The Pennsylvania State University

The Graduate School

SCALED EXPERIMENTS IN VISION-BASED APPROACH AND LANDING IN HIGH SEA STATES

A Thesis in

Aerospace Engineering

by

Duncan J. Nicholson

© 2022 Duncan J. Nicholson

Submitted in Partial Fulfillment

of the Requirements

for the Degree of

Master of Science

May 2022

The thesis of Duncan J. Nicholson was reviewed and approved by the following:

Jacob W. Langelaan

Professor of Aerospace Engineering

Thesis Advisor

Joseph F. Horn

Professor of Aerospace Engineering

Eric Johnson

Professor of Aerospace Engineering

Amy Pritchett

Professor of Aerospace Engineering

Head of the Department of Aerospace Engineering

Abstract

Landings on moving, rotating ship decks pose significant challenges to UAV operators. In this thesis, a vision-based autonomous deck landing system was developed and tested with scale models in a series of wave conditions. A single monocular smart camera was used for detection and 6DOF pose estimation of a recursive AprilTag marker array. The fiducial marker array was detected and localized at 48 Hz from distances up to 5 m. The scalable fiducial marker system and wide field of view camera used were found to improve deck observability and the quality of deck state estimates over a wider range of distances compared to non-scalable visual aids.

Fusion of vision and inertial sensor data was performed using an Unscented Kalman Filter for relative deck state estimation. Tau trajectories were generated and followed using an explicit model following controller created from identified vehicle dynamic models. Performance of the vision system and estimator was measured using two separate motion capture systems for ground truth in hovering and landing flight tests. Fifteen successful autonomous landings were performed on the model ship deck in scaled sea states as high as six.

Contents

List of Figures

List of Tables

List of Symbols

$\mathbf{x}$ State vector

$\hat{\mathbf{x}}$ Estimated state vector

$\hat{\mathbf{x}}_0$ Initial estimated state vector

$\dot{\mathbf{x}}$ State derivative vector

$\mathbf{r}_d^i$ Deck position in the inertial frame [m]

$\mathbf{r}_d^i$ Vehicle position in the inertial frame [m]

$\mathbf{r}_{d/v}^i$ Deck position with respect to the vehicle expressed in the inertial frame [m]

$\mathbf{r}_{d/v}^v$ Deck position with respect to the vehicle expressed in the vehicle frame [m]

$\mathbf{r}_{c/v}^v$ Camera position with respect to the vehicle expressed in the vehicle frame [m]

$\mathbf{r}_{v/v_o}^{v_o}$ Vehicle position with respect to the vehicle origin marker expressed in the vehicle origin marker frame [m]

$\mathbf{r}_{d/d_o}^{d_o}$ Deck position with respect to the deck origin marker expressed in the deck origin marker frame [m]

$\mathbf{r}_{v_o/o}^o$ Vehicle origin marker position with respect to the Optitrack world frame expressed in the Optitrack world frame [m]

$\mathbf{r}^o_{d_o/o}$ Deck origin marker position with respect to the Optitrack world frame expressed in the Optitrack world frame [m]

$\Phi_{c/v}$ Array of 321 Euler angles describing sequence of intrinsic rotations from the vehicle to the camera frame [rad]

$\phi_{c/v}$ Roll Euler angle from the vehicle to the camera frame [rad]

$\theta_{c/v}$ Pitch Euler angle from the vehicle to the camera frame [rad]

$\psi_{c/v}$ Yaw Euler angle from the vehicle to the camera frame [rad]

x^v_d Deck x position in the vehicle frame [m]

y^v_d Deck y position in the vehicle frame [m]

z^v_d Deck z position in the vehicle frame [m]

$\mathbf{r}^c_{d/c}$ Deck position with respect to the camera expressed in the camera frame [m]

x^c_d Deck x position in the camera frame [m]

y^c_d Deck y position in the camera frame [m]

z^c_d Deck z position in the camera frame [m]

$\mathbf{v}^v_{d/v}$ Deck velocity with respect to the vehicle expressed in the vehicle frame [m s^{-1}]

$\Phi_{d/v}$ Array of 321 Euler angles describing sequence of intrinsic rotations from the vehicle to the deck frame [rad]

$\phi_{d/v}$ Roll Euler angle from the vehicle to the deck [rad]

$\theta_{d/v}$ Pitch Euler angle from the vehicle to the deck [rad]

$\psi_{d/v}$ Yaw Euler angle from the vehicle to the deck [rad]

$\dot{\Phi}_{d/v}$ Rate of change of Euler angles describing deck orientation with respect to vehicle [rad s^{-1}]

$\dot{\phi}^v_d$ Rate of change of deck roll Euler angle [rad s^{-1}]

$\dot{\theta}^v_d$ Rate of change of deck pitch Euler angle [rad s^{-1}]

$\dot{\psi}_d^v$	Rate of change of deck yaw Euler angle [rad s^{-1}]
$\boldsymbol{\omega}_{v/i}$	Vehicle angular velocity with respect to the inertial frame [rad s^{-1}]
$\boldsymbol{\omega}_{d/i}$	Deck angular velocity with respect to the inertial frame [rad s^{-1}]
$\boldsymbol{\omega}_{d/v}$	Deck angular velocity with respect to the vehicle frame [rad s^{-1}]
P_d	Deck roll rate [rad s^{-1}]
Q_d	Deck pitch rate [rad s^{-1}]
R_d	Deck yaw rate [rad s^{-1}]
$\mathbf{z}_{accel}$	Accelerometer measurement of vehicle proper acceleration [m s^{-2}]
$\mathbf{n}_{accel}$	Gaussian noise corrupting accelerometer measurements
$\mathbf{g}$	Acceleration due to gravity [m s^{-2}]
$\mathbf{R}_{v/i}$	Rotation matrix describing the orientation of the vehicle frame with respect to the inertial frame
$\mathbf{R}_{d/c}$	Rotation matrix describing the orientation of the deck with respect to the camera frame
$\mathbf{z}_{gyro}$	Gyroscope measurement of vehicle angular velocity [rad s^{-1}]
$\mathbf{n}_{gyro}$	Gaussian noise corrupting gyroscope measurements
$\mathbf{P}$	State covariance matrix
$\mathbf{P}_0$	State covariance matrix
y	Measurement used by UKF
y_k	Measurement at time step k
$\mathbf{n}_k$	Measurement noise covariance matrix
$\mathbf{v}_k$	Process noise covariance matrix
$\mathbf{v}_{IMU}$	Contribution to process noise covariance matrix from IMU
$\mathbf{P}_{IMU}$	IMU covariance
$\mathbf{v}_{ship}$	Contribution to process noise covariance matrix from ship motion

$N_{hull\ length}$ Scale factor based on ship hull length

$N_{rotor\ distance}$ Scale factor based on rotor-to-rotor distance

A_{waves} Wave amplitude [m]

F_{waves} Wave Frequency [Hz]

$A_{\phi_{d/o}}$ Amplitude of deck roll motion [°]

$A_{\theta_{d/o}}$ Amplitude of deck pitch motion [°]

ψ_{waves} Wave direction measured clockwise from bow [°]

$\epsilon_{\mathbf{r}^v_{d/v}}$ Error in the estimated relative deck position [m]

$\epsilon_{\Phi_{d/v}}$ Error in the estimated relative deck position [rad]

ϵ_{pos} Normalized position error [m]

ϵ_{rot} Normalized orientation error [rad]

$\bar{\epsilon}_{pos}$ Average normalized position error [m]

σ_{pos} Standard deviation of normalized position error [m]

$\bar{\epsilon}_{rot}$ Average normalized orientation error [rad]

σ_{rot} Standard deviation of normalized orientation error [rad]

Acknowledgments

This work was sponsored by the Office of Naval Research (ONR) under grant N00014-20-1-2092. The views and conclusions contained herein are those of the authors only and should not be interpreted as representing those of ONR, the U.S. Navy, or the U.S. Government.

I am grateful to all those with whom I have had the pleasure of working with during this and other related projects. First, I would like to thank my advisor, Jack Langelaan. I sincerely appreciate your guidance, feedback, and the learning opportunities you provided. I would also like to thank the rest of my committee: Dr. Horn, Dr. Pritchett, and Dr. Johnson for taking the time to read my thesis. Your advice and suggestions were invaluable.

My sincere thanks go out to Chris Hendrick, who worked actively alongside me. Without your help and dedication this work would not have been possible. A debt of gratitude is also owed to Anish Sydney for his assistance and effective planning. I would also like to acknowledge my colleagues Emma Jaques, Tomás Opazo Toro, and Nathan Kimmel for their generous collaboration.

Finally, I would like to thank my parents: I am immeasurably grateful for your endless support.

Chapter 1

Introduction

Landing of rotorcraft on arbitrarily moving ship decks can present significant challenges to operators. In a typical manually piloted shipboard landing, pilots are required to anticipate and handle sometimes violent disturbances like rough seas and high winds. When combined with the elevated pilot workload that accompanies normal approach planning, these issues can contribute to deck landing being a high-risk maneuver even for skilled pilots.

Vertical takeoff and landing (VTOL) aircraft are particularly useful in maritime environments. Helicopters have seen heavy use by the United States Navy, with the first sea-based operations occurring during the second world war. Today, maritime helicopters and tiltrotor aircraft are based from destroyers, frigates, and similarly sized vessels and fill diverse roles in naval aviation. These include but are not limited to: combat, search and rescue, transport, and reconnaissance.

Miniaturization of electronics has allowed for implementation of complex automatic control systems that allow traditional rotorcraft to operate with a high level of autonomy. In some cases, Unmanned Aerial Systems (UAS) have already taken over missions previously requiring manually piloted aircraft. A UAS is the combination of an Unmanned Aerial Vehicle (UAV or "vehicle" throughout this work) and the Ground

Control Station (GCS) used to control and monitor it. UAS can have significantly lower initial and operating costs than manned aircraft and minimize risk of endangering human lives upon total failure. UAS also eliminate the need to carry a pilot, increasing payload capacity and flight time. Small UAS (sUAS), weighing less than 55 lbs, have seen rapid adoption for military uses as well as by marine researchers and videographers. The increasing autonomy of UAS and sUAS is a key factor in their position as disruptor of a wide array of industries.

Mishaps often occur during the takeoff and landing phases of flight. Human decision-making and control inputs can be significantly hampered by stress imposed by collision risks. Implementation of automatic control during these phases is more difficult than in climb-out, cruise, and loiter due to the unpredictable nature of the environment. For maritime UAV landing, factors contributing to difficulty include sea spray, deck motion, and ship air wakes.

This thesis presents a complete autonomous deck landing system and analysis of flight test results. A small scale, 6 lb, coaxial hexacopter and a 19.5 ft ship model were used to demonstrate the efficacy of the completed system. Testing was conducted at the Maneuvering and Seakeeping Basin (MASK), part of the Naval Surface Warfare Center, Carderock Division (NSWCCD). Multiple Autonomous landings were executed while the ship model was forced by waves produced by the wave making system in the MASK.

1.1 Motivation

Current commercially available automated deck landing systems are cost prohibitive for small scale operations and require complicated (heavy) sensing and wireless transmission devices to be installed on the ship and aerial vehicle. Figure 1.1 shows a

Figure 1.1: A U.S. Navy MQ-8C Fire Scout approaches the deck of the destroyer USS Jason Dunham off the coast of Virginia in December 2014. (Source: [1])

modern UAV attempting an autonomous deck landing [1]. The UAV pictured in Figure 1.1 uses a specialized system developed by the Sierra Nevada Corporation® called the UAV Common Automatic Recovery System (UCARS) to compute and transmit precise ship motion data for navigation and control during landing. A system that enables autonomous landing on non-specialized ships using only passive sensing and on-board processing has the potential to greatly extend the mission capabilities of ship-borne unmanned air systems.

1.2 Framework of A Landing System

A high-level block diagram for a deck landing system is shown in Figure 1.2. The Inertial Measurement Unit (IMU), camera, and Global Positioning System (GPS)

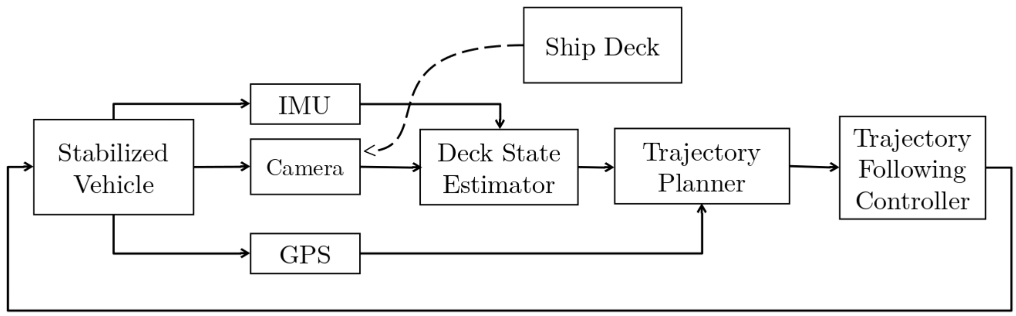


Figure 1.2: A high-level block diagram of an autonomous shipboard landing system

blocks represent sensors available on-board the UAV. The IMU provides measurements of the aircraft's raw acceleration and angular velocity, as well as a filtered orientation estimate. The camera block produces measurements describing the relative position and orientation of the UAV and the deck. The GPS block provides UAV position, velocity, and altitude measurements.

The deck state estimator takes sensor data from the camera and IMU and recursively estimates the relative system states. The trajectory generator block takes the current deck state estimates, determines the desired state, and defines a touchdown trajectory. The trajectory following controller block generates guidance commands to follow the desired trajectory. Finally, the stabilized vehicle block accepts guidance commands (for example roll, pitch, and yaw rate, as well as body-z axis acceleration) and generates the low-level commands that are sent to actuators.

1.3 The Problem of Landing in High Sea States

Deck landing is a time limited maneuver that requires accurate sensing, navigation and control. Wave motion itself is uncertain, and deck motion is driven by the complex interaction between the ship's hull and the water. The air wake of the ship produced by winds accompanying high sea states is an unpredictable external disturbance that must be rejected.

A landing system that is self-contained (or at least requires minimal infrastructure on the ship) would enable broader deployment of autonomous systems and increase the mission space. Hence, this thesis examines a solution that is self-contained (i.e. all sensing and computation is done on-board the UAV) and requires only a passive visual target at the landing site.

1.4 Related Work

Precision landing of UAVs is a large research area. Autonomous landing techniques vary in the aircraft configurations used (VTOL or fixed wing), sensor configurations, control techniques, and recovery objectives. This section briefly overviews some important works informing the methodology employed in this thesis. Focus was placed on vision based methods for deck identification, localization and pose estimation.

In 2014, Gautam *et al.* performed an extensive survey of autonomous landing techniques for UAVs [4]. The authors make a distinction between methods using GPS and vision-based methods that do not rely on GPS, noting that height measurements from GPS are inaccurate and that for robust landing design, a combination of GPS, Inertial Navigation Sensors (INS), and close range sensors are typically used. Linear control techniques, including PID (proportional-integral-derivative) control, and nonlinear techniques are analyzed with applications to UAV landing.

In 2013, Truskin described wave motion modeling and relative deck state estimation [5]. Estimator performance was analyzed using Monte Carlo simulations of different wave conditions and wave modeling methods. The estimator design presented in [5] was further developed by Holmes and implemented on hardware similar to that used in this work [6,7]. The deck landing system implemented by Holmes in 2017 used light beacons and a monocular camera [7]. A set of four red LEDs were used to mark the landing area. The Munkres algorithm was used for data association of detected lights with expected lights. Bearings to each light as well as bearing rates were used to estimate deck states using an Unscented Kalman Filter. Tau guidance was used to generate trajectories, while a PID attitude controller and PI throttle controller were used for trajectory following. Testing was completed on a static and moving Stewart platform.

