# ON THE COMPLETE AUTOMATION OF VERTICAL FLIGHT AIRCRAFT SHIP LANDING

A Dissertation

by

BOCHAN LEE

Submitted to the Graduate and Professional School of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

| | |
|---|---|
| Chair of Committee, | Benedict Moble |
| Committee Members, | John Valasek |
| | Sivakumar Rathinam |
| | Dileep Kalathil |
| Head of Department, | Srinivas Rao Vadali |

August 2021

Major Subject: Aerospace Engineering

# ABSTRACT

The current study focuses on developing an autonomous vertical flight aircraft ship landing system by directly automating the established Navy helicopter ship landing procedure. The central idea involves visually tracking a gyro-stabilized horizon bar installed on most Navy ships to approach and land vertically independent of deck motions. This was accomplished through the development of a rotorcraft flight dynamics modeling framework and vision-based control systems as well as conducting simulations and flight tests.

The framework, named Texas A&M Rotorcraft Analysis Code (TRAC), was developed as a modular tool that could model any rotorcraft configuration at a low computational cost. A UH-60 helicopter was modeled as a baseline aircraft and validated using the US Army flight test data. A linear quadratic regulator (LQR) controller was utilized to stabilize and control the helicopter during autonomous ship landing simulations.

The vision system was developed to obtain the visual information that a pilot perceives during ship approach and landing. It detects the ship at long-distance by utilizing machine/deep learning-based detection and at close range, it utilizes uniquely developed vision algorithms to detect the horizon bar to precisely estimate the aircraft position and orientation relative to the bar. It demonstrated 250 meters of detection range for a 6 x 6 ft sub-scale ship platform, which translates to a range of 17.3 kilometers for a full-scale 50 x 50 ft typical small ship. The distance and attitude estimations were validated using the measurements from an accurate 3D motion capture system (VICON), which demonstrated sub-centimeter and sub-degree accuracy.

To control the aircraft based on the perceived visual information, both nonlinear control and deep reinforcement learning control strategies were developed. The nonlinear controller demonstrated robust tracking capability even with 0.5 seconds of time delay and estimation noise. When flight-tested in 5 m/s wind gust, the deep reinforcement learning control demonstrated superior disturbance rejection capability, with 50% reduced drift at a 3 times faster rate compared to conventional control systems. Both vision and control systems were implemented on a quadrotor

unmanned aircraft and extensive flight tests were conducted to demonstrate accurate tracking in challenging conditions and safe vertical landing on a translating ship platform with 6 degrees of freedom motions.

# DEDICATION

To my parents Byungkyun Lee and Okhwa Lee, for giving me wings,

and to my wife Sunhye Lee and my sons Jaeho Lee and Thomas Jaehee Lee, for flying with me.

# ACKNOWLEDGMENTS

First and foremost, I owe a large debt of gratitude to Dr. Moble Benedict. Without his careful guidance and supervision, development of the system presented in this dissertation would have been nearly impossible. Every time I lost the way to go, he always shed light on the path to follow. He always listened to my opinions with open mind and shared great insights with me. Every moment I spent with him was a great pleasure and made me open eyes to the new world.

I also would like to express my sincere gratitude to Dr. Ananth Sridharan at the University of Maryland for the guidance and support in modeling and control system design. He always answered my questions and his feedback played an important role in my work. I am also thankful to my committee members Dr. John Valasek, Dr. Sivakumar Rathinam, and Dr. Dileep Kalathil for giving me precious feedback and showing me visions.

I am also grateful to all the friends that I met here at Texas A&M University, Vinicius Goecks, Niladri Das, Kookjin Sung, Woosang Park, Sunsoo Kim, David Coleman, Carl Runco, Hunter Denton, Atanu Halder, Farid Saemi, Roshan Suresh, Roshan Thomas, and Utkarsh Mishra for their help and sharing ideas. It is impossible to remember all, and I apologize to those I have inadvertently left out.

I owe my deepest thanks to my family - my father, Byungkyun Lee, my mother, Okhwa Lee, and my sister Younhye Lee, who have always stood by me and supported me through my career. My wife, Sunhye Lee has sacrificed so many things in her life and beared with my busy life. I promise that I will pay back my debts in the rest of life. My lovely two sons, Jaeho Lee and Thomas Jaehee Lee are my reason for life. They motivated me to become a father they can be proud of. I always love you as you are.

CONTRIBUTORS AND FUNDING SOURCES

## Contributors

This work was supported by a dissertation committee consisting of professors Moble Benedict and John Valasek in the department of aerospace engineering, professor Sivakumar Rathinam in the department of mechanical engineering, and professor Dileep Kalathil in the department of electrical and computer engineering. Also, Dr. Ananth Sridharan at the University of Maryland gave guidance for mathematical rotorcraft modeling.

Graduate student Vishnu Saj in the department of electrical and computer engineering contributed to the development of the classical computer vision system and all the other work for the dissertation was completed by the author.

## Funding Sources

This study has been funded by the Center for Unmanned Aircraft Systems (C-UAS), a National Science Foundation Industry/University Cooperative Research Center (I/UCRC) under NSF award Numbers IIP-1161036 and CNS-1946890, along with significant contributions from C-UAS industry members. Also, the graduate study was supported by a fellowship from the Republic of Korea Navy and a research scholarship from the Vertical Flight Foundation.

NOMENCLATURE

| | |
|---|---|
| $A$ | Acceleration vector |
| | State matrix |
| | Camera intrinsic matrix |
| | Action space |
| $a$ | Lift-curve slope |
| | Long-range controller constant |
| $a_t$ | Action at time t |
| $B$ | Control matrix |
| $b$ | Close-range controller constant |
| $CE$ | Current yaw estimate |
| $CM$ | Current yaw measurement |
| $c$ | Blade section chord length |
| | Setpoint |
| $c_d$ | Blade section drag coefficient |
| $c_l$ | Blade section lift coefficient |
| $c_m$ | Blade section pitching moment coefficient |
| $c_u$ | Corner position-column |
| $c_v$ | Corner position-row |
| $D$ | Replay Buffer |
| $d$ | Long-range controller constant |
| | Deviation from target, meter |
| $de(t_k)$ | Error difference between time $t_k$ and $t_{k-1}$ |

| | |
|---|---|
| $dt_k$ | Time difference between time $t_k$ and $t_{k-1}$ |
| $E$ | Young's Modulus |
| | Acceleration coupling matrix |
| $e$ | Hinge offset of main rotor blade |
| $e(t_k)$ | Error at time $t_k$ |
| $f(x)$ | Probability density function |
| $f_x$ | Focal length of camera in horizontal direction |
| $f_y$ | Focal length of camera in vertical direction |
| $G$ | Shear modulus |
| $g$ | Acceleration due to gravity |
| $H$ | Hermite interpolation polynomials |
| $h(t_k)$ | Height of bounding rectangle at time $t_k$ |
| $I_b$ | Mass flapping moment of inertia |
| $I_{xx}$, $I_{yy}$, $I_{zz}$ | Aircraft mass moments of inertia about body axes |
| $I_{xy}$, $I_{xz}$, $I_{yz}$ | Aircraft products of inertia |
| $i_B$, $j_B$, $k_B$ | Unit vectors of body-fixed frame |
| $i_F$, $j_F$, $k_F$ | Unit vectors of blade flapped frame |
| $i_G$, $j_G$, $k_G$ | Unit vectors of earth-fixed frame |
| $i_{NR}$, $j_{NR}$, $k_{NR}$ | Unit vectors of blade non-rotating frame |
| $i_R$, $j_R$, $k_R$ | Unit vectors of blade rotating frame |
| $i_L$, $j_L$, $k_L$ | Unit vectors of blade lagged frame |
| $i_{TPP}$, $j_{TPP}$, $k_{TPP}$ | Unit vectors of tip path plane frame |
| $K$ | Linear portion of stiffness matrix |
| $KG$ | Kalman gain |
| $K_D$ | Derivative gain |

| | |
|---|---|
| $K_I$ | Integral gain |
| $K_P$ | Proportional gain |
| $L$ | Total applied moment about body x-axis |
| $L(\phi, D)$ | Mean squared Bellman error function |
| $M$ | Total applied moment about body y-axis |
| | Linear portion of mass matrix |
| $m$ | Long-range controller constant |
| $m_b$ | Blade mass per unit length |
| $N$ | Total applied moment about body z-axis |
| $N_b$ | Number of blades |
| $O_{1x3}$ | 1x3 zero matrix |
| $PE$ | Previous yaw estimate |
| $p$ | Roll rate of aircraft |
| | Indicator to check whether a terminal state or not |
| $QN$ | Measurement noise covariance |
| $Q(s, a)$ | Q-value function |
| $q$ | Pitch rate of aircraft |
| | Dynamic pressure |
| $R$ | Main rotor radius |
| | Position vector |
| | Rotation matrix |
| $R_{basic}$ | Basic rotation matrix |
| $R_{modified}$ | Modified rotation matrix |
| $R_t$ | Cumulative Rewards |
| $RN$ | Process noise covariance |

| | |
|---|---|
| $r$ | Yaw rate of aircraft |
| $r(t_k)$ | Relative position at time $t_k$ |
| $S$ | State Space |
| $s$ | Scaling factor |
| $s_t$ | State at time t |
| $sw$ | Window size |
| $T_{BG}$ | Transformation matrix from earth-fixed frame to body-fixed frame |
| $T_{GB}$ | Transformation matrix from body-fixed frame to earth-fixed frame |
| $T_{NB}$ | Transformation matrix from body-fixed frame to blade non-rotating frame |
| $T_{RN}$ | Transformation matrix from blade non-rotating frame to blade rotating frame |
| $T_{FR}$ | Transformation matrix from blade rotating frame to blade flapped frame |
| $T_{LF}$ | Transformation matrix from blade flapped frame to blade lagged frame |
| $T_{TN}$ | Transformation matrix from blade non-rotating frame to tip path plane frame |
| $t$ | Time (sec) |
| | Translation vector |
| $u$ | Forward speed of the aircraft |
| | Control vector |
| | Image pixel position-column |
| $u(t_k)$ | Control law |
| $u_0$ | Center of image-column |
| $V$ | Velocity vector |
| $V(s)$ | Value function |
| $v$ | Lateral speed of the aircraft |
| | Image pixel position-row |

| | |
|---|---|
| $v_x$ | Forward relative speed, m/s |
| $v_y$ | Sideward relative speed, m/s |
| $v_0$ | Center of image-row |
| $w$ | Vertical speed of the aircraft, angular rates vector |
| $w(t_k)$ | Width of bounding rectangle at time $t_k$ |
| $X$ | Total applied force along body x-axis |
| | Sideward relative distance, meter |
| $x$ | Trim vector |
| | Forward relative distance, meter |
| $Y$ | Total applied force along body y-axis |
| | Vertical relative distance, meter |
| $y$ | State vector |
| | Sideward relative distance, meter |
| $Z$ | Total applied force along body z-axis |
| | Forward relative distance, meter |
| $\alpha$ | Relative yaw angle, rad |
| $\alpha_F$ | Aircraft angle of attack |
| $\alpha_s$ | Longitudinal shaft tilt angle |
| $\beta$ | Relative pitch angle, rad |
| $\beta_F$ | Fuselage sideslip angle |
| $\beta_S$ | Lateral shaft tilt angle |
| $\beta_o$ | Blade coning angle |
| $\beta_{1c}$ | Longitudinal blade flapping coefficient |
| $\beta_{1s}$ | Lateral blade flapping coefficient |
| $\Lambda$ | Tail rotor cant angle |

| | |
|---|---|
| $\gamma$ | Flight path angle |
| | Relative roll angle, rad |
| | Discount factor |
| $\zeta_o$ | Blade mean lagging angle |
| $\zeta_{1c}$ | Longitudinal blade lagging coefficient |
| $\zeta_{1s}$ | Lateral blade lagging coefficient |
| $\theta$ | Aircraft pitch attitude Euler angle |
| | Blade pitch angles associated with geometric rotation |
| $\theta_o$ | Collective control stick angle |
| $\theta_{1c}$ | Lateral cyclic control stick angle |
| $\theta_{1s}$ | Longitudinal cyclic control stick angle |
| $\theta_{TR}$ | Tail rotor control stick (pedal) angle |
| $\theta_{tw}$ | Built-in blade twist |
| $\lambda_o$ | Steady portions of main rotor inflow |
| $\lambda_{1c}$ | Cosine portions of main rotor inflow |
| $\lambda_{1s}$ | Sine portions of main rotor inflow |
| $\lambda_{tr}$ | Tail rotor inflow |
| $\mu$ | Main rotor advance ratio |
| | Mean value |
| $\Pi$ | Policy space |
| $\pi$ | Policy |
| $\pi^*$ | Optimal policy |
| $\phi$ | Q-value network parameter |
| $\rho$ | Air density |
| $\rho_u$ | Parameter to convert pixels to SI units-column |

| | |
|---|---|
| $\rho_v$ | Parameter to convert pixels to SI units-row |
| $\sigma$ | Standard deviation |
| $\phi$ | Aircraft roll attitude Euler angle |
| $\phi_F$ | Fuselage roll angle |
| $\chi$ | Wake skew angle |
| $\psi$ | Aircraft yaw attitude Euler angle |
| | Blade azimuth angle |
| $\Omega$ | Main rotor speed |
| $\nabla f(u, v)$ | Image gradient |
| BEMT | Blade Element Momentum Theory |
| DDPG | Deep Deterministic Policy Gradient |
| DGPS | Differential Global Positioning System |
| DOF | Degrees Of Freedom |
| DQN | Deep Q-Networks |
| FAA | Federal Aviation Administration |
| FDR | Flight Data Recorder |
| FHD | Full High Definition |
| GPS | Global Positioning System |
| HAFOV | Horizontal Angular Field of View |
| HOSTAC | Helicopter Operations from Ships other Than Aircraft Carriers |
| HSV | Hue-Saturation-Value |
| IR | Infrared Radiation |
| JPALS | Joint Precision Approach and Landing System |
| LQR | Linear Quadratic Regulator |
| LTI | Linear Time Invariant |

| | |
|---|---|
| MAP | Missed Approach Point |
| MPC | Model Predictive Control |
| NATO | North Atlantic Treaty Organization |
| NATOPS | Naval Air Training and Operating Procedures Standardization |
| NSF | National Science Foundation |
| ODE | Ordinary Differential Equation |
| PD | Proportional-Derivative |
| PDE | Partial Differential Equation |
| PID | Proportional-Integral-Derivative |
| PnP | Perspective-n-Point |
| PPO | Proximal Policy Optimization |
| RGB | Red, Green, Blue |
| RL | Reinforcement Learning |
| TD3 | Twin Delayed DDPG |
| TPP | Tip Path Plane |
| TRPO | Trust Region Policy Optimization |
| UAV | Unmanned Aerial Vehicle |
| VTOL | Vertical Takeoff and Landing |

TABLE OF CONTENTS

Page

LIST OF FIGURES

FIGURE Page

LIST OF TABLES

TABLE Page

# 1. INTRODUCTION

Landing a helicopter on a moving ship deck has always been a challenging task. However, this is unavoidable and forms the basis of all maritime air operations. Looking at the naval history of ship landing, it was recorded that XOP-1, an autogyro shown in Fig. 1.1, was the first rotary-wing aircraft landed and took off from a ship at sea on September 23, 1931. The ship was the U.S. Navy's first aircraft carrier United States Ship (USS) Langley (CV-1), which had a large flight deck that was 165.2 meters long and 19.9 meters wide providing sufficient space for the operation of vertical take-off and landing (VTOL) capable aircraft.

(a) Pitcairn autogyro XOP-1 [10]

(b) USS Langley (CV-1) [11]

Figure 1.1: First rotary-wing aircraft landing on a ship

Since this historical autogyro landing on the aircraft carrier, it took another 13 years for a helicopter to land on a ship with a single flight deck. In early January 1944, the Sikorsky R-4 helicopter also known as Hoverfly/HNS-1 landed on the British steam tanker Daghestan as shown in Fig. 1.2. This is the first recorded helicopter landing on a single deck ship.

(a) Sikorsky R-4 helicopter

(b) British steam tanker Daghestan

Figure 1.2: First helicopter landing on a single deck ship [1]

On June 15th, 1944 the Sikorsky R-4 helicopter also landed on the United States Coast Guard Cutter (USCGC) Cobb (WPG-181). It was the first U.S. helicopter carrier that was converted from the turbine-powered steamship and had a 38 x 63 ft helicopter flight deck. Since then, helicopters have been widely utilized in Navy operations such as air transportation, search and rescue, sea patrol, anti-submarine warfare, replenishment, and amphibious warfare. Later, unmanned helicopters also appeared in Navy operations. On December 7th, 1960 the unmanned helicopter QH-50 shown in Fig. 1.3 landed on USS Hazelwood (DD-531) without a safety pilot on board. It was the world's first unmanned, remotely piloted helicopter ship landing.

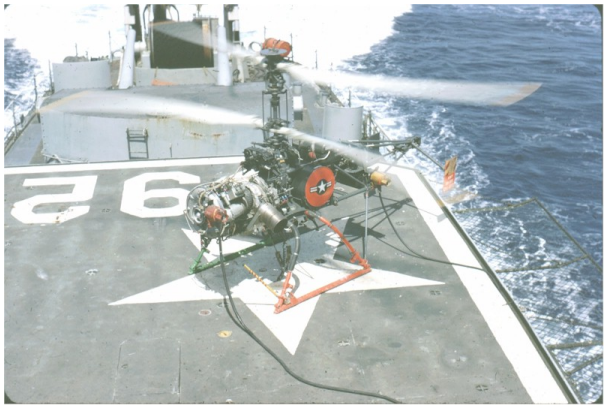


Figure 1.3: First unmanned helicopter (QH-50) that landed on a ship [2]

As aircraft technology advanced, multiple vertical take-off and landing (VTOL) capable aircraft were introduced and used more actively at sea for various purposes, which led to growing safety concerns. To enhance the safety related to landing on small ships, safety countermeasures were implemented such as the establishment of ship landing procedures and pilot aid equipment. One of the efforts towards safe helicopter ship landing is the Helicopter Operations from Ships other Than Aircraft Carriers (HOSTAC) program initiated by the North Atlantic Treaty Organization (NATO). It aims to establish the appropriate standards and also exchange national information for successful and safe helicopter operations. It aims to establish the appropriate standards and also exchange national information for successful and safe helicopter operations. It became the global standard as more countries participated. Although significant efforts over the past few decades contributed to enhancing safety, helicopter ship landing accidents continue to occur even today. Therefore, automating such a demanding job is crucial to aircraft safety and survivability; however, autonomous landing is still not common in VTOL aircraft shipboard operations even 80 years after the first historical helicopter ship landing. In recent years, with the rapid rise of Unmanned Aerial Vehicles (UAVs) for maritime applications, many efforts are targeted at developing autonomous ship landing systems for VTOL UAVs. Since the first autonomous ship landing of an unmanned VTOL UAV RQ-8A Fire Scout in January 2006, the autonomous ship landing capable S-100 CAMCOPTER UAV was introduced in 2009 and the VSR700 optionally piloted vehicle (OPV) demonstrated the autonomous deck landing in 2020, which are shown in Fig. 1.4.

(a) Northrop Grumman RQ-8A Fire Scout [12]

(b) Boeing S-100 CAMCOPTER [13]

(c) Airbus VSR700 OPV [14]

Figure 1.4: Autonomous ship/deck landing capable VTOL unmanned aircraft

All these aircraft executed autonomous ship/deck landing by tracking the landing deck using multiple sensors. Although it might appear natural to track the landing location, this landing mechanism is different from the helicopter ship landing procedure that has been practiced for several decades. Therefore, this study begins by deeply understanding the helicopter ship landing procedure in order to seek a unified and practical autonomous ship landing solution that can be applied to any manned or unmanned VTOL aircraft.

### 1.1 Helicopter Ship Landing Procedure

There are many factors that make helicopter ship landing extremely difficult such as small landing space, six degrees of freedom (DOF) deck motions, limited visual references for pilots, and lack of alternative landing places as discussed in Refs. [15, 16]. As the environment is completely different from landing on the ground, the ship landing carried out in a unique fashion while adapting to the given situations. Each ship has its own approach chart and Fig. 1.5 shows an example tactical air navigation system (TACAN) approach chart presented in the HOSTAC.

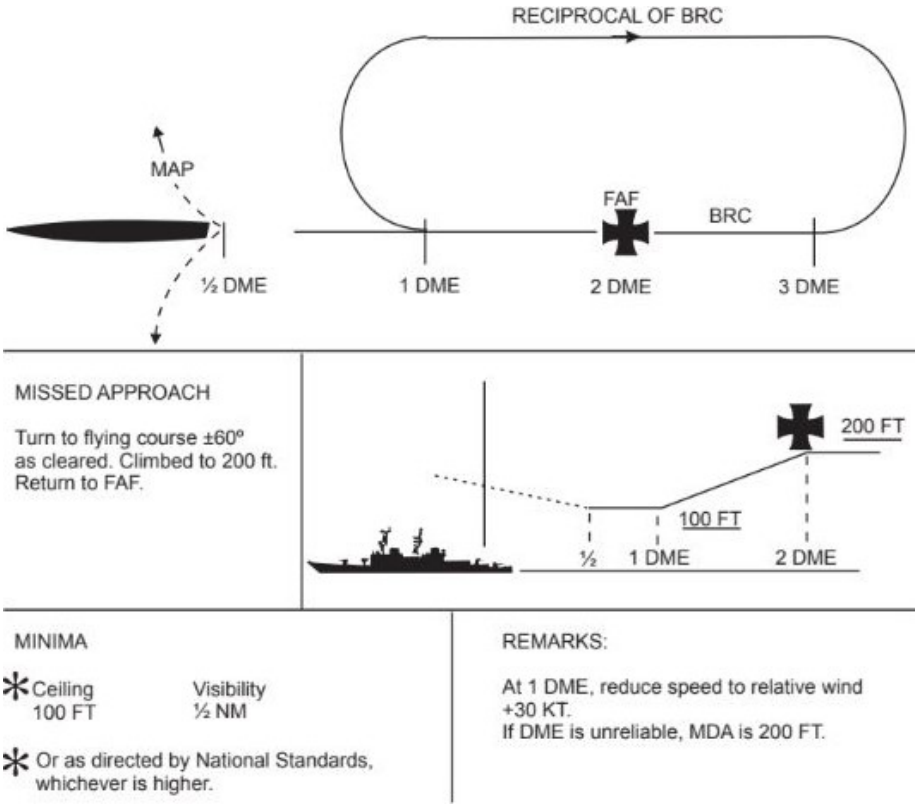



Figure 1.5: TACAN approach chart [3]

TACAN is radio-based navigational aid widely used in the military since 1950 to provide the relative bearing and distance from a ship, which is sufficient information for aircraft to approach the ship from a distance. The missed approach point (MAP) and approach angle are two important things to note in the chart. First, the MAP is designated at 0.5 nautical miles away from the ship. If the pilot fails to identify the ship visually at the MAP, he/she should abort ship landing as instructed in the chart. The approach angle can be calculated using distance and altitude information between the points. Aircraft descends 100 ft during 1 nautical mile thus the approach angle is approximately 1 degree, which characterizes the ship approach as a horizontal approach. During the last phase of approach and landing, pilots are assisted by visual aids as shown in Fig. 1.6.

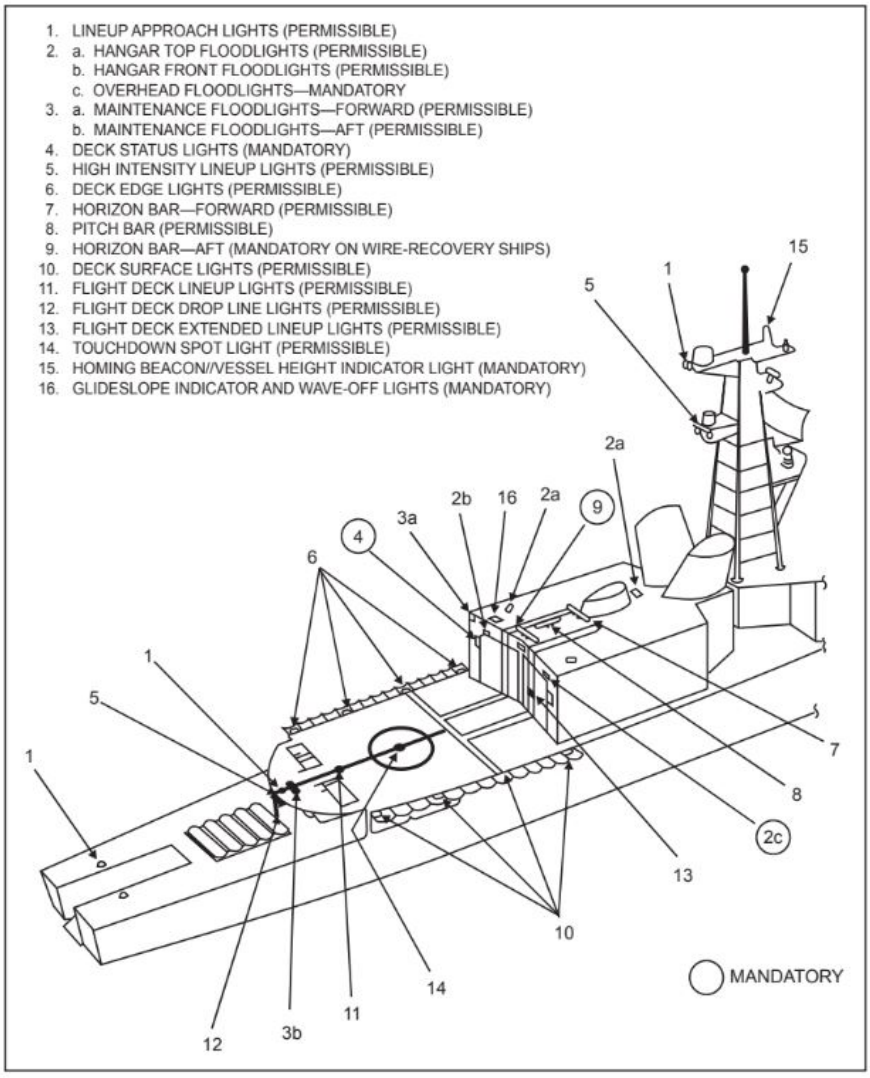
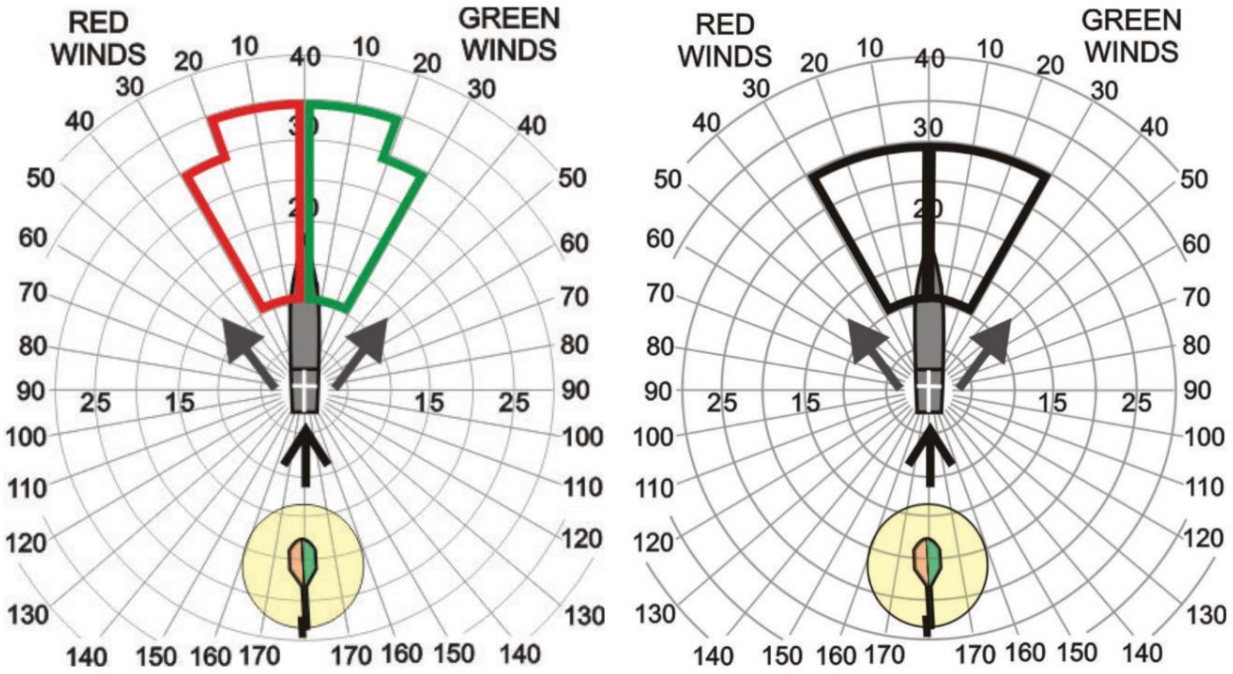



Figure 1.6: Standard flight deck references [3]

The standard flight deck references consist of lights and pose indicating bars. Above all, the horizon reference bar plays a key role for safe landing and an example horizon bar is shown in Fig. 1.7.

Figure 1.7: Horizon reference bar [4]

Horizon bar is gyro-stabilized and therefore, it indicates the true horizon independent of the ship motions. Navy helicopter pilots are trained not to follow the ship deck motions but to refer to the horizon bar. In this way, pilots can avoid spatial disorientation caused by the ship motions and the horizon bar acts as an excellent solution to the lack of availability of a fixed horizon to refer to. This is a critical element to maintain the right sense of helicopter attitude. Once the helicopter hovers in a stable fashion over the deck, the pilot waits for a quiescent moment in the ship motion, then conducts vertical landing quickly to prevent the ship deck from impacting one of the helicopter landing gear skids/wheels causing a rollover. This vertical landing manner is proven safe within the operating limits of the helicopter and currently being used in practice. Ship landing limits vary depending on the type of helicopter and ship, approach direction, day or night, and assist equipment. The HOSTAC ship helicopter operating limit (SHOL) in the case of the straight-in approach is defined as shown in Fig. 1.8.




(a) Day operation

(b) Night operation

Figure 1.8: HOSTAC SHOL envelope [3]

The envelopes show wind direction and magnitude boundaries for safe take-off and landing, which are differently imposed for daytime and night-time operations. Along with the wind conditions, HOSTAC SHOL also includes deck motion limits depending on helicopter configurations, day operation, night operation, and night operation with a night vision device (NVD) as described in Table 1.1.

| Configuration | Day | | Night(Conventional) | | Night(NVD) | |
|---|---|---|---|---|---|---|
| | Pitch | Roll | Pitch | Roll | Pitch | Roll |
| Skid/Narrow Wheel Base/ High C.G./Tail Wheel | $\pm 1°$ | $\pm 3°$ | $\pm 1°$ | $\pm 2°$ | $\pm 1°$ | $\pm 2°$ |
| Wheel without deck securing system | $\pm 2°$ | $\pm 4°$ | $\pm 1°$ | $\pm 3°$ | $\pm 1°$ | $\pm 3°$ |
| Wheel with deck securing system | $\pm 2°$ | $\pm 6°$ | $\pm 1°$ | $\pm 4°$ | $\pm 1°$ | $\pm 4°$ |

Table 1.1: HOSTAC SHOL deck motion limits [3]

In the Naval Air Training and Operating Procedures Standardization (NATOPS) program, these limits are defined for a specific type of helicopter and ship [17]. SHOLs are derived by test pilots during the first of class flying trial (FOCFT) for each helicopter and ship. The boundaries are set when either the pilot's workload becomes too much for safe, accurate, and consistent landings or aircraft reaches its operating limits.

As shown in Table 1.2, The key characteristics of helicopter ship landing can be summarized as horizontal approach assisted by visual aids, vertical landing without following the deck motions, and varying operational limits.

| | |
|---|---|
| Approaching manner | Horizontal approach (descent angle: $1^\circ$) |
| Approaching course | Port(left side of ship), starboard(right side of ship), stern(straight-in) |
| Landing manner | Vertical landing without following deck motions |
| Visual aids | Lights, reference bars |
| Wind condition* | Day: max. 35 knots within $20^\circ$ from ship heading,<br>max. 30 knots between $20^\circ$ and $30^\circ$ from ship heading<br>Night: max. 30 knots within $30^\circ$ from ship heading |
| Deck motion limits* | Day: $\pm 1^\circ \sim \pm 2^\circ$ pitch, $\pm 3^\circ \sim \pm 6^\circ$ roll<br>Night: $\pm 1^\circ$ pitch, $\pm 2^\circ \sim \pm 4^\circ$ roll |

* can vary subject to a specific helicopter and ship.

Table 1.2: Characteristics of helicopter ship landing

This study intends to develop a practical autonomous landing solution based on an in-depth understanding of the helicopter ship landing procedure and demonstrate it through extensive simulations and flight tests. The first step towards this objective is to conduct helicopter ship landing simulations by developing a helicopter flight dynamics model coupled with autonomous flight control systems.

## 1.2 Helicopter Flight Dynamics Modeling

In order to predict helicopter responses in various flight conditions, there have been multiple comprehensive helicopter flight dynamics modeling tools from government research institutes, industry, and academia. Some of the major helicopter dynamic modeling frameworks since 1960 are shown in Fig. 1.9 by their organizations and the approximate time frame at which they were introduced.

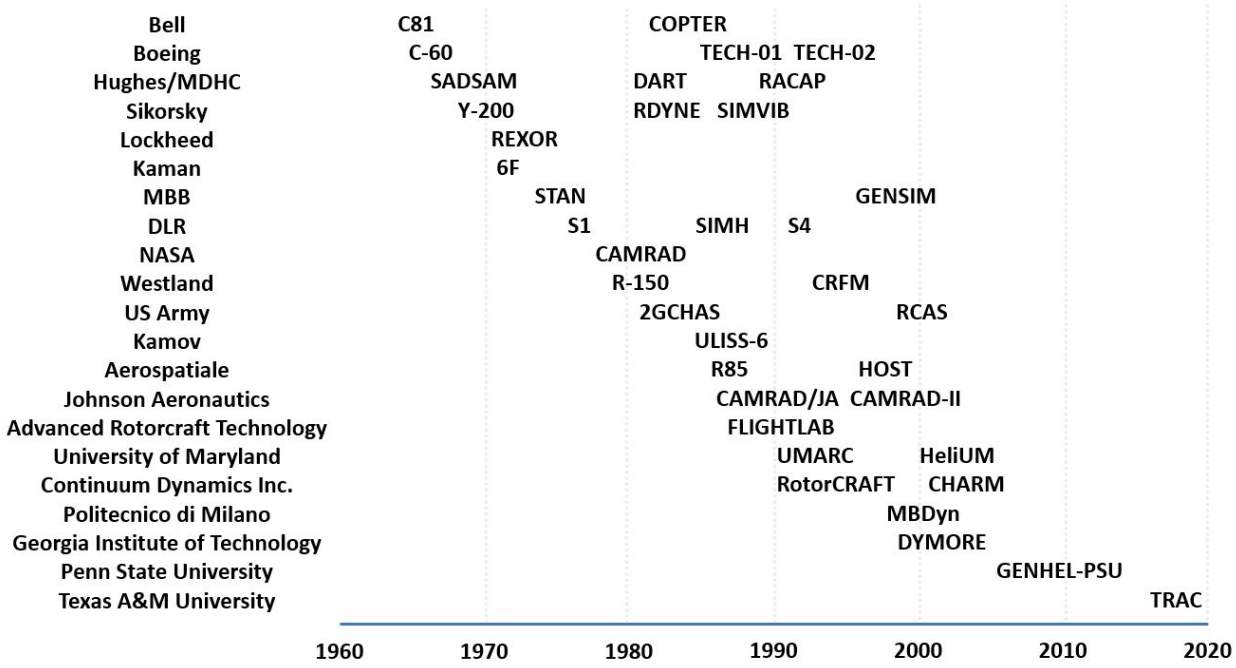



Figure 1.9: Comprehensive helicopter flight dynamics modeling frameworks

The helicopter flight dynamics models were first introduced by the helicopter manufacturing companies during the early years followed by government organizations and university-based research groups. These models have been used for various purposes such as trim and stability analyses, optimizing handling qualities, and flight control design. The modeling frameworks can be categorized based on the level of fidelity defined according to the characteristics of aerodynamics and dynamics models as shown in Table 1.3. With the computational resources available today, many of the modern helicopter flight dynamics modeling tools can yield accurate predictions that closely match flight testing results.

| Fidelity level | Aerodynamics Characteristics | Dynamics Characteristics |
|---|---|---|
| Level 0 | Linear airfoil aerodynamics<br>Static uniform inflow<br>Averaged aerodynamic loads | Rigid blades with<br>steady-state flapping motion |
| Level 1 | Linear airfoil aerodynamics<br>Dynamic inflow<br>Analytically integrated aerodynamic loads | Rigid blades with<br>quasi-steady motion |
| Level 2 | Nonlinear airfoil aerodynamics<br>Dynamic inflow<br>Local effects of blade-vortex interaction<br>Numerically integrated aerodynamic loads | Rigid blades with<br>flap and lag dynamics |
| Level 3 | Nonlinear 3D aerodynamics with full wake analysis<br>Unsteady aerodynamics and compressibility effects<br>Numerically integrated aerodynamic loads | Elastic blades |

Table 1.3: Classification of helicopter flight dynamics models [7]

Level 0 and 1 models are based on linear airfoil aerodynamics and rigid blade dynamics with steady steady motion, which is far from the state of the art compared to current standards of modeling. Level 2 and 3 models can capture complicated airflow and blade motions thus yield sophisticated predictions validated by experimental data.

Comprehensive Analytical Model of Rotorcraft Aerodynamics and Dynamics (CAMRAD) was developed at NASA Ames Research Center and the U.S. Army in 1980. Later in 1988, CAMRAD/JA was developed by Johnson Aeronautics. The dynamics model of CAMRAD/JA includes a single main rotor, tandem main rotor, and coaxial rotor systems. It can also model articulated, hingeless, gimballed, and teetering rotor systems with elastic fuselage and blades. The aerodynamics model of CAMRAD/JA covers dual-peak blade circulation distributions, second-order lifting-line theory, roll-up representations, swept blade tips, and rotor-body interactions. Also, the computational fluid dynamics (CFD) model is loosely coupled with prescribed air load increments and local angle of attack computations for boundary conditions. The wake model of CAMRAD/JA,

which is based on the Scully free-wake geometry, was also used in the other modeling frameworks such as COPTER, UMARC, and 2GCHAS.

FLIGHTLAB was developed by Advanced Rotorcraft Technology Inc. in 1989. This commercially available framework developed as a real-time blade element helicopter simulation program has been updated until today. The dynamics model of FLIGHTLAB includes a rigid fuselage and elastic blades with coupled flapping and lagging motions. The aerodynamics model of FLIGHTLAB uses the Peters-He finite-state dynamic inflow [18].

University of Maryland Advance Rotorcraft Code (UMARC) was developed in 1990. It can model various configurations including articulated, hingeless, bearingless rotors, servo-flap control, and tiltrotor aircraft. The dynamics model of UMARC covers both rigid and elastic fuselage and blades with coupled flapping, lagging, and torsion. The aerodynamics model of UMARC uses the Bagai–Leishman relaxation-free wake model [19]. It has been used for investigation of coupled trim and rotor response, blade and hub loads, aeroelastic stability, ground and air resonance, vibration, gust response, higher harmonic control, dynamics of composite rotors, optimization, and has been coupled with CFD.

HeliUM was introduced in 2010 [20]. It was derived from the original GenHel [21]. The dynamics model of HeliUM includes rigid fuselage and elastic blades with coupled flapping, lagging, and torsion. The aerodynamics model of HeliUM includes the Bagai–Leishman relaxation-free wake [19] and the Bhagwat–Leishman time-marching model [22]. It can find a trim solution, extract linearized models, and obtain responses from pilot input.

Texas A&M University Rotorcraft Analysis Code (TRAC) was derived from the Helium by the author [23, 24], with some modifications in aerodynamics and more added features. The TRAC has three major functionalities which are (1) modeling comprehensive helicopter flight dynamics with fast computation and reasonable accuracy, (2) adopting and analyzing a new component or new aircraft configuration, and (3) integrating a control system into the model for autonomous flight simulations.

UH-60 helicopter is selected as a baseline rotorcraft for TRAC due to a large amount of flight

test data available for validation. The fuselage, main and tail rotor, horizontal tail (stabilator), and vertical tail (fin) are individually modeled and integrated into a complete UH-60 model. Instead of using a free wake model or CFD for fast computation, the rotor component model adopts Pitt-Peters linear inflow model [25, 26] to simulate the main rotor wake, and experimental data to predict fuselage drag and look-up tables for aerodynamic coefficients. This mathematical UH-60 helicopter model computes trim results for helicopter attitude, rotor blade flapping and lagging angles, required power, and control inputs at given flight conditions, which are validated by the US Army flight test data [27].

Since TRAC integrates multiple sub-component models into one, it is convenient to add or remove a component and analyze the effect. It can also model a new aircraft configuration conveniently by modifying the existing components as shown in Fig. 1.10. For example, TRAC was used to develop a coaxial rotor personal air vehicle (PAV) model and conduct extensive flight simulations as part of the Boeing GoFly challenge [28].

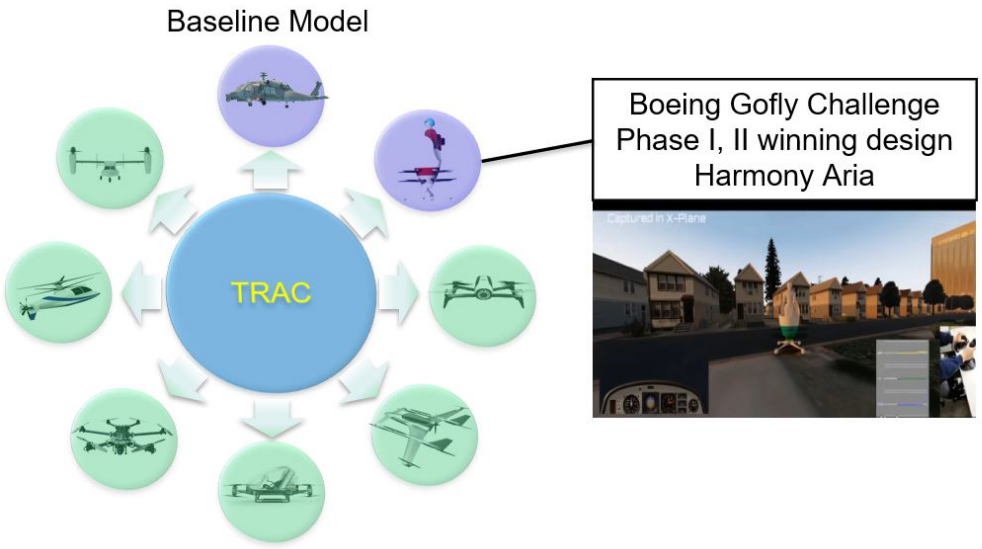



Figure 1.10: Expandability of rotorcraft flight dynamics modeling framework TRAC

Beyond the trim analysis, TRAC can extract linearized models at various flight conditions

such as hovering, vertical climb and descent, level forward flight, climb and descent with different forward velocities, and coordinated turn with or without climb and descent. All of the linear models are extracted based on the first-order Taylor series expansion, which can be coupled with any linear control system. Linear Quadratic Regulator (LQR), which is a class of optimal control systems, is constructed using the extracted model for non-zero setpoint tracking. The full flight simulation can be accomplished by the LQR control system in the MATLAB program and visualized in the X-plane flight simulator.

At an earlier stage of the study towards automating real helicopter ship landing, TRAC is used to prove the concept by simulating fully autonomous flight including steady forward flight, coordinated turn, descent, deceleration, and final landing on a ship deck. For each phase of this complex flight maneuver, different linearized models for the UH-60 helicopter dynamics are extracted.

### 1.3 Autonomous Flight Control Systems

There are numerous ways to configure the flight control system for autonomous ship/moving-platform landing and various control algorithms have been applied such as proportional-derivative (PD) control [29, 30, 31], proportional-integral-derivative (PID) control [32, 33, 34], gain-scheduled PID control [35], linear quadratic regulator (LQR) [24, 23, 36], adaptive control [37, 38, 39], nonlinear control [40, 41, 42, 43], and reinforcement learning-based control [44, 45]. In the present study, to figure out the most appropriate control solution, a wide variety of control techniques are implemented such as gain-scheduled PID control, nonlinear PID control, LQR control, and reinforcement learning-based control along with the moving average filter and/or Kalman filter.

Depending on the type of sensors used to obtain information, autonomous ship landing control systems can be classified mainly by either GPS- or vision-based methods. Approaches using GPS can be used anywhere on the globe and the accuracy has been progressed remarkably. On the other hand, approaches using vision-based navigation are operational in a GPS-denied environment. Thus, previous researches regarding the GPS- and vision-based approaches have been thoroughly studied to understand the current state and to analyze the pros and cons of each approach.

### 1.3.1 GPS-based approach

First, GPS-based navigation methods are widely surveyed and categorized as shown in Table 1.4, including landing on a ship/moving platform and ground/runway.

| Ref. | Controller | Aircraft | System Details |
|---|---|---|---|
| [34], [31] | PID [34],<br>PD with fuzzy<br>logic [31] | VTOL | 6 DOF simulations<br>Pitch, roll and altitude control |
| [41], [42]<br>[43] | Nonlinear Feedback<br>Linearization | Fixed Wing [41], [43],<br>VTOL [42] | 3 DOF aircraft model<br>Ground effect |
| [46] | Sliding Mode Control | Fixed Wing | 6 DOF Nonlinear Model<br>Comparison with PID controller<br>Lyapunov stability criteria |
| [47], [48]<br>[6] | Backstepping and<br>Neural Networks | Fixed Wing [47],<br>VTOL [48], [6] | 6 DOF Model<br>Flapping correction and<br>servo dynamics |
| [49] | Hybrid Control | Simulated for a<br>general UAV | Landing as a sequence of tasks<br>Switching strategy between<br>controllers<br>Design correctness analyzed by<br>a reachability computation |

Table 1.4: Comparison of GPS-based landing methods [8]

Since the altitude measurement from the GPS is not accurate, additional sensors such as radar altimeter, barometric altimeter or ultrasonic sensor are also used in conjunction with GPS. With the advances in GPS technology, the accuracy has progressed remarkably as shown in Table 1.5, which is enabling the use of GPS for autonomous ship landing.

| Category | Mode | Horizontal Accuracy |
|---|---|---|
| Stand-Alone | Civilian receiver, SA on (historical) | 100 m |
| Stand-Alone | Civilian receiver, SA off (current) | 5 - 8 m |
| Stand-Alone | Military receiver, (dual frequency) | 3 - 5 m |
| Differential | Code differential | 1 - 3 m |
| Differential | Carrier-smoothed code differential | 0.1 - 1 m |
| Differential | Precise carrier-phase (kinematic) | 1 - 2 cm |
| Differential | Precise carrier-phase | 1 - 2 mm |

Table 1.5: Typical GPS Accuracy [9]

Joint Precision Approach and Landing System (JPALS) was developed by the department of the Navy [5, 50, 51].

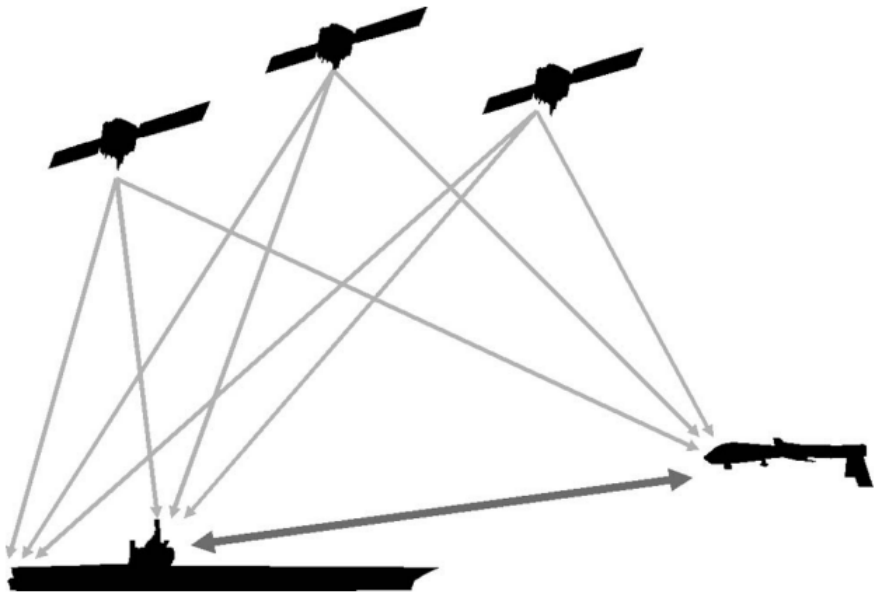


Figure 1.11: JPALS navigation signals include GPS broadcasts and a ship-based communication link [5]

To approach and land on a ship, JPALS acquires the orientation states of the landing deck

with respect to the earth-fixed frame, then the aircraft matches its attitude to the deck's attitude. In addition, the GPS sensors are located around the landing deck and the relative position of the landing point from the sensors should be known. A unique challenge with the ship landing is that the flight deck is constantly moving. This increases uncertainty and the chance of error during aircraft ship landing, which makes it extremely challenging compared to landing on the stationary ground. Thus, the appropriate placement of the GPS sensors on a ship is important to obtain the pose data accurately. It is known that a distinctive spacing between the positions of sensors helps to increase the accuracy. Thus sensor accuracy plays a huge role in the precise approach and landing while using this method. The advantage of JPALS is that it can be used in low visibility conditions and has good precision in approach and landing. However, its application is limited to landing on an aircraft carrier that has a large flight deck and is less sensitive to sea states. Also, it requires multiple GPS antennas on a ship and reprogramming for a particular ship depending on the GPS antenna placement. Above all, it cannot be operational when the GPS signals are unavailable or spoofed.

Voos et al. [42] uses a nonlinear controller taking differential GPS (DGPS) positioning data as states to land a quadrotor UAV on a moving platform. It comprises an inner-loop attitude control system and an outer-loop velocity control system. It linearizes the UAV model to decouple the differential equations then configure the inner-loop attitude control. The outer-loop velocity control system contains sinusoidal nonlinear terms in feedback gains. The main advantage of the overall control system is that the feedback linearization and the controllers are comparatively fast and convenient to be implemented while taking the full nonlinear behavior of the vehicle into account. However, the overall control algorithm requires accurate measurements of all state variables from the DGPS and inertial measurement unit (IMU). Also, the proposed method is suitable for cruising, but not sufficient for landing due to the lack of precise measurement of the vertical relative distance between the UAV and platform.

Ahmed et al. [6] presents the control method to land a small helicopter UAV on a moving platform using DGPS and tether as shown in Fig. 1.12.

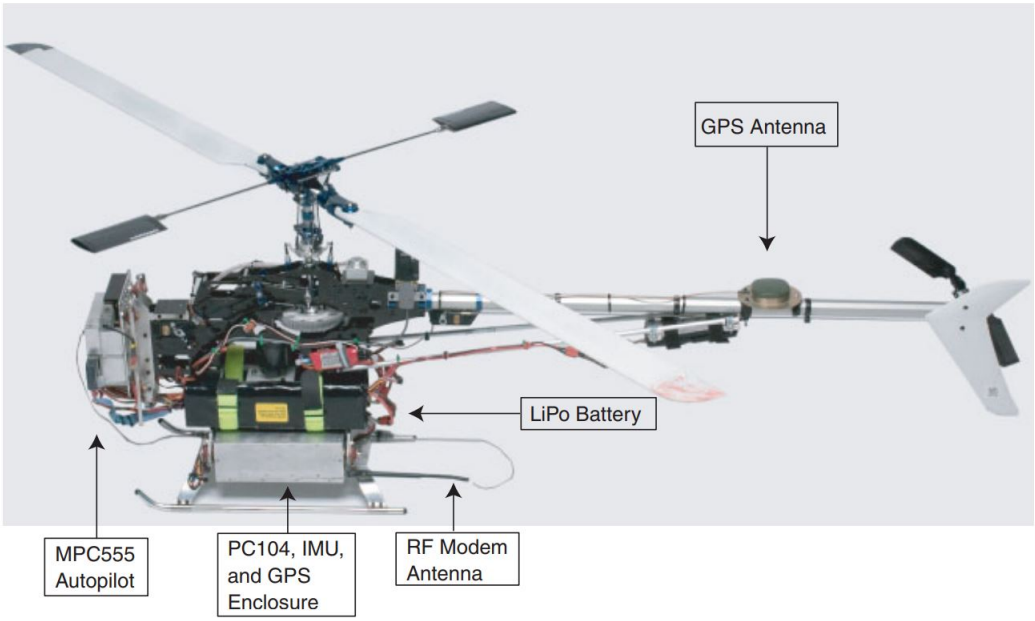



Figure 1.12: Instrumented experimental small helicopter UAV, Eagle [6]

One end of the tether is secured to the moving platform deck and the other end is reeled out from the hovering UAV. The proposed controller is designed based on a backstepping method, which is a recursive design procedure that links the selection of a control Lyapunov function with the design of a feedback control system and guarantees global asymptotic stability of strict feedback systems. It also includes blade flapping and servo dynamics. The DGPS on the UAV provides the position data with 2cm accuracy. Also, rate gyroscopes provide angular velocities and accelerometers provide translational accelerations. Using these sensor data, it computes four control inputs which are longitudinal cyclic, lateral cyclic, collective, and yaw control inputs. Although simulations demonstrate the possibility of autonomous landing using this method, it requires the installation of a tether on the moving platform and depends on the accuracy of DGPS.

There are also GPS-based methods for landing on small ships. Yang et al. [52] propose the control method to land a small helicopter UAV on a ship by synthesizing information from GPS, IMU, and vision sensors. The extended Kalman filter (EKF) fuses data from the sensors to provide navigation information. The developed method simulates ship landing and the accuracy is 2 - 4 cm. However, ship motion is assumed as constant linear speeds along three axes (heave, sway, surge) without considering the angular motions (pitch, roll, yaw).

In 2012, Hardesty et al. [53] demonstrated autonomous helicopter landing on a ship. The real-time kinematic (RTK) algorithms solve for the relative position vector from the base to the receiver on a helicopter, which allows the computation of precise relative position. Continuous communication between the receiver and the base station as well as maintaining a sufficient number of common satellites are required.

In conclusion, the GPS-based navigation approaches heavily depend on the accuracy of GPS and their applications are not operational in the GPS-denied environment, which is the motivation for vision-based solutions. A vision-based ship landing system is a particular point of interest in this research because it is crucial to the automation of the Navy helicopter ship landing procedure, which is the objective of the present study.

### 1.3.2 Vision-based approach

Regarding vision-based approaches in ship landing applications, classical computer vision has been widely applied so far. All the previous studies have a common process that is to estimate or track ship deck motions first, then control the aircraft attitude to match the ship motions for landing. In order to extract the ship deck motion information, specific objects or patterns are targets for detection such as H landing marking [54], T landing marking [55], points dispersed on the deck [33], lights [30], and infrared cooperated targets on a ship [56, 57] as shown in Fig. 1.13.

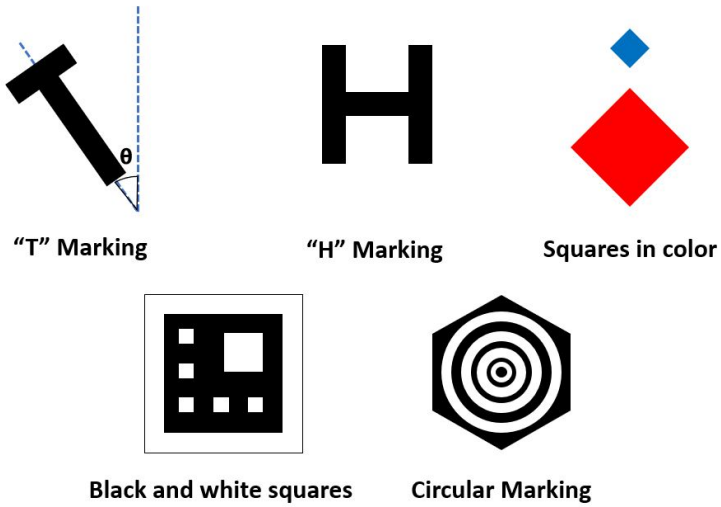



Figure 1.13: Example fiducial markings on a landing deck

To detect such fiducial markings on a deck or features of a ship, several computer vision algorithms are used such as edge and line detection [58, 59], corner detection [60, 61], and contour detection [62].

Xu et al. [55] proposed a method to detect the "T" marking by using infrared images. It calculates the yaw using the centerline of the marking and reference line. The use of infrared images makes this technique effective at night as well.

Saripalli et al. [63, 64] tracks the "H" marking for ship landing. First, it converts the obtained image to greyscale, then a median filter removes the noise in the image. Second, the area that contains the marking is obtained by segmentation and labeling. Using the helicopter height information from GPS, the relationship between the image plane and the landing pad plane is derived. It demonstrated 7cm of accuracy in position estimation and 4° of accuracy in orientation accuracy.

Wang et al. [58, 59] designed a unique marking on the deck that has two rectangles, which have different sizes and colors with white backgrounds. The bigger square is used for landing deck identification and estimation of relative position. The smaller square is used to estimate relative orientation. To detect each side of the square, it uses both edge detection by the Sobel operator and line detection that combines the Hough transform and least square regression. To estimate relative position and orientation, it uses the theory of perspective projection, the pinhole model of the camera, and the known relationship between each side. It demonstrated 2 cm of accuracy in position estimation and 1.2° of accuracy in orientation estimation within 10 meters of range.

Sharp et al. [60] designed a marking that consists of black and white squares in different sizes. First, it processes an image into a binary one and segments the landing area to detect corners of the marking. It also applies both linear and nonlinear optimization algorithms for better state estimation. The flight testing demonstrated 5 cm of accuracy in position estimation and 5° of accuracy in orientation estimation.

Lange et al. [62] designed a circular marking that consists of several concentric white rings with a black background. Each ring diameter is known and the ratios between each ring size are uniquely defined. First, it converts the image into grayscale and then to a binary one. Second, seg-

mentation, contour detection, and roundness check are performed in order. To obtain the position and orientation states, it uses additional sensors such as sonar and optical flow sensors. According to the flight testing results, a minimum and maximum landing deviation from the center are 3.8 cm and 23 cm, respectively.

Vision-based navigation methods including landing on a ship/moving platform and ground/runway are categorized as shown in Table 1.6.

| Ref. | Aircraft | Equipment | Details |
|---|---|---|---|
| [64], [65] | VTOL<br>VTOL | Downward CCD camera<br>Ultrasonic sonar and INS | Invariant moments<br>Kalman filter based tracking |
| [66], [67] | MAV [66], VTOL [67] | Optic flow sensor<br>IMU, GPS<br>Barometric pressure sensor | Optic flow and barometric altimeter based HAG estimation |
| [68] | VTOL | CCD Camera mounted on a PTU<br>Accelerometer and angular rate gyros | Contour extraction<br>Kalman filter fusion |
| [55] | Not specified | Treatment on an object<br>Infrared thermal imager<br>GPS and INS | Recognition from infrared images<br>Affine moment invariants |
| [69], [70], [71] | Not specified | Two cameras | Monocular, stereo, machine vision<br>EKF, Hough transform<br>Vanishing geometry<br>RANSAC algorithm |
| [35], [72] | VTOL | Single camera, IMU | Machine learning-based vision<br>Contour and corner extraction<br>Nonlinear control with Kalman<br>Reinforcement learning-based control |

Table 1.6: Comparison of vision-based landing methods

Even though several vision-based methods were proposed that could be applicable in the proximity of the ship, visually tracking fiducial markings on the deck is fundamentally not ideal for aircraft ship landing for the following reasons. First, it is challenging to track the highly uncertain and complicated ship motions at sea. It requires utilizing an array of powerful sensors and complex algorithms to capture the motions of the ship deck. Even if one could capture the ship motions precisely, its application range is limited to the vertical space where the deck can be clearly captured in the image as shown in Fig. 1.14.

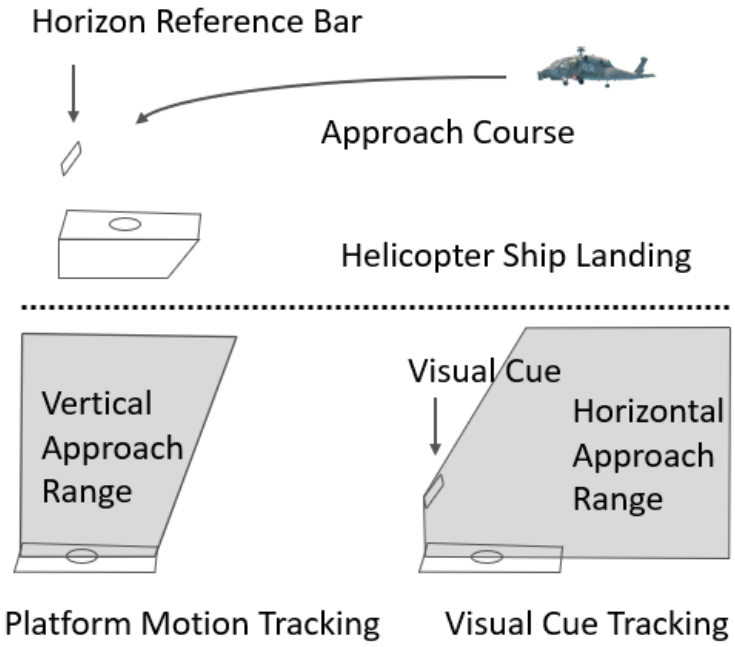



Figure 1.14: Limited application range of platform/deck tracking method

Also, actively controlling the UAV to match the complex deck motions could excite dynamically unstable modes of the aircraft. This is even more unsafe when the aircraft is close to the moving deck because even a small control error can be catastrophic due to an impact by the deck. Moreover, any roll/pitch motion of the UAV to match the deck motion tilts the thrust vector, which induces translational motions leading to the aircraft circling over the deck before landing as seen in some of the previous studies [39, 30, 73, 54]. Furthermore, none of the previous methods were

based on the established Navy helicopter ship landing procedure that is successfully executed by pilots for decades. In fact, visually tracking the moving landing deck is exactly the opposite of what Navy helicopter pilots are trained to do.

In conclusion, the vision-based navigation method is selected over GPS-based navigation to automate the real helicopter ship landing as it is. In the present study, the vision system is developed in two different ways depending on the relative distance from the ship. In order to track a ship from a long distance, the machine/deep learning-based vision technique is applied due to its long-range detection capability. In the close distance, classical computer vision techniques are used not to track features on a moving deck, but to track the horizon reference bar which is installed parallel to the aircraft approach course and indicates the true horizon independent of ship motions. By using this flight control strategy, it is able to control the aircraft in a stable and precise fashion for the final approach and landing in the same manner that Navy pilots fly a helicopter for ship landing.

### 1.4 Objectives of Study and Thesis Organization

As seen from the literature survey on autonomous ship landing, previous methods have not been developed by closely following the real helicopter ship landing procedure. Instead, ship landing has been regarded as a control task, and therefore, those studies focused on technical details to detect certain markings on the deck, capture the deck motions, and match the aircraft to the deck motions for ship landing. However, this is completely orthogonal to the real helicopter ship landing procedure where the aircraft hovers in a stable fashion over the deck and lands vertically independent of ship motions. Therefore, unlike the previously proposed methods, the present study aims to develop a highly practical autonomous ship landing solution for vertical flight aircraft based on an in-depth understanding of the Navy helicopter ship landing procedure. Moreover, this novel autonomous ship landing method can be used for manned and unmanned aircraft without modifying the current Navy procedure. The specific objectives of the study are described below.

