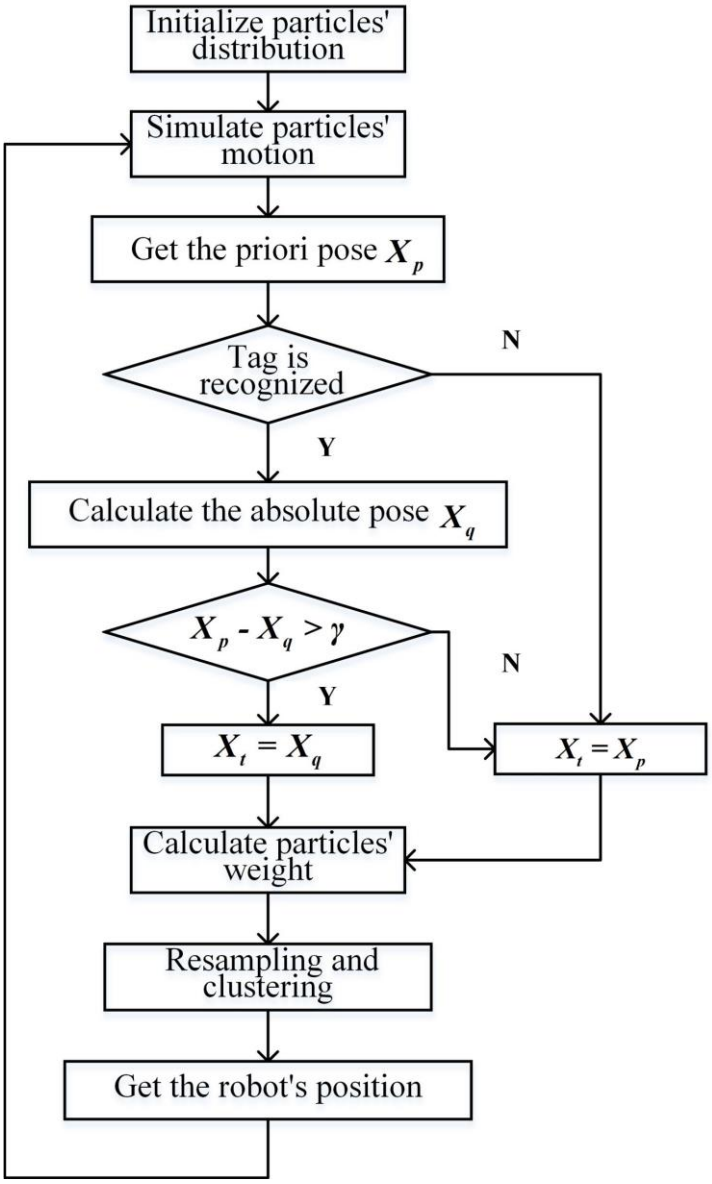

Figure 2.4. AprilTag based visual localization algorithm

Figure 2.5 shows the detection of an AprilTag with a camera.

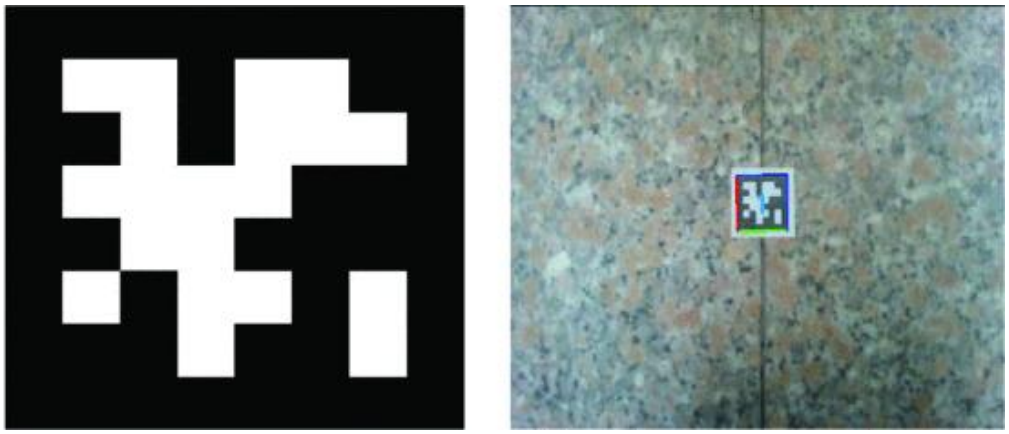

Figure 2.5. AprilTag detection

Olsen [29] introduces a robust method for detecting visual fiducials, employing a graph-based image segmentation algorithm based on local gradients for precise line estimation. Additionally, a quad extraction method is described, capable of handling significant occlusions in the fiducials. The authors demonstrate that their detection system achieves significantly better localization accuracy compared to previous methods. A new coding system is proposed to address challenges unique to 2D barcoding systems, focusing on rotation robustness, and minimizing false positives from natural imagery. Experimental results indicate that the novel coding system provides substantial theoretical and real-world benefits over existing approaches. In summary, AprilTags presents an advanced and comprehensive approach to visual fiducial detection, incorporating innovations in image segmentation, occlusion handling, accuracy improvement, and coding systems.

The studies [30-39] explore vision-based control algorithms for UAV landings on moving platforms under varying conditions, each with its unique emphasis. These studies likely cover different aspects such as algorithm robustness, adaptability to dynamic environments, precision in tracking and landing on the platforms, and

integration with UAV systems for enhanced autonomous capabilities. This approach underscores the significance of vision-based systems in improving UAV interactions with moving targets, contributing valuable insights into autonomous landing technology advancements.

In [31], Paris presents an advanced autonomous system enabling multirotor UAVs to land on moving platforms, overcoming challenges such as precise localization, swift trajectory planning, and robust control in windy conditions. The system integrates localization, planning, and control for efficient landings, utilizing an extended Kalman filter for distant estimations and a visual fiducial system, AprilTag, for closer approaches. It features an online-generated landing trajectory via receding horizon control, executed by a sliding controller that ensures stability against disturbances. This method facilitates a fast, direct landing without the need for hovering, showcasing its effectiveness through simulations and practical experiments.

The dynamic nature of ship landing, characterized by moving and heaving target positions, demands a system that can quickly and accurately identify and track the landing platform. AprilTags could facilitate this by providing a reliable reference point that can be easily detected by simple cameras. The tags' design allows for efficient detection even in challenging conditions, which is essential in the maritime environment where factors such as lighting conditions, weather, and the movement of the sea can affect visual recognition systems.

2.3 Quadplane Ship Landing

The studies referenced as [35-43] concentrate on algorithms for autonomous ship landing. These works likely explore various methodologies, technologies, and control strategies to enable UAVs to land on moving ship decks safely and efficiently, accounting for dynamic maritime conditions and operational challenges.

Each study may offer unique insights into algorithm design, integration with UAV systems, and adaptation to environmental factors such as wind and wave motion.

Tan and Wang [41] focus on the vertical landing of a quadrotor on a ship deck in high seas. It proposes a novel approach using the invariant ellipsoid technique, which eliminates the need for complex ship deck motion modeling. Instead, this method controls tracking error within an optimal bound, significantly improving landing accuracy and safety. The study includes simulations that demonstrate the effectiveness of this technique in realistic ship deck scenarios, showcasing the potential for improved UAV landing operations in maritime environments.

The research concentrates exclusively on the vertical oscillations of the ship, neglecting its lateral movements. Achieving synchronization with the vertical fluctuations of a ship in motion introduces additional complexities.

Lin and co-workers [42] introduce an innovative approach for the autonomous landing of VTOL UAVs on ships. The method employs Tau theory to create a landing algorithm that effectively synchronizes the UAV's movements with the dynamic conditions of a moving ship. The Tau theory is a perspective-based approach, which is used to estimate time-to-contact and guide the UAV during its descent, ensuring a safe and accurate landing. This approach is particularly advantageous in maritime environments where ship motion can be unpredictable due to sea conditions. The study's findings demonstrate the potential of this method for improving the precision and safety of UAV landings in complex and variable environments.

The study's suggested landing path for the UAV fails to account for its real-world dynamic limitations, focusing solely on achieving the shortest possible time. Also, the suggested method is not implementable to work on microcontrollers because of its complexity.

Trong et al. [43] proposes a deep learning-based autonomous landing system for VTOL UAVs on moving ships. The system employs a companion computer to

process data onboard the UAV, using a deep learning model for real-time detection and tracking of the ship and landing pad. The approach enhances the UAV's capability to land autonomously on a moving ship by improving the accuracy and speed of image processing. The system's effectiveness is validated through hardware-in-the-loop simulations using Jetson Nano and X-Plane, demonstrating its potential for real-time maritime applications. This research addresses the challenges of autonomous UAV landings in dynamic marine environments, utilizing advanced image processing and deep learning techniques.

The key oversight in this study is the omission of the ship deck's vertical motion caused by sea waves. Additionally, VTOL UAV systems are designed to operate without the need for large ships and helipads, capitalizing on their ability to take off and land in confined spaces.

In this thesis study, a PID controller is utilized for its simplicity in implementation on a microcontroller. For effective ship landing operations in wavy sea conditions, it's crucial that the PID controllers are meticulously calibrated and tuned to handle the complexities of the marine environment. There are many methods used to tune gain of the PID controllers such as Ziegler Nichols method [44] and Cohen Coon method [45]. Tan et al [46] introduces a new PID gain tuning method based on the invariant ellipsoid technique, addressing the challenge of persistent bounded disturbances like ship deck motion. The method aims to optimize controller gain for system stability under these disturbances. Simulations using MATLAB and Simulink demonstrate the method's effectiveness compared to traditional tuning methods, showing improved disturbance rejection and tighter tracking error control. This research provides an alternative approach for PID controller gain selection in scenarios with persistent disturbances.

CHAPTER 3

METHODOLOGY

3.1 System Architecture

Designing the system architecture for an autonomous landing system for a VTOL UAV on a moving ship deck involves the integration of various components such as ship, quadplane, flight controller units (FCU), companion computer, sensors, communication links and GCS. Figure 3.1 shows the overview of the system architecture including all the parts that will be explained in detail.

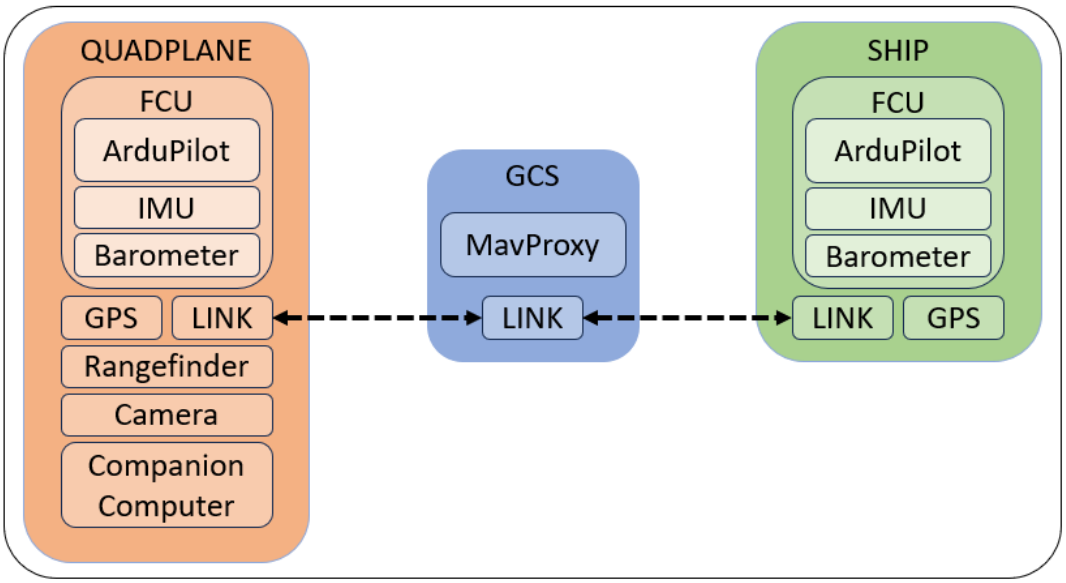

Figure 3.1. System architecture

GCS consists of software working on a PC and a communication link which works as a bridge between the quadplane and the ship. Quadplane and ship both have an

FCU including Inertial Measurement Unit (IMU), barometer and ArduPilot firmware, GPS, and a communication link. Quadplane also has a laser rangefinder, camera and a companion computer which will be needed to implement the suggested algorithms for this work.

Position information from the GPS of the ship is transmitted to the quadplane through the GCS. Quadplane uses this position information to follow the ship and match the horizontal velocity with it. Then quadplane uses the camera and companion computer to adjust the precise landing position. Finally, rangefinder data is used to synchronize vertical movements of the quadplane with the heaving motion of the ship.

3.1.1 VTOL Fixed Wing UAV Platform (Quadplane)

A quadplane is a hybrid UAV that seamlessly combines features of both quadcopters and fixed wing airplanes. This design [9] incorporates five rotors—four arranged in a quadcopter configuration for VTOL, and an additional rotor dedicated to fixed wing operations during horizontal flight as seen in Figure 3.2.

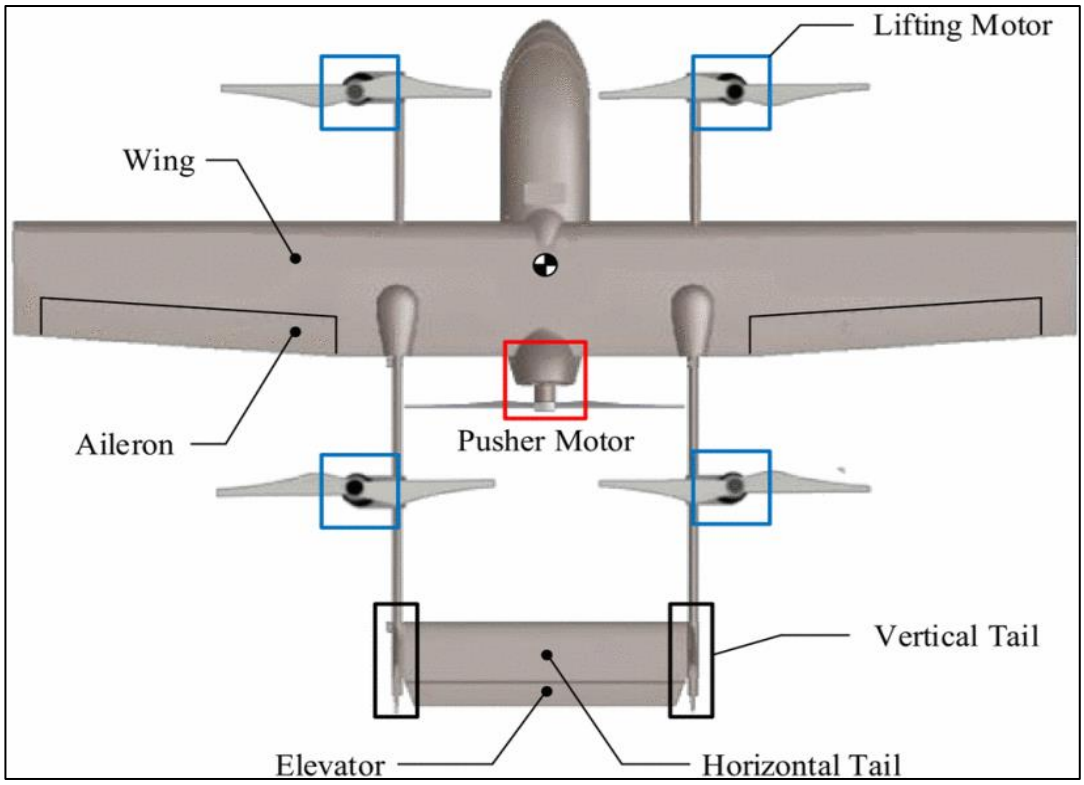

Figure 3.2. A quadplane system architecture

During VTOL mode, the quadplane utilizes its quadcopter rotors for vertical ascent and descent, providing adaptability in confined spaces. Once airborne, the UAV transitions to fixed wing flight without the need for tilting mechanisms. The fifth rotor takes over, enabling efficient horizontal travel. This design optimizes the advantages of both quadcopters and fixed wing aircraft.

Figure 3.3 shows the position and the attitude controller scheme of a quadplane. The main controller is the VTOL attitude controller block. The other parts can be divided into mainly two parts as Multi Copter (MC) and Fixed Wing (FW). MC and FW position controllers feed virtual attitude setpoint to the main controller. Main controller supplies attitude setpoints to the MC and FW attitude controllers. Then, according to switched mode (MC or FW) MC or FW attitude controllers supplies body rate setpoints to the MC and FW rates controllers. MC and FW rates controllers

feed the main controller with the virtual actuator controls. Finally, the main controller controls the actuators.

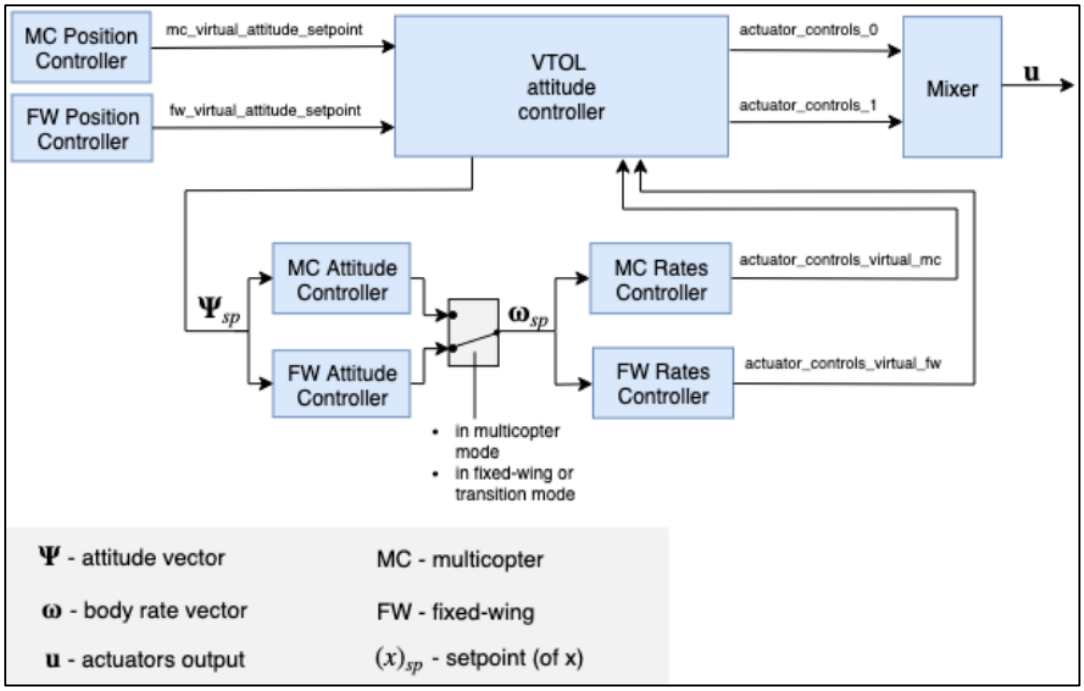

Figure 3.3. Position and attitude control of a quadplane

Quadplanes find applications in various fields, including surveillance, mapping, agriculture, and scenarios requiring the combined benefits of vertical takeoff and landing capabilities and the efficiency of fixed wing flight for longer-range missions.

3.1.2 Flight Controller Unit (FCU)

In the realm of UAVs, an FCU serves as the central intelligence governing the aircraft's flight dynamics. Acting as the UAV's "brain," the FCU processes data from onboard sensors like gyroscopes and accelerometers, enabling real-time stabilization. Additionally, it interprets user inputs, translating them into motor adjustments to execute desired maneuvers. The FCU plays a pivotal role in

navigation by incorporating signals from GPS modules to ascertain the UAV's location and heading. It manages altitude through barometric pressure sensors and facilitates autonomous features such as waypoint navigation and return-to-home functions. Communication between the UAV and ground control is also orchestrated by the FCU, providing telemetry data, and ensuring fail-safe mechanisms in emergencies. This critical component underpins the UAV's safe and controlled operation across various applications.