Lumsden *et al.* describes challenges associated with maritime landing procedures and details a device known as a Landing Period Designator (LPD) [8]. This informs the pilot (or autonomous system) of the onset of quiescent periods in wave motion. The LPD works by summing kinetic and potential energy in the ship. When this value is low, it signifies a desirable condition for landing.

In 2018, Mohammadi *et al.* used GPS, a gimbaled camera, and a single AprilTag to land a UAV on a moving platform using model predictive control [9].

Ling *et al.* developed a three stage landing controller architecture that utilized a vision-based relative position estimator as well as GPS and magnetometer measurements [10]. The position and orientation (referred to throughout this work as "pose") of the landing area was estimated in the body-fixed inertial frame of the UAV which limited the effect of noisy GPS and magnetometer measurements.

Lee *et al.* introduces a vision-based, GPS-Denied approach for landing UAVs on moving platforms [11]. The computer vision system used a single front facing monocular camera and a vertically oriented visual cue. This approach mimics modern Naval use of a gyro-stabilized horizontal reference bar, and is beneficial for a horizontal approach angle rather than a purely vertical descent. Detection of the visual cue (a 120 mm square, 4×4 checkerboard) was done using Förstner corner detection. An image resolution of 1280×720px was used to achieve detections at a maximum distance of 17.5 m. Simulations and flight tests were conducted to demonstrate the ability of the multiphase landing system on platforms moving in arbitrary trajectories using both stationary and moving platforms.

Lee *et al.* later built upon work in [11] using a combination of a horizon bar corner detection approach for short-range pose estimation of the platform, combined with YOLOv3 Convolutional Neural Network architecture for object recognition and long range detection. A single state Kalman filter and probabilistic PID controller with nonlinear derivative gain was used. A Parrot ANAFI was used in flight tests. Images were captured on the vehicle and streamed to an external computer via Wi-Fi where vision processing, and outer-loop controller calculations were performed. The control inputs were then sent back to the embedded autopilot inner-loop via Wi-Fi. An integrated NVIDIA GeForce GTX 1660 Ti with 8GB of memory was used for graphics computation on the external computer.

Additionally, Lee *et al.* implemented a reinforcement learning control strategy focused on wind gust disturbance rejection [12]. The same vision system from [11] was used. Training simulations were performed in Gazebo, and flight tests were performed with similar procedures as in [11, 13].

In 2019, Bhargavapuri *et al.* used a fully actuated quadcopter with a downward facing camera to land on a moving ground vehicle using a compound fiducial marker [14]. A fully actuated quadcopter has a larger flight envelope which is advantageous for perception of the landing platform's position and orientation, however most UAVs do not have this ability.

Biological perching and the aerodynamics of surface takeoff and landing as well as surface attachment methods are analyzed in [15]. This paper looks in detail at air-surface transitions in flying animals and looks at future directions for perching aerial robots. The authors note animal reliance on visual feedback to gauge speed before perching is the foundation of Tau Theory (introduced in [16] and used extensively in [7])

Gajjar *et al.* gives an overview of fixed wing UAV landing procedures [17]. Pravitra *et al.* developed a guidance algorithm for landing fixed-wing UAVs on moving platforms and briefly overviewed algorithms for rotorcraft [18]. Pravitra *et al.* later simulated station-keeping and landing of fixed-wing UAVs on moving platforms with zero relative pitch, roll, and heading using adaptive control [19].

In 2021, Pravitra demonstrated shipboard UAS operations with an 800 size helicopter and US Naval Academy yard patrol craft [20]. A moving-base RTK GPS implementation was used for relative positioning, and a dedicated ship computer ran an extended Kalman filter that fused GPS and IMU measurements. Ship navigation data was streamed between the UAV and the ship using Wi-Fi. An algorithm denoted Ensemble Iterative Linear Quadratic Regulator (EiLQR) was used to generate multiple optimal landing trajectories with relative attitude minimization.

Chang *et al.* fused position measurements from a UAV mounted ArUco marker and YOLOv4 algorithm using an Intel Realsense D455 stereo camera and offboard Intel NUC computer. Ground effect free trajectories were demonstrated in experiments with a DJI F450 and moving ground vehicle.

Vision-based deck localization methods commonly applied in robotics include fiducial marker systems [21–23], IR LED beacons, standard RGB LEDs, and reflective fiducial markers.

While these works show autonomous ship deck landings have been achieved using a number of different sensor configurations, the majority of systems require significant data processing capabilities and depend on telemetry data streamed between ship sensors and computer systems.

1.5 Contributions

The areas identified to be in need of development to achieve reliable and safe vision based maritime autonomous landings were deck identification, pose estimation, and relative deck state estimation. The contributions included in this thesis are:

1. An exploration of vision-based deck identification and pose estimation methods was completed. Systems were evaluated on accuracy, recall, and speed during all phases of landing.

2. The selected deck identification and pose estimation solution was configured and implemented on a custom sUAS platform designed for shipboard landing. All vision and estimation algorithms were processed onboard the vehicle using low-cost computer hardware.

3. A series of flight tests were completed to analyze the performance of the landing system. Landings were performed on a small scale ship model under a number of different wave conditions. 15 successful landings were performed autonomously using onboard vision and estimation algorithms.

1.6 Reader's Guide

- Chapter 2 contains an overview of the shipboard landing problem and the specific constraints under which the problem is approached in this thesis.

- Chapter 3 details computer vision methods for deck localization and pose estimation.

- Chapter 4 discusses the implementation of an Unscented Kalman Filter (UKF) in detail.
- Chapter 5 describes flight test hardware, procedures, and gives relative deck state estimator accuracy measurements
- Chapter 6 presents approach and landing flight test results.
- Chapter 7 is the conclusion and summarizes findings and suggests improvements for future work.

Chapter 2

The Ship Deck State Estimation Problem

This chapter describes the system models and methods used to develop the landing system described in this thesis. Section 2.1 defines the states that will be estimated and Section 2.2 defines the sensor and system models as well as the required coordinate transforms. The specifics of the vision system and the use of fiducial markers is then described in Chapter 3.

2.1 Problem Statement

Traditional deck landing systems have a significant disadvantage in that they usually require specialized equipment to be installed on the ship. Specialized sensing and communication devices used only for landing add complexity and costs to a UAS, while also limiting the system's applicability to ships without these devices installed. It was desired to design, build, and field test an automated deck landing system for use with non-specialized ships.

This work strove to use only measurements produced onboard the UAV. This eliminated the ability to use sensors installed on the ship for localization or pose estimation. Any streaming of data requiring Radio Frequency (RF) communication devices to be installed, configured, or operated on the ship was discouraged. Traditionally, GPS or Real Time Kinematic GPS is used in applications where accurate localization is required. It was desired to perform landing without aid of any GNSS to simulate degraded conditions at sea as reducing reliance on GPS is a concern of the US Navy. The method defined here is not currently independent of external navigation or localization systems in that it uses GPS measurements of aircraft position. However, there are no RF signals transmitted from either the ship or the aircraft during landing. Note that the use of vision does imply that this system can be made GPS independent in the future: here GPS serves to define an inertial coordinate frame that is used in trajectory planning and control.

To make the deck landing technology solution as transferable as possible, only low-cost sensors were to be used. This limited the use of most scanning LIDAR, laser rangefinders, RADAR altimeters, and other cost-prohibitive sensors that provide high-fidelity reconstructions of the environment. It was desired to use only a single monocular camera to enable the UAV to perceive its environment if possible.

In addition to measurements being produced on the UAV using low-cost sensors, all data processing was to be performed onboard the UAV.

Finally, the system shall be easily implemented on vehicles of different scales, ranging from micro aerial vehicles to full scale aircraft. To allow the system to be as applicable as possible to a variety of aircraft sizes and configurations, the system shall be designed around a minimal set of parameters to allow performance tuning based on the expected sea state.

Vision measurements providing deck localization and pose estimation should not be used directly for trajectory generation due to the possibility of false positive measurements and time jumps in the measurements. Rather than directly using vision measurements, an estimator that fuses expected deck motion with real time measurements can still provide useful information even during periods of temporary loss of vision data (for example, due to occlusions caused by spray or waves). The state vector is

$$\mathbf{x} = \begin{bmatrix} {\mathbf{r}^v_{d/v}}^T & {\Phi_{d/v}}^T & {\mathbf{v}^v_{d/v}}^T & {\boldsymbol{\omega}_{d/i}}^T \end{bmatrix}^T \tag{2.1}$$

where

$$\begin{aligned}
\mathbf{r}^v_{d/v} &= \begin{bmatrix} x^v_d & y^v_d & z^v_d \end{bmatrix}^T - \text{deck position in the vehicle frame [m]} \\
\Phi_{d/v} &= \begin{bmatrix} \phi_{d/v} & \theta_{d/v} & \psi_{d/v} \end{bmatrix}^T - \text{321 Euler angles from vehicle to deck [rad]} \\
\mathbf{v}^v_{d/v} &= \dot{\mathbf{r}}^v_{d/v} - \text{deck velocity in the vehicle frame [m/s]} \\
\boldsymbol{\omega}_{d/i} &= \begin{bmatrix} P_d & Q_d & R_d \end{bmatrix}^T - \text{deck angular velocity [rad/s]}
\end{aligned}$$

The time derivative of $\mathbf{x}$ must be explicitly derived in terms of itself and measurements from the INS.

2.2 Sensor and System Models

2.2.1 Reference Frames and Coordinate Systems

In this work, a number of acronyms are used to describe both earth-fixed and moving, rotating reference frames and their associated coordinate systems. Figure 2.1 shows the reference frames used in the following sections.

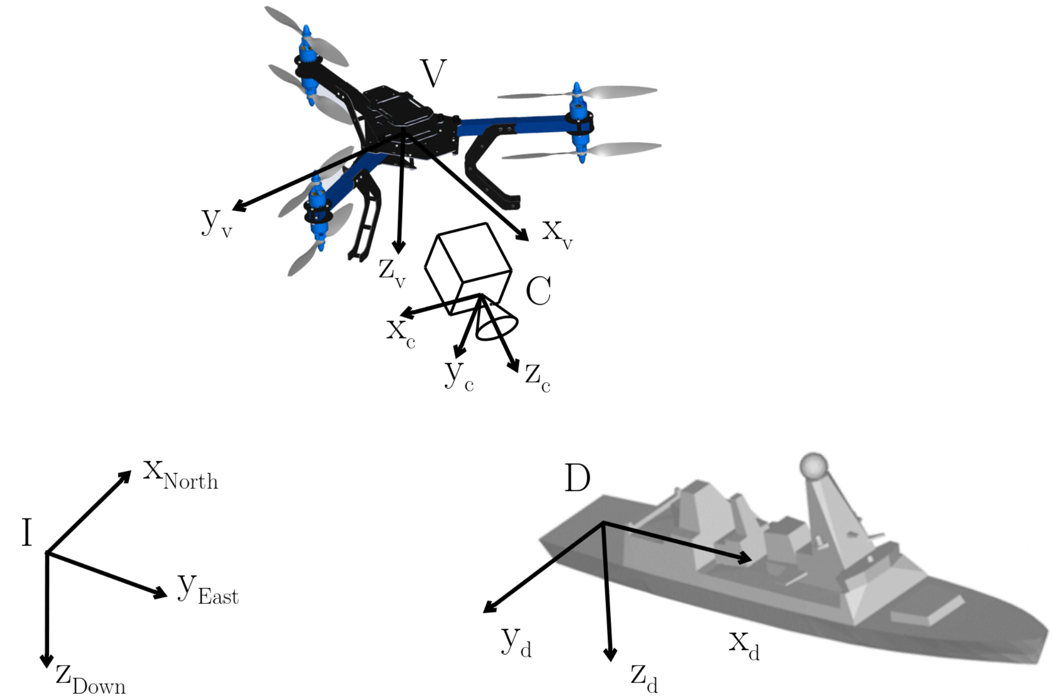


Figure 2.1: Reference frames used in kinematics calculations

For earth-fixed frames, the axes convention used was one of two local tangent plane coordinates: x-North, y-East, z-Down (NED) or x-East, y-North, z-Up (ENU). To convert a vector between the two coordinate systems, assuming the origin of the two frames are coincident, the following transform is used:

$$\mathbf{r}_{NED} = \mathbf{C}_{NED/ENU}\mathbf{r}_{ENU} \tag{2.2}$$

$$\mathbf{C}_{NED/ENU} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & -1 \end{bmatrix} \tag{2.3}$$

For moving, rotating reference frames (such as those affixed to a rigid body undergoing unconstrained motion in 6DOF) the axes convention used was either FRD or FLU. FRD denotes the frame is attached to the body with axes aligned in an x-Front, y-Right, z-Down orientation, while FLU denotes the frame is attached to the body with axes aligned in an x-Front, y-Left, z-Up orientation. To convert a vector between FRD and FLU coordinate systems, again assuming the origin of the two frames are coincident, the following transform is used:

$$\mathbf{r}_{FRD} = \mathbf{C}_{FRD/FLU}\mathbf{r}_{FLU} \tag{2.4}$$

$$\mathbf{C}_{FRD/FLU} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{bmatrix} \tag{2.5}$$

The camera frame is centered on the origin of the camera's sensor plane, and has a coordinate system with x-right, y-down, and z pointing out of sensor, along its principal axis.

The deck frame origin was defined coincident to the center of the landing area. The x-axis was parallel to the longitudinal axis of the ship with the direction towards the bow considered positive. The y-axis was defined parallel to the transverse axis of the ship with the starboard direction considered positive. The z-axis pointed downwards parallel to the ship's yaw axis. Importantly, the z-axis sense of the deck frame matches that of the camera frame for most orientations where detections are possible. This ensures that no discontinuities arise due to the estimation of Euler angles. In other words, the deck and vehicle frames would be parallel after a perfect landing.

2.2.2 Kinematics of the Ship Deck with respect to the Vehicle

The kinematics describing the relative motions of the deck and UAV are detailed below. It was assumed the vehicle and deck were rigid bodies.