In order to prove the proposed concept in simulation, a comprehensive rotorcraft flight dynamics modeling framework called TRAC, including LQR optimal control system is developed. It is important to have a high-fidelity model that is incorporated into the flight control system for re-

alistic helicopter ship landing simulations. The UH-60 helicopter model is derived as a baseline helicopter model. Power, shaft tilt angles, and control inputs at different trimmed flight conditions are validated through comparison with flight test data. Also, LQR optimal control system is constructed with the model to demonstrate the feasibility of the developed method via ship landing simulations. In Chapter 2, the details of the modeling framework are described and the corresponding ship landing simulation with the LQR control system design is detailed in Chapter 4.

After the simulation of the novel ship landing method, the autonomous vision-based flight control system to be implemented in actual flight experiments is developed. The first step is to develop a vision system that provides information for the autonomous flight control system. The vision system can be divided into two parts, which are machine/deep learning-based vision applied in long-range tracking and classical computer vision applied in close-range. To detect a ship and horizon bar as a whole at a long distance, machine/deep learning-based object detectors are developed. However, classical computer vision techniques are used to develop a close-range object detector that detects corner points on the bar and estimates the relative position and orientation of the aircraft. In Chapter 3, details of the machine vision for long- and close-range detections are described.

To control the aircraft using the perceived visual information, several control systems are developed and implemented. At earlier stages of development, linear PID and gain-scheduled PID controllers are implemented with different flight modes. Later, nonlinear control systems are developed and implemented to enhance the tracking capabilities in the presence of sensor noise and time delay. Also, the state-of-the-art reinforcement learning-based control is introduced to improve the robustness of tracking capabilities in more challenging conditions. In Chapter 4, the development and effect of control systems with their simulation results are detailed.

To demonstrate the reliability and capabilities of the control system in practice, extensive indoor and outdoor flight tests are conducted. A sub-scale ship landing platform is constructed to reproduce ship structure and deck motions using a 6 DOF Stewart platform. A VTOL capable quad-rotor-UAV live streams a video to a ground-station computer, which processes the image

frames to transmit control inputs through WIFI. A crucial element of the developed ship landing method is landing vertically on the deck independent of ship motions. Thus, multiple landing tests are conducted while the deck is in motion. To this end, realistic ship motions that simulate the Oliver Hazard Perry-class frigate at sea state 6 and NATOPS helicopter ship landing limits are implemented using a 6 DOF motion platform. Vertical landings are executed at random instances of deck motions. Full flight tests are conducted in challenging conditions such as GPS-denied environments, unpredictable and complicated ship motions, and adverse weather conditions. In Chapter 5, extensive flight testing results are detailed, followed by the conclusions inferred from the study in Chapter 6.

# 2. ROTORCRAFT FLIGHT DYNAMICS MODELING FRAMEWORK

This chapter describes the mathematical modeling methodology of Texas AM University Rotorcraft Analysis Code (TRAC), including details on the development of each component model, trim analysis, and linearized model extraction. A UH-60 helicopter is used as a baseline model and validated through flight test data. The first section describes the governing equations of the system in state-space form and the second section defines coordinate systems used in modeling. The third section describes the computation of forces and moments from the helicopter component models. The fourth section details the trim analysis and the last section explains the method to extract a linearized model. UH-60 parameters used in the modeling are presented in Table A.1.

## 2.1 Governing Equations

The governing equations of the system are formulated in state-space form as a system of first-order nonlinear coupled ordinary differential equations (ODEs):

$$f(y, \dot{y}, u, t) = \epsilon = 0 \tag{2.1}$$

*y* is a vector of system states, $\dot{y}$ is time derivatives of *y*, *u* is a vector of control inputs, and *t* is the current time in seconds. Numerical solutions of these equations with zero body-axis accelerations for trim are used to study vehicle performance in steady flight. The state vector consists of the following components:

$$y \quad = \quad \{ \quad y_F^T \quad y_\lambda^T \quad y_{rotor}^T \quad \}^T \tag{2.2}$$

- $y_F$ represents the vector of the 9 airframe rigid-body states
- $y_\lambda$ represents the induced inflow coefficients for main rotor and tail rotor
- $y_{rotor}$ represents the vector of rotor deflection states for all blades

$$u \quad = \quad \{ \quad \delta_o \quad \delta_{lat} \quad \delta_{lon} \quad \delta_{ped} \quad \}^T \tag{2.3}$$

The control inputs are the same as the helicopter pilot inputs which are the positions of the collective lever, lateral and longitudinal cyclic stick, and the pedal, in order.

## 2.2 Coordinate Systems

Various reference frames are used for efficient modeling and simulations. To name a few, there are earth-fixed axes to track a trajectory in the inertial frame, body axes for force and moment equilibrium equations, hub-fixed axes for hub loads, and rotating axes for blade deflections. Displacements and loads must be transferred from one axis system to another through coordinate transformations to use in the governing equations for each component. Mathematically, this rotation can be expressed as the pre-multiplication of a vector (X, Y, Z components) with a rotation matrix. A rotation from one coordinate system to another is performed in "3($\psi$) - 2($\theta$) - 1($\phi$)" Euler angle sequence. The rotations are positive in the anti-clockwise sense. Although it has a singularity at a rotation angle of $\pm 90^\circ$, this does not happen in general helicopter maneuver. The rotation matrices for the yaw, pitch, and roll rotations are given in Eqn. 2.4.

$$T_\psi = \begin{bmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} T_\theta = \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix} T_\phi = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{bmatrix} \tag{2.4}$$

Since the sequence occurs in the order Z $\rightarrow$ Y $\rightarrow$ X, the rotation matrices must be premultiplied in this order. Thus, the final rotation matrix from the earth-fixed coordinate system "G" to the body-fixed coordinate system "B" through angles ($\psi$,$\theta$,$\phi$) is computed as described in Eqn. 2.5.

$$T_{BG} \quad = \quad T_\phi \ T_\theta \ T_\psi \quad = \quad R(\psi, \theta, \phi) \tag{2.5}$$

The subscript "BG" means transformation from the ground (earth-fixed inertial frame) to body-fixed frame. The reverse rotation from frame "B" to "G" follows the exact opposite sequence in

reverse, i.e. angles ($-\psi$,$-\theta$,$-\phi$) about the ($X$, $Y$, $Z$) axes. In this case, the rotation matrix is given in Eqn. 2.6.

$$T_{GB} \quad = \quad T_{-\phi}\ T_{-\theta}\ T_{-\psi} \tag{2.6}$$

The rotation from "B" to "G" can be simplified to Eqn. 2.8.

$$T_{GB} \quad = \quad T_{\phi}^{T}\ T_{\theta}^{T}\ T_{\psi}^{T} \tag{2.7}$$

Using the matrix property described in Eqn. 2.8.

$$T_{GB} \quad = \quad (T_{\phi}\ T_{\theta}\ T_{\psi})^{T} \quad = \quad T_{BG}^{T} \tag{2.8}$$

### 2.2.1 Earth-fixed Frame

The earth-fixed axes coordinates represent an inertial reference system used to track the motion of objects in space. The origin of this axis system is a fixed point on the ground. The unit vectors along the earth-fixed axes are represented by ($i_G$, $j_G$, $k_G$). The earth-fixed axes are oriented so that $i_G$ points North, $j_G$ points East and $k_G$ points towards the ground. The position vector of the helicopter CG in space is given in Eqn. 2.9.

$$r_{CG} \quad = \quad x_{CG}\, i_G \quad + \quad y_{CG}\, j_G \quad + \quad z_{CG}\, k_G \tag{2.9}$$

### 2.2.2 Helicopter Body-fixed Frame

The body axes of the helicopter, shown in Fig. 2.1, are obtained from the earth-fixed coordinates using three transformations to shift the origin to the helicopter center of gravity, followed by "3($\psi$) - 2($\theta$) - 1($\phi$)" Euler angle rotations. It is positive for nose-right, pitch-up, and roll-right motions respectively. The unit vectors along the body axes are calculated as shown in Eqn. 2.10.

$$\begin{bmatrix} i_B \\ j_B \\ k_B \end{bmatrix} = T_{BG} \begin{bmatrix} i_G \\ j_G \\ k_G \end{bmatrix} \tag{2.10}$$

The rotation matrix from inertial to helicopter body axes is given in Eqn. 2.11.

$$T_{BG} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi_F & \sin\phi_F \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \cos\theta_F & 0 & -\sin\theta_F \\ 0 & 1 & 0 \\ \sin\theta_F & 0 & \cos\theta_F \end{bmatrix} \begin{bmatrix} \cos\psi_F & \sin\psi_F & 0 \\ -\sin\psi_F & \cos\psi_F & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.11}$$

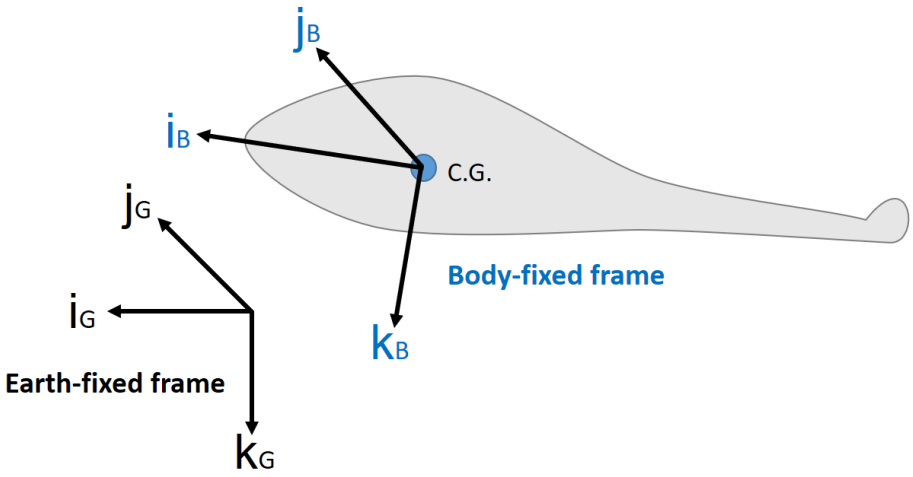



Figure 2.1: Earth-fixed frame and helicopter body-fixed frame

### 2.2.3 Blade Non-rotating Frame

The blade non-rotating axes, shown in Fig. 2.2, is transformed from the helicopter body frame, followed by two Euler rotations $\alpha_s$ which is a shaft longitudinal tilt angle, $\beta_s$ which is a shaft lateral tilt angle in the order Y $\rightarrow$ X, followed by a $180^\circ$ rotation about the intermediate Y-axis. The first

two rotations are positive when the shaft tilt causes the hub to move aft and starboard, respectively. The new origin of this axis system is at the center of the hub. In the case of UH-60, the shaft tilts $3^\circ$ forward and has no lateral tilt angle. The unit vectors along the blade non-rotating frame are given in Eqn. 2.12.

$$\begin{bmatrix} i_{NR} \\ j_{NR} \\ k_{NR} \end{bmatrix} = T_{NB} \begin{bmatrix} i_B \\ j_B \\ k_B \end{bmatrix} \tag{2.12}$$

The rotation matrix from the helicopter body frame to the blade non-rotating frame is expressed in Eqn. 2.13.

$$T_{NB} = \begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\beta_s & \sin\beta_s \\ 0 & -\sin\beta_s & \cos\beta_s \end{bmatrix} \begin{bmatrix} \cos\alpha_s & 0 & -\sin\alpha_s \\ 0 & 1 & 0 \\ \sin\alpha_s & 0 & \cos\alpha_s \end{bmatrix} \tag{2.13}$$

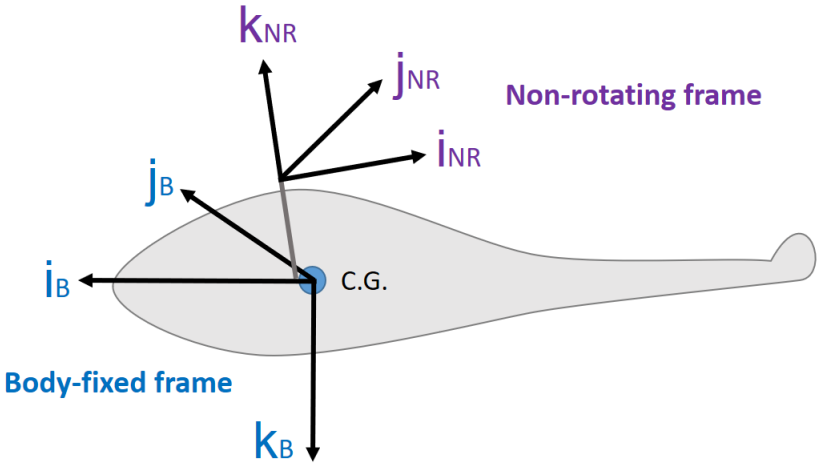



Figure 2.2: Blade non-rotating frame

### 2.2.4 Blade Rotating Frame

The blade rotating frame, shown in Fig. 2.3, is obtained from the blade non-rotating frame using one rotation $\psi_n$ about the blade non-rotating Z-axis $k_H$. The origin of the blade rotating frame is at the center of the hub which is identical to the origin of the blade non-rotating frame. The $\psi_n$ is the azimuth angle of the $n^{th}$ blade, zero when the blade passes over the tail boom, positive in counter-clockwise direction and is given by $\psi_n = \Omega_{MR}\, t + \frac{2\pi}{N_b}(n - 1)$. The unit vectors along the blade rotating frame are given in Eqn. 2.14.

$$\begin{bmatrix} i_R \\ j_R \\ k_R \end{bmatrix} = T_{RN} \begin{bmatrix} i_{NR} \\ j_{NR} \\ k_{NR} \end{bmatrix} \tag{2.14}$$

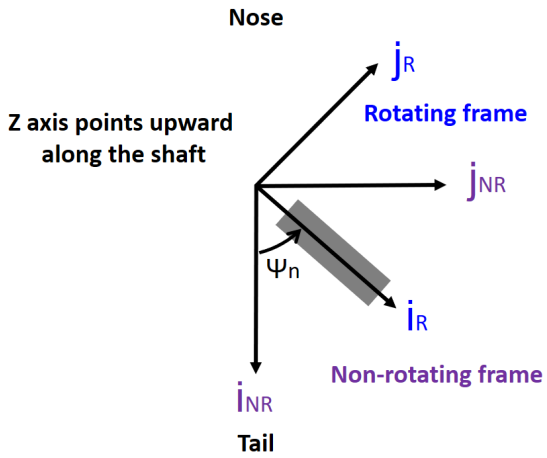



Figure 2.3: Blade rotating frame

The rotation matrix from the blade non-rotating frame to the blade rotating frame is given in Eqn. 2.15.

$$T_{RN} = \begin{bmatrix} \cos\psi_n & \sin\psi_n & 0 \\ -\sin\psi_n & \cos\psi_n & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.15}$$

### 2.2.5 Blade Flapped Frame

The blade flapped frame, shown in Fig. 2.4, is obtained from the blade rotating frame using one rotation through an angle -$\beta_n$ about the $j_R$ blade rotating axis, and is positive for vertically upward motion of the blade tip. The value of an angle $\beta_n$ depends on the azimuthal location of the $n^{th}$ blade. The origin of the blade flapped frame is identical to the origin of the blade rotating frame. The unit vectors along the blade flapped frame are given in Eqn. 2.16.

$$\begin{bmatrix} i_F \\ j_F \\ k_F \end{bmatrix} = T_{FR} \begin{bmatrix} i_R \\ j_R \\ k_R \end{bmatrix} \tag{2.16}$$

The rotation matrix from the blade rotating axes to the blade flapped axes is given in Eqn. 2.17.

$$T_{FR} = \begin{bmatrix} \cos\beta_n & 0 & \sin\beta_n \\ 0 & 1 & 0 \\ -\sin\beta_n & 0 & \cos\beta_n \end{bmatrix} \tag{2.17}$$

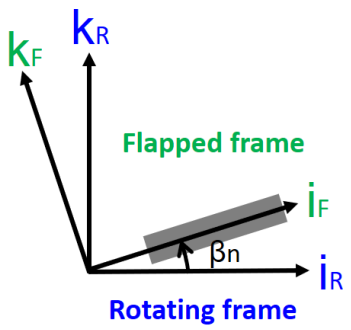



Figure 2.4: Blade flapped frame

### 2.2.6 Blade Lagged Frame

The blade lagged frame, shown in Fig. 2.5, is obtained from the blade flapped frame using one rotation through an angle -$\zeta_n$ about the $k_F$ blade flapped frame, and is positive for horizontally

forward (lead) motion of the counter-clockwise rotating $n^{th}$ blade tip. The origin of the blade lagged frame is identical to the origin of the blade flapped frame. The unit vectors along the blade lagged frame are given in Eqn. 2.18.

$$\begin{bmatrix} i_L \\ j_L \\ k_L \end{bmatrix} = T_{LF} \begin{bmatrix} i_F \\ j_F \\ k_F \end{bmatrix} \tag{2.18}$$

The rotation matrix from the blade flapped frame to the blade lagged frame is given in Eqn. 2.19.

$$T_{LF} = \begin{bmatrix} \cos\zeta_n & -\sin\zeta_n & 0 \\ \sin\zeta_n & \cos\zeta_n & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.19}$$

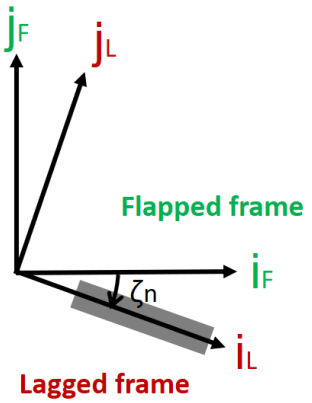



Figure 2.5: Blade lagged frame

### 2.2.7 Tip Path Plane

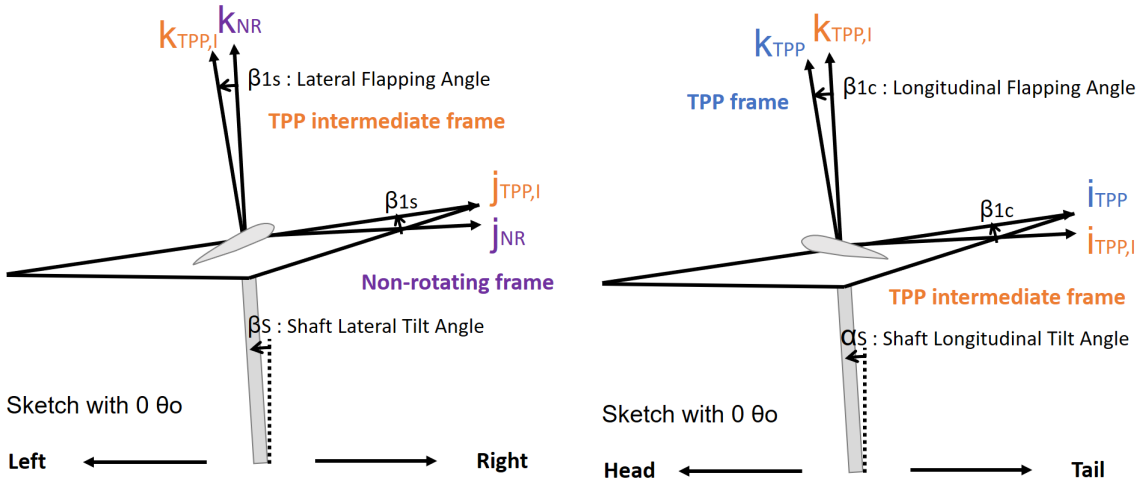
It is convenient to define Tip Path Plane (TPP) since the thrust vector is perpendicular to the TPP. The tip path plane is the plane that contains the path described by the blade tips, assuming that the blade motion is first harmonic only (1/rev). An observer on the tip path plane will see only the coning angle $\beta_o$. Hence, the rotor blade flapping is not a function of blade azimuth ($\psi$) if it

is defined with respect to the TPP. However, the blade pitch angle $\theta$ will be a function of $\psi$. The TPP frame, shown in Fig. 2.6, is obtained from the blade rotating frame using rotations through an angle $\beta_{1s}$ about the $i_{NR}$ blade non-rotating axis and through an angle -$\beta_{1s}$ about the $j_{TPP,I}$ TPP intermediate axis. The unit vectors along the TPP frame are given in Eqn. 2.20.

$$\begin{bmatrix} i_{TPP} \\ j_{TPP} \\ k_{TPP} \end{bmatrix} = T_{TN} \begin{bmatrix} i_{NR} \\ j_{NR} \\ k_{NR} \end{bmatrix} \tag{2.20}$$

The rotation matrix from the blade non-rotating axes to the TPP axes is given in Eqn. 2.21.

$$T_{TN} = \begin{bmatrix} \cos\beta_{1c} & 0 & \sin\beta_{1c} \\ 0 & 1 & 0 \\ \sin\beta_{1c} & 0 & \cos\beta_{1c} \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\beta_{1s} & \sin\beta_{1s} \\ 0 & -\sin\beta_{1s} & \cos\beta_{1s} \end{bmatrix} \tag{2.21}$$




(a) TPP in rear view

(b) TPP in side view

Figure 2.6: Geometry of tip path plane

## 2.3 Component Modeling and Integration

The modeling methodology of TRAC is to integrate individually developed component models as one complete aircraft model. To this end, a complete UH-60 helicopter is decomposed to each modeling part as shown in Fig. 2.7.

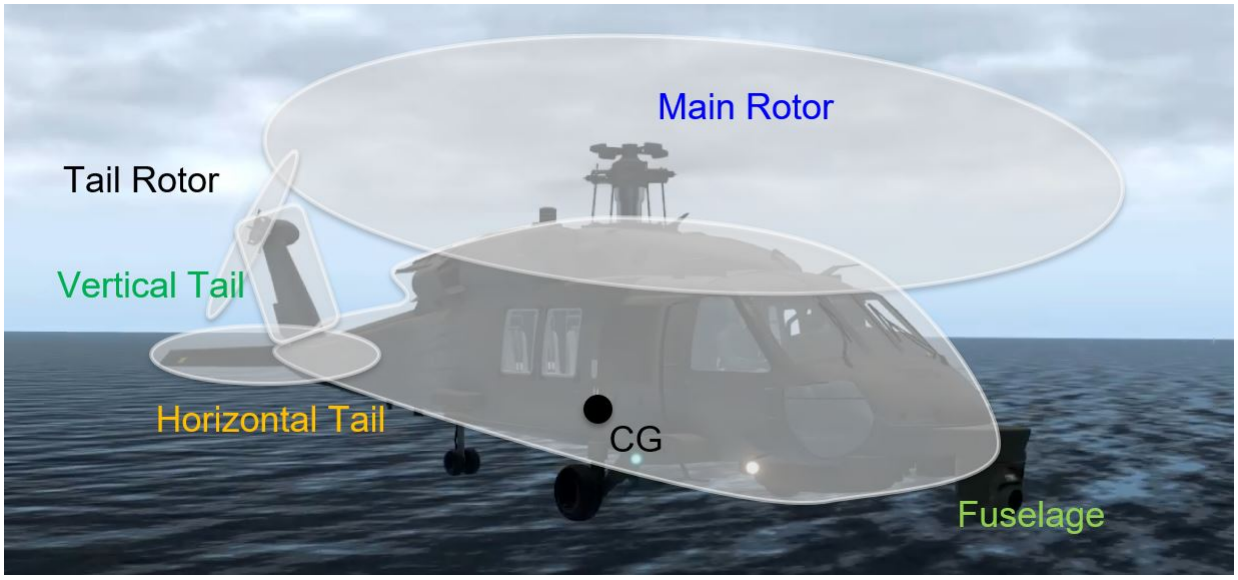



Figure 2.7: UH-60 helicopter modeling parts

Since the fuselage is regarded as a rigid body, the position and orientation of the lifting surfaces such as the main rotor, tail rotor, horizontal tail (stabilator), and vertical tail (fin) remain constant in the body-fixed frame. Furthermore, the moments of inertia of a rigid body stay the same in the body-fixed frame. Hence, force and moment contributions from each component are integrated to formulate the force and moment equilibrium equations along the helicopter fuselage body-fixed frame as described in Eqn. 2.22.

$$
\begin{aligned}
X &= X_{MR} + X_{TR} + X_H + X_V + X_F \\
Y &= Y_{MR} + Y_{TR} + Y_H + Y_V + Y_F \\
Z &= Z_{MR} + Z_{TR} + Z_H + Z_V + Z_F \\
L &= L_{MR} + L_{TR} + L_H + L_V + L_F \\
M &= M_{MR} + M_{TR} + M_H + M_V + M_F \\
N &= N_{MR} + N_{TR} + N_H + N_V + N_F
\end{aligned}
\tag{2.22}
$$

The subscripts which are MR, TR, H, V, and F denote the main rotor, tail rotor, horizontal tail, vertical tail, and fuselage, respectively. The mathematical modeling for loads generated by each component is detailed in the following sections.

### 2.3.1 Fuselage

The inertial loads of a fuselage can be computed from the body-axis components of the airframe linear and angular velocities. These components are obtained from the partition of the system state vector that contains the fuselage states as given in Eqn. 2.23.

$$
y_F = \{ \; u_F \;\; v_F \;\; w_F \;\; p_F \;\; q_F \;\; r_F \;\; \phi_F \;\; \theta_F \;\; \psi_F \; \}^T \tag{2.23}
$$

The terms ($u_F$, $v_F$, $w_F$, $p_F$, $q_F$, $r_F$) are the linear velocities and angular velocities of the helicopter along and about body-fixed frame and ($\phi_F$, $\theta_F$, $\psi_F$) are the Euler angles which define the fuselage orientation with respect to the earth-fixed frame. The fuselage force equilibrium equations are given in Eqn. 2.24.

$$
\begin{aligned}
X_F &= m_F(\dot{u_F} + q_F w_F - r_F v_F + g \sin\theta_F) \\
Y_F &= m_F(\dot{v_F} + r_F u_F - p_F w_F - g \sin\phi_F \cos\theta_F) \\
Z_F &= m_F(\dot{w_F} + p_F v_F - q_F u_F - g \cos\phi_F \cos\theta_F)
\end{aligned}
\tag{2.24}
$$

$p_F$, $q_F$, and $r_F$ can be expressed in terms of the Euler angles ($\phi_F$, $\theta_F$, $\psi_F$) and their time derivatives as shown in Eqn. 2.25.

$$
\begin{aligned}
p_F &= \dot{\phi_F} - \dot{\psi_F} \sin\theta_F \\
q_F &= \dot{\theta_F} \cos phi_F + \dot{\psi_F} \cos\theta_F \sin\phi_F \\
r_F &= -\dot{\theta_F} \sin phi_F + \dot{\psi_F} \cos\theta_F \cos\phi_F
\end{aligned}
\tag{2.25}
$$

The moment equilibrium equations can be expressed as shown in Eqn. 2.26.

$$
\begin{aligned}
L_F &= I_{xx}\dot{p_F} - I_{xy}(\dot{q_F} - p_F r_F) - I_{xz}(\dot{r_F} - p_F q_F) - I_{yz}(q_F^2 - r_F^2) - (I_{yy} - I_{zz}) q_F r_F \\
M_F &= I_{yy}\dot{q_F} - I_{yz}(\dot{r_F} - q_F p_F) - I_{yx}(\dot{p_F} - q_F r_F) - I_{zx}(r_F^2 - p_F^2) - (I_{zz} - I_{xx}) r_F p_F \\
N_F &= I_{zz}\dot{r_F} - I_{zx}(\dot{p_F} - r_F q_F) - I_{zy}(\dot{q_F} - r_F p_F) - I_{xy}(p_F^2 - q_F^2) - (I_{xx} - I_{yy}) p_F q_F
\end{aligned}
\tag{2.26}
$$

Also, the position vector of the fuselage reference point with respect to the center of gravity is given in Eqn. 2.27.

$$
r_{ref} \quad = \quad x_{ref} i_B \quad + \quad y_{ref} j_B \quad + \quad z_{ref} k_B \tag{2.27}
$$

The aerodynamic loads acting on the body of the fuselage are computed based on the flow velocity components at the reference point on the fuselage as given in Eqn. 2.28.

$$
\begin{aligned}
u_{ref} &= u_F + y_{ref} r_F - z_{ref} q_F + u_{in,F} \\
v_{ref} &= v_F + z_{ref} p_F - x_{ref} r_F + v_{in,F} \\
w_{ref} &= w_F + x_{ref} q_F - y_{ref} r_F + w_{in,F}
\end{aligned}
\tag{2.28}
$$

$u_{in,F}$, $v_{in,F}$, and $w_{in,F}$ are interference velocity components along body axes, which are computed from the average main rotor downwash $\lambda_o \Omega_{MR} R$, the nose-down tilt of the rotor tip path plane $\beta_{1c}$, and wake skew angle $\chi$ as shown in Eqn. 2.29.

$$
\begin{aligned}
u_{in,F} &= \lambda_o \Omega_{MR} R v_x(\beta_{1c}, \chi) \\
v_{in,F} &= 0 \\
w_{in,F} &= \lambda_o \Omega_{MR} R v_z(\beta_{1c}, \chi)
\end{aligned}
\tag{2.29}
$$

The functions $v_x(\beta_{1c}, \chi)$ and $v_z(\beta_{1c}, \chi)$ are obtained from look-up tables, and the wake skew angle is obtained from the free-stream velocity components along shaft axes $(u_s, v_s, w_s)$ as shown in Eqn. (2.30).

$$\begin{aligned} \chi &= \tan^{-1}\frac{u_s}{\lambda\Omega_{MR}R - w_s} + \beta_{1c} \\ \alpha_F &= \tan^{-1}\frac{w_F}{u_F} \\ \beta_F &= \tan^{-1}\frac{v_F}{\sqrt{u_F^2 + w_F^2}} \end{aligned} \tag{2.30}$$

The longitudinal flow incidence angle $\alpha_F$ is positive when the fuselage is tilted nose-up with respect to the free-stream flow, and the lateral flow incidence angle $\beta_F$ is positive when the starboard side is facing the free-stream flow. Using these two flow angles and the dynamic pressure at the fuselage reference point $q_F$, the aerodynamic coefficients in the wind-axes system are obtained using a table look-up procedure based on wind-tunnel measurements. In addition, to precisely predict the fuselage drag force, the function derived from the wind-tunnel testing is adopted. The fuselage forces and moments with respect to the center of gravity are given in Eqn. 2.31.

$$\begin{aligned} \begin{bmatrix} X_F \\ Y_F \\ Z_F \end{bmatrix}_F &= q_F \begin{bmatrix} f(\alpha_F) \\ C_Y \\ C_Z \end{bmatrix}_F \\ \begin{bmatrix} L_F \\ M_F \\ N_F \end{bmatrix}_F &= q_F \begin{bmatrix} C_L \\ C_M \\ C_N \end{bmatrix}_F + r_{ref} \times \begin{bmatrix} X_F \\ Y_F \\ Z_F \end{bmatrix}_F \\ q_F &= \frac{1}{2}\,\rho\,(u_F^2 + v_F^2 + w_F^2) \end{aligned} \tag{2.31}$$

Fuselage drag force is the dominant source of the fuselage aerodynamic force. Instead of using constant aerodynamic coefficients, it adopts the function that predicts the drag using the flat-plate area and fuselage angle of attack. To predict the fuselage drag force precisely, several different estimations were tested and the model finally selected for the present study is the estimation by

Yeo et al. [74] and given in Eqn. 2.32, where 35.14 $ft^2$ is the flat-plate area of the UH-60 helicopter fuselage at zero angles of attack. The fuselage drag force is computed by multiplying the flat-plate area with dynamic pressure.

$$f(\alpha_F) \quad = \quad 35.14 \quad + \quad 0.016(1.66\alpha_F)^2 \tag{2.32}$$

### 2.3.2 Main Rotor

This section covers the description of the mathematical model of the main rotor system. The main rotor blades are individually analyzed. The blades are assumed to be rigid. Forces and moments from each blade are translated to the body-fixed frame. Each blade experiences flap and lead-lag motions which are angles $\beta$ and $\zeta$, respectively. The flap and lead-lag hinge with its own spring and damper are placed at the same location. The blade equations of motion are nonlinear, coupled, partial differential equations with periodic coefficients. The inertial and aerodynamic load vectors are calculated numerically with the assumption of the first harmonic blade motion. Each blade is discretized to 100 finite elements and travels with $1^\circ$ step displacement in azimuth. In addition, the negative twist rate of $18^\circ$ is considered in the computation of the elemental angle of attack. The summation of the loads on each blade element is collected and transformed to the body-fixed frame.

#### *2.3.2.1 Inertial Loads*

In the formulation of the main rotor equations of motion, the distributed loads due to blade inertia are required. These inertia loads depend on the absolute acceleration of a point on the rotor blade, $A_P$. Taking into account the helicopter hub accelerations, an arbitrary point "P" on a blade is expressed in mixed coordinate systems as given in Eqn. 2.33.

$$R_P \quad = \quad R_{cg} \quad + \quad R_{hub} \quad + \quad R_b \tag{2.33}$$

$R_{cg}$ represents the position of the CG from the earth-fixed frame origin. $R_{hub}$ represents the position of the hub from the CG. $R_b$ represents the position of an arbitrary point on a blade with

consideration for the hinge offset. They can be expressed in the body-fixed frame as shown in Eqn. 2.34.

$$\begin{aligned} R_{cg,B} &= T_{BG}\ R_{cg,G} \\ R_{hub,B} &= T_{BN}\ R_{hub,NR} \\ R_{b,B} &= T_{BL}\ R_{b,L} \\ &= T_{BN}\ T_{NR}\ T_{RF}\ T_{FL}\ R_{b,L} \\ R_{P,B} &= R_{cg,B} + R_{hub,B} + R_{b,B} \end{aligned} \tag{2.34}$$

The velocity and acceleration of a generic point of the blade in the body-fixed frame need to be calculated. Using transport theorem, the velocity $V_{P,B}$ of the point P is computed as shown in Eqn. 2.35.

$$\begin{aligned} V_{P,B} &= \frac{dR_{P,B}}{dt} = \frac{\partial R_P}{\partial t} + W \times R_P \\ &= \frac{\partial R_{cg,B}}{\partial t} + \frac{\partial R_{b,L}}{\partial t} + (W_{B\leftarrow NR} + W_{NR\leftarrow R} + W_{R\leftarrow F} + W_{F\leftarrow L}) \times R_{b,L} \end{aligned} \tag{2.35}$$

$\frac{\partial R_{cg,B}}{\partial t}$ and $\frac{\partial R_{b,L}}{\partial t}$ are the velocity of the CG in the body-fixed frame and the velocity of the point P on a blade in the blade lagged frame, respectively. The time derivative of the hub position in the body-fixed frame is zero since it is constant with respect to time. The other term $W \times R_{b,L}$ is taking into account the transformation of coordinate systems and the fact that the blade is rotating. The subscripts of $W$ describe the transformation from one frame to another frame. In order to show the transformation sequence from the lagged frame to the body-fixed frame, $W$ is decomposed to each step. In the same manner, the acceleration $A_{P,B}$ of the point P is calculated as shown in Eqn. 2.36.

$$\begin{aligned} A_{P,B} &= \frac{dV_{P,B}}{dt} = \frac{\partial V_P}{\partial t} + W \times V_P \\ &= \frac{\partial V_{cg,B}}{\partial t} + \frac{\partial V_{b,L}}{\partial t} + (W_{B\leftarrow NR} + W_{NR\leftarrow R} + W_{R\leftarrow F} + W_{F\leftarrow L}) \times V_{b,L} \end{aligned} \tag{2.36}$$

The total inertial force in the body-fixed frame is obtained by integrating along the blade and along the azimuth as described in Eqn. 2.37.

$$F_{I,B} \quad = \quad \frac{Nb}{2\pi} \int_0^{2\pi} \int_{eR}^{R} \; m_b \;\; A_{P,B} \;\; dr d\psi \tag{2.37}$$

$N_b$ is the number of blades, $m_b$ is the blade mass per unit length, and $eR$ is the hinge offset distance from the hub. By adding a hinge, the sum of applied moments about the hinge has to be zero. First, inertial moment about the hinge for a blade $M_I$ is calculated as shown in Eqn. 2.38.

$$M_I \quad = \quad \int_{eR}^{R} \; m_b \;\; (R_{b,L} \quad \times \quad A_{P,L}) \;\; dr \tag{2.38}$$

*2.3.2.2 Aerodynamic Loads*

A key ingredient for the calculation of the aerodynamic forces and moments is the absolute velocity of a point on the blade in the blade lagged frame. The total velocity $V_T$ of the point on a blade is expressed as Eqn. 2.39.

$$V_T = V_P - V_I \tag{2.39}$$

$V_I$ is the induced velocity at the point on the blade by the rotor wake. In the blade flapped frame, the components of the velocity $V_P$ can be written in the form as shown in Eqn. 2.40.

$$V_P = V_{i,F} \;\; i_F + V_{j,F} \;\; j_F + V_{k,F} \;\; k_F \tag{2.40}$$

$i_F$, $j_F$, and $k_F$ are the unit vectors of the blade flapped frame and $V_{x,F}$, $V_{y,F}$, and $V_{z,F}$ are the velocity components along the blade flapped axes.

$$V_I = \lambda_i \;\; i_F + \lambda_j \;\; j_F + \lambda_k \;\; k_F \tag{2.41}$$

$\lambda_i$, $\lambda_j$, and $\lambda_k$ are the i, j, and k components of the induced velocity in the blade-flapped frame.

The dynamic inflow model takes only the k component of the induced velocity. Thus, the total velocity is expressed in Eqn. 2.42.

$$\begin{aligned} V_T &= V_{i,F}\ \ i_F + V_{j,F}\ \ j_F + (V_{k,F} - \lambda_k)\ \ k_F \\ &= V_{x,F}\ \ i_F + V_{y,F}\ \ j_F + V_{z,F}\ \ k_F \end{aligned} \tag{2.42}$$

$V_x$, $V_y$, and $V_z$ are the velocity components in the blade-flapped frame. Using the coordinate transformation that takes into account the lead-lag angles, this total velocity can be expressed in terms of the airflow velocity components in the blade-lagged frame as shown in Eqn. 2.43.

$$V_T = U_R\ \ i_L + U_T\ \ j_L + U_P\ \ k_L \tag{2.43}$$

As shown in Fig. 2.8, $V_T$ is the resultant velocity of the airflow at the quarter-chord location. The $U_T$ and $U_R$ components follow the sign conventions by which $U_T$ is positive for an airflow coming toward the leading edge of the airfoil and $U_R$ is positive for an outboard flow. The component $U_P$ is defined as positive for a flow going downward.

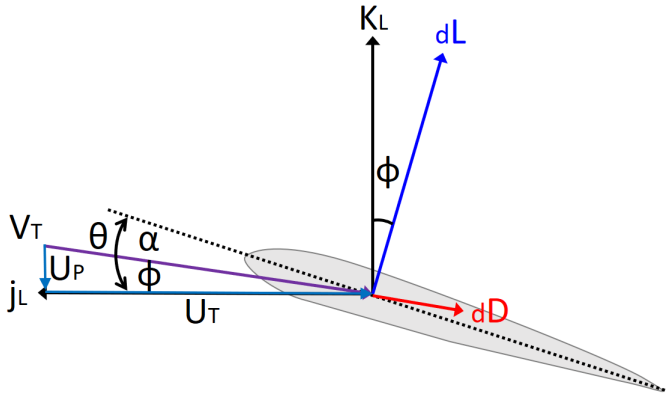



Figure 2.8: Blade elemental lift and drag

The body-fixed linear velocity components which are u, v, and w along the $i_B$, $j_B$, and $k_B$ can be related to the fuselage angles and the total velocity $V_T$ as expressed in Eqn. 2.44.

$$
\begin{aligned}
u &= V_T \ \cos\alpha_F \ \cos\beta_F \\
v &= V_T \ \sin\beta_F \\
w &= V_T \ \sin\alpha_F \ \cos\beta_F
\end{aligned}
\tag{2.44}
$$

The elemental lift dL, drag dD, and force along the blade-lagged frame are expressed in Eqn. 2.45.

$$
\begin{aligned}
dL &= \frac{1}{2}\ \rho\ V_T^2\ c_l\ c\ dr \\
dD &= \frac{1}{2}\ \rho\ V_T^2\ c_d\ c\ dr \\
dF_L &= (dL\cos\phi - dDsin_\phi)\ k_L \ - \ (dL\sin\phi + dD\cos\phi)\ j_L
\end{aligned}
\tag{2.45}
$$

$\rho$ is the air density, *c* is the chord length, and *dr* is the blade element of span. By assuming linear incompressible aerodynamics, $c_l$ can be a function of the lift slope and angle of attack($c_l$ = *a* $\alpha$). *a* is the lift slope and $\alpha$ is the angle of attack. $c_d$ is assumed constant. These are reasonable under the linear range of $c_l$ and $c_d$ with respect to the angle of attack. The aerodynamic angle of the cross section $\alpha$ is given in Eqn. 2.46.

$$
\alpha \ = \ \theta \ - \ \phi \ = \ \theta \ - \ \tan^{-1}\frac{U_P}{U_T}
\tag{2.46}
$$

The local geometric pitch at each radial station $\theta$ is defined as shown in Eqn. 2.47.

$$
\theta \ = \ \theta_o \ + \ \theta_{1c}\cos\psi_n \ + \ \theta_{1s}\sin\psi_n \ + \ \theta_{tw}
\tag{2.47}
$$

$\theta_o$ is the collective pitch, $\theta_{1c}$ is the lateral cyclic, $\theta_{1s}$ is the longitudinal cyclic, and $\theta_{tw}$ is the the built-in twist. Thus, *dL* can be re-written as Eqn. 2.48.

$$
\begin{aligned}
dL &= \frac{1}{2}\ \rho\ V_T^2\ a\ \alpha\ c\ dr \\
&= \frac{1}{2}\ \rho\ U_T^2\ a\ (\theta\ -\ \frac{U_P}{U_T})\ c\ dr \\
&= \frac{1}{2}\ \rho\ a\ (\theta\ U_T^2\ -\ U_P\ U_T)\ c\ dr
\end{aligned}
\tag{2.48}
$$

Integrating over the blade span and along the azimuth yields the total aerodynamic forces as shown in Eqn. 2.49.

$$
\begin{aligned}
F_{A,L} &= \frac{Nb}{2\pi}\int_0^{2\pi}\int_{eR}^{R}\ dF_{P,L}\ \ drd\psi \\
F_{A,B} &= T_{BL}\ \ F_{A,L}
\end{aligned}
\tag{2.49}
$$

Aerodynamic flap moment $M_A$ about the hinge is computed as Eqn. 2.50.

$$
\begin{aligned}
M_A &= \int_{eR}^{R} r\ \times\ dL\ dr \\
&= \frac{1}{2}\ \rho\ a\ \int_{eR}^{R}(\theta\ U_T^2\ -\ U_P\ U_T)\ c\ dr \\
&= \frac{\gamma}{2}\ \Omega^2\ I_b\ \int_{eR}^{R}\ \Big[\theta\ (\frac{U_T}{\Omega R})^2\ -\ (\frac{U_P}{\Omega R})\ (\frac{U_T}{\Omega R})\Big]\ dr
\end{aligned}
\tag{2.50}
$$

The lock number $\gamma$, the mass flapping moment of inertia $I_b$ are shown in Eqn. 2.51.

$$
\begin{aligned}
\gamma &= \frac{\rho\ a\ c\ R^4}{I_b} \\
I_b &= \int_{eR}^{R} m_b\ (r\ -\ eR)^2\ dr
\end{aligned}
\tag{2.51}
$$

Since $M_A$ has the blade tip speed term in common, advance ratio $\mu$ and inflow ratio $\lambda$ with respect to the blade tip speed are defined as Eqn. 2.52.

$$
\begin{aligned}
\mu &= \frac{V_T\ \cos\alpha}{\Omega R} \\
\lambda &= \frac{v}{\Omega R} \\
&= \frac{v_i\ -\ V_T\ \sin\alpha_F\ \cos\beta_F}{\Omega R} \\
&= \frac{v_i\ -\ w}{\Omega R}
\end{aligned}
\tag{2.52}
$$

$v$ is the inflow velocity and $v_i$ is the induced velocity which is normal to TPP.

#### *2.3.2.3 Hub Loads*

The forces and moments transmitted to the hub are obtained by integrating the loads along the span and summing the contributions from each of the blades. The force components along the blade rotating frame from the $n^{th}$ blade are given in Eqn. 2.53.

$$\begin{aligned} X_{MR,R}(n) &= \int_{eR}^{R} dX_{MR,R} dr \\ Y_{MR,R}(n) &= \int_{eR}^{R} dY_{MR,R} dr \\ Z_{MR,R}(n) &= \int_{eR}^{R} dZ_{MR,R} dr \end{aligned} \tag{2.53}$$

$dX_{MR,R}$, $dY_{MR,R}$, and $dZ_{MR,R}$ represent the moment components per unit span along the blade rotating frame which contain the sum of inertial and aerodynamic loads. The hub loads are obtained by resolving the blade loads along the blade non-rotating frame and summing the contributions from individual blades. The hub force and moment components are expressed in Eqn. 2.54.

$$\begin{aligned} \begin{bmatrix} X_{MR} \\ Y_{MR} \\ Z_{MR} \end{bmatrix}_{NR} &= \sum_{n=1}^{Nb} T_{NL} \begin{bmatrix} X_{MR}(n) \\ Y_{MR}(n) \\ Z_{MR}(n) \end{bmatrix}_{L} \\ \begin{bmatrix} L_{MR} \\ M_{MR} \\ N_{MR} \end{bmatrix}_{NR} &= \sum_{n=1}^{Nb} T_{NL} \begin{bmatrix} L_{MR}(n) \\ M_{MR}(n) \\ N_{MR}(n) \end{bmatrix}_{L} \end{aligned} \tag{2.54}$$

The hub loads are converted to the helicopter body-fixed frame using the transformation matrix $T_{BN}$ which are the main rotor contributions to the helicopter force and moment equilibrium as shown in Eqn. 2.55.

$$\begin{bmatrix} X_{MR} \\ Y_{MR} \\ Z_{MR} \end{bmatrix}_B = T_{BN} \begin{bmatrix} X_{MR} \\ Y_{MR} \\ Z_{MR} \end{bmatrix}_{NR}$$
$$\begin{bmatrix} L_{MR} \\ M_{MR} \\ N_{MR} \end{bmatrix}_B = T_{BN} \begin{bmatrix} L_{MR} \\ M_{MR} \\ N_{MR} \end{bmatrix}_{NR} + \begin{bmatrix} y_{hub}Z_{MR,B} - z_{hub}Y_{MR,B} \\ z_{hub}X_{MR,B} - x_{hub}Z_{MR,B} \\ x_{hub}Y_{MR,B} - y_{hub}X_{MR,B} \end{bmatrix} \tag{2.55}$$

#### *2.3.2.4 Blade Flapping Dynamics*

The equilibrium position of the blade is determined by the balance of inertial, aerodynamic, and centrifugal forces (CF). The flapping angle can be assumed small due to the fact that the centrifugal force is larger than the aerodynamic force. Moment equilibrium about the flapping hinge is expressed in Eqn. 2.56.

$$\begin{aligned} M_{I,F} \quad + \quad M_{CF,F} \quad + \quad M_{A,F} \quad &= \quad 0 \\ \int_{eR}^{R} m_b(y-eR)^2\ddot{\beta}dy \; + \; \int_{eR}^{R} m_b\Omega^2 y(y-eR)\beta dy \; - \; \int_{eR}^{R} L(y-eR)dy \quad &= \quad 0 \end{aligned} \tag{2.56}$$

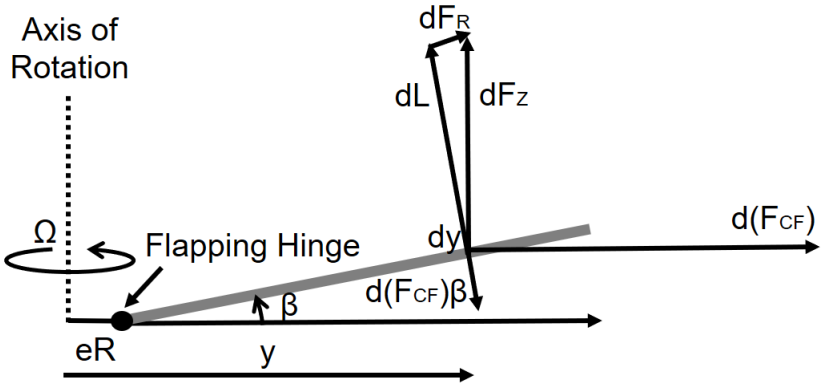



Figure 2.9: Geometry of blade flapping equilibrium

Integrating along the blade and dividing by $\Omega^2$ yields the relationship in Eqn. 2.57.

$$\begin{aligned} I_b(\overset{**}{\beta} + \nu_\beta^2\beta) &= \frac{1}{\Omega^2}\int_{eR}^{R} L(y - eR)dy \\ \nu_\beta^2 &= 1 + \frac{eR\int_{eR}^{R} m_b(y - eR)dy}{I_b} \\ \nu_\beta &= \omega_n = \sqrt{1 + \frac{3e}{2(1-e)}} \end{aligned} \tag{2.57}$$

$\nu_\beta$ is the non-dimensional flapping frequency in terms of the rotational speed and it can also be considered as the undamped natural frequency in a spring-mass-damper system. Hence, the flapping equation can be expressed as Eqn. 2.58.

$$\overset{**}{\beta} + \nu_\beta^2\beta = \gamma M_{A,F} \tag{2.58}$$

In general, the value of e varies from 4 to 6% for an articulated blade, so that the natural frequency of the rotor is slightly greater than $\Omega$ or 1/rev. This also means that the phase lag between the forcing and the rotor flapping response has to be less than 90° and the flapping displacements now also depend on aerodynamic damping. After expanding all terms in the aerodynamic flapping moment $M_A$, it can be organized as Eqn. 2.59.

$$M_{A,F} = \frac{1}{2}\rho ac\Omega^2R^4(M_\theta\theta + M_\lambda\lambda + M_{\overset{*}{\beta}}\overset{*}{\beta} + M_\beta\beta + M_p\frac{p}{\Omega} + M_q\frac{q}{\Omega}) \tag{2.59}$$

The flap damping term is the coefficient $M_{\overset{*}{\beta}}$ and it is associated with lock number. Investigating hover case, the damping is approximately 50% which means the blade flapping motion is stable and well damped. Assuming the blade flapping motion is the first harmonic, it can be expressed as a function of the blade azimuth $\psi_n$ which is shown in Eqn. 2.60.

$$\beta_{\psi_n} = \beta_o + \beta_{1c}\cos\psi_n + \beta_{1s}\sin\psi_n \tag{2.60}$$

By replacing flapping terms with constant and periodic terms on both sides of the flapping equation flapping angles can be related to control angles ($\theta_o$, $\theta_{1c}$, $\theta_{1s}$).

#### 2.3.2.5 *Blade Lagging Dynamics*

The equilibrium of the blade about the lead-lag hinge is determined by a balance of centrifugal and aerodynamic moments. The aerodynamic moments are generated by the aerodynamic drag of the blade as it rotates. Moment equilibrium about the lead-lag hinge is expressed as Eqn. 2.61.

$$\begin{aligned} M_{I,L} \quad &+ \quad M_{CF,L} \quad + \quad M_{A,L} \quad = \quad 0 \\ -\int_{eR}^{R} m_b(y-eR)^2\ddot{\zeta}dy \; &+ \; \int_{eR}^{R} m_b\Omega^2 y(y-eR)\frac{eR}{y}\zeta dy \; + \; \int_{eR}^{R} D(y-eR)dy \; = \; 0 \end{aligned} \tag{2.61}$$

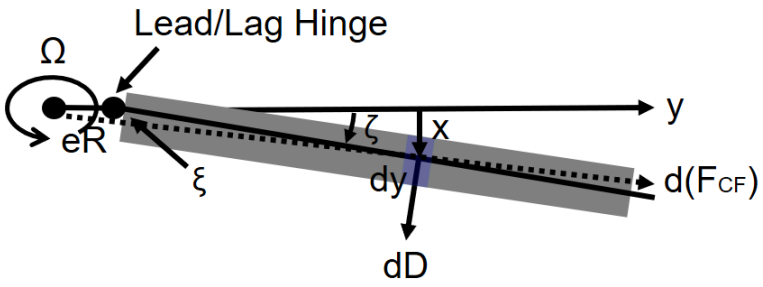



Figure 2.10: Geometry of blade lagging equilibrium

Integrating along the blade and dividing by $\Omega^2$ yields the relationship as Eqn. 2.62.

$$\begin{aligned} I_\zeta(\overset{**}{\zeta} \; + \; \nu_\zeta^2\zeta) \; &= \; \frac{1}{\Omega^2}\int_{eR}^{R} D(y \; - \; eR)dy \\ \nu_\zeta^2 \; &= \; \frac{eR\int_{eR}^{R} m_b(y \; - \; eR)dy}{I_\zeta} \\ \nu_\zeta \; &= \; \sqrt{\frac{3}{2}(\frac{eR}{R \; - \; eR})} \\ (\; I_\zeta \; &= \; \int_{eR}^{R} m_b(y \; - \; eR)^2 dy \;) \end{aligned} \tag{2.62}$$

$I_\zeta$ is the mass moment of inertia about the lead-lag hinge, $\nu_\zeta$ is the non-dimensional lag frequency. The centrifugal restoring moment about the lag hinge is much smaller than in flapping,

the corresponding uncoupled natural frequency of the lag motion is much smaller. For articulated rotors such as UH-60 rotors, the uncoupled rotating lag frequency varies from about 0.2 to 0.3$\Omega$. The lead-lag displacements about the hinge are small and aerodynamic forces are produced by changes in velocity and dynamic pressure normal to the blade's leading edge. However, it is much smaller than the aerodynamic forces which are produced through flapping motion by changes in the angle of attack. Furthermore, the drag forces acting on the blades are also much smaller than the lift forces.

#### *2.3.2.6 Inflow Dynamics*

In this study, a linear inflow distribution is assumed, thus it is expressed with respect to the blade azimuth $\psi_n$ as shown in Eqn. 2.63.

$$\lambda \quad = \quad \lambda_o \; + \; \lambda_{1c}\frac{r}{R}\cos\psi_n \; + \; \lambda_{1s}\frac{r}{R}\sin\psi_n \tag{2.63}$$

Basically, modeling a linear inflow distribution is to estimate the values of $\lambda_o$, $\lambda_{1c}$, and $\lambda_{1s}$. For this mathematical UH-60 helicopter model, the Pitt-Peters linear inflow model is used.

$$[M]\begin{bmatrix}\dot{\lambda}_o\\ \dot{\lambda}_{1c}\\ \dot{\lambda}_{1s}\end{bmatrix} \; + \; [L]^{-1}\begin{bmatrix}\lambda_o\\ \lambda_{1c}\\ \lambda_{1s}\end{bmatrix} \quad = \quad \begin{bmatrix}C_T\\ -C_{M_y}\\ C_{M_x}\end{bmatrix} \tag{2.64}$$

These dynamic inflow components are related to the forces on the rotor disk which are the rotor thrust coefficient $C_T$, pitching moment coefficient $C_{M_y}$, and rolling moment coefficient $C_{M_x}$. The matrix size of M and L is 3 by 3 which are given in Eqn. 2.65.

$$
\begin{aligned}
& = \frac{1}{C_V}\begin{bmatrix} \frac{1}{2} & 0 & \frac{15\pi}{64}\sqrt{\frac{1-\sin\alpha}{1+\sin\alpha}} \\ 0 & \frac{-4}{1+\sin\alpha} & 0 \\ \frac{15\pi}{64}\sqrt{\frac{1-\sin\alpha}{1+\sin\alpha}} & 0 & \frac{4}{1+\sin\alpha} \end{bmatrix} \\
[M] \quad & = \begin{bmatrix} \frac{128}{75\pi} & 0 & 0 \\ 0 & -\frac{16}{45\pi} & 0 \\ 0 & 0 & -\frac{16}{45\pi} \end{bmatrix}
\end{aligned} \tag{2.65}
$$

$\alpha$ is the rotor disk tilt angle with respect to the free stream and $C_V$ is the mass-flow parameter which can be computed as Eqn. 2.66.

$$
C_V \quad = \quad \frac{\mu^2 + \lambda(\lambda + \lambda_i)}{\sqrt{\mu^2 + \lambda^2}} \tag{2.66}
$$

For dynamic analysis of the blade, the dynamic inflow components are treated as additional degrees of freedom. It is formulated on the basis of experimental results or more advanced vortex theories and it is well suited for helicopter rotor aerodynamics, aeroelasticity, and flight dynamics.

### 2.3.3 Tail Rotor

The tail rotor model is based on a simplified implementation of the closed-form solution given by F. J. Bailey [75], which relates the free-stream velocity to the rotor thrust, torque, and induced inflow. The velocity at the tail rotor reference point (hub) is given in Eqn. 2.67.

$$
V_{TR} \quad = \quad V_b \quad + \quad \omega \times r_{TR} \quad + \quad V_{in,TR} \tag{2.67}
$$

$V_{in,TR}$ represents the induced velocity at the tail rotor reference point due by the wake of the main rotor and fuselage as given in Eqn. 2.68.

$$
V_{in,TR} \quad = \quad \lambda_o \Omega_{MR} R \Big[ v_{x_{TR}}(\beta_{1c}, \chi) i_B \quad + \quad v_{z_{TR}}(\beta_{1c}, \chi) k_B \Big] \tag{2.68}
$$

The functions $v_{x,TR}$, $v_{z,TR}$ are obtained from the lookup tables based on the wake skew angle

$\chi$ and the tip-path plane tilt $\beta_{1c}$ with respect to the fuselage. The velocity $V_{TR}$ at the tail rotor reference point $r_{TR}$ is resolved into components along the tail rotor axes. The tail rotor axes system is obtained using two rotations in the sequence Z $\rightarrow$ Y through angles ($\Gamma_{TR}$, $\Lambda_{TR}$) starting from the helicopter body axes. The rotation matrix from fuselage body axes to tail rotor axes is given in Eqn. 2.69.

$$T_{TR,B} = \begin{bmatrix} \cos\Lambda_{TR} & 0 & -\sin\Lambda_{TR} \\ 0 & 1 & 0 \\ \sin\Lambda_{TR} & 0 & \cos\Lambda_{TR} \end{bmatrix} \begin{bmatrix} \cos\Gamma_{TR} & \sin\Gamma_{TR} & 0 \\ -\sin\Gamma_{TR} & \cos\Gamma_{TR} & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.69}$$

The velocity components in the tail rotor reference frame are given in Eqn. 2.70.

$$\begin{bmatrix} u \\ v \\ w \end{bmatrix}_{TR} = T_{TR,B}\ V_{TR} \cdot \begin{bmatrix} i_B \\ j_B \\ k_B \end{bmatrix} \tag{2.70}$$

The tail rotor thrust which is assumed to act along the shaft direction is given in Eqn. 2.71.

$$\begin{aligned} T_{TR} &= \pi\ R_{TR}^4\ \Omega_{TR}^2\ |V|_{TR}\ v_{i,TR}\ K_{TR} \\ v_{i,TR} &= \lambda_{TR}\Omega_{TR}R_{TR} \\ |V|_{TR} &= \sqrt{u_{TR}^2\ +\ v_{TR}^2\ +\ (w_{TR}^2\ -\ \lambda_{TR}\Omega_{TR}R_{TR})^2} \end{aligned} \tag{2.71}$$

$v_{i,TR}$ is the average induced velocity of the tail rotor, $K_{TR}$ accounts for blockage effects of the vertical fin, and $|V|_{TR}$ is the magnitude of the total velocity (including induced inflow) at the tail rotor. The tail rotor torque due to induced and profile drag is given in Eqn. 2.72.

$$Q_{TR} = C_{Q_{TR}}\ \rho\ \pi\ \Omega_{TR}^2\ R_{TR}^5 \tag{2.72}$$

The force and moment components in fuselage body axes exerted by the tail rotor on the airframe center of gravity are obtained using a coordinate transformation as shown in Eqn. 2.73.

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix}_{TR} = T^T_{TR,B} \begin{bmatrix} 0 \\ -T_{TR} \\ 0 \end{bmatrix}$$
$$\begin{bmatrix} L \\ M \\ N \end{bmatrix}_{TR} = T^T_{TR,B} \begin{bmatrix} 0 \\ -Q_{TR} \\ 0 \end{bmatrix} + r_{TR} \times (X_{TR} i_B + Y_{TR} j_B + Z_{TR} k_B) \tag{2.73}$$

The induced inflow of the tail rotor is assumed to be uniform over the disk and is represented using a 1-state Pitt-Peters dynamic inflow model. The ODE governing the inflow dynamics is given in Eqn. 2.74.

$$\frac{4R_{TR}}{2\pi|V_{TR}|}\dot{\lambda}_{TR} + \lambda_{TR} = \frac{C_{T_{TR}}\Omega_{TR}R_{TR}}{2|V_{TR}|} \tag{2.74}$$

### 2.3.4 Horizontal and Vertical Tail

The aerodynamic loads acting on the horizontal stabilator and vertical fin are computed using a procedure similar to that followed for the fuselage. The velocity at the reference point for each lifting surface is computed from the fuselage translation velocity $V_b$, angular velocity $w_b$ and the position of the reference points with respect to the vehicle center of gravity $r_H$, $r_V$ as given in Eqn. 2.75.

$$\begin{aligned} V_H &= K_H V_b + \omega \times r_H + V_{in,H} \\ V_V &= K_V V_b + \omega \times r_V + V_{in,V} \end{aligned} \tag{2.75}$$

$K_H$ and $K_H$ are used to empirically model the dynamic pressure loss at the tail surfaces, which occurs as a result of operating in the wake of the airframe. $V_{in,H}$ and $V_{in,V}$ represent the velocities at the tail surfaces induced by the main rotor wake, obtained from wind-tunnel tests as given in Eqn. 2.76.

$$\begin{aligned} V_{in,H} &= \lambda_o \Omega_{MR} R\Big[v_{x_H}(\beta_{1c},\chi) i_B + v_{z_H}(\beta_{1c},\chi) k_B\Big] \\ V_{in,V} &= \lambda_o \Omega_{MR} R\Big[v_{x_V}(\beta_{1c},\chi) i_B + v_{z_V}(\beta_{1c},\chi) k_B\Big] \end{aligned} \tag{2.76}$$

The functions $v_{x_H}$, $v_{z_H}$, $v_{x_V}$, $v_{z_V}$ are obtained from look-up tables based on the wake skew angle $\chi$ and the tip-path plane tilt $\beta_{1c}$ with respect to the fuselage. Using $(u_H, v_H, w_H)$ and $(u_V, v_V, w_V)$ to represent the velocity components at the horizontal stabilator and vertical fin, respectively, along vehicle body axes, the angles of attack and sideslip at the tail surfaces are computed as given in Eqn. 2.77.

$$\begin{aligned} \alpha_H &= \tan^{-1}\frac{w_H}{u_H} + \theta_H \\ \beta_H &= \tan^{-1}\frac{v_H}{\sqrt{u_H^2 + w_H^2}} \\ \alpha_V &= \tan^{-1}\frac{w_V}{u_V} \\ \beta_V &= \tan^{-1}\frac{u_V}{\sqrt{u_V^2 + w_V^2}} \end{aligned} \tag{2.77}$$

The pitch of the horizontal tail (stabilator) $\theta_H$ is scheduled to change with the forward flight speed in a prescribed manner. An approach similar to that followed for the fuselage aerodynamics is utilized for computing the forces on the horizontal stabilator and vertical fin. Using the incidence angles $\alpha$ and $\beta$ for each surface and the dynamic pressure at the reference points, the aerodynamic lift and drag coefficients are obtained using a table look-up procedure based on wind-tunnel measurements, and transformed to the helicopter body axes. Using ($F_H$, $F_V$) and ($M_H$, $M_V$) that represent the body-axes forces and moments at the reference points, the loads at the vehicle center of gravity are given in Eqn. 2.78.

$$
\begin{aligned}
\begin{bmatrix} X \\ Y \\ Z \end{bmatrix}_{EM} &= q_H \begin{bmatrix} C_X \\ C_Y \\ C_Z \end{bmatrix}_H + q_V \begin{bmatrix} C_X \\ C_Y \\ C_Z \end{bmatrix}_V \\
&= F_H + F_V \\
\begin{bmatrix} L \\ M \\ N \end{bmatrix}_{EM} &= r_H \times F_H + r_V \times F_V
\end{aligned}
\tag{2.78}
$$

The dynamic pressures are given in Eqn. 2.79.

$$
\begin{aligned}
q_H &= \frac{1}{2} \rho \; V_H \cdot V_H \\
q_V &= \frac{1}{2} \rho \; V_V \cdot V_V
\end{aligned}
\tag{2.79}
$$

## 2.4 Trim Analysis

The term *trim* refers to a steady flight condition that the linear accelerations along the body axes and angular accelerations about the body axes are zero. The trim unknowns *X* are given in Eqn. 2.80.

$$
\begin{aligned}
X &= [\; X_C \;\; X_F \;\; X_R \;\; X_I \;]^T \\
X_C &= [\; \theta_o \;\; \theta_{1c} \;\; \theta_{1s} \;\; \theta_{TR} \;]^T \\
X_F &= [\; \alpha_F \;\; \beta_F \;\; \phi_F \;\; \theta_F \;]^T \\
X_R &= [\; \beta_o \;\; \beta_{1c} \;\; \beta_{1s} \;\; \zeta_o \;\; \zeta_{1c} \;\; \zeta_{1s} \;]^T \\
X_I &= [\; \lambda_o \;\; \lambda_{1c} \;\; \lambda_{1s} \;\; \lambda_{TR} \;]^T
\end{aligned}
\tag{2.80}
$$

$X_C$ contains control angles, $X_F$ contains fuselage angles, $X_R$ contains flap and lead-lag angles, and $X_I$ contains inflow angles. Trim unknowns which meet the trim condition are calculated under several flight conditions. The flight condition is defined by the velocity V along the trajectory, the flight path angle $\gamma$ (positive for climbing), and the rate of turn $\dot{\psi}$ (positive for a right turn).