Pixhawk Cube Orange, shown as Figure 3.4, is a powerful commercial autopilot hardware which includes an STM32H757 ARM processor, a 9 axis IMU and two barometers. It supports ArduPilot firmware.

Figure 3.4. Pixhawk cube orange

ArduPilot [47, 48], an autopilot firmware, is stacked on the FCU of the system. The FCU is the core component responsible for stabilizing and controlling the UAV during flight. Also, it includes the Lua scripts for the ship landing algorithms. In this thesis work, Lua scripts are used for mainly three purposes as velocity matching with the ship, horizontal position control and vertical descent rate control. Firstly, the position of the ship and quadplane are used to track the ship and match the velocities. Then, horizontal position offsets calculated by the companion computer vision algorithm are used to adjust the precise landing position. Finally, rangefinder data is used to synchronize the vertical movements of the ship and quadplane.

3.1.3 Companion Computer

In the realm of robotics, drones, or autonomous systems, a companion computer is an additional computing device that works alongside the main computer or controller of the system. It serves as a supplementary unit designed to handle specific tasks or functions, often related to perception, computation, or data processing. The term "companion" emphasizes its collaborative role with the primary computing system.

For example, in the field of UAVs, a companion computer might be used to manage tasks such as image processing, computer vision, or additional sensor data analysis. This allows the main flight controller to focus on navigation and control, while the companion computer handles more computationally intensive tasks.

In robotics, a companion computer could be dedicated to tasks like object recognition, mapping, or decision-making, allowing the primary controller to focus on motor control and overall system coordination. In autonomous vehicles, a companion computer might process data from sensors like cameras, lidar, or radar, contributing to perception and decision-making tasks while the main vehicle controller manages navigation and control.

The use of companion computers allows for a modular and distributed approach to system design, where specific functions can be offloaded to specialized hardware.

This can enhance the overall efficiency, responsiveness, and capabilities of the autonomous or robotic system.

In our system, companion computer is responsible for getting images from the camera, searching, and detecting the tags on the ship deck and supplying horizontal position offsets to the FCU. As a companion computer, Raspberry Pi 4 can be used with the features such as being lightweight and low cost thanks to low computational cost of the proposed algorithm.

3.1.4 Sensors

3.1.4.1 IMU, barometer

In various applications, including robotics, drones, and navigation systems, an IMU and a barometer are two distinct sensors that play crucial roles. The IMU is designed to measure and report an object's acceleration, angular rate, and sometimes magnetic field strength. It typically includes accelerometers for linear acceleration, gyroscopes for angular velocity, and sometimes magnetometers for measuring the Earth's magnetic field. IMUs find applications in navigation systems, drones, robotics, and wearable devices, aiding in determining orientation, tracking movement, and stabilizing devices.

On the other hand, a barometer is specifically designed to measure atmospheric pressure. Barometers come in various types, including mercury barometers, aneroid barometers, and electronic barometers. They are commonly used in weather monitoring, altimeters for measuring altitude, and, when combined with other sensors, for estimating changes in elevation. In drones and UAVs, barometers contribute to altitude control, playing a crucial role in maintaining stable flight by sensing changes in atmospheric pressure. Both IMUs and barometers provide essential data for a variety of applications, allowing systems to navigate, stabilize, and respond to changes in their surroundings. Their integration is often part of a

sensor fusion strategy to obtain a more comprehensive understanding of an object's state and its interaction with the environment.

At the main controller of the quadplane, roll, pitch and yaw states are calculated from the IMU raw data and altitude state is calculated from the atmospheric pressure data of the barometer. And these states are used for the MC and FW attitude controllers.

3.1.4.2 GPS

In various technological applications, GPS is a pivotal component providing accurate location information. GPS utilizes a network of satellites to triangulate the position of a GPS-enabled device on Earth. The system consists of a receiver, which interprets signals from multiple satellites to determine the device's latitude, longitude, and sometimes altitude.

The primary purpose of GPS is to enable precise navigation and location-based services. It finds widespread use in navigation systems for vehicles, smartphones, aviation, marine operations, and outdoor activities. GPS technology facilitates route planning, real-time tracking, and mapping applications. Additionally, it plays a critical role in synchronizing time across various devices and systems.

GPS is integral to modern life, influencing diverse sectors such as logistics, emergency services, agriculture, and scientific research. Its continuous evolution and integration with other technologies contribute to advancements in autonomous vehicles, geospatial analysis, and the Internet of Things (IoT). The ubiquity of GPS has transformed the way we navigate, communicate, and interact with the world around us.

GPS receivers are used on quadplane and ship to estimate the global positioning of the vehicles. Then, positions are used for following ship by quadplane.

3.1.4.3 Laser Rangefinder

A laser rangefinder is a device that utilizes laser technology to measure distances accurately. It operates by emitting a laser beam towards a target and measuring the time it takes for the laser pulse to travel to the target and back. By using the speed of light, the rangefinder calculates the distance to the target. Key components include a laser emitter, beam collimator, target reflector, and a time-of-flight measurement system. Laser rangefinders find applications in surveying, golf, military and defense, hunting, construction, and other fields where precise distance measurements are essential. They offer a quick and precise method for determining distances, contributing to increased accuracy in various activities.

In our scenario, it is used to measure height of the UAV over the ship deck precisely. The measurement is used to control vertical velocity of the UAV when landing on the ship deck.

3.1.4.4 Camera

Cameras on drones play a crucial role in facilitating precision landings. They are essential for providing real-time visual information during the landing process, ensuring accuracy and safety. These cameras assist in assessing the landing site, avoiding obstacles, and achieving a controlled descent. Key considerations for camera usage in landing operations include resolution, stabilization mechanisms, and the integration of sensors for precise navigation. Continued advancements in camera technology contribute to improving the effectiveness and reliability of drone landings in various scenarios.

A camera is mounted at the bottom of quadplane to detect the AprilTag on the ship deck. Then images from the camera are used on the suggested vision-based position control algorithm runs on the companion computer.

3.1.5 Beacon

Beacon includes an FCU and GPS receiver that are stacked on the ship. For the ship landing purpose, the position of the ship is needed. The global position of the surface vehicle, ship, reaches to UAV through the GCS.

3.1.6 Communication Links

Communication links on UAVs are essential for enabling data exchange between the UAV and its operators or ground control systems. These links serve various purposes, including radio control for remote UAV operation, telemetry for real-time data transmission, video transmission for live streaming of footage, command and control for mission planning, and satellite communication for extended range or beyond-line-of-sight operations. Additionally, mesh networks can be employed to enhance communication reliability between multiple UAVs or between UAVs and ground stations. The selection of communication links depends on factors such as mission requirements, range, data bandwidth, and environmental conditions. Effective communication is crucial for ensuring the success and safety of UAV operations by enabling real-time control, monitoring, and data acquisition.

For the system, links are used to communicate between the UAV and the ship using the GCS as a bridge.

3.1.7 Ground Control Station (GCS)

A GCS is a central hub for operating unmanned vehicles, such as drones. It serves as a mission control interface where operators plan, execute, and monitor missions. The GCS displays real-time telemetry data, including altitude, speed, GPS coordinates, and sensor readings. Operators use the GCS to establish communication links, send control inputs, and implement safety measures. It also provides interfaces for adjusting payload settings and visualizing data. Some GCS systems offer

mapping tools, and the station is responsible for managing software updates and maintenance tasks. The user-friendly interface facilitates efficient interaction with the unmanned vehicle, ensuring safe and effective operations.

Although the system is autonomous for takeoff, landing and other missions. GCS is essential to edit and start the missions. Also, live data from the vehicles can be observed. MavProxy (Mavlink Proxy) will be used for the GCS role. It consists of a command line console, console module and map module. Also, it has features such as graphs for live data which will be used in the implementation and simulation of the system. Figure 3.5 shows a screenshot of a MavProxy module.

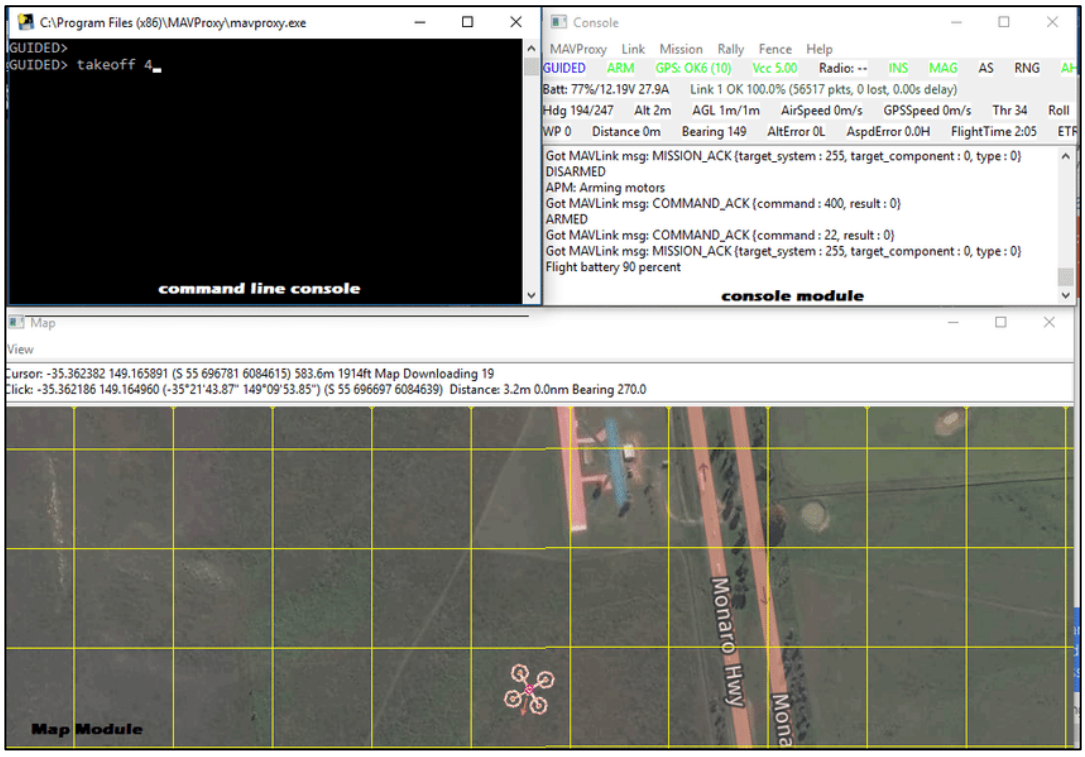

Figure 3.5. MavProxy

3.2 Control Algorithms

Landing a Quadplane on a ship involves a unique combination of capabilities that make it well-suited for maritime operations. The Quadplane, functioning as both a quadcopter and a fixed wing aircraft, begins by utilizing its quadcopter configuration for vertical takeoff from the ship's deck. This VTOL capability allows it to launch from confined spaces without the need for a traditional runway.

Once airborne, the Quadplane smoothly transitions to fixed wing flight, relying on its wings for lift and a dedicated propulsion system for forward motion. This phase enables the UAV to cover larger distances efficiently. As the Quadplane approaches the ship for landing, it undergoes a controlled transition back to the quadcopter configuration.

During the landing phase, the Quadplane descends vertically, leveraging its quadcopter setup to precisely control its descent rate and align with the designated landing spot on the ship's deck. This ability to switch between vertical and horizontal flight modes is a key advantage, allowing the Quadplane to operate in environments where both types of capabilities are essential.

The challenges of landing on a moving ship are addressed through advanced navigation and control algorithms. These algorithms factor in the ship's dynamic movements, including pitch, roll, and heave, ensuring that the Quadplane can adjust its approach and descent accordingly. The result is a sophisticated and adaptable UAV system capable of performing complex maritime operations with precision and reliability.

In scenarios where GPS position precision may be insufficient, visual assistance becomes crucial for precise landing on a ship. The integration of computer vision technologies, such as the AprilTag library, is a common solution to address this challenge.

Absolutely, addressing the heaving of the ship caused by waves is a crucial consideration in precise UAV landings. To overcome this challenge, integrating a laser rangefinder into the quadplane's navigation system can be a valuable solution.

A laser rangefinder measures the distance between the UAV and the landing surface by emitting laser beams and measuring the time it takes for the beams to reflect. By continuously measuring the distance to the ship's deck during the landing approach, the laser rangefinder provides real-time data on the ship's vertical movements, including heave caused by waves.

This information becomes essential for the UAV's control system to dynamically adjust its altitude and descent rate to compensate for the ship's motion. By actively responding to changes in the ship's heaving, the Quadplane can maintain a stable and controlled descent, ensuring a safe and accurate landing on the moving platform.

The integration of multiple sensor modalities, such as GPS, computer vision with AprilTag, and a laser rangefinder, allows for a robust and redundant system. This sensor fusion approach enhances the UAV's ability to adapt to diverse and challenging environments, providing a higher level of precision and reliability, particularly in maritime settings where environmental conditions can be unpredictable.

3.2.1 Quadplane Ship Landing

The ArduPilot software's standard landing algorithm with ship landing script is adept at ship tracking and velocity synchronization; however, it falls short in accurately addressing the complexities of wave dynamics. This shortfall results in less precise landings on undulating sea surfaces. To enhance accuracy, the proposed algorithm is designed to activate once the quadplane achieves velocity parity with the ship, hovering at an approximate landing position.

In the standard landing protocol, the quadplane initially enters MODE_QRTL (Quadplane Return to Launch Mode), positioning itself strategically above the

landing pad. It then adjusts its horizontal speed to mirror that of the ship, maintaining a predetermined altitude, as depicted in Figure 3.6.

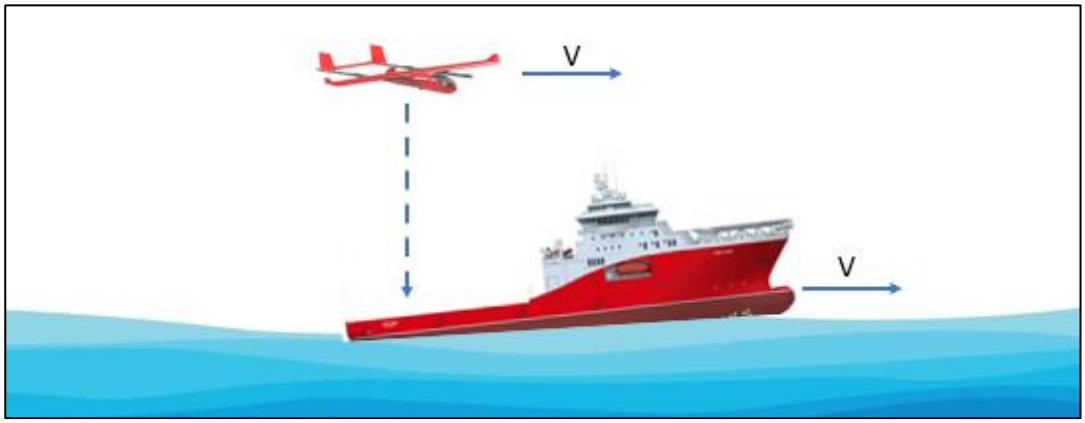

Figure 3.6. Landing on a ship deck

This process sets the stage for the implementation of the newly devised landing algorithm, illustrated in Figure 3.7. The algorithm begins by recalibrating the quadplane's horizontal position, guided by a tag identified on the ship's deck. This is followed by a critical stabilization phase of 10 seconds, ensuring that the aircraft is optimally positioned for the descent. Subsequently, the quadplane commences a controlled and steady descent, the rate of which is linked to its altitude relative to the ship deck. The final phase of the landing sequence is carefully designed, gradually reducing the descent rate to facilitate a gentle and precise touchdown, thereby significantly enhancing the landing smoothness and overall safety.

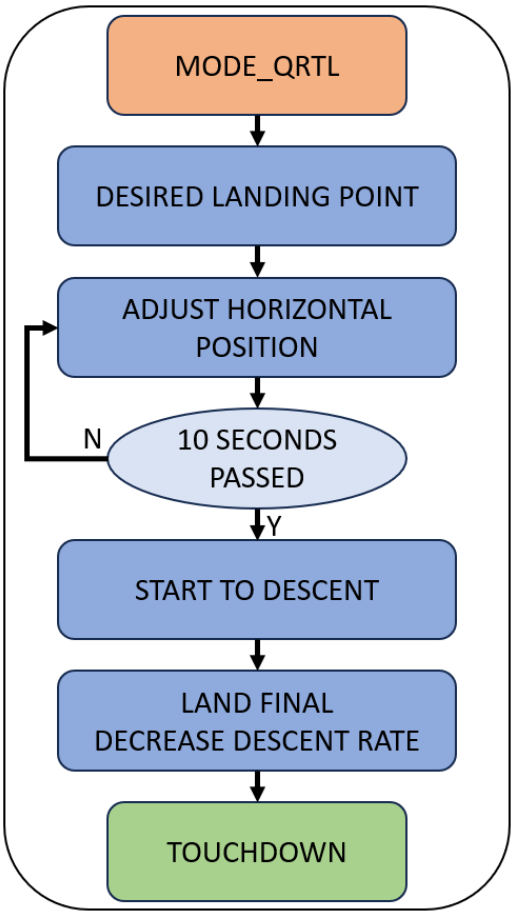

Figure 3.7. Landing algorithm

3.2.2 Vision Based Position Control

The AprilTag library [49] is employed for precise horizontal positioning on a ship deck. The software on the companion computer captures images from the camera, identifies the tags on the ship deck, and provides position error data directly to the FCU of the UAV.

AprilTags, visual fiducials with unique square-shaped markers, are strategically placed on the ship's deck. Equipped with a camera, the UAV uses computer vision

algorithms to recognize and interpret these markers, enabling accurate localization and navigation.

Two cascaded AprilTag is used in this project. It is designed as big as to be detected at high altitudes and smaller one in the tag to be detected at lower altitudes. The tag is shown on Figure 3.8.

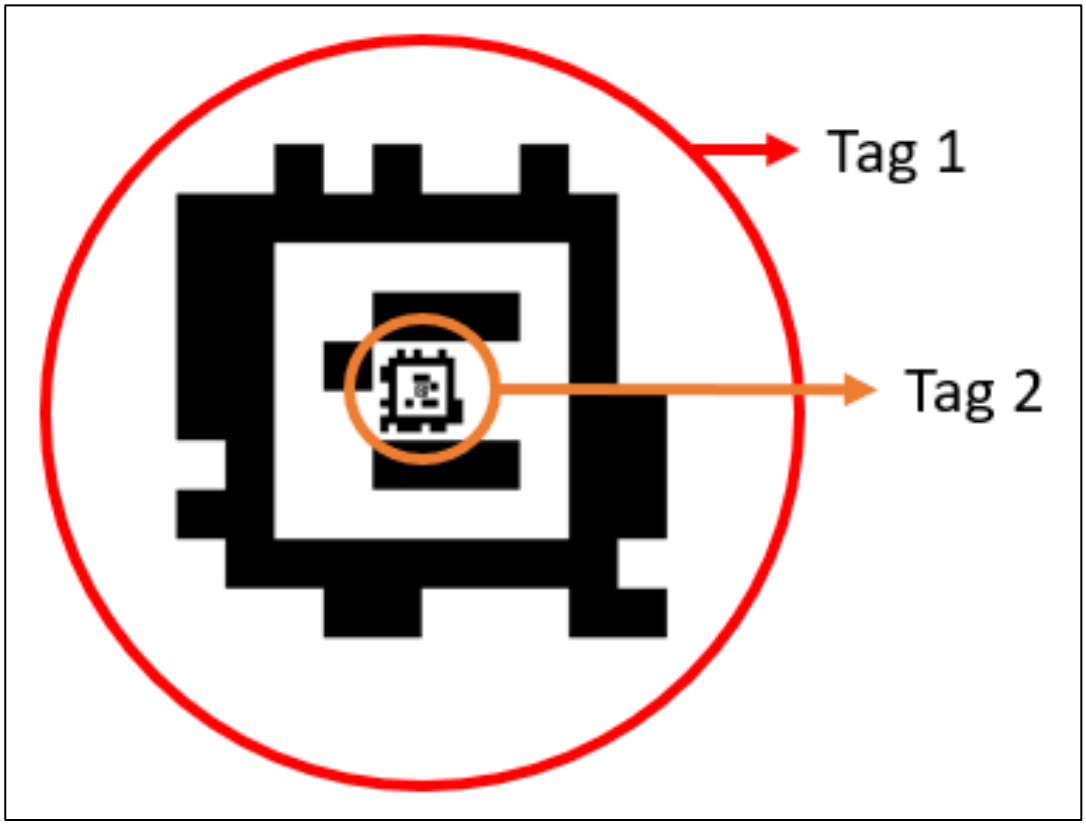

Figure 3.8. Custom cascaded AprilTag

During the landing process, the companion computer processes images from the camera, identifies AprilTags, and calculates position errors. This information is then relayed directly to the UAV's FCU, allowing real-time adjustments to maintain precise positioning.