2.2.2.1 Translational Kinematics

We begin by describing the position of the deck with respect to the vehicle in terms of the deck and vehicle positions expressed in the inertial frame.

$$\mathbf{r}^i_{d/v} = \mathbf{r}^i_d - \mathbf{r}^i_v \tag{2.6}$$

Differentiating Equation 2.6 gives the velocity of the deck relative to the vehicle in terms of the deck and vehicle velocities both expressed in the inertial frame.

$$\dot{\mathbf{r}}^i_{d/v} = \dot{\mathbf{r}}^i_d - \dot{\mathbf{r}}^i_v \tag{2.7}$$

We can also write $\dot{\mathbf{r}}^i_{d/v}$ using the definition for the derivative of a vector in rotating reference frame, $\frac{d}{dt}(\cdot) = \frac{d}{dt}(\cdot)_r + \boldsymbol{\omega} \times (\cdot)$. In this formula, the r subscript denotes the vector as it is seen from the rotating frame with angular velocity vector $\boldsymbol{\omega}$.

$$\dot{\mathbf{r}}^i_{d/v} = \dot{\mathbf{r}}^v_{d/v} + \boldsymbol{\omega}_{v/i} \times \mathbf{r}_{d/v} \tag{2.8}$$

Equating Equation 2.7 and Equation 2.8 gives an expression for the relative velocity of the deck expressed in the vehicle body frame.

$$\dot{\mathbf{r}}^v_{d/v} = \dot{\mathbf{r}}^i_d - \dot{\mathbf{r}}^i_v - \boldsymbol{\omega}_{v/i} \times \mathbf{r}_{d/v} \tag{2.9}$$

Differentiating again and reorganizing gives the acceleration of the deck with respect to the vehicle expressed in the vehicle body frame.

$$\ddot{\mathbf{r}}^v_{d/v} = \ddot{\mathbf{r}}^i_d - \ddot{\mathbf{r}}^i_v - \dot{\boldsymbol{\omega}}_{v/i} \times \mathbf{r}_{d/v} - 2\boldsymbol{\omega}_{v/i} \times \dot{\mathbf{r}}_{d/v} - \boldsymbol{\omega}_{v/i} \times \left(\boldsymbol{\omega}_{v/i} \times \mathbf{r}^v_{d/v}\right) \tag{2.10}$$

2.2.2.2 Rotational Kinematics

Angular velocity is additive over multiple frames [24]. Therefore, we can express the angular velocity of the deck in terms of the deck's angular velocity relative to the vehicle and the vehicle's angular velocity.

$$\boldsymbol{\omega}_{d/i} = \boldsymbol{\omega}_{d/v} + \boldsymbol{\omega}_{v/i} \tag{2.11}$$

The instantaneous rate of change of a set of Euler angles $\Phi_{b/r} = \begin{bmatrix}\phi_{b/r} & \theta_{b/r} & \psi_{b/r}\end{bmatrix}^T$ describing the orientation of reference frame b fixed to a rotating body with respect to another frame r is related to the relative angular velocity of that body, $\boldsymbol{\omega}_{b/r} = \begin{bmatrix}P & Q & R\end{bmatrix}^T$ by the relationship:

$$\begin{bmatrix}\dot{\phi}_{b/r} \\ \dot{\theta}_{b/r} \\ \dot{\psi}_{b/r}\end{bmatrix} = \begin{bmatrix}1 & \sin(\phi_{b/r})\tan(\theta_{b/r}) & \cos(\phi_{b/r})\tan(\theta_{b/r}) \\ 0 & \cos(\phi_{b/r}) & -\sin(\phi_{b/r}) \\ 0 & \frac{\sin(\phi_{b/r})}{\cos(\theta_{b/r})} & \frac{\cos(\phi_{b/r})}{\cos(\theta_{b/r})}\end{bmatrix} \begin{bmatrix}P \\ Q \\ R\end{bmatrix} \tag{2.12}$$

which can be written as

$$\dot{\Phi}_{b/r} = H(\Phi_{b/r})\boldsymbol{\omega}_{b/r} \tag{2.13}$$

where

$$H(\Phi) = \begin{bmatrix} 1 & \sin(\phi)\tan(\theta) & \cos(\phi)\tan(\theta) \\ 0 & \cos(\phi) & -\sin(\phi) \\ 0 & \frac{\sin(\phi)}{\cos(\theta)} & \frac{\cos(\phi)}{\cos(\theta)} \end{bmatrix} \tag{2.14}$$

Thus, the rate of change of the Euler angles describing the orientation of the deck with respect to the vehicle are

$$\dot{\Phi}_{d/v} = H(\Phi_{d/v})\boldsymbol{\omega}_{d/v} \tag{2.15}$$

Rearranging Equation 2.11 for $\boldsymbol{\omega}_{d/v}$ and substituting into Equation 2.15 gives the desired time derivative of the Euler angles

$$\dot{\Phi}_{d/v} = H(\Phi_{d/v})\left(\boldsymbol{\omega}_{d/i} - \boldsymbol{\omega}_{v/i}\right) \tag{2.16}$$

where $\boldsymbol{\omega}_{v/i}$ is measured by the gyro and $\Phi_{d/v}$ and $\boldsymbol{\omega}_{d/i}$ are contained in the estimated state vector.

2.2.3 Inertial Measurement Model

The IMU provided measurements of the vehicle's proper acceleration (acceleration relative to free-fall), angular velocity, and an estimate of attitude represented as a quaternion.

A measurement of acceleration thus takes the form

$$\mathbf{z}_{accel} = \ddot{\mathbf{r}}_{v/i} + \mathbf{R}_{v/i}\mathbf{g} + \mathbf{n}_{accel} \tag{2.17}$$

Where $\mathbf{g}$ is the gravity vector, $\mathbf{R}_{v/i}$ is the transformation from the inertial frame to the vehicle frame, and $\mathbf{n}_{acc}$ is Gaussian noise.

A measurement of angular velocity is of the form

$$\mathbf{z}_{gyro} = \boldsymbol{\omega}_{v/i} + \mathbf{n}_{gyro} \tag{2.18}$$

where $\mathbf{n}_{gyro}$ is Gaussian noise.

2.2.4 Vision Model

The purpose of the computer vision system employed is to identify the deck and estimate its position and orientation relative the camera and vehicle. The desired measurement from the vision system was therefore some representation of the pose of the deck, which can be expressed for example as a translation vector, $\mathbf{r}^c_{d/c}$, and rotation matrix, $\mathbf{R}_{d/c}$. Conversion of the deck pose from the camera frame to the vehicle frame is detailed in Section 4.1.2.

The camera was assumed to be an ideal pinhole camera, meaning distortions due to the non-zero lens aperture and imperfect lens focus were propagated into vision measurements.

Chapter 3

Vision System

This chapter describes the computer vision system implemented in this work. Section 3.1 describes different approaches to creating fiducial markers; Section 3.2 describes the specific implementation of the vision system; Section 3.3 discusses detection; Section 3.5 describes pose estimation using AprilTags; and Section 3.6 discusses performance of the vision system. The estimator used to generate real time estimates of deck state is then described in Chapter 4.

A focus was placed on vision based deck localization methods that provide high deck observability throughout the range of relative positions and orientations expected in landing trajectories, since Holmes identified that the quality of deck state estimates was degraded just before touchdown due to one or more of the LEDs going out of the field of view of the camera [7]. Thus, Fiducial Marker Systems (FMSs) are proposed to be a more robust, flexible, and inexpensive method of deck detection and 6DOF pose estimation compared to the deck localization method in [7].

3.1 Fiducial Marker Systems

Fiducial markers are widely used in robotics. Some FMSs are designed specifically for low-latency detection and pose estimation. FMSs allow deck identification and pose estimation to be performed using only a single monocular camera, and constitute a flexible solution for this task as one can generate markers or layout of markers that is tailored to the capabilities of the hardware employed. Fiducial markers are easily printed on standard paper and do not require specialized electronics on the ship or vehicle.

ArUco [2,25,26] and AprilTag [3,27,28] are two popular fiducial marker generation, detection, and pose estimation schemes. Both implementations differ in features and performance for detecting various configurations of marker layouts. Performance also varies based on camera sensor performance and processing speed. For the shipboard landing problem both the pose estimation results and the possibilities for fiducial layouts and positioning must be considered in choosing a particular fiducial marker system.

3.1.1 ArUco

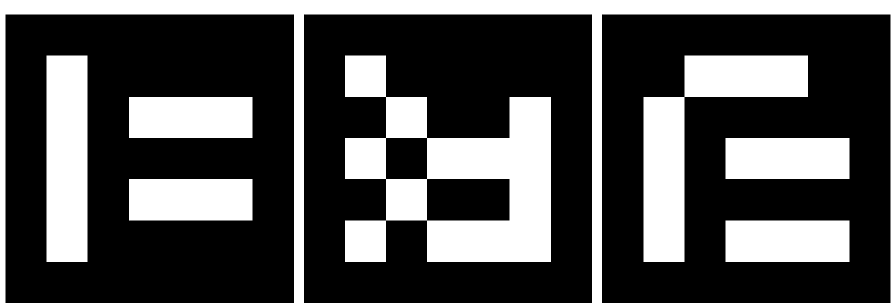


Figure 3.1: ArUco marker with IDs 68, 153, and 785 from the original ArUco family

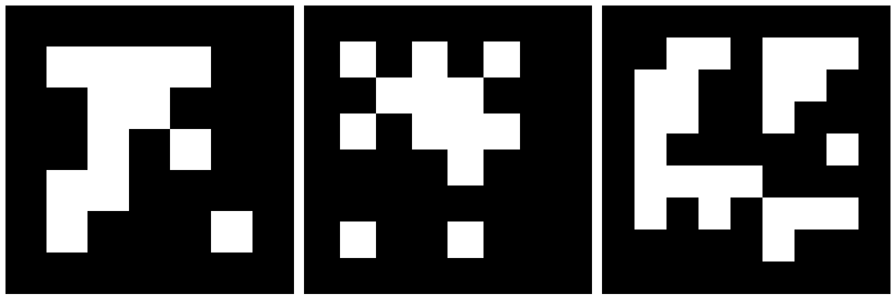


Figure 3.2: ArUco markers with IDs 12, 24, and 48 from 5×5, 6×6, and 7×7 ArUco marker families respectively

Figure 3.1 and Figure 3.2 show a collection of different ArUco Markers from different families. The ArUco detector works in a multistage fashion illustrated in Figure 3.3.

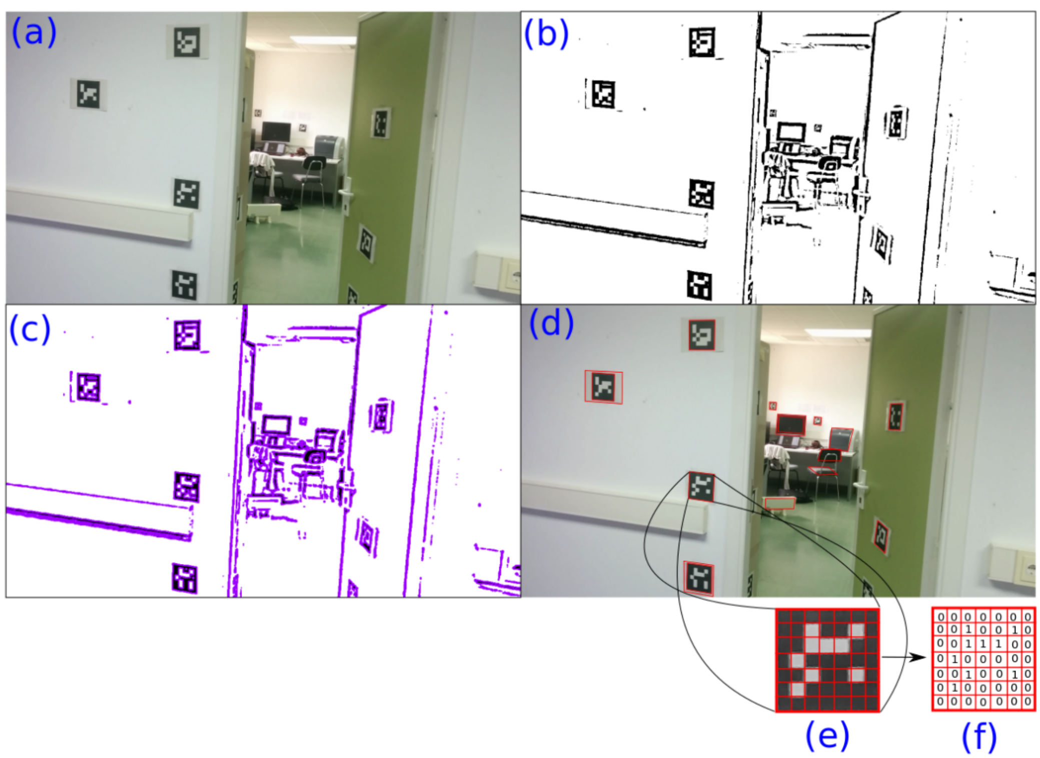


Figure 3.3: Detection process for ArUco markers (a) original (b) threshold (c) contours (d) approximated quadrilaterals (e) transform interior of square contours (f) binarization (Source: [2])

The ArUco marker detection algorithm can deal with ambiguities resulting from

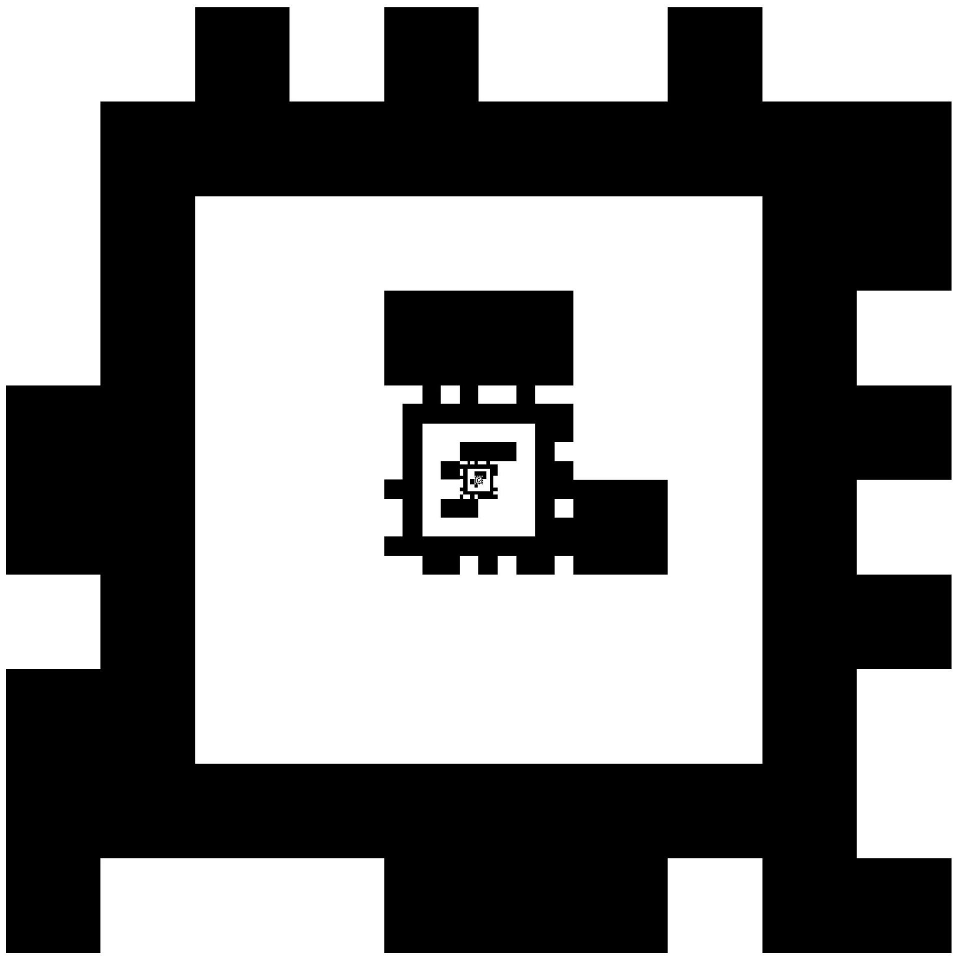


Figure 3.4: Recursive Apriltag marker array composed of four markers with IDs 0–4 from tag family tagCustom48h12

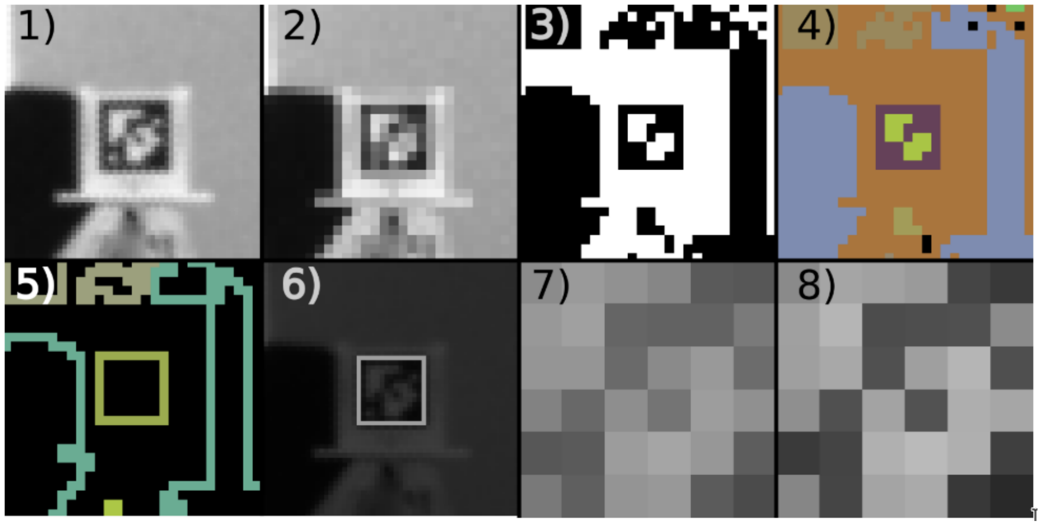


Figure 3.5: Apriltag Detection Process (1) starting image (2) decimation (4x in this example) (3) connected components (4) segmentation (5) contours (6) quadrilateral (7) crop and rectify data (8) sharpen and read data (Source: [3] © 2019 IEEE)

multiple solutions for camera homography inherent to the n-point problem with $n = 4$. This can be done by implementing a filter that rejects orientation estimates that significantly differ from previous orientations. See [26] for more detail on the ambiguity problem.