### 2.4.1 Trim in Hover and Steady Forward Flight

Straight and level flight is a particular case in which both the flight path angle and the rate of a turn are zero. Hover is a particular case in which the velocity is also zero. First of all, the main rotor power along the forward flight speed is investigated and compared to flight test data. These power curves are obtained for two conditions, which are, a helicopter gross weight of 16,360 lbs at an altitude of 3,670 feet and a helicopter gross weight of 16,000 lbs at an altitude of 5,250 feet. The comparison of predicted (simulated) main rotor power with flight test data from hover to 160 knots is shown in Fig. 2.11.

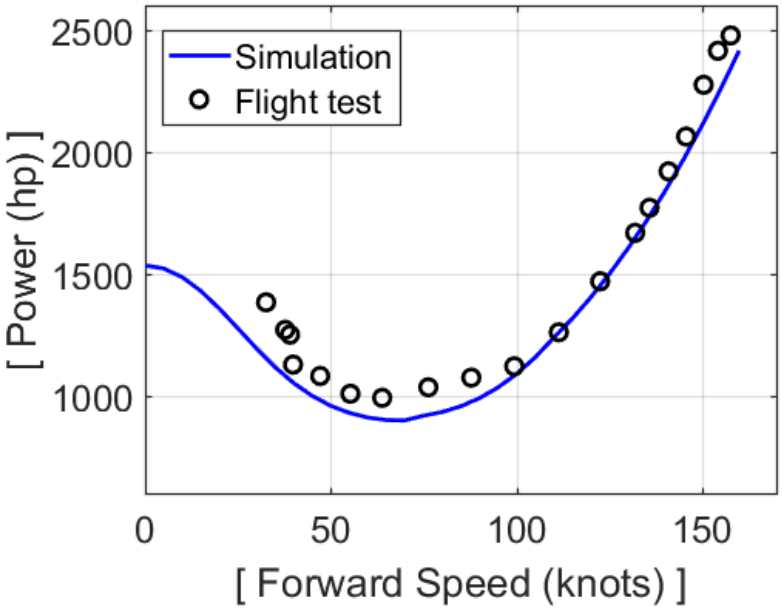



Figure 2.11: Main rotor power vs. forward flight speed (16,360 lbs at 3,670 feet)

In this comparison, the gross weight is 16,360 lbs and the density altitude is 3,670 feet. The comparisons show good agreement at speeds above 40 knots. At low speeds (below 30 knots), the simulated power curve under-predicts the power due to the linear inflow assumption. It can be enhanced by using an inflow model which captures rotor-wake interference. Typically, rotor-wake

interference is stronger at $\mu < 0.1$. Thus, it requires more power at low speeds. The additional comparison of predicted (simulated) main rotor power with flight test data from hover to 160 knots is shown in Fig. 2.12.

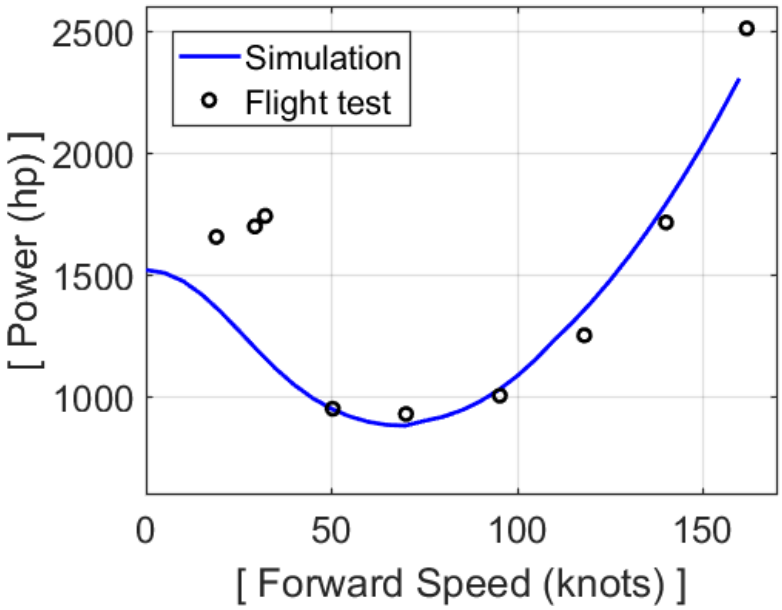



Figure 2.12: Fuselage angle vs. forward flight speed (16,000 lbs at 5,250 feet)

In this comparison, the gross weight is 16,000 lbs and the density altitude is 5,250 feet. The comparison results are similar to the previous one in Figure 3.1. It also shows good agreement at speeds above 50 knots. However, at low speeds (below 30 knots), the simulated power curve under-predicts the power due to the linear inflow assumption due to the same reason. Analyzing the power curves which are obtained from different conditions, the mathematical model with the linear inflow assumption has good agreement at higher speeds. However, predictions at lower speeds underpredict the power. Considering the results by Theodore [20] and Ribera [76], the free-vortex wake model can improve the power curve at lower speeds. The fuselage angles for forward speeds from hover to 160 knots are shown in Fig. 2.13.

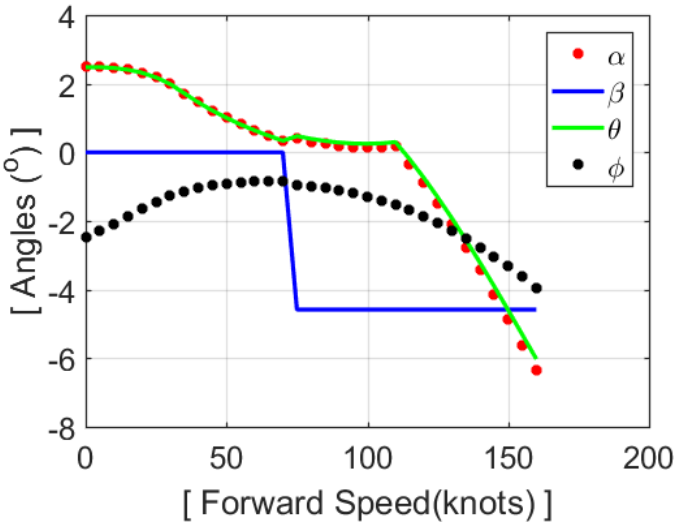



Figure 2.13: Fuselage angle vs. forward flight speed (16,000 lbs at 5,250 feet)

$\alpha$ is the flow Angle of Attack (AOA) with respect to TPP. $\beta$ is the side slip angle of the fuselage. $\theta$ is the pitch angle of the fuselage. $\phi$ is the roll angle of the fuselage. These angles describe the helicopter trim attitude at various forward flight speeds. Fuselage pitch angle and roll angle are also validated with flight test date as shown in Figs. 2.14 and 2.15.

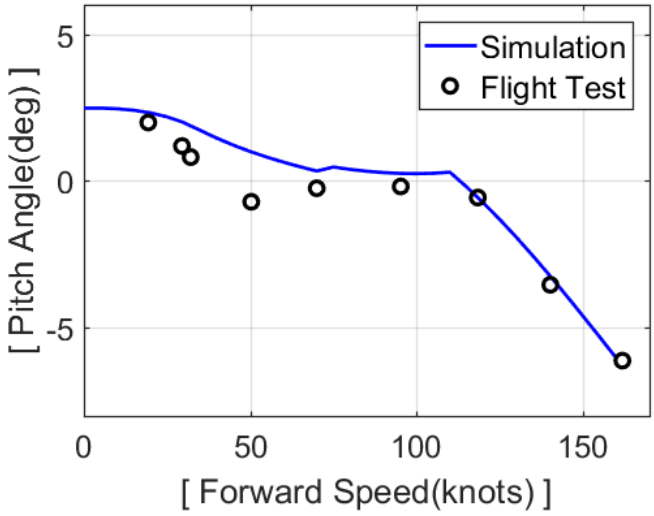



Figure 2.14: Fuselage pitch angle vs. forward flight speed (16,000 lbs at 5,250 feet)

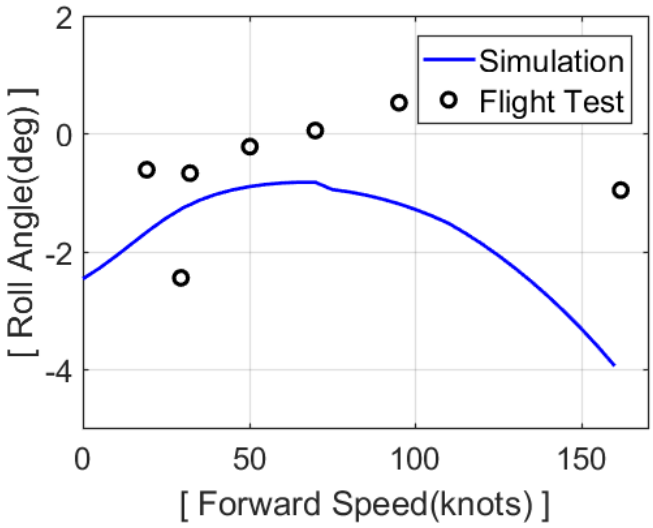



Figure 2.15: Fuselage roll angle vs. forward flight speed (16,000 lbs at 5,250 feet)

The predicted fuselage pitch angles correlate well with the flight test data in the entire forward speed range. Even though the predicted fuselage roll angle is underpredicted by a few degrees at higher forward speeds, it shows a similar trend on the whole as the flight test data. The trim values of flap angles and lead-lag angles at various forward speeds are shown in Fig. 2.16.

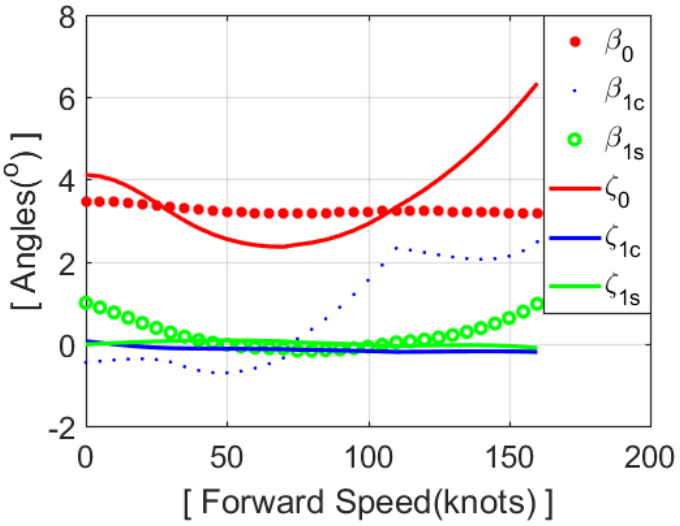



Figure 2.16: Flap and lead-lag angle vs. forward flight speed (16,000 lbs at 5,250 feet)

The angles are obtained by assuming the first harmonic (1/rev) flap and lead-lag motions. Coning angle $\beta_o$ is the mean part of the flapping motion which is independent of time or blade azimuth. $\beta_{1c}$ is the longitudinal flapping angle, which represents the amplitude of the pure cosine flapping motion. $\beta_{1s}$ is the lateral flapping angle, which represents the amplitude of the pure sine flapping motion. Lead-lag motion is also separated by $\zeta_o$, $\zeta_{1c}$, and $\zeta_{1s}$ in the same manner. $\zeta_o$ represents the mean lagging motion. $\zeta_{1c}$ is the longitudinal lagging angle, which represents the amplitude of the pure cosine lagging motion. $\beta_{1s}$ is the lateral lagging angle, which represents the amplitude of the pure sine lagging motion. The trim values of inflow ratios at various forward speeds are shown in Fig. 2.17.

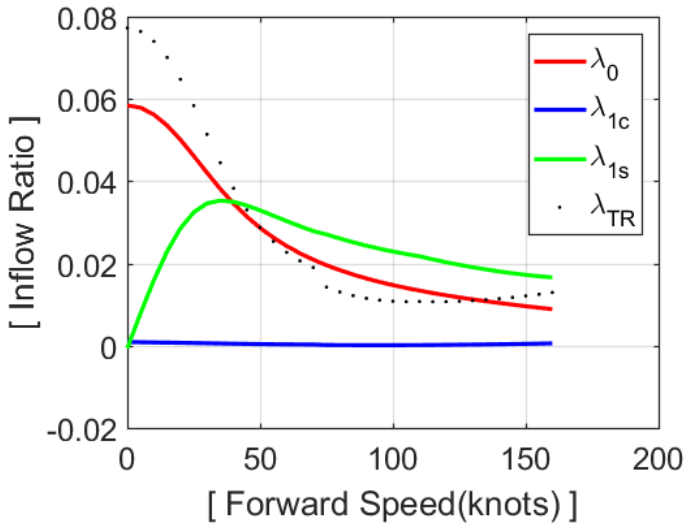



Figure 2.17: Inflow ratio vs. forward flight speed (16,000 lbs at 5,250 feet)

Dividing inflow velocities which are perpendicular to TPP by rotor tip speed, inflow ratios can be calculated. The main rotor inflow ratios are obtained by using the Pitt-peter linear inflow dynamic model. Tail rotor inflow ratio is obtained by assuming a uniform inflow due to its relatively small contribution to the entire system. $\lambda_o$ is the mean part of the main rotor inflow ratio which is independent of time or blade azimuth. $\lambda_{1c}$ is the longitudinal main rotor inflow ratio. $\lambda_{1s}$ is the lateral main rotor inflow ratio. $\lambda_{TR}$ is the tail rotor uniform inflow ratio. In addition, corresponding

control inputs are also validated with flight test data as shown in Figs. 2.18, 2.19, 2.20, and 2.21. The control inputs are closely predicted to available flight testing data in the entire flight speeds.

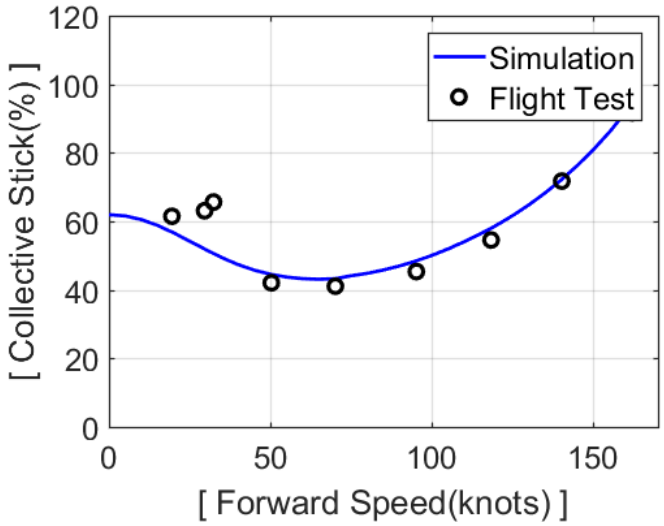



Figure 2.18: Collective stick vs. forward flight speed (16,000 lbs at 5,250 feet)

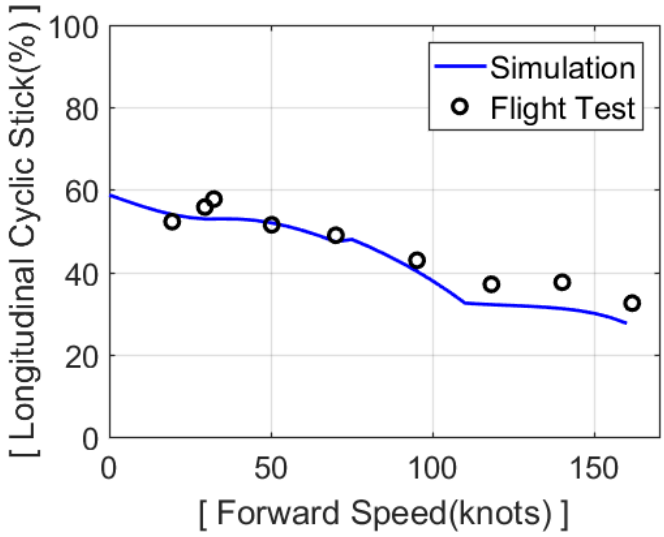



Figure 2.19: Longitudinal cyclic vs. forward flight speed (16,000 lbs at 5,250 feet)

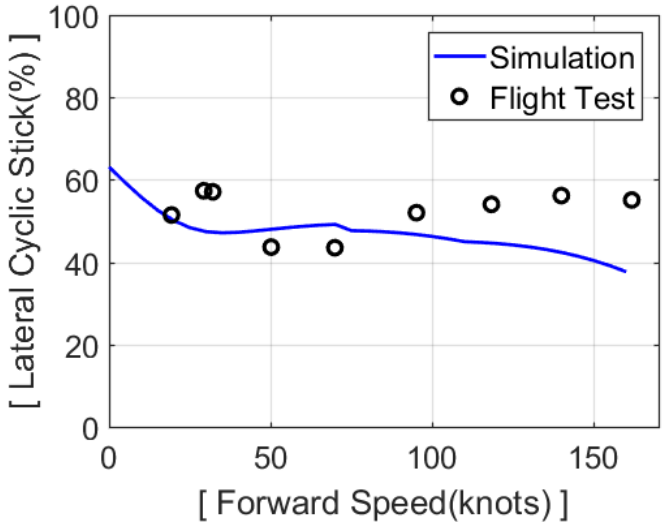



Figure 2.20: Lateral cyclic vs. forward flight speed (16,000 lbs at 5,250 feet)

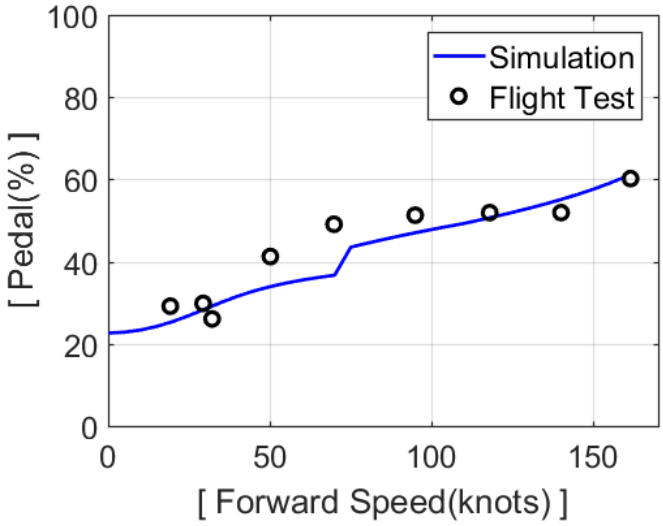



Figure 2.21: Pedal vs. forward flight speed (16,000 lbs at 5,250 feet)

### 2.4.2 Trim in Climbing and Descending Flight

Climbing and descending flight condition are also simulated with a gross weight of 16,000 lbs at an altitude of 5,250 feet. The following results are investigated with 60 knots forward flight speed at the flight path angle range from -20° to 25°. The fuselage pitch angles along the flight path (climb and descent) angle are shown along with the flight testing data in Fig. 2.22.

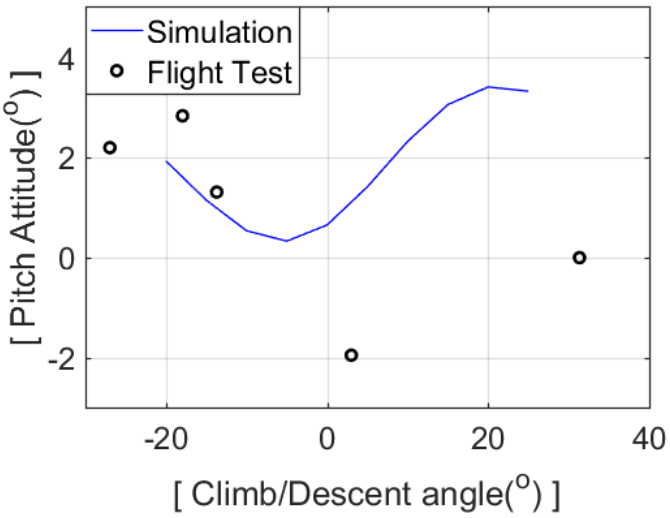



Figure 2.22: Fuselage pitch angle vs. climb/descent angle (16,000 lbs at 5,250 feet)

Zero flight path angle means the steady level flight at 60 knots which has an approximately 1-degree nose-up helicopter pitch angle. Negative angles mean descending flight and positive angles mean climbing flight states. Considering the tendency and magnitude, the prediction is still in the reasonable range. However, it is also difficult to investigate this case since flight test data only have 5 data points. The comparisons of the four predicted (simulated) control inputs with flight test data are presented in Figs. 2.23, 2.24, 2.25, and 2.26. The predicted control inputs are close to the available flight test data for the entire flight speed range.

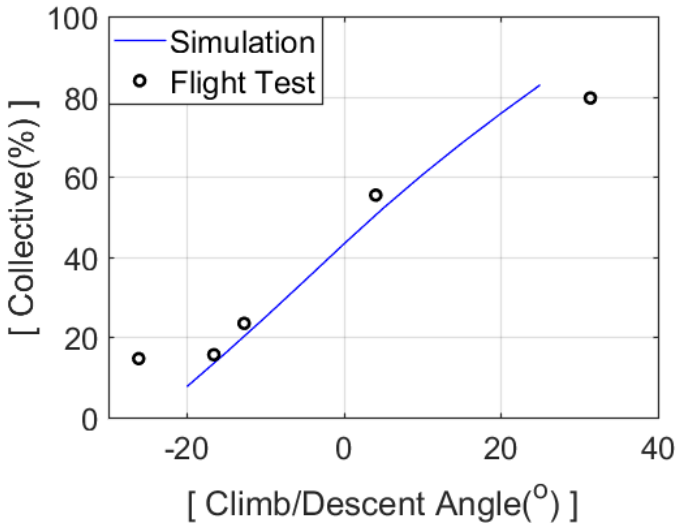



Figure 2.23: Collective stick vs. climb/descent angle (16,000 lbs at 5,250 feet)

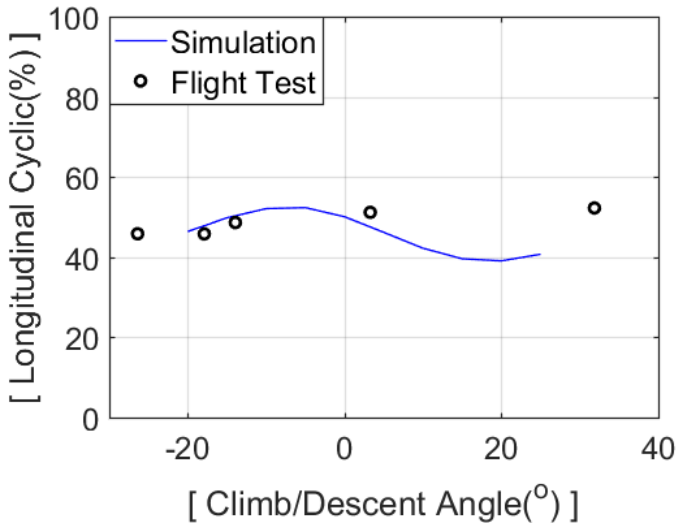



Figure 2.24: Longitudinal cyclic vs. climb/descent angle (16,000 lbs at 5,250 feet)

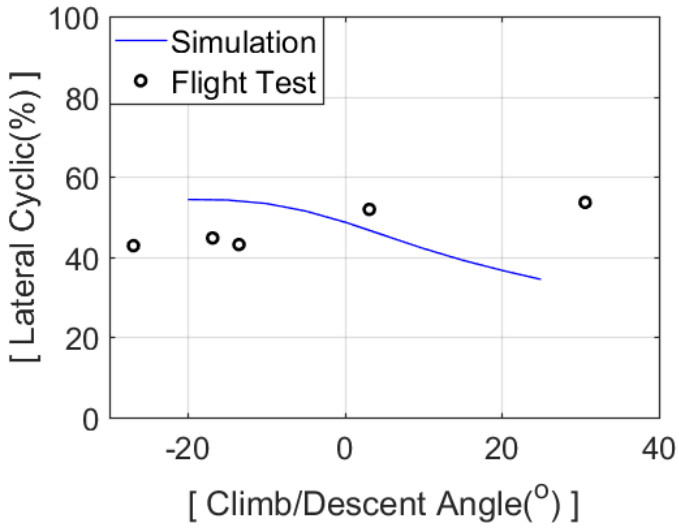



Figure 2.25: Lateral cyclic vs. climb/descent angle (16,000 lbs at 5,250 feet)

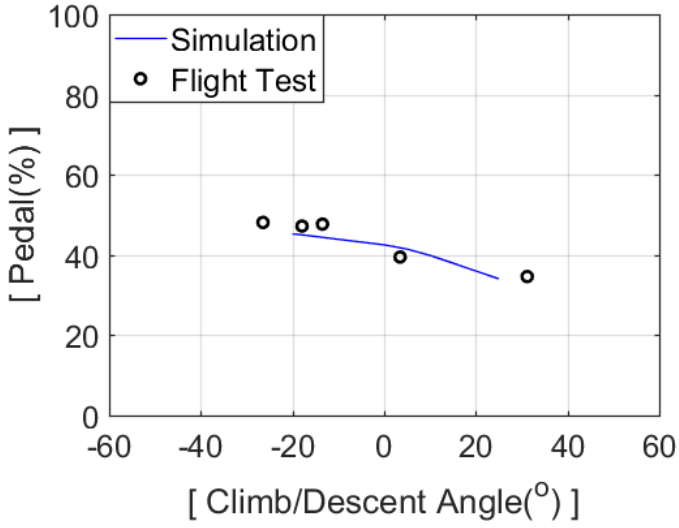



Figure 2.26: Pedal vs. climb/descent angle (16,000 lbs at 5,250 feet)

### 2.4.3 Trim in Steady Turning Flight

In trimmed turning flight, the turn rate is a constant, which is nonzero. Since flight test data are obtained with respect to the roll angle, the relation between the turn rate and roll angle is

investigated first. Steady turning is also simulated with a gross weight of 16,000 lbs at an altitude of 5,250 feet. The range of turn rates is from -25 to 25 ($^\circ/sec$). The fuselage roll angles corresponding to the turn rates and the fuselage pitch angles are investigated in Figs. 2.27 and 2.28, respectively.

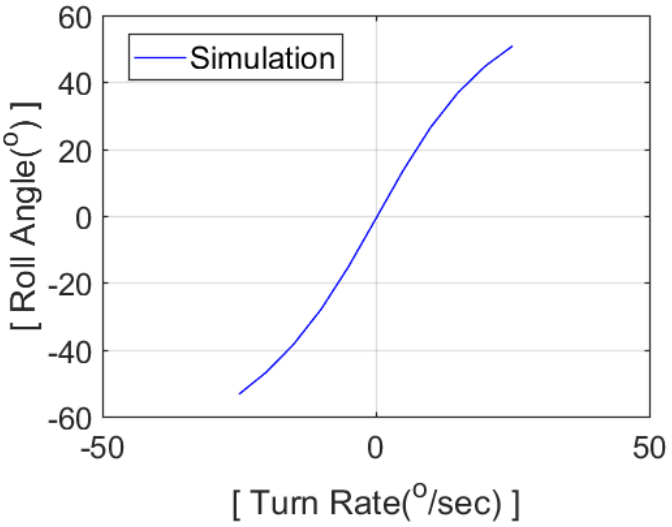



Figure 2.27: Fuselage roll angle vs. turn rate (16,000 lbs at 5,250 feet)

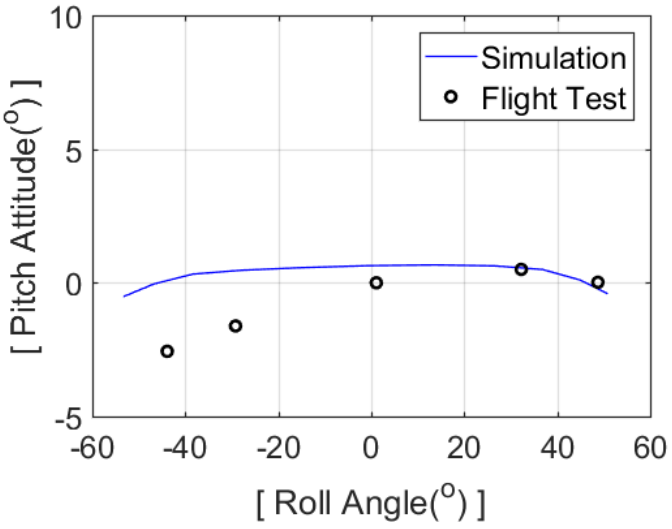



Figure 2.28: Fuselage pitch angle vs. roll angle (16,000 lbs at 5,250 feet)

The flight test data exist as a function of roll angles, so the predicted control inputs are compared as shown in Figs. 2.29, 2.30, 2.31, and 2.32.

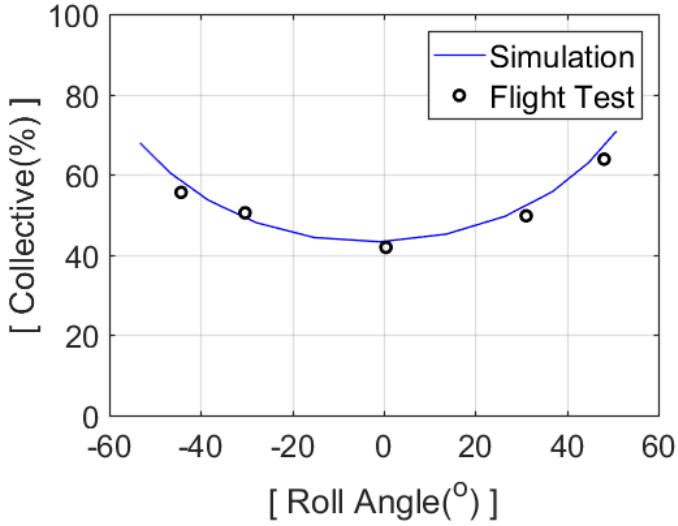



Figure 2.29: Collective stick vs. roll angle (16,000 lbs at 5,250 feet)

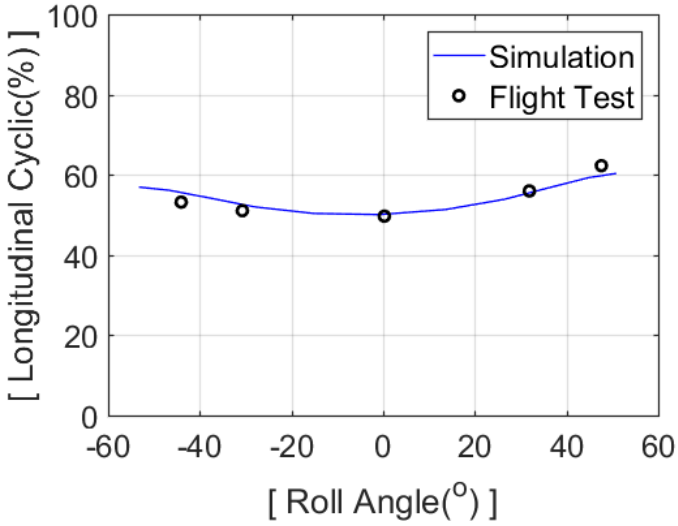



Figure 2.30: Longitudinal cyclic vs. roll angle (16,000 lbs at 5,250 feet)

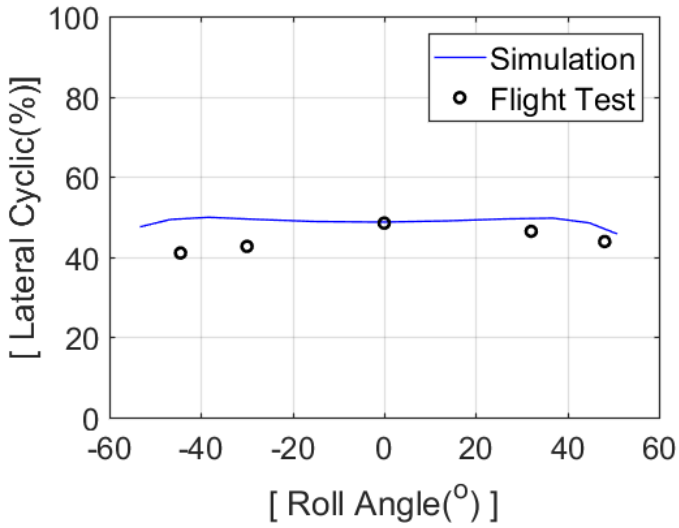



Figure 2.31: Lateral cyclic vs. roll angle (16,000 lbs at 5,250 feet)

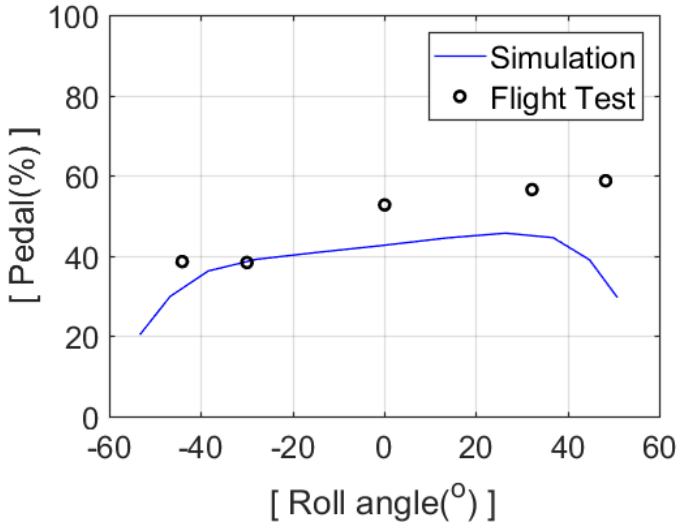



Figure 2.32: Pedal vs. roll angle (16,000 lbs at 5,250 feet)

Although the available flight test data for validation are limited, the control inputs are predicted closely.

## 2.5 Extraction of Linearized Model

A linearized model is extracted based on a first-order Taylor series expansion of the nonlinear system about an equilibrium (trim) point. By expanding the left-hand side of the Eqn. 2.1 in a Taylor series, it yields the Eqn. 2.81.

$$f \quad + \quad \frac{\partial f}{\partial \dot{y}}\Delta\dot{y} \quad + \quad \frac{\partial f}{\partial y}\Delta y \quad + \quad \frac{\partial f}{\partial u}\Delta u \quad + \quad \cdots \quad = \quad \epsilon \tag{2.81}$$

At equilibrium(f = $\epsilon \stackrel{def}{=} 0$), the Jacobian matrices are expressed in Eqn. 2.82.

$$E = \frac{\partial \epsilon}{\partial \dot{y}}\Big|_{trim}, \qquad F = \frac{\partial \epsilon}{\partial y}\Big|_{trim}, \qquad G = \frac{\partial \epsilon}{\partial u}\Big|_{trim} \tag{2.82}$$

Neglecting the higher-order terms, it yields the linearized system dynamics about equilibrium as shown in Eqn. 2.83.

$$E\Delta\dot{y} \quad + \quad F\Delta y \quad + \quad G\Delta u \quad = \quad 0 \tag{2.83}$$

By rearranging the above equation with respect to $\Delta\dot{y}$, it yields the Eqn. 2.84.

$$\begin{aligned} \Delta\dot{y} &= A\Delta y + B\Delta u \\ A &= -E^{-1}F \\ B &= -E^{-1}G \end{aligned} \tag{2.84}$$

A and B matrix are extracted at a given flight condition which defines the specific model(i.e. 60 knots forward flight model). By using Eqn. 3.2 and a state-to-output conversion matrix C, transfer functions between pilot inputs and system outputs for the relevant physical quantities can be constructed as shown in Eqn. 2.85.

$$H(s) \quad = \quad C(sI \quad - \quad A)^{-1}B \quad + \quad D \tag{2.85}$$

D matrix represents the direct influence of the inputs on the outputs. The nature of the aero-

dynamics and rotor dynamics introduces time delays between the application of input and the establishment of steady-state response. The control inputs influence the force distributions over the rotor disks, modifying the rotor and airframe accelerations. These accelerations, integrated over time, manifest as changes in the positions and velocities which are the system states and outputs. Therefore, the matrix D is identically zero.

The objective of the vision system is to provide the visual information required in a given situation. This chapter explains how to detect objects of interest and estimate the relative position and orientation and then validate its accuracy. Considering the visual references a helicopter pilot refers to while approaching the ship, different detection strategies are developed depending on the distance. Throughout the study, a gimballed camera of the VTOL UAV is used to capture images that can mechanically compensate for the roll and pitch motions of the UAV.

## 3.1 Detection Methodology

The development begins by understanding how a helicopter pilot perceives and acts during different situations while approaching and landing on a ship. First, the pilot visually confirms the ship's location in the long-distance and then determines a course and speed for the approach. Once the helicopter reaches close proximity of the ship, the horizon reference bar becomes visible, and from that point, the pilot controls the helicopter in a stable fashion by referring to the horizon bar, which indicates a true horizon independent of the ship motions, for a safe landing. In the developed vision system, the same strategy is automated for VTOL UAVs by taking advantage of state-of-the-art machine learning based object detection and classical computer vision methods.

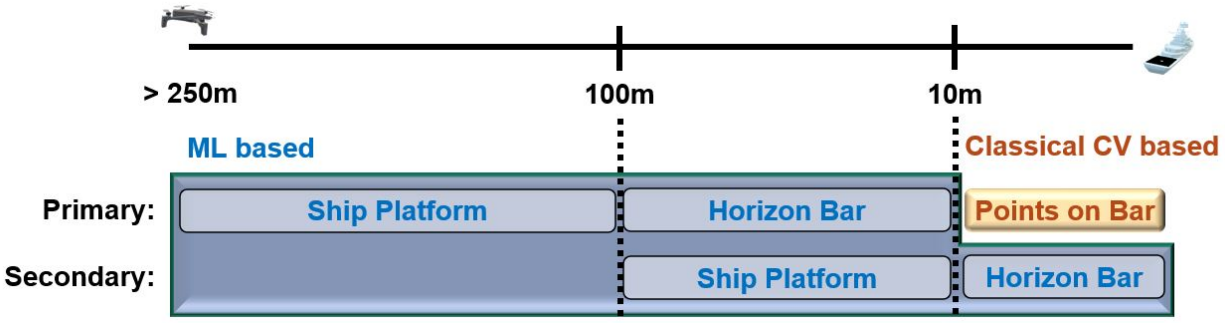



Figure 3.1: Tracking object depending on distance

As shown in Fig. 3.1, the ship, horizon bar, and corner points on the bar are the primary detection targets depending on the distance. In case those target objects are not successfully detected, the vision system detects the secondary objects as backups. Note that it is only possible to detect the corner points on the horizon bar at close proximity to the landing pad.

### 3.1.1 Machine/Deep Learning-based Vision

To detect the ship as a whole in the long-range and the horizon bar in the mid-range, machine/deep learning-based object detectors are developed. Classical computer vision may achieve the same task, however, it requires explicit algorithms for detection which leads to being complicated and also challenging to capture every aspect of the object. On the contrary, machine/deep learning-based object detection learns the characteristics of the object thoroughly using neural networks. The speed and accuracy of this detection process are remarkably improved by using advanced Graphics Processing Units (GPUs).

The ImageNet Large Scale Visual Recognition Challenge (ILSVRC) which was held annually from 2010 to 2017 became an opportunity to enhance the object detection capability by utilizing machine/deep learning algorithms [77]. The accuracy achieved each year by the state-of-the-art algorithms is compared as shown in Fig. 3.2.

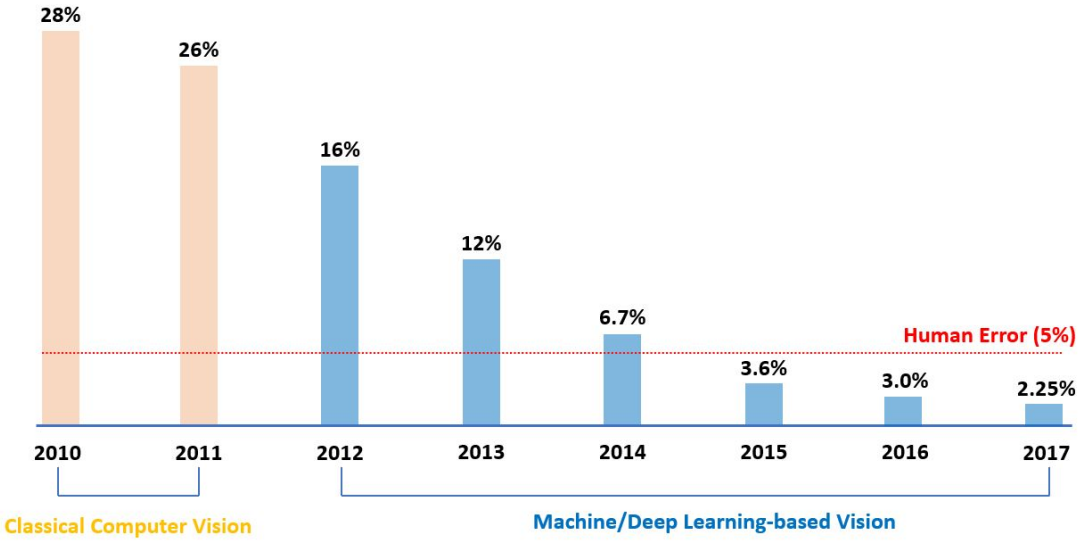



Figure 3.2: Accuracy achieved by year in ILSVRC

ILSVRC is a benchmark in object category classification and detection using millions of images. There have been many algorithms tried such as AlexNet, ZF, VGG, GoogLeNet, ResNet, and SENet. Since 2015, it began performing better than human eyes. ILSVRC investigated the accuracy in three different categories which are image classification, single-object localization, and object detection as shown in Fig. 3.3.

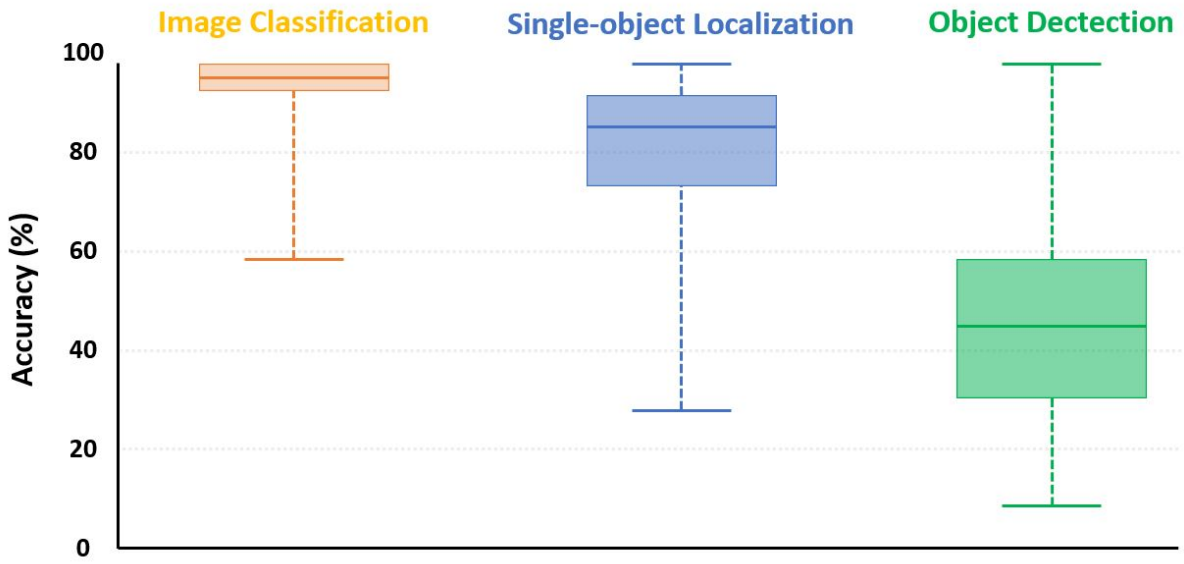



Figure 3.3: Accuracy of image classification, single-object localization, and object detection in ILSVRC during 2012 - 2014

The image classification task is to classify the object in an image without identifying its position. The accuracy is based on the rate that a label created by the algorithm is matched to the actual object. The single-object localization task includes identifying the object position in the image and the object detection task is to detect multiple objects and localize them in the image.

According to the accuracy achieved in the three tasks from 2012 to 2014, the image classification model achieves 94.6% accuracy on average, however, there is a 41% absolute difference in accuracy between the most and least accurate object class. The single-object localization model achieves 81.5% accuracy on average with a 77% total range and the object detection model

achieves 44.7% accuracy on average with an 84.7% total range. The accuracy will increase as research progresses; however, it is clear that a single-object localization task has better accuracy than a multiple object detection task. Thus, instead of having one object detector that identifies two objects (ship and horizon bar), two separate object detectors are developed for each object so that they do not have to distinguish one object from the other. By this approach, each detector only needs to find a particular object in the image, and the object class is automatically assigned without incurring the risk of false classification.

Developing such object detectors involves three steps, which are, collecting images, labeling objects in the images, and training the object detector. First, 2,000 ship platform images and 1,000 horizon bar images are individually collected by UAV's onboard camera. To include various aspects of object figures in training sets, images are captured in different perspectives, lights, weather conditions, and distances. Second, the objects in the collected images are manually labeled with their bounding boxes. Even though the ship images contain the horizon bar, only one object per image is labeled and training sets for the ship detection and horizon bar detection are strictly separated to develop two single-object detectors. These data are stored as the pixel position of the bounding box and the object identification number that indicates object class/name. Third, the set of labeled images are used for training by a state-of-the-art machine learning algorithm.

To implement the machine/deep learning-based vision on a real-time flight control system, there is also an important factor, which is detection speed. Several algorithms have been developed for the purpose of fast detection as well as higher accuracy such as Region-based CNN (R-CNN) [78, 79, 80], Single Shot Detector (SSD) [81], and You Look Only Once (YOLO) [82, 83, 84]. They commonly have the architecture of convolutional neural network (CNN) which is a class of deep neural networks. R-CNN is classified as a two-stage detector that combines region-proposal algorithms with CNN to extract 2,000 regions through a selective search, then classifies the selected regions on the image. Its recent variant, Faster R-CNN, is developed to improve the detection speed by replacing the slow selective region search process. Meanwhile, SSD and YOLO are classified as one-stage detectors that treat object detection as a regression problem by taking an input image and

learning the probability of an object class and bounding box coordinates simultaneously. There was a study that compared the speed and accuracy of such state-of-the-art algorithms, where YOLOv3 demonstrated faster detection speed than Faster R-CNN and SSD [84, 85]. Hence, the YOLOv3 algorithm is finally selected to train an object detector. The YOLOv3 object detection task consists of object classification and localization as shown in Fig. 3.4.

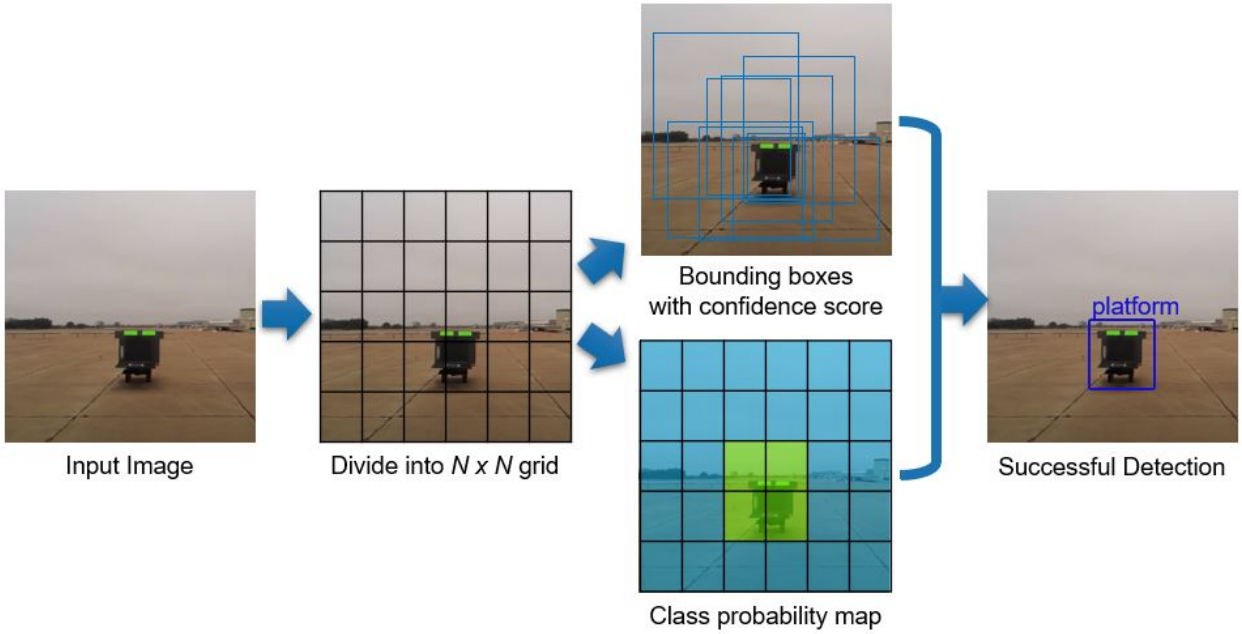



Figure 3.4: Ship platform detection process by YOLOv3 algorithm

The one-stage detector YOLOv3 regards the detection as a regression problem and uses a single neural network. It analyzes the entire image to predict the object-bounding box. The input image is divided into an N x N grid and each grid cell predicts bounding boxes with a confidence score and class probability. To better detect the object in different sizes, it predicts bounding boxes at three different scales, which helps to detect the object from a far distance. The predictions of developed detector are encoded as an $N \times N \times [3 * (4 + 1 + 1)]$ tensor for $N \times N$ grid cells, 3 different scales, 4 bounding box offsets, 1 object confidence score, and 1 class prediction. The observed maximum ranges for detecting the present sub-scale ship platform and horizon bar are approximately 250m

and 100m, respectively. The real-time detection at different distances is shown as bounding boxes with object names in Fig. 3.5.


Figure 3.5: Long- and mid-range real-time object detection result

The verified maximum range is approximately 250 meters (820 feet) in the case of tracking a 6 x 6 feet object (sub-scale ship platform). Since the detection range is proportional to the object's occupying area in the image, a typical small ship whose rear-side occupies 50 x 50 feet area can be detected from 17.3 kilometers (9.3 nautical miles) away. This is 18 times greater than the Missed Approach Point (MAP) distance, which is a point at which a pilot must identify the ship visually. Once the target object is detected, it provides the object position and its bounding box in the image to the control system. Although the actual relative position and orientation are not estimated, the developed control system achieves autonomous flight in a long distance using the object size and its position in the image.

### 3.1.2 Classical Computer Vision

The state-of-the-art machine/deep learning-based approach has the advantage of detecting the visual cue from far away. However, the disadvantage of the machine/deep learning-based approach is that training is non-deterministic, meaning that the detection can be failed for unknown reasons. For example, the model could identify some other objects as the visual cue and the object bounding box could be larger or smaller than the original object size. These failures are not acceptable in the case of close-range tracking and accurate landing. Also, the machine/deep learning-based approach requires high computational loads, which can lead to latency. In this regard, classical computer vision algorithms that explicitly instruct what to do for detection, and execute in no time are used to develop a close-range vision system.

Once the UAV reaches close proximity of the ship platform, it is required to detect the visual cue and then estimate its relative position and orientation for the final approach and landing. The visual cue should be installed perpendicular to the landing pad and parallel to the pilot's/aircraft's line of sight in the same way that the horizon reference bar is installed on a ship. The visual cue does not need to have any particular form as long as the dimensions are known apriori. However, unique features that are distinguishable from the surroundings are favorable for detection. since the onboard front-facing camera has pitch and roll mechanical gimbals, the camera stays level with the horizon. Thus, captured images do not experience the effects of pitch or roll motions of the UAV.

This reduces the complexity caused by rotational motions. However, the effect of yaw is reflected in the images, which is utilized to extract the heading of the UAV with respect to the visual cue. A number of different visual cues are tested and representative cases are shown in Fig. 3.6.

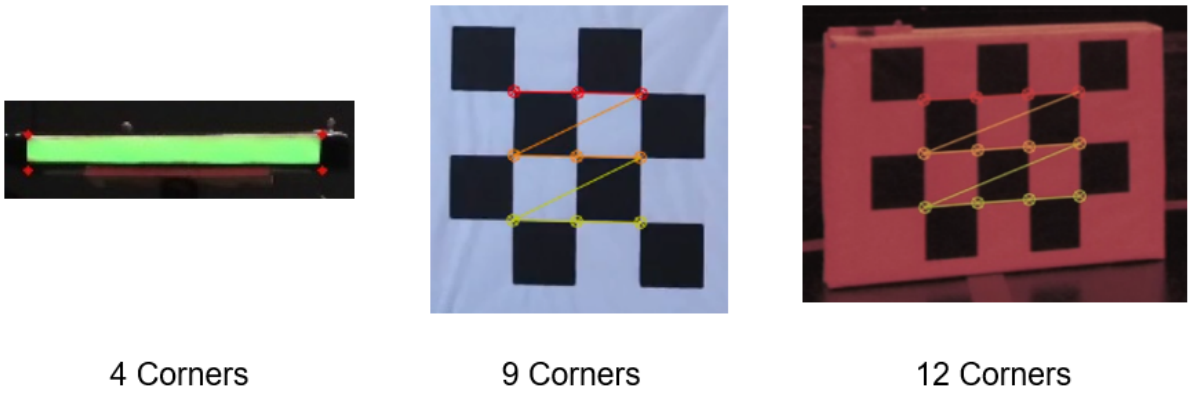



Figure 3.6: Representative visual cues tested

Among many candidates, checkerboard pattern visual cue is investigated first due to its distinctive features and a redundant number of corners. Its geometry exhibits local image features such as edges, lines, and corners which are greatly helpful in detection. From the experiments that utilize checkerboard pattern visual cues, the effect of camera resolution and the size of squares in the checkerboard on the detection range is studied. The maximum resolution of the camera is 4k ultra-high definition, UHD (4096 x 2160), and the higher resolution helps in increasing the detection range. The size of squares on the checkerboard, technically the spacing between corners, is the other factor that affects the detection range. As the square size increases, the effective range for detection also increases. However, there is a trade-off between the maximum effective range and the closest proximity from the visual cue since the camera has to be located farther away for the visual cue detection in case of a bigger square size. The effective ranges for each case are found through experiments and are provided in Table 3.1.

Table 3.1: Valid range for varying resolution & square size

| Resolution | 80 mm Square | 120 mm Square |
|---|---|---|
| qHD(540p) | 13 m | 15.5 m |
| HD(720p) | 14 m | 17.5 m |
| FHD(1080p) | 15 m | 19 m |
| 4K UHD(2160p) | 17 m | 25 m |

According to the results, the square size is selected as 120 mm since it can be detected accurately from distances as high as 25 meters, and the square size is small enough to be captured even when the visual cue is only 1 meter away. Although higher resolution increases the effective range, it also increases the quantity of data to process. Therefore, HD 720p (1280 x 720) resolution is selected considering the latency in live streaming and image processing for a relatively high bandwidth integrated vision-based feedback control system as well as a good effective range. This visual cue is used in the earlier development phase of the vision-based autonomous ship landing system. Finally, the visual cue that closely mimics the horizon reference bar on Navy ships is constructed and used for close-range detection and pose estimation. It has two green-colored rectangles on a grey background with known size and separation as shown in Fig. 3.7.

Figure 3.7: Constructed horizon reference bar

Considering the characteristics of the installed horizon bar, the corner points of the green rectangles are determined as the targets to be detected. In order to ensure robust detection and estimation in scenarios involving large UAV movements and different light conditions, the established vision system sequentially conducts the image filtering, contour and corner detection, detected points screening, and the estimation of position and orientation.

#### *3.1.2.1 Image Filtering*

The image filtering takes the processing steps as shown in Fig. 3.8.

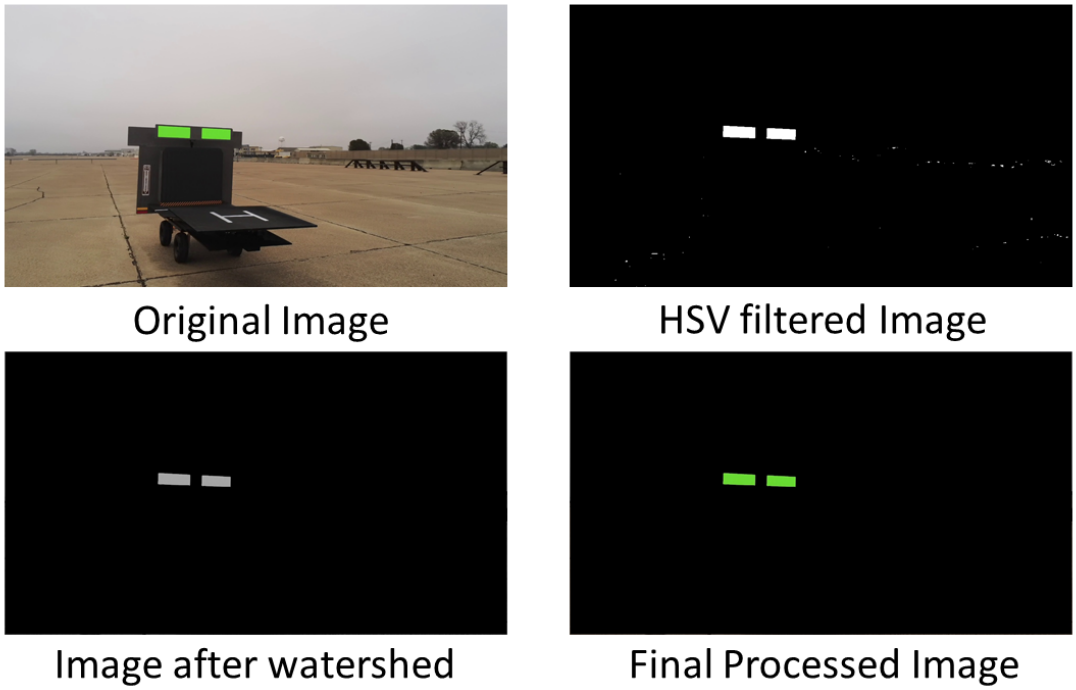



Figure 3.8: Image filtering process

The initial approach involves applying a Hue-Saturation-Value (HSV) filter to sort out the green rectangles. The HSV filter is preferred over Red-Green-Blue (RGB) because the RGB color space is more sensitive to light conditions. To ensure the capture of the green rectangles in different light conditions, the greater range of HSV (H: 35 - 85, S: 70 - 255, V: 90 - 255) is assigned;

however, this inevitably leads to the capture of undesired portions. In the HSV filtered image, there possibly exists small white patches outside the rectangles and black voids inside the rectangles. The morphological opening technique is used to remove the white patches in the image. It first erodes an image removing any small white patches and then dilates the eroded image to preserve the original shape and size. Morphological closing is used to fill up small voids in those rectangles by dilating an image first then eroding the dilated image. By performing these operations, the rectangles are depicted as white regions on the black background. The watershed algorithm [86] is exploited to obtain clear boundaries of the rectangles. The two reference areas are obtained by eroding the processed image by 1% and dilating the processed image by 1%. The gap between the two areas is assumed to contain the boundaries of the rectangles. The algorithm simultaneously expands the area of the background and the rectangles toward each other until they meet at a one-pixel point. By connecting the points, clear boundaries of the rectangles are obtained.

#### *3.1.2.2 Contour and Corner Detection*

Once the green rectangles are isolated by the image filtering process, the detection of contours and corner points is conducted. Even after the filtering, directly implementing any pre-existing corner detection algorithms is prone to detect some false corners. To detect the eight corner points precisely, the contours of the detected region are found and bounded in rectangles first. Thus, the size and shape of the detected areas are very close to the green rectangles and the corners of those bounding rectangles can be used as rough estimates of the actual corners. Second, the Förstner corner detection method is adopted to detect the corner points of the rectangles precisely based on the rough corners obtained by contour detection [87]. The Förstner corner detection increases the accuracy by sub-pixel refinement process as shown in Fig. 3.9. It is based on the fact that an ideal corner is a single point that tangent lines of the object cross perpendicular to each other. This way, false corner detection can be avoided. The pixel information around a corner is not perfectly clear. Therefore, an approximation process for defining the corner position $(c_u, c_v)$ is required and is expressed in Eqn. (3.1).

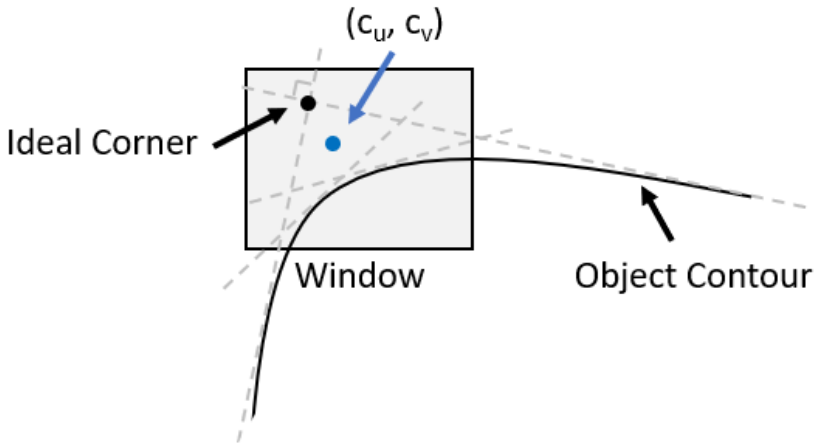



Figure 3.9: Förstner corner detection method

$$(\hat{c_u}, \hat{c_v}) = \underset{c_u, c_v}{\arg\min} \sum_{u,v \in N} ([\nabla f(u,v)]^T (u - c_u, v - c_v))^2 \tag{3.1}$$

The image gradient $\nabla f(u,v)$ at the image pixel position $(u,v)$ is perpendicular to line from $(u,v)$ to corner position $(c_u, c_v)$. In order to obtain $(c_u, c_v)$, a least square estimation in a small window is used. The projection of line from $(c_u, c_v)$ to $(u,v)$ on to the tangent line at $(u,v)$ is required to be maximized in the given window. In other words, the projection of the image gradient onto the line segment connecting $(c_u, c_v)$ and $(u,v)$ has to be minimized for all $(u,v)$ inside the window $N$. One crucial part here is the size of the window chosen for each image. It cannot be a fixed window size because the UAV is always moving towards the landing pad. The size of the detected region increases as it approaches closer to the visual cue. Hence a variable window size which is a function of the width and height is assigned as expressed in Eqn. (3.2).

$$sw(w(t_k), h(t_k)) = \frac{1}{5}\{w(t_k) \times h(t_k)\} \tag{3.2}$$

$sw(w(t_k), h(t_k))$ is the window size at time $t_k$ where $w(t_k)$ and $h(t_k)$ are the width and height

of bounding rectangles in pixels. The detection results are shown in Fig. 3.10.