Figure 3.9 shows the steps of the proposed algorithm. The algorithm is separated into two parts. Python code is running on the companion computer using OpenCV [50] and AprilTag libraries handling the image processing part. Lua script code is running on the FCU beside ArduPilot firmware for getting pixel offsets from Python code, calculating the position error, updating the position using PID controller.

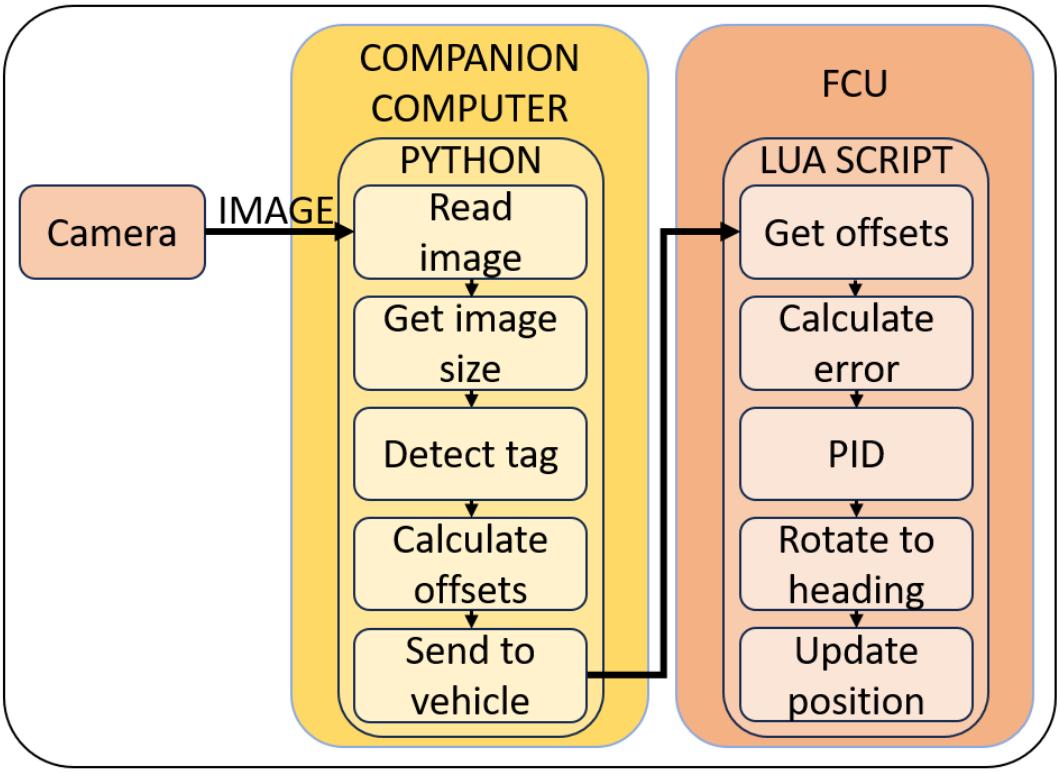

Figure 3.9. Vision based position control algorithm

Figure 3.10 explains the AprilTag detection and the pose estimation according to camera frame.

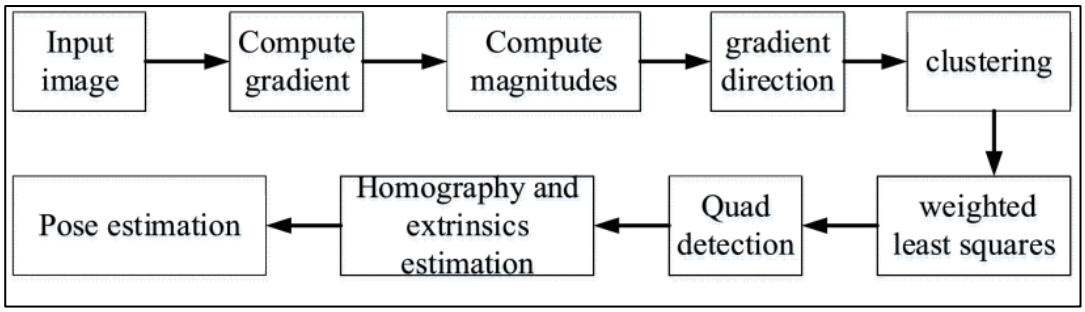

Figure 3.10. AprilTag detection and pose estimation algorithm

3.2.2.1 Coordinate Transformation

The whole system involves a total of five coordinate systems' conversion: pixel coordinate system, image coordinate system, camera coordinate system, world coordinate system, and the body coordinate system.

The first is the conversion between the pixel coordinate system and the image coordinate system.

Both the pixel plane and the image plane are on the imaging plane, except that the respective origins and units of measurement are different. The origin of the image coordinate system is the intersection of the camera's optical axis and the imaging plane, usually the midpoint of the imaging plane. The unit of the image coordinate system is mm, which belongs to the physical unit, and the unit of the pixel coordinate system is pixel. A pixel is described as a few rows and columns, so the relationship between the two is as follows:

$$u = \frac{x}{dx} + u_0 \ , \tag{3.1}$$

$$v = \frac{y}{dy} + v_0 \ , \tag{3.2}$$

where u and v in Eq. (3.1) and (3.2) are the coordinates of the image point $p(x, y)$ in the pixel plane, u_0 and v_0 are the principal point offsets of the image coordinate system in the pixel coordinate system. The form of the matrix is written as:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{1}{dx} & 0 & u_0 \\ 0 & \frac{1}{dy} & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}. \tag{3.3}$$

Then there is the conversion of the world coordinate system and the camera coordinate system.

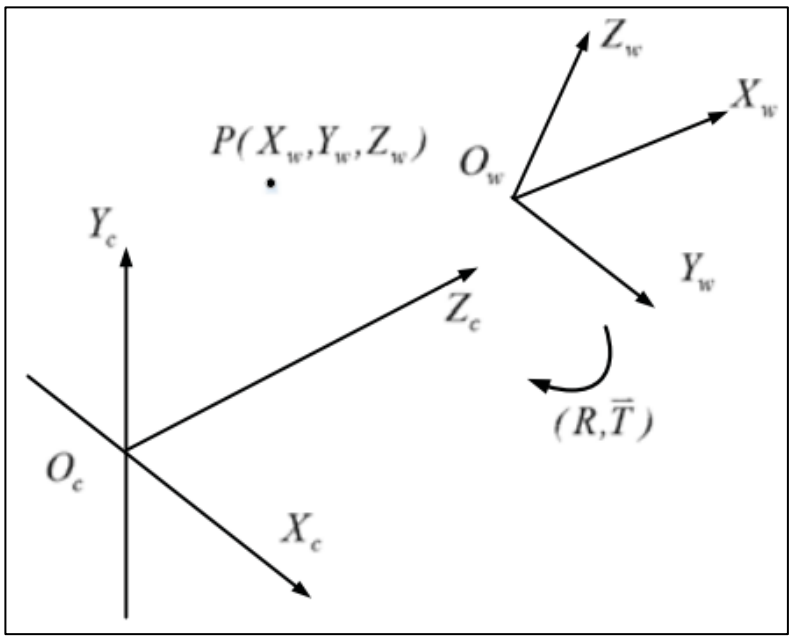

Figure 3.11. World coordinate system and camera coordinate system

In Figure 3.11, $P(X_W, Y_W, Z_W)$ is a point in the world coordinate system, which is the true point in life. It can be converted from the world coordinate system to the camera coordinate system by a certain rotation and translation.

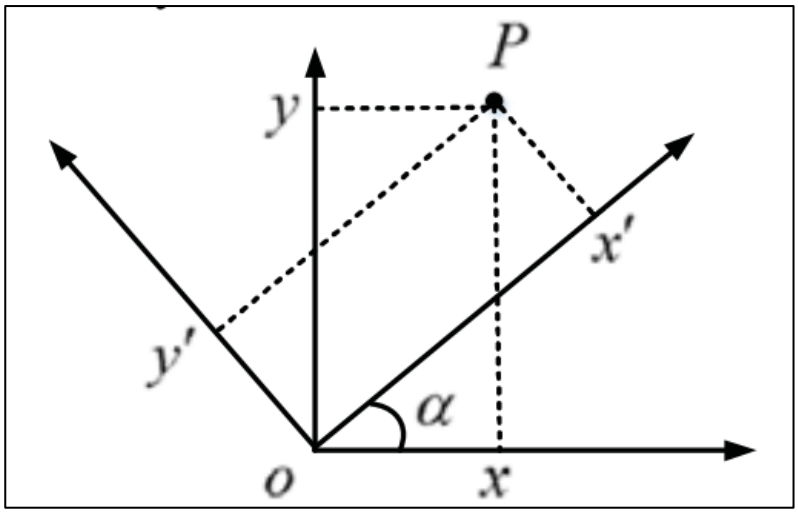

Figure 3.12. Rotation of the coordinate system

Figure 3.12 shows the rotation of the coordinate system around the X-axis. The coordinates of the point P under the coordinate system and the coordinates under the coordinate system satisfy the following relationship:

$$x' = x\cos\alpha - y\sin\alpha\ , \tag{3.4}$$

$$y' = x\sin\alpha + y\sin\alpha\ . \tag{3.5}$$

So, the rotation matrix around a single coordinate axis can be written as:

$$R_z(\alpha) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{bmatrix}. \tag{3.6}$$

The same principle can be obtained by $R_x(\beta)$ and $R_y(\theta)$. The rotation matrix of the coordinate system rotating around x-axis and y-axis, according to:

$$R = R_z(\alpha)R_y(\theta)R_x(\beta)\ . \tag{3.7}$$

Finally, the rotation matrix R in three directions:

$$R = \begin{bmatrix} \sin\theta\cos\alpha & \sin\beta\sin\theta\cos\alpha - \cos\beta\sin\alpha & \cos\beta\sin\theta\cos\alpha + \sin\beta\sin\alpha \\ \cos\theta\sin\alpha & \sin\beta\sin\theta\cos\alpha + \cos\beta\cos\alpha & \cos\beta\sin\theta\cos\alpha - \sin\beta\cos\alpha \\ -\sin\theta & \sin\beta\sin\theta\cos\alpha + \cos\beta\cos\alpha & \cos\beta\cos\theta \end{bmatrix}$$

(3.8)

The translation matrix $\vec{T}$ in Figure 3.11 represents the amount of translation of the world coordinate system and the camera coordinate system in the x , y , and z directions.

According to the conversion relationship between the world coordinate system and the camera coordinate system, the relative position and attitude between the AprilTags and the camera can be obtained.

Finally, it is needed to figure out the relationship between the frame coordinate system and the camera coordinate system.

3.2.2.2 PID Control

The classical PID control algorithm is chosen to solve the issue of response overshoot and lag or the system. Figure 3.13 shows the basic structure of the PID controller.

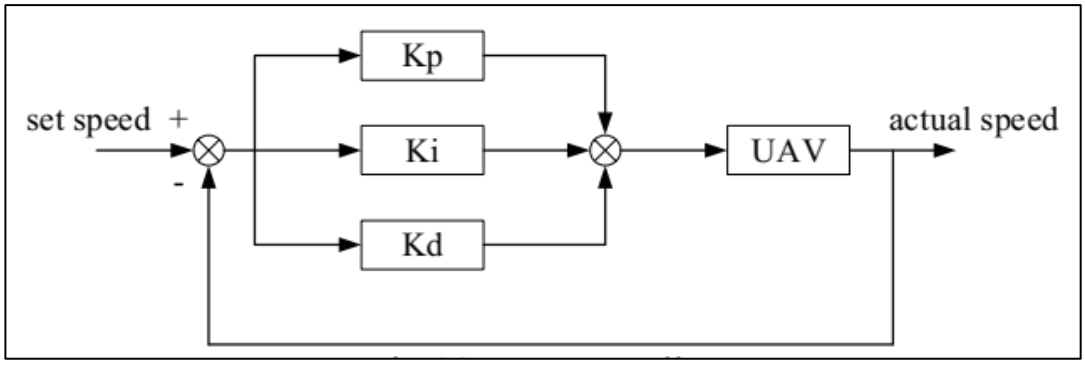

Figure 3.13. PID controller

Because the speed of the UAV is directly proportional to the position offset, the offset between the UAV and the tag is taken as the input speed of the UAV in a certain proportion, to solve this problem, the PID control law of speed control is given:

$$V = k_p err + k_i integral + k_d(err - last_{err}) , \quad (3.9)$$

where k_p, k_i, and k_d are the proportional, integral, and differential coefficients, respectively, and err is the error at the current time. The $last_{err}$ is the error of the previous moment.

The use of AprilTags enhances the UAV's landing system, providing a visual reference independent of external conditions, such as ship movement and GPS inaccuracies. This approach contributes to the adaptability and reliability of the landing system in dynamic environments, such as maritime operations.

3.2.3 Descent Rate Control

In developing an autonomous landing system for a UAV on a moving ship, several critical components and challenges are addressed. The UAV's landing process is complicated by the ship's heaving motion, imprecise barometer measurements affected by waves, and potential communication lag issues.

To overcome these challenges, a laser rangefinder is implemented to provide highly accurate height measurements, offering a reliable alternative to barometer data. The autopilot software incorporates a PID controller that dynamically adjusts the UAV's descent rate based on real-time information from the laser rangefinder. This adaptive approach ensures that the UAV can respond to changes in the ship's deck height, particularly during heaving motion, by modifying its descent rate accordingly.

Also, roll and pitch movements of UAV are considered to achieve the exact vertical distance to the landing point.

Figure 3.14 shows the algorithm for descent rate control. The distance between the ship and the quadplane is measured via rangefinder. Roll and pitch corrections are done for the exact vertical distance. Error with the target altitude is calculated. PID controller is applied. Then, the vertical velocity is updated with the PID controller output.

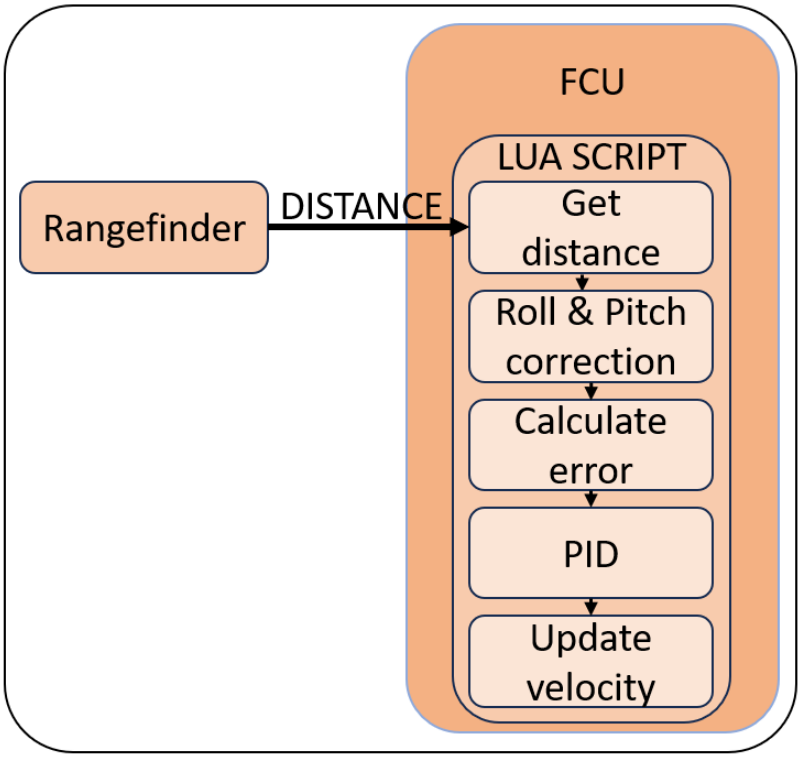

Figure 3.14. Descent rate control algorithm

In this controller, the heaving control of the quadrotor is primary concern. The attitude stabilization of the quadrotor is beyond the scope of this work. Therefore, the translational dynamics of the quadrotor will be described in the absence of the rotational dynamics. The equations of motion of the quadrotor are formulated in the inertial reference frame, the translational kinematics is:

$$\dot{p}^{(i)} = v^{(i)} , \tag{3.10}$$

where $p^{(i)}$ and $v^{(i)}$ are position and velocity of the quadrotor in the inertial reference frame. The translational dynamics is given by:

$$m\dot{v}^{(i)} = -R_{bi}^{T} T \hat{e}_3 + mg\hat{e}_3 \,, \tag{3.11}$$

where m is the mass of the quadrotor, g is the gravitational acceleration, $\hat{e}_3$ is the unit vector in the z axis in the body reference frame pointing downwards and R_{bi} is the rotation matrix from the body reference frame to the inertial reference frame. In this formulation, minor forces acting on the quadrotor such as the aerodynamic forces are assumed to be insignificant.

Then, the ship deck landing control problem is formulated. This is followed by a PID controller synthesis for the altitude control that lands the quadrotor onto a heaving ship deck. The vertical and planar dynamics are decoupled. The vertical dynamics is closely related to the tracking of the ship deck heave motion and maintaining the relative position between the quadrotor and the ship deck. The thrust vector in the inertial reference frame is:

$$T^{(i)} = -R_{bi}^{T} T \hat{e}_3 = -\begin{pmatrix} \sin\varphi \sin\psi + \cos\varphi \sin\theta \cos\psi \\ -\sin\varphi \cos\psi + \cos\varphi \sin\theta \sin\psi \\ \cos\varphi \cos\theta \end{pmatrix} T \,, \tag{3.12}$$

where φ, θ, ψ are the roll, pitch, and yaw angles respectively.

The z component of Eq. (3.11) can be rewritten as:

$$\ddot{h} = T\cos\varphi\cos\theta - mg \,, \tag{3.13}$$

where $h = -Z$ is the height of the quadrotor.

Letting $h_1 = h$, $h_2 = \dot{h}_1$ and $u = T\cos\varphi\cos\theta - mg$, Eq. (3.11) can be rewritten as:

$$\dot{h}_1 = h_2 \,, \tag{3.14a}$$

$$\dot{h}_1 = u \,. \tag{3.14b}$$

It is noted that $T = \frac{u+mg}{\cos\varphi \cos\theta}$ is limited by mechanical bounds thus the virtual control u is also limited. In most cases, φ and θ is small and bounded therefore it is reasonable to assume $-mg \leq u \leq T_{max} - mg$, where T_{max} is the maximum achievable thrust.