3.1.2 AprilTag

AprilTag is a fiducial marker system that has several advantages over light beacons [27]. The AprilTag detector employs a multiphase quadrilateral detection algorithm and extracts the marker's "code word", or the string that makes the fiducial uniquely identifiable against image backdrops commonly occurring in nature or industrial environments. Although detecting Apriltags can be more computationally intensive than detecting light beacons, they are specifically designed to allow users to generate tag patterns and change detector parameters to achieve high detection rates while maintaining low false positive rates desirable for identifying the ship deck with high certainty.

AprilTag3 builds on the improved AprilTag2 detector, allowing users to generate custom layouts [3, 28]. Tag Families such as tagCustom48h12 employ a layout with no data bits encoded in the center of the tag, allowing the fiducials to be nested recursively. The resulting recursive tags have the benefit of being detectable from a wider range of distances than a single tag. Using recursive tags reduces the need for a high-resolution camera if large enough tags are used. The proposed advantages of such a scalable marker system, pictured in Figure 3.4, are notable for precision landing.

The AprilTag FMS can be tailored to suit the needs of a specific set of hardware. As an example, a new family with 24 data bits, tagCustom24h13, can be generated using AprilRobotics apriltag-generation java code. Instructions to generate this family are given in Appendix B. The resulting family tagCustom24h13 has two members, shown nested inside one another in Figure 3.6. Table 3.1 shows the theoretical false positive detection rates for tagCustom24h13 and tagCustom48h12.

Table 3.1: Theoretical false positive rates vs corrected bits for two nested AprilTag families

Maximum Hamming Distance	False positive rate, %	
	48h12	24h13
0	0.00000001	0.00001192
1	0.00000073	0.00029802
2	0.00001765	0.00358820
3	0.00027703	0.02771616
4	0.00319502	0.15438795
5	0.02887335	0.66107512
6	–	2.26558447

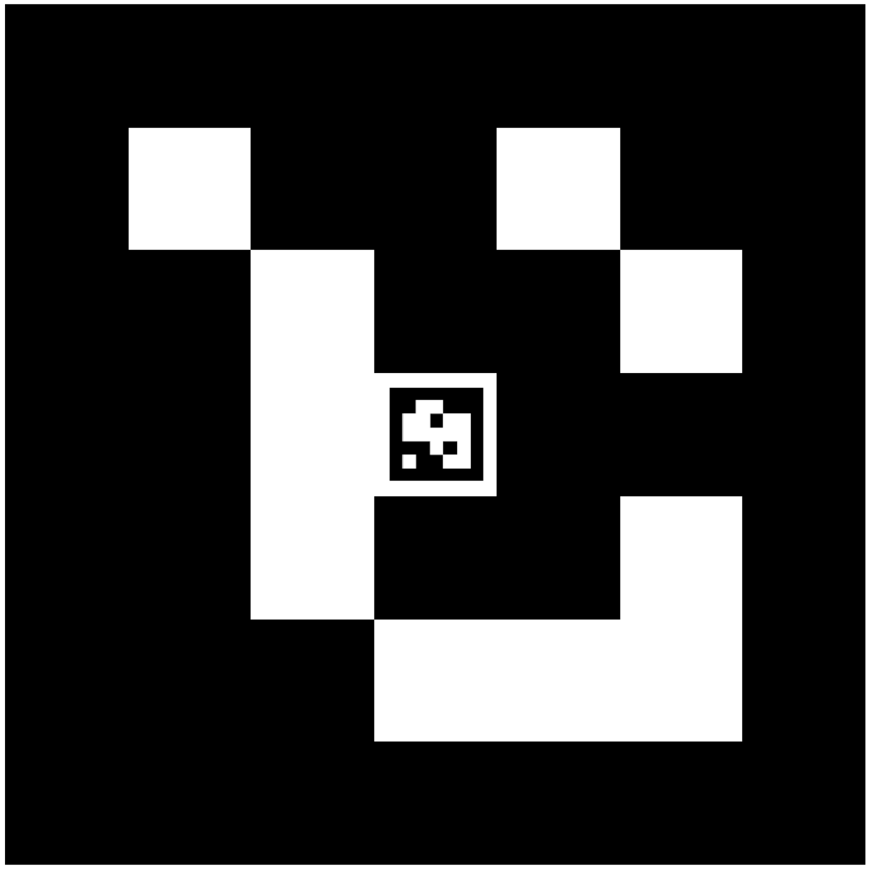


Figure 3.6: Recursive array comprised of markers with IDs 0 and 1 from AprilTag family tagCustom24h13

3.1.3 Fiducial Marker System Comparison

Two fiducial marker systems — Apriltag and ArUco, were run and tested in this experiment. Each was debugged, and optimized for detection rates under varying lighting conditions and with two different image sensors*.

AprilTag and ArUco rotation resistances were accurately measured in [29]. ArUco

*The upgraded Jevois camera model with global shutter sensor showed a notably crisper, brighter, more colorful image than the regular version with the pinhole lens detection rates were better than AprilTag at extreme angles, but both performed well with 10% Gaussian noise applied to the image [30]. The scalable layouts possible in AprilTag3 outweighed ArUco's slightly better performance at extreme angles, therefore AprilTag markers were selected for use in this work.

3.2 Vision Algorithm Implementation

Apriltag3 markers were used in all experiments in deck motion experiments and landing flight tests. To offload as much computation as possible from the on-board computer, a Jevois a33 smart machine vision camera was chosen to run marker detection and pose estimation. The base Jevois camera, shown in Figure 3.7, has a quad-core ARM Cortex A7 processor @ 1.35 GHz while weighing only 18 grams and costing $49. The upgraded camera version used in the hexacopter, priced at $99 adds a faster, global shutter AR0135 sensor, a larger vented stereolithography printed case, and an InvenSense ICM-20948 Inertial Measurement Unit (IMU). This version is shown in Figure 3.8. The fan affixed in the cameras kept the processor around 67°C during operation inside the enclosures and under the load produced by algorithms used.

The monochrome, global shutter AR0135 sensor was capable of imaging at a resolution of 640×480px at an advertised rate of 54Hz. The AprilTag algorithm was cross-compiled for ARM and deployed on the camera such that only a serial message containing marker pose data is output to the on-board computer, eliminating overhead from transmitting video.

Figure 3.7: JeVois A33 camera used on quadcopter

3.3 Fiducial Marker Detection

The Jevois camera performed the AprilTag detection algorithm, consisting of a four-step threshold and quadrilateral detection phase, followed by identification of the encoded data within the marker, as shown in Figure 3.5. The results of the detection step performed on an image containing a fiducial array could contain a number of different markers with different IDs and sizes, so they need to be correlated using the expected array's marker IDs, positions, and sizes. If the Identified marker's ID matches one in the array, its size can then be correlated.

The Apriltag detector is wrapped by custom C++ code compiled for the Jevois camera enabling multithreaded decimation, detection, and pose estimation using the cameras on-board processor. The locations of the four corners of the largest visible fiducial in the array were used for marker pose estimation.

Figure 3.8: JeVois A33 camera with OnSemi AR0135 global shutter sensor used on the hexacopter

The Apriltag C code was built into a Jevois machine vision module, written in C++ and compiled with the Cmake build system to run in parallel on the camera's processor. The Apriltag algorithm is initialized by the module which then enables USB communication between the camera and the Odroid Single Board Computer (SBC).

All AprilTag detector parameters, including sizes and positions of tags making up the array were implemented via switchable serial parameters. This allowed configuring of vision algorithm parameters remotely by using Minicom in an SSH session to communicate directly with the Jevois smart camera. The serial interface of the camera was exposed to the SBC via a short 28Ga USB mini-B to A cable. Additional strain relief was added to the USB mini-B connector by melting small films of solder to the sides in order to eliminate sideways free play. Fillets of hot glue were used to affix the cable in place and further prevent damage to the system during operation. Reliability of data connections between the camera and the SBC was found to be sufficient to provide a stable stream of deck pose information.

3.4 Camera Calibration

Since the desire was to achieve accurate pose estimation, each camera sensor used was calibrated using a ChArUco marker array printed on A4 paper. This unique array of checkerboard and ArUco markers is designed to be a target for camera calibration. The 8×5 ChArUco array that was used for calibration is shown in Figure 3.9.

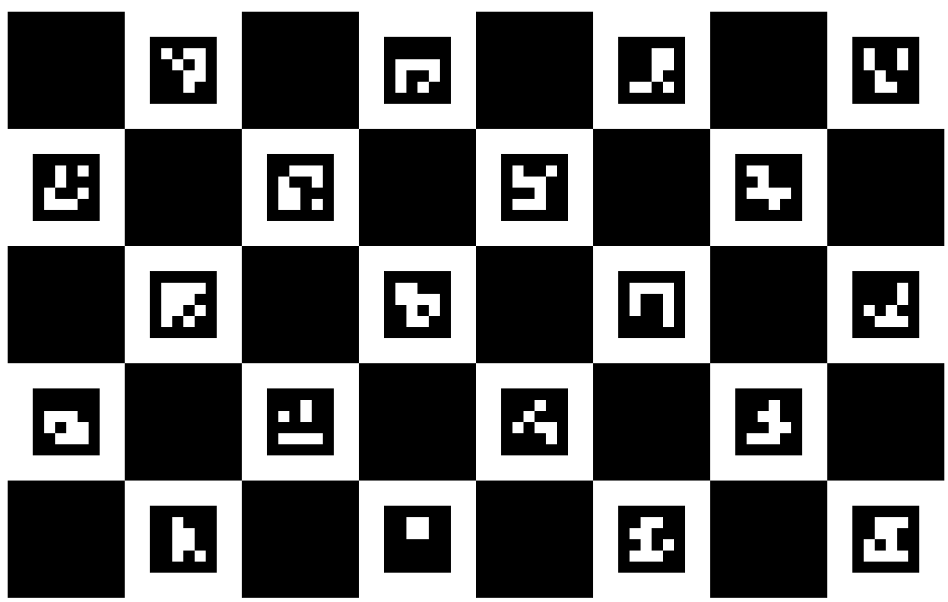


Figure 3.9: 8×5 ChArUco board used for camera calibration.

Camera intrinsic parameter determination was performed using a C++ module from OpenCV's extra modules Repository, opencv_contrib, called calibrate_camera_charuco.

Table 3.2: Camera intrinsic parameters for the Jevois AR0135 sensor with m12 125° lens at 640×480px resolution

f_x	393.9
f_y	331.1
p_x	393.3
p_y	256.0
k_1	-4.28e-1
k_2	2.15e-1
p_1	-3.66e-3
p_2	-9.51e-4
k_3	-5.74e-2

Radial and tangential distortion are parameterized by distortion coefficients k_1, k_2, p_1, p_2, k_3

An alternative method for camera calibration is given in [31]. The values for f_x, f_y, p_x, and p_y matched theoretical values calculated from the sensor size and lens aperture for the AR0135 Jevois camera closely, and are listed in Table 3.2 along with distortion coefficients.

The entire image could be rectified using this information, however use of the Jevois processor to do this slowed down the detection rate significantly. Rectifying the entire image could however improve marker pose estimate accuracy by removing radial distortion thus improving quadrilateral detection. Therefore, only the detected corners of marker used for pose estimation were corrected. It is possible to implement this using the Mali GPU on the camera, however this was not tested.

3.5 Camera Pose Estimation

Finding the 6DOF pose of the camera using the 2D projection of n 3D feature points is formally defined as the Perspective-n-Point (PnP) problem. The desired pose of the marker with respect to the camera is simply the inverse of the camera pose. Obtaining a solution for camera pose normally requires use of a camera projection model. One such model is the ideal pinhole camera model, which maps a point in the three-dimensional world, (x_1, x_2, x_3), to its projection onto the two-dimensional image plane, (y_1, y_2) using the camera's focal length $f > 0$.

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \frac{f}{x_3} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \tag{3.1}$$

Wu and Hu performed a detailed analysis of the PnP problem and solutions under various constraints [32]. For $n = 2$, camera pose can be determined if the orientation of reference points are available. For $n = 3$, or P3P, up to four geometrically feasible solutions can be obtained. For $4 \geq n \leq 5$, (*e.g.* if the four detected corners of a rectangular marker are used) two solutions are viable. For $n \geq 6$ a unique solution can be found as long as no four points are coplanar.

Numerical methods for solving PnP under different conditions is an active research area. Many open source algorithms exist for P3P such as OpenCV's *SolvePnP* and *solvePnPRansac* methods, and more recently, Persson and Nordberg's *LambdaTwist* algorithm [33].

The AprilTag library uses a nonlinear solver for the case of $n = 4$ (P4P) to estimate the pose of a single marker. The 3×3 homography matrix is found using the Direct Linear Transform algorithm from which the elements of a rotation matrix are solved and normalized to yield a proper rotation matrix. The pose estimates from the AprilTag3 algorithm were observed to be effected by noisy image data and outliers in the detected corner locations.

3.6 Vision System Performance

The nested tag family released along with AprilTag3, tagCustom48h12, proved to be readily detected using the Jevois camera. Using an image resolution of 640×480 pixels, the detection and pose estimation rate for the 24-inch square recursive marker array shown in Figure 3.4 was 48 Hz. Detection and pose estimation of multiple (three or more) tags in the same image reduced detection rates by 7–15Hz.

Two factors that affect the performance of the detector were identified: the number of bits of data encoded in the tag, and the number of tags in the family. The tag family 48h12 uses 48 bits of encoded data and has 42,211 unique codes. The memory requirements in bytes for AprilTag families follows

$$\mathcal{O}(\binom{n}{k}) = \mathcal{O}(\frac{n!}{k!(n-k)!}) \tag{3.2}$$

Where n is the number of tags in the family and k is the maximum corrected bits allowed in the detection. k is also referred to as the maximum hamming distance. It can be interpreted as how far away the decoded marker data can be from the references stored in memory.

Here n can be minimized to match the number of printed fiducials in the final 24 inch quad-marker recursive array used. The other 42,207 markers were commented out of the AprilTag source code built for the Jevois smart camera. Therefore, only four "code words" were loaded in memory during operation. The resulting memory usage of 432 Kb was small enough for the 256 Mb of memory on the Jevois camera.

Increasing the max hamming distance resistance can lead to higher true positive detection rates at the expense of more false positive detections. The theoretical false positive marker detection rates are shown for different values of k, the max hamming distance in Table 3.1. Setting the maximum hamming distance to three or higher can cause false positive detections from unexpected quadrilaterals in the image with large numbers of corrected bits. Most detections have zero or one corrected bit. In practice, the maximum hamming distance was set to two to improve occlusion resistance.

Chapter 4

Implementing a UKF using Fiducial Markers

This chapter describes the implementation of the unscented Kalman filter used with fiducial markers. Section 4.1 describes the time update and measurement update steps; Section 4.2 highlights the advantage of using coded fiducials (in short, the problem of data association is avoided); Section 4.3 discusses filter initialization and filter resets that may be needed if visual lock is lost for an extended period; flight test results of this estimator are then described in Chapter 5.