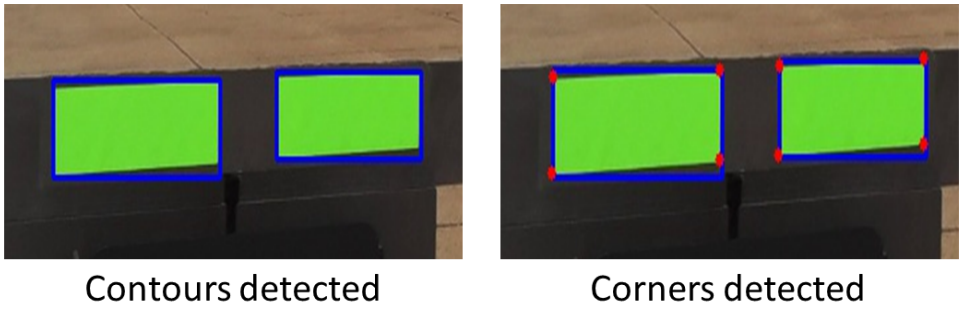



Figure 3.10: Detection of contours and corners

### *3.1.2.3 Detected Points Screening*

The screening procedure is established to assure that no false corners are present. All the corners are sorted in a particular order as shown in Fig. 3.11, which helps in finding the length and slope of each side of the rectangles in the image. Even though they are not perfect rectangles in the image, width 1, 2, 3, and 4 have similar lengths and slopes. The height 1, 2, 3, and 4 also have similar lengths and slopes. A $\pm 10\%$ tolerance level is set for the lengths and a $\pm 5\%$ tolerance level is set for the slopes.

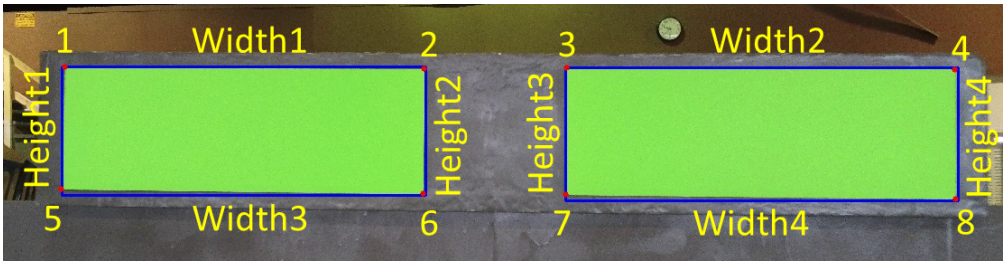



Figure 3.11: Detected corners in the sorted order

## 3.2 Estimation of Relative Position and Orientation

The estimation is based on a single camera calibration method using a planar object [88, 89]. The geometric relation of the image and real-world coordinates is shown in Fig. 3.12.

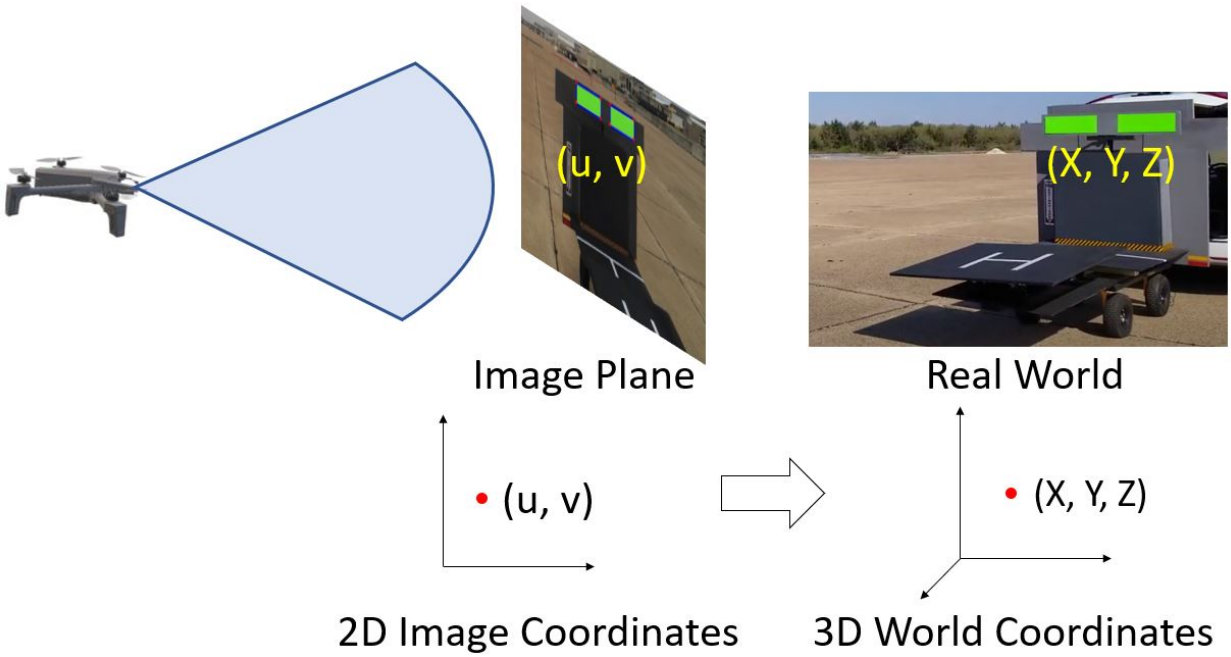



Figure 3.12: Geometric relationship of 2D image coordinates and 3D real-world coordinates

A conventional pinhole camera model is used to derive the geometric relation. A 3D coordinate system can be defined with respect to the visual cue and there is a 2D coordinate system associated with the image frame. $(X, Y, Z)$ is a point on the object described in the visual cue body-fixed frame, and $(u, v)$ is a corresponding point on the image frame (pixel position in the image). The relationship between the image coordinates and the body-fixed coordinates in matrix form is described in Eqn. (3.3).

$$s \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \boldsymbol{A} \begin{bmatrix} \boldsymbol{R} & t \\ 0_{1x3} & 1 \end{bmatrix}^{-1} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{3.3}$$

The equation is derived in a homogeneous coordinate system. In the homogeneous coordinate system, the 2D image point $(u, v)$ becomes a 3D vector $(u, v, 1)$ and the 3D world coordinates $(X, Y, Z)$ become 4D vector $(X, Y, Z, 1)$. The scaling factor $s$ comes from transforming the homogeneous coordinates to cartesian coordinates. $R$ is a 3 x 3 rotation matrix and $t$ is a 3 x 1 translation vector. $R$ matrix has the information about the camera orientation with respect to the visual cue and $t$ provides the information on how far the camera is from the particular $(X, Y, Z)$ point in the visual cue body-fixed frame. $O_{1x3}$ is 1 x 3 zero matrix. The matrix which includes $R$, $t$, $O_{1x3}$, and 1 is called the camera extrinsic matrix, which varies from image to image. The matrix $A$ is called the camera intrinsic matrix and its components are shown in Eqn. (3.4).

$$\boldsymbol{A} = \begin{bmatrix} 1/\rho_u & 0 & u_0 \\ 0 & 1/\rho_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} f_x & 0 & 0 & 0 \\ 0 & f_y & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{3.4}$$

$1/\rho_u$ and $1/\rho_v$ are parameters that scale the image coordinates, which are in pixels to values in the International System of Units (SI). $u_0$ and $v_0$ are the center position in the image plane. The first matrix shifts the center of the image plane to the top left corner point. The second matrix contains the focal length information $f_x$ and $f_y$, which are the focal lengths of the camera in the $x$ and $y$ directions, respectively. The product of two matrices forms the camera intrinsic matrix, which depends on the particular camera. This unique camera intrinsic matrix can be found using a camera calibration [90].

To determine the relative position and orientation of the camera, the Perspective-n-Point (PnP) algorithm is applied. Given a set of $n$ 3-D coordinates of an object and its corresponding 2-D

projections on the image, this algorithm solves Eqn. (3.3) to obtain the rotation vectors $R$ and the translation vectors $t$. There are 6 DOF for a camera, which are 3 DOF in rotation (roll, pitch, yaw) and 3 DOF in translation $(X, Y, Z)$. A minimum of 3 points are required to find a solution, but the solution is not unique. There should be a minimum of 4 points to obtain a unique solution; however, it can be more reliable and redundant when there are more points. An iterative method is used for the PnP algorithm since it is robust for objects which consist of a planar surface and gave more accurate results. The iterative method is based on Levenberg-Marquardt optimization [91, 92]. In this method, the function minimizes re-projection error, which is the sum of squared distances between the observed image points $(u, v)$ and projected object points $(X, Y, Z)$. By default, the iterative algorithm sets the initial value of rotation and translation as zero and then updates during each iteration.

The RANSAC method [93, 94] is used to find a rough initial guess for the extrinsic matrix in an iterative approach. The RANSAC method also identifies the outliers and removes them during the calculation. The initial guess and inliers are fed into the iterative algorithm to have a more accurate estimation. The solvePnP algorithm [90] returns a rotation vector and the translation vector. The rotation vector can be converted to the rotation matrix using the Rodrigues function. Thus, once the rotation matrix ($R$) and translation vector ($t$) are obtained, $-R^{-1}t$ gives the relative distances in 3D from the camera to the origin of the visual cue body-fixed frame.

The present estimation method is modified from the existing algorithm to take advantage of the gimbal camera. Since the gimbal corrects for the roll and pitch motion of the UAV, the images only reflect the yaw motion (heading angle) with respect to the visual cue. Thus, roll and pitch angles computed by the image can be regarded as camera noise and gimbal correction errors. Therefore, the roll and pitch angles are excluded from the position estimation. The basic and modified rotation matrix are specified in Eqs. (3.5) and (3.6), respectively.

$$\boldsymbol{R_{basic}} = \begin{bmatrix} c\alpha c\beta & c\alpha s\beta s\gamma - s\alpha c\gamma & c\alpha s\beta c\gamma + s\alpha s\gamma \\ s\alpha c\beta & s\alpha s\beta s\gamma - c\alpha c\gamma & s\alpha s\beta c\gamma + c\alpha s\gamma \\ -s\beta & c\beta s\gamma & c\beta c\gamma \end{bmatrix} \tag{3.5}$$

$$\boldsymbol{R_{modified}} = \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & -\cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.6}$$

$\alpha$, $\beta$, and $\gamma$ represent yaw, pitch, and roll angles, respectively. Each row of the $R$ matrix contributes to their respective coordinates $(X, Y, Z)$. By setting the pitch and roll angle to zero, the modified rotation matrix is obtained from the basic rotation matrix. Only the yaw angle remains, which takes care of the relative heading angle. The third row of the modified rotation matrix shows that the $Z$-coordinate is independent of the yaw angle, $\alpha$. Hence, the yaw angle has no contribution in the estimation of the forward relative distance ($Z$-coordinate) between the UAV and the visual cue. However, the sideward relative distance ($X$-coordinate) and vertical relative distance ($Y$-coordinate) are heavily dependent on the yaw angle. The camera position with respect to the visual cue is given in Eqn. (3.7).

$$\begin{bmatrix} X_{cam} \\ Y_{cam} \\ Z_{cam} \end{bmatrix} = -\boldsymbol{R_{modified}}^{-1} t \tag{3.7}$$

It is apparent from multiple experiments that the yaw angle estimation becomes noisy as the camera gets farther away from the visual cue. Since the estimation is based on the number of visual cue pixels in the image, the changes in pixels with distance result in the noise. Despite this sensitivity issue, yaw estimation still shows a reasonable trend within the range that the forward relative distance is accurately estimated. To utilize the yaw estimation trend, instead of directly taking the noisy yaw angle estimation, a moving average filter is configured first and the results are shown in Fig. 3.13.

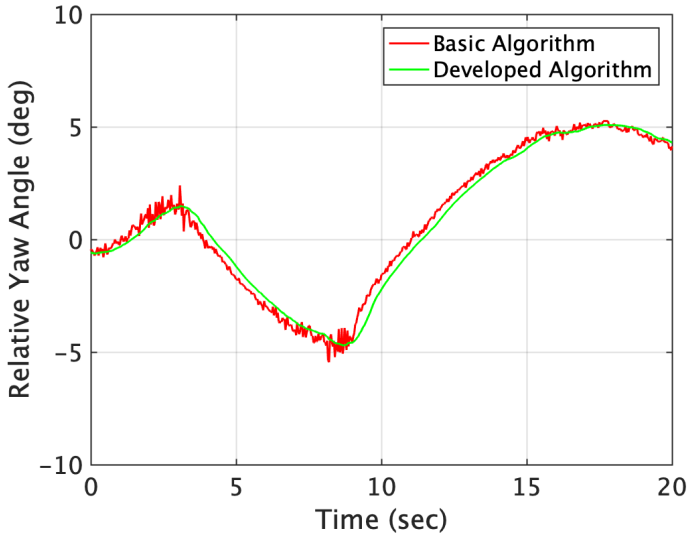



Figure 3.13: Effect of moving average filter on yaw

The red line denotes the yaw angle estimation results by the basic algorithm and the blue line denotes the results after the moving average filter is applied. The moving average is calculated using the history of the previous estimation data. A lower and upper bound are defined for the next yaw angle estimation value with respect to the current average. Whenever the estimated yaw angle is out of the defined range, it rejects that value and takes an average value. In this way, the moving average filter provides stable yaw estimation by tracking a trend in a further distance, and the accuracy naturally increases as it gets closer to the visual cue due to having more pixel data in the image.

However, the moving average filter requires multiple accumulated data to predict the current estimation. Thus, a simple low pass filter is configured to reduce the noise level. One of the commonly used real-time data filters is Kalman Filter. Kalman filter predicts the current estimate by taking into account the current measured value, the previous estimate, and the noise level in the data. A single state Kalman filter was opted over other real-time filters because of the lesser number of variables required while computation. Gaussian noise is assumed to be present in the measurement data with a specified mean and variance. The yaw estimation method is given in Eqn. 3.8.

$$
\begin{aligned}
CE &= PE + KG \times CM \\
KG &= PrE \times \frac{1}{PrE + RN} \\
PrE &= PE + QN
\end{aligned}
\tag{3.8}
$$

CE is the current yaw estimate, PE is the previous yaw estimate, KG is the Kalman Gain, CM is the current yaw measurement, RN is the process noise covariance, and QN is the measurement noise covariance. RN and QN values are experimental values found by analyzing different sets of yaw angle measurements and they are set to 0.005 and 0.05, respectively. The Kalman filter effectively reduced the fluctuations in yaw estimations as shown in Fig. 3.14.

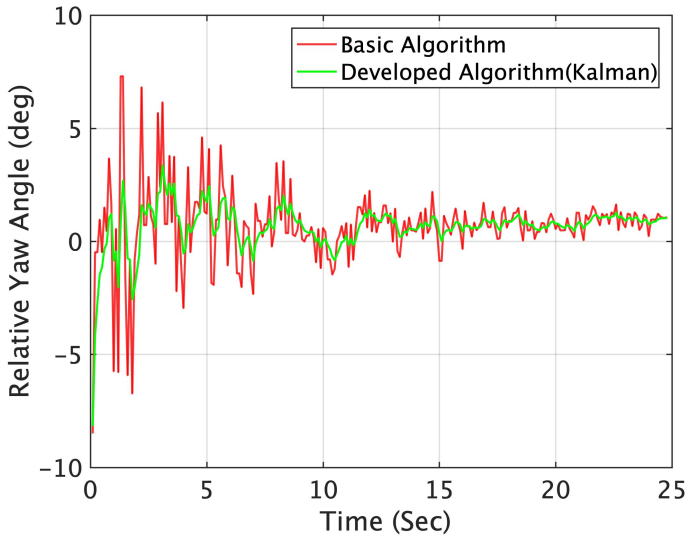



Figure 3.14: Effect of Kalman filter on yaw

## 3.3 Validation

The results obtained from the developed computer vision system are validated through position and attitude measurements using a Vicon motion capture system shown in Fig. 3.15.

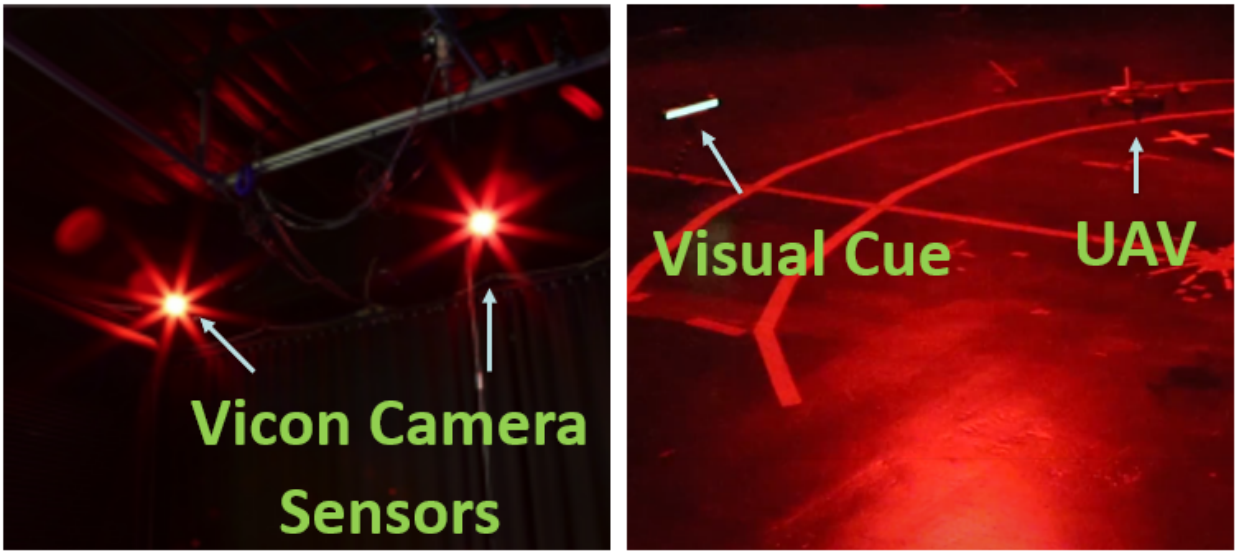



Figure 3.15: Experimental setup in Vicon system

Vicon motion capture system is widely used in the entertainment industry to track the motion of people and other objects even up to sub-millimeter resolution. The Vicon system used for this study can capture images at 120 Hz using eight 16-megapixel digital video cameras. Camera-mounted strobes illuminate small, retro-reflective markers, which are identified and processed by the imagers. The Vicon cameras track reflective markers on both the UAV and the visual cue to obtain precise position and orientation data, which is then used as the ground truth for this study to validate the computer vision system.

The validation is for the estimation method along with the developed algorithm and conducted with the checkerboard visual cue with 9 corners. The detection method varies depending on the selection of visual cues, however, the estimation method that computes the relative position and angle is the same for every visual cue. The developed algorithm applied here is based on the moving average filter which is replaced with the Kalman filter later. The comparisons of the forward relative distance and sideward relative distance are shown in Fig. 3.16 and 3.17, respectively.

The blue, red, and green lines denote the Vicon measurements, baseline algorithm estimations, and estimations from the present improved algorithm, respectively. As seen from the figure, the

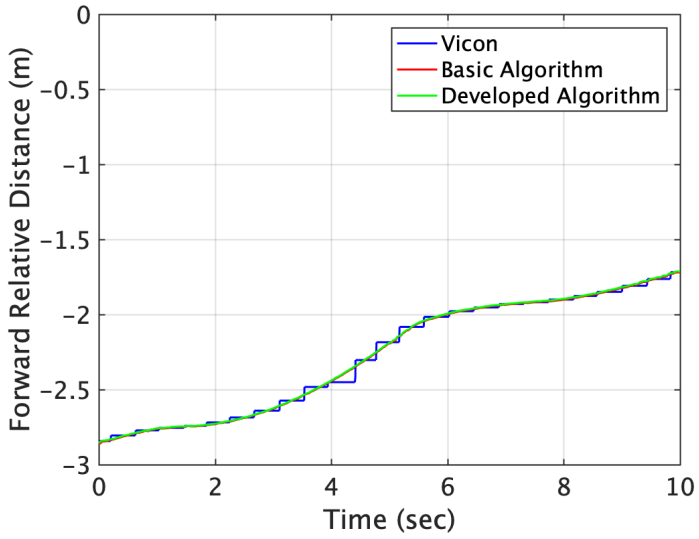



Figure 3.16: Validation of forward distance estimation

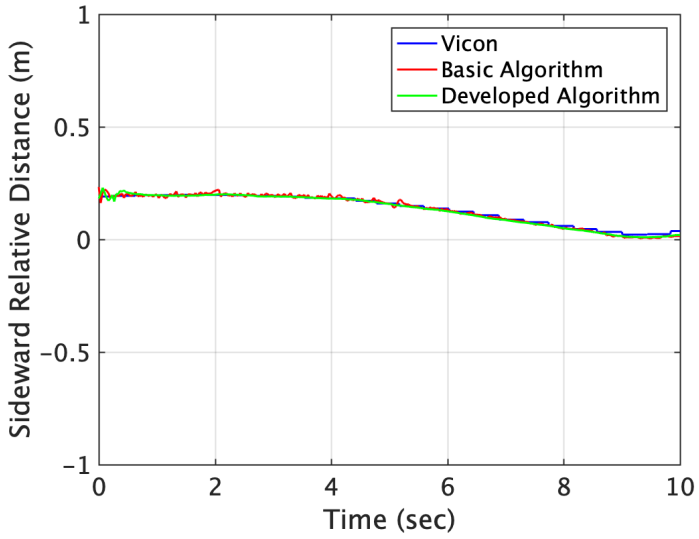



Figure 3.17: Validation of sideward distance estimation

baseline and the improved algorithms have no difference when it comes to the forward relative distance estimation, and both these methods can obtain the same level of accuracy as the Vicon measurements. The sideward distance estimation by the baseline algorithm has fluctuations; however, the improved algorithm shows smooth results due to the moving average filter. The maximum

error in sideward distance estimation is 1 centimeter when compared to the Vicon result. The comparisons of the vertical relative distance and relative yaw angle are as shown in Fig. 3.18 and 3.18, respectively.

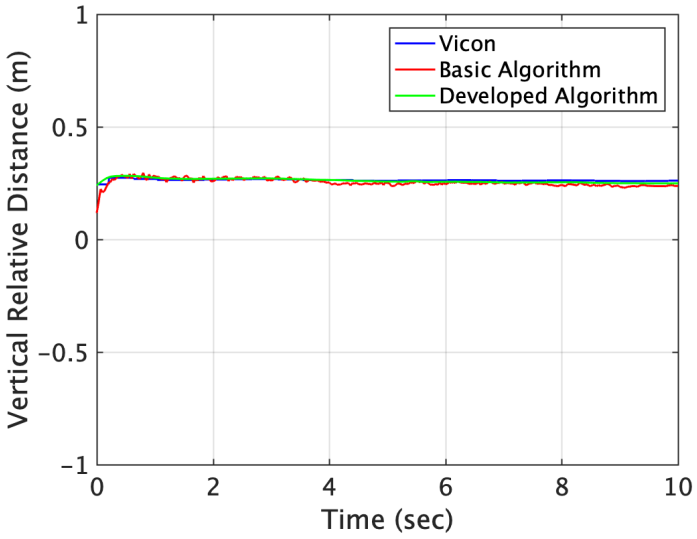



Figure 3.18: Validation of vertical distance estimation

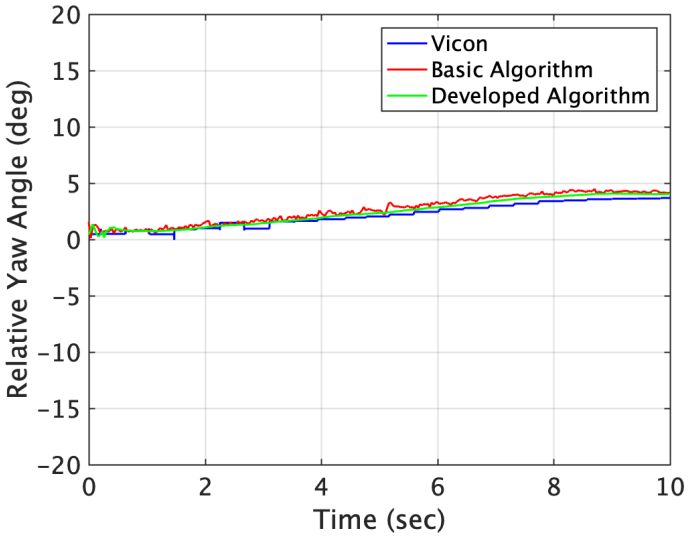



Figure 3.19: Validation of yaw angle estimation

In both vertical relative distance and relative yaw angle comparison, the improved algorithm yields smooth estimation results. The maximum error in the vertical relative distance is below 1 centimeter and the maximum error in the relative yaw angle is 1 degree. Through the Vicon experiments, the ability of the present computer vision system to detect the visual cue and precisely estimate the position and orientation is demonstrated.

# 4. CONTROL SYSTEMS AND SIMULATIONS

This chapter discusses the details of control systems that have been applied for autonomous ship landing along with simulation results. Various control strategies such as optimal control, gain-scheduled control, nonlinear control, and deep reinforcement learning control are developed with respective control objectives. First, helicopter ship landing simulations with the optimal control system are conducted to prove the novel ship landing concept. To verify tracking capability, landing accuracy, and disturbance rejection capability as a preliminary step of flight testing, the gain-scheduled control, nonlinear control, and deep reinforcement learning control systems are developed and extensively simulated in the Gazebo simulation program.

## 4.1 Optimal Control Strategy

The optimal control system is implemented on the linearized UH-60 helicopter model extracted by the TRAC. The objective of the optimal control system is to investigate realistic helicopter maneuvers while approaching and landing on a ship. To this end, it is assumed that the vision system provides perfect estimations for the relative position and orientation. It demonstrates the tracking capability from approach to landing on a ship. For each phase of flight, corresponding models are used which are extracted from trimmed flight conditions such as hover, steady forward flight, and steady coordinated turn.

### 4.1.1 Linear Quadratic Regulator (LQR) Control Design

LQR for set-point tracking method is used to track prescribed vehicle motions and obtain the control inputs required to fly the desired trajectory which is designed to simulate real helicopter ship approach and landing closely. The trajectory begins 680 meters away from the ship and the helicopter maneuvers consist of initial descent, steady forward flight, steady coordinated turn, deceleration, and final vertical landing.

In order to obtain the feedback gains K from the linearized dynamics, the LQR provides a methodology to stabilize and control a linear system by minimizing a quadratic cost function in

the state deviations from targets and the control inputs. For a linear time-invariant (LTI) system with dynamics given in Eqn. 4.1, the infinite-horizon continuous-time LQR controller yields state feedback gains K to minimize the quadratic cost function.

$$\begin{gathered} J = \int_0^\infty (x^T Q x + \Delta u^T R \Delta u) dt \\ ( \text{ where } \quad x \quad = \quad y \quad - \quad y_{target} \ ) \end{gathered} \tag{4.1}$$

Computing the steady-state values of the states and the control inputs results in zero output error and then forces them to take these values. If the desired final values of the states and control inputs are $x_{ss}$ and $u_{ss}$ respectively, then the new control formula is formed as given in Eqn. 4.2.

$$\Delta u = u_{ss} - K(x - x_{ss}) \tag{4.2}$$

Plugging it in the standard form yields the Eqn. 4.3.

$$\begin{aligned} \dot{x} &= Ax + B(u_{ss} - Kx + Kx_{ss}) \\ y &= Cx \end{aligned} \tag{4.3}$$

When x = $x_{ss}$(no error), and u = $u_{ss}$, it is expressed as Eqn. 4.4.

$$\begin{aligned} O &= Ax_{ss} + Bu_{ss} \\ y_{ss} &= Cx_{ss} \end{aligned} \tag{4.4}$$

It can be re-arranged in matrix form as Eqn. 4.5.

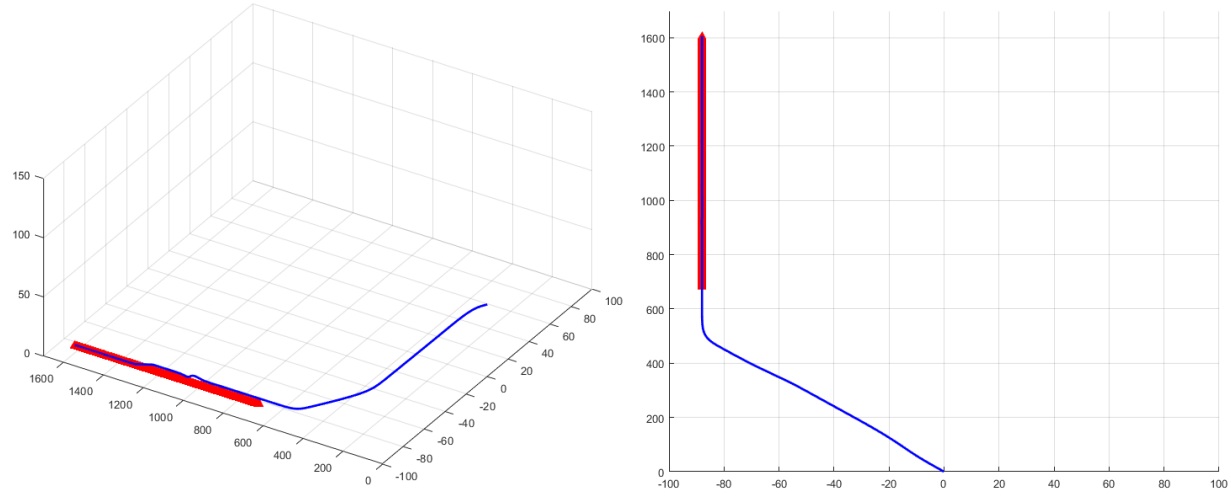
$$\begin{bmatrix} x_{ss} \\ u_{ss} \end{bmatrix} \quad = \quad \begin{bmatrix} A & B \\ Cs & Ds \end{bmatrix}^{-1} \begin{bmatrix} O \\ y_{ss} \end{bmatrix} \tag{4.5}$$

In order to make it feasible, the matrix consisting of A, B, Cs, and Ds components has to be invertible. Hence, Cs and Ds are selected to meet the size of the matrix. In this control system, the number of rows in Cs has to be the same as the number of control inputs which is 4. In other

words, it is able to provide 4 non-zero reference states (setpoints) to track and the other states are regulated to zero at the same time.

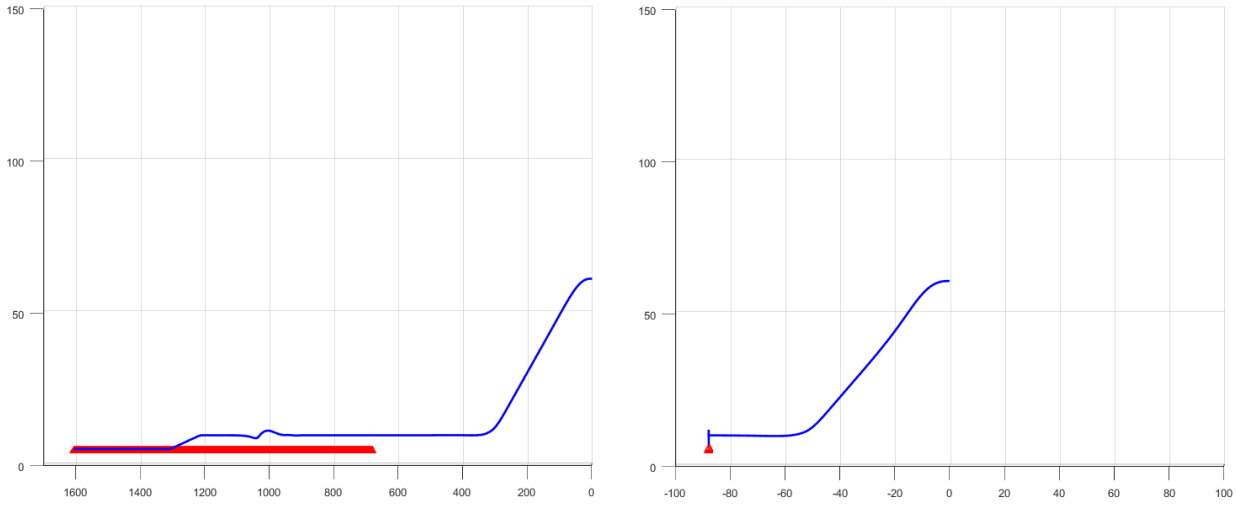
### 4.1.2 Simulation Results

In this section, helicopter states and the corresponding trajectory of the CG are described. The simulation is conducted for 3 minutes without disturbances. For initial conditions, a helicopter is flying forward with 30 knots at an altitude of 200 ft (60.96 m), and its initial position is defined as (0, 0, -60.96) in the earth-fixed frame. A target ship is moving forward at 10 knots. The horizon bar position represents the position of the target ship and its initial position is at (679.73, -88, -5) in the earth-fixed frame. All units are in meters and in order to demonstrate the trajectory more intuitively, the sign of the Z-axis component in the earth-fixed frame is swapped in the following plots (positive upwards). In all the trajectory plots shown in Figs. 4.1 and 4.2, the red markers represent the ship trajectory and the blue line represents the helicopter trajectory.




(a) Diagonal view (b) Top view

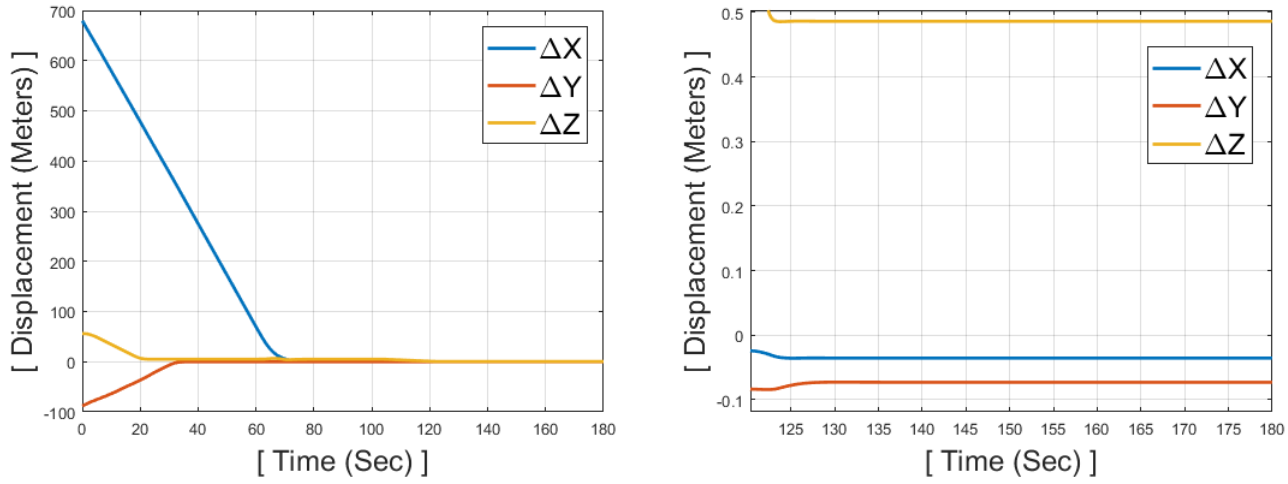
Figure 4.1: Isometric and top view of entire trajectory




(a) Side view (b) Rear view

Figure 4.2: Side and rear view of entire trajectory

In order to check the trajectory more accurately, relative distance in the earth-fixed frame is investigated and the final landing position is also plotted as shown in Fig. 4.3.




(a) Entire relative displacement (b) Final relative displacement

Figure 4.3: Relative displacement in time

The relative distance is calculated by "Ship position" - "Helicopter position" in the earth-fixed frame. $\Delta X$, $\Delta Y$, and $\Delta Z$ are relative distances along the earth-fixed X, Y, and Z-axis. In the following plots, the Z-axis component sign also follows a re-defined direction(positive upwards). The final landing circle on the flight deck means that landing anywhere inside the circle is safe. Thus, it can be considered as an allowable error range. Final values of $\Delta X$, $\Delta Y$ are considered as errors, but $\Delta Z$ is the summation of the distance from the landing gear to the CG (48.26 cm) and error. Hence, the final errors mean the deviation from the center of the circle and are expressed along each axis in meters (0.0353, 0.0728, 0.0037) as shown in Fig. 4.4.

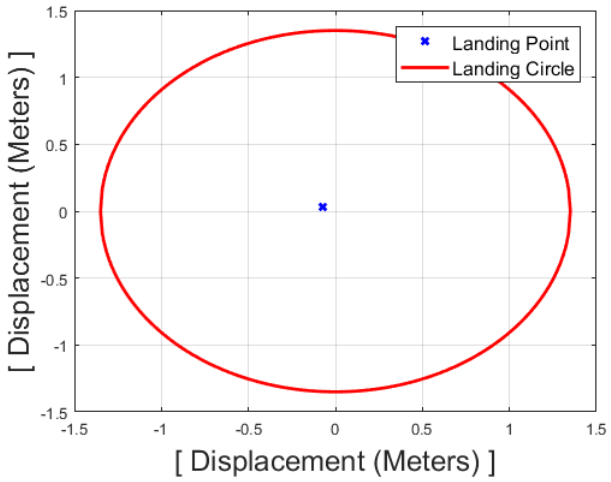
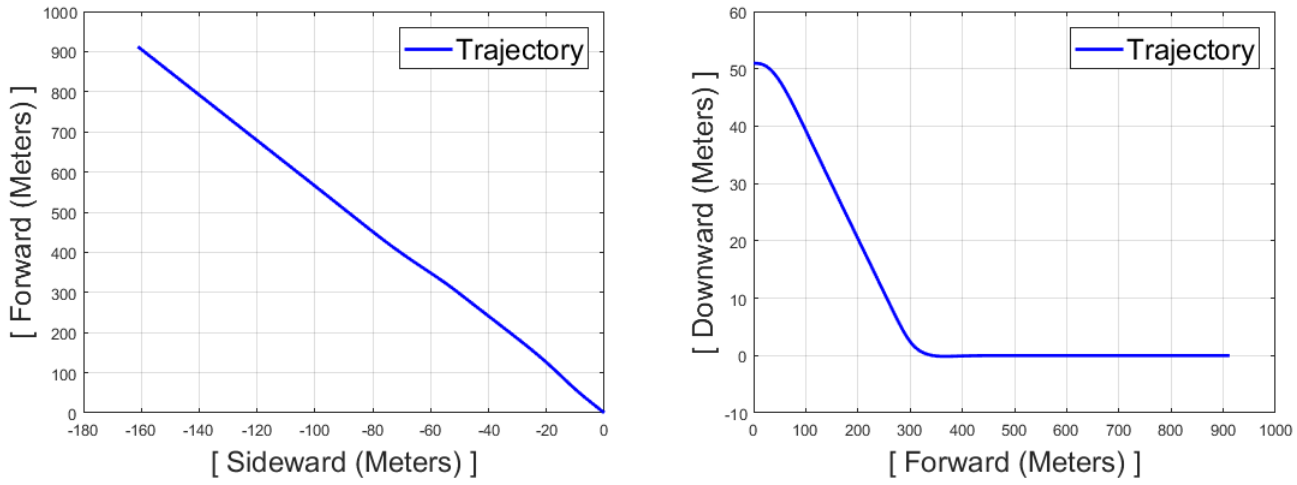



Figure 4.4: Final landing point

#### *4.1.2.1 Initial Descent*

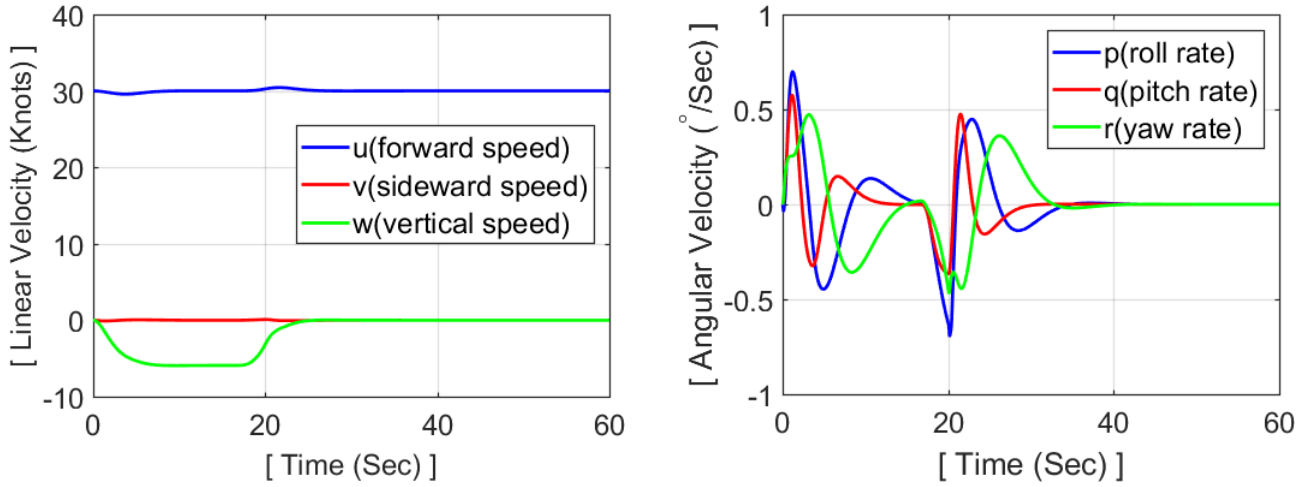
For the initial descent maneuver, the UH-60 linearized model, which is extracted at forward speed of 30 knots is used. It is controlled by LQR for set-point tracking to achieve desired states. In order to descend, non-zero setpoints for vertical speed *w* are assigned and the controller effectively regulates the error which is the difference between the setpoint and current state. Gains for the controller are determined by changing weights on Q and R matrices. Weights are carefully chosen since there is a trade-off between transient responses and control efforts. Therefore, it is

important to check if the control inputs are in the reasonable range. With the selected gains, it takes 41.68 seconds to regulate the errors in states to less than $1e^{-05}$ and the corresponding final altitude deviation is 0.03 cm. Trajectory for descent, fuselage states as a function of time, and relative control inputs (deviation from the trim control inputs) are shown in Figs. 4.5, 4.6, and 4.7.



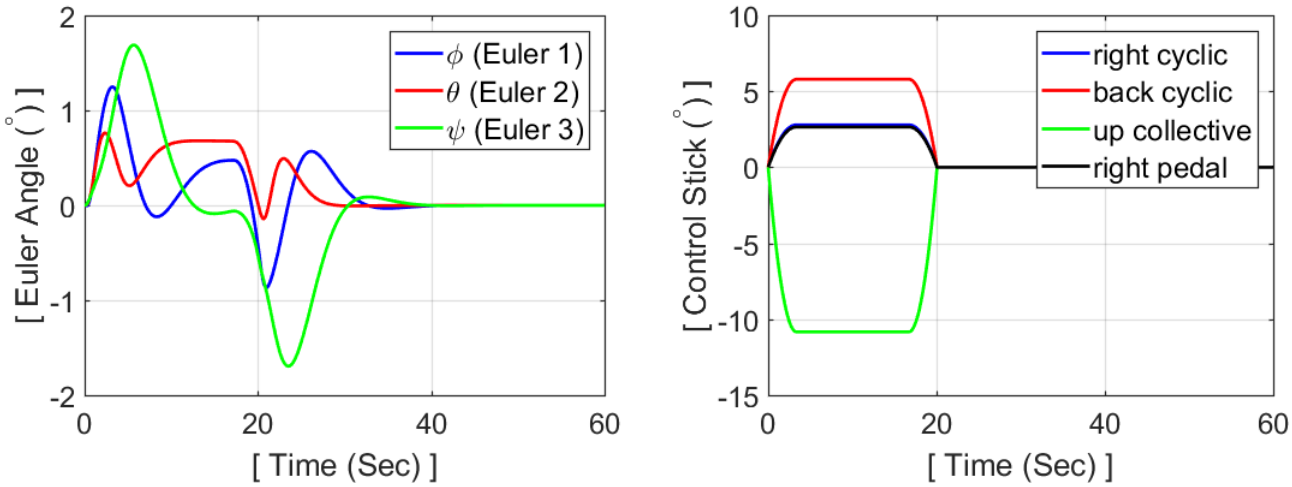

(a) Top view (b) Side view

Figure 4.5: Trajectory in descent maneuver




(a) Linear velocities along axes (b) Angular velocities about axes

Figure 4.6: Body-fixed linear and angular velocities in descent maneuver




(a) Helicopter attitude (Euler angles)

(b) Control inputs

Figure 4.7: Helicopter attitude and control inputs in descent maneuver

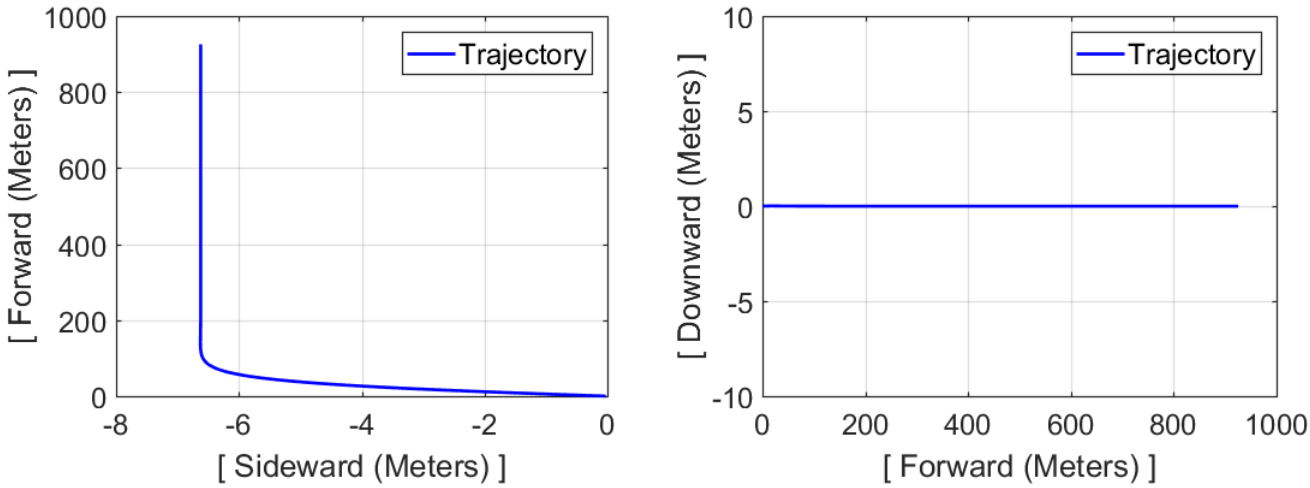
Considering the helicopter's relative position, it starts the next maneuver before the states are settled. In this case, it requires to begin the next maneuver (steady turn) at 30.14 seconds. Thus, the state values at 30.14 seconds are used as initial values for the next maneuver.

| | | |
|---|---|---|
| u = 15.4295 (m/s) | v = 0.0029 (m/s) | w = -0.0060 (m/s) |
| p = -0.0018 ($^\circ/s$) | q = -8.2214 ($^\circ/s$) | r = 0.0019 ($^\circ/s$) |
| $\phi$ = 0.0028 ($^\circ$) | $\theta$ = -9.0965 ($^\circ$) | $\psi$ = -0.0005 ($^\circ$) |

Table 4.1: State values at 30.14 seconds

#### *4.1.2.2 Steady Turn*

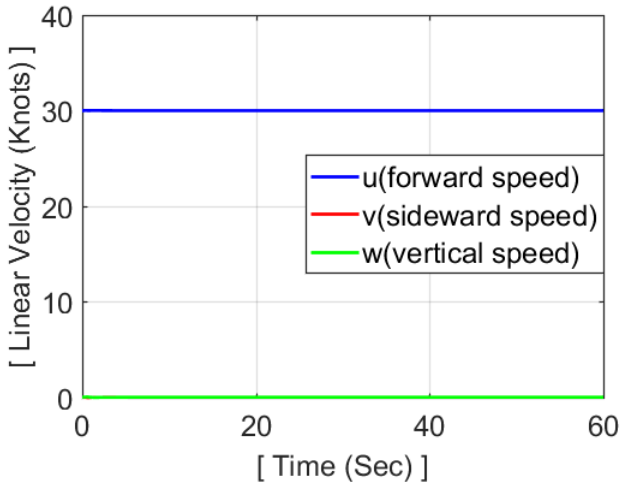
For the steady turn maneuver, the UH-60 linearized model, which is extracted at forward speed of 30 knots is used. It is controlled by LQR for set-point tracking to achieve desired states. In order to turn, non-zero setpoints for Euler angle $\psi$ are assigned and the controller effectively regulates the error which is the difference between the setpoint and current state. It takes 17.71 seconds to regulate the errors in states to less than $1e^{-05}$ and the corresponding final Euler angle deviation is $8.45e^{-07}(^{\circ})$, which is nearly zero. According to this simulation, the displacement along the earth-fixed Y-axis is computed as -6.68 meters. From this point, the lead time to start the steady turn maneuver can be calculated. Considering the lead time, the turning maneuver begins 30.14 seconds after the start of the descent. Corresponding steady turn trajectory, fuselage states as a function of time, and relative control stick inputs are shown in Figs. 4.8, 4.9, and 4.10.



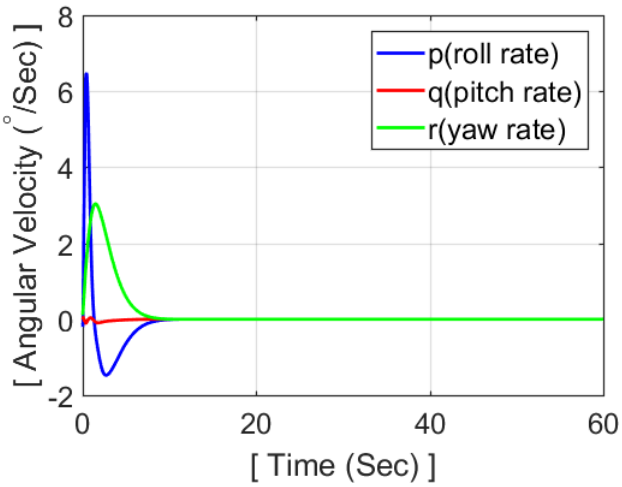

(a) Top view

(b) Side view

Figure 4.8: Trajectory in steady turn

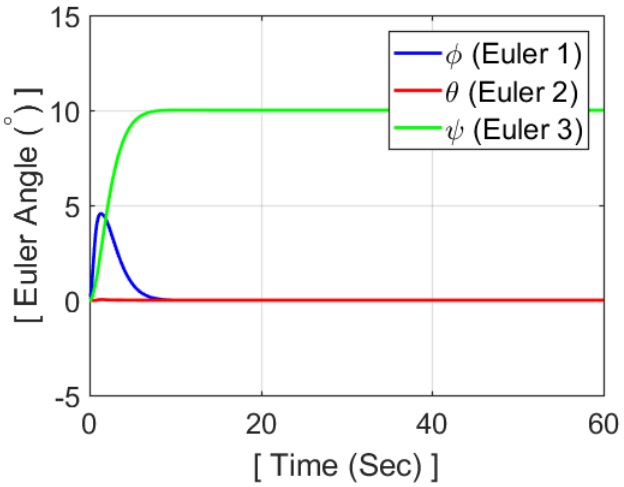



(a) Linear velocities along axes

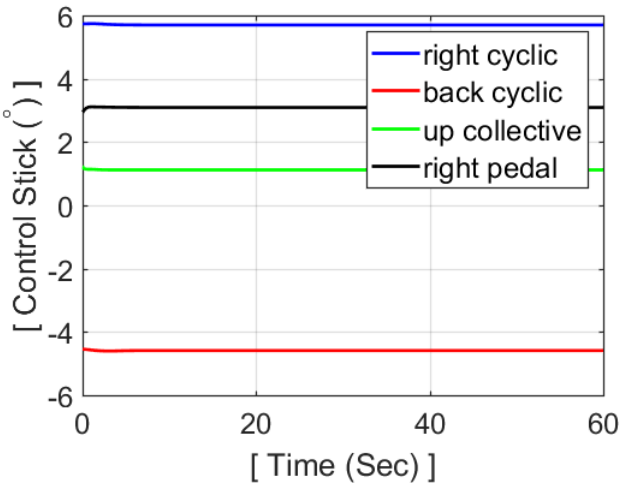



(b) Angular velocities about axes

Figure 4.9: Body-fixed linear and angular velocities in steady turn




(a) Helicopter attitude (Euler angles)



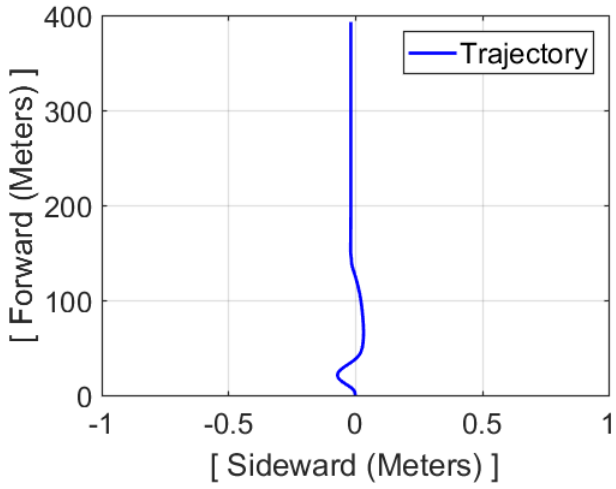

(b) Control inputs

Figure 4.10: Helicopter attitude and control inputs in steady turn

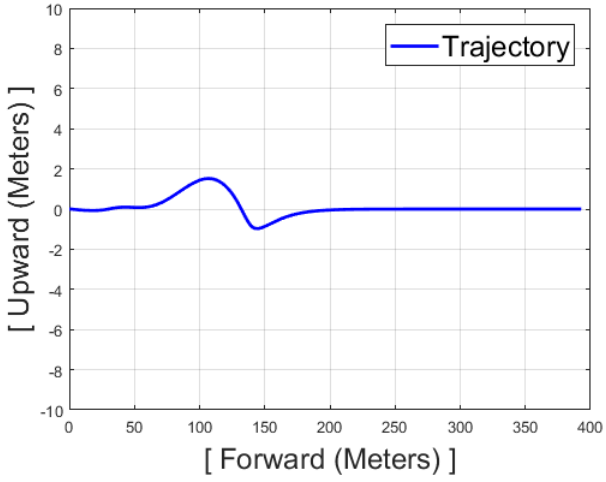
#### *4.1.2.3 Deceleration*

For the deceleration maneuver, the UH-60 linearized model which is extracted at forward speed of 10 knots is used. First, a 20 knots headwind is applied to simulate the deceleration from 30 knots to 10 knots. Although it has no problem regulating all states, the helicopter gains undesired altitude which is 14.78 meters. This occurs due to the way that a helicopter typically decelerates. In order to decelerate, the helicopter has to pitch nose up, which tilts the TPP and hence the thrust vector backwards, which creates a force in the backward direction. The nose-up motion can cause undesired altitude changes if the collective input is not controlled perfectly to maintain the altitude. Hence, besides assigning an initial value for forward speed *u*, non-zero setpoints for vertical speed $w$ are assigned to minimize undesired vertical altitude increase.

It is controlled by LQR for set-point tracking to regulate the errors, which are the differences between setpoints and the current states at each time step. It takes 37.08 seconds to regulate the errors in states to less than $1e^{-05}$. The final deviation in altitude is 0.024 cm and the lateral deviation is 1.69 cm. During this deceleration maneuver, the maximum altitude gain is 1.52 meters and the maximum altitude loss is 0.97 meters. Assigning non-zero setpoints for vertical speed *w* with appropriate LQR gains minimize this variation in altitude. To achieve the desired position for the helicopter when it completes deceleration, this deceleration maneuver begins 28.43 seconds after the start of the steady turn. Since the previous maneuver steady turn takes 17.71 seconds to achieve settlement in states, initial values for deceleration maneuver is set to zero other than forward speed *u* (which simulates deceleration from 30 knots to 10 knots). Corresponding deceleration trajectory, fuselage states as a function of time, and relative control stick inputs are shown in Figs. 4.11, 4.12, and 4.13.

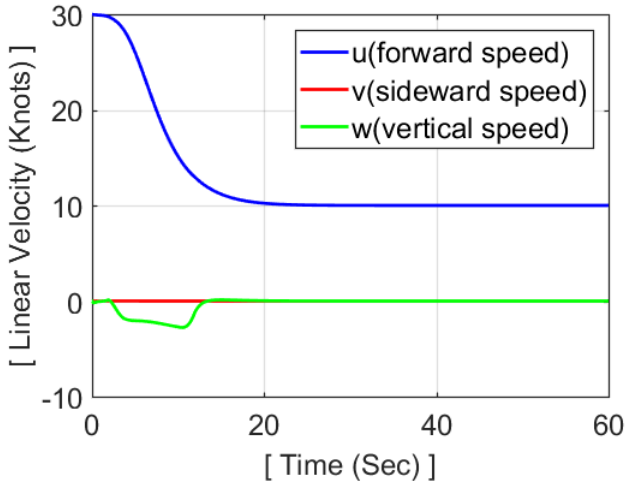



(a) Top view

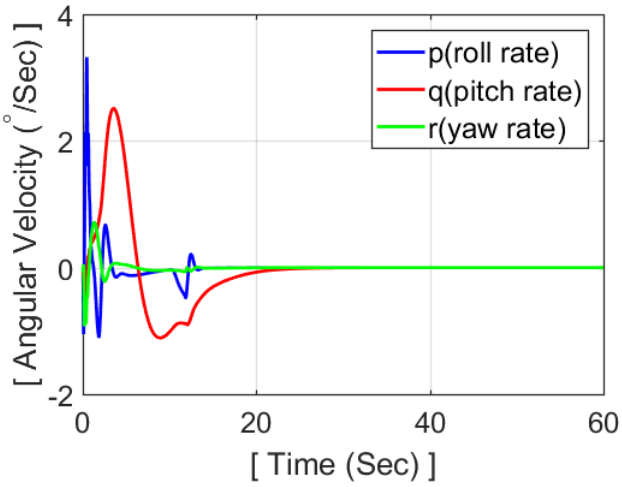



(b) Side view

Figure 4.11: Trajectory in deceleration maneuver

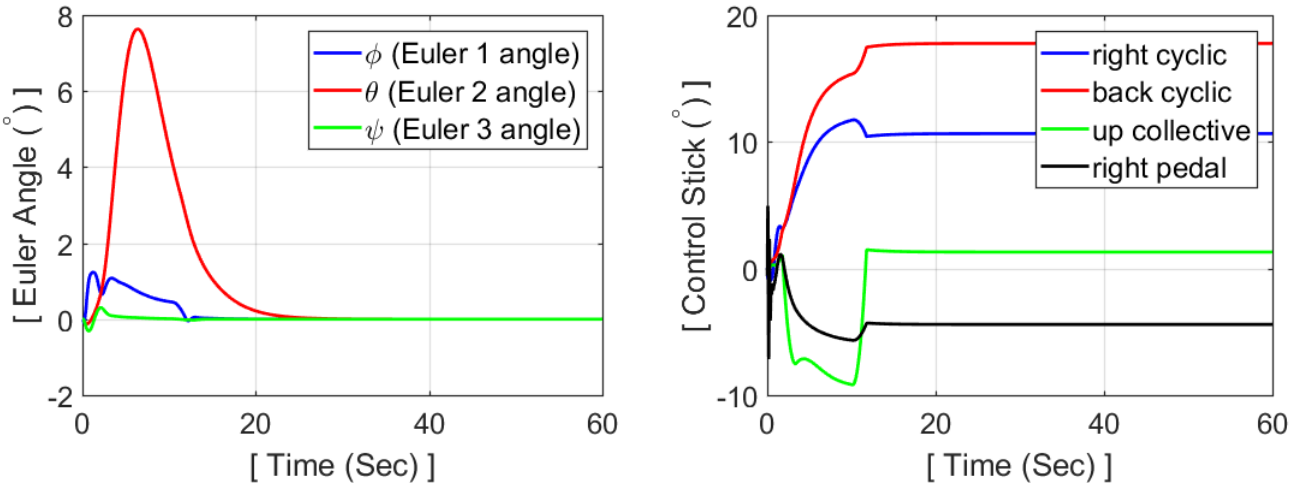



(a) Linear velocities along axes




(b) Angular velocities about axes

Figure 4.12: Body-fixed linear and angular velocities in deceleration maneuver



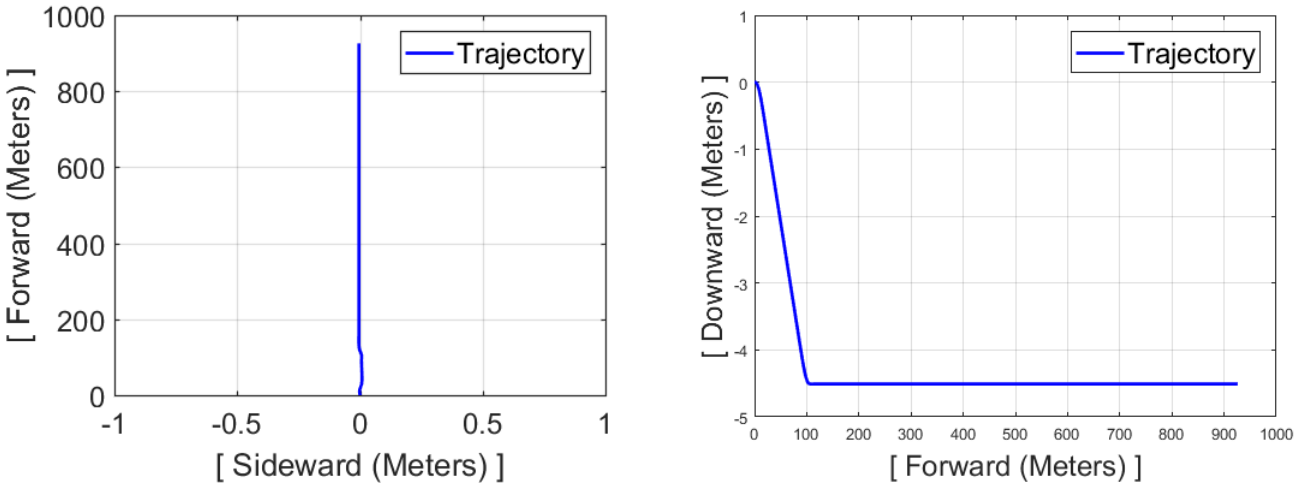

(a) Helicopter attitude (Euler angles)

(b) Control inputs

Figure 4.13: Helicopter attitude and control inputs in deceleration maneuver

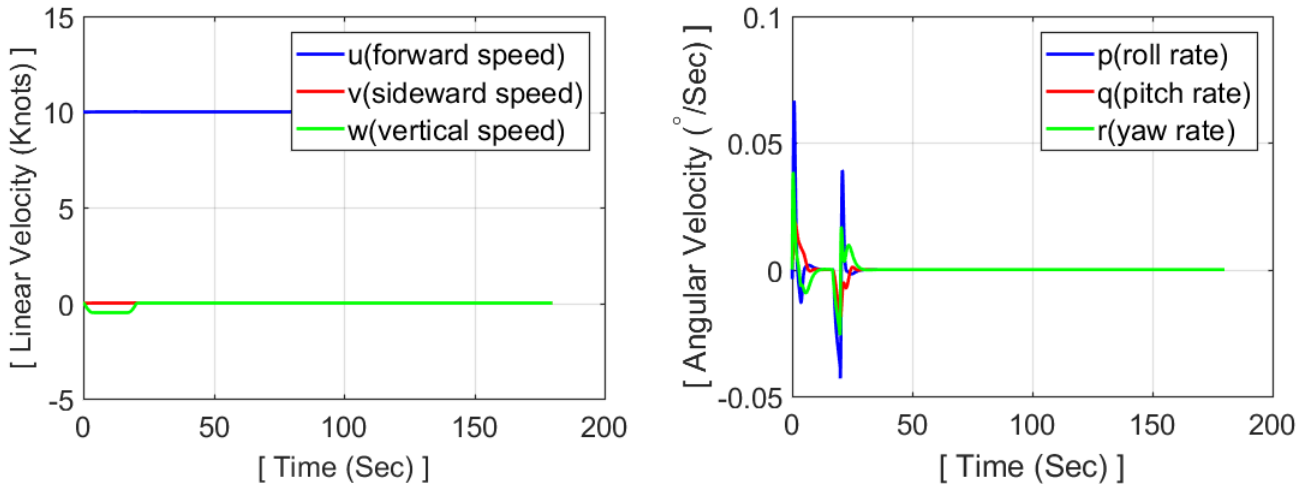
#### *4.1.2.4 Landing*

For the landing maneuver, the UH-60 linearized model, which is extracted at forward speed of 10 knots is used. It is controlled by LQR for set-point tracking to achieve the desired states. In order to land vertically, non-zero setpoints for vertical speed *w* are assigned and the controller effectively regulates the error which is the difference between the setpoint and current state. Although it takes 28.34 seconds to regulate the errors in states to less than $1e^{-05}$, state responses after the 20.41-second mark are meaningless since it touches down the deck at 20.41 seconds. In this simulation, the final landing maneuver begins 48.87 seconds after the start of deceleration. Landing gear position from the CG is considered and the final landing point deviation from the center of the landing circle is 0.035 meters along the earth-fixed x-axis and 0.073 meters along the earth-fixed Y-axis. Corresponding landing trajectory, fuselage states in time, and relative control stick inputs are identified as shown in Figs. 4.14, 4.15, and 4.16.