In this problem, the quadrotor is required to track the heave motion of the ship deck (d). The tracking error is given by $e_1 = h_1 - d - H$, where $H > 0$ is the commanded height above the ship deck. The tracking error dynamics becomes:

$$\dot{e}_1 = e_2 \,, \tag{3.15a}$$

$$\dot{e}_2 = u - \ddot{d} - \ddot{H} = u_{pid} - \ddot{d} \,, \tag{3.15b}$$

where $u_{pid} = u - \ddot{H}$. This can be further expressed in the linear state space form as:

$$\dot{e} = Ae + Bu_{pid} + Dw \,, \tag{3.16}$$

where $e = \begin{pmatrix} e_1 \\ e_2 \end{pmatrix}$, $A = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}$, $B = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$, $D = -\begin{pmatrix} 0 \\ \Delta \end{pmatrix}$ and $\Delta w(t) = \ddot{d}(t)$ for $\|x(t)\|_\infty = a$ is defined as $\{x(t): \|x(t)\|_2 < a \; for \; all \; t \geq 0\}$ and $\|x(t)\|_2$ is the Euclidean norm at time t. The parameter Δ can be estimated from the historical data of the ship deck heave motion at various sea states.

Eq. (3.16) shows the equation for a typical double integrator in the presence of external disturbance. A PID controller is proposed such that:

$$u_{pid} = K_i \int e_1 dt + K_p e_1 + K_d \dot{e}_1 \,, \tag{3.17}$$

where K_i, K_p and K_d are the gains of PID controller. A schematic of the closed-loop system is shown in Figure 3.15.

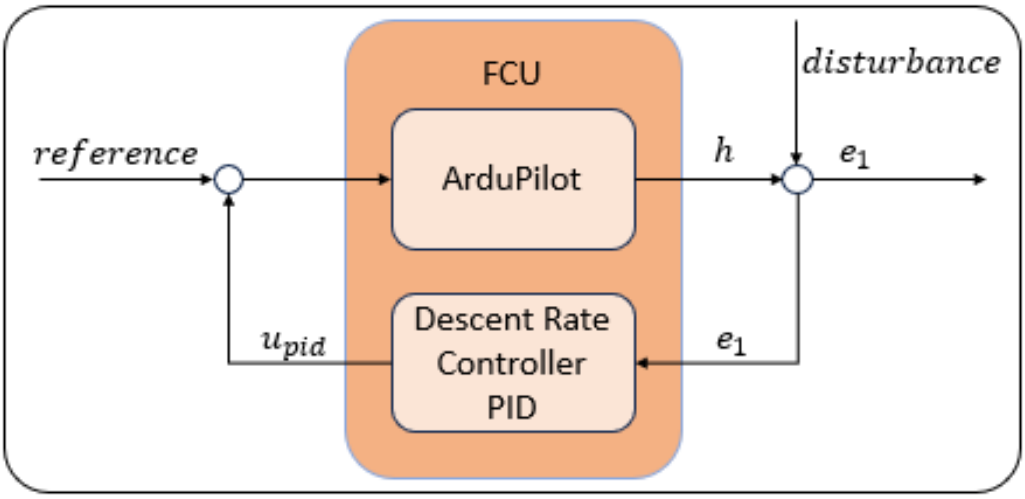

Figure 3.15. Schematic of the closed-loop control system

Letting $\tilde{e}_1 = \int e_1 dt$, $\tilde{e}_2 = e_1$ and $\tilde{e}_3 = e_2$, Eq. 3.16 becomes:

$$\dot{\tilde{e}} = \tilde{A}\tilde{e} + \tilde{B}\tilde{u}_{pid} + \widetilde{D}\widetilde{w}, \tag{3.18}$$

where $\tilde{e} = \begin{pmatrix} \tilde{e}_1 \\ \tilde{e}_2 \\ \tilde{e}_3 \end{pmatrix}$, $\tilde{A} = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix}$, $\tilde{B} = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$, $\widetilde{D} = -\begin{pmatrix} 0 \\ 0 \\ \widetilde{\Delta} \end{pmatrix}$, $\tilde{u}_{pid} = \dot{u}_{pid}$ and $\widetilde{\Delta}\widetilde{w}(t) = \Delta\dot{w}(t)$ for all $\|w\|_\infty = \|\widetilde{w}\|_\infty = 1$.

With respect to Eq. (3.18), the PID controller becomes:

$$\tilde{u}_{pid} = K_{pid}\tilde{e}\ , \tag{3.19}$$

where $K_{pid} = (K_i \quad K_p \quad K_d)$ is a form of state feedback.

CHAPTER 4

SIMULATIONS

In this section, a comprehensive overview of the simulation environment is provided, offering detailed insights into its structure and components. The focus extends to an in-depth analysis of the performance results obtained from various scenarios. Each scenario is meticulously examined, with selected parameters and their corresponding outcomes thoroughly discussed.

4.1 Simulation Environment

The simulation architecture, Figure 4.1, is comprehensively explained, comprising a 3D dynamic simulator, Gazebo [51] and the ArduPilot Software in the Loop (SITL) [52] simulator, serving as an open-source autopilot system for both quadplane and the ship. MavProxy [53] functions as the GCS software facilitating communication among vehicles. The vision-based position control algorithm is executed on the companion computer using Python with AprilTag library. Simultaneously, the descent rate controller operates within the ArduPilot SITL instance as a Lua script.

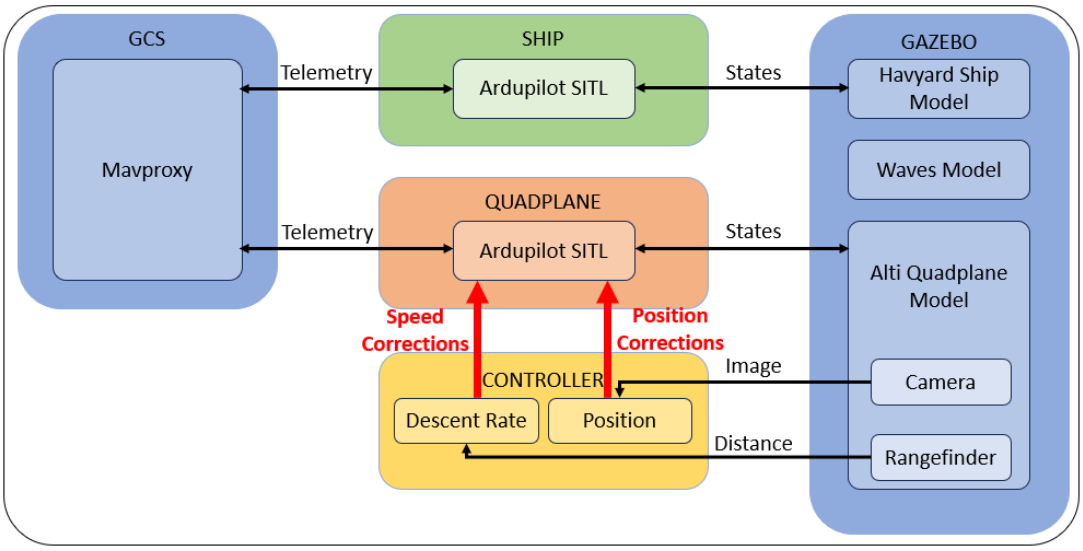

Figure 4.1. Quadplane ship landing simulation architecture

4.1.1 Gazebo Simulator

Gazebo is an open-source 3D robot simulation environment that allows users to simulate the motion and interaction of robots in a realistic, physics-based environment. It is widely used in the field of robotics for testing and developing robot algorithms, control strategies, and sensor integration. Gazebo provides a platform for simulating both ground, surface, and aerial robots in various indoor and outdoor environments.

Key features of Gazebo include:

Physics Simulation: Gazebo incorporates a physics engine to simulate the dynamics of objects, including robots and their interactions with the environment. This enables realistic simulations of robot movements and interactions.

Sensor Simulation: Gazebo supports the simulation of various sensors, such as cameras, lidars, and depth sensors. This allows developers to test and validate sensor integration and perception algorithms in a simulated environment.

Robot Models: Gazebo supports the import and simulation of robot models from various formats, including Unified Robot Description Format (URDF) and Simulation Description Format (SDF). Users can define the physical and visual properties of their robots for accurate simulations.

Plugins: Gazebo can be extended using plugins, which enable users to add custom functionalities or modify existing ones. This flexibility makes it suitable for a wide range of applications.

Scenario and World Editor: Gazebo includes tools for creating and editing simulation scenarios and environments. Users can design custom worlds with various terrains, obstacles, and features to test and evaluate their robots in different conditions.

The simulation world consists of mainly two parts: waves model and the vehicle models including the sensors.

4.1.1.1 Waves Model

Wave Sim project [54] is used for the simulation of the sea environment. The project contains plugins that support the simulation of waves and surface vessels in Gazebo. It has features including Fast Fourier Transform (FFT) wave generation methods, ocean tiling, and support for the Ogre2 render engine. The Computational Geometry Algorithms Library (CGAL) libraries are used for the wave field and model meshes. The Unmanned Underwater Vehicle (UUV) Simulator package for the original vertex shaders used in the wave field visuals. The VMware Remote Console (VMRC) package for textures and meshes used in the wave field visuals.

The model gives an ability as shown in Figure 4.2 to change wind speed, wind angle and steepness by the waves control plugin of the Gazebo Graphical User Interface (GUI). They are used for simulating the system in different sea conditions.

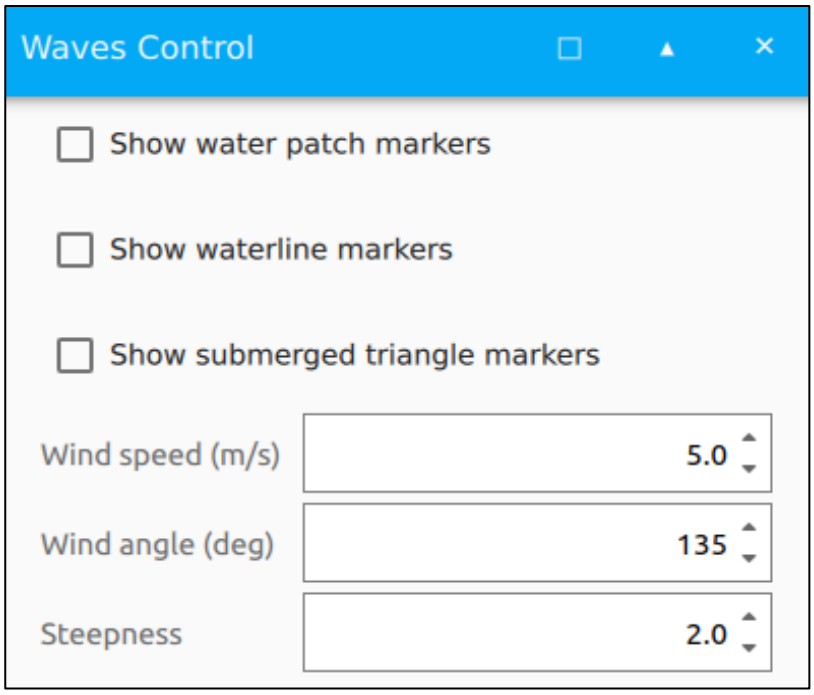

Figure 4.2. Gazebo waves control plugin panel

The wave generation method in the library is based on a study of Tessendorf [55].

In oceanographic literature, Gerstner waves are often considered less realistic compared to statistical models combined with experimental observations for modeling ocean waves. In these statistical models, wave height $h(\boldsymbol{x}, t)$ is treated as a random variable dependent on horizontal position and time. These models rely on decomposing the wave height field into a sum of sine and cosine waves, a method valued for its mathematical and statistical simplicity in model building. Computationally, the decomposition uses FFTs, which are a rapid method of evaluating the sums.

The FFT-based representation of a wave height field expresses the wave height $h(\boldsymbol{x}, t)$ at the horizontal position $\boldsymbol{x} = (x, z)$ as the sum of sinusoids with complex, time-dependent amplitudes:

$$h(\boldsymbol{x}, t) = \sum_k \tilde{h}(\boldsymbol{k}, t) e^{(i\boldsymbol{k}\cdot\boldsymbol{x})} \tag{4.1}$$

where t is the time and $\boldsymbol{k}$ is a two-dimensional vector with components $\boldsymbol{k} = (k_x, k_z)$, $k_x = 2\pi n/L_x, k_z = 2\pi m/L_z$, and n and m are integers with bounds $-N/2 \leq n \leq N/2$ and $-M/2 \leq m \leq M/2$. The FFT process generates the height field at discrete points $\boldsymbol{x} = (nL_x/N, mL_z/M)$. The value at other points can also be obtained by switching to a discrete Fourier transform, but under many circumstances this is unnecessary and is not applied here. The height amplitude Fourier components, $\tilde{h}(\boldsymbol{k}, t)$, determine the structure of the surface. The remainder of this subsection is concerned with generating random sets of amplitudes in a way that is consistent with oceanographic phenomenology.

For computer graphics purposes, the slope vector of the wave-height field is also needed to find the surface normal, angles of incidence, and other aspects of optical modeling as well. One way to compute the slope is through a finite difference between FFT grid points, separated horizontally by some 2D vector Δx. While a finite difference is efficient in terms of memory requirements, it can be a poor approximation to the slope of waves with small wave- length. An exact computation of the slope vector can be obtained by using more FFTs:

$$\epsilon(\boldsymbol{x}, t) = \nabla h(x, t) = \sum_k i\boldsymbol{k}\tilde{h}(\boldsymbol{k}, t)e^{(i\boldsymbol{k}\cdot\boldsymbol{x})} \tag{4.2}$$

In terms of this FFT representation, the finite difference approach would replace the term $i\boldsymbol{k}$ with terms proportional to

$$e^{(i\boldsymbol{k}\cdot\Delta\boldsymbol{x})} - 1 \tag{4.3}$$

which, for small wavelength waves, does not well approximate the gradient of the wave height. Whenever possible, slope computation via the FFT in Eq. (4.2) is the preferred method.

The FFT representation produces waves on a patch with horizontal dimensions L_x x L_z, outside of which the surface is perfectly periodic. In practical applications, patch sizes vary from 10 meters to 2 kilometers on a side, with the number of discrete sample points as high as 2048 in each direction (i.e. grids that are 2048 × 2048, or over 4 million waves). The patch can be tiled seamlessly as desired over an area. The

consequence of such a tiled extension, however, is that an artificial periodicity in the wave field is present. If the patch size is large compared to the field of view, this periodicity is unnoticeable. Also, if the camera is near the surface so that the effective horizon is one or two patch lengths away, the periodicity will not be noticeable in the look-direction, but it may be apparent as repeated structures across the field of view.

Oceanographic research [55] has demonstrated that Eq. (4.1) is a reasonable representation of naturally occurring wind-waves in the open ocean. Statistical analysis of a few wave-buoy, photographic, and radar measurements of the ocean surface demonstrates that the wave height amplitudes $\tilde{h}(\boldsymbol{k}, t)$ are nearly statistically stationary, independent, gaussian fluctuations with a spatial spectrum denoted by:

$$P_h(\boldsymbol{k}) = \langle \left| \tilde{h}^*(\boldsymbol{k}, t) \right|^2 \rangle \tag{4.4}$$

for data-estimated ensemble averages denoted by the brackets $\langle\ \rangle$.

There are several analytical semi-empirical models for the wave spectrum Ph (k). A useful model for wind-driven waves larger than capillary waves in a fully developed sea is the Phillips spectrum:

$$P_h(\boldsymbol{k}) = A \frac{e^{-\frac{1}{(kL)^2}}}{k^4} \left| \hat{\boldsymbol{k}} \cdot \hat{w} \right|^2, \tag{4.5}$$

where $L = V^2/g$ is the largest possible waves arising from a continuous wind of speed V, g is the gravitational constant, and $\hat{w}$ is the direction of the wind. A is a numeric constant. The cosine factor $\left| \hat{\boldsymbol{k}} \cdot \hat{w} \right|^2$ in the Phillips spectrum eliminates waves that move perpendicular to the wind direction. This model, while relatively simple, has poor convergence properties at high values of the wave number $|\boldsymbol{k}|$. A simple fix is to suppress waves smaller that a small length $l \ll L$, and modify the Phillips spectrum by the multiplicative factor:

$$e^{-k^2 l^2}\ . \tag{4.6}$$

4.1.1.2 Vehicle Models

4.1.1.2.1 Ship

The utilized surface vessel is the Havyard 842 tug model [54], specifically designed with a spacious deck area capable of accommodating a quadplane. This vessel, boasting a length of approximately 90 meters, is equipped with two motors serving as thrusters, a steering mechanism. The model, as shown in Figure 4.3, has an ArduPilot plugin [56] that encompasses an IMU simulation system.

To facilitate vision-based position control, an AprilTag image is strategically placed on the deck of the model. This AprilTag image plays a crucial role in aiding the vision-based position controller in accurately determining and controlling the position of the vessel.

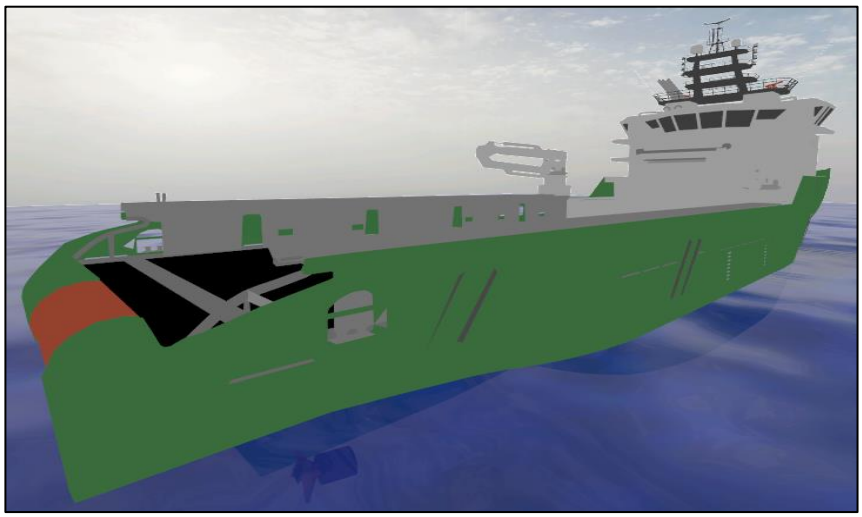

Figure 4.3. Havyard vessel model on Gazebo

4.1.1.2.2 Quadplane

Alti Transition Quadplane model [57] is used for the simulation. The model has five rotors, four for the quadcopter configuration and one for the fixed wing configuration. Also, ailerons and elevator functions are capable for the roll and pitch movements of the fixed wing plane configuration. Model, as seen in Figure 4.4, has an ArduPilot plugin including IMU simulation.

The camera and laser rangefinder sensors are also attached for the control algorithms.

Camera captures images from the simulation environment and the images can be viewed by the image display plugin of GUI. Camera resolution is 320x240, field of view (FOV) is 1.47 and frame rate is 10Hz. All these specifications could be arranged with the camera model. Current resolution and FOV values may be increased in real world experiments, however they are sufficient for the simulation.

Rangefinder can measure from 15cm to 50m distances with 1cm accuracy. Laser rays can be observed in the simulation environment.

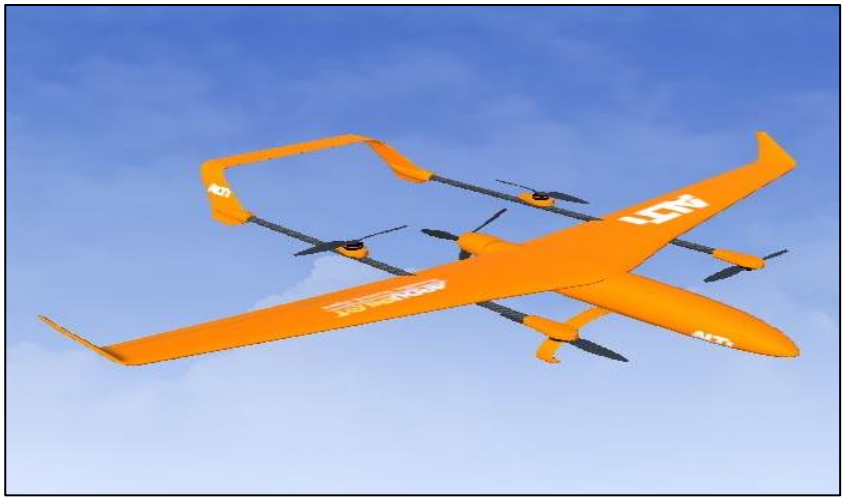

Figure 4.4. Alti transition quadplane model on Gazebo

4.1.2 ArduPilot Software in the Loop (SITL)

ArduPilot SITL serves as autopilot software which has same features as on the hardware autopilot firmware, offering extensive capabilities for both aerial and surface vehicles. ArduPilot's built-in flight controller utilizes a cascaded PID control system for angle rate, velocity, and position inputs. The PID controllers are optimized for each quadcopter and fixed wing configurations of the quadplane and the ship as a type of rover. SITL instance supplies UAV states to the environment simulator, Gazebo, while gathering sensor data and environment state.