Information from multiple noisy sensors must be fused to obtain an estimate of the true relative deck state. The Extended Kalman Filter (EKF) is a widely used method for state estimation of nonlinear systems. The Unscented Kalman Filter, proposed by Julier and Uhlmann, is superior to the EKF for its ease of implementation and accuracy for systems with highly nonlinear process and observation models [34].

4.1 Unscented Kalman Filter implementation

Recursive estimation of the state of the discrete-time nonlinear dynamic system with known models F and G is desired, where $\mathbf{v}_k$ and $\mathbf{n}_k$ denote the process and observation noise respectively.

$$\mathbf{x}_{k+1} = F(\mathbf{x}_k, \mathbf{v}_k) \tag{4.1}$$

$$\mathbf{y}_k = G(\mathbf{x}_k, \mathbf{n}_k) \tag{4.2}$$

The states estimated were

$$\hat{\mathbf{x}} = \begin{bmatrix} {\mathbf{r}^v_{d/v}}^T & {\Phi_{d/v}}^T & {\mathbf{v}^v_{d/v}}^T & {\boldsymbol{\omega}_d}^T \end{bmatrix}^T \tag{4.3}$$

The state derivative vector is

$$\dot{\hat{\mathbf{x}}} = \begin{bmatrix} {\mathbf{v}^v_{d/v}}^T & \dot{\Phi}^T_{d/v} & \mathbf{a}^{vT}_d & {\boldsymbol{\alpha}_d}^T \end{bmatrix}^T$$

where

$$\dot{\Phi} - \text{Euler angle rates [rad/s]}$$

$$\mathbf{a}^v_d = \dot{\mathbf{v}}^v_d - \text{relative deck acceleration [m/s}^2\text{]}$$

$$\boldsymbol{\alpha}_d = \dot{\boldsymbol{\omega}}_d - \text{deck angular acceleration [rad/s}^2\text{]}$$

4.1.1 Time Update

The UKF time update, or prediction step, uses the previous state estimate and state covariance matrix to propagate expected state forward through system dynamics.

The prediction step occurred upon the arrival of a new acceleration and angular velocity measurement from the IMU located inside the flight controller. The flight controller was configured to output measurements at 200 Hz, but measurements were received by ROS at rate of approximately 185 Hz.

The IMU was placed on a stationary bench while 1500 measurements of linear acceleration and angular velocity were collected. The standard deviation of these measurements was used to determine the IMU covariance,

$$\mathbf{P}_{IMU} = \begin{bmatrix} 0.00093\ \frac{\mathrm{m}^2}{\mathrm{s}^4} & & & & & \\ & 0.00095\ \frac{\mathrm{m}^2}{\mathrm{s}^4} & & & & \\ & & 0.00012\ \frac{\mathrm{m}^2}{\mathrm{s}^4} & & & \\ & & & 0.022\ \frac{\mathrm{rad}^2}{\mathrm{s}^2} & & \\ & & & & 0.029\ \frac{\mathrm{rad}^2}{\mathrm{s}^2} & \\ & & & & & 0.023\ \frac{\mathrm{rad}^2}{\mathrm{s}^2} \end{bmatrix} \tag{4.4}$$

where the first three diagonal elements represent the linear acceleration covariance and the last three represent the angular velocity covariance.

The process noise used by the filter in the time update was scaled by the integer sea state parameter designed to be input by the user. The process noise was calculated using the same method as [5] and [7] and contained contributions from the IMU and the expected maximum amplitude and frequency of assumed sinusoidal ship motion associated with a particular sea state,

$$\mathbf{v}_k = \mathbf{v}_{IMU} + \mathbf{v}_{ship} \tag{4.5}$$

where $\mathbf{v}_k$ is the process noise covariance matrix and $\mathbf{v}_{IMU}$ and $\mathbf{v}_{ship}$ are the contributions from the IMU and ship motion respectively.

4.1.2 Measurement Update

The AprilTag algorithm running on the Jevois camera output the pose of the largest visible marker in the array as a translation vector, $\mathbf{r}^c_{d/c}$, and a 3×3 rotation matrix, $\mathbf{R}_{d/c}$. Marker pose estimation utilized the camera's processor while the UKF was run separately on the Odroid computer, therefore it was desired to use a more compact representation of orientation to reduce errors in transmission of pose data between the two devices. The data in AprilTag's custom rotation matrix type was mapped to an Eigen rotation matrix from which a quaternion was constructed. Unit quaternions consisting of a scalar and vector part, $q = q_w + \mathbf{q}$, often represented as a 4-tuple $\begin{bmatrix} q_w & q_x & q_y & q_z \end{bmatrix}^T$ are convenient method of describing spatial rotation. A quaternion $q_{b/a}$ represents the 4-tuple used to rotate vectors from system a to system b.

The translation vector and quaternion representing the AprilTag pose with respect to the camera was converted into a character array which was sent as a serial message to the Odroid computer. The serial message was parsed by the Odroid computer using a Python ROS node. The deck pose was converted into the vehicle frame using a standard ROS way: the tf2 package, specifically the *lookup_transform* method. This method consisted of broadcasting the deck pose measurement to the tf tree (see Appendix A), then looking up the pose with respect to the desired frame.

This method was used because it allowed the assumed static orientation of the camera with respect to the vehicle to be dynamically updated easily in the future. By calculating relative orientation using the orientation estimates from both the camera and flight controller IMUs, it would be possible to reduce error in deck pose estimates resulting from imperfectly setting camera rotation. Inconsistencies experienced during initialization of the Jevois camera IMU orientation estimate however caused this modification to be left to future work.

Thus, the translation vector and Euler angles describing the pose of the camera frame with respect to the vehicle were assumed to be constant:

$$\mathbf{r}^v_{c/v} = \begin{bmatrix} 0.0938 & 0.0000 & 0.0883 \end{bmatrix}^T \text{ m} \tag{4.6}$$

$$\Phi_{c/v} = \begin{bmatrix} \phi_{c/v} & \theta_{c/v} & \psi_{c/v} \end{bmatrix}^T = \begin{bmatrix} \frac{\pi}{4} & 0 & \frac{\pi}{2} \end{bmatrix}^T \text{ rad} \tag{4.7}$$

The camera displacement and rotation described by Equations 4.6 and 4.7 was added to the tf tree via a static transform publisher. The deck orientation with respect to the vehicle frame looked up from the tf tree was in the form of a quaternion, which was converted to Euler angles using the relationship

$$\begin{bmatrix} \phi \\ \theta \\ \psi \end{bmatrix} = \begin{bmatrix} \operatorname{atan2}\left(2\left(q_w q_x + q_y q_z\right), 1 - 2\left(q_x^2 + q_y^2\right)\right) \\ \operatorname{asin}\left(2\left(q_w q_y - q_z q_x\right)\right) \\ \operatorname{atan2}\left(2\left(q_w q_z + q_x q_y\right), 1 - 2\left(q_y^2 + q_z^2\right)\right) \end{bmatrix} \tag{4.8}$$

Thus, a measurement of the first six estimated states,

$$y = \begin{bmatrix} \mathbf{r}^v_{d/v} \\ \Phi_{d/v} \end{bmatrix} \tag{4.9}$$

is used by the filter. Note that the pose estimates provided by the Jevois camera were effectively a linear measurement of the states to be estimated. Hence, only the time update step of the estimator actually needs to be done using sigma points: the measurement update step can be done using standard linear Kalman filter equations.

The observation noise associated with the measurement in Equation 4.9 was

$$\mathbf{n}_k = \begin{bmatrix} (0.05\text{ m})^2\,\mathbf{I}_3 & \mathbf{0} \\ \mathbf{0} & \left((1\text{ deg})\,\frac{\pi}{180}\right)^2\mathbf{I}_3 \end{bmatrix} \tag{4.10}$$

4.1.3 Unscented Kalman Filter

The Unscented Kalman Filter equations implemented in this work were adapted from Cai and Zhao and are reproduced in Algorithm 1 on the following page [35].

4.2 On the Use of Coded Fiducials

Some challenges associated with Kalman filters are avoided through the use of April tags: data association (the process of associating a measurement with a particular tag) is handled through unique tag identifiers. The ID associated with each fiducial marker in the array is unique the family. Due to the low false positive rates observed for the detection of markers in AprilTag family tagCustom48h12, it can be assumed with a high degree of certainty that a tag detection correlated to a positive identification of the desired landing area.

False positive detections however did occur when a quadrilateral was fit to noise in the image and a string matching one of the IDs of the markers the algorithm was searching for was decoded from the noise inside that falsely identified quadrilateral. A false positive detection would result in an erroneous deck pose estimate from the vision system. Infrequent false positive detections were observed during testing that temporarily increased error in the estimated state and increased the estimated state

Algorithm 1: Unscented Kalman Filter Equations

Initialize

$$\hat{\mathbf{x}}_0 = E\left[\mathbf{x}_0\right]$$
$$\mathbf{P}_0 = E\left[\left(\mathbf{x}_0 - \hat{\mathbf{x}}_0\right)\left(\mathbf{x}_0 - \hat{\mathbf{x}}_0\right)^T\right]$$
$$\hat{\mathbf{x}}_0^a = E\left[\mathbf{x}^a\right] = \left[\begin{array}{ccc}\hat{\mathbf{x}}_0^T & \mathbf{0} & \mathbf{0}\end{array}\right]^T$$
$$\mathbf{P}_0^a = E\left[\left(\mathbf{x}_0^a - \hat{\mathbf{x}}_0^a\right)\left(\mathbf{x}_0^a - \hat{\mathbf{x}}_0^a\right)^T\right] = \begin{bmatrix}\mathbf{P}_0 & \mathbf{0} & \mathbf{0}\\ \mathbf{0} & \mathbf{P_v} & \mathbf{0}\\ \mathbf{0} & \mathbf{0} & \mathbf{P_n}\end{bmatrix}$$

where $\mathbf{x}^a = \left[\mathbf{x}^T \mathbf{v}^T \mathbf{n}^T\right]^T$, $\boldsymbol{\mathcal{X}}^a = \left[\left(\boldsymbol{\mathcal{X}}^x\right)^T \left(\boldsymbol{\mathcal{X}}^v\right)^T \left(\boldsymbol{\mathcal{X}}^n\right)^T\right]^T$

For $k \in \{1, \ldots, \infty\}$,

Calculate sigma points $\boldsymbol{\mathcal{X}}_{k-1}^a = \left[\hat{\mathbf{x}}_{k-1}^a \; \hat{\mathbf{x}}_{k-1}^a \pm \sqrt{(L+\lambda)\mathbf{P}_{k-1}^a}\right]$

Time update:

$$\boldsymbol{\mathcal{X}}_{k|k-1}^x = \mathbf{F}\left[\boldsymbol{\mathcal{X}}_{k-1}^x, \boldsymbol{\mathcal{X}}_{k-1}^v\right]$$
$$\hat{\mathbf{x}}_k^- = \sum_{i=0}^{2L} W_i^{(m)} \boldsymbol{\mathcal{X}}_{i,k|k-1}^x$$
$$\mathbf{P}_k^- = \sum_{i=0}^{2L} W_i^{(c)} \left[\mathcal{X}_{i,k|k-1}^x - \hat{\mathbf{x}}_k^-\right]\left[\mathcal{X}_{i,k|k-1}^x - \hat{\mathbf{x}}_k^-\right]^T$$
$$\boldsymbol{\mathcal{Y}}_{k|k-1} = \mathbf{G}\left[\boldsymbol{\mathcal{X}}_{k|k-1}^x, \boldsymbol{\mathcal{X}}_{k-1}^n\right]$$
$$\hat{\mathbf{y}}_k^- = \sum_{i=0}^{2L} W_i^{(m)} \mathcal{Y}_{i,k|k-1}$$

Measurement Update:

$$\mathbf{P}_{\tilde{\mathbf{y}}_k \tilde{\mathbf{y}}_k} = \sum_{i=0}^{2L} W_i^{(c)} \left[\mathcal{Y}_{i,k|k-1} - \hat{\mathbf{y}}_k^-\right]\left[\mathcal{Y}_{i,k|k-1} - \hat{\mathbf{y}}_k^-\right]^T$$
$$\mathbf{P}_{\mathbf{x}_k \mathbf{y}_k} = \sum_{i=0}^{2L} W_i^{(c)} \left[\mathcal{X}_{i,k|k-1} - \hat{\mathbf{x}}_k^-\right]\left[\mathcal{Y}_{i,k|k-1} - \hat{\mathbf{y}}_k^-\right]^T$$
$$\mathcal{K} = \mathbf{P}_{\mathbf{x}_k \mathbf{y}_k} \mathbf{P}_{\tilde{\mathbf{y}}_k \tilde{\mathbf{y}}_k}^{-1}$$
$$\hat{\mathbf{x}}_k = \hat{\mathbf{x}}_k^- + \mathcal{K}\left(\mathbf{y}_k - \hat{\mathbf{y}}_k^-\right)$$
$$\mathbf{P}_k = \mathbf{P}_k^- - \mathcal{K}\mathbf{P}_{\tilde{\mathbf{y}}_k \tilde{\mathbf{y}}_k}\mathcal{K}^T$$

where λ = composite scaling parameter, L = dimension of augmented state, $\mathbf{P_V}$ = process noise cov., $\mathbf{P_V}$ = measurement noise cov., W_i = weights covariance. Each time the state estimate quickly recovered due to the ratio of true positive detections and the high-rate measurements from the vision system.

4.3 Filter initialization

The initial state vector, $\hat{\mathbf{x}}_0$, was chosen to place the estimated ship deck two meters below and in front of and parallel to the vehicle with zero velocity and angular velocity.

$$\hat{\mathbf{x}}_0 = [\ 2.0\ 0\ 2.0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ ]^T \tag{4.11}$$

Note that this initialization is ultimately dependent on the overall scale of the landing problem. In the MASK testing (discussed in Section 5.3.2 and Chapter 6) the vehicle is often significantly farther away when the deck is first detected, but this initial estimate converges quickly.

The initial covariance matrix, $\mathbf{P}_0$, was chosen to be sufficiently large, with the expectation that uncertainty will quickly reduce when the first measurements are received.

$$\mathbf{P}_0 = \begin{bmatrix} \mathbf{P}_{0,(xyz)} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{P}_{0,(\phi\theta\psi)} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{P}_{0,(\dot{x}\dot{y}\dot{z})} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{P}_{0,(PQR)} \end{bmatrix} \tag{4.12}$$

where

$$\begin{aligned} \mathbf{P}_{0,(xyz)} &= (0.5\ \mathrm{m})^2\,\mathbf{I}_3 \\ \mathbf{P}_{0,(\phi\theta\psi)} &= \mathrm{diag}\left(\left(\left(\begin{bmatrix}15.0 & 15.0 & 20.0\end{bmatrix}\ \mathrm{deg}\right)\frac{\pi}{180}\right)^2\right. \\ \mathbf{P}_{0,(\dot{x}\dot{y}\dot{z})} &= \left(0.05\ \frac{\mathrm{m}}{\mathrm{s}}\right)^2\mathbf{I}_3 \end{aligned}$$

$$\mathbf{P}_{0,(PQR)} = \left(\left(2.0\ \frac{\mathrm{deg}}{\mathrm{s}}\right)\frac{\pi}{180}\right)^2 \mathbf{I}_3$$

4.3.1 Filter Resets

A counter was implemented such that the state estimate was reset to it's initial state, $\hat{\mathbf{x}}_0$ from Equation 4.11, if no vision measurements of the fiducial marker were received for a period of approximately seven seconds. This was done so the estimated state did not blow up to extremely large values due to noisy accelerometer measurements among other sources of error. Without measurements from the vision system, the UKF as implemented operated similar to dead-reckoning, which is not sufficient for maintaining an accurate state estimate over long periods of time.