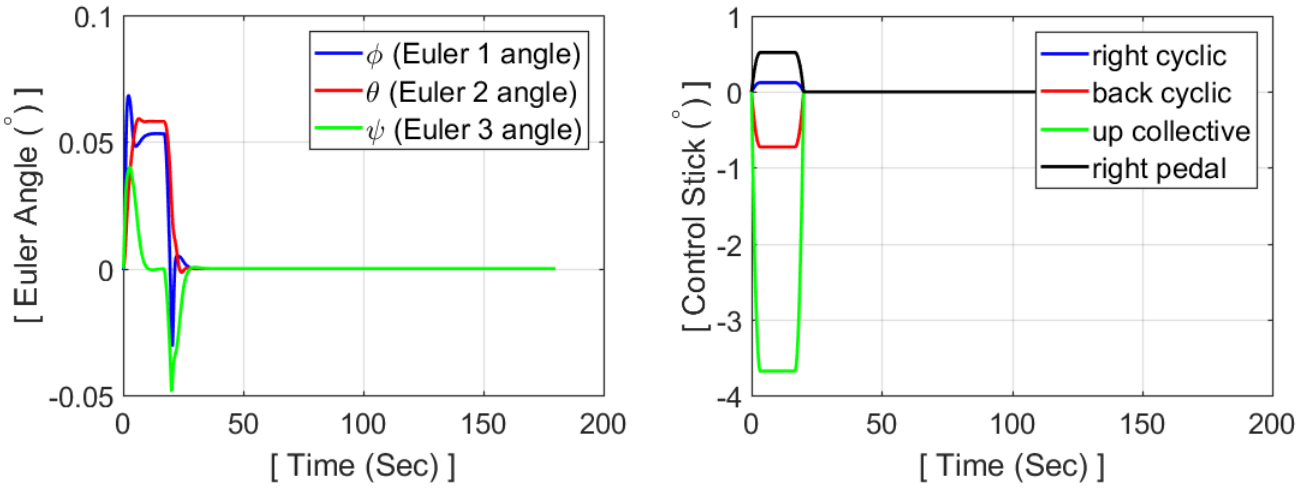

(a) Top view (b) Side view

Figure 4.14: Trajectory in landing maneuver




(a) Linear velocities along axes (b) Angular velocities about axes

Figure 4.15: Body-fixed linear and angular velocities in landing maneuver




(a) Helicopter attitude (Euler angles)

(b) Control inputs

Figure 4.16: Helicopter attitude and control inputs in landing maneuver

### 4.1.3 Visualization

Simulation results are visualized by using X-Plane 11 flight simulator software due to its excellent graphic quality and capability to implement a flight data recorder (FDR) file. It visualizes a UH-60 helicopter and a Navy ship realistically. Visualization through the FDR file is useful primarily in accident investigation and re-creation. In that case, it can be done by taking the data from the black box of an aircraft and put it in a format that X-Plane can load.

The mathematical simulation conducted in the MATLAB program stores the data, which is then re-written in the FDR format for visualization. It visualizes the values of fuselage attitude (roll, pitch, yaw) at each time step by taking those values directly from the simulation results. The helicopter position data at each time step are converted to latitude and longitude values, then they are visualized for the entire helicopter ship landing.

Figure 4.17: Visualized ship landing from initial position in cockpit view

Figure 4.18: Visualized ship landing at flight deck in outside view


Figure 4.19: Visualized ship landing under bad weather in tail view


Figure 4.20: Visualized ship landing with night vision goggle (NVG) at night in cockpit view

## 4.2 Gain-scheduled Control Strategy

As a first step towards implementing the novel autonomous ship landing method in practice, a gain-scheduled control system is designed to approach and land by tracking a checkerboard pattern visual cue installed on the ship platform as shown in Fig. 3.6 of Chapter 3. This vision-based control system forms the outer-loop of a cascaded feedback control system, where the inner-loop handles the basic attitude stability of the aircraft and yields the required pitch, roll, heave (throttle), and yaw control inputs as demanded by the outer loop. The standard form of the PID controller is represented in Eqn. 4.6.

$$u(t) = K_P e(t) + K_I \int_0^\tau e(\tau)d\tau + K_D \frac{d}{dt} e(t) \tag{4.6}$$

$e(t)$ denotes an error between a setpoint and relative position with respect to the visual cue. The positions and heading angle computed from the vision system reflect the disturbances and sensor noise. In this case, the update rate of the closed-loop feedback control system is 0.1 seconds including the live-streaming, detection, and estimation. $K_P$, $K_I$, and $K_D$ gains are tuned based on the effect of each parameter as shown in Table 4.2.

Table 4.2: Effect of $K_P$, $K_I$, and $K_D$

| Characteristic | Increase $K_P$ | Increase $K_I$ | Increase $K_D$ |
|---|---|---|---|
| Rise Time | Decrease | Small Decrease | Small Decrease |
| Overshoot | Increase | Increase | Decrease |
| Settling Time | Small Increase | Increase | Decrease |
| Steady-state Error | Decrease | Large Decrease | Minor Change |
| Stability | Degrade | Degrade | Improve |

The different sets of gains are scheduled based on relative positioning data to control the behavior of a UAV as desired. It allows the UAV to fly at high speed in the long-distance and then slow down the relative speed for cautious tracking when it gets closer to the landing point. The flight mode selector is also configured in order to adapt the UAV movements to the flight conditions and enhance safety. The states are the relative positions in three dimensions and the heading angle of the UAV with respect to the visual cue obtained from the computer vision system. The control inputs are pitch, roll, yaw, and throttle commands. Each control magnitude ranges from -100 to 100 as a percentage of the total input. The outputs are UAV attitude and raw images from the camera. The UAV attitude consists of roll angle $\phi$, pitch angle $\theta$, and yaw angle $\psi$.

### 4.2.1 Control Architecture

The vision-based PID control system with scheduled gains is structured as shown in Fig. 4.21. Setpoints are the desired relative positions and heading angle of the UAV. The mode selector automatically chooses a proper flight mode based on the flight conditions. Depending on the selected flight mode, the control inputs are obtained through the gain-scheduled PID controllers or from the pre-assigned values. The parameters that are fed back include the estimated UAV position and heading angle obtained from the computer vision system.

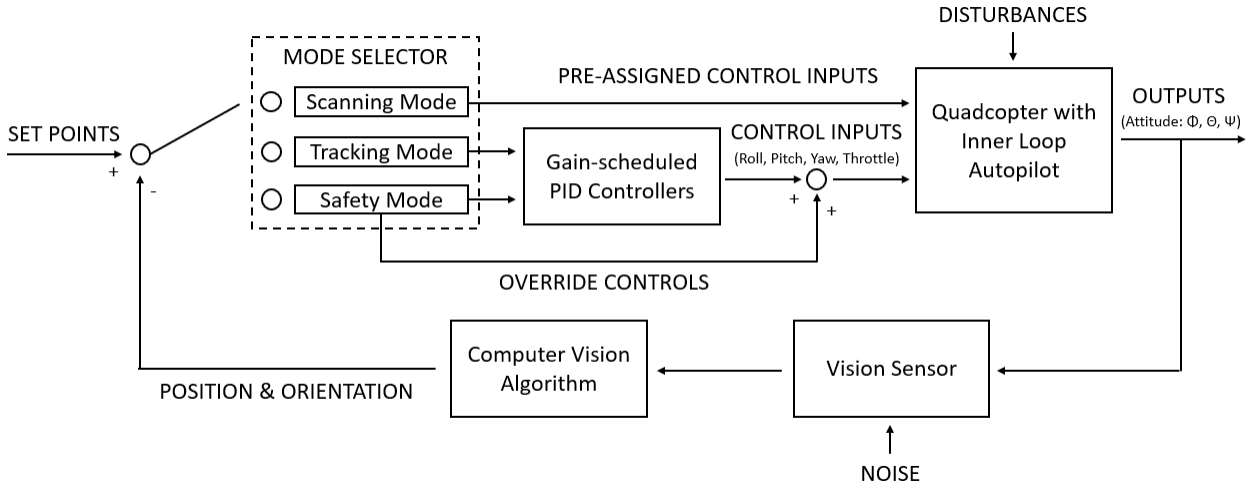



Figure 4.21: Gain-scheduled PID control system architecture

### 4.2.2 Flight Modes

A robust flight control system should be able to operate under various flight conditions. Thus, three flight modes are configured to fly the UAV in an appropriate manner given the scenario. They consist of the mode selector and gain-scheduled PID controllers in order to obtain the proper control inputs depending on the scenario.

#### *4.2.2.1 Scanning mode*

In the case of a visual cue detection failure, the scanning mode is activated anytime during the flight to search for the visual cue. Once it is engaged, the UAV starts yawing by commanding the pre-assigned control inputs instead of relying on the gain-scheduled PID controllers. It continues to yaw until it detects the visual cue and one successful detection disengages this mode instantly. This allows the UAV to track the visual cue irrespective of its initial heading orientation as long as the visual cue is within the detection range. During the flight, if the UAV loses the visual cue from the camera view, it stops moving and then scans for the cue in order to get back on track. Hence, the scanning mode enhances safety as well as increases operational range and robustness.

#### *4.2.2.2 Tracking mode*

This mode is activated once the visual cue is properly detected. Based on the relative distance and heading angle, different gain-scheduled PID controllers are engaged as shown in Fig. 4.22.

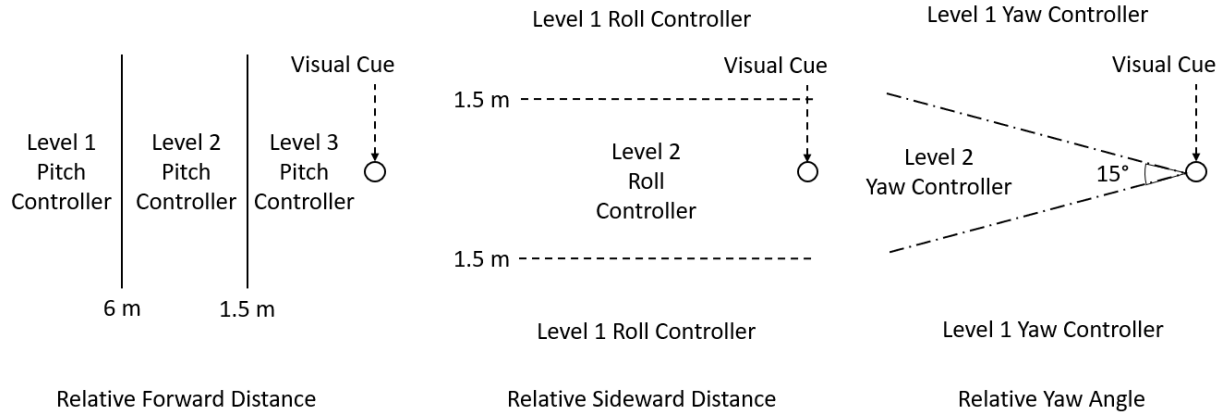



Figure 4.22: Composition of gain-scheduled PID controllers

First, three different pitch controllers are configured based on the forward relative distance. Level 1 pitch PID controller gains are scheduled to track the visual cue at the maximum speed when the distance to the landing spot is more than 6 meters. Level 2 pitch PID controller gains are scheduled to track the visual cue at moderate speeds when the distance range is between 6 and 1.5 meters. The controller is tuned to prevent overshoots and oscillations. Level 3 pitch PID controller gains are scheduled to track the visual cue within a range of 1.5 meters. It focuses on minimizing steady-state error for precise tracking and mimics the cautious final approach maneuver of a helicopter.

Second, two different roll controllers are configured based on the sideward relative distance. Level 1 roll controller gains are scheduled to track the visual cue when the sideward distance is more than 1.5 meters and is tuned to achieve fast tracking, no overshoots, and no oscillations. Level 2 roll controller gains are scheduled to track the visual cue precisely within the range of 1.5 meters focusing on steady-state error minimization.

Third, two different yaw controllers are configured based on the relative heading (yaw) angle to maintain the desired approach angle. In order to prevent the visual cue from getting out of the camera view by commanding large pitch and roll motions with excessive yaw angle, it gives priority to the yaw correction by reducing the magnitude of pitch and roll control inputs. Level 1 yaw controller is designed to slow down the pitch and roll motion as well as correct the yaw angle quickly when the relative yaw angle is more than 15 degrees. Level 2 yaw controller is designed to correct the yaw angle precisely without slowing down the pitch and roll motion when the relative yaw angle is less than 15 degrees. Lastly, one throttle PID controller is configured to control the heave motion based on the vertical relative distance. Considering the vertical displacement that the UAV can have within the camera view, one set of PID gains is enough to handle its heave motion.

#### *4.2.2.3 Safety mode*

This mode is activated to enhance safety by responding appropriately to three different situations shown in Fig. 4.23 and the safety mode supersedes any other mode.

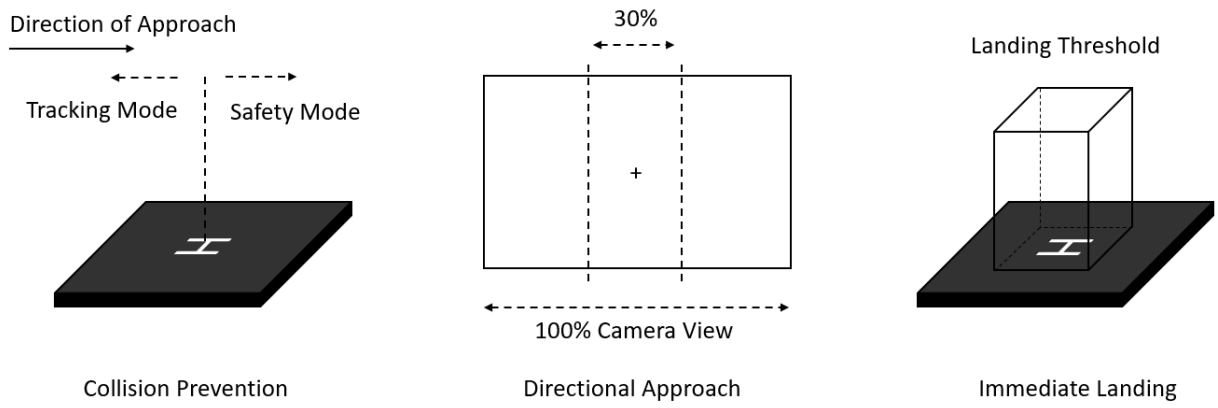



Figure 4.23: Schematic of safety mode

First, a backward (nose-up) pitch is applied instantly to prevent a collision when crossing the landing pad centerline instead of relying on the pitch PID controller. Second, the directional approach is designed to control the UAV when estimated relative positions are not reliable. Particularly, there exists a distance range where the forward distance estimation is accurate enough but the sideward distance and relative yaw angle estimation are not accurate due to a small number of pixels. In this case, it directly uses pixel position data in a 2D camera plane to control roll motion instead of estimating a position from an image. It does not correct for yaw angle but the roll is commanded if the visual cue is located in the area which is more than 15 percent offset from the center of the image in order to make sure the visual cue is not lost from the camera view. Third, an immediate landing is attempted to reduce the time for the final vertical movement by commanding throttle down instantly when the UAV enters into the landing threshold.

### 4.2.3 Simulation Results

The objective is to verify every aspect of the vision-based flight control system under different scenarios and to demonstrate that extreme accuracy can be achieved by adopting this novel method of tracking a visual cue installed parallel to a UAV approach course instead of tracking a landing platform. In the implementation, the entire coding is done in the Python programming language and a realistic robot simulation program, Gazebo [95], is used to simulate the physical and visual

surroundings of a UAV as shown in Fig. 4.24 and Video [96].

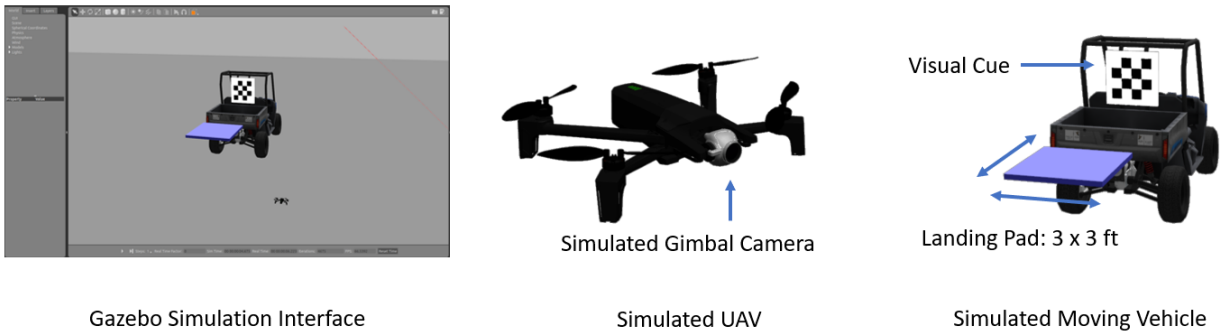



Figure 4.24: Simulated UAV and moving platform in Gazebo simulation program

The simulated UAV model has the same dimensions, mass moments of inertia, gimbal camera, and flight mechanism as the physical UAV, Parrot Anafi. The simulation also implements a moving vehicle that carries the visual cue and landing platform as shown in Fig. 4.24. As determined by vision system experiments, the visual cue has a checkerboard pattern consists of 4 x 4 squares with a side length of 120 mm each, and the landing platform size is 3 x 3 ft. These characteristics remain the same throughout simulations and flight tests.

The simulations are conducted in the same way as the actual flight tests including live-streaming, real-time vision processing, and feedback control loop. First, the simulated UAV is connected via WIFI to the base station computer. Once the UAV takes off, the images from the simulated camera are streamed to the computer. The images are processed one by one to obtain the relative position and heading information, which is utilized by the feedback controller running on the computer to generate the control inputs. Then, the computed control inputs are sent back to the UAV via the WIFI connection. This one cycle from streaming to sending commands back to the UAV takes 0.1 seconds. This simulated system, once verified, can be directly applied to a real flight test by simply switching an IP address from the simulated UAV to the physical UAV. The following simulation results show the tracking capability and landing accuracy in detail.

#### 4.2.3.1 *Landing Accuracy*

To demonstrate the tracking and landing accuracy of the present vertical-visual-cue based method, multiple simulations are conducted with different landing thresholds and varying the speed of the moving platform. The final landing locations from 100 independent simulations are plotted in Fig. 4.25.

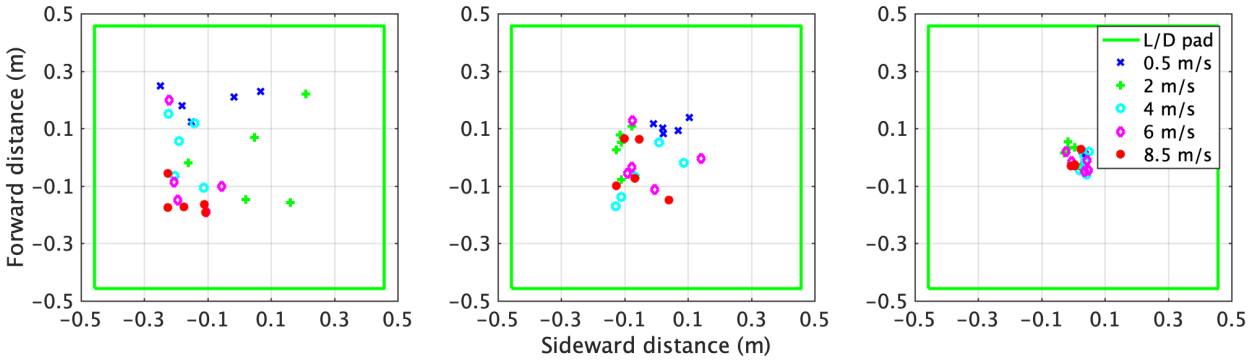



Figure 4.25: Landing points with landing threshold of 25 x 25cm (left), 15 x 15cm (center), and 5 x 5 cm (right)

The green line depicts the landing pad and each mark denotes the landing point for different speeds of the moving platform. Note that landing anywhere within the 25 x 25 cm area can be regarded as a safe landing considering the UAV size. The present vision-based control system achieves precise landing within a 5 x 5 cm landing threshold for all platform speeds. When the platform is moving at 8.5 m/s, it requires the UAV to fly close to its maximum speed but it still can make accurate landings. The dispersion of landing points in the figure can be attributed to the assigned landing threshold.

To investigate the effect of landing thresholds and platform speeds on tracking time, various cases are simulated and the time history of relative distance for each landing case is shown in Fig. 4.26.

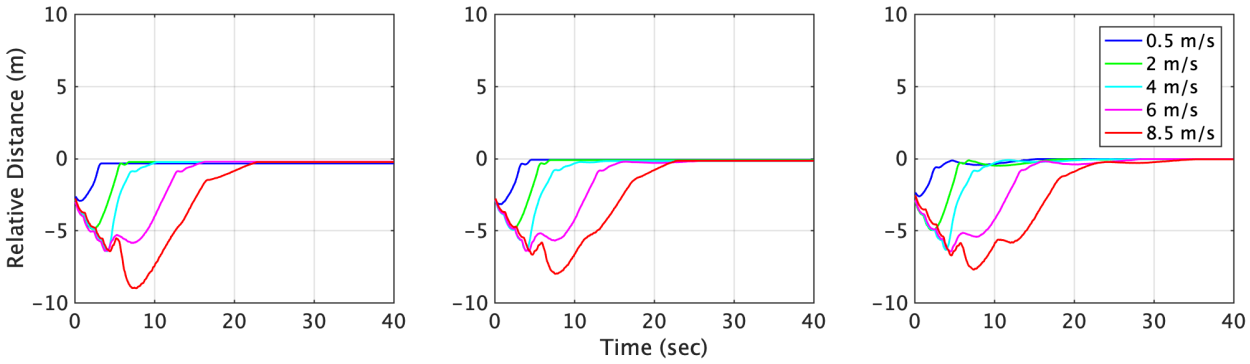



Figure 4.26: Relative distance with landing threshold of 25 x 25cm (left), 15 x 15cm (center), and 5 x 5 cm (right)

Each line denotes the relative distance between the UAV and landing pad center at different platform speeds. By the time the UAV starts tracking after take-off, the platform is already accelerating forward up to its designated speed. Thus, the relative distance increases at the beginning of the tracking phase. Also, it takes more time to land as the landing threshold is made smaller. The average time to achieve a landing threshold in each case is specified in Table 4.3.

Table 4.3: Average time to achieve landing threshold

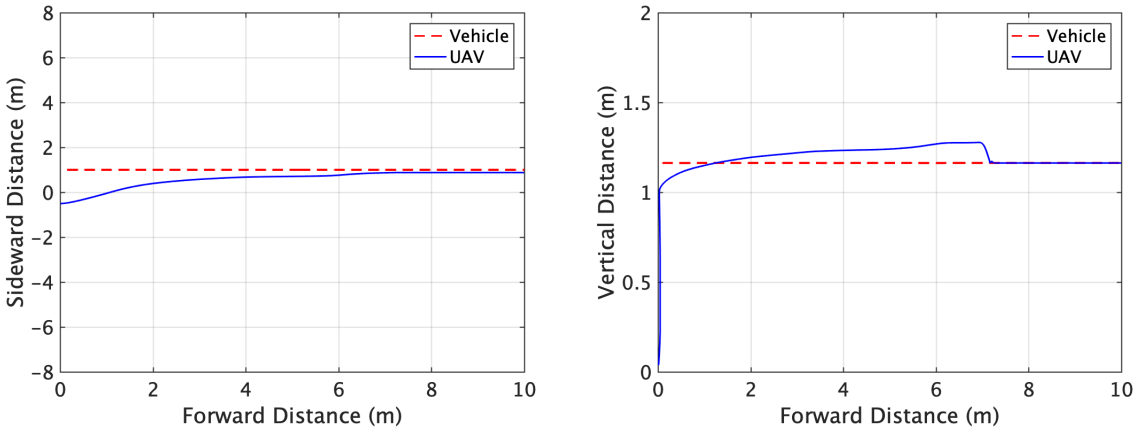
| Vehicle Speed | 5x5cm | 15x15cm | 25x25cm |
|---|---|---|---|
| 0.5m/s | 15.3s | 4.6s | 3.3s |
| 2.0m/s | 25.6s | 7.0s | 6.8s |
| 4.0m/s | 26.9s | 13.8s | 10.3s |
| 6.0m/s | 28.6s | 16.4s | 16.2s |
| 8.5m/s | 35.2s | 23.0s | 22.9s |

In conclusion, the landing accuracy depends on an assigned landing threshold. The present system can achieve a precise landing, however, there is a trade-off between the time and landing accuracy. Thus, the landing threshold should be selected based on the priority of a mission, which could be accuracy or time. In the following simulations, the medium landing threshold of 15 x 15 cm is applied.

#### 4.2.3.2 *Case-1: Platform moving in a straight line at 0.5 m/s*

To examine the tracking capability, simulations are conducted by varying the speed and the course of the moving platform. The four cases discussed in this paper include the platform moving forward in a straight line at speeds 0.5 m/s and 8.5 m/s as well as the platform moving in an S-pattern and a circular pattern. The trajectories, final landing points, set-point tracking data in time, and time history of control inputs are analyzed for each case.

First, the platform is programmed to move forward at 0.5 m/s speed and the trajectories viewed from the top and side are shown in Fig. 4.27.

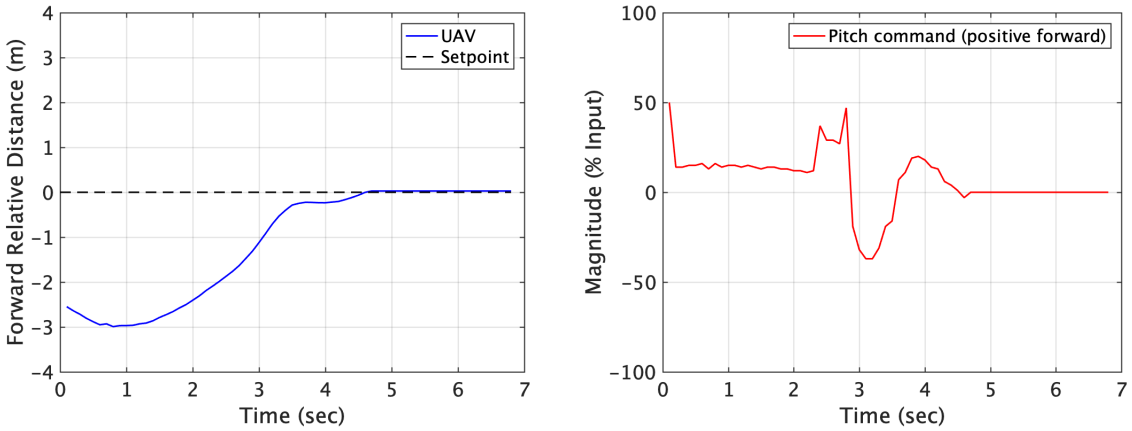



(a) Top view (b) Side view

Figure 4.27: Vehicle and UAV trajectories in case-1

The solid blue line denotes the UAV trajectory whereas the dashed red line denotes the platform trajectory from take-off to landing. The initial position of the UAV is on the ground at a sideward distance of 1.5 meters from the platform. The UAV takes off and then tracks the visual cue until satisfying the landing condition. Once the UAV enters the 15 x 15 cm landing space from the pad center, it lands quickly.

The forward relative distance and pitch command input in time are shown in Fig. 4.28.



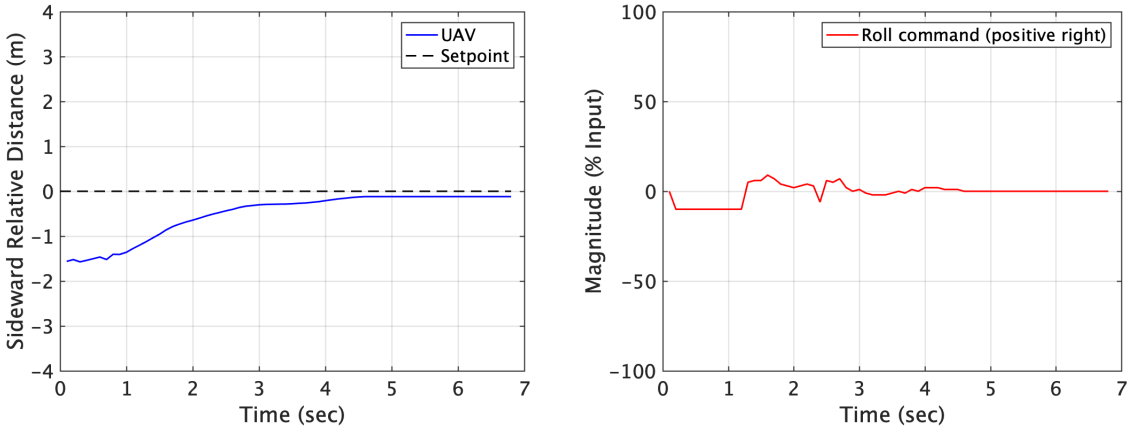

(a) Forward relative distance

(b) Pitch command inputs

Figure 4.28: Forward relative distance and pitch command in case-1

The blue and red solid lines denote the time history of the forward relative distance and pitch command input, respectively. The pitch command input ranges from -100 to 100 as percentage input where -100 is the maximum pitch backward and 100 is the maximum pitch forward. Initially, the distance increases since the UAV has to pick up speed from 0 m/s speed while the platform is already moving forward at 0.5 m/s. Due to the different levels of gain-scheduled PID controllers activated based on the relative distance, the tracking rate changes from fast-tracking at a large distance to precise tracking in the proximity of the landing pad. It takes 4.4 seconds to reach the landing threshold of 15 cm from the landing pad center and the final landing deviation is 2 cm.

The sideward relative distance and roll command input in time are shown in Fig. 4.29. The blue and red solid lines denote the time history of sideward relative distance and roll command input, respectively. The roll command input ranges from -100 to 100 as percentage input where -100 is maximum roll left and 100 is maximum roll right.

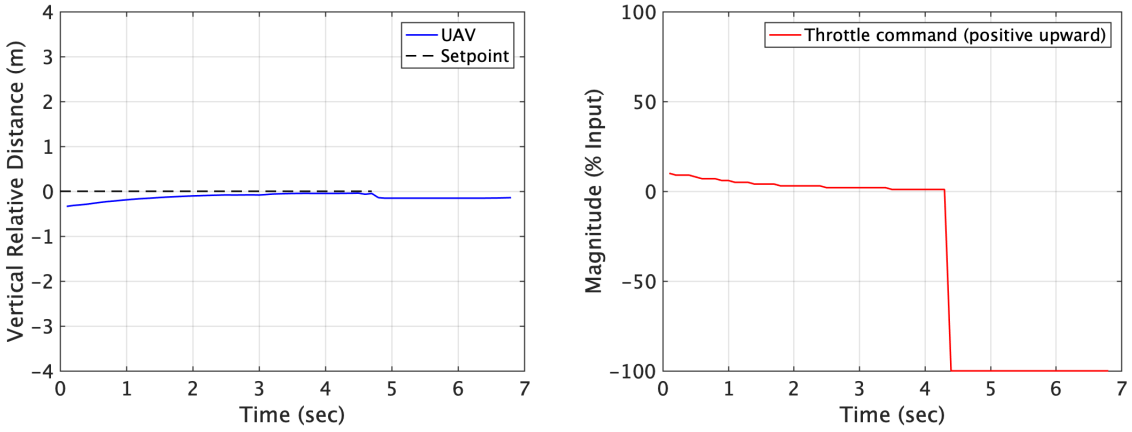



(a) Sideward relative distance (b) Roll command inputs

Figure 4.29: Sideward relative distance and roll command in case-1

For an initial 1 second, the roll is commanded based on the visual cue position in a 2D camera view, which is the directional approach of safety mode. It prevents responding to the noisy sideward distance estimates that happen while the moving average method is not activated. It takes 10 previous data (1 second) to activate the moving average method. The final landing deviation in the sideward direction is 11 cm.

The vertical relative distance and throttle command input in time are shown in Fig. 4.30.

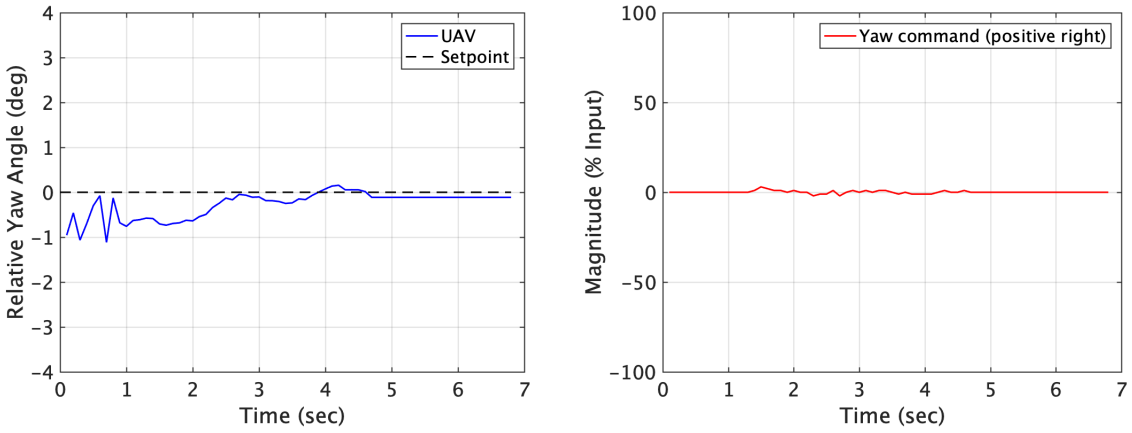



(a) Vertical relative distance (b) Heave(throttle) command inputs

Figure 4.30: Vertical relative distance and heave(throttle) command in case-1

The blue and red solid lines denote the time history of vertical relative distance and throttle command input, respectively. The zero setpoint means 15 cm above the landing pad. The throttle command input ranges from -100 to 100 as percentage input where -100 is maximum throttle down and 100 is maximum throttle up. At 4.4 seconds, the immediate landing is conducted by applying the maximum throttle down command.

The relative yaw angle and yaw command input in time are shown in Fig. 4.31.




(a) Relative yaw angle

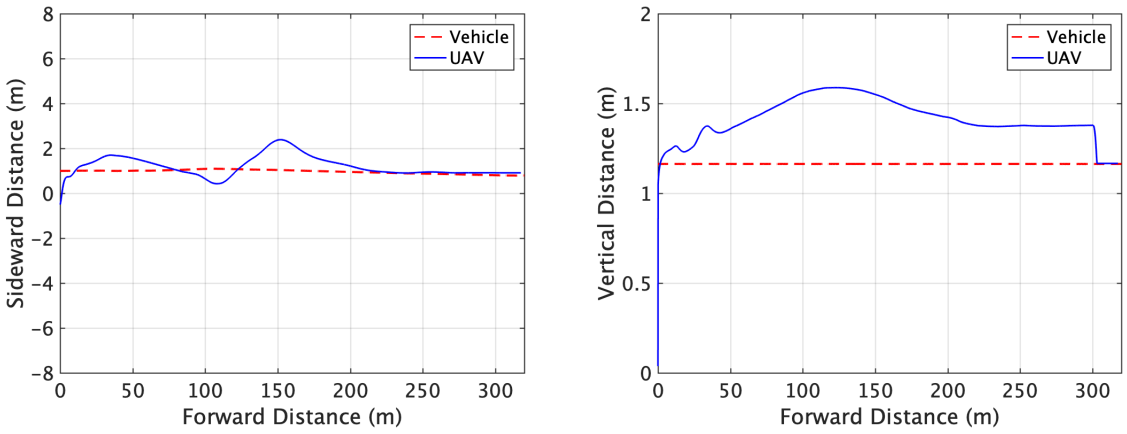
(b) Yaw command inputs

Figure 4.31: Relative yaw angle and yaw command in case-1

The blue and red solid lines denote the time history of relative yaw angle and yaw command input, respectively. The zero setpoint means that a heading angle is set to zero for making a parallel approach. The yaw command input ranges from -100 to 100 as percentage input where -100 is the maximum yaw left and 100 is the maximum yaw right. As in the case of the roll, it takes one second for the moving average method to be active. Therefore, instead of responding to initial fluctuating yaw angle estimations, the directional approach of safety mode commands no yaw for the first one second. Then, it starts regulating the yaw angle by the yaw PID controller once the moving average method is active.

#### 4.2.3.3 *Case-2: Platform moving in a straight line at 8.5 m/s*

The platform is programmed to move forward at 8.5 m/s, which requires the UAV to fly at its maximum speed during tracking. Therefore, it occasionally experiences control saturation; however, it still makes a successful approach and landing. The trajectories viewed from the top and side are shown in Fig. 4.32.



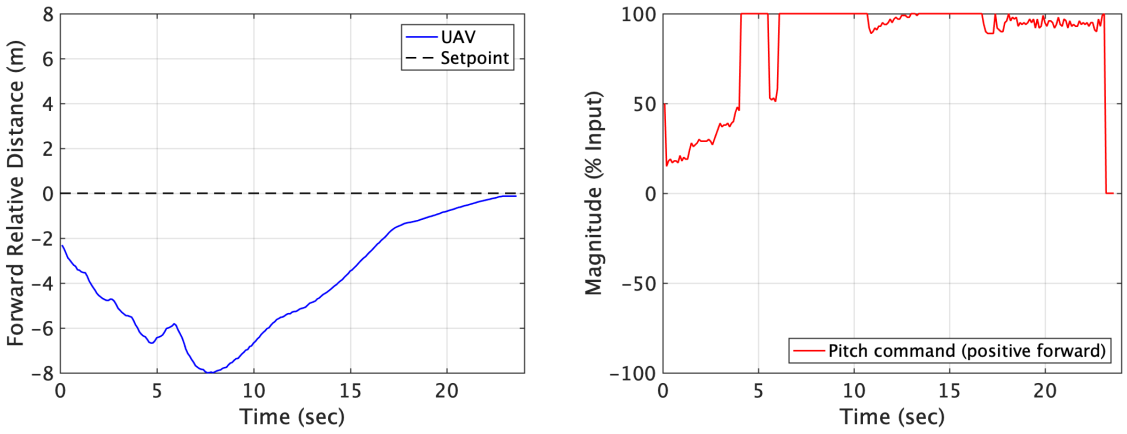

(a) Top view (b) Side view

Figure 4.32: Vehicle and UAV trajectories in case-2

The solid blue line denotes the UAV trajectory whereas the dashed red line denotes the platform trajectory from take-off to landing. The overshoots and undershoots observed in the trajectories are due to the control saturation when the UAV flies with its maximum pitch forward command which limits other commands.

The forward relative distance and pitch command input in time are shown in Fig. 4.33. Up to 7.5 seconds, the relative distance increases since the platform is moving faster than the UAV. When the relative distance is further than 6 meters, the level 1 pitch controller yields maximum forward pitch for high-speed tracking. Since UAV control inputs are achieved through propeller speed variation, the maximum forward pitch that requires the highest rotating speed for the two rear propellers affects the capability to yield the other control inputs as desired. Level 2 and 3 pitch

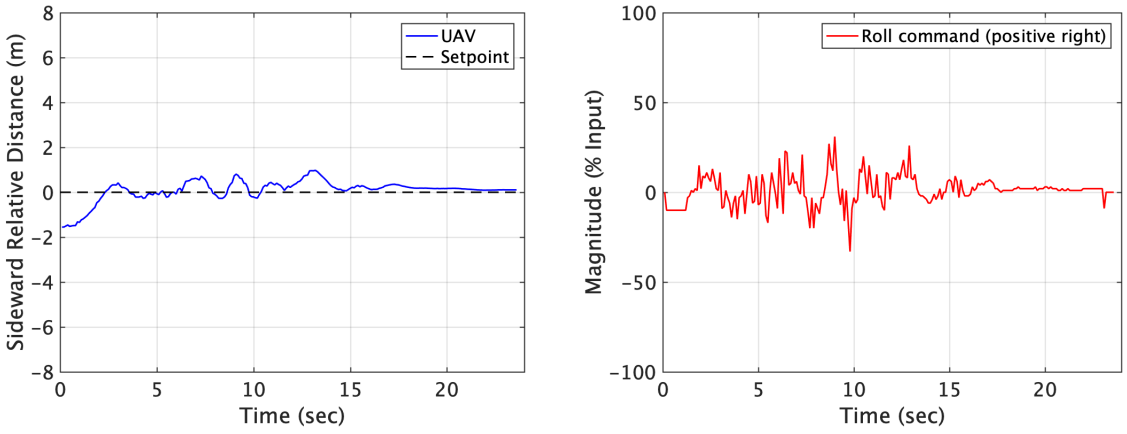
controllers are activated at distances of 6 and 1.5 meters, respectively. It takes 23 seconds to reach the landing threshold of 15 cm from the pad center and the final landing deviation in the forward direction is 12 cm.




(a) Forward relative distance (b) Pitch command inputs

Figure 4.33: Forward relative distance and pitch command in case-2

The sideward relative distance and roll command input in time are shown in Fig. 4.34.

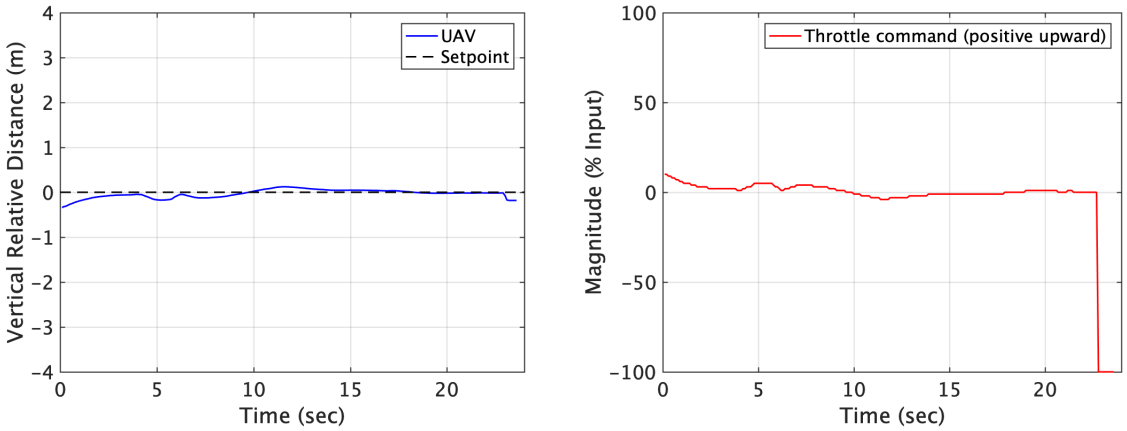



(a) Sideward relative distance (b) Roll command inputs

Figure 4.34: Sideward relative distance and roll command in case-2

Initially, the directional approach of safety mode is activated to command roll with a magnitude of 10 based on the visual cue position in a 2D camera view, and the level 2 roll controller takes over the control after 1.1 seconds. Between 4 and 17 seconds, it experiences limited control authority caused by maximum pitch forward commands. Thus, the relative distance is fully regulated after 17 seconds. The final sideward landing deviation is 11 cm.

The vertical relative distance and throttle command input in time are shown in Fig. 4.35.



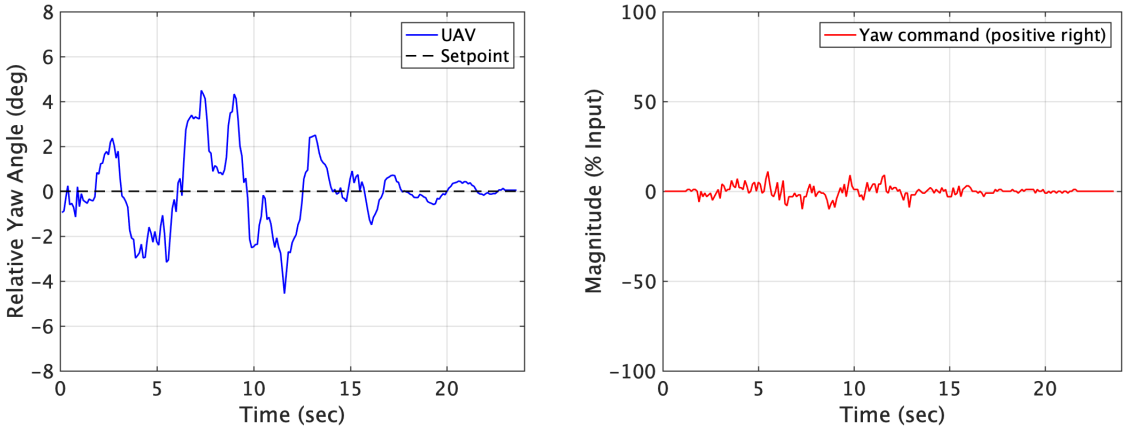

(a) Vertical relative distance

(b) Heave(throttle) command inputs

Figure 4.35: Vertical relative distance and heave(throttle) command in case-2

It shows the time history of vertical relative distance with respect to the setpoint which is 15 cm above the landing pad as well as throttle command input after take-off. At 23 seconds, the immediate landing is executed by applying the maximum throttle down command.

The relative yaw angle and yaw command input in time are shown in Fig. 4.36. After the directional approach of safety mode commands no yaw for an initial 1 second, it starts regulating the yaw angle by the yaw PID controller with the moving average method using 10 previous data which allows stable tracking. However, it experiences limited control authority due to maximum pitch forward commands between 4 and 17 seconds. Therefore, the yaw angle is fully regulated after 17 seconds.

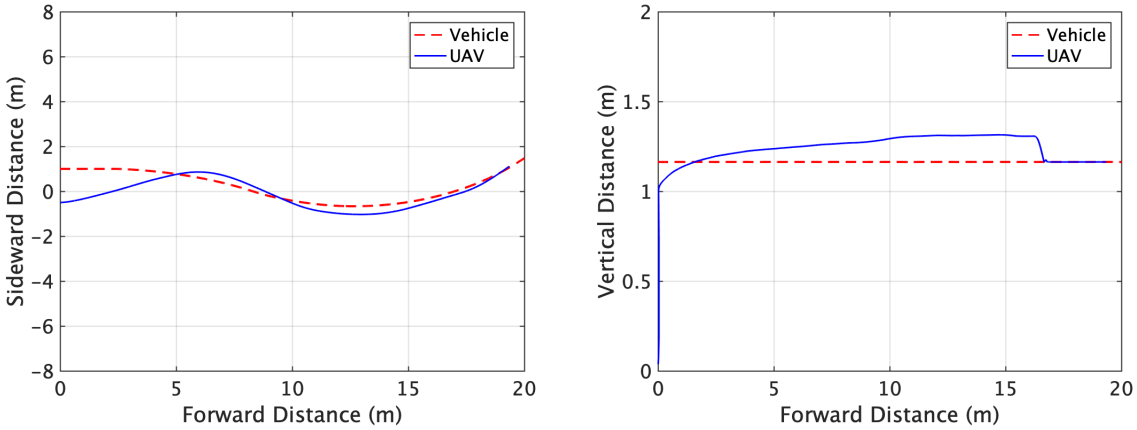



(a) Relative yaw angle (b) Yaw command inputs

Figure 4.36: Relative yaw angle and yaw command in case-2

#### *4.2.3.4 Case-3: Platform moving in S-pattern*

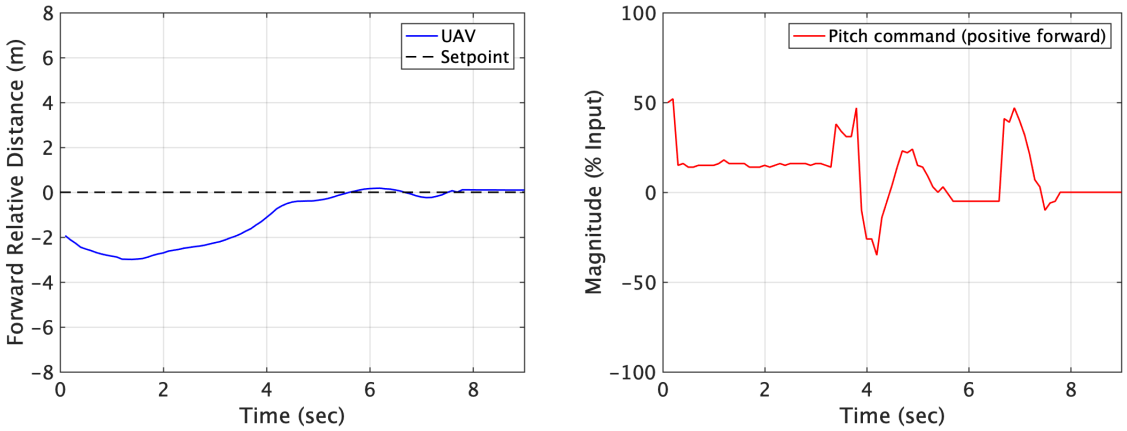
The platform is programmed to move in an S-pattern at a constant speed of 1 m/s and the trajectories viewed from the top and side have minimal overshoot as shown in Fig. 4.37. It shows smooth tracking capability with minimum overshoot while the platform is moving in S-pattern.




(a) Top view (b) Side view

Figure 4.37: Vehicle and UAV trajectories in case-3

The forward relative distance and pitch command input in time are shown in Fig. 4.38.

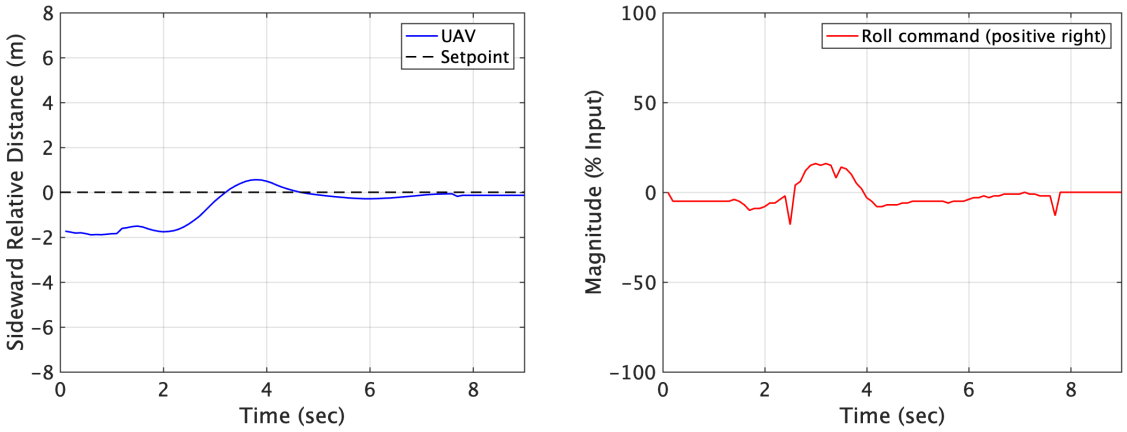



(a) Forward relative distance (b) Pitch command inputs

Figure 4.38: Forward relative distance and pitch command in case-3

The pitch controller is switched from level 2 to 3 upon reaching the distance of 1.5 meters at 3.7 seconds and the final forward landing deviation is 9 cm.

The sideward relative distance and roll command input in time are shown in Fig. 4.39.

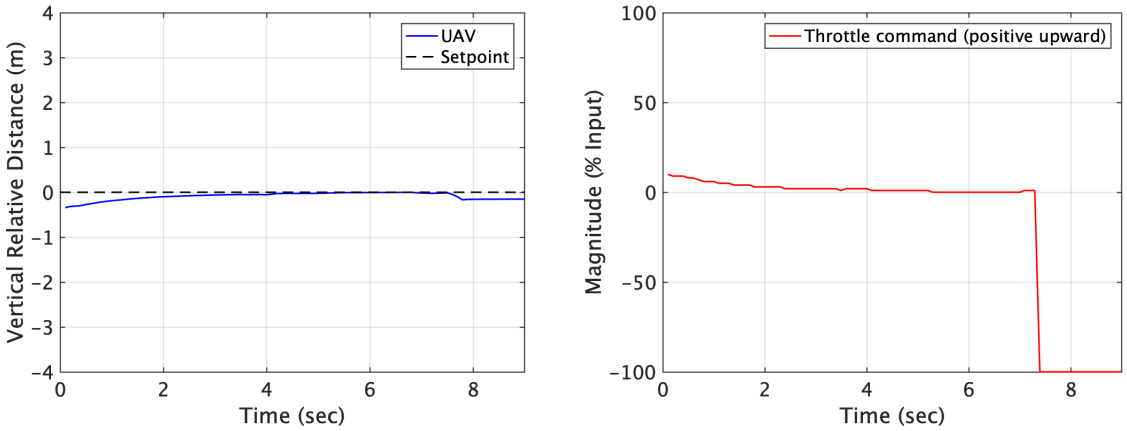



(a) Sideward relative distance (b) Roll command inputs

Figure 4.39: Sideward relative distance and roll command in case-3

The directional approach of safety mode is activated for an initial 1 second followed by the level 1 roll controller. At 2.5 seconds, the level 2 controller is activated for precise tracking and the sideward landing deviation is 13 cm.

The vertical relative distance and throttle command input in time are shown in Fig. 4.40.



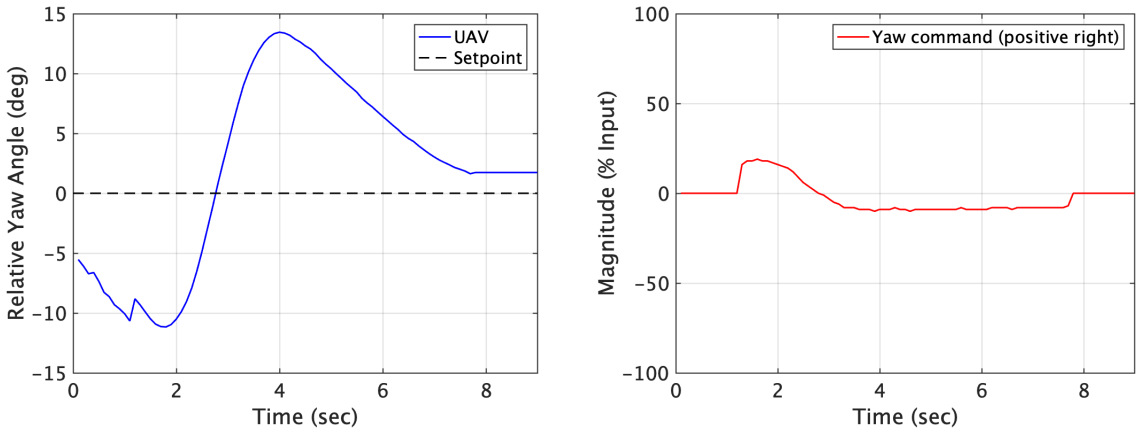

(a) Vertical relative distance

(b) Heave(throttle) command inputs

Figure 4.40: Vertical relative distance and heave(throttle) command in case-3

The zero setpoint is 15 cm above the landing pad and it commands maximum throttle down to land immediately at 7.8 seconds where the landing conditions are satisfied.

The relative yaw angle and yaw command input in time are shown in Fig. 4.41.




(a) Relative yaw angle

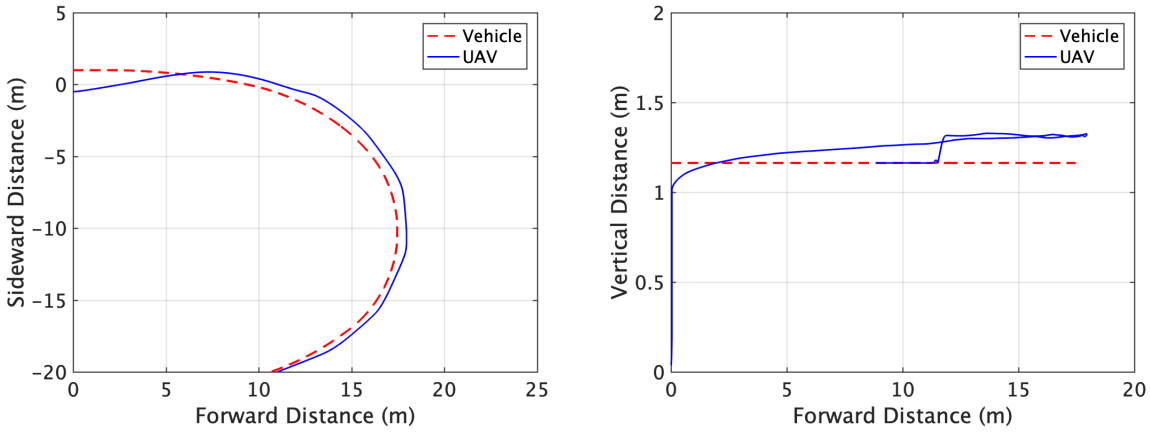
(b) Yaw command inputs

Figure 4.41: Relative yaw distance and yaw command in case-3

Due to the platform changing its course, the yaw angle is relatively big. However, the level 2 yaw controller regulates the yaw angle effectively. Since the yaw angle is not one of the landing conditions, the UAV lands with a yaw angle of 1.7 degrees.

#### *4.2.3.5 Case-4: Platform moving in circular pattern*

The platform is programmed to move in a circular pattern at a speed of 1 m/s and the trajectories viewed from the top and side are shown in Fig. 4.42.

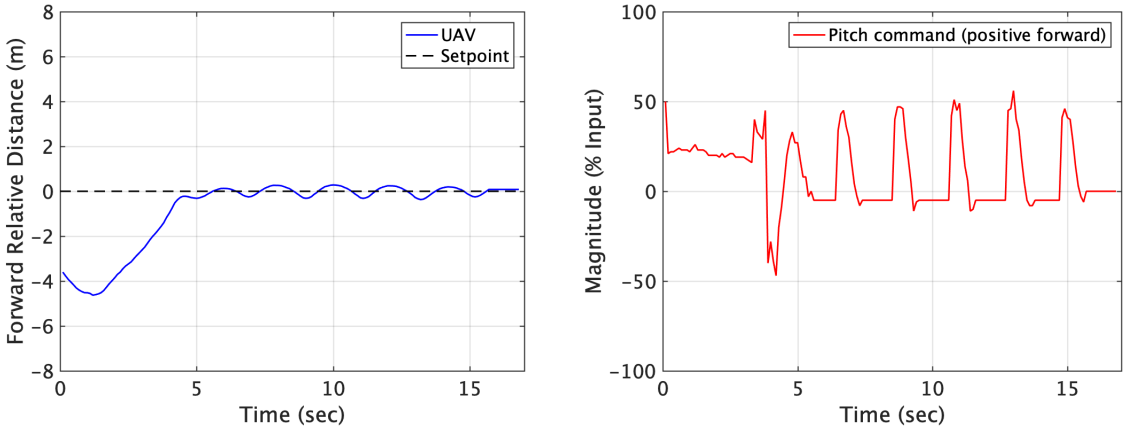



(a) Top view (b) Side view

Figure 4.42: Vehicle and UAV trajectories in case-4

Since the platform turns more than 90 degrees, the trajectory in the side view overlaps as the platform comes back. The entire trajectory shows that the UAV tracks the circling platform with minimal overshoot.

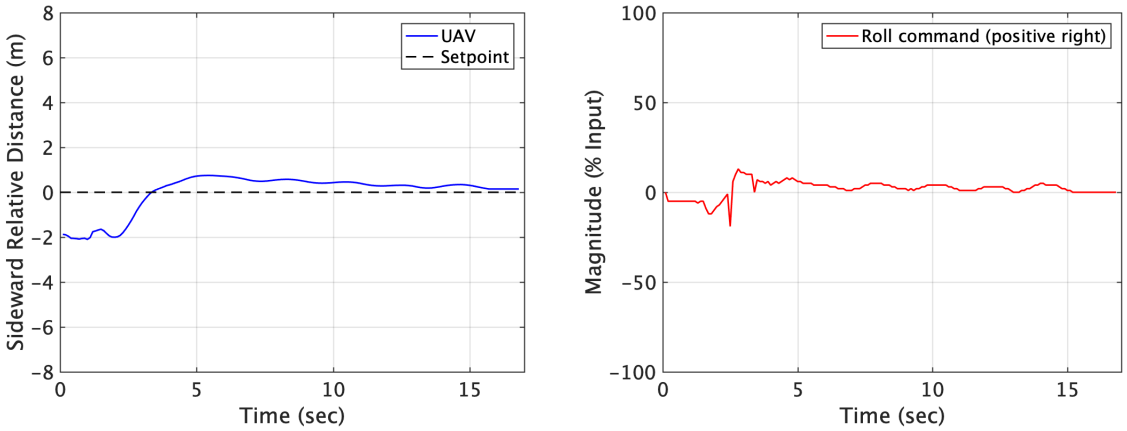
The forward relative distance and pitch command input in time are shown in Fig. 4.43. The level 2 pitch controller operates until 3.8 seconds followed by the level 3 controller. While tracking, the safety mode is intermittently activated to prevent a potential collision with the platform structure, which commands -10 pitch input if the UAV crosses the pad center. It lands with a forward deviation of 8 cm.