In the simulation, two simultaneous ArduPilot SITL instances operate concurrently, one dedicated to the quadplane (with a system ID of 1) and the other to the ship (with a system ID of 17).

The ship conveys emulated global position data from Gazebo to the GCS, and the GCS subsequently relays this position information to the quadplane.

Ship landing script appears to be a part of the control logic for a quadplane, specifically tailored for landing on a moving ship. The script aims to ensure that the VTOL vehicle remains vertically aligned above the takeoff or landing pad. Here are key points related to the described functionality:

VTOL Tracking during Takeoff and Landing: The script is likely part of a larger system that involves a VTOL vehicle capable of tracking the motion of a platform during both takeoff and landing procedures.

Vertical Alignment Above Pad: The primary goal is to maintain vertical alignment above the designated takeoff or landing pad. This is crucial for safe and controlled operations.

Tracking Hold-Off Loitering Point: During landing, the script appears to establish a "tracking hold-off loitering point." This point serves as a reference for the VTOL vehicle during the landing phase, ensuring that it maintains a safe and controlled distance from potential obstructions, such as vehicle superstructures.

Controlled Angular Approach: The script mentions a "controlled angular approach" during landings. This implies that the VTOL vehicle is programmed to approach the landing pad at a specific angle, likely to avoid obstacles or structures on the platform.

Superstructure Avoidance: The angular approach is particularly designed to avoid vehicle superstructures, or any other obstructions present on the platform. This is crucial for preventing collisions and ensuring the safety of the landing operation.

Automatic Offset Calculation: The script includes functionality for automatically calculating and setting the follow offset based on the current position of the VTOL vehicle and the landing target. This feature contributes to precise alignment during the landing phase.

Altitude Control and Throttle Positioning: Altitude control is a key aspect during landing, and the script incorporates logic for reaching and maintaining the target altitude. Throttle positioning is dynamically adjusted based on the VTOL vehicle's state, including whether it is in hold-off or descent stages.

Error Handling and Display: The script includes error handling mechanisms to detect and display internal errors. This is important for diagnosing issues during execution.

Autonomous Landing Modes: The script mentions different autonomous landing modes, such as MODE_RTL (Return to Launch Mode) and MODE_QRTL. These modes likely define specific behaviors during landing operations.

Parameter Definitions and Setup: Various parameters are defined and initialized to configure specific behaviors of the VTOL system, including landing activation, landing angle, and automatic offset triggers.

In summary, the provided implementations of control algorithms contribute to the development of a robust control system for VTOL takeoffs and landings. It focuses on maintaining precise alignment, altitude control, and obstacle avoidance during both takeoff and landing phases on a moving platform.

While the ship landing script excels in simulations, real-world challenges like inaccurate GPS and communication delays require addressing. Leveraging camera feedback and laser rangefinder measurements can enhance precision in positioning and altitude control. Sensor fusion techniques, combining data from various sources, improve reliability. Optimizing communication protocols and implementing backups further fortify the system. Extensive real-world testing is crucial for refining and ensuring the ship landing system's reliability in practical applications.

4.2 Control Algorithms

4.2.1 Horizontal Position Control

The AprilTag system, known for its cost efficiency, serves as a visual fiducial system in our application. It can be implemented on a straightforward single-board computer, such as a Raspberry Pi, paired with a basic camera. In our specific setup, a Gazebo camera model is used, and the coding for tag detection and position offset calculation is performed using Python on the simulation computer. Dronekit library [58] supplied the communication between the algorithm and the SITL.

During simulation, the Python code detects AprilTags in the Gazebo camera model's images and calculates position offsets. These offsets are then transmitted to the quadplane's STIL instance. Subsequently, the offsets are adjusted according to the quadplane's heading and are employed in a PID controller to correct the horizontal position. Finally, they are given to the autopilot as ship following offsets of the x and y axes.

This process creates a closed-loop feedback system where the AprilTag information obtained from the simulated environment influences the positioning of the quadplane. The use of a PID controller ensures that the corrections are proportionate to the observed position offsets, contributing to stable and precise horizontal positioning during simulated operations.

4.2.2 Descent Rate Control

A laser rangefinder model has been integrated into the quadplane model. During simulation, the quadplane's height from the ship deck is measured in Gazebo environment and transmitted to the quadplane SITL instance. Descent rate control is managed by a PID controller, implemented using Lua scripts within the SITL. It facilitates a controlled and gradual descent, ensuring a smooth landing with a slow touch down onto the ship deck.

4.3 Results

The simulation results [59] validate the effectiveness of the integrated system, encompassing Gazebo, ArduPilot SITL, vision-based position control, and descent rate control mechanisms. The simulation environment accurately replicates real-world scenarios, providing a reliable platform for testing and refining the autonomous landing system for VTOL UAVs on a ship deck in standing wavy seas.

Figure 4.5 illustrates the screenshot of the beginning of the simulation.

In the Gazebo window (left side of the screen), the ship is sailing on the water with the tag which is located at (-20, 0, 1.5) m according to the ship autopilot. The quadplane is located at (-22, -2, 1.5) m according to the ship autopilot, to prove the horizontal position control algorithm is working. Also, live camera view shows the ship deck for now. The waves control panel shows the wind speed as 7.5m/s and direction is 0 degree.

On the right upper corner of the screen includes the MavProxy communication with the quadplane with system id =1. (Sys:1). The map view on the mid right side of the screen shows the ship with id=17 and quadplane on it with id=1. Their speed is zero in the beginning.

The camera reader python code is working on the right lower corner of the screen showing the vehicle (quadplane) connected and waiting for the images to detect.

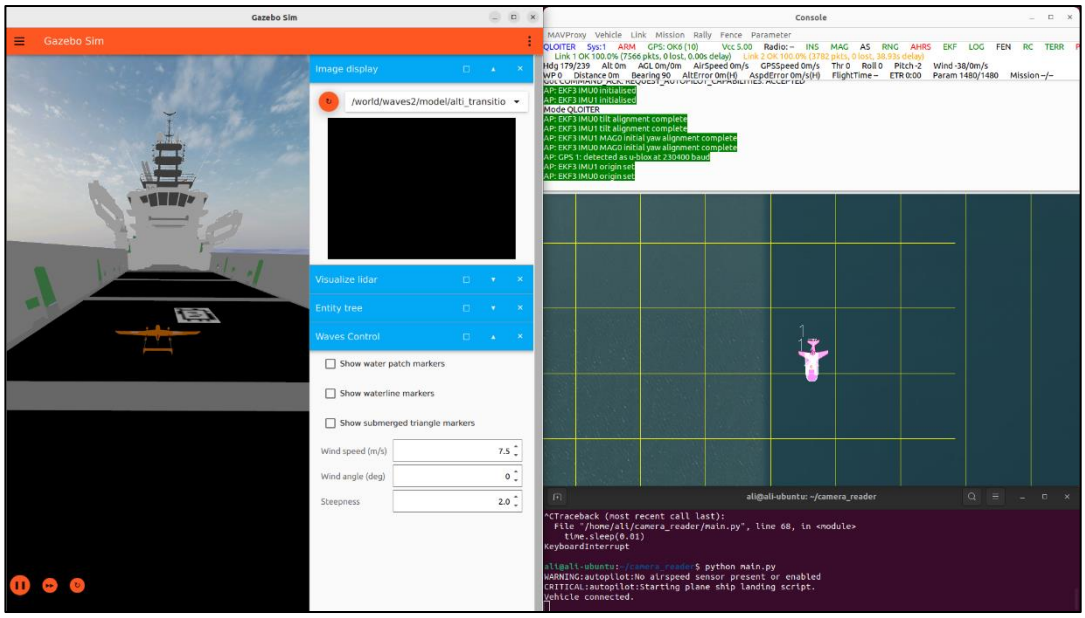

Figure 4.5. First stage of the simulation

Figure 4.6 shows the second stage of simulation. Ship is sailing at 5m/s speed on the water carrying the quadplane on its deck.

In the MavProxy console, it is shown the quadplane is in Auto mode which is indicated on the left upper corner of the console with blue. It is ready to be armed which can be seen with the console message “pre-arm good”.

The graph shows the last 50 seconds of the altitudes of quadplane. The red line shows the constant 0.13m which is the height of the rangefinder from the ship deck. The green line shows the altitude of the quadplane according to Mean Sea Level (MSL) which is oscillating about 2m with varying amplitude and frequency.

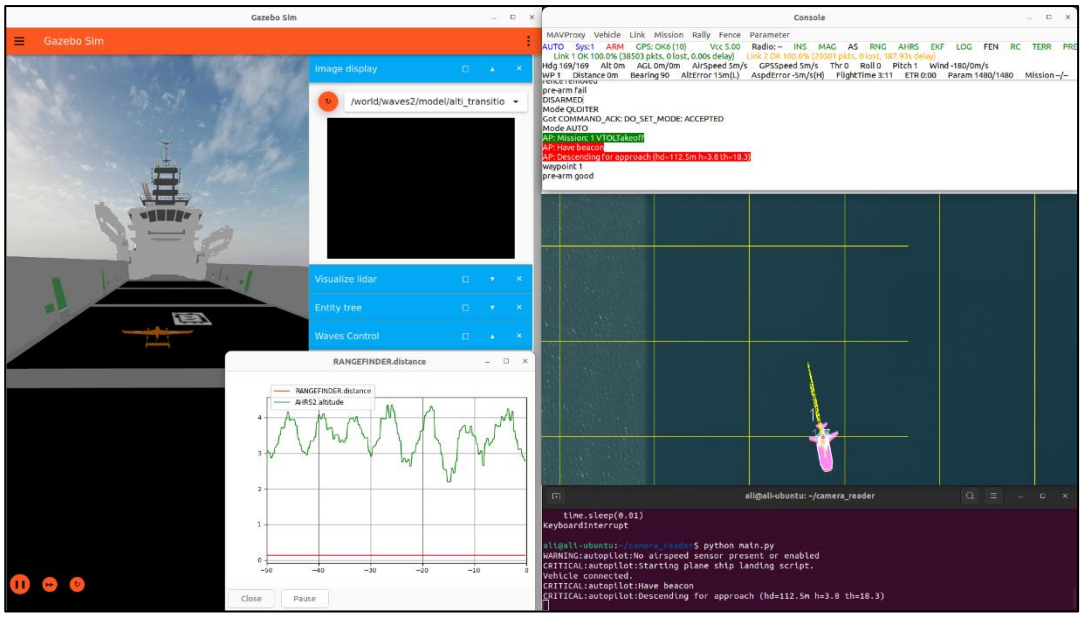

Figure 4.6. Second stage of the simulation

Quadplane is armed and succeed takeoff from the ship deck as shown in Figure 4.7. In the MavProxy console, it is shown the quadplane is switched to RTL mode after completed the takeoff mission at 15m altitude. The graph shows the increasing altitudes during the takeoff mission. In gazebo screen, rangefinder is pointing the takeoff point with a blue laser, and Image display panel shows the image of ship deck including the tag. Camera reader code detects the tag from the image and calculates the offsets as ratio errors for the x and y axes. The offsets are not used right now before it is not the landing stage of the quadplane.

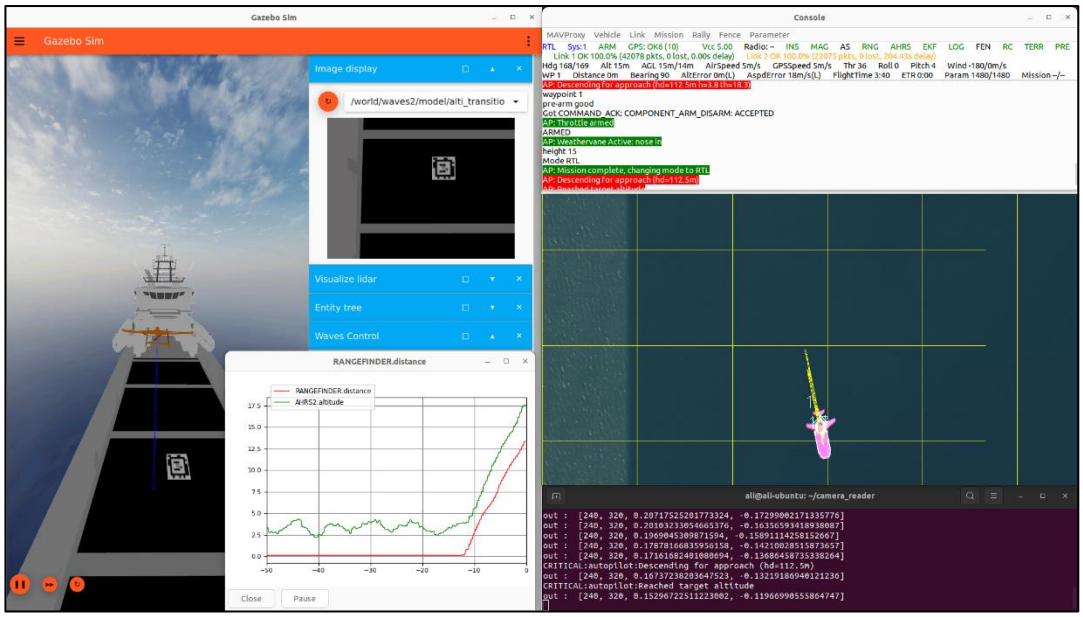

Figure 4.7. Quadplane takeoff from the ship deck

In the RTL mode, the quadplane tracks the ship and aligns itself with the desired approach angle to the ship which is zero for our case. It means quadplane approaches from the back side of the ship. When the desired altitude and desired distance is succeeded, the quadplane switches its mode to the QRTL.

In Figure 4.8, the first part of the landing can be observed.

In the graph, the rangefinder altitude is constant at 50m for the first 30 seconds which means the sensor is out of range because the laser rangefinder cannot work on the water. At -20 seconds, the quadplane comes over the ship while flying to landing point. The next 10 seconds the quadplane comes to the initial landing point which is located with offset (-2, -2) m from the desired landing point which tag is located on there. At nearly -10 seconds, the tag can be observed with the camera. The horizontal position control started the work.

Camera reader code detects the tag and sends the offsets to the autopilot. In the MavProxy console it is seen that FOLL_OFS_X and FOLL_OFS_Y parameters are changed according to calculated offsets.

In Gazebo screen, the quadplane hovers over the exact landing position as the proof of the horizontal position corrections.

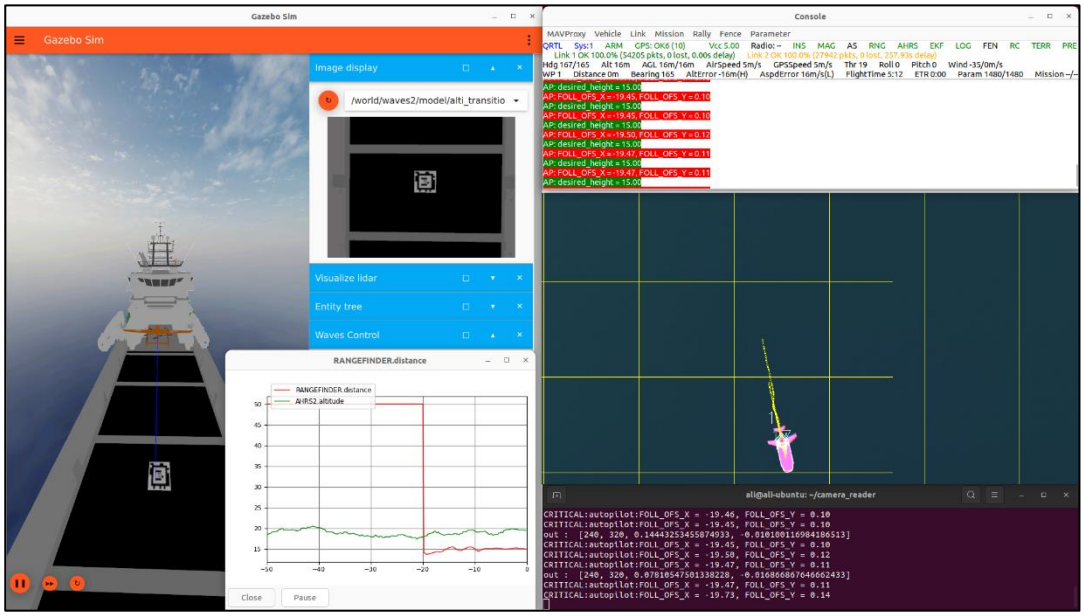

Figure 4.8. Quadplane horizontal position correction

After 10 seconds passed from the beginning of the suggested algorithm started, the quadplane starts to decrease the altitude. Figure 4.9 demonstrates the second part of the landing stage.

The graph in the figure shows the decrease in altitudes. While the MSL altitude of the quadplane is wavy, altitude Above Ground Level (AGL) from the rangefinder is changing nearly a constant rate which is the proof of the descent rate control algorithm is working. They will be analyzed in detail later in this part.

Also, horizontal position correction is still working simultaneously.

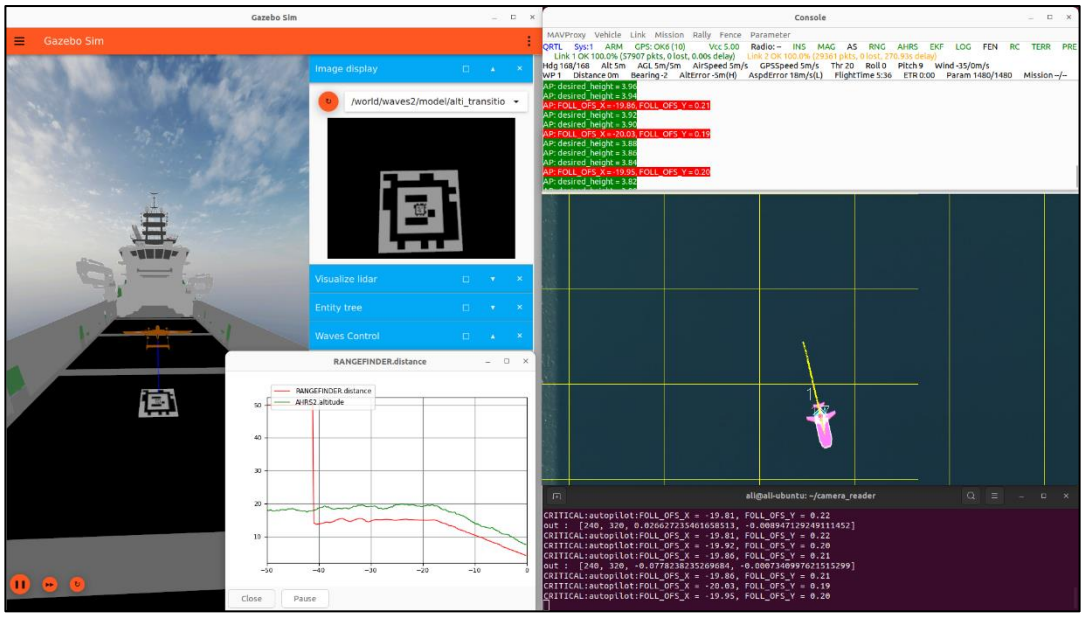

Figure 4.9. Quadplane landing to the ship deck

Figure 4.10 includes the final landing stage. At about -42 seconds, the descent rate decreased for smoother landing. The landing completed successfully while the ship was still sailing at the speed of 5m/s.