Chapter 5

Relative Deck State Estimator Flight Tests

5.1 Vehicle Test Platform

An electric UAV platform was constructed for use in this research and is shown in Figure 5.1. The vehicle used was a modified 3DR Y6 to perform ship landing flight tests. The driving requirement for the vehicle design was water resistance. Sensitive electronics were housed in a waterproof enclosure, leaving only brushless DC motors exposed. Wiring and hardware interfaces were sealed using rubber grommets, gasket material, and adhesives. The six Electronic Speed Controllers (ESCs) were fitted to aluminum heat sinks to exchange heat outside the enclosure.

The vehicle was equipped with a CubePilot® Orange Flight Controller (FC) and an Odroid XU4 Single Board Computer. The Odroid was configured to use Robot Operating System (ROS) to communicate with the Pixhawk, on-board camera, and the ground control station. This allowed for state estimation and control to be performed on-board the vehicle while simultaneously synchronizing data collected from external

Figure 5.1: Coaxial hexacopter used in experiments described in this thesis

motion capture systems and streaming telemetry information to the GCS. The layout of electronics inside the enclosure is pictured in Figure 5.2. An upgraded fan was used to prevent thermal throttling of the on-board computer. Initial flight testing showed the hexacopter could perform multiple back-to-back flight tests without overheating.

The enclosure provided additional challenges for testing visual navigation methods. An optically clear acrylic dome was used to house camera sensor. A 3D printed camera mount with adjustable pitch angle was created to hold the camera in the center of the dome to minimize optical distortions. The camera pitch angle was set to $-45°$ relative to the vehicle's longitudinal axis. Figure 5.3 shows the camera as installed in the vehicle. This angle was chosen due to the desired approach angle, despite the favorable perception of deck pitch achieved by from a down facing camera and purely vertical flight path.

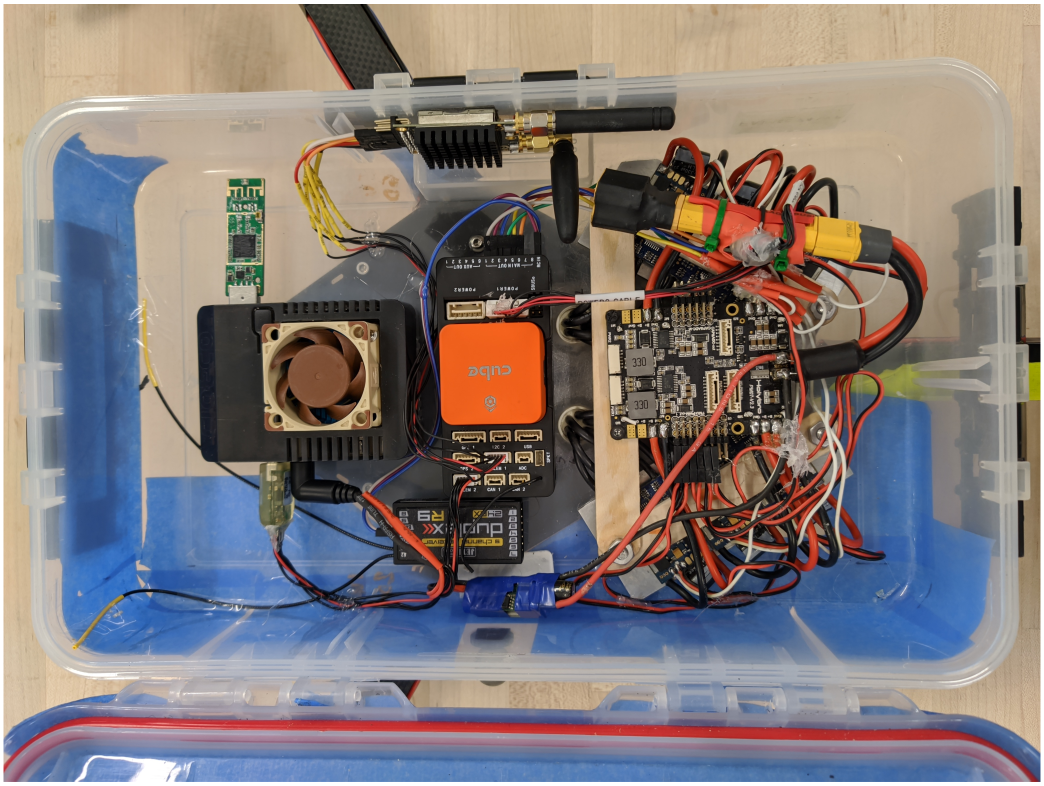


Figure 5.2: Layout of electronics inside the hexacopter's fuselage enclosure

5.2 Motion Capture Systems

Two motion capture (MOCAP) systems were used in this work, a VICON system and an Optitrack system. A description of the setup and uses of both systems is contained in Sections 5.2.1 and 5.2.2. The VICON system was used for preliminary testing and development, while the Optitrack system was used for scale tests over water.

5.2.1 VICON

A VICON system consisting of 12 cameras was used at Penn State to acquire ground truth data to analyze the performance of the deck state estimator. Figure 5.4 shows the hexacopter in flight in Penn State's indoor MOCAP lab.

Figure 5.3: Camera dome and camera mount with adjustable pitch angle

After calibration, the VICON system was configured to stream the pose of the vehicle and fiducial marker array at 120Hz. Latency of the VICON system was reported from the ground station running Vicon_bridge ROS source code modified to print latency[†]. The average latency of the VICON ground truth measurements was 16.6 ms.

The VICON motion capture system was used for vehicle translation velocity control during estimator accuracy testing at Penn State. The vehicle position and heading were fused in PX4's 24 state Extended Kalman Filter (EKF or EKF2) to enable position hold. Figure 5.5 shows an example top-down position time history for a flight in position mode. The flight depicted by Figure 5.5 was completed to verify external vision data was properly fused into the FC position estimate, and that the

[†]The node is now maintained on Github by the ETH Zurich Autonomous Systems Lab.

Figure 5.4: Hexacopter flying over the AprilTag Fiducial Marker array inside Pennsylvania State University's indoor MOCAP lab. Four of the 12 VICON cameras are pictured.

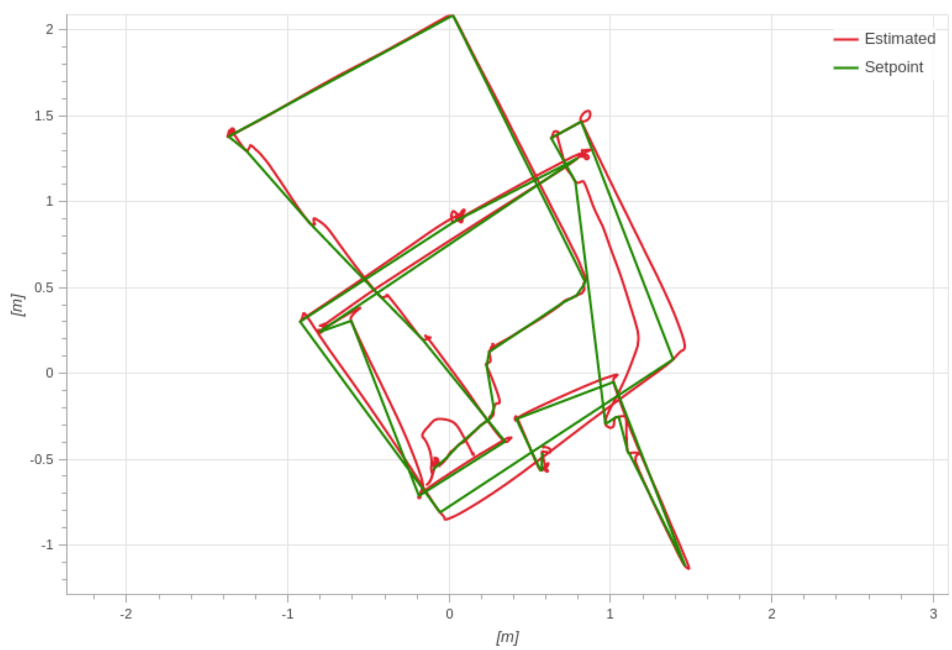


Figure 5.5: Flight Controller's estimated vehicle inertial x and y position plotted against desired position set point for a single flight in Penn State's VICON Motion Capture lab.

PX4 position control response was tuned appropriately.

5.2.2 Optitrack

A 14 camera Optitrack motion capture system was used during experiments in the MASK. The Optitrack cameras were affixed to the railing in one corner of the MASK. The cameras extended far enough out and around a slightly curved portion that there was enough viewpoints for accurate object pose tracking. Figure 5.6 shows the Optitrack MOCAP system during configuration.

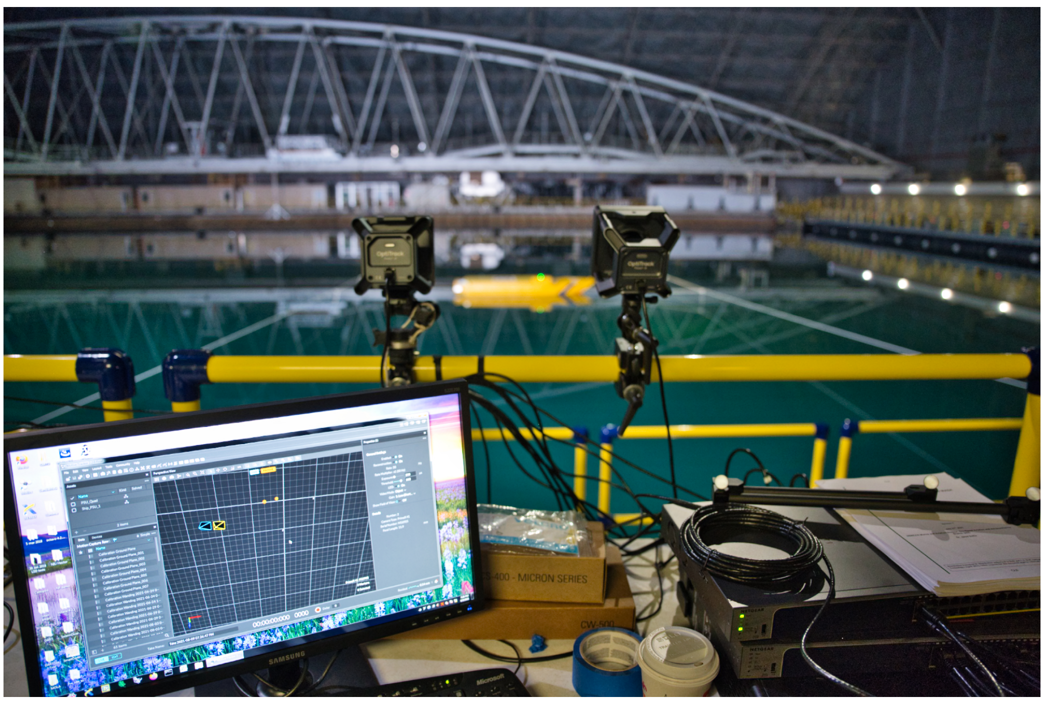


Figure 5.6: Two of the 14 Optitrack MOCAP cameras and the Motive software being configured to track deck and vehicle pose. Figure courtesy NSWCCD.

It was not possible to define the vehicle and deck body frames as easily in the Motive software package used to control the Optitrack system, therefore constant offsets in Equations 5.1 and 5.2 were used to reference the body frames against a single IR reflective marker chosen to be the origin. These coordinate frames are defined parallel to the Optitrack world frame at the time of creation. The Optitrack world frame, denoted O, was an X-East, Y-North, Z-Up coordinate system positioned close to the nominal position and orientation of the deck.

$$\mathbf{r}^{v_o}_{v/v_o} = \begin{bmatrix} 0.0 & -0.122 & 0.0 \end{bmatrix} \tag{5.1}$$

$$\mathbf{r}^{d_o}_{d/d_o} = \begin{bmatrix} 1.197 & 0.0 & 0.012 \end{bmatrix} \tag{5.2}$$

Where v_o and d_o represent the vehicle and deck origin marker frames respectively. The Optitrack reported vehicle and deck origin marker frame positions $\mathbf{r}^{o}_{v_o/o}$, and $\mathbf{r}^{o}_{d_o/o}$ as well as quaternions describing rotation to each respective frame with respect to the Optitrack world frame. The *mocap_optitrack* ROS node running on the GCS communicated with the Windows PC running the Motive software over Ethernet.

5.3 Deck State Estimation Results

5.3.1 State Estimation Quality Dependence on Distance

A test was designed to quantify the dependence of the quality of the state estimates on the distance between the fiducial marker and the UAV. This test was completed in the MOCAP laboratory at Penn State and utilized the VICON MOCAP system. Five 30 second hovers were performed over the fiducial marker array at altitudes varying from approximately 0.5 to 3.5 meters. The VICON system was used for position control in each flight, and the fiducial marker was stationary during the entire test. The total variance in the estimated state, $Tr\left(\mathbf{P}\right)$ was averaged for each flight and plotted against the average Euclidean distance from the center of the tag to the vehicle's center of gravity (measured using the VICON MOCAP system). These two quantities are plotted in Figure 5.7.

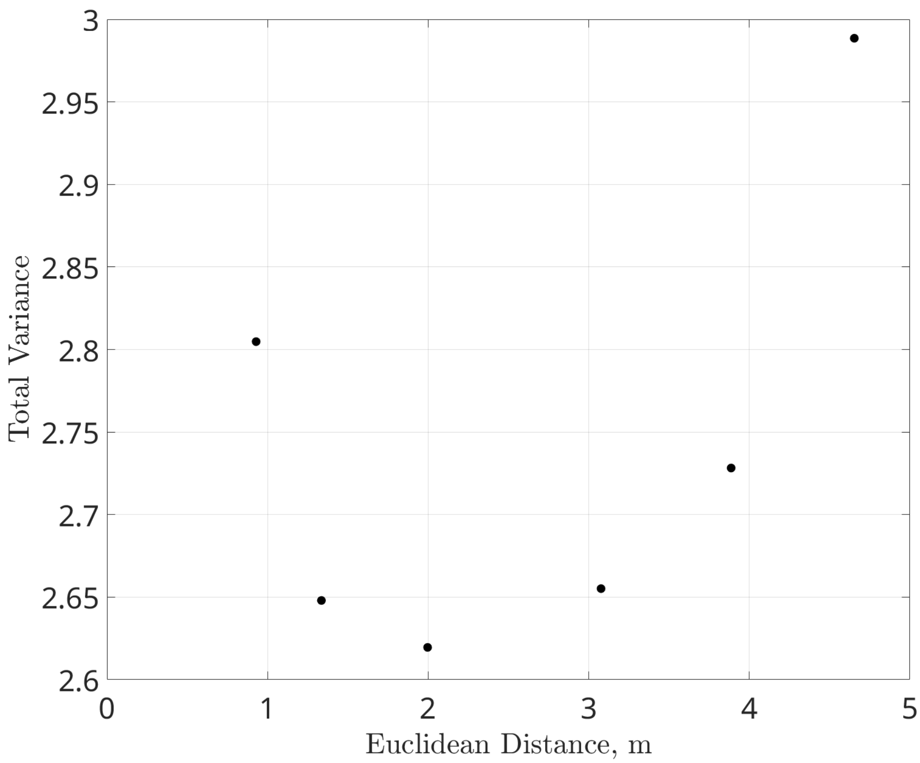


Figure 5.7: Total variance in the UKF state estimates plotted against the distance from the fiducial marker array to the UAV for five 30 second hovering flights using VICON position hold

The total variance in the state estimate is seen to increase when the vehicle is both very close and very far away from the marker array. Theoretically, if more distances were tested, discrete jumps would be visible in Figure 5.7 due to the vision algorithm switching between pose estimation of the largest tag visible in the image.

5.3.2 State Estimation Accuracy in Hover over Moving Deck

Verification of vision system and estimator performance for different wave conditions was performed during the first week of testing in the Maneuvering and Seakeeping

Basin at the Naval Surface Warfare Center in Carderock, Maryland. The MASK is shown in Figure 5.8.