(a) Forward relative distance (b) Pitch command inputs

Figure 4.43: Forward relative distance and pitch command in case-4

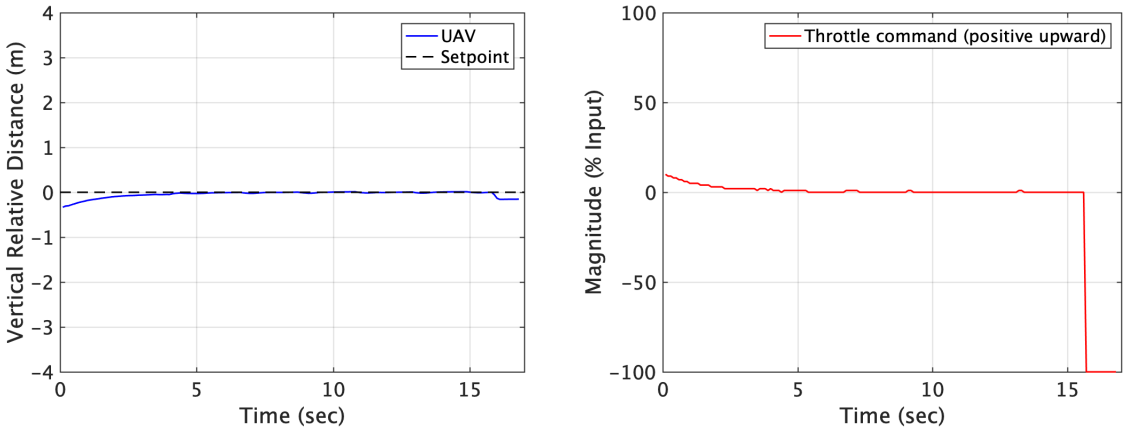
The sideward relative distance and roll command input in time are shown in Fig. 4.44. The directional approach of safety mode is activated for an initial 1 second followed by the level 1 controller. At 2.5 seconds, the level 2 controller is activated for precise tracking and it takes a relatively long time to regulate because the vehicle moves in a way to increase sideward relative distance. The final landing deviation in the sideward direction is 14 cm.




(a) Sideward relative distance (b) Roll command inputs

Figure 4.44: Sideward relative distance and roll command in case-4

The vertical relative distance and throttle command input in time are shown in Fig. 4.45.

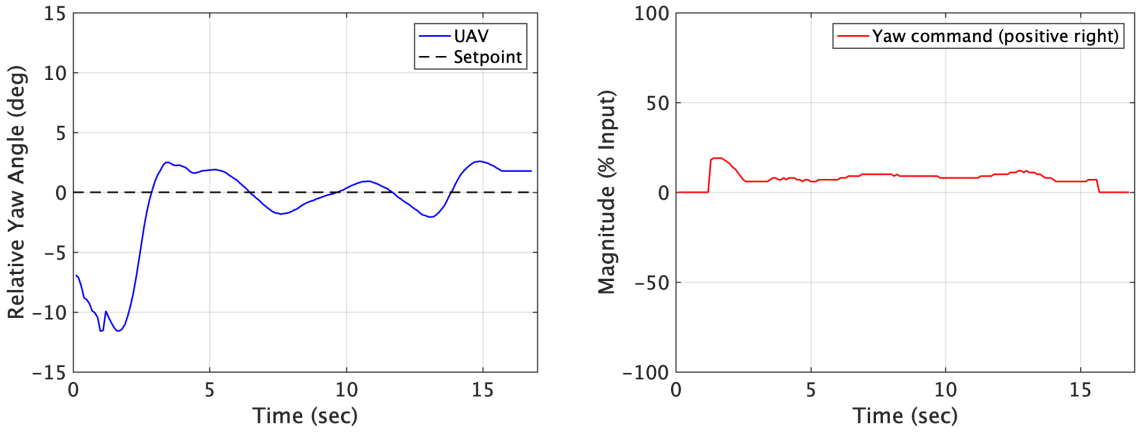



(a) Vertical relative distance (b) Heave(throttle) command inputs

Figure 4.45: Vertical relative distance and heave(throttle) command in case-4

The zero setpoint means 15 cm above the landing pad and it commands the maximum throttle down to land immediately at 15.7 seconds at which the landing conditions are satisfied.

The relative yaw angle and yaw command input in time are shown in Fig. 4.46.




(a) Relative yaw angle (b) Yaw command inputs

Figure 4.46: Relative yaw distance and yaw command in case-4

The directional approach of safety mode commands no yaw for an initial 1 second. Due to the circular trajectory of the platform, the yaw angle is relatively large, however, kept less than 15 degrees. Thus, the level 2 yaw controller with the moving average method is activated to regulate the yaw angle. Since the yaw angle is not one of the landing conditions, the UAV lands with having a yaw angle of 1.8 degrees at 15.7 seconds.

### 4.3 Nonlinear Control Strategy

To improve the tracking capability and robustness of the gain-scheduled PID controller, nonlinear control systems are designed which adapt to situations for generating control inputs. In this implementation, the visual cue that is closely mimicking the actual horizon reference bar on a ship as shown in Fig. 3.7 of Chapter 3 as well as a ship platform with 6 DOF motion deck that reproduces the actual ship motion is constructed. Two different nonlinear control systems were developed depending on the vision system engaged, which are the machine/deep learning-based vision for long-range tracking and classical computer vision for close-range tracking. Each vision system provides different types of visual information at different rates, thus corresponding nonlinear control systems are developed separately.

The sequential steps involved in the vision-based nonlinear control system are live-streaming the video from UAV's onboard camera to an external base station computer, image processing, feedback controller generating the control inputs, and transmitting the control inputs back to the UAV. The processing time is considerably affected by the type of detection method engaged in the vision system. The average time to complete one cycle is 0.5 seconds when the machine learning detection is engaged and 0.03 seconds when the detection of the rectangle corner points is engaged. In order to cope with the time delay as well as potential sensor and estimation noise, nonlinear controllers, which can adapt to different situations are developed. Particularly, five different flight modes are configured according to the situations perceived by the vision system as shown in Fig. 4.47.

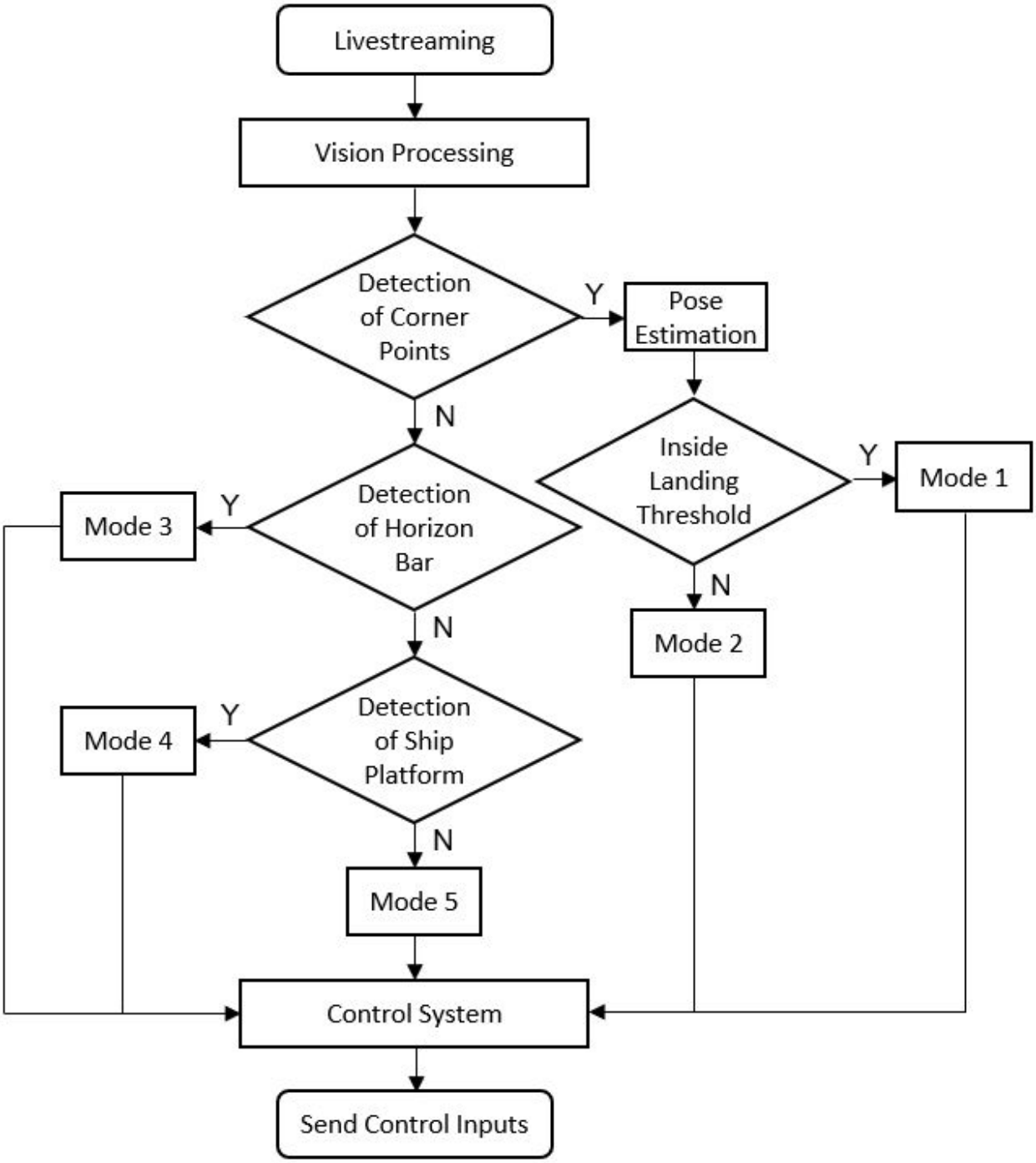



Figure 4.47: Flowchart of vision-based nonlinear control system

### 4.3.1 Control Architecture

The nonlinear control system is structured as shown in Fig. 4.48. Setpoints are different depending on the engaged vision system. When the machine/deep learning-based vision detects an object in a long distance, it returns the position of the object and the size of its rectangle bounding box in the image. The position and size are used to determine approach course and speed respectively by the long-range nonlinear control system. Since the detection range is proportional to the area occupied by the detected object in the image, during the early phase of the approach the whole ship is detected to maximize the detection range, and the horizon bar is detected in the mid-range. As the UAV gets closer to the ship, it detects the corner points on the horizon bar and estimates the relative distance and heading angle. In the close distance, the nonlinear control system yields control inputs according to the estimated positioning data.

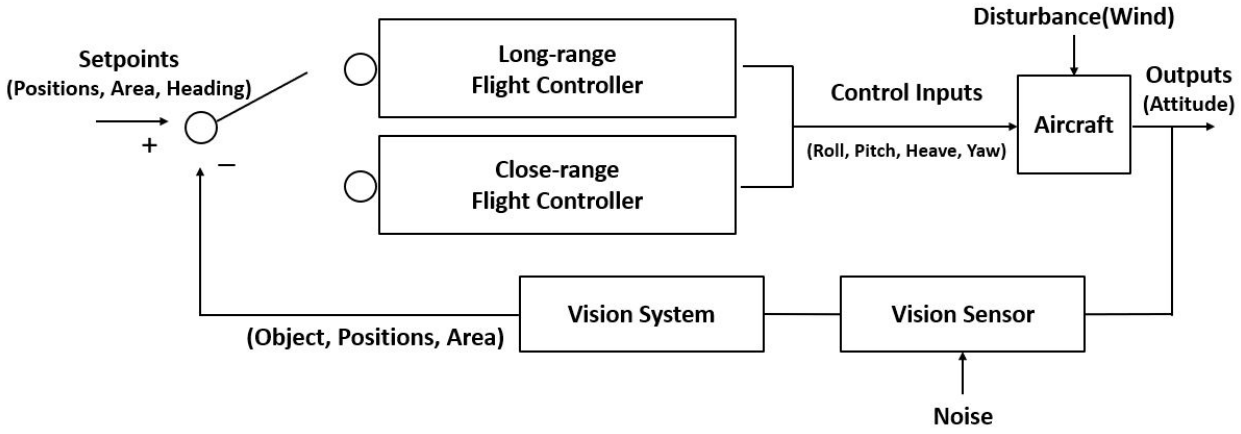



Figure 4.48: Nonlinear control system architecture

### 4.3.2 Long-range Tracking Controller

In the long range where the entire landing platform is detected as a whole, the information provided by the vision system is the platform size and position in the camera view. The flight control system in this region deals with a relatively slow update rate that is 0.5 seconds on average. When

the update rate is fast enough, the discrete system that receives sensor data at each update time can be a good approximation to the continuous system. Thus, integral and derivative controllers can be configured by using the summation and difference of error. However, the summation and difference are not good approximations for integral and derivative controllers when the update rate is slow. Hence, the nonlinear controller is designed to achieve the control task in the presence of the slow update rate by applying the nonlinear exponential gain $K_P\{e(t_k)\}$ as shown in Eqn. (4.7).

$$\begin{aligned} e(t_k) &= r(t_k) - c \\ u(t_k) &= K_P\big\{e(t_k)\big\} \times e(t_k) \\ &= \begin{matrix} -m(e^{ae(t_k)} - d) \times e(t_k) & (e(t_k) < 0) \\ m(e^{ae(t_k)} - d) \times e(t_k) & (e(t_k) \geq 0) \end{matrix} \end{aligned} \tag{4.7}$$

In Eqn. (4.7), $e(t_k)$ denotes the error at time $t_k$, $r(t)$ is the relative position at time $t_k$, and $c$ is the setpoint. The control law, $u(t_k)$, has exponential term in the nonlinear gain $K_P\{e(t_k)\}$ to decay control magnitude exponentially near zero error. The constants $m$, $a$, $c$, and $d$ selected for the ship platform and bar tracking are provided in Table 4.4.

Table 4.4: Selected constants of long-range controllers

| Flight Mode | Controller | m | a | c | d |
|---|---|---|---|---|---|
| Ship Platform Tracking | pitch | 0.008 | 0 | 5000 | 0 |
| | roll | 1.2 | 0.0158 | 640 | 1 |
| | heave | 3.0 | 0.0108 | 360 | 1 |
| Bar Tracking | pitch | 0.004 | 0 | 3400 | 0 |
| | roll | 1.0 | 0.0158 | 640 | 1 |
| | heave | 5.0 | 0.0108 | 360 | 1 |

The desired object size, the image center position in the horizontal direction, and the image center position in the vertical direction are the setpoints for the pitch, roll, and heave controllers, respectively. Unlike the roll and heave controllers, the pitch controller with the assigned parameters returns to a linear proportional controller, which is sufficient to approach the ship in the long range. There is also a yaw controller that can control the heading angle to set the approach course. It is designed as a nonlinear probabilistic control system, which is the same as the close-range tracking yaw controller and detailed in the following section. The only difference is that a magnetometer is used to read the current heading in the long-range and the vision system provides the estimated heading angle during close-range tracking.

The exponential nonlinear roll control input is shown in Fig. 4.49.

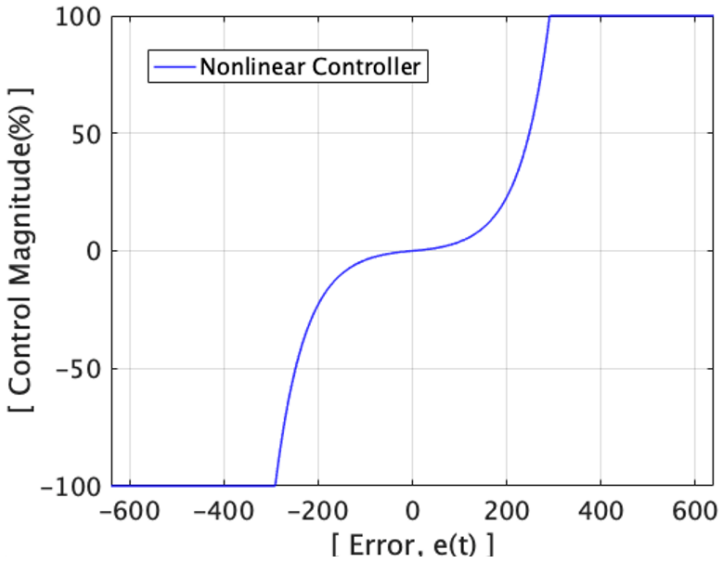



Figure 4.49: Exponential nonlinear roll control input

Depending on the constant value $a$, the rate of change of control magnitude varies. In the roll controller case, constant $a$ is selected as 0.0158. The maximum and minimum control magnitudes are limited to 100 and -100, respectively. Even though it utilizes only the error at time $t$, it can minimize the overshoot around the setpoint where the error is zero by decaying quickly. On contrary, it yields a large control magnitude as the error becomes larger.

In the case of a slow update rate, the nonlinear controller is able to achieve stable setpoint tracking. The effect of the nonlinear roll controller is compared to the conventional linear proportional controller and proportional-integral-derivative (PID) controller as shown in Fig. 4.50.

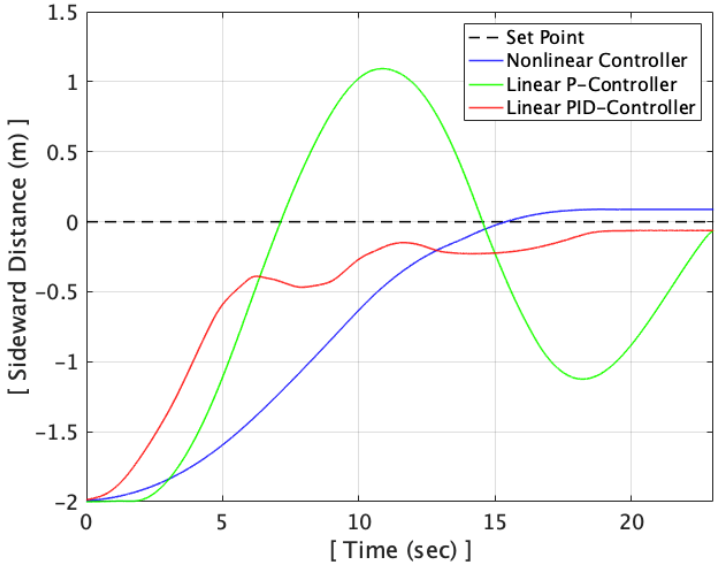



Figure 4.50: Effect of exponential nonlinear controller

It demonstrates that the nonlinear controller can prevent the system from overshooting the setpoint. However, the linear proportional controller cannot stabilize the oscillations because it computes the control input by multiplying the error with a fixed gain value. Even the linear PID controller is not able to stabilize the oscillations in an effective fashion due to the slow update rate.

However, as seen from Fig. 4.50, the nonlinear controller has some steady-state error. Considering the long distances at which the nonlinear controller is engaged, the steady-state error is not an issue because this is not the final error of the entire approach and landing maneuver, but the initial error for the corner tracking system, which takes over the control at close distances. Therefore, it is more imperative to control the aircraft in a stable fashion than to achieve the minimum steady-state error while the UAV is approaching from a long distance.

### 4.3.3 Close-range Tracking Controller

At close range, when the vision system can reliably detect the corners of the rectangles on the visual cue, the vision-based controller utilizes the estimated relative position and orientation data to yield control inputs. The average update rate is 0.03 seconds, which is significantly faster than the machine learning object detection that is applied at long distances. Having integral and derivative controllers along with the proportional controller enables precise tracking. Even though the estimation provides accurate position and orientation data with sub-centimeter and sub-degree error, a small error difference between subsequent time steps can yield large and noisy control magnitude since it depends on the difference in error divided by the small update rate of 0.03 seconds. To take advantage of the derivative controller with minimum noise, the Kalman filter and nonlinear derivative controller are designed.

A single state Kalman filter is applied with the unity feedback, which means that it does not require the prediction from the model. This model-free estimator reduces the noise, however, it uses error difference values for estimation. Therefore, it will be affected by incorrect and unrealistic error difference values that may occur from time to time. To reject this intermittent unrealistic estimation effectively, a novel nonlinear derivative controller with linear proportional and integral controllers is designed by utilizing the normal distribution concept as described in Eqn. (4.8).

$$\begin{aligned} de(t_k) =& e(t_k) - e(t_{k-1}) \\ u(t_k) =& K_P e(t_k) + K_I \sum_{k=1}^{k} e(t_k) dt_k + K_D\big\{de(t_k)\big\} \frac{de(t_k)}{dt_k} \end{aligned} \tag{4.8}$$

$de(t_k)$ is an error difference between time $t_k$ and $t_{k-1}$, and $u(t_k)$ is a control law that has linear proportional and integral terms as well as the nonlinear derivative term. $K_P$ and $K_I$ are constant proportional and integral gains, and $K_D\{de(t_k)\}$ is the nonlinear derivative gain that is a function of error difference, $de(t_k)$, as specified in Eqn. (4.9).

$$
\begin{aligned}
f(x) &= \frac{1}{\sqrt{2\pi\sigma^2}} e^{-0.5\frac{(x-\mu)^2}{\sigma^2}} \\
K_D\{de(t_k)\} &= be^{-0.5\frac{(de(t_k)-\mu)^2}{\sigma^2}}
\end{aligned}
\tag{4.9}
$$

$f(x)$ is the probability density function that forms a normal distribution. $\sigma$ denotes the standard deviation and $\mu$ denotes the mean value. The area under the function indicates the probability that a certain range of deviation occurs. The probabilistic nonlinear derivative controller is constructed by taking the exponential term from the probability density function and multiplying constant $b$ that determines the control magnitude. Based on the observation of aircraft movement, $\sigma$ is determined as 0.04, which means the distance that the aircraft can move during 0.03 seconds has a 68.2 percent chance of being within 4 cm and a 95.4 percent chance of being within 8 cm. When $b$ is 1 and $\mu$ is 0, the probabilistic nonlinear derivative controller computes derivative gains as shown in Fig. 4.51.

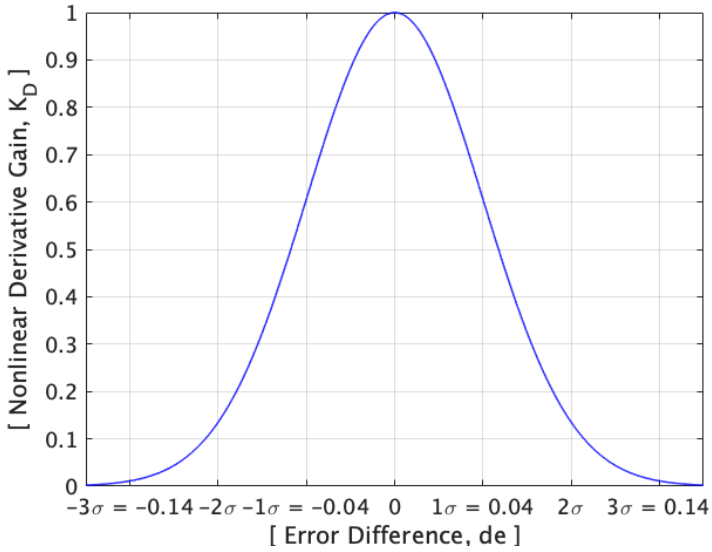



Figure 4.51: Probabilistic nonlinear derivative controller

Depending on the error difference, $de(t_k)$, the corresponding nonlinear derivative gain, $K_D$, is selected and multiplied by the derivative term $de(t_k)/dt_k$. If the estimated error difference is too high (or unrealistic), then it takes a small $K_D$ value to significantly minimize the control

input. In this way, the controller does not respond to the large noise in the error data, which can trigger undesired and unstable maneuvers. Also, the magnitude of gain can vary according to the selection of constant $b$ in Eqn. (4.9). The effects of the Kalman filter and the probabilistic nonlinear controller are shown in Fig. 4.52.

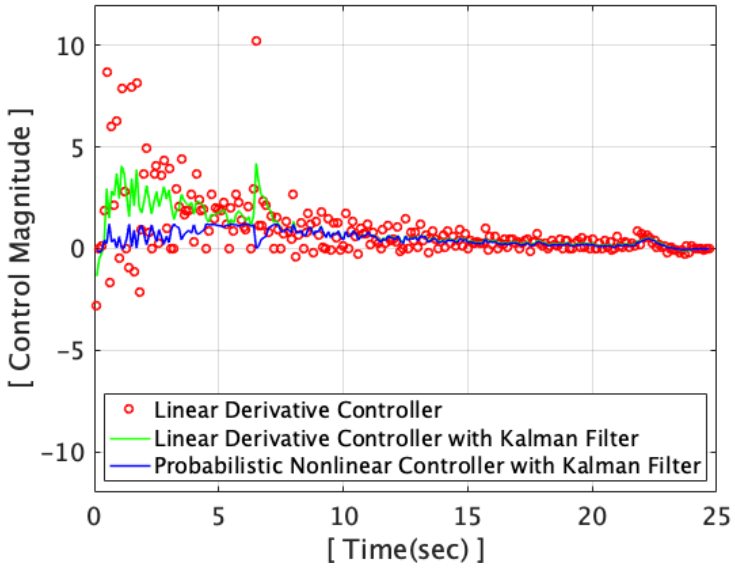



Figure 4.52: Effect of kalman filter and probabilistic nonlinear derivative controller

Red circles denote the control magnitude of the linear derivative controller without the Kalman filter. The high noise that occurs in the range of 0 to 10 seconds are due to the derivative term, $de(t_k)/dt_k$. This term is sensitive because the error is divided by a small $dt_k$, which has an average value of 0.03 seconds. Thus, even a small error in estimation can be magnified in the derivative controller. The green line is the result after applying the Kalman filter to the linear derivative controller. The noise is reduced, however, it still yields large control inputs in response to the incorrect estimation values. The blue line shows the control inputs generated by the probabilistic nonlinear controller with the Kalman filter and these inputs are relatively small and insensitive to the large unrealistic error differences. At the 6.5 second mark, the error difference $de$ has a large value caused by incorrect estimation. In this case, the linear derivative controller with the Kalman filter reduced the magnitude to some extent but it is still affected by that particular spurious error value. However, the probabilistic nonlinear controller with the Kalman filter is able to screen out

the wrong value, thereby minimizing the effect of incorrect error estimation. The finally selected gains through extensive flight tests and simulations are specified in Table 4.5.

Table 4.5: Selected gains of close-range controllers

| Flight Mode | Controller | $K_P$ | $K_I$ | $K_D$ | | |
|---|---|---|---|---|---|---|
| | | | | b | $\mu$ | $\sigma$ |
| Corner Points Tracking | pitch | 7.5 | 0.05 | 4.5 | 0.02 | 0.04 |
| | roll | 7.5 | 0.01 | 8.5 | 0 | 0.04 |
| | heave | 15 | 0.01 | 7 | 0 | 0.02 |
| | yaw | 5.5 | 0.05 | 1.75 | 0 | 5 |

## 4.4 Deep Reinforcement Learning Control Strategy

Designing a conventional feedback control system requires an explicit algorithm, a precise model, and a significant effort to tune the control gains. Even a meticulously designed conventional control system for a certain task is subject to failure when unforeseen conditions are encountered, which is a crucial factor that can lead to an unsuccessful ship landing. At the proximity of a ship, the airflow is highly unsteady and turbulent due to the wake from the moving ship structure. This turbulent wake has significant impacts on aircraft aerodynamics. Therefore, a deep reinforcement learning (RL)-based control strategy is developed to ensure robust flight performance in the presence of uncertainties such as sudden wind gusts.

Using the representation power of deep neural networks, RL provides an attractive model-free approach for developing nonlinear control algorithms for high dimensional systems in an automated and computationally tractable way. A clever modification of the standard off-the-shelf RL approach is made to adapt it to the vision-based autonomous ship landing problem and to make it robust against adversarial disturbances in the environments.

Multiple previous works have developed RL algorithms to address the autonomous landing of UAVs. RL-based high-precision, time-critical flight attitude control that could operate in unpre-

dictable and harsh environments is discussed in [97]. It also discusses different policy gradient RL algorithms such as deep deterministic policy gradient (DDPG), proximal policy optimization (PPO), and trust region policy optimization (TRPO). An RL approach for UAV landing task on a moving platform using a variant of the DDPG algorithm is discussed in [98]. An actor-critic RL framework used to fly a UAV by following designated waypoints is presented in [99]. A variant of the DDPG algorithm is used to recover a UAV attitude quickly from an out-of-trim flight state [100]. However, these works do not address the problem of designing RL controllers that are robust against adversarial disturbances such as wind gusts. We adapt the state-of-the-art RL algorithm called twin delayed DDPG (TD3) and use the idea of domain randomization [101, 102] to make it robust. The disturbance rejection capability of the RL controller is compared to the previously developed nonlinear control system with the Kalman filter [72].

### 4.4.1 Reinforcement Learning Overview

Reinforcement Learning is a very intuitive state-of-the-art machine learning-based approach for finding the optimal control policy for an unknown dynamical system. In this approach, an agent interacts with an environment to learn the best action for any given state. After executing each action, the agent receives a reward and the system evolves to the next state. The primary objective of the agent is to learn the optimal policy that can maximize the cumulative rewards.

Mathematically, a policy $\pi \in \Pi$ is a function that maps state $s \in S$ to an action $a \in A$, where $\Pi$ is the policy space, $S$ is the state space, and $A$ is the action space. The transition probability model $P(s_{t+1} \mid s_t, a_t)$ determines the dynamics of the system. At each timestep, policy $\pi$ determines an action to be taken and the agent receives a reward $r(s_t, a_t)$ from the environment. The policy can be stochastic $\pi(a, s)$ (probability of taking action $a$ in state $s$) as well as deterministic $\pi(s)$. The agent tries to maximize the cumulative discounted reward $R_t(s) = \sum_{i=t}^{T} \gamma^{(i-t)} r(s_i, a_i)$ of state $s$ from time $t$ to $T$. The $T$ can be finite value (finite horizon episodes) or infinite value (infinite horizon episodes). The parameter $\gamma$ is know as discount factor that determines the importance of future rewards. The two other important terms in RL formulation is the value function $V^{\pi}$ and action-value function or Q-function $Q^{\pi}$. Value function is the expectation of discounted cumulative

reward for a state $s$, defined as $V^{\pi}(s) = \mathbf{E}_{\pi}[R_t(s) \mid s]$. The optimal policy $\pi^*$ can be expressed as one which maximizes the value function, $\pi^*(s) = \arg\max_{\pi} V^{\pi}(s)$.

### 4.4.2 Algorithm Selection

Robotic problems in general are associated with continuous state space and continuous action space. With the advancement in deep neural networks over the years, combining RL with deep learning shows impressive learning results. A deep version of Q-learning (DQN) algorithms has been successful in playing video games [103, 104]. Two key features of DQN are the experience replay and a target network. The experience replay stores prior experience $(s, a, r, s', a')$ in a buffer and randomly select one for updating Q-value network. Target network helps in target remaining unchanged during several gradient updates. However, DQN algorithms were not successful in continuous action-continuous states space. Policy gradient algorithms are more useful in continuous action-continuous states space problems. Deep deterministic policy gradient (DDPG) algorithm [105] can operate over continuous action and state spaces and it uses experience replay and slow-learning neural networks of the double Q learning. Particularly for the control of a UAV, DDPG has demonstrated its successful implementations [100]. In this study, an improved version of DDPG, twin delayed DDPG (TD3) [106, 107], is selected as the RL algorithm.

TD3 learns two Q-functions $Q_{\phi_1}$ and $Q_{\phi_2}$ by mean square Bellman error minimization. It adds clipped noise to actions that make the policy difficult to extract the Q-functions. The target action is obtained as described in Eqn. (4.10).

$$a'(s') = \text{clip}(\pi_{\theta_{\text{targ}}}(s') + \text{clip}(\epsilon, -c, c), a_{\text{low}}, a_{\text{high}}), \epsilon \sim N(0, \sigma) \tag{4.10}$$

$\pi_{\theta_{\text{targ}}}(s')$ is the target policy and $a_{\text{low}} < a < a_{\text{high}}$ is the valid action range. This is known as target policy smoothing which essentially serves as regularizer for the algorithm. TD3 updates the policy less frequently than the Q-functions. Both Q-functions use the same target $y(r, s', p)$ ($p$ indicates whether state $s'$ is terminal state or not) which is the calculated using the smaller Q-value among the two Q-functions as given in Eqn. (4.11).

$$y(r, s'p) = r + \gamma(1-p) \min_{i=1,2} Q_{\phi_{\text{targ},i}}(s', a'(s')) \tag{4.11}$$

The two Q-functions are then optimized by using the following two loss functions (mean squared Bellman error function) given in Eqs. (4.12) and (4.13).

$$L(\phi_1, D) = \mathbf{E}[(Q_{\phi_1}(s,a) - y(r,s',p))^2 \quad | \quad (s,a,r,s',p) \sim D] \tag{4.12}$$

$$L(\phi_2, D) = \mathbf{E}[(Q_{\phi_2}(s,a) - y(r,s',p))^2 \quad | \quad (s,a,r,s',p) \sim D] \tag{4.13}$$

$\phi_1$ and $\phi_2$ are network parameters and $D$ is the replay buffer.

### 4.4.3 Training Procedure

The objective of the RL control strategy is for the UAV to stably hover and fly as desired against sudden wind gusts. To obtain an RL control policy that is effective in such a challenging condition, training is conducted in a realistic Gazebo simulation environment along with the precise UAV model as shown in Fig. 4.53.

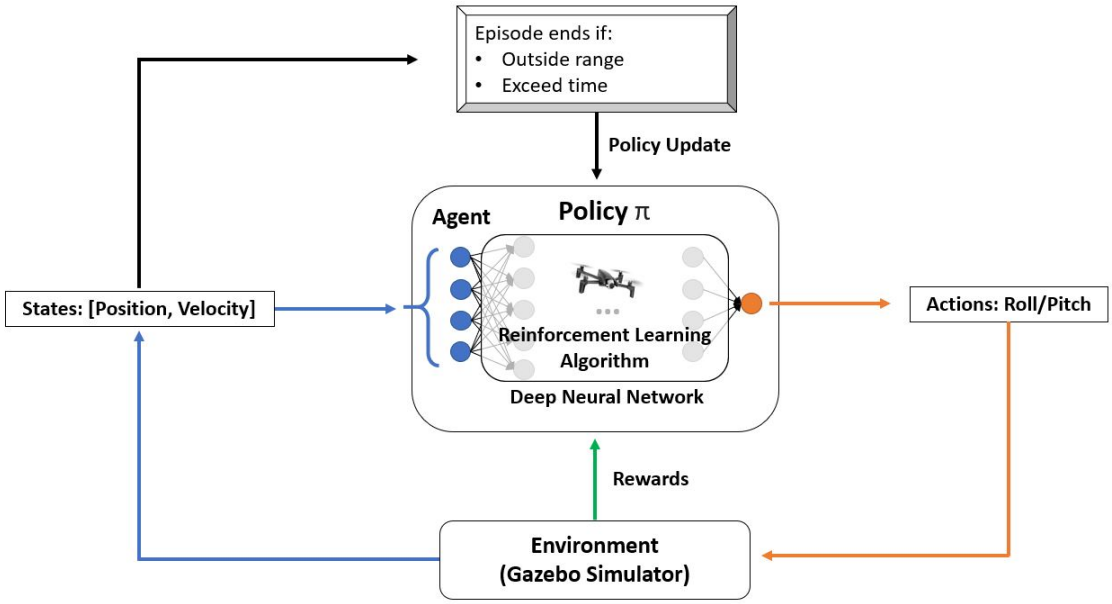



Figure 4.53: Schematic showing the training process for reinforcement learning

During the training session, an agent takes some set of actions in an environment to achieve a particular task. The agent is a UAV that decides actions based on the rewards by using neural networks. The quality of those actions is quantified by user-defined rewards that are assigned in the environment and the better sets of actions to achieve the task will acquire higher cumulative rewards. A particular period that the agent takes for a set of actions by the RL control policy is called an episode. As the episode continues, the control policy keeps getting updated according to an RL algorithm and the cumulative rewards will be converged to a certain value.

To develop a successful RL control strategy, there are key requirements which are: (1) a precise model of the agent, (2) a realistic simulation engine, (3) an appropriate selection of states, (4) careful reward engineering, and (5) a powerful RL algorithm.

First, a commercial off-the-shelf (COTS) quadrotor UAV, Parrot Anafi, is used for this study. Along with the physical UAV, the precise UAV model (agent) that can be used in the Gazebo simulation is provided. Second, Gazebo is a realistic simulation engine that is widely used for robotics applications. It has closely simulated the real flight dynamics. The agent can communicate with the simulation engine using a framework called Olympe. Olympe allows the control of UAV and access to its sensors through python scripts.

#### *4.4.3.1 Action Space*

The UAV has four different control actions: Roll, Pitch, Yaw, and Heave. It is observed in the simulation that roll and pitch actions were heavily influenced by the wind as compared to the yaw and heave actions. Hence the roll and pitch actions are chosen to be part of the action space. It is worth noting that for the given UAV, the roll and pitch actions are independent of each other. Roll action only makes the UAV move right and left, while pitch action only makes it move forward and backward. Taking advantage of this uncoupled behavior, it is decided to split the 2-D action space into two 1-D action spaces. Two independent RL models were trained for roll and pitch controllers, respectively. This not only helps in reducing the dimension of the action space but also reduces the dimension of the state space used for the independent models. The splitting of the two controllers helps in reducing the complexity of the policy network and hence resulted in faster

training and better convergence.

#### *4.4.3.2 State Space*

As mentioned earlier, the mission objective of the UAV is to achieve a target point and hover at that position in the presence of wind. So the position of the UAV is an obvious choice as a state. Since the action space is 1-D, the position state is also 1-D that is the position along the respective action axis for each controller. The roll controller objective is to achieve a certain target on the roll axis and maintain that position without considering the pitch motion. Similarly, for the pitch controller, the objective is to maintain a position on the pitch axis independent of the roll motion. Also, velocities are included as states, which can take an action quickly responding to velocity changes before the drift becomes large. Finally, the history of position and velocity are also selected as states for both roll and pitch controllers, which allows determining actions based on current states and trends.

To verify the effect of selected states, three different sets of states as shown in Table 4.6 are modeled for pitch control and roll control individually and applied for learning.

Table 4.6: Models with different sets of states

| Model No. | Model 1 | Model 2 | Model 3 |
|---|---|---|---|
| States for pitch control | $[x]_t$ | $[x, v_x, v_y]_t$ | $[(x, v_x, v_y)]_{t-i}$ where i = 0 to 5 |
| States for roll control | $[y]_t$ | $[y, v_x, v_y]_t$ | $[(y, v_x, v_y)]_{t-i}$ where i = 0 to 5 |

$x$ and $y$ are the deviations along the x-axis and the y-axis, respectively. $v_x$ and $v_y$ are the velocities along the x-axis and the y-axis, respectively. $t$ is the current time, thus model 3 is also taking five previous deviations and velocities as states.

During the training, the simulation engine imposes four different kinds of wind conditions: zero wind, constant magnitude wind (-10 m/s to 10 m/s), sudden wind magnitude change (-10 m/s to 10 m/s magnitude change), and a sinusoidal wind (amplitude of 5 m/s and time periods of 10 secs, 20

secs, 30 secs, 40 secs and 50 secs). These four different conditions were applied among different episodes. The idea here is that this makes the agent learn proper control actions for various wind scenarios. This is known as domain randomization and it has been an active area of research in RL [108]. It is observed that headwind affects the forward drift and crosswind affects the sideward drift. Hence crosswinds are applied for roll controller training and headwinds are applied for pitch controller training.

#### *4.4.3.3 Reward Function*

Reward functions are very important in the RL framework. A proper design of the reward function can yield a better convergence rate and good performance in the testing phase. A normalized reward function is opted here, which penalizes every action based on the deviation from the target point. The reward function is carefully designed to represent the control objective in numbers as shown in Eqn. (4.14).

$$\text{Reward} = \begin{cases} -4 \times |d| & \text{if } |d| < 0.25 \\ -1 & \text{if } 0.25 \leq |d| \leq 2 \\ -(T_{\max} - T_{\text{inside}}) & \text{if } |d| > 2 \end{cases} \tag{4.14}$$

$d$ denotes the deviation along the x-axis for the pitch control case and the deviation along the y-axis for the roll control case. $T_{max}$ is the maximum episode time and $T_{inside}$ is the time that the UAV stays within 2 meters from the hovering point. Thus, it can accumulate higher rewards (less negative) as it stays closer to the point. In other words, it will get more penalties as it deviates from the point. One episode ends if the deviation is greater than 2 meters and a new episode is started. Thus, the reward written by the time factor, $-(T_{max} - T_{inside})$, is acquired only once in the unsuccessful case. The idea here is that rather than exploring unnecessary states, it is better to give the future cumulative rewards in one step. This helps in maintaining the consistency of episodic rewards. In each episode, the UAV is spawned at a random position near the origin. Apart from the other common RL control strategies, the episode does not end even though the UAV achieves

the target position. Instead, it is designed to continue maintaining the position until the end of the episode time, which can allow sufficient time to learn the proper control actions.

Successful training is heavily dependent on the selection of an RL algorithm. One of the modern policy gradient algorithms, TD3 is used for the study. The applied hyper-parameters that control the learning process are given in Table 4.7.

Table 4.7: Selected hyper-parameters for training

| | | | |
|---|---|---|---|
| gamma | 0.99 | policy delay | 2 |
| learning rate | 0.0001 | target policy noise | 0.2 |
| buffer size | 100000 | target noise clip | 0.5 |

#### *4.4.3.4 Convergence*

The training convergences of pitch and roll control cases are shown in Figs. 4.54 and 4.55, respectively.

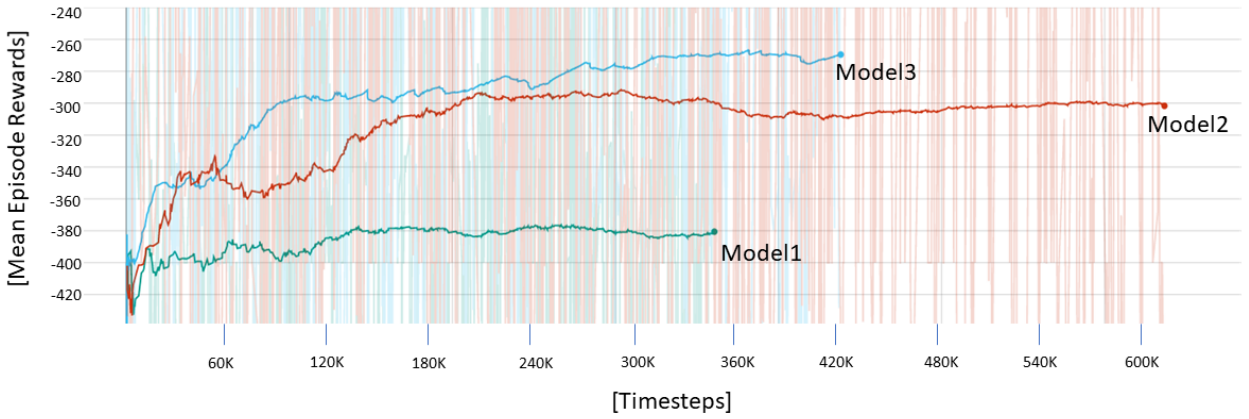



Figure 4.54: Training convergence of different models for the pitch control case

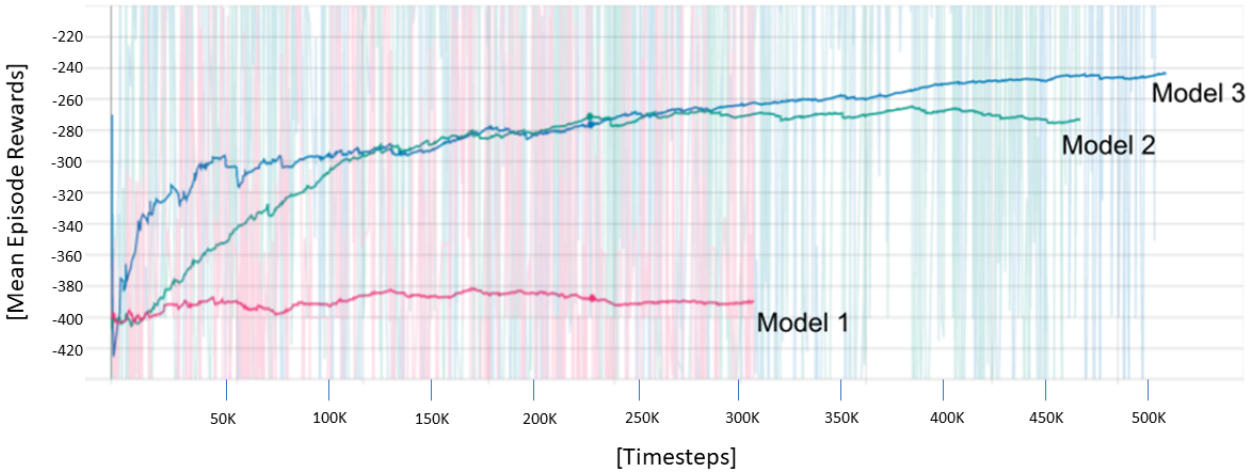



Figure 4.55: Training convergence of different models for the roll control case

In both the pitch and roll control cases, model 3 converges to the higher accumulated reward value and learns faster than the other models on average. It demonstrates that taking previous positions and velocities as states is helpful, which is designed based on the intuition that neural networks can decode the wind gust information from those states. The obtained control policy is used in the simulations and flight tests.

The system used for training is LENOVO Legion Y740-15IRH, which is composed of Intel(R) Core(TM) i7-9750H CPU @ 2.60GHz, 2592 Mhz, 6 Cores, and 12 Logical Processors. It features an integrated NVIDIA GeForce GTX 1660 Ti 6GB Graphics and 8GB of LPDDR4 memory with a 128-bit interface. The Ubuntu 18.04 OS with Nvidia driver version 440 and CUDA version 10.2 are used. The same system is used for flight simulations and testing.

### 4.4.4 Simulation Results

In order to verify the disturbance rejection capability in challenging situations, sudden wind gusts are imposed while the UAV is hovering. 5 m/s diagonal wind (45°) and cross-wind (90°) based on UAV heading are simulated using a step function. The performances of the previously developed nonlinear PID-based feedback control system with the Kalman filter and the deep RL control strategy are compared based on the sideward and forward drifts. The pitch and roll control inputs are also examined. The simulation environment is shown in Fig. 4.56.

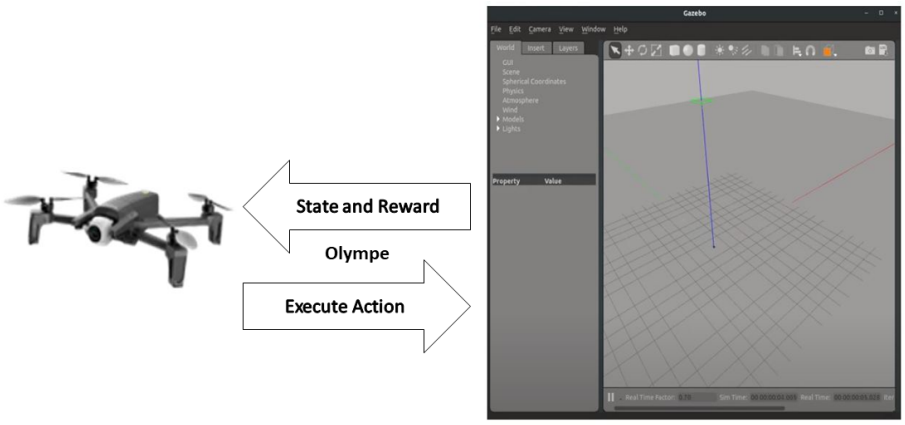
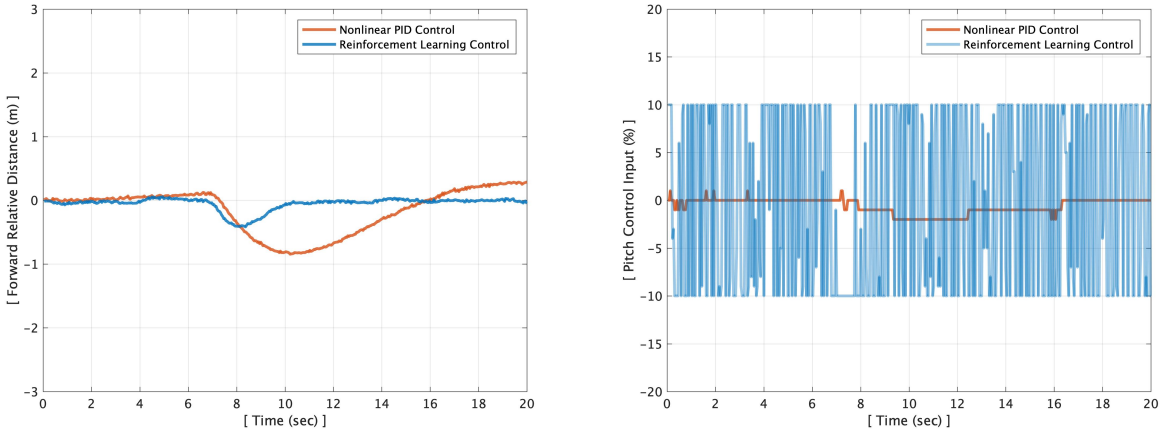



Figure 4.56: Simulated UAV (agent) and Gazebo simulation environment

#### *4.4.4.1 Case-1: Hovering at 5 m/s of diagonal wind gust (45°)*

The sudden diagonal wind is imposed approximately at the 7 second mark, which affects the forward and sideward drifts equally. The forward drift from the hovering point and pitch control inputs in the case of the nonlinear PID controller and RL controller are shown in Fig. 4.57.

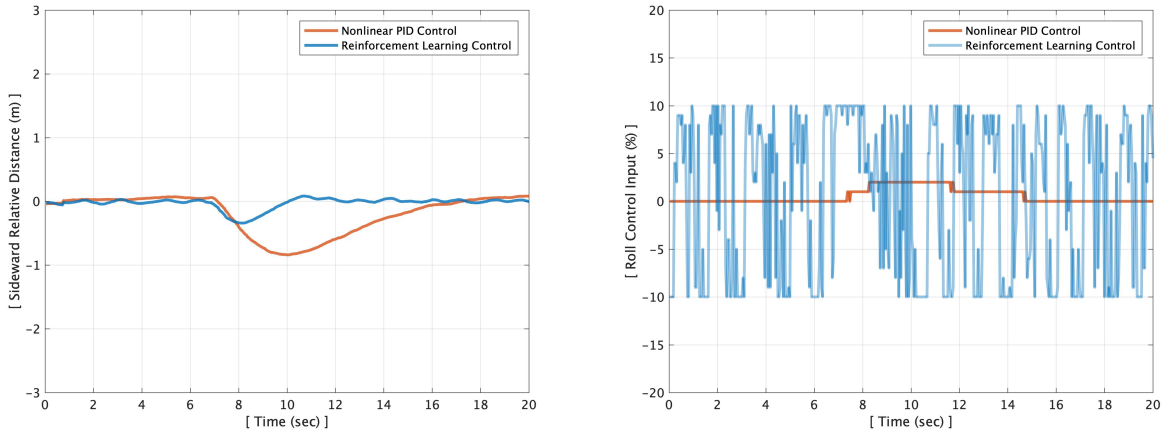



(a) Forward drift (b) Pitch control inputs

Figure 4.57: Forward drift and pitch control inputs for case-1

The RL control regulates the impact of wind gust within 2 seconds with a maximum forward drift of 0.4 meters, which is 2 times smaller drift and 5.5 times faster settling time than the nonlinear PID control system. The RL control strategy yields high-frequency pitch commands via the trained control policy, which rejects the disturbance more effectively than the nonlinear PID control system.

The sideward drift from the hovering point and roll control inputs in the case of the nonlinear PID control and RL control are shown in Fig. 4.58. The roll response looks similar to pitch, which is expected for hovering flight because of the symmetry.




(a) Sideward drift

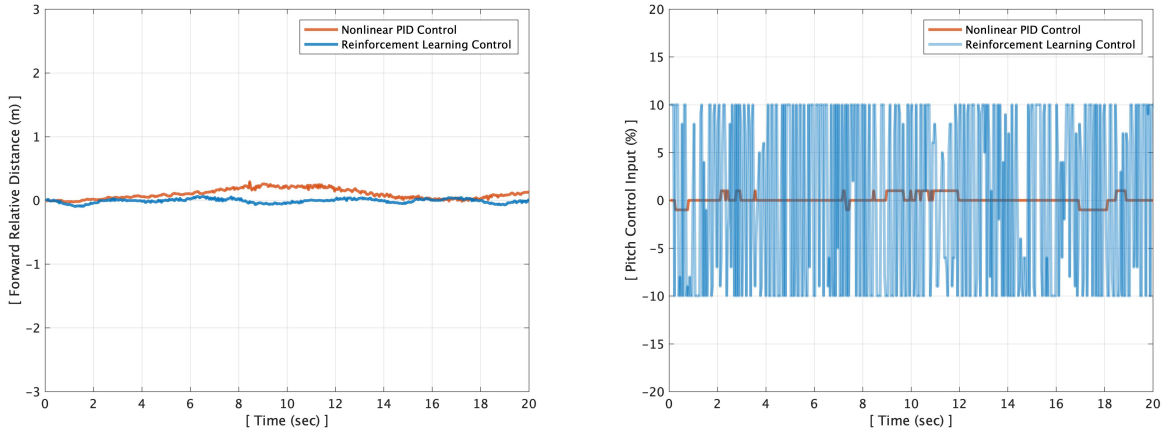
(b) Roll control inputs

Figure 4.58: Sideward drift and roll control inputs for case-1

The RL control regulates the effect of wind gust within 3 seconds with a maximum sideward drift of 0.3 meters, which is 3 times smaller drift and 3 times faster settling time than the nonlinear PID control system. Similar to the pitch commands, the RL control system generates high-frequency roll commands that effectively reject the disturbances better than the nonlinear PID control system.

#### 4.4.4.2 *Case-2: Hovering at 5 m/s of crosswind gust (90°)*

In this case, a sudden crosswind is imposed approximately at the 7 second mark. This will have a significant impact on the sideward drift. The forward drift from the hovering point and pitch control inputs in the case of the nonlinear PID control and RL control are shown in Fig. 4.59. Since the wind gust direction is orthogonal to the UAV heading, its effect on forward distance is small.

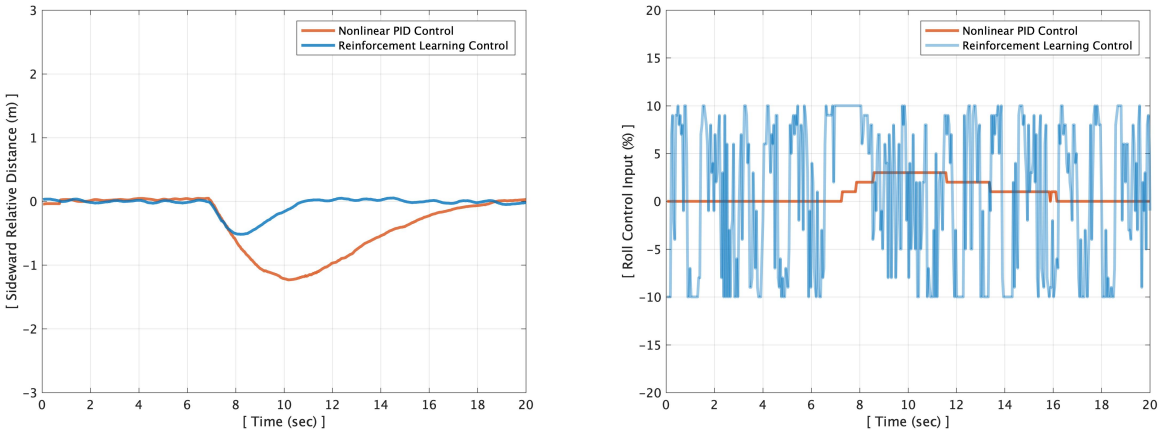



(a) Forward drift

(b) Pitch control inputs

Figure 4.59: Forward drift and pitch control inputs for case-2

The sideward drift from the hovering point and roll control inputs in the case of the nonlinear PID control and RL control are shown in Fig. 4.60. Compared to the control inputs commanded by the nonlinear PID control inputs, the RL control quickly responds to the crosswind gust with higher control magnitude, which demonstrates improved disturbance rejection capability. The RL control regulates the wind gust within 3 seconds with the maximum sideward drift of 0.4 meters, which is 2 times smaller drift and 3 times faster settling time than the nonlinear PID control system.

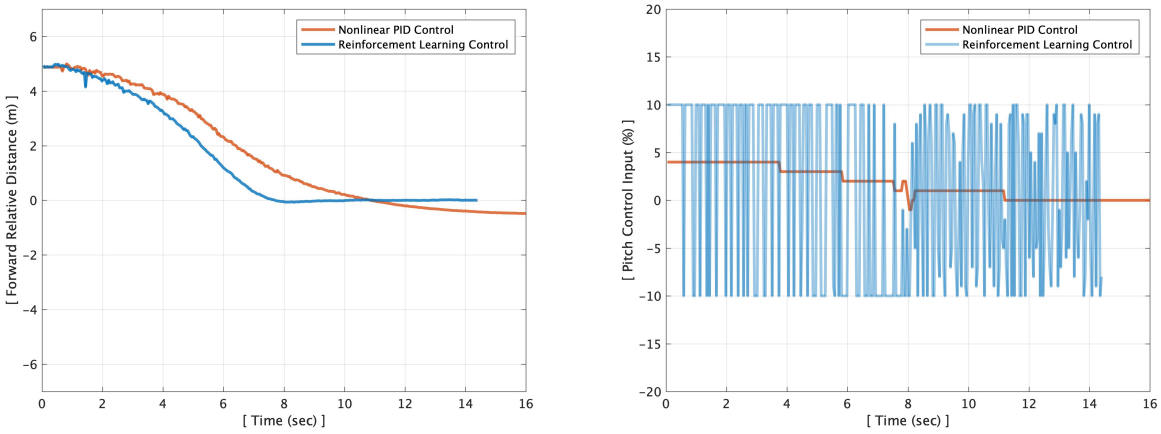



(a) Sideward drift (b) Roll control inputs

Figure 4.60: Sideward drift and roll control inputs for case-2

#### *4.4.4.3 Case-3: Tracking at 5 m/s of diagonal wind gust (45°)*

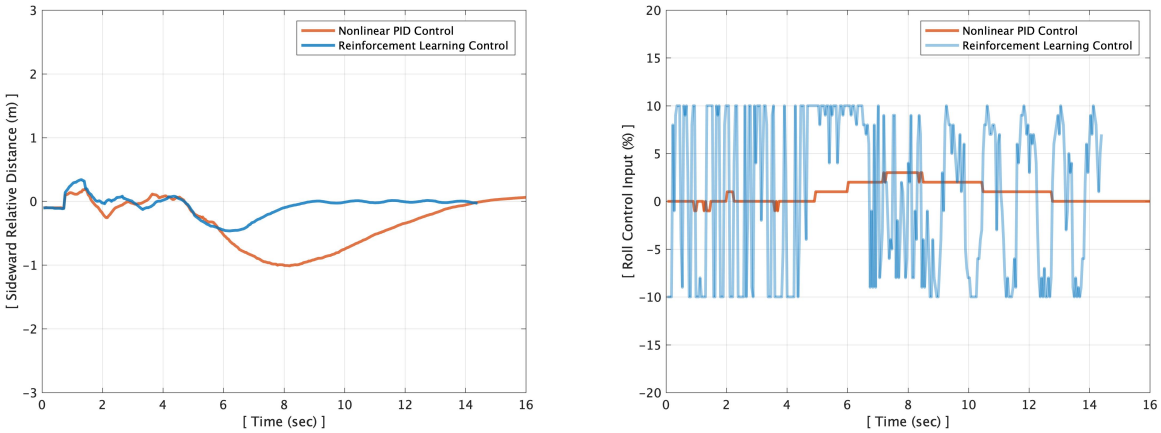
A 5 m/s of sudden diagonal wind is imposed approximately at the 5-second mark while the UAV approaches the ship platform. The forward relative distance between the UAV and landing deck center and the pitch control inputs in the case of the nonlinear PID control and RL control are shown in Fig. 4.61.




(a) Forward drift (b) Pitch control inputs

Figure 4.61: Forward drift and pitch control inputs for case-3

Since the initial forward relative distance of 4.5 meters is dominant, the drift by the wind gust is not clearly seen. The RL control inputs yielded by the control policy using deep neural networks are not fully describable, however, it is obvious that this control strategy demonstrates faster and accurate tracking than the nonlinear PID control system. The sideward drift during approach in the presence of diagonal wind gust and the roll control inputs in the case of the nonlinear PID control and RL control are shown in Fig. 4.62.




(a) Sideward drift

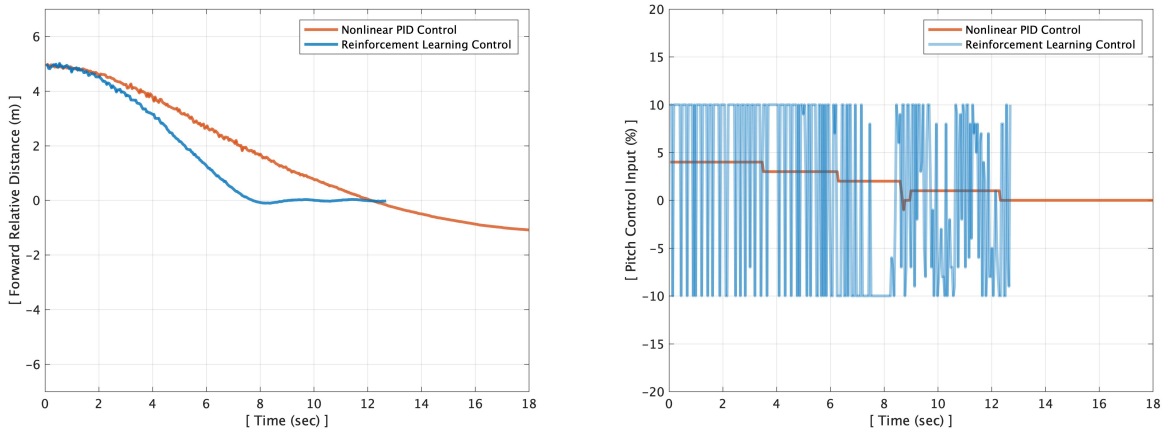
(b) Roll control inputs

Figure 4.62: Sideward drift and roll control inputs for case-3

The RL control is able to adjust and stabilize the UAV in the wind gust within 3 seconds with the maximum sideward drift of 0.45 meters, which is 2 times smaller drift and 3 times faster settling time than the nonlinear PID control system.