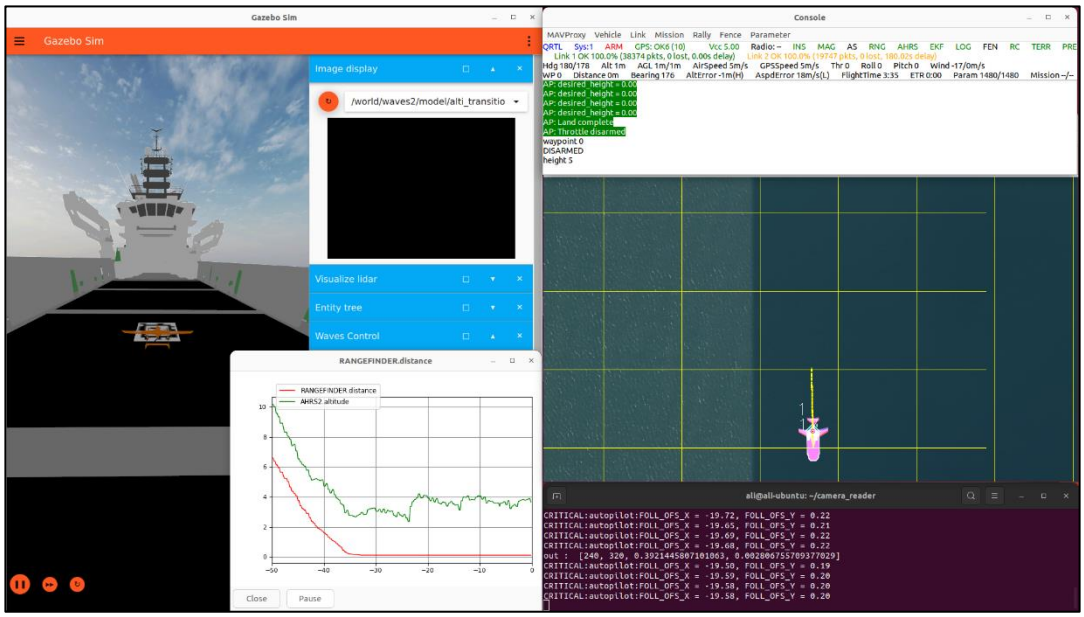

Figure 4.10. Landing completed success

An additional flight simulation was conducted without the suggested algorithm, maintaining the same wind conditions and ship speed for comparison. The log files from this simulation, as well as the one with the algorithm, were obtained using Mission Planner software. The necessary data from these logs were exported and analyzed using MATLAB to produce the subsequent graphics, which provided insights into the performance differences between the simulations. This comparison aimed to highlight the effectiveness of the proposed algorithm in enhancing the UAV's autonomous landing capabilities.

Firstly, the simulation without the suggested algorithm (initial simulation) is analyzed. Figure 4.11 shows the differences of altitude of ship, measured as MSL, altitude of quadplane MSL and altitude of quadplane AGL. The values are only from the landing stage of the flight. Ship altitude oscillates about 2m due to waves. After landing completed, it is clearly seen that quadplane altitude MSL is changing in synchronous with the ship altitude.

During the flight, quadplane altitude AGL which means the distance between the quadplane and the ship is oscillating about 2m such as the oscillation of the ship while the quadplane altitude MSL is changing with smaller differences.

The quadplane has no information about the distance between itself and the ship and decreases the altitude MSL constantly which causes the rapid changes in the distance between itself and the ship. In the final, while the quadplane decreases its altitude, the ship altitude increases and hits the quadplane. Figure 4.12 shows the Z axis acceleration during the flight and the landed part. The acceleration value is about -200m/s²which is a very high value at about 273 seconds. This is a very rough touchdown which most probably can cause huge damage and malfunction.

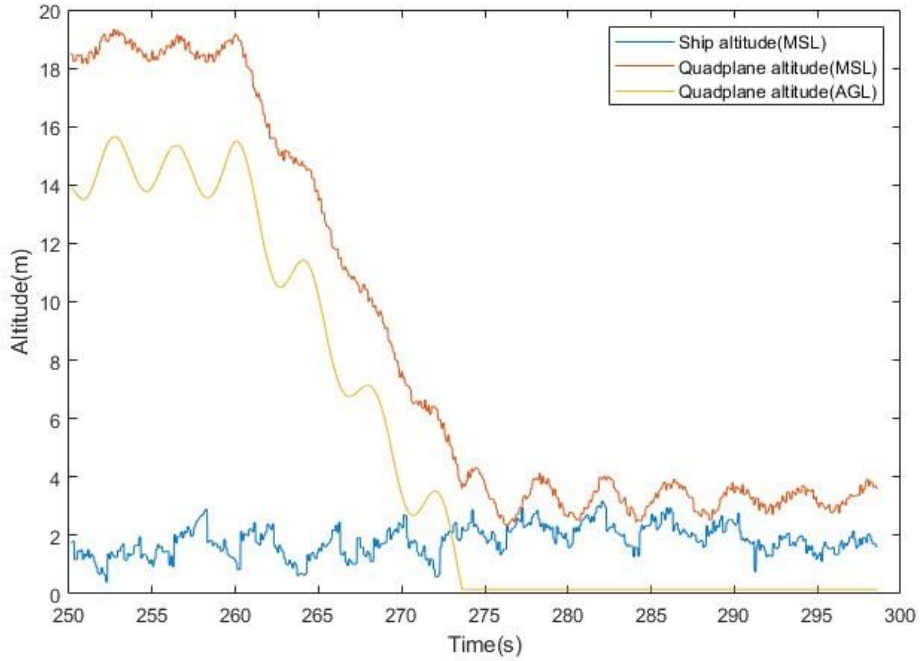

Figure 4.11. Altitudes of the quadplane and ship from the initial simulation

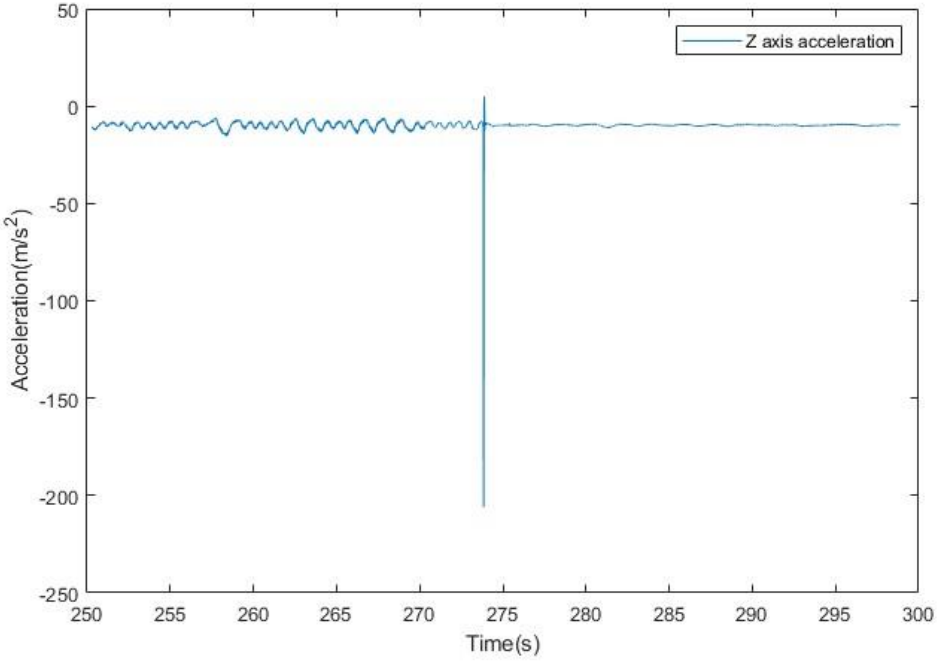

Figure 4.12. Z axis acceleration from the initial simulation

Secondly, the simulation with the suggested algorithm (final simulation) is analyzed. Figure 4.13 shows the differences of altitudes of ship MSL, quadplane MSL and quadplane AGL. The values are only from the landing stage of the flight again. Ship altitude oscillates about 2m due to waves. After landing is completed, it is clearly seen that quadplane altitude MSL is changing in synchronous with the ship altitude.

During the flight, quadplane altitude MSL which means the distance between the quadplane and the ship is oscillating about 2m such as the oscillation of the ship while the quadplane altitude AGL is changing at nearly a constant rate thanks to suggested descent rate algorithm contrary to the initial simulation.

The quadplane measures the distance between itself and the ship and decreases the altitude AGL constantly which causes very small changes in the distance between itself and the ship. In the final, algorithm decreases the descent rate for a softer touchdown. Figure 4.14 highlights the Z-axis acceleration during flight and landing, showing a significant reduction in acceleration values (about -16m/s²) compared to the initial simulation (about -200m/s²). This lower acceleration value indicates a smoother and safer landing, underscoring the effectiveness of the descent rate control algorithm.

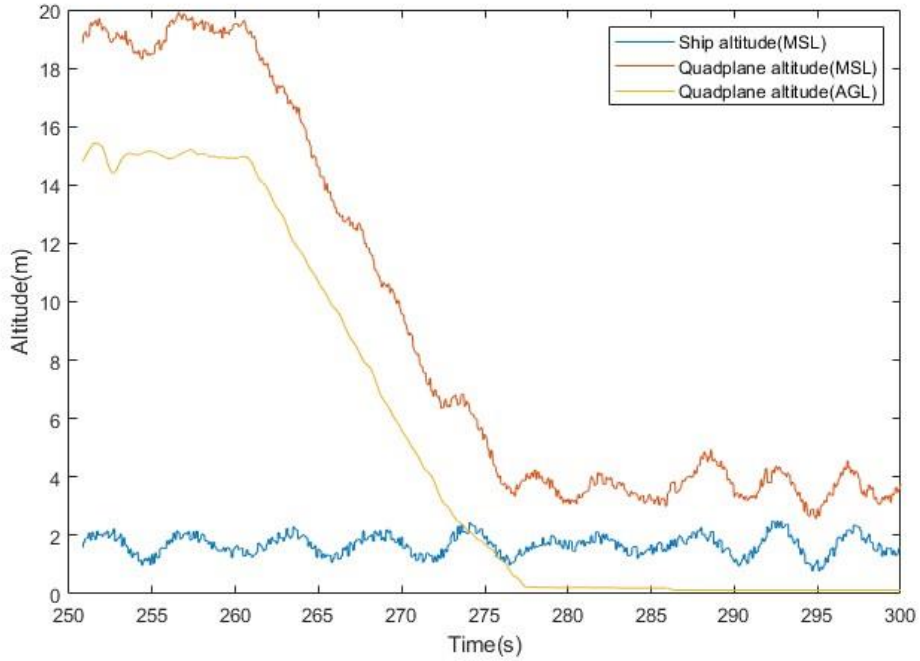

Figure 4.13. Altitudes of the quadplane and ship from the final simulation

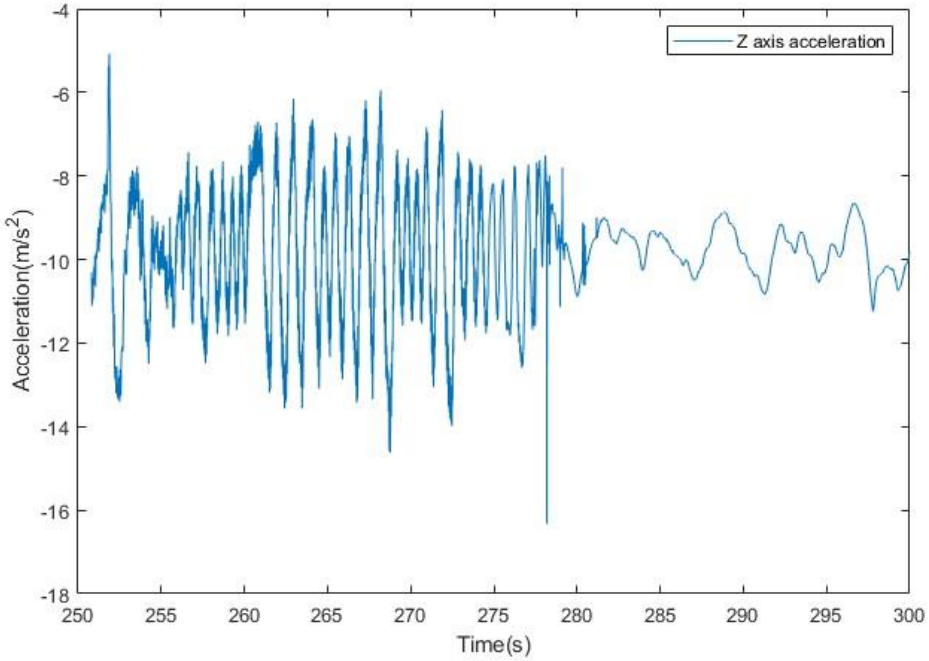

Figure 4.14. Z axis acceleration from the final simulation

The analysis of throttle commands applied by the autopilot to control the quadplane's vertical altitude reveals significant differences between the initial and final simulations. In the initial simulation, as shown in Figure 4.15, the throttle control was relatively relaxed, characterized by small amplitude and frequency, leading to oscillations in the quadplane's altitude relative to the ship deck. However, with the implementation of the proposed descent rate control algorithm in the final simulation, throttle responses became more aggressive and responsive to the altitude changes caused by wave-induced ship movements. This is evidenced by the rapid throttle reactions and higher amplitude differences observed in the final simulation, as detailed in Figure 4.16.

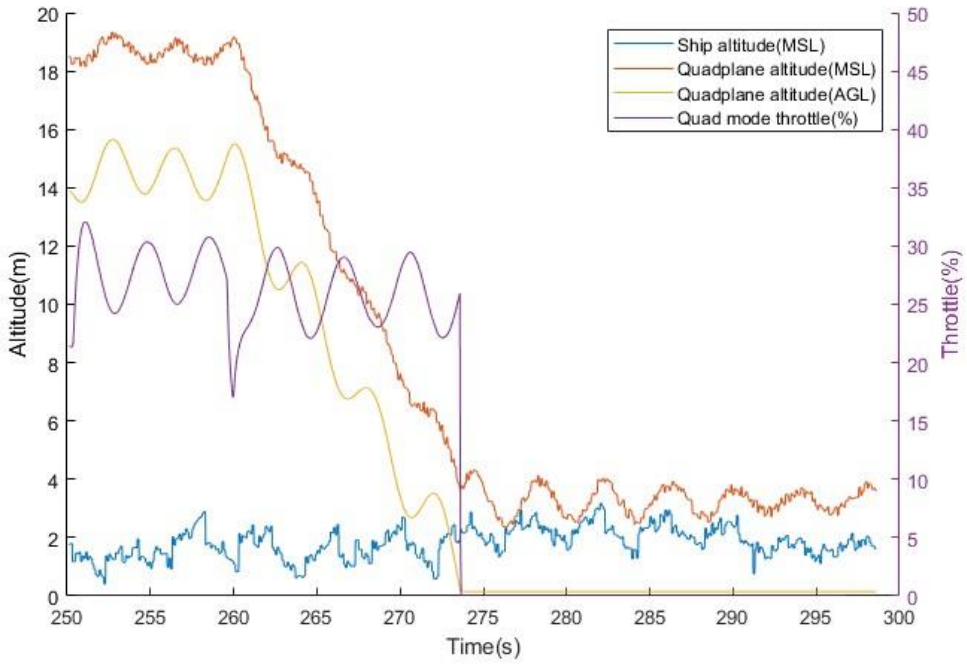

Figure 4.15. Quad mode throttle percentage of the quadplane with altitudes from the initial simulation

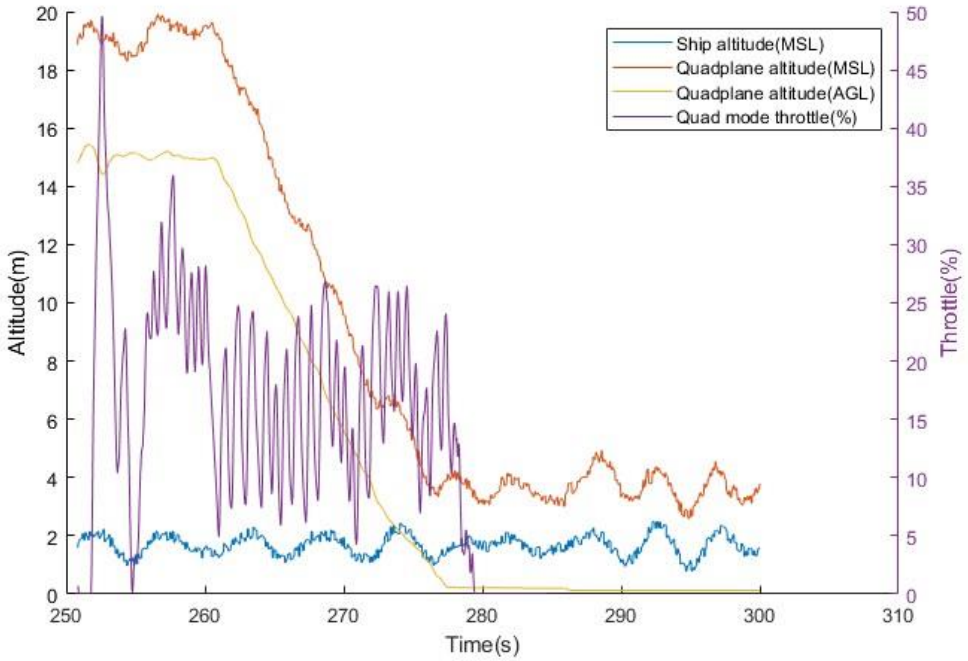

Figure 4.16. Quad mode throttle percentage of the quadplane with altitudes from the final simulation

CHAPTER 5

EXPERIMENTS

5.1 Experiment Setup

5.1.1 Quadplane

In a similar way in the thesis study by Çakıcı [60], the FPV Raptor model airplane by Volantex RC was selected for its traditional airplane design, featuring a pusher propulsion system, ailerons, rudder, and elevator. Key advantages include its sturdy fuselage and large internal space for modifications to add avionics, companion computer.

The base platform was enhanced for VTOL capabilities by adding a quad-propulsion system. An H-shaped frame is made of carbon tubes where the motors are fixed at the ends. These motors are arranged to rotate the propellers in alternating clockwise and counterclockwise directions. The frame is attached beneath the fuselage to maintain the center of gravity, resulting in the final quadplane configuration as depicted in the document. The initial and final configuration is shown in Figure 5.1.

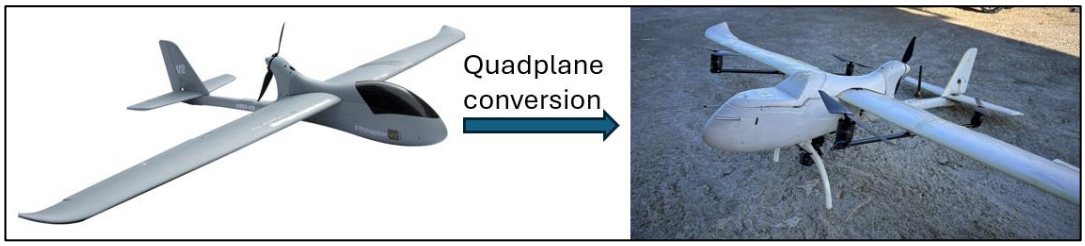

Figure 5.1. Quadplane conversion of a model airplane (Volantex FPV Raptor V2)

Quad-propulsion system includes four brushless DC motors (Tmotor AIR2216/KV880) and four in one Electronic Speed Controller (ESC) (Tmotor F55A Pro II) for Pulse Width Modulation (PWM) control of the motors as shown in Figure 5.2.