Figure 5.8: Maneuvering and Seakeeping Basin at the Naval Surface Warfare Center in Carderock, Maryland. Figure courtesy NSWCCD.

The Optitrack MOCAP system was configured and tested at this time, including calibrating the MOCAP space, placing reflective spherical markers on both vehicle and ship, and defining clusters of markers as objects. While testing the estimator in hover, all waves were directed at the bow of the ship model.

The vehicle position measured by the Optitrack MOCAP system was forwarded to the flight controller for fusion with the FC's position and heading estimates using PX4's EKF. This allowed position mode to be utilized for hovering the vehicle in a constant position above the deck.

5.3.2.1 Ship Model

A 19.5 ft ship model with a flat deck approximately 1.5 m wide and 3 m long was used for testing in the MASK. The model was loosely tethered in place as shown in Figure 5.9. A 24 in AprilTag fiducial array was printed on Tyvek® for its durability and water resistance. The marker was laminated to 1/8 inch plywood and affixed to the center of the ship deck using double-sided tape.

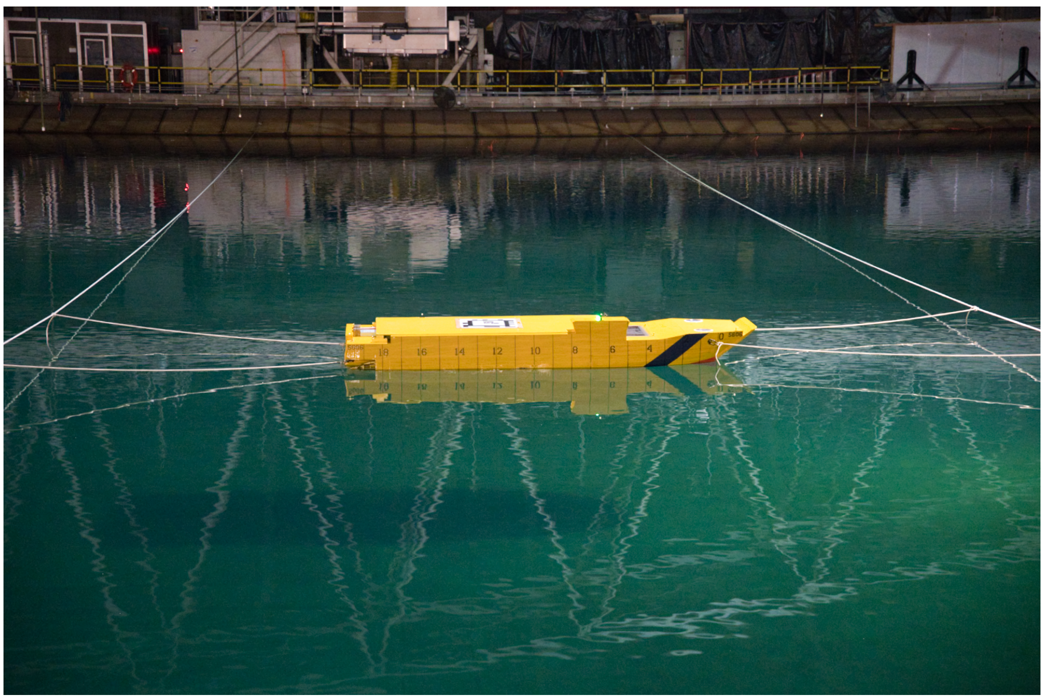


Figure 5.9: Ship model used during estimation and landing tests in the MASK. Figure courtesy NSWCCD.

5.3.2.2 Describing Wave Conditions

The World Meteorological Association (WMO) developed a scale of "sea state codes" designed to be used as a simple integer scale making it possible to understand conditions at a glance and simplify weather broadcasts. The WMO sea state codes were adapted from the Douglas Sea Scale, which is partially correlated with the Beaufort Wind

Scale [36]. Table 5.1 shows the associated wave amplitudes in meters for each code in the system.

Table 5.1: WMO Code Table 3700: sea state, an integer scale for describing wave conditions. Correlated wind speeds from the Beaufort Wind Scale are also shown.

WMO sea state code	Wave Amplitude, m	Wind speed, m s^{-1}
0	no waves	0.0–0.2
1	0.00–0.10	0.3–1.5
2	0.10–0.50	1.6–3.3
3	0.50–1.25	3.4–5.4
4	1.25–2.50	8.0–10.7
5	2.50–4.00	13.9–17.1
6	4.00–6.00	20.8–24.4
7	6.00–9.00	24.5–28.4
8	9.00–14.00	28.5–32.6
9	14.00+	46.2–50.9

5.3.2.3 Scaling Laws

Freud scaling laws are being investigated for use in interpreting results in the context of large scale UAVs and ships currently in use by the US Navy. Figure 5.10 shows representative vehicles used to extrapolate wave conditions used in sub-scale testing.

The DDG-51 has a hull length of 505 ft compared to the 19.5 ft ship model used for testing. The scale factor based on hull length is

$$N_{hull\ length} = \frac{505}{19.5} = 25.90 \tag{5.3}$$

The MV-22 Osprey rotor distance used was 46.58 ft, and the average rotor distance for the two UAVs used in experiments (the hexacopter and the quadcopter described in [37]) was 1.66 ft. For verification purposes, the scale factor based on rotor-to-rotor distance is

(a)

(b)

Figure 5.10: (a) MV-22 Osprey in flight near the Marine Corps Air Station Miramar, San Diego 2014 (Source: Wikimedia Commons) (b) DDG-51, the USS Arleigh Burke, is the main vessel from the *Arleigh Burke* variant guided missile destroyers (Source: Wikimedia Commons)

$$N_{rotor\ distance} = \frac{46.58}{1.66} = 28.1 \tag{5.4}$$

which approximately matches the hull length scale factor. The hull length scale factor was used to scale the wave amplitudes and frequencies using the relationships:

$$A_{waves_{scaled}} = N \cdot A_{waves} \tag{5.5}$$

$$f_{waves_{scaled}} = \frac{f_{waves}}{\sqrt{N}} \tag{5.6}$$

5.3.2.4 Wave Conditions for Hover Testing

Table 5.2 lists the actual wave heights, scaled wave heights, actual frequencies, scaled frequencies, deck roll and pitch amplitudes, and wave direction for the conditions created for testing the deck state estimator.

Table 5.2: Actual and scaled wave amplitudes A_{waves} and frequencies F_{waves} with Optitrack deck roll and pitch Euler angle amplitudes during MOCAP position control hover tests. $A_{\phi_{d/o}}$ and $A_{\theta_{d/o}}$ denote the amplitude of deck roll and pitch Euler angles during the respective conditions. ψ_{waves} is the wave direction measured clockwise from the bow of the ship. Irregular waves were a sum of sinusoids and only the frequency and amplitude of the dominant mode is listed.

Description	A_{waves} (scaled)	F_{waves} (scaled)	$A_{\phi_{d/o}}$	$A_{\theta_{d/o}}$	ψ_{waves}
No Waves	0	-	-	-	
Regular	0.06 m (1.6 m)	0.660 Hz (0.100 Hz)	0.75°	1.74°	0°
Regular	0.12 m (3.1 m)	0.500 Hz (0.0983 Hz)	0.64°	5.47°	0°
Regular	0.18 m (4.7 m)	0.400 Hz (0.0786 Hz)	0.59°	5.91°	0°
Regular	0.24 m (6.2 m)	0.500 Hz (0.0983 Hz)	0.87°	6.54°	0°
Irregular	0.22 m (5.7 m)	0.315 Hz (0.0620 Hz)	0.61°	4.48°	0°
Irregular	0.33 m (8.5 m)	0.281 Hz (0.0552 Hz)	0.91°	6.53°	0°

5.3.2.5 Estimator Error in Hover over Moving Deck

Detailed flight test data for seven hovering flights above the deck experiencing wave conditions in Table 5.2 are given in Appendix C. The plots in Appendix C include measurements of relative deck pose directly from the vision system and also provide a visual comparison between estimated pose and MOCAP ground truth. This section summarizes the results from these seven flights

The error in the UKF position estimate was calculated using the Optitrack measured positions of the vehicle IMU and AprilTag array center as ground truth:

$$\epsilon_{\mathbf{r}^v_{d/v}} = \begin{bmatrix} \epsilon_x & \epsilon_y & \epsilon_z \end{bmatrix}^T = \mathbf{r}^v_{d/v} - \hat{\mathbf{r}}^v_{d/v} \tag{5.7}$$

The error in the UKF Euler angle estimate was calculated similarly using the Optitrack measured vehicle IMU and AprilTag array Euler angles with respect to the

Optitrack/world inertial frame:

$$\epsilon_{\Phi_{d/v}} = \begin{bmatrix} \epsilon_\phi & \epsilon_\theta & \epsilon_\psi \end{bmatrix}^T = \Phi_{d/v} - \hat{\Phi}_{d/v} \tag{5.8}$$

Figures 5.11 and 5.12 show the position and orientation error for eight of the deck state estimator test flights. The wave conditions for these flights are given in Table 5.2.

The error in the relative deck position and orientation estimates for each hovering

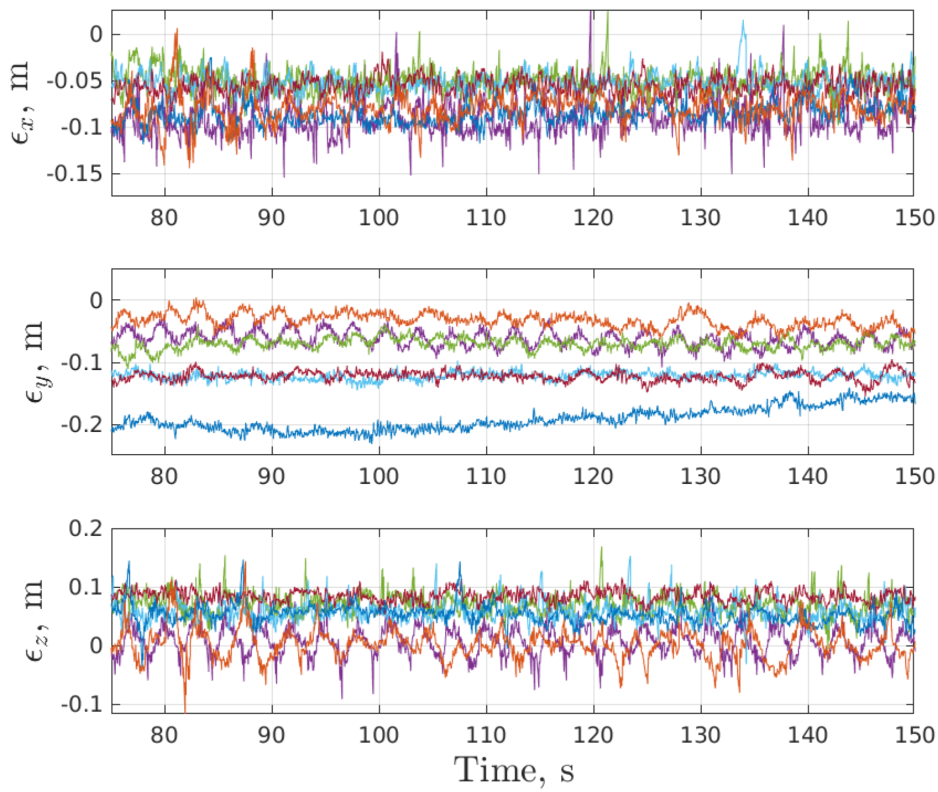


Figure 5.11: Error in estimated deck position in the vehicle frame for seven hovering flights above the ship model in wave conditions listed in Table 5.2

flight test conducted is summarized in Table 5.3. The translational and rotational

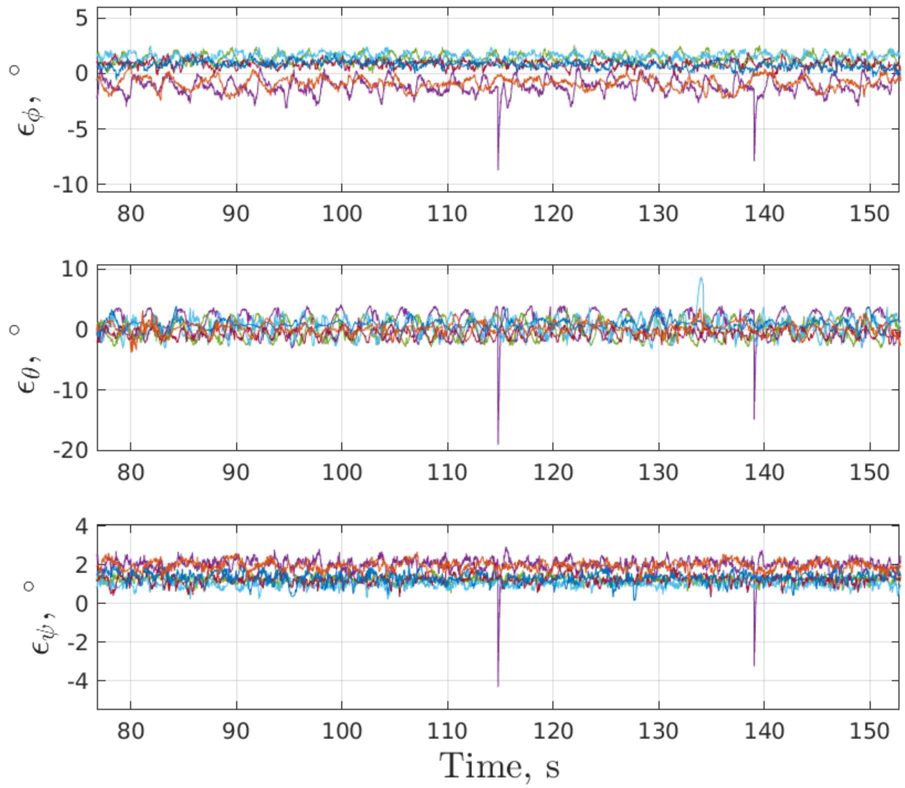


Figure 5.12: Error in estimated Euler angles from the vehicle to the deck for seven hovering flights above the ship model in wave conditions listed in Table 5.2

error was calculated at each time step using the relationships

$$\epsilon_{pos} = \sqrt{{\epsilon_x}^2 + {\epsilon_y}^2 + {\epsilon_z}^2} \tag{5.9}$$

and

$$\epsilon_{rot} = \sqrt{{\epsilon_\phi}^2 + {\epsilon_\theta}^2 + {\epsilon_\psi}^2} \tag{5.10}$$

which were averaged for a 35-second portion of each flight where the tag was visible, yielding average values for translational and rotational error, $\bar{\epsilon}_{pos}$ and $\bar{\epsilon}_{rot}$ respectively. The standard deviation of the errors are small compared to the mean, indicating

Table 5.3: Relative deck pose state estimate translational and rotational error and standard deviations listed by wave amplitude

Amplitude, m	$\bar{\epsilon}_{pos}$, m	σ_{pos}, m	$\bar{\epsilon}_{rot}$, $^\circ$	σ_{rot}, $^\circ$
0	0.083	0.0057	1.832	0.2090
0.06	0.158	0.0080	1.833	0.3735
0.12	0.147	0.0088	2.312	0.5123
0.18	0.114	0.0137	2.314	0.5339
0.24	0.113	0.0164	3.137	0.7270
0.22	0.234	0.0096	1.966	0.3434
0.33	0.092	0.0163	2.393	0.4754

the errors were more systemic than random. Two possible sources of systemic error were identified. As alluded to in Section 4.1.2, small inaccuracies in the setting of camera orientation relative to the vehicle would have caused large errors in deck pose estimates that increased proportionally with the distance between the UAV and the marker. Additionally, some error was likely introduced from the Optitrack MOCAP system's deck and vehicle y position measurements, which had a high uncertainty relative to the other axes due to the layout of cameras along only one side of the MASK. A more even spatial distribution of cameras with more varied vantage points would have reduced the uncertainty in ground truth position measurements along this axis.