#### *4.4.4.4 Case-4: Tracking at 5 m/s of crosswind gust (90°)*

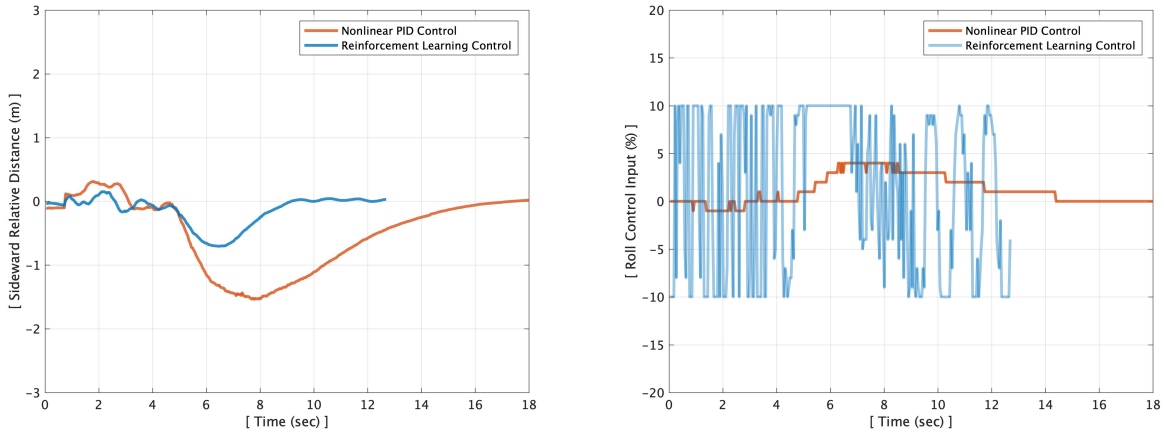
At approximately the 5 second mark, a sudden crosswind is imposed which significantly affects the sideward drift and has minimum impact on the forward drift. The forward relative distance between the UAV and landing deck center and pitch control inputs for nonlinear PID control and RL control are shown in Fig. 4.63.




(a) Forward drift (b) Pitch control inputs

Figure 4.63: Forward drift and pitch control inputs for case-4

Since the wind gust direction is orthogonal to the UAV heading, its effect on forward distance is small. The RL control demonstrates 1.5 times faster tracking capability without overshoot than the nonlinear PID control system. The sideward drift during approach and roll control inputs in the case of the nonlinear PID control and RL control are shown in Fig. 4.64.




(a) Sideward drift (b) Roll control inputs

Figure 4.64: Sideward drift and roll control inputs for case-4

The RL control regulates the wind gust within 3 seconds with the maximum sideward drift of 0.7 meters, which is 2 times smaller drift and 3 times faster settling time than the nonlinear PID control system.

Since the RL control strategy computes control inputs (actions) from states, which include the current and 5 previous positions and velocities using deep neural networks, it is impossible to explicitly describe the relationship between the states and actions. As seen from the simulations, the resultant control inputs are significantly different from the control inputs from the nonlinear PID, which is an explicit control algorithm. However, this completely new way of controlling the UAV using the RL controller clearly demonstrates superior disturbance rejection capability than the nonlinear PID control system.

# 5. FLIGHT TESTING

To demonstrate the novel autonomous ship landing methodology in practice, extensive flight tests are conducted. The specific objectives are to prove the safety of vertical landing without considering the platform motion as well as to demonstrate the tracking capability and landing accuracy. This chapter divides the flight testing into three sections based on the objectives and each section includes detailed results obtained from the implementation of gain-scheduled PID control system, nonlinear control system, and deep reinforcement learning-based control system. Throughout the flight tests, the quadrotor UAV that has a gimballed camera has been used. However, the visual cue and the landing platform have progressed during the course of the project. In the earlier phase of flight testing, a checkerboard pattern visual cue is used along with a landing platform that has a fixed landing pad. Later, the horizon bar visual cue is structured to the ship platform that has a motion deck. Independent of visual cues, platforms, and control systems, the autonomous flight shares the same process as shown in Fig. 5.1. It has an embedded (onboard) inner-loop autopilot that controls each propeller's rpm to achieve the commanded inputs generated by the outer-loop vision-based control system (offboard) developed in this study.

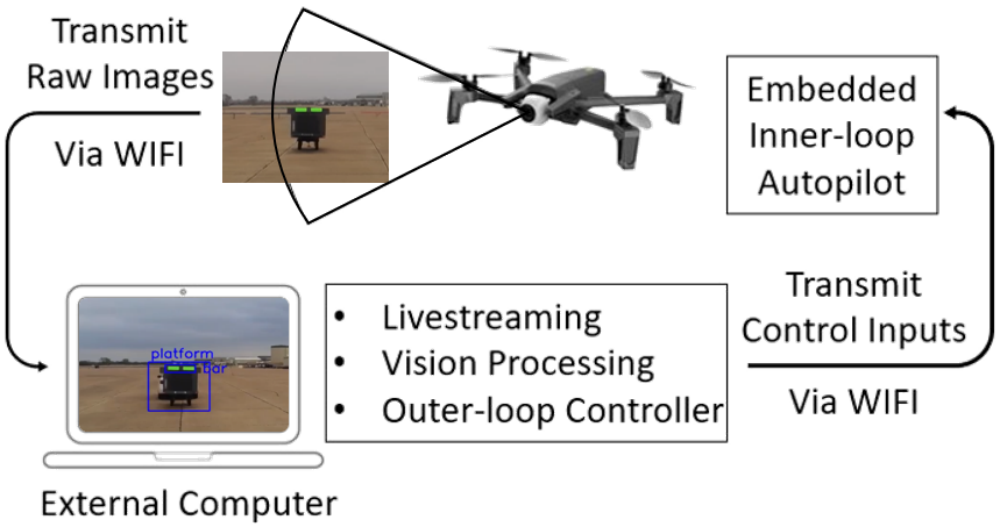



Figure 5.1: Process of autonomous flight system

The UAV is controlled by a Python script that runs on an external base station computer. The external computer communicates with the UAV through the WIFI connection. The UAV transmits raw images captured by the onboard camera to the computer in real-time. Then, the computer processes the images to provide perceived visual information. The image resolution used is 1280 x 720p and this affects the effective range for detection and estimation. Once the perceived visual information is fed into the feedback control loop, it yields the corresponding VTOL UAV command inputs which are roll, pitch, throttle (heave), and yaw. The commands are sent back to the UAV and then the embedded inner-loop autopilot controls the rotating speed of each propeller to achieve the commanded inputs.

The system used for processing is LENOVO Legion Y740-15IRH, which is composed of Intel(R) Core(TM) i7-9750H CPU @ 2.60GHz, 6 Cores, and 12 Logical Processors. It features an integrated NVIDIA GeForce GTX 1660 Ti 6GB Graphics and 8GB of LPDDR4 memory with a 128-bit interface. The Ubuntu 18.04 with Nvidia driver version 440 and CUDA version 10.2 are used. The WIFI communication is established by an external TP-LINK TL-WN722N Wireless N150 High Gain USB Adapter.

## 5.1 Vertical Landing Safety Tests

In order to demonstrate that it is safe to land vertically without matching the UAV attitude dynamics to platform motion, multiple landing tests are conducted. While the 6-DOF platform is simulating realistic and challenging ship motions, vertical landings are executed at random moments of motion.

### 5.1.1 Experimental Setup

To simulate the 6 DOF motions of the ship deck, a Servos and Simulation Inc Generic Motion System (model 710-6-500-220) with a 4 ft × 4 ft (1.22 m x 1.22 m) landing deck as shown in Fig. 5.2 is used. The ranges of roll, pitch, and yaw are ± 13°, ± 15°, and ± 16°, respectively. The ranges of a surge, sway, and heave are ± 10.2 cm, ± 10.2 cm, and ± 6.4 cm, respectively.

Figure 5.2: Servos and Simulation Inc. 6 DOF motion system with 4 ft x 4 ft landing deck

### 5.1.2 Ship Motions and Landing Tests Results

The first prescribed motion is the Oliver Hazard Perry Class frigate at the sea state of 6 and a wave direction of 60° introduced by Sanchez-Lopez, Jose Luis, et al. [54]. The frigate is 136 meters long and 14 meters wide and has a single flight deck. Sea state 6 is defined by the World Meteorological Organization (WMO) as a very rough condition that has a wave height of 4 to 6 meters. While the platform is undergoing this complicated motion, vertical landings are executed at random time instances as shown in Fig. 5.3. Solid red, blue, and green lines denote continuous angular (pitch, roll, and yaw) and linear (surge, sway and heave) motions of the 6 DOF platform. 50 landing tests are conducted in total and the motions of the platform at the time of each successful landing are depicted by dots. The randomly distributed landing timings demonstrate the vertical landing is safe at any moment of the Oliver Hazard Perry Class FFG frigate ship motions under the given conditions.

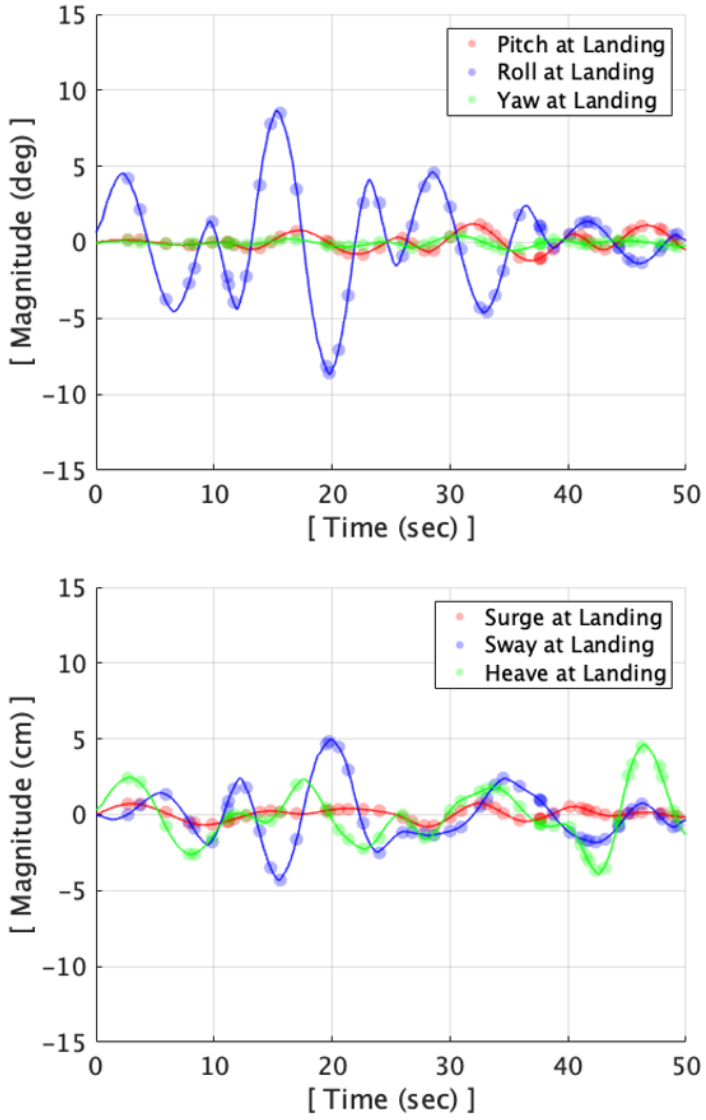



Figure 5.3: Oliver Hazard Perry class frigate motion and vertical landing moments depicted by dots

The second prescribed motion is the Navy helicopter ship landing limits defined by NATOPS. In the case of the FFG 7 Class Ships, which the Oliver Hazard Perry frigate belongs to, the ship motion limits for landing are set as ± 8° of roll and ± 3° of pitch. Even though the limits are defined by the maximum roll and pitch magnitudes, the frequency of the motion is also a crucial factor for the motions. According to the report for a similar size ship (length: 152.4 m, width: 15.2 m) motions conducted by the Sandia National Laboratory [109], the roll period is 10.1 seconds and

the pitch period is 6.5 seconds as shown in Table 5.1.

Table 5.1: Typical ship characteristics

| Ship Type | Length (m) | Width (m) | Roll Period (secs) | Pitch Period (secs) |
|---|---|---|---|---|
| Destroyer | 152.4 | 15.2 | 10.1 | 6.5 |
| Aircraft Carrier | 304.8 | 38.1 | 15.8 | 8.8 |

The maximum pitch and roll magnitude with the reported periods are applied to the platform and 50 vertical landings are conducted at random moments as shown in Fig. 5.4. Solid red and blue lines denote continuous pitch and roll motions of the 6 DOF platform. The dots denote the motion at the time of each successful landing. The randomly distributed landing timings demonstrate the vertical landings are safe and independent of the motion as long as it is within the operational limits.

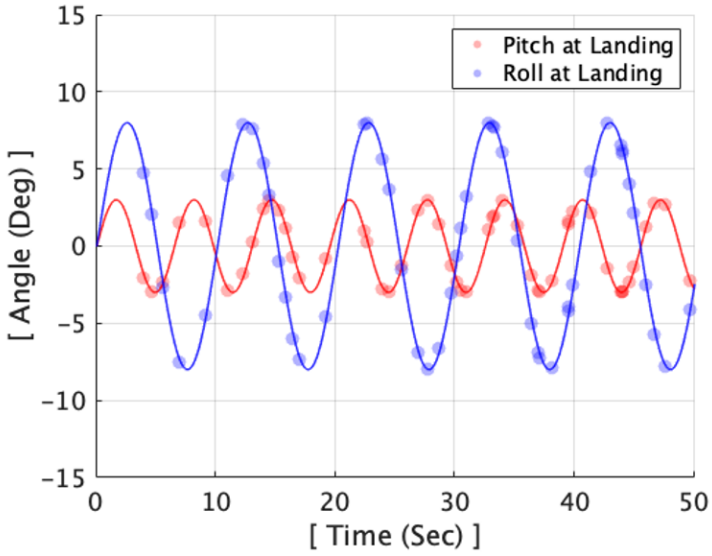



Figure 5.4: Motions of helicopter ship landing limits and vertical landing moments depicted by dots

## 5.2 Full Flight Tests

To demonstrate the tracking capability and landing accuracy of the developed autonomous vision-based control systems, extensive flight tests are conducted in indoor and outdoor environments. The flight tests consist of two different cases, which are tracking the checkerboard pattern visual cue by the gain-scheduled PID control system and tracking the horizon bar by the nonlinear control system and reinforcement learning based control system.

### 5.2.1 Experimental Setup

In the first flight test case, which is tracking the checkerboard, the platform is constructed with the visual cue that is installed parallel to the UAV's approach course as shown in Fig. 5.5. It has the checkerboard pattern visual cue that has 120 x 120 mm squares. The UAV captures the 9 corner points where the black and white squares cross each other to compute relative position and heading angle. The landing pad is designed as 3 x 3 ft and positioned 2 meters away from the visual cue. This setup has been used for the gain-scheduled PID control system.

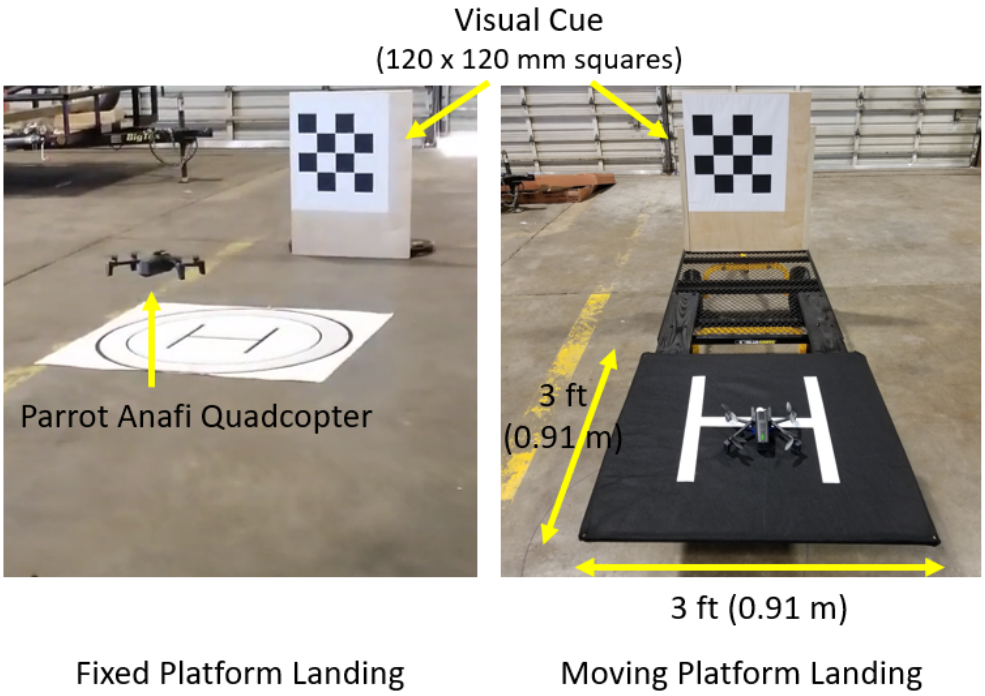



Figure 5.5: Checkerboard tracking experimental setup

Beyond the autonomous landing on the platform by tracking the checkerboard visual cue, the ship platform is constructed including the horizon bar and motion deck as shown in Fig. 5.6. The width, height, and length of the ship platform are 5 ft, 5 ft, and 10 ft, respectively. The horizon bar always indicates a perfect horizon, and the motion deck has its own 6 DOF motions in addition to the forward translational motion, which is similar to what would be experienced on a real ship. This setup has been used for the nonlinear control system and reinforcement learning-based control system.

Figure 5.6: Constructed ship platform with horizon bar and motion deck

In addition, a drum fan is installed for flight tests to demonstrate the disturbance rejection capability of the developed reinforcement learning control strategy as shown in Fig. 5.7. The fan blows 5 m/s of wind from both diagonal and orthogonal directions and the same constructed ship platform with the motion deck is used.


(a) Crosswind testing setup


(b) Diagonal wind testing setup

Figure 5.7: Disturbance rejection experimental setup

### 5.2.2 Tracking and Regulation Capability

The tracking capability can be evaluated by how closely the system can follow the setpoints. In this section, different control systems are implemented to achieve an autonomous approach and landing on the deck. Independent of specific control systems, it shares the common procedure that the UAV stops streaming, then conducts a quick landing once it satisfies the desired landing

condition. Thus, the results presented in this section show the relative positions, heading angle, and control inputs while the streaming is operational, which is from after take-off to before landing.

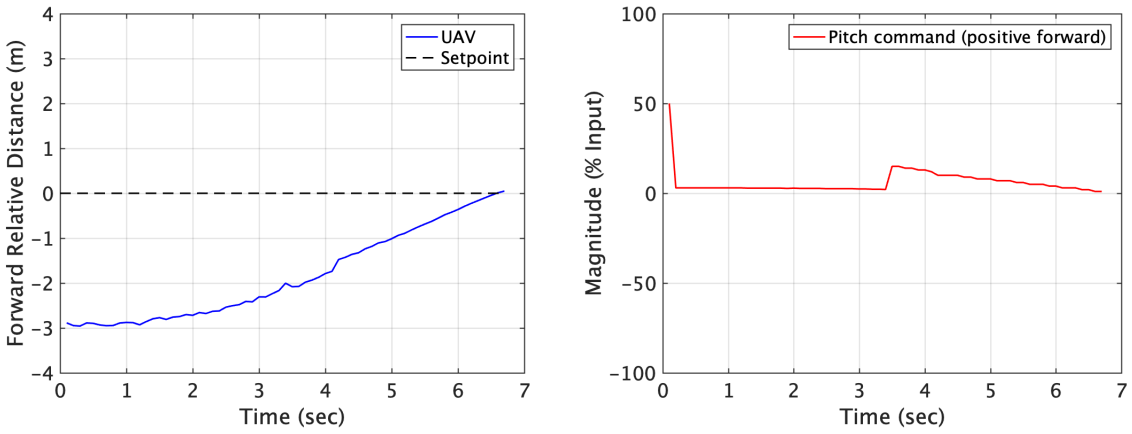
First, the gain-scheduled PID control system with various flight modes is constructed to track a stationary and moving platform by visually tracking the checkerboard pattern visual cue. The developed gain-scheduled PID control system has been applied to track a stationary platform and a platform moving forward. Along with the control system, the moving average filter is added to reduce the fluctuations in yaw estimation by taking 40 previous yaw data. In this checkerboard tracking case, the update rate is 0.1 seconds thus it takes 4 seconds for the moving average method to be in effect. The number of yaw data for the moving average method can be chosen properly according to the flight conditions; for example, the simulations took 10 previous yaw data for the moving average method. Even though realistic simulations for the same flight cases have been done, the flight tests still require more PID gain tuning process. The specific gain values obtained from the simulations are used as initial gains for flight tests, and the final gains are determined via multiple flight tests. More than 100 flight tests are conducted and the representative results are presented. The flight testing videos are available via the following links: Stationary platform landing [110], Moving platform landing [111].

Second, the nonlinear control system is designed to enhance the tracking capability where the time delay exists and the estimation error occurs. Extensive flight tests are conducted in realistic and challenging scenarios. The ship landing environment is mimicked by the landing pad via implementing the Oliver Hazard Perry Class FFG Frigate motions with helicopter ship landing limits. While the landing pad is in motion, the ship platform is also translating forward just like a real ship during a helicopter landing. During the approach and landing, the UAV makes transitions between the flight modes that engage with a particular vision and control method based on the perceived situation, then lands vertically without matching the UAV attitude to that of the pad. The maximum wind speed experienced was around 9 m/s (17.5 knots) and the wind direction kept changing over the course of these experiments. The GPS is only used to log positions during flight and not for control. The different flight test cases are shown in this video link video [112].

Third, the reinforcement learning-based control system is implemented to regulate the disturbance effectively in the presence of uncertainties.

#### *5.2.2.1 Gain-scheduled PID control for tracking a stationary platform*

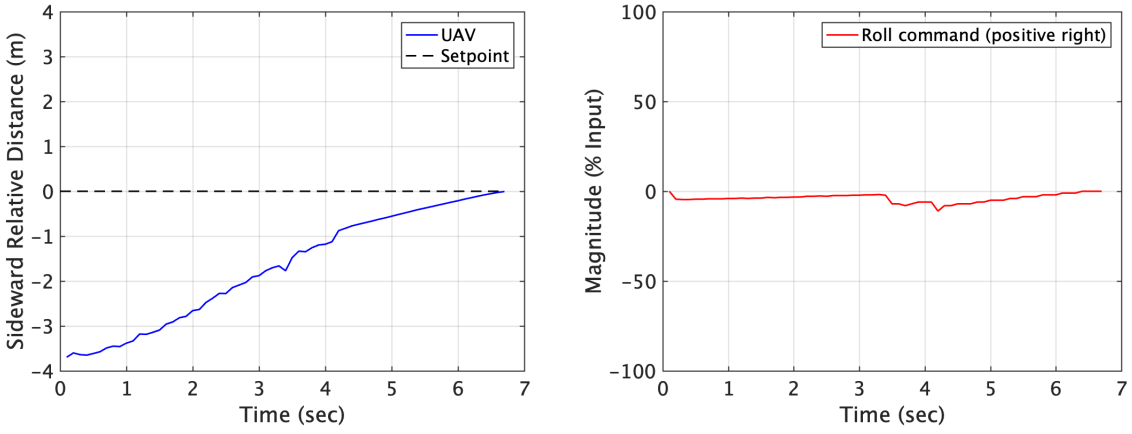
Multiple tests are performed with various initial positions and heading angles. As a representative challenging case of stationary platform landing, 45 degrees of initial yaw angle case is presented. The forward relative distance and pitch command input in time are shown in Fig. 5.8.




(a) Forward relative distance (b) Pitch command inputs

Figure 5.8: Forward relative distance and yaw command during stationary platform landing

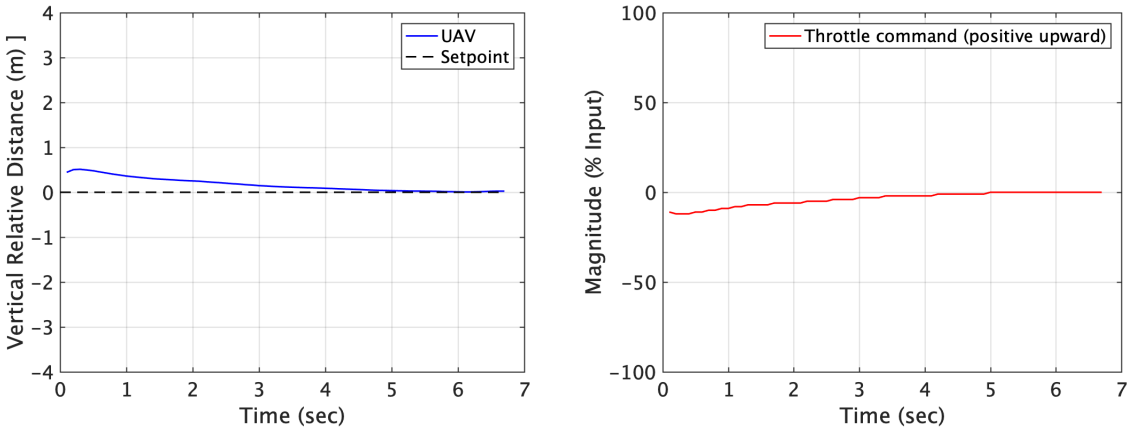
The time history of sideward relative distance and roll command input during tracking are shown in Fig. 5.9.




(a) Sideward relative distance (b) Roll command inputs

Figure 5.9: Sideward relative distance and roll command during stationary platform landing

Up to 3.4 seconds, the magnitudes of pitch and roll commands are small due to the activation of the level 1 yaw controller which gives priority to yaw correction. It takes 6.7 seconds to satisfy the landing conditions. The time history of vertical relative distance and throttle command input during tracking are shown in Fig. 5.10.

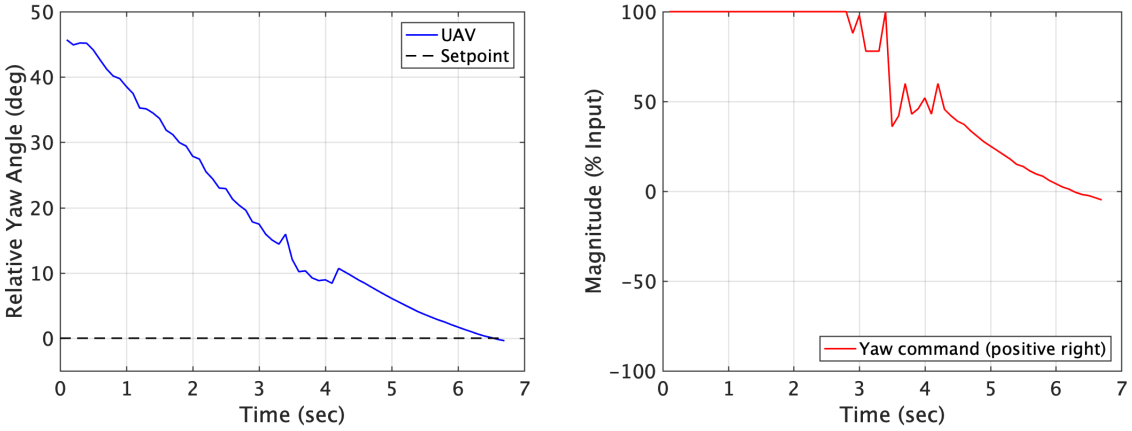



(a) Vertical relative distance

(b) Heave(throttle command inputs

Figure 5.10: Vertical relative distance and heave command during stationary platform landing

The relative yaw angle and yaw command input as a function of time during tracking are shown in Fig. 5.11.




(a) Relative yaw angle

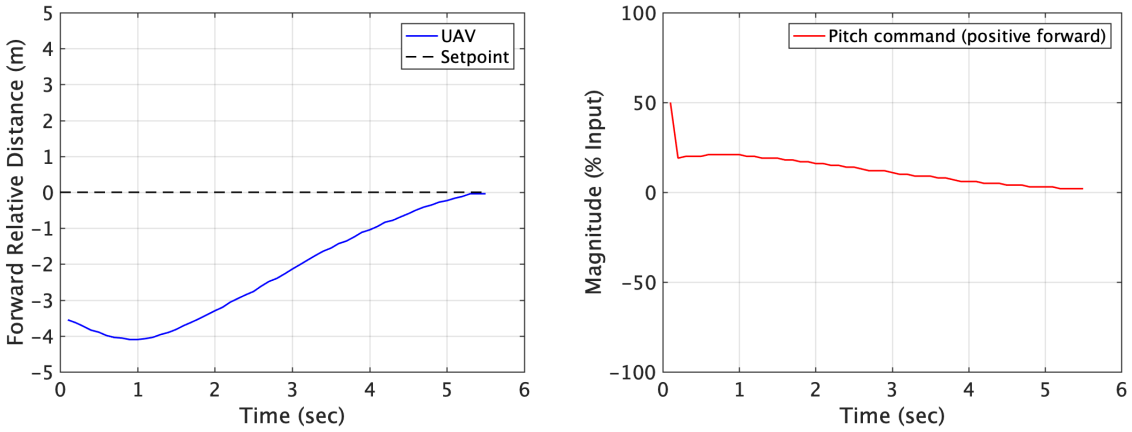
(b) Yaw command input

Figure 5.11: Relative yaw angle and yaw command during stationary platform landing

Initially, the UAV has 45 degrees of heading angle, which is near the maximum that can be assigned without losing the visual cue in the camera view. Due to the higher image quality of the real camera compared to the virtual camera in Gazebo simulations, the relative position and heading estimates in real experiments experience fewer fluctuations compared to the simulations. Thus, it commands yaw from the beginning instead of applying the safety mode for the directional approach, which had to be activated in the simulations. Until 3.4 seconds, the level 1 yaw controller corrects yaw and also reduces the magnitudes of pitch and roll commands to not lose the visual cue from the camera view when the yaw angle is large. After 4.1 seconds, the estimation and control commands become smoother due to the moving average method. By the time the UAV satisfies the landing conditions, the vertical distance is regulated to the visual cue height and the yaw angle is decreased to zero degrees.

#### *5.2.2.2 Gain-scheduled PID control for tracking a moving platform*

Multiple flight tests are performed while the platform is slowly moving forward and the results from a representative case are presented in detail. The time history of forward relative distance and pitch command input during tracking are shown in Figs. 5.12.

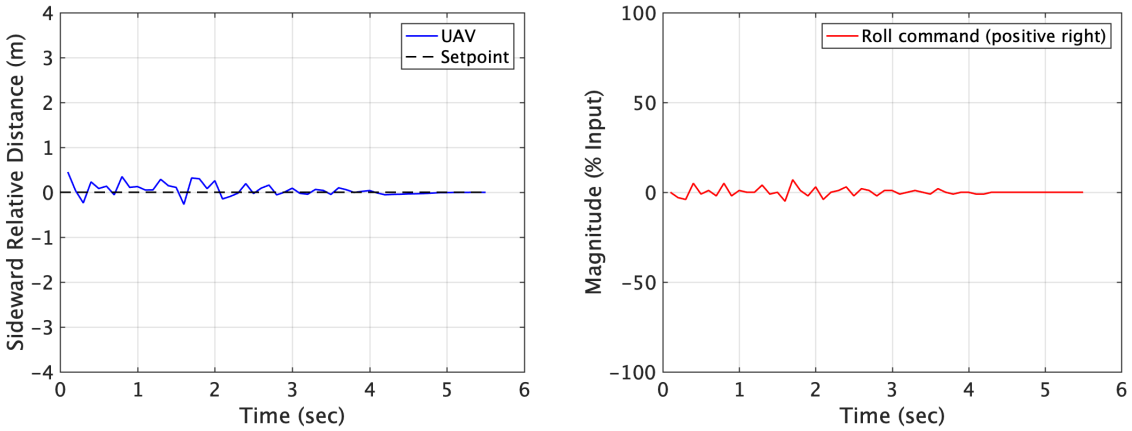



(a) Forward relative distance

(b) Pitch command input

Figure 5.12: Forward relative distance and pitch command during moving platform landing

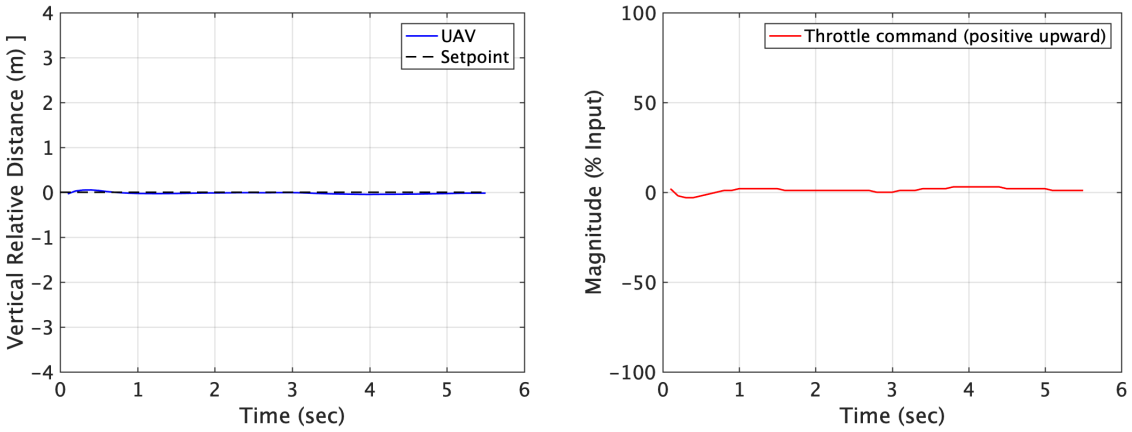
The time history of sideward relative distance and roll command input during tracking are shown in Fig. 5.13.




(a) Sideward relative distance (b) Roll command input

Figure 5.13: Sideward relative distance and roll command during moving platform landing

The time history of vertical relative distance and throttle command input during tracking are shown in Fig. 5.14.

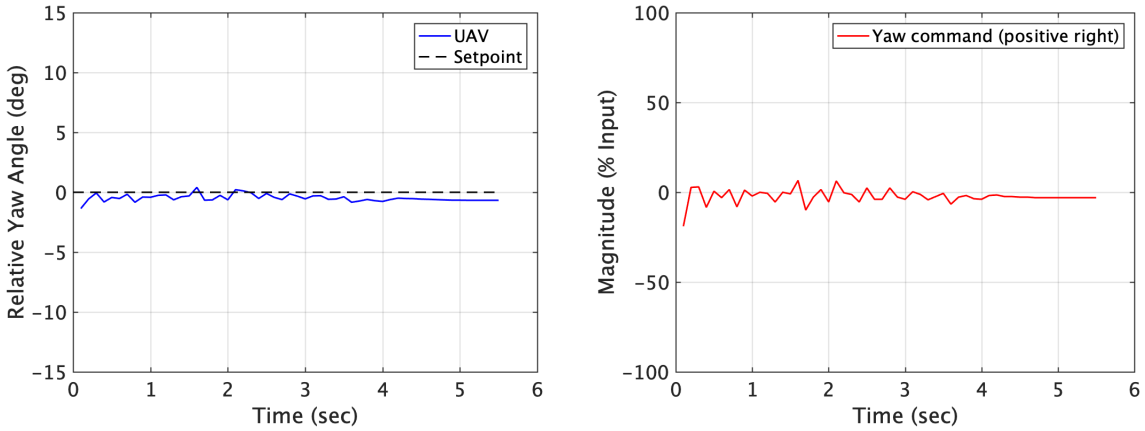



(a) Vertical relative distance (b) Heave(throttle) command input

Figure 5.14: Vertical relative distance and heave(throttle) command during moving platform landing

The forward distance increases due to the platform already moving in the beginning even before the UAV takes off, and the sideward distance is kept small throughout the flight. The setpoint for the vertical relative distance is the visual cue height on the moving platform and it is almost the same height when the UAV starts streaming, right after take-off. Therefore, it maintains the height until landing. The landing conditions are satisfied at 5.5 seconds, and then it stops streaming and executes landing.

The relative yaw angle and yaw command input during tracking are shown in Fig. 5.15.




(a) Relative yaw angle

(b) Yaw command input

Figure 5.15: Relative yaw angle and yaw command during moving platform landing

Since the initial heading angle of the UAV is intended to be parallel to the visual cue, it has small yaw maneuvers during the flight. The fluctuations that exist until 4.1 seconds are at the noise level which is $\pm$ 1 degree and become smooth after 4.1 seconds as the moving average method is activated. When the UAV satisfies the landing condition at 5.5 seconds, the final yaw angle is less than 1 degree.

#### *5.2.2.3 Nonlinear control for tracking a ship from long distance*

The tracking capability and landing accuracy of the developed vision-based autonomous flight control system are verified in challenging situations such as random initial positions, maximum

distance of 250 meters, realistic ship motions, communication latency, sensor noise, low visibility, and windy conditions. In this fully autonomous vision-based system, GPS and a magnetometer are used only to log positions and heading angles and not for control. During flight testing, the landing pad is mimicking the ship deck motions that are introduced in the previous section. First, flight tests are conducted to verify the maximum range and smooth transition between flight modes. The trajectories from take-off to landing are logged as shown in Fig. 5.16.

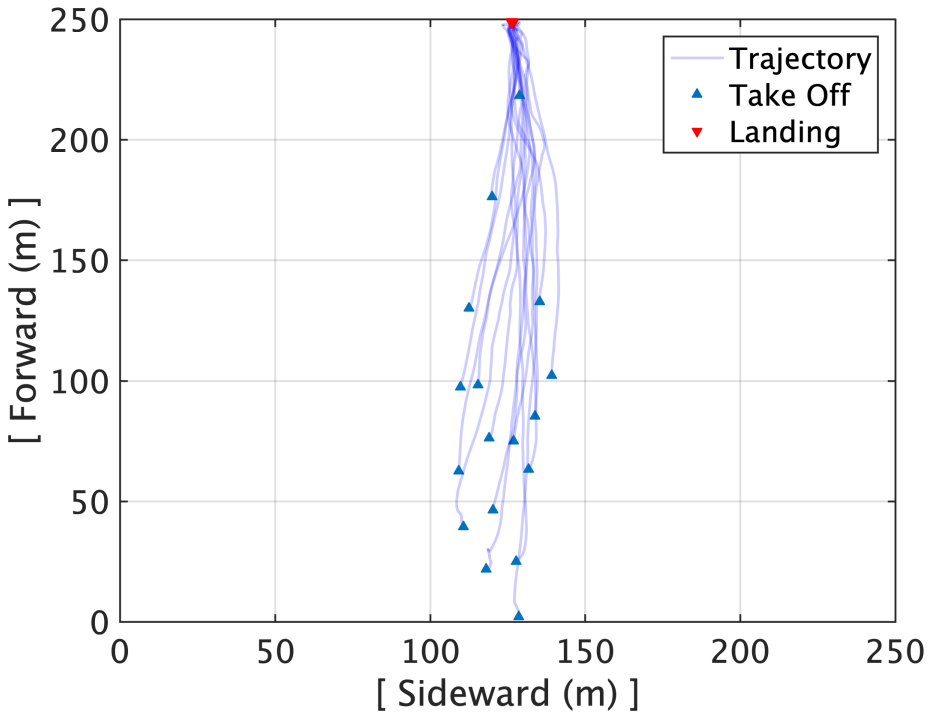



Figure 5.16: Trajectories of long-range tracking

The initial positions are widely distributed and the maximum distance between the UAV and the ship platform is approximately 250 meters. During flights, the flight modes are switched from the ship platform/horizon bar tracking to the corner points tracking depending on the distance. The results demonstrate stable long-range tracking capability in the presence of wind up to 9 m/s (17 knots).

The control inputs, distances, and heading angles of the maximum distance tracking case are logged after take-off until the execution of vertical landing. Each line denotes pitch, roll, heave, and yaw control inputs described as percentages that range from -100 to 100 and generated based on forward, sideward, vertical relative distances, and relative heading angle, respectively. Zero control input means neutral control input that maintains current UAV pose such as pitch, roll, altitude, and heading. While the UAV approaches the ship platform, the vision-based flight control system effectively flies the UAV by smoothly switching between the flight modes from the machine learning object tracking for ship platform and horizon bar to the rectangle corner points tracking, depending on the relative distance. The time history of pitch control inputs and forward relative distances is shown in Fig. 5.17.

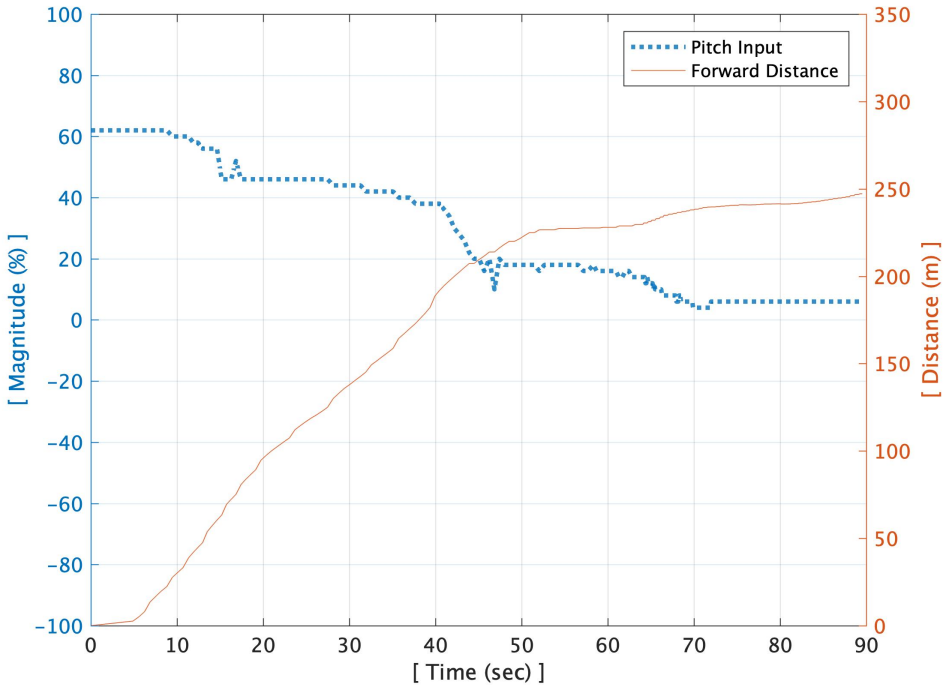



Figure 5.17: Pitch control inputs and forward relative distances as a function of time during long-range tracking

It demonstrates its maximum tracking range of 250 meters and approaching the ship platform at different flight speeds depending on the relative distance. The magnitude of control inputs has notable changes when the flight mode switches. The time history of roll control inputs and sideward relative distances are shown in Fig. 5.18.

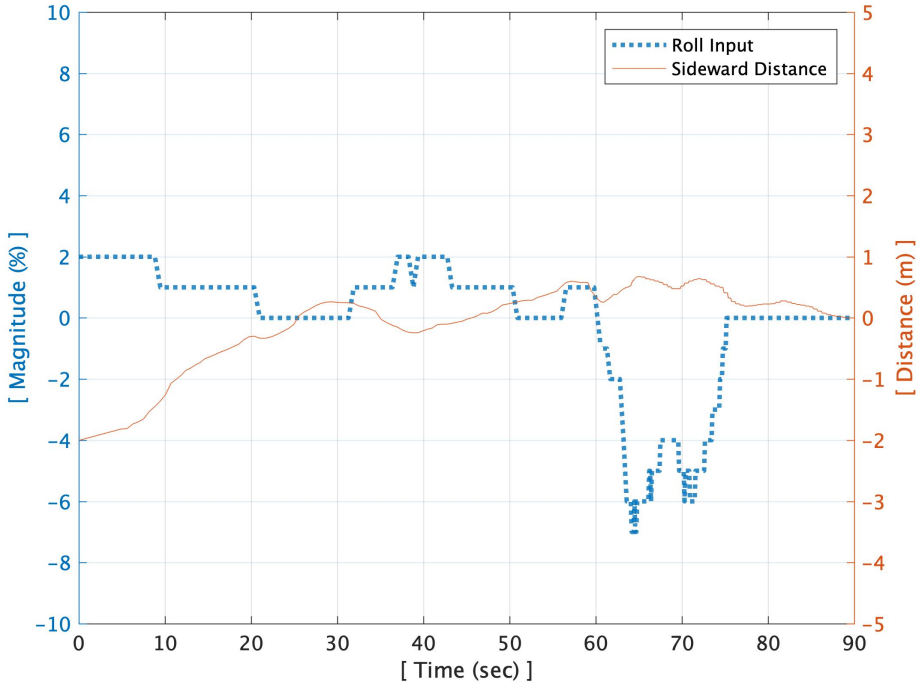



Figure 5.18: Roll control inputs and sideward relative distances during long-range tracking in time

It demonstrates the tracking of sideward distance by the roll control inputs that loosely control the distance at long distances and tightly controls the sideward deviation in the close range, which is appropriate for the present approach. Since the different flight modes are engaged depending on the forward relative distance, the magnitude of control inputs is different even at the same sideward distance. The heave control inputs with vertical relative distances and yaw control inputs with heading angles are shown in Figs. 5.19 and 5.20, respectively.

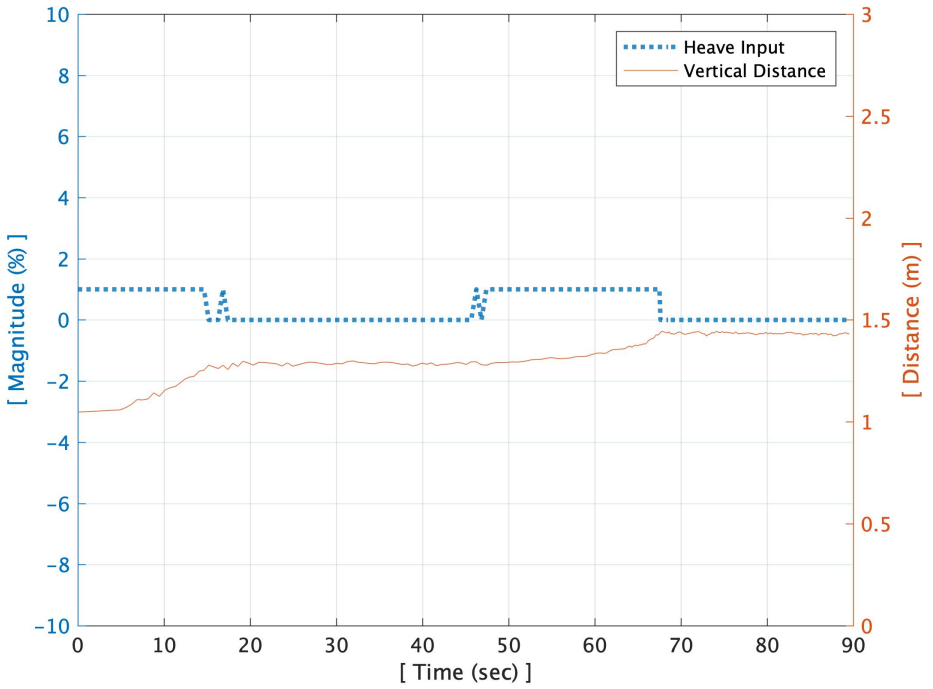



Figure 5.19: Heave control inputs and vertical relative distances during long-range tracking in time

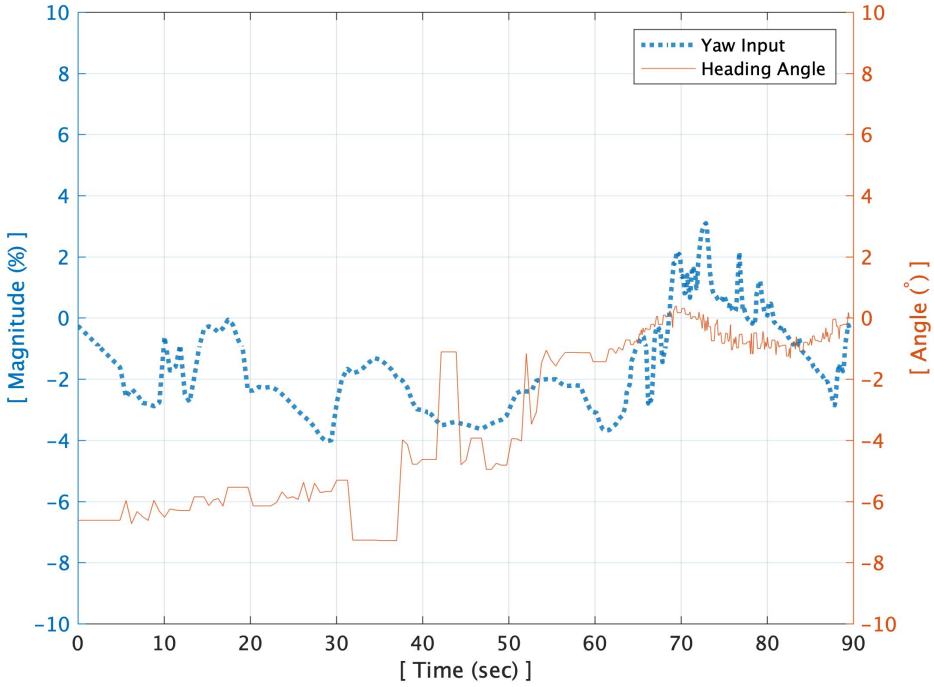



Figure 5.20: Yaw control inputs and relative heading angles during long-range tracking in time

Since the ship landing approach is nearly a horizontal approach the vertical distance change is relatively small. At a long distance, the heading angle is controlled by the yaw controller to match the known ship platform heading/course. As it gets closer, the yaw controller commands to achieve the zero heading angle based on the estimation by the close-range vision system.

#### *5.2.2.4 Nonlinear control for tracking a dynamically moving ship*

Flight tests are also conducted to verify the robust tracking while the ship platform is moving in different courses along with its own 6 DOF motions. The three representative trajectories of the ship platform and UAV are shown in Fig. 5.21.

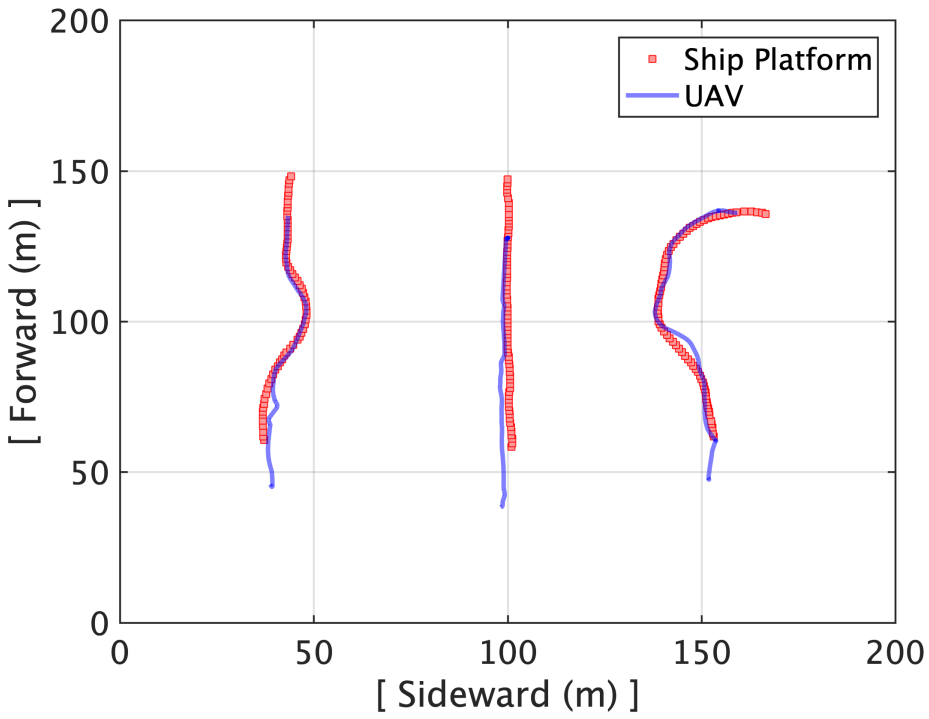



Figure 5.21: Trajectories of moving ship tracking

Red squares denote the trajectory of the ship platform and blue lines denote the trajectory of the UAV. It shows robust tracking while the ship platform is moving in straight, S-pattern, and 90° turn.

The control inputs, distances, and heading angles after take-off until execution of vertical landing are logged while the UAV is tracking the ship platform moving in an S-pattern. The pitch control inputs and forward relative distances of the S-pattern moving platform tracking are shown in Fig. 5.22.

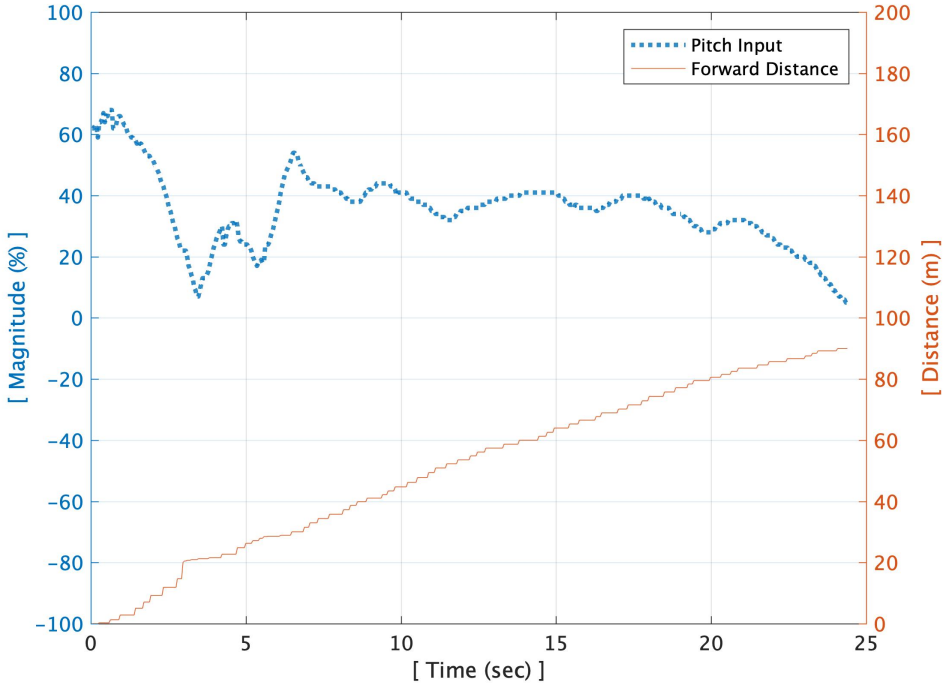



Figure 5.22: Pitch control inputs and forward relative distances during S-pattern tracking in time

The pitch control inputs are generated to approach the ship platform that varies its speed from 0 to 10 mph. It takes approximately 25 seconds to track the ship platform moving in a random S-pattern until it flies into the 35 x 35 centimeters of the area from the deck center, which is the designated landing condition. The pitch controller positively commands to track the ship platform at a stable speed while it moves 90 meters in the forward direction. The roll control inputs with sideward relative distances and heave control inputs with vertical relative distances are shown in Figs. 5.23 and 5.24, respectively.

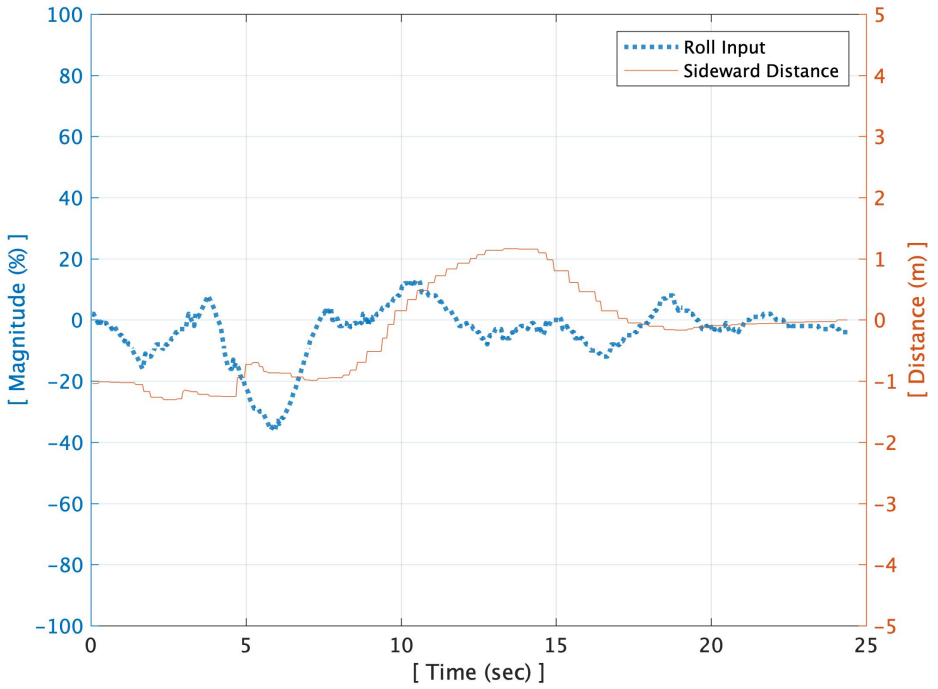



Figure 5.23: Roll control inputs and sideward relative distances during S-pattern tracking in time

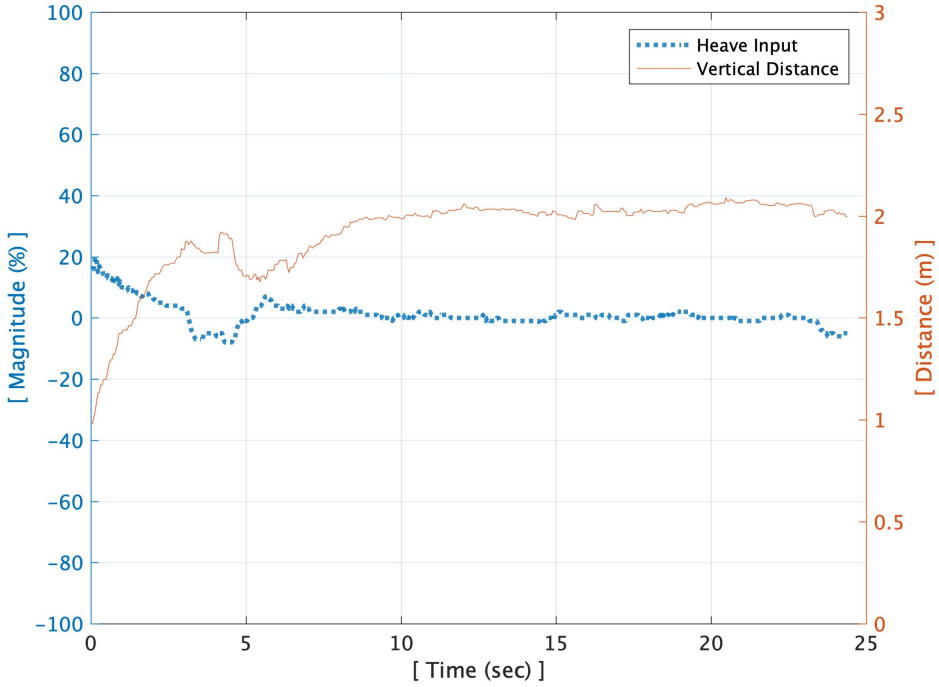



Figure 5.24: Heave control inputs and vertical relative distances during S-pattern tracking in time

While tracking, the ship platform moves in a random S-pattern, which changes the sideward relative distance unpredictably. Even in this challenging situation, it tracks the sideward distances effectively. It is also nearly a horizontal approach and therefore, the vertical relative distance change is relatively small. The yaw control inputs and relative heading angles for the S-pattern moving platform tracking are shown in Fig. 5.25.

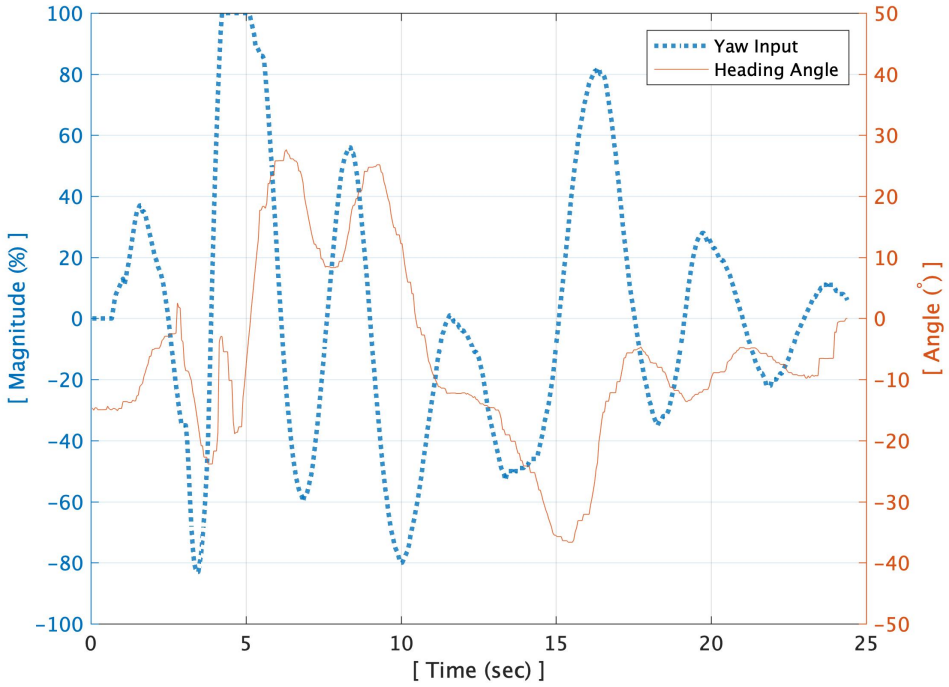



Figure 5.25: Yaw control inputs and relative heading angles during S-pattern tracking as a function of time

Since the ship platform changes the course abruptly up to 130°, the relative heading angles also change to a great extent. Accordingly, the corresponding yaw controller commands aggressively to achieve zero relative heading angles. Vertical landings are conducted once the UAV flies into the 35 x 35 centimeters landing threshold while the ship platform is in motion.

#### 5.2.2.5 *Reinforcement learning control for disturbance rejection (diagonal wind gust)*

Flight tests are conducted to demonstrate the disturbance rejection capability of the RL control strategy when the wind gust is suddenly imposed. During flight testing, the fan generates a 5 m/s wind gust at an angle of 45 degrees with respect to UAV heading while the UAV tries to autonomously approach and land on the ship platform. The performances of the developed nonlinear PID-based feedback control system with the Kalman filter and the deep RL control strategy are compared based on the sideward drift. The control inputs and the sideward drift using the RL control strategy are also examined. The sideward drift while using the nonlinear PID control and the RL control for the diagonal wind gust case is shown in Fig. 5.26.

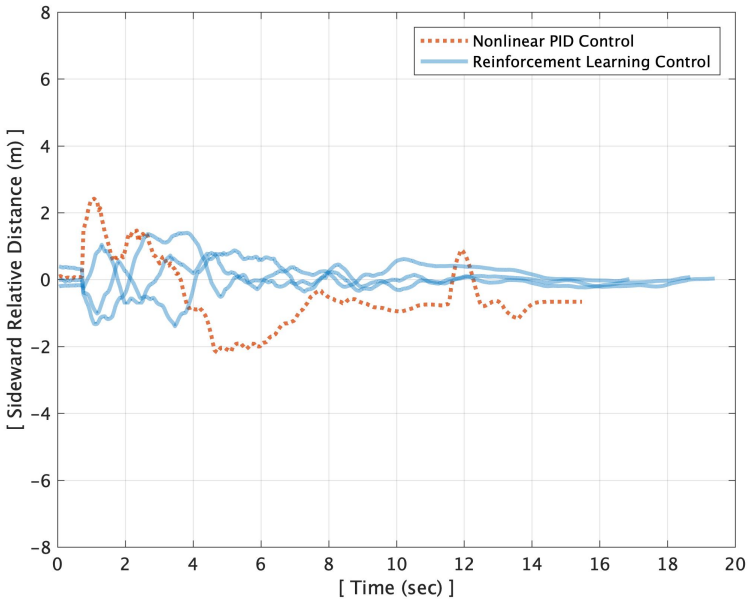
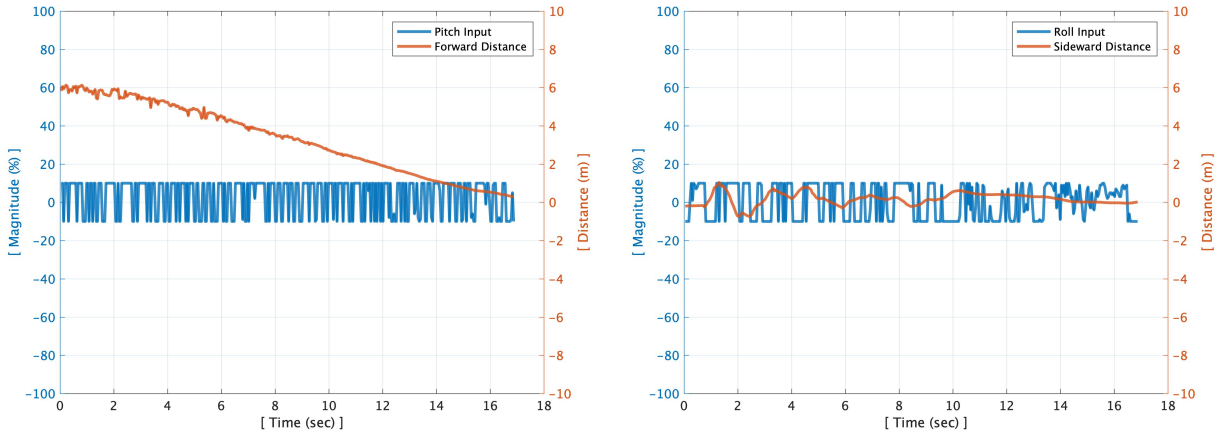



Figure 5.26: Sideward drift for the 5 m/s diagonal wind gust case

The flight tests using the nonlinear PID control experience more than 2 meters of sideward drift and fail to land on the landing deck. However, the RL control strategy demonstrates less than 1.5 meters of sideward drift and successfully lands on the deck repeatedly.