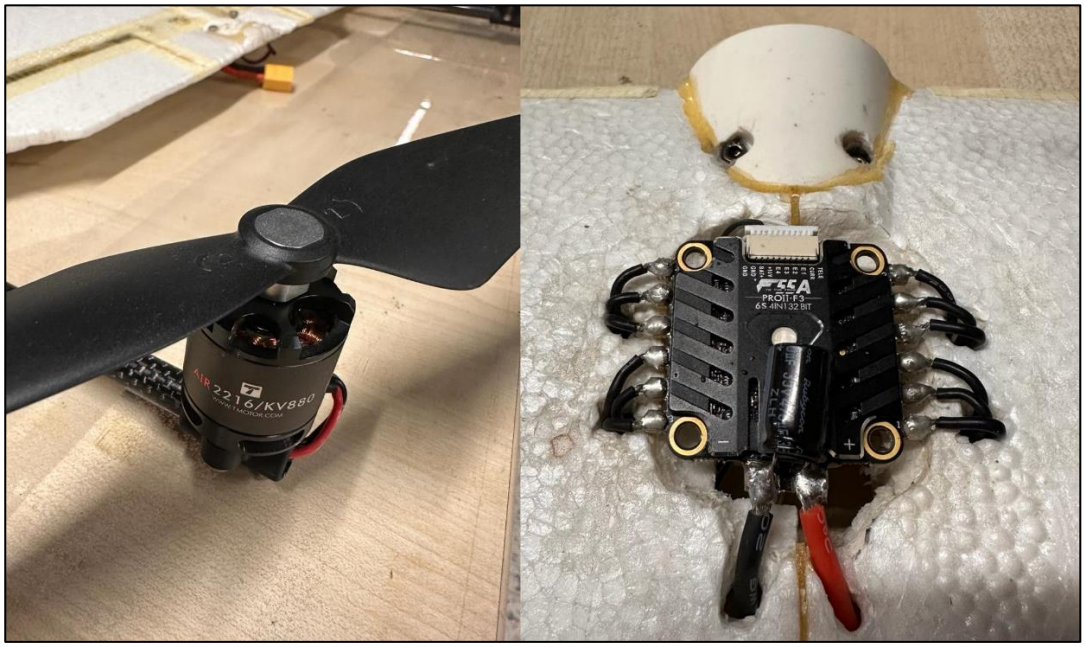

Figure 5.2. Quad-propulsion system motor and ESC

As autopilot hardware, Pixhawk Cube Orange is installed in the middle of the fuselage, under the wing as shown in Figure 5.3. The autopilot is responsible for the autonomous flight of quadplane including the IMU and barometer sensors. ArduPilot firmware with the proposed control algorithm implementations run on the autopilot.

Figure 5.3. Autopilot hardware (Pixhawk cube orange) integration

For the autonomous flight, quadplane needs to get global position and to measure ground speed and airspeed. For the global positioning and ground speed a GPS receiver (Pixhwak here) is installed on the quadplane. Also, for the airspeed measurements two airspeed sensors (Sensirion SDP33 and Matek ASPD-DLVR) are added. Two airspeed sensors with different communication protocols, I2C and CAN, are used to make a redundant system. Figure 5.4 shows the GPS receiver, airspeed sensors and pitot tubes installed on the quadplane.

Also as shown in Figure 5.5 a 433MHz telemetry module (3DR radio telemetry) is installed at the tail part of fuselage. It is used for communication with GCS to send commands and receive telemetry data.

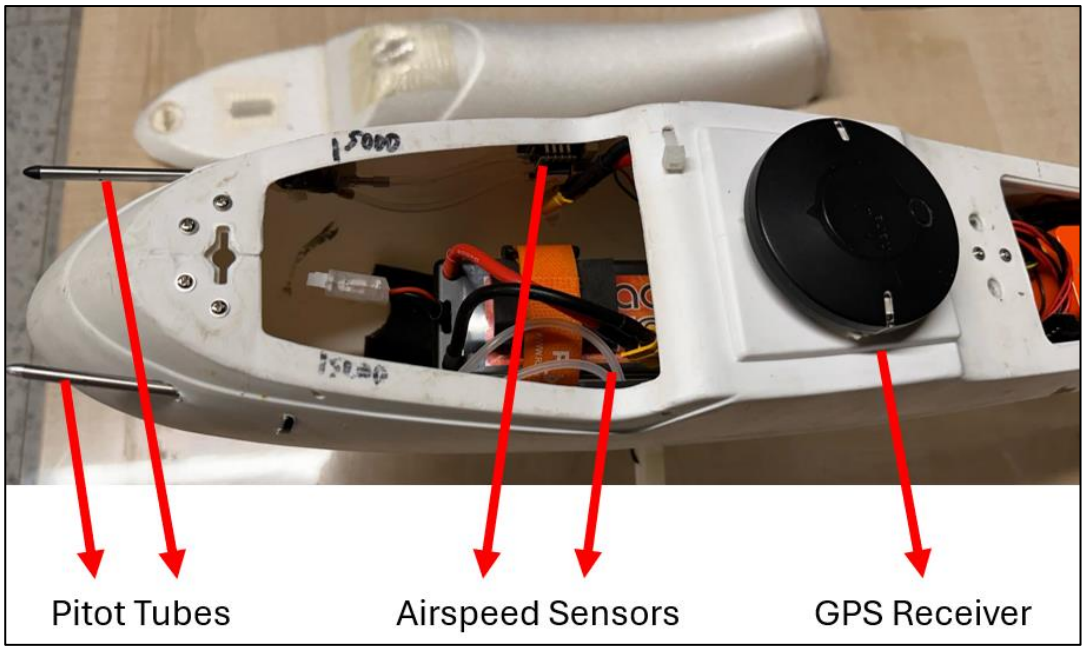

Figure 5.4. Pitot tubes, airspeed sensors and GPS receiver installed on the quadplane

Figure 5.5. 3DR 433MHz radio telemetry with antenna

For the proposed control algorithms, a camera, a rangefinder, and a companion computer are needed. As camera SIYI A8 mini gimbal camera is chosen. It is a 95g lightweight 4K 8MP camera with gimbal supporting ethernet connection. TFmini-S Lidar sensor is used as a rangefinder to measure height between the quadplane and the landing point. It can measure the distance to an object as close as 10 centimeters (+/- 6cm up to 6m) and as far as 12 meters (+/-1% starting at 6m). Raspberry Pi 4 Model B is used as companion computer to get images from the camera and calculate the position offsets for the proposed horizontal position control algorithm. Figure 5.6 shows these components added at the bottom of the fuselage.

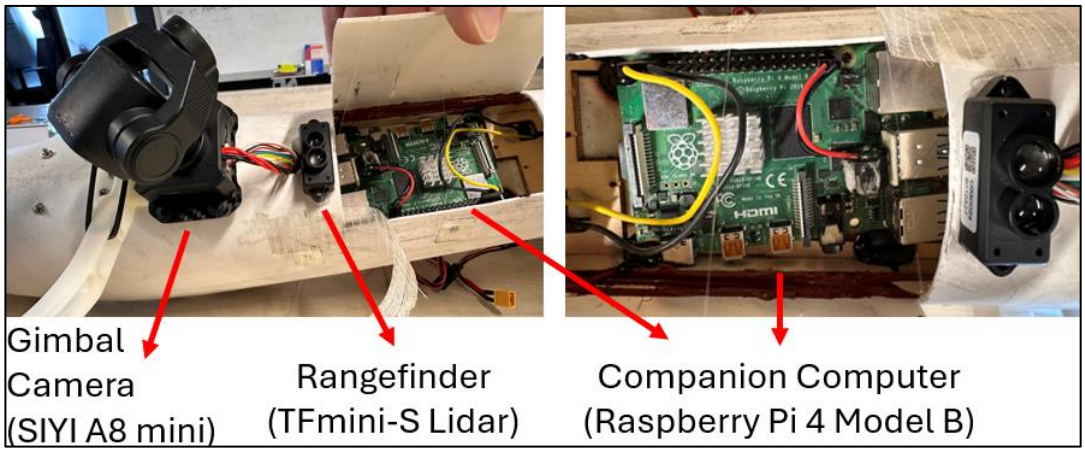

Figure 5.6. Camera, rangefinder and companion computer installations

5.1.2 Ship (Remote Controlled Electric Ride on Truck)

For the ship role of the experiments a commercial remote controlled (RC) electric kids ride on truck is used as shown in Figure 5.7. A 2m x 2m aluminum platform is constructed and the same size proposed AprilTag is mounted on it to be the landing platform for the quadplane's takeoff and landing operations.

Figure 5.7. RC truck for the ship role

Figure 5.8 shows the proposed AprilTag as landing platform installation on the RC truck. The beacon is installed on the platform. The beacon includes an autopilot (Pixhawk Cube Black) and a GPS receiver (Pixhawk Here). Also, there is a telemetry module (3DR 915MHz radio telemetry) with an antenna for communication with GCS.

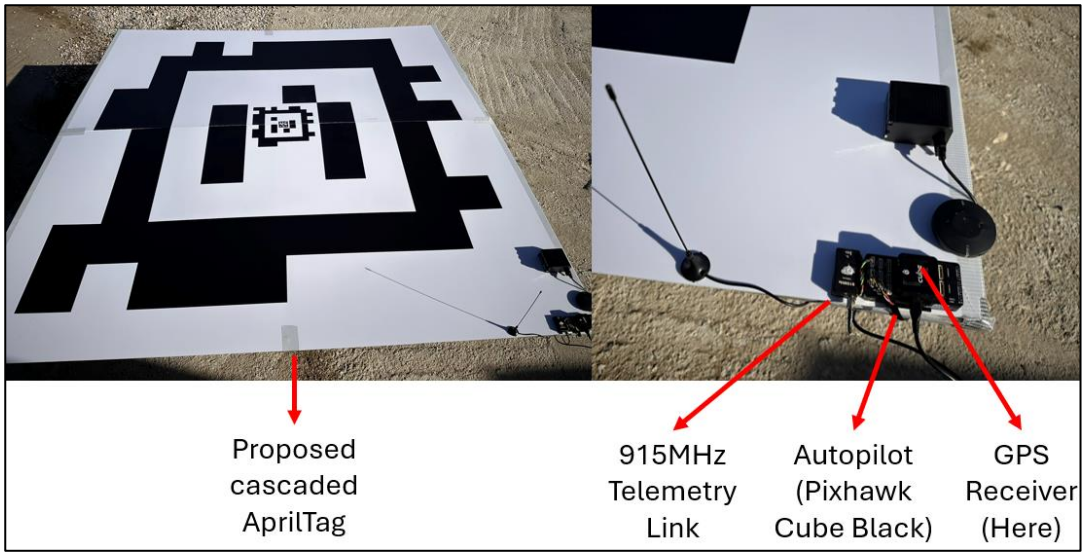

Figure 5.8. Proposed AprilTag and beacon (Autopilot and GPS receiver) installation

5.1.3 GCS

Figure 5.9 shows the GCS setup for the real-world flights. A laptop computer is used as GCS. MavProxy software is running on it. MavProxy is connected to quadplane using 433MHz telemetry module and ship (RC truck) using 915MHz telemetry module. The main task of GCS is forwarding the position data of the ship to the quadplane for tracking. Also, GCS shows the real time telemetry data of the quadplane and the ship.

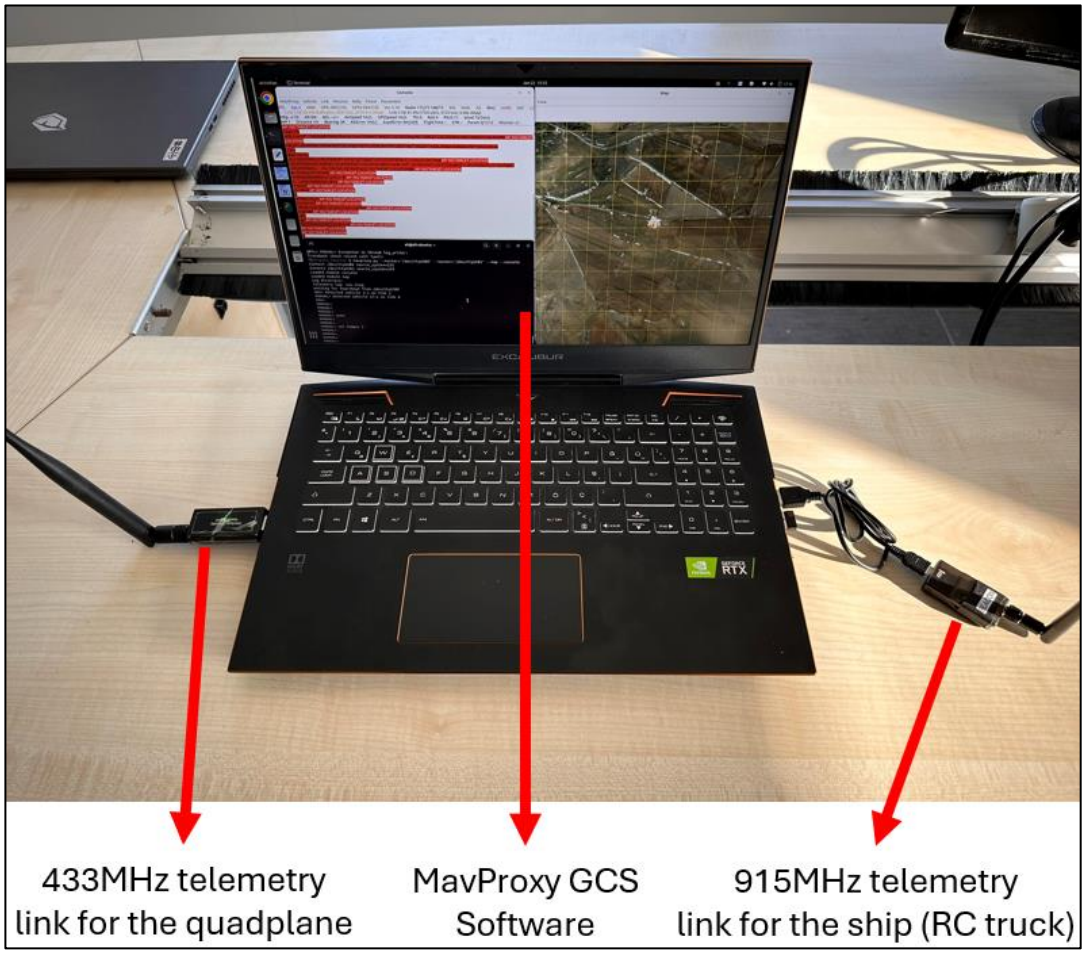

Figure 5.9. GCS setup with telemetry modules

5.2 Flights

Two real world flights are performed to validate the supported algorithms are working. The first flight [61] is performed for the vision-based position control algorithm on a fixed landing platform. The second flight [62] is performed to prove the complete quadplane ship landing algorithm is working.

5.2.1 First Flight

Figure 5.10 shows the camera view of the quadplane at about 30m height without the control algorithm implementation. Proposed cascaded AprilTag is placed in the middle of an 8m x 8m sized landing area.

Figure 5.10. Camera view from the quadplane

The outer AprilTag id is 27 and the inner AprilTag id is 28. While the vision based autonomous landing of the quadplane, Figure 5.11 shows that the outer AprilTag is detected at the height about 30m (Z: 30.8713). The horizontal offsets are calculated as X: 0.4547m and Y: -0.2536m to be used in the control algorithm.

Figure 5.11. Camera view of the AprilTag tag detection from the quadplane at 30m height

In Figure 5.12, again only outer AprilTag is detected at 20m height while quadplane descending and adjusting its position using the offsets calculated with the control algorithm implemented.

Figure 5.12. Camera view of the outer AprilTag detection from the quadplane at 20m height

As shown in Figure 5.13, at 10m height of the quadplane, both outer (id: 27) and inner (id: 28) AprilTags are detected and offsets according to each AprilTags calculated separately. It can be observed that the offsets are so closed which means detection algorithm works well. Also, they are used to maintain precise horizontal positioning while descending.

Figure 5.13. Camera view of the outer and inner AprilTag detection from the quadplane at 10m height

At the final stage of landing, only inner AprilTag detection is succeeded at 1.3m height of the quadplane as seen in Figure 5.14. Then, landing completed successfully on the exact desired landing point.

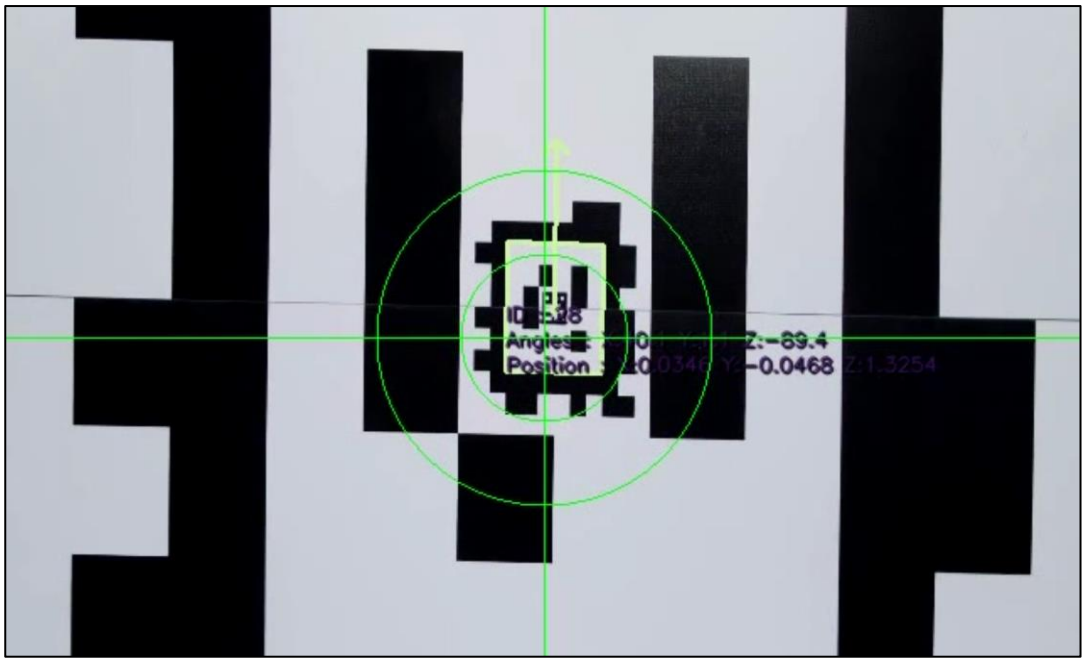

Figure 5.14. Camera view of the inner AprilTag detection from the quadplane at 1.3m height

5.2.2 Second Flight

RC ride on truck is used for the ship role for the flight tests. As shown in Figure 5.15, an autonomous takeoff and landing mission is accomplished while the truck is moving. The flight proves the quadplane tracks the moving platform and succeeds in the precise landing on the desired position.

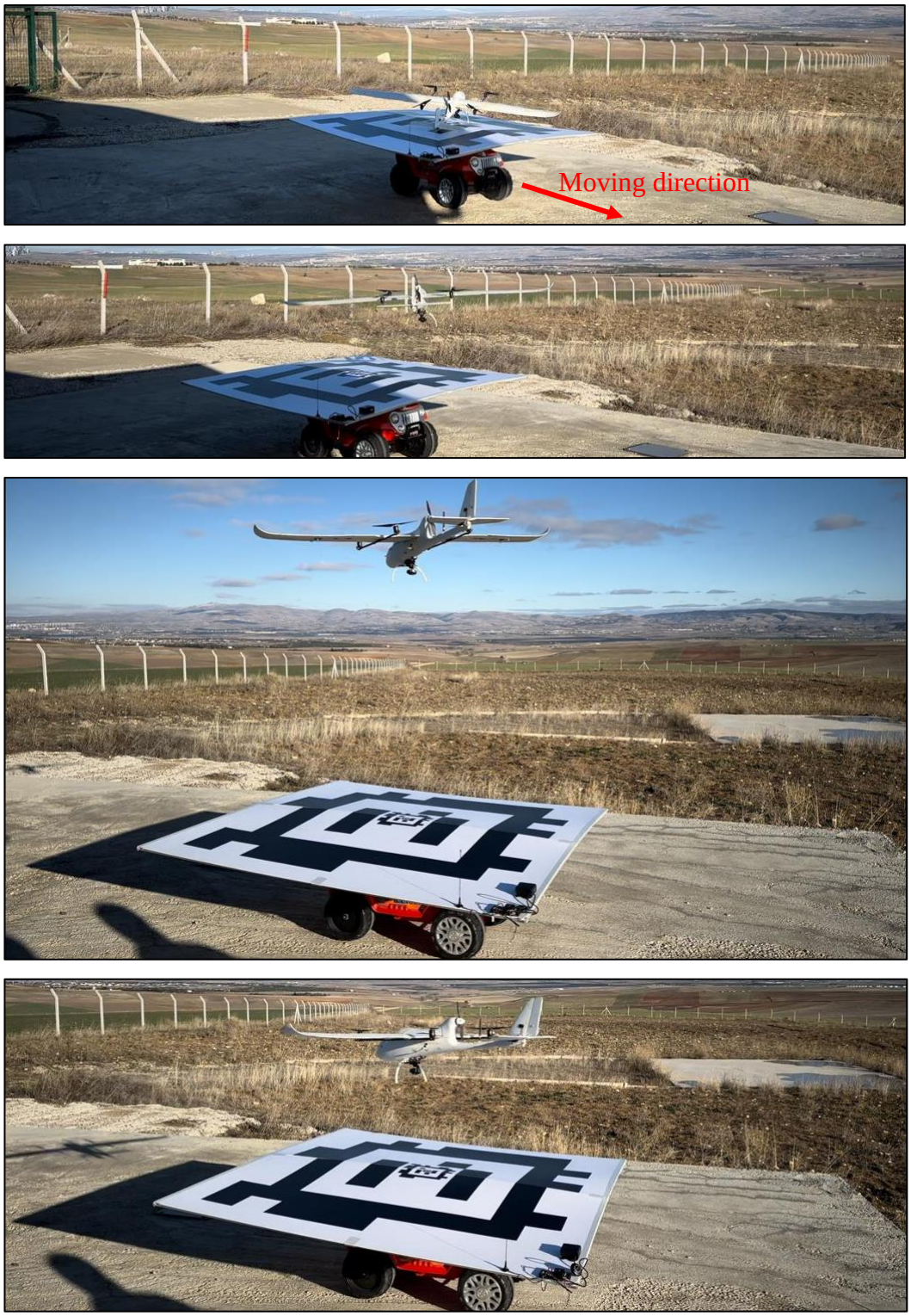

Figure 5.15. Quadplane autonomous takeoff and landing to a moving platform

CHAPTER 6

CONCLUSION AND FUTURE WORK

6.1 Conclusion

In conclusion, this thesis has addressed the intricate challenge of autonomous landing for VTOL UAVs on a ship deck amid the complexities introduced by the vessel's motion on standing wavy seas. The proposed system, incorporating a combination of camera and laser rangefinder sensors, was designed to achieve precision and kind landing.

The integration of computer vision techniques, the detection and tracking of the ship's landing platform through the onboard camera, emerged as an important element in the success of the autonomous landing system. Vision based position control algorithm ensured accurate alignment with the landing platform before and during the descent phase of the UAV. The low operational cost feature of the algorithm is vital for real-time applications.

The Descent rate control algorithm using the rangefinder data contributes for the most important problem of the study. The proposed algorithm can achieve good landing performance in the presence of pseudorandom ship deck heave motion.

Simulations and experiments validated the effectiveness of the proposed autonomous landing system. The system demonstrated its capability to autonomously land a VTOL UAV on a moving ship deck under varying sea conditions. The performance evaluations emphasized landing accuracy, robustness to disturbances, and overall system stability.

This research significantly contributes to the advancement of autonomous maritime UAV operations, addressing a critical aspect of ship-based UAV applications. By showcasing the potential for enhancing the safety and efficiency of maritime operations, including surveillance, reconnaissance, and logistics support, the developed system underscores the importance of prioritizing visual control in enabling UAVs to autonomously land on moving ship decks.

As the field of autonomous UAV operations continues to evolve, this research sets the stage for further refinement, optimization, and expansion of autonomous landing systems, fostering innovation and advancements in maritime robotics and unmanned systems. The lessons learned from this study provide valuable insights for future research endeavors seeking to enhance the capabilities and reliability of autonomous landing systems in dynamic and challenging environments.

6.2 Future Work

The most important feature of ArduPilot SITL is the ability to work as an autopilot firmware in many types of hardwares such as Pixhwak cube, Cuav PX4 etc. The simulation studies in the thesis, including the Lua scripts, can be directly in a hardware autopilot.

As a future work, the proposed ship landing algorithm will be used in the Tulpars [63] quadplane platform as shown in Figure 6.1.

Figure 6.1. Tulpars hybrid quadplane

REFERENCES

[1] Pope, M. T., Kimes, C. W., Jiang, H., Hawkes, E. W., Estrada, M. A., Kerst, C. F., Roderick, W. R. T., Han, A. K., Christensen D. L. and Cutkosky, M. R. 2016. A multimodal robot for perching and climbing on vertical outdoor surfaces. IEEE Transactions on Robotics, 33 (1), 38–48.

[2] Ahmed, F., Mohanta, J. C., Keshari, A. and Yadav, P. S. 2022. Recent Advances in Unmanned Aerial Vehicles: A Review. Arabian Journal for Science and Engineering, 47, 7963–7984.

[3] Ozdemir, U., Aktas, Y. O., Vuruskan, A., Dereli, Y., Tarhan, A.F., Demirbag, K., Erdem, A., Kalaycioglu, G. D., Özkol, I. and Inalhan, G. 2013. Design of a Commercial Hybrid VTOL UAV System. Journal of Intelligent and Robotic Systems, 74, 371-393.

[4] Gautam, A., Sujit, P. B. and Saripalli, S. 2014. A survey of autonomous landing techniques for UAVs. 2014 International Conference on Unmanned Aircraft Systems (ICUAS), Orlando, FL, USA, 1210-1218, doi: 10.1109/ICUAS.2014.6842377.

[5] Baca, T., Stepan, P., Spurny, V., Hert, D., Penicka, R., Saska, M., Thomas, J., Loianno, G. and Kumar, V. 2019. Autonomous Landing on a Moving Vehicle with an Unmanned Aerial Vehicle. Journal of Field Robotics, 36, 874-891.

[6] Ambroziak, L., Ciężkowski, M., Wolniakowski, A., Romaniuk, S., Bożko, A., Ołdziej, D. and Kownacki, C. 2022. Experimental tests of hybrid VTOL unmanned aerial vehicle designed for surveillance missions and operations in maritime conditions from ship-based helipads. Journal of Field Robotics, 39, 203-217.

[7] Hervas, J. R., Reyhanoglu M. and Tang, H. 2014. Automatic landing control of Unmanned Aerial Vehicles on moving platforms. 2014 IEEE 23rd International Symposium on Industrial Electronics (ISIE), Istanbul, Turkey, 69-74, doi: 10.1109/ISIE.2014.6864588.

[8] Çakıcı, F. and Leblebicioglu, M. K. 2016. Design and analysis of a mode-switching micro unmanned aerial vehicle. International Journal of Micro Air Vehicles, 8 (4), 221–229.

[9] Gu, H., Lyu, X., Li, Z., Shen, S. and Zhang, F. 2017. Development and experimental verification of a hybrid vertical take-off and landing (VTOL) unmanned aerial vehicle (UAV). 2017 International Conference on Unmanned Aircraft Systems (ICUAS), Miami, FL, USA, 160-169, doi: 10.1109/ICUAS.2017.7991420.

[10] Wang, S., Chen, H. J., Zhang, Z., Wang, G. and Du, N. 2018. A Deep-Learning-Based Sea Search and Rescue Algorithm by UAV Remote Sensing. 2018 IEEE CSAA Guidance, Navigation and Control Conference (CGNCC), Xiamen, China, 1-5, doi: 10.1109/GNCC42960.2018.9019134.

[11] Waharte, S. and Trigoni, N. 2010. Supporting search and rescue operations with UAVs. 2010 International Conference on Emerging Security Technologies, Canterbury, UK, 142-147, doi: 10.1109/EST.2010.31.

[12] Wang, A. R. 2017. Conceptual Design of a QuadPlane Hybrid Unmanned Aerial Vehicle. 2017 AIAA Student Conference Region VII-AUA, Melbourne, Australia, 6-11, doi: 10.13140/RG.2.2.23090.84163.

[13] Okulski, M. and Ławrynczuk, M. 2022. A Small UAV Optimized for Efficient Long-Range and VTOL Missions: An Experimental Tandem-Wing Quadplane Drone. Journal of Applied Sciences, 12 (14), 7059-7082.

[14] Anacleto, C., Agua, P. B. and Rafael, G. 2021. Machine Learning in coastal incident detection, identification and classification with UAVs. 2021 16th Iberian Conference on Information Systems and Technologies (CISTI), Chaves, Portugal, 1-6, doi: 10.23919/CISTI52073.2021.9476271.

[15] Nugroho, G., Zuliardiansyah, G. and Rasyiddin, A. 2022. Performance Analysis of Empennage Configurations on a Surveillance and Monitoring Mission of a VTOL-Plane UAV Using a Computational Fluid Dynamics Simulation. Aerospace, 9 (4), 208-239.

[16] Geng, L., Zhang, Y., Wang, J., Fuh, J. and Teo, S. 2013. Mission planning of autonomous uavs for urban surveillance with evolutionary algorithms. 2013 10th IEEE International Conference on Control and Automation (ICCA), Hangzhou, China, 828-833, doi: 10.1109/ICCA.2013.6564992.

[17] Dan, P. and Ichim, L. 2015. Image Recognition in UAV Application Based on Texture Analysis. 16th International Conference on Advanced Concepts for Intelligent Vision Systems (ACIVS), 9836, 693-704, doi: 10.1007/978-3-319-25903-1_60.

[18] Ramanathan, V., Ramana, M. V., Roberts, G., Kim, D., Corrigan, C., Chung, C. and Winker, D. 2007. Warming trends in Asia amplified by brown cloud solar absorption. Nature, 448, 575–578.

[19] Herwitz, S. R., Johnson, L. F., Dunagan, S. E., Higgins, R. G., Sullivan, D. V., Zheng, J., Lobitz, B. M., Leung, J. G., Gallmeyer, B., Aoyagi, M., Slye, R. E. and Brass, J. A. 2004. Imaging from an unmanned aerial vehicle: agricultural surveillance and decision support. Computers and Electronics in Agriculture, 44 (1), 49-61.

[20] Girard, A. R., Howell, A. S. and Hedrick, J. K. 2004. Border patrol and surveillance missions using multiple unmanned air vehicles. 2004 43rd IEEE Conference on Decision and Control (CDC), Nassau, Bahamas, 1, 620-625.

[21] Persson, L. and Wahlberg, B. 2019. Model predictive control for autonomous ship landing in a search and rescue scenario. AIAA Scitech 2019 Forum, San Diego, CA, UAS, 1169, doi: 10.2514/6.2019-1169.

[22] Trong, T. D., Hai, Q. T., Duc, N. T. and Thanh, H. T. 2021. A Novelty Approach to Emulate Field Data Captured by Unmanned Aerial Vehicles for Training Deep Learning Algorithms Used for Search-and-Rescue Activities at Sea. 2020 IEEE Eighth International Conference on Communications and Electronics (ICCE), Phu Quoc Island, Vietnam, 288-293, doi: 10.1109/ICCE48956.2021.9352109.

[23] Blachut, K., Szolc, H., Wasala, M., Kryjak, T. and Gorgon, M. 2020. A Vision Based Hardware-Software Real-Time Control System for the Autonomous Landing of an UAV. Computer Vision and Graphics, International Conference, ICCVG 2020, Warsaw, Poland, 13-24, doi: 10.1007/978-3-030-59006-2_2.

[24] Huang, Y., Zhong, Y., Cheng, S. and Ba, M. 2019. Research on UAV's autonomous target landing with image and GPS under complex environment. 2019 International Conference on Communications, Information System and Computer Engineering (CISCE), Haikou, China, 97-102, doi: 10.1109/CISCE.2019.00029.

[25] Lee, S., Shim, T., Kim, S., Park, J., Hong, K. and Bang, H. 2018. Vision-based autonomous landing of a multi-copter unmanned aerial vehicle using reinforcement learning. 2018 International Conference on Unmanned Aircraft Systems (ICUAS), Dallas, TX, USA, 108-114, doi: 10.1109/ICUAS.2018.8453315.

[26] Xu, C., Tang, Y., Liang, Z. and Yin, H. 2018. UAV autonomous landing algorithm based on machine vision. 2018 IEEE 4th Information Technology

and Mechatronics Engineering Conference (ITOEC), Chongqing, China, 824-829, doi: 10.1109/ITOEC.2018.8740483.

[27] Patruno, C., Nitti, M., Petitti, A., Stella, E. and D'Orazio, T. 2019. A vision-based approach for unmanned aerial vehicle landing. Journal of Intelligent & Robotic Systems, 95, 645–664.

[28] Yu, L., Li, M. and Pan, G. 2021. Indoor Localization Based on Fusion of AprilTag and Adaptive Monte Carlo. 2021 IEEE 5th Information Technology, Networking, Electronic and Automation Control Conference (ITNEC), Xi'an, China, 464-468, doi: 10.1109/ITNEC52019.2021.9587205.

[29] Olson, E. 2011. AprilTag: A robust and flexible visual fiducial system. 2011 IEEE International Conference on Robotics and Automation, Shanghai, China, 3400-3407, doi: 10.1109/ICRA.2011.5979561.

[30] Lee, B., Saj, V., Benedict, M. and Kalathil, D. M. 2020. A Vision-Based Control Method for Autonomous Landing of Vertical Flight Aircraft on a Moving Platform Without Using GPS. ArXiv, abs/2008.05699.

[31] Paris, A., Lopez, B. T. and How, J. P. 2020. Dynamic Landing of an Autonomous Quadrotor on a Moving Platform in Turbulent Wind Conditions. 2020 IEEE International Conference on Robotics and Automation (ICRA), 9577-9583, doi: 10.1109/ICRA40945.2020.9197081.

[32] Araar, O., Aouf, N. and Vitanov, I. 2017. Vision based autonomous landing of multirotor UAV on moving platform. Journal of Intelligent and Robotic Systems, 85, 369–384.

[33] Falanga, D., Zanchettin, A., Simovic, A., Delmerico, J. and Scaramuzza, D. 2017. Vision-based autonomous quadrotor landing on a moving platform. 2017 IEEE International Symposium on Safety, Security and Rescue

Robotics (SSRR), Shanghai, China, 200-207, doi: 10.1109/SSRR.2017.8088164.

[34] Acuna, R. Zhang, D. and Willert, V. 2018. Vision-Based UAV Landing on a Moving Platform in GPS Denied Environments Using Motion Prediction. 2018 Latin American Robotic Symposium, 2018 Brazilian Symposium on Robotics (SBR) and 2018 Workshop on Robotics in Education (WRE), João Pessoa, Brazil, 515-521, doi: 10.1109/LARS/SBR/WRE.2018.00096.

[35] Wynn, J. S. and McLain, T. W. 2019. Visual Servoing with Feed-Forward for Precision Shipboard Landing of an Autonomous Multirotor. 2019 American Control Conference (ACC), Philadelphia, PA, USA, 3928-3935, doi: 10.23919/ACC.2019.8814694.

[36] Meng, Y., Wang, W., Han, H. and Ban, J. 2019. A visual/inertial integrated landing guidance method for UAV landing on the ship. Aerospace Science and Technology, 85, 474-480.

[37] Lee, B., Saj, V., Benedict, M. and Kalathil, D. M. 2023. Intelligent Vision-based Autonomous Ship Landing of VTOL UAVs. Journal of the American Helicopter Society, 68 (2), 113-126.

[38] Polvara, R., Sharma, S. and Wan, J. 2018. Vision-Based Autonomous Landing of a Quadrotor on the Perturbed Deck of an Unmanned Surface Vehicle. Drones, 2 (2), 15-33.

[39] Trong, T. D., Hai, Q., Van, M. V., Thai, B. N., Chi, T. N. and Quang, T. N. 2021. Autonomous Detection and Approach Tracking of Moving Ship on the Sea by VTOL UAV based on Deep Learning Technique through Simulated Real-time On-Air Image Acquisitions. 2021 8th NAFOSTED Conference on Information and Computer Science (NICS), Hanoi, Vietnam, 374-380, doi: 10.1109/NICS54270.2021.9701509.

[40] Xia, K., Lee, S. and Son, H. 2020. Adaptive control for multi-rotor UAVs autonomous ship landing with mission planning. Aerospace Science and Technology, 96, 105549, doi: 10.1016/j.ast.2019.105549

[41] Tan, C. K. and Wang, J. 2014. Ship deck landing of a quadrotor using the invariant ellipsoid technique. 2014 Proceedings of the SICE Annual Conference (SICE), Sapporo, Japan, 1443-1448, doi: 10.1109/SICE.2014.6935276.

[42] Lin, H., Wang, B., Tang, C. and Xu, G. 2022. Autonomous Landing of a VTOL UAV on a Ship based on Tau Theory. 2022 34th Chinese Control and Decision Conference (CCDC), Hefei, China, 2785-2790, doi: 10.1109/CCDC55256.2022.10034344.

[43] Trong, T. D., Van, M. V., Hai, Q. T. and Thai, B. N. 2022. Autonomous Landing Scheme of VTOL UAV on Moving Ship Using Deep Learning Technique Embedded in Companion Computer. 2022 11th International Conference on Control, Automation and Information Sciences (ICCAIS), Hanoi, Vietnam, 493-498, doi: 10.1109/ICCAIS56082.2022.9990036.

[44] Ziegler, J. G. and Nichols, N. B. 1942. Optimum settings for automatic controllers. Transactions of the ASME, 64, 759-768.

[45] Cohen, G. and Coon, G. A. 1953. Theoretical Consideration of Retarded Control. Transactions of the ASME, 75 (1), 827-834.

[46] Tan, C. K. and Wang, J. 2014. A novel PID controller gain tuning method for a quadrotor landing on a ship deck using the invariant ellipsoid technique. 2014 14th International Conference on Control, Automation and Systems (ICCAS), Gyeonggi-do, Korea (South), 1339-1344, doi: 10.1109/ICCAS.2014.6987764.

[47] ArduPilot, open-source autopilot system, https://ardupilot.org/

[48] ArduPilot repository, https://github.com/ArduPilot/ardupilot

[49] AprilTag, a visual fiducial system, https://github.com/AprilRobotics/apriltag

[50] OpenCV library for Python, https://github.com/opencv/opencv-python

[51] Gazebo Garden, open-source 3D simulator, https://gazebosim.org/docs/garden

[52] ArduPilot Software in the Loop, https://ardupilot.org/dev/docs/sitl-simulator-software-in-the-loop.html

[53] MavProxy, GCS software, https://github.com/ArduPilot/MAVProxy

[54] Wave Sim library, https://github.com/srmainwaring/asv_wave_sim

[55] Tessendorf, J. 2004. Simulating Ocean Water. Lecture notes, 3, 1-26

[56] ArduPilot plugin for Gazebo, https://github.com/ArduPilot/ardupilot_gazebo

[57] ArduPilot SITL models, https://github.com/ArduPilot/SITL_Models

[58] Dronekit library for Python, https://github.com/dronekit/dronekit-python

[59] Autonomous landing of a VTOL UAV on a ship deck moving on standing wavy sea using camera and laser rangefinder simulation. https://youtu.be/vZch4I5O11o

[60] Çakıcı, F. 2016. Control and guidance of a multi-mode unmanned aerial vehicle for increased versatility [Ph.D. - Doctoral Program]. Middle East Technical University.

[61] Vision based position control algorithm for quadplane landing flight test. https://youtu.be/jSFwajqya0E

[62] Quadplane autonomous landing on a moving platform with GPS tracking and image processing. https://youtu.be/m43z_f9vsSU

[63] Tulpars hybrid quadplane, https://www.tulpars.com