The relative distances and control inputs of successful RL control strategy are shown in Fig. 5.27. RL controller demonstrates robust tracking by rejecting the diagonal wind gust effectively from about 6 meters distance from the landing deck. Also, it keeps the sideward drift below 1 meter during approach and landing by actively commanding the roll control inputs of high magnitude and frequency. The trained control policy yields control inputs by deep neural networks that consider the current and 5 previous positions and velocities. Therefore, for an RL controller, it is not possible to explicitly correlate the relationship between the states and actions. However, the flight tests clearly demonstrated that the RL control effectively rejected the disturbance while approaching and landing.




(a) Forward relative distances and pitch control inputs

(b) Sideward relative distances and roll control inputs

Figure 5.27: Disturbance rejection of reinforcement learning control for the 5 m/s diagonal wind gust case

#### *5.2.2.6 Reinforcement learning control for disturbance rejection (crosswind gust)*

Flight tests are also conducted to demonstrate the disturbance rejection capability of the developed RL control strategy in a more challenging condition. During these flight tests, the fan is installed perpendicular to the approach course so that it experiences a sudden 5 m/s crosswind gust

while approaching and landing on the ship platform. The performance of the deep RL-based control strategy is compared with the nonlinear PID and Kalman filter-based feedback control system in terms of the sideward drift and control inputs. The sideward drift of the nonlinear PID control and RL control in the sudden crosswind case is shown in Fig. 5.28.

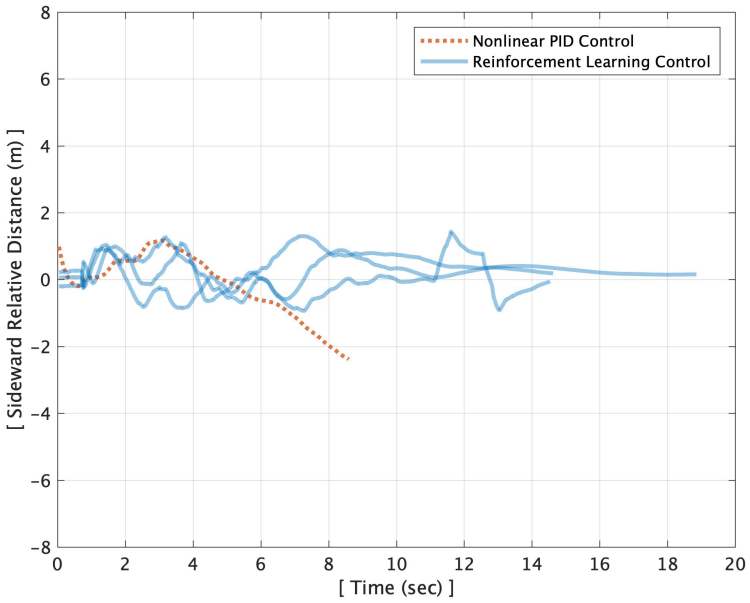
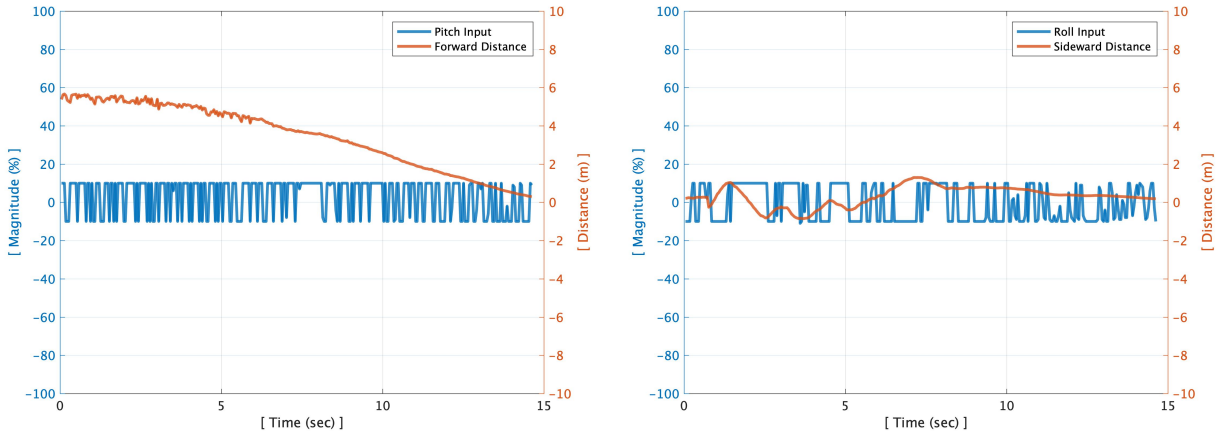



Figure 5.28: Sideward drift for the 5 m/s crosswind gust case

This sudden crosswind affects the sideward drift in both cases. The nonlinear PID control fails to reject the disturbance quickly enough, and therefore, the UAV loses the visual cue at approximately 8.5 seconds. On the other hand, the RL control keeps the sideward drift below 1.5 meters and successfully lands on the deck repeatedly.

The relative distances and control inputs of the RL control strategy are shown in Fig. 5.29. The RL controller demonstrates robust tracking by rejecting the crosswind gust effectively at 6 meters distance from the landing deck. It has a limited effect on forward drift but a distinguishable effect on sideward drift due to the direction of the wind. The pitch control inputs are mainly commanded to move forward in such a wind condition and the roll control inputs are commanded to minimized the sideward drift caused by the wind gust. It kept the sideward drift below 1.3 meters during the approach and landing phases.




(a) Forward relative distances and pitch control inputs (b) Sideward relative distances and roll control inputs

Figure 5.29: Disturbance rejection of reinforcement learning control the 5 m/s crosswind gust case

### 5.2.3 Landing Accuracy

The final landing point from the gain-scheduled PID control case for the stationary platform landing is shown in Fig. 5.30.

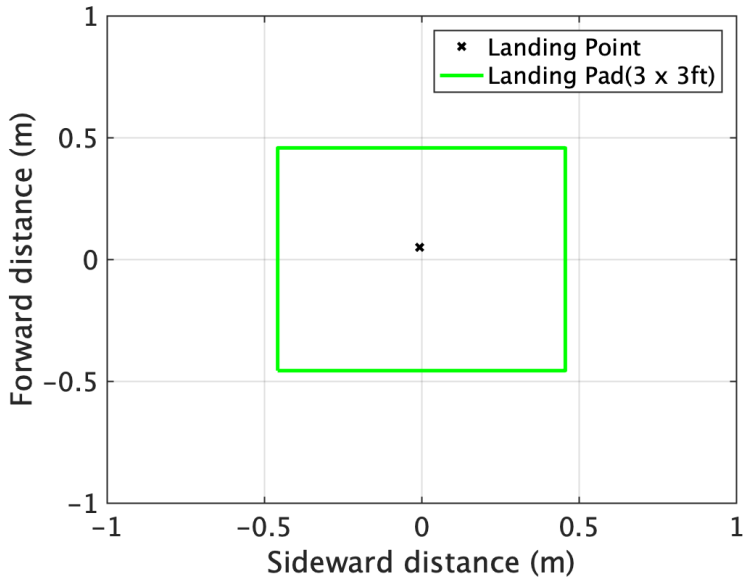



Figure 5.30: Final landing point in stationary platform landing case

It is designed to land when the UAV enters the landing threshold which is 15 x 15 cm from the landing pad center. The final landing deviation from the center is 5 cm. The final landing point from the gain-scheduled PID control case for the forward-moving platform landing is shown in Fig. 5.31, which demonstrates the landing accuracy of 4 cm.

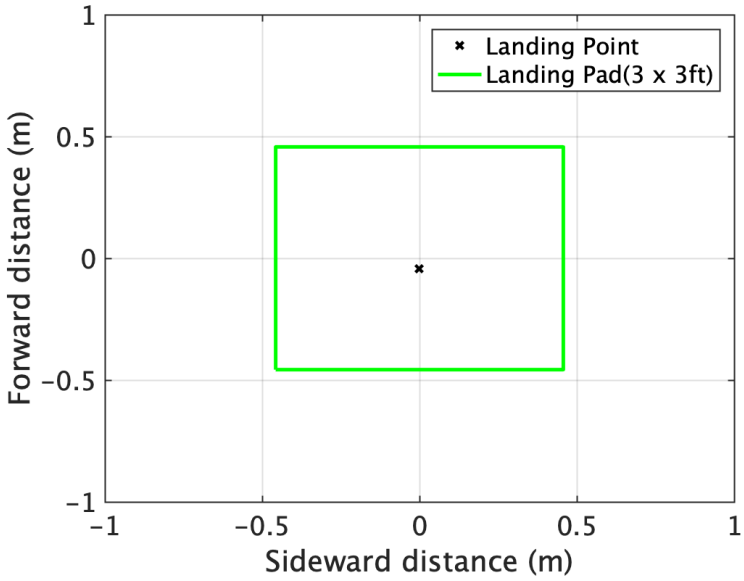



Figure 5.31: Final landing point in moving platform landing case

In the cases of landings on the sub-scale ship platform, the safe landing boundary is set by a 35 x 35 centimeters square from the deck center considering the UAV size, landing deck size, and safety margin. Therefore, once the UAV reaches the landing boundary the controller commands vertical landing while the deck is in motion. The final landing points are collected from the multiple flight tests of tracking a ship in long distance with random initial positions, different initial heading angles, and different weather conditions. As shown in Fig. 5.32, they are distributed randomly within the set landing boundary.

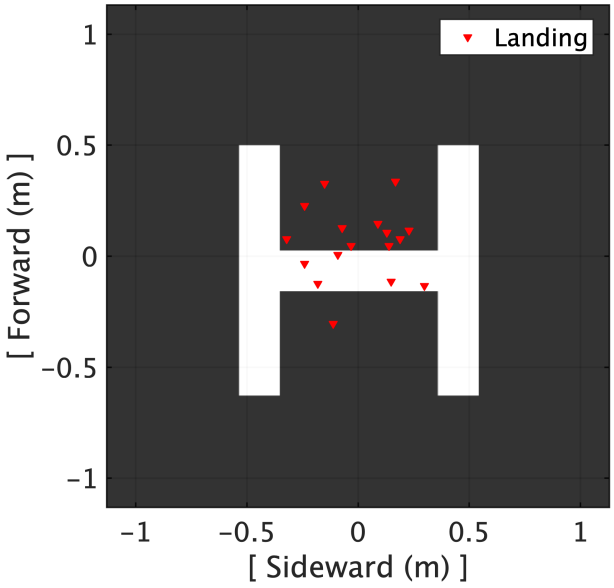



Figure 5.32: Landing points on the deck of long-Range tracking

The final landing points are also collected from the multiple flight tests of tracking a ship that is dynamically moving in forward, S-pattern, and 90° turn. They are distributed within the set landing threshold as shown in Fig. 5.33.

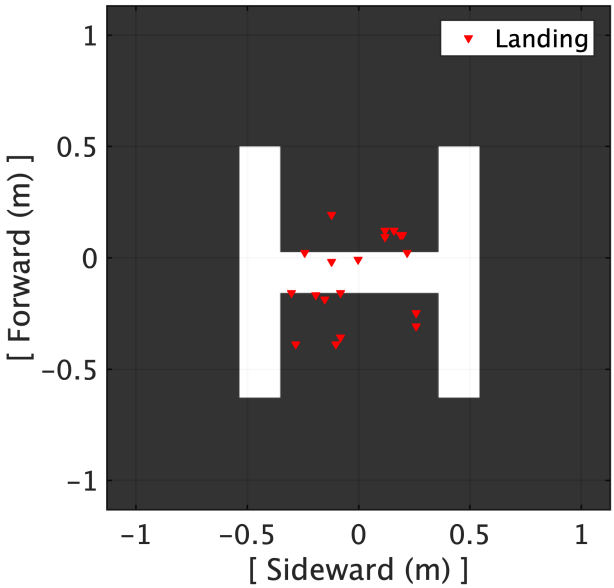



Figure 5.33: Landing points on the deck of moving ship tracking

The final landing points are also collected from the multiple flight tests of tracking a ship in the presence of wind gust where reinforcement learning control strategy is implemented. While approaching and landing, 5 m/s diagonal wind (45°) and cross-wind (90°) are suddenly imposed by a drum fan as shown in Fig. 5.7 of the experimental setup section. They are distributed within the set landing threshold as shown in Fig. 5.34.

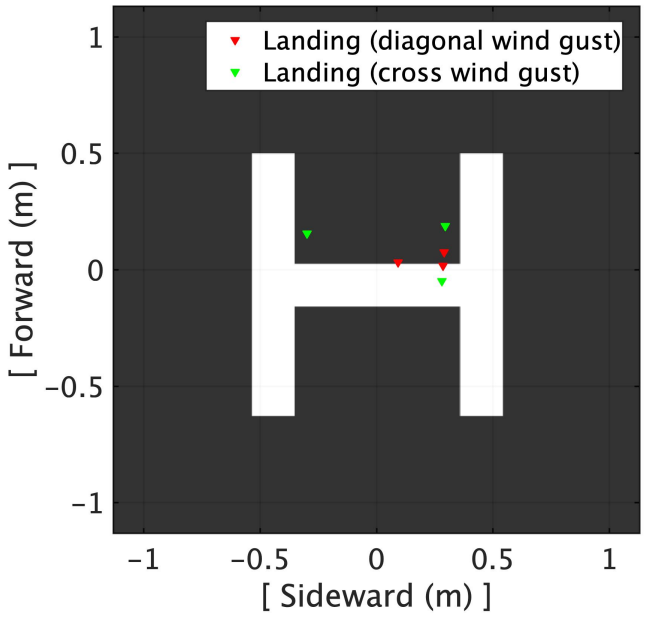



Figure 5.34: Landing points on the deck from the flight tests with wind gusts

# 6. SUMMARY REMARKS, CONCLUSIONS, AND FUTURE WORK

Fully autonomous helicopter ship landing has always been a highly desired and high-priority goal since the beginning of helicopter operation at sea. With the recent advances in autonomy and the emergence of UAV utilization in recent years, there have been several attempts to include vision-based autonomous ship landing capability on VTOL UAVs so that it could be operational in GPS-denied environments; and there have been a few VTOL UAV ship landing demonstrations. The ship landing methods that have been developed so far almost always try to track the deck motions for the landing control of aircraft, which is completely orthogonal to the Navy helicopter ship landing procedure that has been practiced over decades and proven to be safe.

To summarize the Navy ship landing procedure, the aircraft can fly using several conventional navigation aids on a ship such as TACAN, VHF Omnidirectional Range with a Distance Measuring Equipment (VOR/DME), and VHF Omnidirectional Range beacon with TACAN beacon (VORTAC) before reaching the Missed Approach Point (MAP) that is typically designated at 0.5 nautical miles away from the ship. However, it requires visual identification of the target ship at the MAP otherwise the pilot must abort the ship landing approach. Based on the visual contact of the target ship, it approaches the ship according to the ship approach chart. During the approach from the MAP to the ship, the descent angle is only $1^{\circ}$, thus it is nearly a horizontal approach. For the final approach and landing, the pilot stabilizes the aircraft attitude by referring to the gyro-stabilized horizon reference bar on the ship that indicates the true horizon independent of the ship motions. Once the helicopter hovers over the deck in a stable fashion, it lands vertically regardless of ship motions as long as the motions are within the predefined operational limits.

The key difference between the current Navy procedure and previous autonomous landing methods is the fact that in the Navy procedure tracking the ship deck motions is avoided and landing is executed vertically without attempting to match the aircraft attitude to the deck motions. In previous methods, in order to track the complicated and unpredictable ship deck motions, multiple powerful sensors and complex algorithms have been implemented, which are completely unnec-

essary if the automation of ship landing follows the Navy procedure. Also, following the deck motions is unsafe when the aircraft is near the deck since a small error can be catastrophic due to an impact by the deck causing a rollover. Furthermore, this approach of tracking the deck motion has serious fundamental limitations: (1) its application range is limited to the vertical space, and therefore, it cannot cover the required horizontal range of 0.5 nm (MAP from the ship) and requires separate methods for approach, and (2) since it does not follow the existing ship landing procedure, it will require establishing separate operation procedures, which could possibly decrease the operational speed and efficiency in Navy operations.

Considering all the aspects such as safety, capability, accuracy, simplicity, compatibility, and operational efficiency, this study developed a unified and practical VTOL aircraft autonomous ship landing solution based on an in-depth understanding of the helicopter ship landing procedure. To prove this novel autonomous ship landing concept, a helicopter flight dynamics modeling framework named TRAC was developed and validated using UH-60 flight test data. Then a linear flight dynamics model was extracted and ship landing simulations were conducted via trajectory tracking using an optimal LQR control system. To implement the autonomous ship landing on a real flying system, the technology has been developed in two major parts, which are the machine vision system and the control system. The machine vision system has been developed by taking advantage of state-of-the-art machine learning and classical computer vision techniques. Various control systems have been developed and their strengths and weaknesses have been demonstrated through actual flight testing. First, a gain-scheduled PID controller with different flight modes was constructed to select different PID control gains depending on the situation. Second, a nonlinear controller was designed to enhance the tracking capability in the presence of sensor noise and time delay. Third, a deep reinforcement learning-based control strategy was introduced to improve robustness in the presence of wind gusts. Through this approach, it was able to obtain the autonomous vision-based flight control system that followed human pilot's ship landing procedures but with better precision.

Multiple flight tests are systematically conducted to verify the safety of the vertical landing

maneuver, long-range detection, robust tracking capability, and landing accuracy. The UAV lands vertically independent of ship motions in the same manner as a Navy helicopter lands on a ship. More than 100 landing tests are successfully conducted on a moving deck, which mimicked the realistic and challenging 6 DOF ship motions. The developed control system demonstrated robust tracking capability and precise landing during a wide range of realistic scenarios such as random initial positions, complicated ship motions, communication latency, sensor noise, and in the presence of winds up to 20 mph. The conclusions from different areas of the present study are summarized below.

## 6.1 Conclusions

### 6.1.1 Rotorcraft Flight Dynamics Modeling Framework

A comprehensive rotorcraft flight dynamics modeling framework was developed based on the UH-60 helicopter and named Texas A&M University Rotorcraft Analysis Code (TRAC). This is a complete software package that could perform trim analysis, extract linearized models, and simulate flights by using LQR optimal control system. The dynamic/aerodynamic models of each component of a representative helicopter (UH-60) have been developed individually and integrated to form a complete helicopter model. Thus, it is also convenient to add or remove a component to analyze the effect or configure a new aircraft. It has adopted empirical relations derived from wind tunnel data to improve prediction results. The prediction results including helicopter power, attitude, rotor blade angles, and control inputs have been validated with the US Army UH-60 flight test data. Following are the specific conclusions derived from this study:

1. Based on the comparison of the predicted power curve with the flight test data, it is observed that the present linear inflow dynamics model accurately predicts power at higher forward flight speeds; however, it under-predicts the power at low forward flight speeds. Previous modeling methods using the free-vortex wake models or CFD-coupled models could capture rotor-fuselage interference more accurately thus showed better predictions at the low-speed range where the interference has a nontrivial impact on power. However, more complicated

inflow models increase the computational loads and it does not guarantee better predictions in all the trimmed flight conditions. Actually, TRAC predicted the pitch angles with 1° accuracy over a range of forward flight speeds via rotor analysis using a linear inflow model coupled with the accurate fuselage drag model derived from wind-tunnel test data. Therefore, modeling fidelity cannot be determined by one factor, and modeling the flight dynamics of a helicopter, which is a complex, nonlinear, and highly-coupled system, requires multiple selections of aerodynamic/dynamic models considering the flight conditions, desired accuracy, and computational speed. For the ship landing simulation, TRAC was able to extract sufficiently accurate helicopter models for various flight maneuvers such as hovering, forward flight, climb and descent, and coordinated turn.

2. Component dynamics/aerodynamics models were developed with respect to their own body-fixed frame, and then their force and moment contributions were translated and transformed to the predefined reference point. Also, the coordinate transformations were coded independently and not as part of the component models. In this way, adding or modifying a component model can be done without changing any other portion in TRAC. Due to its modularity, it was also used to develop a completely new aircraft model which is a coaxial rotor personal aerial vehicle (PAV), Harmony Aria [28]. Also, it is currently being used in a different study to investigate the benefits of a mission-adaptive morphing rotor. Hence, this modeling method can expand its applications to various studies.

3. TRAC includes the ship approach and landing simulation conducted using an LQR control system. The LQR method figures out the optimal gains depending on the weights assigned for references (states) and control inputs. The weights on references determine which states are going to be controlled and priority among them. Also, the weights on control inputs determine the priority among control inputs to achieve the desired flight maneuvers. To determine such weights, the author's empirical knowledge from the helicopter piloting experience was used as well as trimmed control inputs and distinctive states obtained from TRAC. It can

be considered as a successful flight simulation only when the desired helicopter behavior is achieved by appropriate control inputs. For example, in order to descend during forward flight, the main control input should be a collective down command rather than a pitch forward command that causes descent and increase of forward flight speed. Thus, it is important to select appropriate weights in the LQR control system to obtain realistic simulation results rather than simply tracking the setpoints.

### 6.1.2 Machine Vision Systems

The machine vision system was developed to provide the visual information perceived in a given situation. The objects to detect were determined based on the visual cues that a helicopter pilot refers to while approaching and landing on a ship. The long-range tracking target is the ship as the pilot visually confirms the ship's location from a long distance to set a course and speed for the approach. At the proximity of the ship where the horizon reference bar becomes visible, the pilot controls the helicopter by referring to the horizon bar, which remains horizontal independent of the ship motions, for the final approach and vertical landing. Accordingly, the vision system utilized machine learning-based object detection for long-range ship tracking, and classical computer vision techniques for the estimation of aircraft relative position and orientation using the horizon bar during the final approach and landing phases. Following are the specific conclusions derived from this study:

1. The long-range vision system developed based on the state-of-the-art machine/deep learning algorithm YOLOv3 detected the 6 x 6 ft sub-scale ship platform from 250 meters away. Since the detection range is proportional to the size of an object shown in the image, it means that the vision system can detect a typical small ship whose rear-side occupies an area of 50 x 50 ft from 17.3 kilometers (9.3 nautical miles) away. This is a 10 times greater detection range than the classical computer vision systems and 18 times greater detection range than the required minimum visual ship identification distance as defined by the ship approach chart.

2. Even though one machine/deep learning-based object detector can detect multiple objects, the computational load is higher and the accuracy is lower compared to single object detection. Thus, two object detectors were developed to detect the ship and horizon bar, respectively. By this approach, the object detector did not need to distinguish one object class/name from the others, thus the chance of false classification was completely removed. Furthermore, in the case that one object detector fails to detect the designated object, the other object detector can provide the detected object information to the control system as a backup.

3. The close-range vision system was developed based on classical computer vision techniques to guarantee fast and reliable detection of the corner points on the horizon reference bar. The existing corner detection methods were tested; however, they easily detected false corners. Hence, the detection algorithm has been uniquely developed to detect corners precisely from different distances, different perspectives, and different light conditions. This was achieved by using sequential steps including image filtering, contour and corner detection, and screening the detected corner points by their spacing and slopes. It demonstrated approximately 15 times faster processing time than the machine/deep learning-based vision system used for long-range detection.

4. In the proximity of the ship, the close-range vision system also estimated the relative position and heading angle. To reduce the fluctuations in estimation, the moving average filter was added first and replaced with the single-state Kalman filter. The estimated values were validated through the VICON motion capture system, which provided true position and attitude data. The estimation demonstrated sub-centimeter accuracy in position and sub-degree accuracy in heading angle, which was sufficient to achieve precise landing.

### 6.1.3 Control Systems

Vision-based control systems were developed to generate situation-adaptive control inputs for excellent autonomous ship approach and landing performance. To achieve such an objective, three

different control strategies were introduced during the development process. First, a scheduled-gain PID controller with flight modes was implemented to yield the appropriate control inputs by engaging different sets of gains based on the position and heading angle. Second, a nonlinear controller was developed based on the idea of nonlinear gain variation and a probabilistic approach to limit the impact of incorrect estimations and ensure its tracking capability even in the presence of high latency. Third, the state-of-the-art reinforcement learning control strategy was applied, which can respond to disturbances (wind gusts) quickly with minimum drift. The specific conclusions derived from this study are enumerated below:

1. The gain-scheduled PID controller along with different flight modes allowed a smart, situation-adaptive control approach for the approach and landing maneuvers. Based on the flight conditions and the relative distance to the landing platform, the controller selected one of the following modes: high-speed gross tracking, precise tracking at low speeds, scanning for the visual cue, collision prevention, or immediate landing. The different flight modes were verified under various flight scenarios including the landing platform moving at high speeds, as well as following circular and S-shaped trajectories. The results confirmed that the proposed visual cue tracking landing method is able to achieve safe, accurate, and robust landings repeatedly.

2. The biggest challenge to implementing the machine/deep learning-based object detection for the real-time autonomous flight was the time delay. To cope with the time delay issue, the nonlinear control system was constructed that responded less sensitively to errors around the setpoint and aggressively to large errors. This was achieved by using an exponential variation of feedback gain with error. The superior performance of the nonlinear controller was demonstrated through comparison with other control systems such as linear P controller, linear PD controller, and linear PID controller. The linear control systems were not able to suppress the oscillations arising from the slow update rate, which could trigger instabilities. However, the developed exponential nonlinear control system achieved stable tracking performance effectively in the same situation. This approach enabled the aircraft to stay in the

correct flight course while approaching the ship platform from a long distance.

3. Even after going through the configured Kalman estimator, large/false estimation errors can still occur from time to time. To prevent responding to such non-physical estimations, the probabilistic nonlinear controller was developed. It probabilistically perceives if the estimation is physically feasible, based on the normal distribution curve and known UAV characteristics. The derivative control term is more sensitive than other terms since it is multiplied by the error difference at each time step, thus one small error can generate large/false control input. However, multiplying the estimation value by its probability of occurrence can effectively reject responding to false estimations. Using this approach, the controller never generated abrupt large control inputs even when the vision system provided inaccurate estimations. This greatly improved the robustness of tracking.

4. Sudden wind gust is an even more challenging condition than a steady strong wind. The developed reinforcement learning-based control strategy was able to react quickly when such a transient disturbance occurred. This control strategy is completely different from conventional control systems. Instead of developing control algorithms that specify what actions should be taken in a given situation, the RL-based controller learned the best control actions through training in a broad range of wind conditions. By examining multiple training cases, the optimal set of states were identified and the reward function was developed, which is substantial for successful training. This control policy was obtained from the training in the Gazebo simulation and directly implemented in the flight testing. Compared to the flight testing results of feedback control systems, it reduced the drift by 50% when a 5 m/s of sudden wind gust was imposed during the ship approach.

### 6.1.4 Flight Testing

Extensive flight tests were conducted to prove the safety of vertical landing without considering ship motions and to demonstrate the tracking capability and landing accuracy. To this end, multiple vertical landings were executed while the 6 DOF platform was simulating realistic and challenging

ship motions. The tracking capability and landing accuracy were verified through full flight testing in situations such as random initial positions, adverse weather conditions, and dynamically moving ship conditions. The specific conclusions derived from flight testing are given below:

1. The 6 DOF platform simulated the Oliver Hazard Perry Class frigate ship at the sea state of 6 and a wave direction of 60° and NATOPS helicopter ship landing limits with small ship pitch and roll motion periods. More than 100 vertical landings were successfully conducted at the random moments of such challenging ship motions. Considering the size of the VTOL UAV, the landing boundary that triggers the vertical landing maneuver was set as 35 x 35 cm from the deck center and this ensured safe landing while the ship's maximum roll hit 9° and maximum pitch reached 5° with small ship motion frequency. Thus, it was concluded that vertical landing that does not follow the deck motions is a safe and appropriate ship landing manner.

2. Full flight testing has been conducted in challenging situations such as random initial positions and heading angles, communication latency, sensor noise, low visibility, and strong winds. The maximum distance was 250 meters away from the ship and the maximum initial heading was 45°. Also, the randomly moving ship platform changed its course up to 130° in 2 seconds and changed its speeds from 0 to 10 mph, while the landing pad was simulating the ship deck motions. The adverse weather affected flight conditions such as the low visibility of 150 meters (490 ft), the strong wind of 17 knots (9 m/s), and different lighting conditions. The communication latency varies from 0.03 to 0.5 seconds, depending on the engaged vision system. Even though the flight tests were conducted under such extreme conditions, the developed robust autonomous vision-based control system demonstrated stable flight performance during approach and landing with minimum overshoots.

3. The factor that affected landing accuracy was the size of the assigned landing boundary and there was a trade-off between the accuracy and flight time. Even in a scenario where the UAV had to fly at its maximum speed, it landed with an accuracy of 4 cm when a landing boundary

was set to 5 x 5 cm from the landing-pad center. Multiple flight tests were safely conducted, which showed landing errors of less than 5 cm. However, the landing boundary was finally set as 35 x 35 cm because that was sufficient to land on the deck safely considering the size of the VTOL UAV and the deck. In challenging flight conditions, the UAV was able to land at random points within the landing boundary. In other words, the study showed that the landing accuracy is not limited by the developed control system, but determined by the set landing boundary. Therefore, the landing boundary can be selected depending on the priority between the flight time and landing accuracy.

## 6.2 Contributions to the State of the Art

Since the beginning of helicopter shipboard operations, numerous ship landing-related accidents have occurred and too many souls have been sacrificed. Needless to say, this led to a yearning for autonomous ship landing technology. Even though there have been several studies for autonomous ship landing, they commonly have regarded it as an extension of the moving-platform landing problem without deeply considering the unique characteristics of ship landing. Hence, automation methods have been developing in a different way from the current Navy helicopter ship landing procedure causing unnecessary complexity, instability, and limited capabilities. To the author's best knowledge, this study is the first attempt in history to automate ship landing in a way that closely mimics the established Navy helicopter ship landing procedure. The main contribution of the study is the demonstration of the feasibility of this novel autonomous ship approach and landing method. Furthermore, this study has made technical contributions to the state of the art in the development of rotorcraft flight dynamics modeling framework and autonomous vision-based control system as below.

Over the past several decades there have been many research efforts focused on modeling a helicopter. As a result, the state-of-the-art helicopter flight dynamics modeling frameworks demonstrate high fidelity but inevitably incur high computational loads. However, TRAC has been developed as a comprehensive flight dynamics tool focusing on fast computation with reasonable fidelity, which can conveniently model arbitrary rotorcraft by adding or re-configuring compo-

nent models from the baseline UH-60 helicopter model. Even though the component models of rotorcraft are highly coupled, TRAC separates out individual component models, and therefore, modifying or replacing a component can be done without changing any other portion in TRAC. Due to its modularity, fast computation speed, and fidelity, it is appropriate for analyzing the effect of design parameters at an early stage of aircraft development and investigating the effect of newly designed lifting surfaces.

In recent years, the area of machine learning has been expanding rapidly and its application areas are increasing exponentially with the increase of computing power. The machine/deep learning-based vision is a powerful method to detect an object of interest. Previous studies focused on developing algorithms to detect an object and classify it. However, the present study went beyond detection, where it incorporated the machine/deep learning-based vision into the autonomous flight control system for the ship/moving platform approach, which is a novel contribution of this study. To achieve real-time autonomous flight, a nonlinear control system that has exponential variations of gain was designed. This involved coupling an autonomous control system with the machine/deep learning-based vision to ensure the control strategy is effective in the presence of time delays. The same approach can be applied to other machine/deep learning vision-based autonomous systems.

Even though the machine/deep learning-based vision has significant advantages, the classical computer vision technique is still very powerful to process images fast and accurately as instructed by an explicit algorithm. The present study included the development of a method to detect specific corner points by combining image filtering, contour detection, and contour detection as well as the screening process. This uniquely developed detection method was able to precisely find the corners and to avoid the false corner detection that occurred when other existing corner detection methods were used. According to the experimental study, it was superior to other existing corner detection methods in both accuracy and reliability. Although the application range was smaller than the machine/deep learning-based detection, the processing time was 15 times faster than the machine/deep learning-based vision. Hence, the developed detection method can be used for other

applications where it is important to detect particular points fast, reliably, and accurately at close distances.

Obtaining information from the vision sensor inevitably raises concerns about noise. The common approaches to reducing the noise are adding a moving average filter or Kalman filter as a state estimator. The present study also reduced the noise level by configuring a moving average filter and Kalman filter; however, the filtered values were still affected by false/large estimation errors yielding unstable control inputs. This was not enough to achieve robust flight performance. Therefore, a nonlinear control system was developed to reject responding to non-physical estimations by using the probability of its existence. This novel probabilistic nonlinear control system is appropriate for any control system that has a derivative term and needs to avoid unnecessary abrupt large control inputs caused by the false/large error.

Conventional feedback control systems were not sufficient to ensure flight performance when a sudden wind gust is imposed because they require different sets of gains depending on flight conditions and even scheduling the gains will not cover all the possible cases. However, the state-of-the-art reinforcement learning-based control strategy can respond quickly to such situations since it has experienced all the possible cases during the training session. This study investigated how to train such a control policy, and demonstrated the apparent effectiveness with respect to conventional feedback control systems via simulations and flight testing in presence of wind gusts.

### 6.3 Recommendations for Future Work

First and foremost, the automation of ship landing should be developed considering its unique characteristics. This is not just landing on a moving platform that has 6 DOF motions, but a comprehensive task that should comply with the established procedure including horizontal approach, visual identification, and vertical landing without following the deck motions. The perturbations to aircraft dynamics induced by the unsteady wake from the ship super-structure should also be considered. Throughout the study, the overwhelming capabilities of the proposed method compared to typical moving deck tracking methods in the aspects of safety, accuracy, simplicity, compatibility, and operational efficiency were demonstrated. However, there exists technical areas that can be

improved as detailed below.

At the earlier stage of this study, the rotorcraft flight dynamics modeling framework TRAC was developed. Although it achieved reasonable accuracy at a fast computation rate, the inflow model can be improved by coupling with CFD analysis or free-vortex wake model to capture the fuselage-rotor interference accurately. This will significantly improve the power predictions in the low-speed forward flight range. However, it can increase the computational loads. To avoid this, higher-order polynomial functions or look-up tables can be derived from aerodynamic interference predictions by CFD or free-vortex wake models. The effect of ship wake on rotor aerodynamics should also be modeled. Furthermore, the current rigid blade model could be replaced with an elastic blade model that can more accurately analyze the rotor blade behavior. In this way, the overall capability of the framework and fidelity of the predictions could be improved while maintaining high computational speeds.

Even though the current vision system can detect the visual cue and estimate the relative position and heading angle with great accuracy at close proximity, it requires knowledge about visual cue dimension and camera specifications in advance. This vision system can be used for different visual cues; however, the parameters in the vision algorithm need to be changed if a different visual cue is used. Hence, it is recommended to use additional sensors such as a Light Detection and Ranging (Lidar), Radio Detection And Ranging (RADAR), and thermal camera. The Lidar system can be useful in close proximity since it does not require knowing the specific values of a visual cue. Moreover, with advances in recent years, multiple COTS Lidar systems that can be conveniently integrated into aircraft systems have been introduced. To measure the relative distance, directional RADAR systems that transmit a short radio pulse with very high pulse power can also be used. A thermal camera that uses Infrared radiation (IR) can be effective at night time and in low visibility conditions. Maritime VTOL capable aircraft such as SH/MH-60, AW159, and MV-22 are equipped with a Forward-Looking Infrared (FLIR) system that provides visual information and distance from a target. Thus, the utilization of a thermal camera sensor can be integrated into existing helicopters without installing additional sensors. Hence, it is expected that the fusion

of the current digital camera, Lidar, RADAR, and the thermal camera can expand its operational capability and reliability.

The developed machine/deep learning-based vision system demonstrated excellent detection of a target ship in the long distance. Even though the YOLOv3 algorithm was carefully selected due to its fast detection rate and good accuracy; however, its performance was not compared with other algorithms. Therefore, a more systematic study on detection algorithms in such a ship landing environment is recommended. The detection accuracy of machine/deep learning-based vision algorithms surpassed the accuracy of human eyes in 2015 and the error was recorded as only 2.25% in 2017. Since then, new algorithms have been developed to increase the detection speed while maintaining the accuracy level. They typically measure the speed of detection from a recorded video, thus the rate is evaluated based on how fast it can detect an object of interest once the image frame is given. However, they still cause latency issues when integrated into the real-time flight control system. Therefore, in order to identify the best algorithm, it is recommended to compare the speed and accuracy of algorithms through flight testing. If the current machine/deep learning-based detection feedback rate of 0.5 seconds can be reduced to below 0.1 seconds, more rigorous control strategies can be constructed to enhance the flight performance in the long distance.

It is very powerful to use the state-of-the-art deep reinforcement learning control strategy at a close distance from the ship where the state feedback from the vision system is fast. It can quickly respond to a sudden wind gust, thus the drift can be minimized better than other conventional control systems. Also, this is an area where fast and accurate control is important, thus machines can perform better than human pilots by design. Even though the control policy obtained by training with TD3 algorithm with selected states and reward engineering demonstrated the improved disturbance rejection capability compared to the nonlinear PID control system, more extensive studies on deep reinforcement learning control algorithms, states, and rewards are recommended. Proximal Policy Optimization (PPO), Trust Region Policy Optimization (TRPO), Soft Actor-Critic (SAC), and Deep Deterministic Policy Gradient (DDPG) are some of the recommended candidate algorithms worth trying for training. The current states are selected as positions and velocities

including their 5 previous values. However, the effect of taking the specific number of previous data as states is not fully investigated, thus the number of previous data can be optimized during further studies. Also, the other sets of states can be analyzed to identify the states that can obtain a more effective control policy. After going through multiple trial and error processes, the current reward function was developed. Nevertheless, It is expected that the current reward function could be improved if the rewards are designed more distinctively to achieve the control task.

It is recommended to take a different control strategy in the long-distance since it requires comprehensive decision making to set an approach course by considering many factors such as weather condition, ship course, air and sea traffic, and so on. This is an area that the experienced pilot's decision-making ability is more important than quickly controlling an aircraft to achieve certain setpoints. Moreover, developing an explicit algorithm that can take accounts of all the possible scenarios is nearly impossible. Thus, imitation learning-based control strategy can be an appropriate approach to develop a higher-level control policy that encodes the human expert's ability. It can be obtained by learning from the expert's demonstrations in given situations, thus successfully trained control policy will behave like the experienced pilot.

Although the current experimental setup simulated ship landing environment closely, it could be improved for larger-scale flight tests. It is recommended to replace the current quadrotor UAV with a small helicopter UAV. The same autonomous ship landing system can be implemented with minimum modifications. Since the small helicopter UAV has similar configurations as existing helicopters such as rotors, landing gear, and center of gravity position, the testing results can be appropriately scaled for full-size helicopter ship landings. Thus, the effect of deck motions (magnitude and frequency) and wind gusts on full-size helicopters can be estimated by considering the differences in vehicle inertial properties and blade tip speeds. Once a small helicopter UAV successfully lands on a ship deck in properly scaled motions, there will be more confidence to implement the system on a full-size helicopter at sea.

Last but not least, the application of developed novel autonomous vision-based approach and landing method can be extended broadly since it demonstrated successful landings in one of the

most difficult moving platform landing cases, which is ship landing. Possible applications are landing on stationary and moving platforms such as cars, tanks, trains, submarines, and aircraft. It is also able to achieve precise approach and landing in GPS-denied environments such as inside a tunnel, inside a building, under a bridge or forest canopy, etc.. Moreover, the same autonomous flight method can also be applied to other purposes such as air refueling, vertical replenishment, and so forth with some modifications. In conclusion, it is expected that the novel autonomous ship landing method and the deliverables from this study could potentially be used across a wide range of areas beyond its original intended application.

# REFERENCES


[1] The Fleet Air Arm Association of Australia, "Of hurricat and hoverfly," *FlyBy*, vol. 11, 2021.

[2] Helicopter History Site - www.helis.com, "Gyrodyne dsn / qh-50 dash."

[3] "Helicopter operations from ships other than aircraft carriers (hostac)," *North Atlantic Treaty Organization Multinational Procedural Publication*, vol. 1, 2017.

[4] Aeronautical General Instruments Limited a portfolio company of AGI Holdings LLC, "Stabilized horizon reference system."

[5] J. Rife, S. Khanafseh, S. Pullen, D. De Lorenzo, U.-S. Kim, M. Koenig, T.-Y. Chiou, B. Kempny, and B. Pervan, "Navigation, interference suppression, and fault monitoring in the sea-based joint precision approach and landing system," *Proceedings of the IEEE*, vol. 96, no. 12, pp. 1958–1975, 2008.

[6] B. Ahmed, H. R. Pota, and M. Garratt, "Flight control of a rotary wing uav using back-stepping," *International Journal of Robust and Nonlinear Control: IFAC-Affiliated Journal*, vol. 20, no. 6, pp. 639–658, 2010.

[7] G. D. Padfield, *Helicopter flight dynamics: the theory and application of flying qualities and simulation modelling*. John Wiley & Sons, 2008.

[8] A. Gautam, P. Sujit, and S. Saripalli, "A survey of autonomous landing techniques for uavs," in *2014 international conference on unmanned aircraft systems (ICUAS)*, pp. 1210–1218, IEEE, 2014.

[9] S. J. Comstock, "Development of a low-latency, high data rate, differential gps relative positioning system for uav formation flight control," tech. rep., AIR FORCE INST OF TECHNOLOGY WRIGHT-PATTERSON AFB OH GRADUATE SCHOOL OF . . . , 2006.

[10] Navsource Naval History, "Navsource online: Aircraft carrier photo archive."

[11] Naval History and Heritage Command, "Navy autogiro takes flight," *History up close*, 2015.

[12] The United States Navy, "Us navy 060117-n-4935l-001 a rq-8a fire scout vertical take-off and landing tactical unmanned aerial vehicle (vtuav) system prepares for the first autonomous landing aboard the amphibious transport dock ship uss nashville (lpd 1)."

[13] Boeing/Schiebel, "Boeing s-100 camcopter."

[14] Eric RAZ / AIRBUS Helicopters, "Vsr700 opv demonstrates fully autonomous take off and landing from a moving platform."

[15] B. Lumsden, C. Wilkinson, and G. Padfield, "Challenges at the helicopter-ship dynamic interface," 1998.

[16] J. Colwell, "Maritime helicopter ship motion criteria-challenges for operational guidance," *Challenges for Operational Guidance-NATO RTO Systems Concepts and Integration Panel SCI-120. Berlin, Germany*, 2002.

[17] "Helicopter operating procedures for air-capable ships natops manual," 2003.

[18] D. A. Peters and C. J. He, "Finite state induced flow models. ii-three-dimensional rotor disk," *Journal of Aircraft*, vol. 32, no. 2, pp. 323–333, 1995.

[19] A. Bagai, *Contributions to the mathematical modeling of rotor flow fields using a pseudo-implicit free-wake analysis*. PhD thesis, University of Maryland, College Park, 1995.

[20] C. R. Theodore, *Helicopter flight dynamics simulation with refined aerodynamic modeling*. PhD thesis, University of Maryland, 2000.

[21] R. Sopher and D. W. Hallock, "Time-history analysis for rotorcraft dynamics based on a component approach," *Journal of the American Helicopter Society*, vol. 31, no. 1, pp. 43–51, 1986.

[22] M. J. Bhagwat and J. G. Leishman, "Stability, consistency and convergence of time-marching free-vortex rotor wake algorithms," *Journal of the American Helicopter Society*, vol. 46, no. 1, pp. 59–71, 2001.

[23] B. Lee and M. Benedict, "Development and validation of a comprehensive helicopter flight dynamics code," in *AIAA Scitech 2020 Forum*, p. 1644, 2020.

[24] B. Lee, "Helicopter autonomous ship landing system," Master's thesis, Texas A&M University, 2018.

[25] D. M. Pitt and D. A. Peters, "Theoretical prediction of dynamic-inflow derivatives," 1980.

[26] D. A. Peters and N. HaQuang, "Dynamic inflow for practical applications," 1988.

[27] W. Y. Abbott, J. O. Benson, R. G. Oliver, and R. A. Williams, "Validation flight test of uh-60a for rotorcraft systems integration simulator (rsis)," tech. rep., ARMY AVIATION ENGINEERING FLIGHT ACTIVITY EDWARDS AFB CA, 1982.

[28] D. Coleman, A. Halder, F. Saemi, C. Runco, H. Denton, B. Lee, V. Subramanian, E. Greenwood, V. Lakshminaryan, and M. Benedict, "Development of "aria", a compact, ultra-quiet personal electric helicopter," in *Vertical Flight Society 77th Annual Forum*, 2021.

[29] J. M. Daly, Y. Ma, and S. L. Waslander, "Coordinated landing of a quadrotor on a skid-steered ground vehicle in the presence of time delays," *Autonomous Robots*, vol. 38, no. 2, pp. 179–191, 2015.

[30] W. K. Holmes and J. W. Langelaan, "Autonomous ship-board landing using monocular vision," in *Proc. 72nd Am. Helicopter Soc Forum*, vol. 2, p. 36, 2016.

[31] K. Nho and R. K. Agarwal, "Automatic landing system design using fuzzy logic," *Journal of Guidance, Control, and Dynamics*, vol. 23, no. 2, pp. 298–304, 2000.

[32] O. Araar, N. Aouf, and I. Vitanov, "Vision based autonomous landing of multirotor uav on moving platform," *Journal of Intelligent & Robotic Systems*, vol. 85, no. 2, pp. 369–384, 2017.

[33] Q. H. Truong, T. Rakotomamonjy, A. Taghizad, and J.-M. Biannic, "Vision-based control for helicopter ship landing with handling qualities constraints," *IFAC-PapersOnLine*, vol. 49, no. 17, pp. 118–123, 2016.

[34] B. Erginer and E. Altug, "Modeling and pd control of a quadrotor vtol vehicle," in *2007 IEEE Intelligent Vehicles Symposium*, pp. 894–899, IEEE, 2007.

[35] B. Lee, V. Saj, M. Benedict, and D. Kalathil, "A vision-based control method for autonomous landing of vertical flight aircraft on a moving platform without using gps," *arXiv preprint arXiv:2008.05699*, 2020.

[36] K. A. Ghamry, Y. Dong, M. A. Kamel, and Y. Zhang, "Real-time autonomous take-off, tracking and landing of uav on a moving ugv platform," in *2016 24th Mediterranean conference on control and automation (MED)*, pp. 1236–1241, IEEE, 2016.

[37] B. Hu, L. Lu, and S. Mishra, "Fast, safe and precise landing of a quadrotor on an oscillating platform," in *2015 American Control Conference (ACC)*, pp. 3836–3841, IEEE, 2015.

[38] J. Kim, Y. Jung, D. Lee, and D. H. Shim, "Landing control on a mobile platform for multicopters using an omnidirectional image sensor," *Journal of Intelligent & Robotic Systems*, vol. 84, no. 1, pp. 529–541, 2016.

[39] K. Xia, S. Lee, and H. Son, "Adaptive control for multi-rotor uavs autonomous ship landing with mission planning," *Aerospace Science and Technology*, vol. 96, p. 105549, 2020.

[40] P. Vlantis, P. Marantos, C. P. Bechlioulis, and K. J. Kyriakopoulos, "Quadrotor landing on an inclined platform of a moving ground vehicle," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2202–2207, IEEE, 2015.

[41] B. prasad B and S. Pradeep, "Automatic landing system design using feedback linearization method," in *AIAA infotech@ Aerospace 2007 conference and exhibit*, p. 2733, 2007.

[42] H. Voos and B. Nourghassemi, "Nonlinear control of stabilized flight and landing for quadrotor uavs," in *Proceedings of the 7th Workshop on Advanced Control and Diagnosis ACD, Zielo Gora, Poland*, pp. 17–18, 2009.

[43] B. T. Burchett, "Feedback linearization guidance for approach and landing of reusable launch vehicles," in *Proceedings of the 2005, American Control Conference, 2005.*, pp. 2093–2097, IEEE, 2005.

[44] A. Rodriguez-Ramos, C. Sampedro, H. Bavle, P. De La Puente, and P. Campoy, "A deep reinforcement learning strategy for uav autonomous landing on a moving platform," *Journal of Intelligent & Robotic Systems*, vol. 93, no. 1-2, pp. 351–366, 2019.

[45] R. Polvara, M. Patacchiola, S. Sharma, J. Wan, A. Manning, R. Sutton, and A. Cangelosi, "Toward end-to-end control for uav autonomous landing via deep reinforcement learning," in *2018 International conference on unmanned aircraft systems (ICUAS)*, pp. 115–123, IEEE, 2018.

[46] D. V. Rao and T. H. Go, "Automatic landing system design using sliding mode control," *Aerospace Science and Technology*, vol. 32, no. 1, pp. 180–187, 2014.

[47] T. Lee and Y. Kim, "Nonlinear adaptive flight control using backstepping and neural networks controller," *Journal of Guidance, Control, and Dynamics*, vol. 24, no. 4, pp. 675–682, 2001.

[48] B. Ahmed and H. R. Pota, "Backstepping-based landing control of a ruav using tether incorporating flapping correction dynamics," in *2008 American Control Conference*, pp. 2728–2733, IEEE, 2008.

[49] T. Koo and S. Sastry, "Hybrid control of unmanned aerial vehicles for autonomous landing," in *2nd AIAA" Unmanned Unlimited" Conf. and Workshop & Exhibit*, p. 6541, 2003.

[50] G. Johnson, J. Waid, M. Primm, and R. Aggerwal, "Ship attitude accuracy trade study for aircraft approach and landing operations," in *Proceedings of IEEE/ION PLANS 2012*, pp. 783–790, 2012.

[51] M. W. Deppe and T. Benedik, "Joint precision approach and landing system (jpals)," *Journal of Air Traffic Control*, vol. 50, no. 1, 2008.

[52] X. Yang, L. Mejias Alvarez, and M. Garratt, "Multi-sensor data fusion for uav navigation during landing operations," in *Proceedings of the 2011 Australasian Conference on Robotics and Automation (ACRA 2011)*, pp. 1–10, Australian Robotics & Automation Association, 2011.

[53] M. Hardesty, S. Kennedy, S. Dixon, T. Berka, J. Graham, and D. Caldwell, "Development of navigation and automated flight control system solutions for maritime vtol uas operations," *USNA12*, vol. 20, 2012.

[54] J. L. Sanchez-Lopez, J. Pestana, S. Saripalli, and P. Campoy, "An approach toward visual autonomous ship board landing of a vtol uav," *Journal of Intelligent & Robotic Systems*, vol. 74, no. 1, pp. 113–127, 2014.

[55] G. Xu, Y. Zhang, S. Ji, Y. Cheng, and Y. Tian, "Research on computer vision-based for uav autonomous landing on a ship," *Pattern Recognition Letters*, vol. 30, no. 6, pp. 600–605, 2009.

[56] Y. Meng, W. Wang, H. Han, and J. Ban, "A visual/inertial integrated landing guidance method for uav landing on the ship," *Aerospace Science and Technology*, vol. 85, pp. 474–480, 2019.

[57] O. A. Yakimenko, I. I. Kaminer, W. J. Lentz, and P. Ghyzel, "Unmanned aircraft navigation for shipboard landing using infrared vision," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 38, no. 4, pp. 1181–1200, 2002.

[58] W. X. P. S. S. Zishan and S. Weiqun, "Computer vision scheme for autonomous landing of unmanned helicopter on ship deck [j]," *Journal of Beijing University of Aeronautics and Astronautics*, vol. 6, 2007.

[59] X. Wang, S. Pan, Z. Song, and W. Shen, "Vision based analytic 3d measurement algorithm for the autonomous landing of unmanned helicopter on ship deck," *Optical Technique*, vol. 33, pp. 264–267, 2007.

[60] C. S. Sharp, O. Shakernia, and S. S. Sastry, "A vision system for landing an unmanned aerial vehicle," in *Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation (Cat. No. 01CH37164)*, vol. 2, pp. 1720–1727, Ieee, 2001.

[61] L. S. H. C. Z. Jihong, "A method for estimating position and orientation of an unmanned helicopter based on vanishing line information [j]," *Computer Engineering and Applications*, vol. 9, 2004.

[62] S. Lange, N. Sunderhauf, and P. Protzel, "A vision based onboard approach for landing and position control of an autonomous multirotor uav in gps-denied environments," in *2009 International Conference on Advanced Robotics*, pp. 1–6, IEEE, 2009.

[63] S. Saripalli, J. F. Montgomery, and G. S. Sukhatme, "Vision-based autonomous landing of an unmanned aerial vehicle," in *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No. 02CH37292)*, vol. 3, pp. 2799–2804, IEEE, 2002.

[64] S. Saripalli, J. F. Montgomery, and G. S. Sukhatme, "Visually guided landing of an unmanned aerial vehicle," *IEEE transactions on robotics and automation*, vol. 19, no. 3, pp. 371–380, 2003.

[65] S. Saripalli and G. S. Sukhatme, "Landing on a moving target using an autonomous helicopter," in *Field and service robotics*, pp. 277–286, Springer, 2003.

[66] D. B. Barber, S. R. Griffiths, T. W. McLain, and R. W. Beard, "Autonomous landing of miniature aerial vehicles," *Journal of Aerospace Computing, Information, and Communication*, vol. 4, no. 5, pp. 770–784, 2007.

[67] B. Herissé, T. Hamel, R. Mahony, and F.-X. Russotto, "Landing a vtol unmanned aerial vehicle on a moving platform using optical flow," *IEEE Transactions on robotics*, vol. 28, no. 1, pp. 77–89, 2011.

[68] T. Merz, S. Duranti, and G. Conte, "Autonomous landing of an unmanned helicopter based on vision and inertial sensing," in *Experimental Robotics IX*, pp. 343–352, Springer, 2006.

[69] X. Pan, D.-q. Ma, L.-l. Jin, and Z.-s. Jiang, "Vision-based approach angle and height estimation for uav landing," in *2008 Congress on Image and Signal Processing*, vol. 3, pp. 801–805, IEEE, 2008.

[70] M. Sereewattana, M. Ruchanurucks, and S. Siddhichai, "Depth estimation of markers for uav automatic landing control using stereo vision with a single camera," in *Int. Conf. Information and Communication Technology for Embedded System*, 2014.

[71] T. Daquan and Z. Hongyue, "Vision based navigation algorithm for autonomic landing of uav without heading & attitude sensors," in *2007 Third International IEEE Conference on Signal-Image Technologies and Internet-Based System*, pp. 972–978, IEEE, 2007.

[72] B. Lee, V. Saj, and M. Benedict, "Machine learning vision and nonlinear control approach for autonomous ship landing of vertical flight aircraft," in *Proceedings of the 77th Annual Forum*, (Virtual), The Vertical Flight Society, May 2021.

[73] O. Shakernia, Y. Ma, T. J. Koo, and S. Sastry, "Landing an unmanned air vehicle: Vision based motion estimation and nonlinear control," *Asian journal of control*, vol. 1, no. 3, pp. 128–145, 1999.

[74] H. Yeo, W. G. Bousman, and W. Johnson, "Performance analysis of a utility helicopter with standard and advanced rotors," *Journal of the American Helicopter Society*, vol. 49, no. 3, pp. 250–270, 2004.

[75] F. Bailey, *A simplified theoretical method of determining the characteristics of a lifting rotor in forward flight*. US Government Printing Office, 1941.

[76] M. Ribera, *Helicopter flight dynamics simulation with a time-accurate free-vortex wake model*. PhD thesis, 2007.

[77] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, *et al.*, "Imagenet large scale visual recognition challenge," *International journal of computer vision*, vol. 115, no. 3, pp. 211–252, 2015.

[78] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 580–587, 2014.

[79] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE international conference on computer vision*, pp. 1440–1448, 2015.

[80] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: towards real-time object detection with region proposal networks," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 6, pp. 1137–1149, 2016.

[81] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *European conference on computer vision*, pp. 21–37, Springer, 2016.

[82] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 779–788, 2016.

[83] J. Redmon and A. Farhadi, "Yolo9000: better, faster, stronger," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7263–7271, 2017.

[84] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, 2018.

[85] B. Benjdira, T. Khursheed, A. Koubaa, A. Ammar, and K. Ouni, "Car detection using unmanned aerial vehicles: Comparison between faster r-cnn and yolov3," in *2019 1st International Conference on Unmanned Vehicle Systems-Oman (UVS)*, pp. 1–6, IEEE, 2019.

[86] M. Couprie and G. Bertrand, "Topological gray-scale watershed transformation," in *Vision Geometry VI*, vol. 3168, pp. 136–146, International Society for Optics and Photonics, 1997.

[87] W. Förstner and E. Gülch, "A fast operator for detection and precise location of distinct points, corners and centres of circular features," in *Proc. ISPRS intercommission conference on fast processing of photogrammetric data*, pp. 281–305, Interlaken, 1987.

[88] Z. Zhang, "Flexible camera calibration by viewing a plane from unknown orientations," in *Proceedings of the seventh ieee international conference on computer vision*, vol. 1, pp. 666–673, Ieee, 1999.

[89] N. Araki, T. Sato, Y. Konishi, and H. Ishigaki, "Vehicle's orientation measurement method by single-camera image using known-shaped planar object," *Int. J. Innov. Comput. Inf. Control*, vol. 7, no. B, pp. 4477–4486, 2011.

[90] OPenCV, "Camera calibration and 3d reconstruction." `https://docs.opencv.org/master/d9/d0c/group__calib3d.html`, 2021 (accessed April 17, 2021).

[91] K. Levenberg, "A method for the solution of certain non-linear problems in least squares," *Quarterly of applied mathematics*, vol. 2, no. 2, pp. 164–168, 1944.

[92] D. W. Marquardt, "An algorithm for least-squares estimation of nonlinear parameters," *Journal of the society for Industrial and Applied Mathematics*, vol. 11, no. 2, pp. 431–441, 1963.

[93] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[94] A. M. Andrew, "Multiple view geometry in computer vision," *Kybernetes*, 2001.

[95] "Gazebo simulation program." `http://gazebosim.org/`.

[96] B. Lee, "Vision-based vtol uav landings on a moving platform in gazebo simulation." `https://youtu.be/8kAeVzGJyN8`, 2020 (accessed June 30, 2020).

[97] W. Koch, R. Mancuso, R. West, and A. Bestavros, "Reinforcement learning for uav attitude control," *ACM Transactions on Cyber-Physical Systems*, vol. 3, no. 2, pp. 1–21, 2019.

[98] A. Rodriguez-Ramos, C. Sampedro, H. Bavle, P. De La Puente, and P. Campoy, "A deep reinforcement learning strategy for uav autonomous landing on a moving platform," *Journal of Intelligent & Robotic Systems*, vol. 93, no. 1-2, pp. 351–366, 2019.

[99] Y. Li, H. Li, Z. Li, H. Fang, A. K. Sanyal, Y. Wang, and Q. Qiu, "Fast and accurate trajectory tracking for unmanned aerial vehicles based on deep reinforcement learning," in *2019 IEEE 25th International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA)*, pp. 1–9, IEEE, 2019.

[100] J. Hwangbo, I. Sa, R. Siegwart, and M. Hutter, "Control of a quadrotor with reinforcement learning," *IEEE Robotics and Automation Letters*, vol. 2, no. 4, pp. 2096–2103, 2017.

[101] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel, "Domain randomization for transferring deep neural networks from simulation to the real world," in *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pp. 23–30, IEEE, 2017.

[102] X. B. Peng, M. Andrychowicz, W. Zaremba, and P. Abbeel, "Sim-to-real transfer of robotic control with dynamics randomization," in *2018 IEEE international conference on robotics and automation (ICRA)*, pp. 3803–3810, IEEE, 2018.

[103] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013.

[104] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, *et al.*, "Human-level control through deep reinforcement learning," *nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[105] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller, "Deterministic policy gradient algorithms," in *International conference on machine learning*, pp. 387–395, PMLR, 2014.

[106] S. Fujimoto, H. Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," in *International Conference on Machine Learning*, pp. 1587–1596, PMLR, 2018.

[107] O. S. Up, "Twin delayed ddpg." `https://spinningup.openai.com/en/latest/algorithms/td3.html`, 2021 (accessed June 21, 2021).

[108] L. Weng, "Domain randomization for sim2real transfer," *lilianweng.github.io/lil-log*, 2019.

[109] A. W. Doerry, "Ship dynamics for maritime isar imaging.," tech. rep., Sandia National Laboratories, 2008.

[110] B. Lee, "Flight testing of vision-based vtol uav landings on a fixed platform." https://youtu.be/w0dzVwBzFGk, 2020 (accessed June 30, 2020).

[111] B. Lee, "Flight testing of vision-based vtol uav landings on a moving platform." https://youtu.be/IboT80OR1T8, 2020 (accessed June 30, 2020).

[112] B. Lee, "Intelligent ship landing." https://youtu.be/ExkyUOdgYaw, 2021 (accessed April 17, 2021).

# APPENDIX A

## UH-60 HELICOPTER CONFIGURATION

| Main Rotor | |
|---|---|
| Number of blades | 4 |
| Radius R, ft | 26.83 |
| Blade chord c, ft | 1.75 |
| Rotational speed, rad/sec | 27.0 |
| Tip speed, ft/sec | 724.41 |
| Longitudinal mast tilt, deg | -3.0 |
| Airfoil section | SC 1095 |
| First airfoil section, ft | 5.08 |
| Blade precone, deg | 0.0 |
| Linear blade twist, deg | -18.0 |
| Solidity | 0.083 |
| Lock number | 5.11 |
| Control phase shift | -9.7 |
| Tail Rotor | |
| Number of blades | 4 |
| Radius, ft | 5.5 |
| Blade chord, ft | 0.81 |
| Rotational speed, rad/sec | 124.62 |
| Tip speed, ft/sec | 685.41 |
| Rotor shaft cant angle, deg | 20.0 |
| Fuselage | |
| Gross weight, lbs | 16000.00 |
| Pitch inertia $I_{yy}$, lbs$\cdot ft^2$ | 38512.0 |
| Roll inertia $I_{xx}$, lbs$\cdot ft^2$ | 4659.0 |
| Yaw inertia $I_{zz}$, lbs$\cdot ft^2$ | 36796.0 |
| $I_{xz}$, lbs$\cdot ft^2$ | 1882.0 |
| Horizontal tail surface area, $ft^2$ | 45.00 |

Table A.1: Main parameter of the UH-60 helicopter configuration