Chapter 6

Landing Flight Tests

6.1 Onboard Vision Based Landings

During the second week of testing in the MASK, autonomous landings were first conducted on a static ship model to verify the system was capable of landing. When it was verified that estimation and control algorithms were functioning as intended, landings were performed on moving model ship deck forced by sinusoidal waves from head on and quartering directions.

6.1.1 Landing Procedure

The sequence of events for an autonomous landing was:

1. Pilot enables autonomous mode using a switch.

2. Hover in the current location for 3 seconds.

3. UAV is commanded to a waypoint 3.5 m above and behind the deck.

4. After the waypoint is reached, wait for 10 seconds.

5. UAV is Commanded to an approach point 0.5 m above and behind the deck.

6. After the approach point is reached, wait at the approach point for 10 seconds.

7. Command the final descent phase.

8. Pilot disarms vehicle.

6.1.2 Wave Conditions for Landing Flight Tests

Table 6.1 shows the wave conditions created by the 216 individually controlled wave boards at the edge of the MASK during estimator-based landings.

Table 6.1: Actual and scaled wave amplitudes A_{waves} and frequencies F_{waves} for on-board camera and estimation based landings. Optitrack deck roll and pitch Euler angle amplitudes $A_{\phi_{d/o}}$ and $A_{\theta_{d/o}}$ denote one sampling of deck roll and pitch Euler angle amplitudes for each wave condition. ψ_{waves} is the wave direction measured clockwise from the bow of the ship. All waves during estimation based landings were regular, noisy sinusoids. The number of successful landings is shown beside the total attempts for each condition in the last column.

A_{waves} (scaled)	F_{waves} (scaled)	$A_{\phi_{d/o}}$	$A_{\theta_{d/o}}$	ψ_{waves}	No. Landings (attempts)
0	-	-	-	-	3 (3)
0.15 m (3.9 m)	0.333 m (0.0654 Hz)	0.469°	3.96°	0°	2 (2)
0.20 m (5.2 m)	0.333 m (0.0654 Hz)	0.690°	5.35°	0°	2 (4)
0.15 m (3.9 m)	0.500 m (0.0983 Hz)	0.595°	4.27°	0°	2 (2)
0.10 m (2.6 m)	0.250 m (0.0497 Hz)	0.800°	1.32°	30°	1 (2)
0.10 m (2.6 m)	0.500 m (0.0983 Hz)	1.840°	4.12°	30°	2 (2)
0.15 m (3.9 m)	0.500 m (0.0983 Hz)	3.060°	6.05°	30°	3 (4)[a]

[a]An unidentified software or mechanical failure caused the motors to stop on the last flight during the approach phase, however the vehicle was undamaged.

6.1.3 Landing Flight Test Results

15 landings were successfully completed using the Apriltag marker array, on-board vision processing and state estimation capabilities of the hexacopter. The hexacopter is shown performing a deck landing in Figure 6.1.

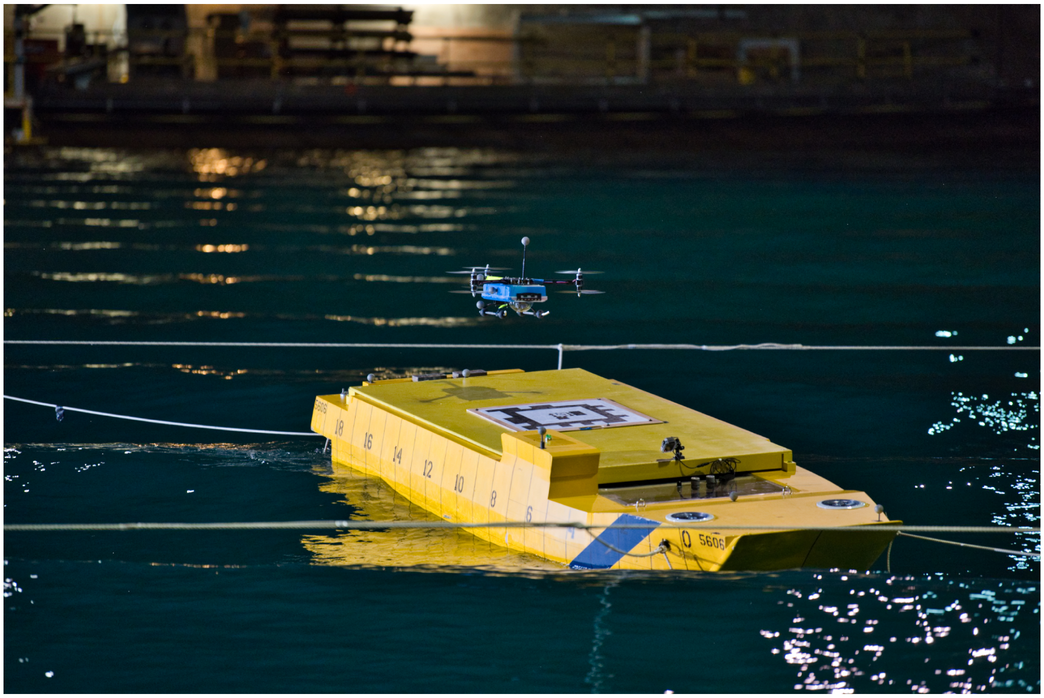


Figure 6.1: Hexacopter autonomously approaching the ship model in quartering waves. Figure courtesy NSWCCD.

Figures 6.2 and 6.3 show the ground truth and estimated relative deck position and orientation respectively for a single landing in quartering waves representing a scaled sea state of five (using the factors from Section 5.3.2.3).

Figure 6.4 shows the three-dimensional path of the vehicle during the autonomous portion of all 15 successful estimator-based landings, while Figure 6.5 shows the deck motion during those same landings. The positions in Figures 6.4 and 6.5 are referenced from the Optitrack world frame which is an X-East, Y-North, Z-Up coordinate system. Figure 6.6 shows more detail of both the vehicle and deck paths during the final approach. The position time histories in Figures 6.4, 6.5, and 6.6 were obtained using the Optitrack motion capture system described in Section 5.2.2.

Figures 6.7 and 6.8 show the relative deck position and orientation with respect to the vehicle frame. The positions in figure 6.7 are with respect to the moving-rotating

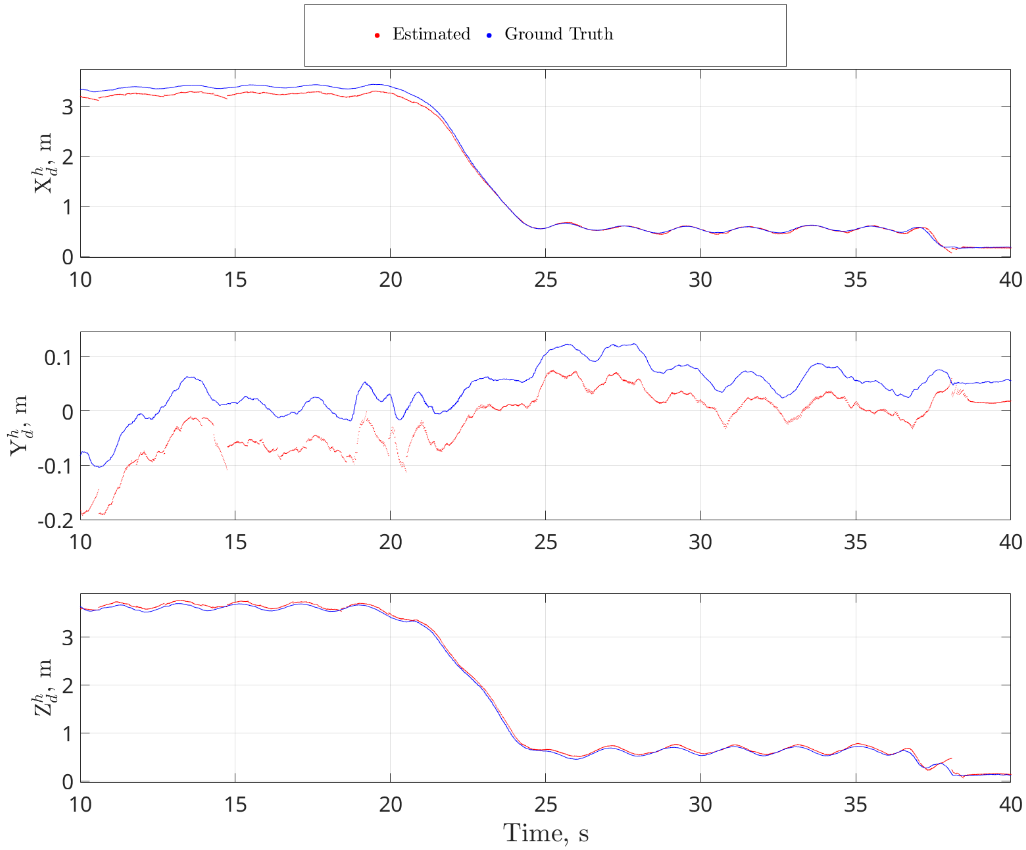


Figure 6.2: Deck position in the vehicle frame for a single landing in quartering waves with amplitude of 0.1 m and frequency of 0.5 Hz. Red dots represent the UKF estimates while blue dots are calculated from Optitrack MOCAP system measurements.

hexacopter body frame, which is an X-Front, Y-Right, Z-Down coordinate system. The orientations in figure 6.8 are 321 Euler angles describing three successive intrinsic rotations from the vehicle to the deck.

The Euclidean distance between the camera sensor origin and the center of the AprilTag array,

$$\|\mathbf{r}_d^c\| = \sqrt{(x_d^c)^2 + (y_d^c)^2 + (z_d^c)^2} \tag{6.1}$$

was calculated from measurements of the vehicle and deck inertial positions collected using the Optitrack Motion Capture system along with constant camera displacement

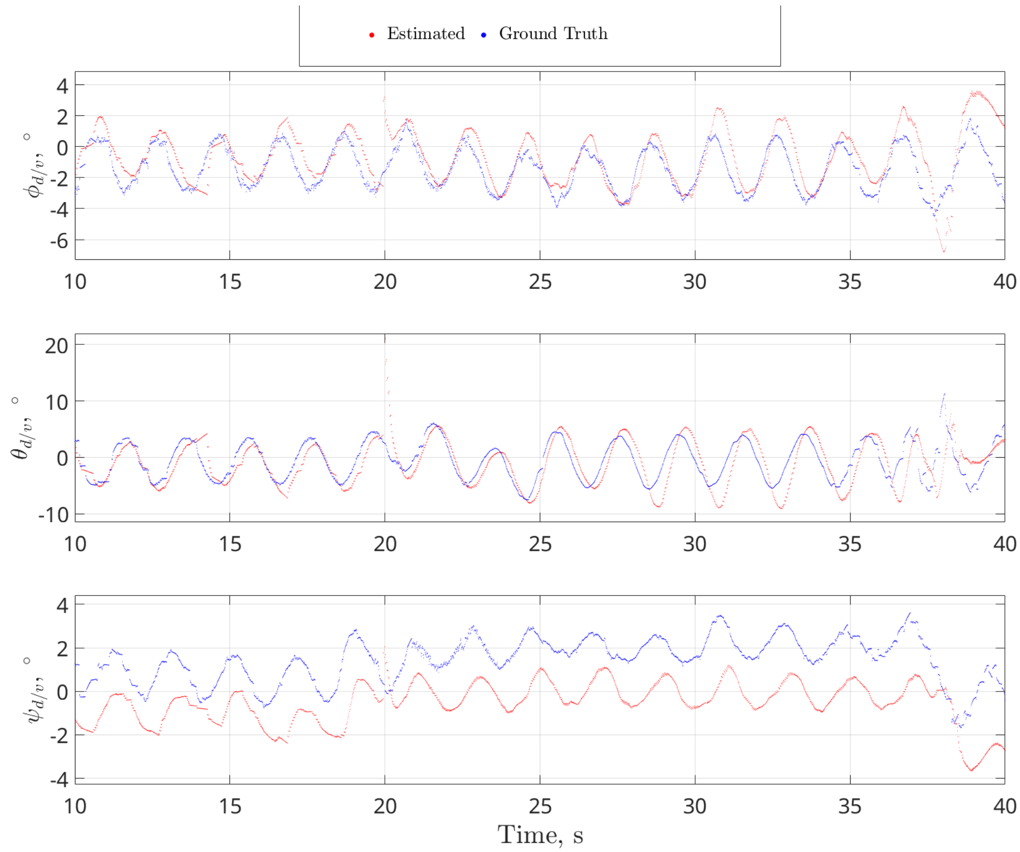


Figure 6.3: Euler angles describing deck orientation with respect to the vehicle for a single landing in quartering waves with amplitude of 0.1 m and frequency of 0.5 Hz. Red dots represent the UKF estimates while blue dots are calculated from Optitrack MOCAP system measurements.

from Equation 4.6 and the AprilTag array displacement from the deck origin in Equation 5.2. The quantity $\|\mathbf{r}_d^c\|$ is plotted for the 15 successful vision-based landings in Figure 6.9.

As shown in Figure 6.9, the system automatically reduced the total distance to the marker after which the motors were disarmed, ending the autonomous portion of the flights. In some of the landings, the smallest marker in the AprilTag array was still detected after touchdown.

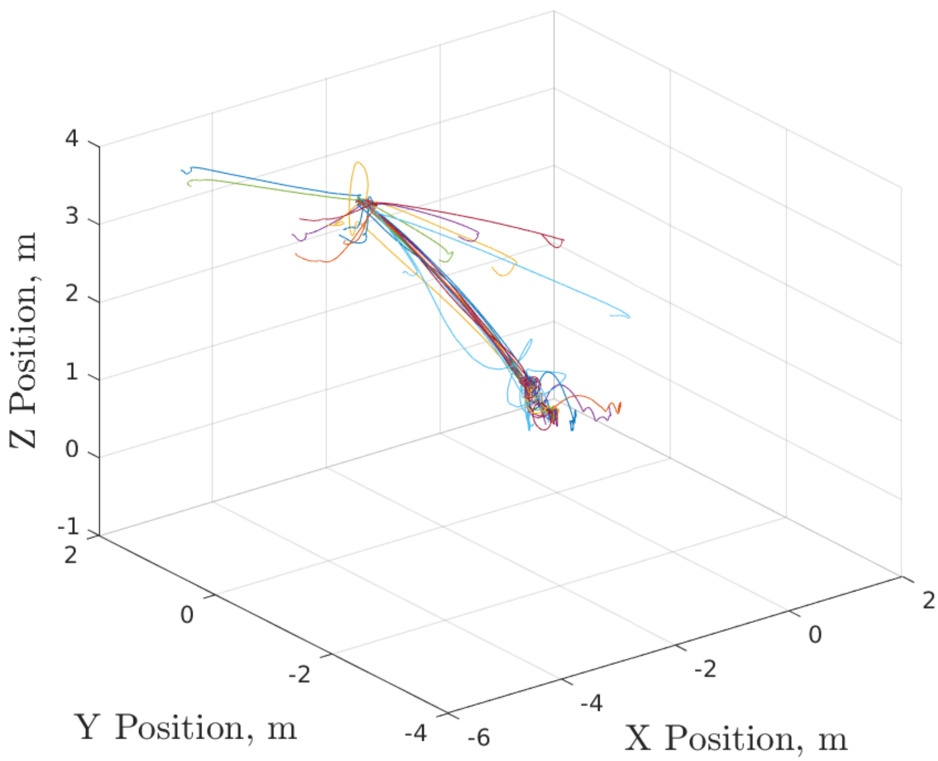


Figure 6.4: Optitrack measured vehicle position for 15 estimator-based landings. The vehicle was in a different location each time autonomy was enabled; the individual flight paths to the initial waypoint 3.5 m above and behind the deck are visible. After this point was reached the trajectories overlap.

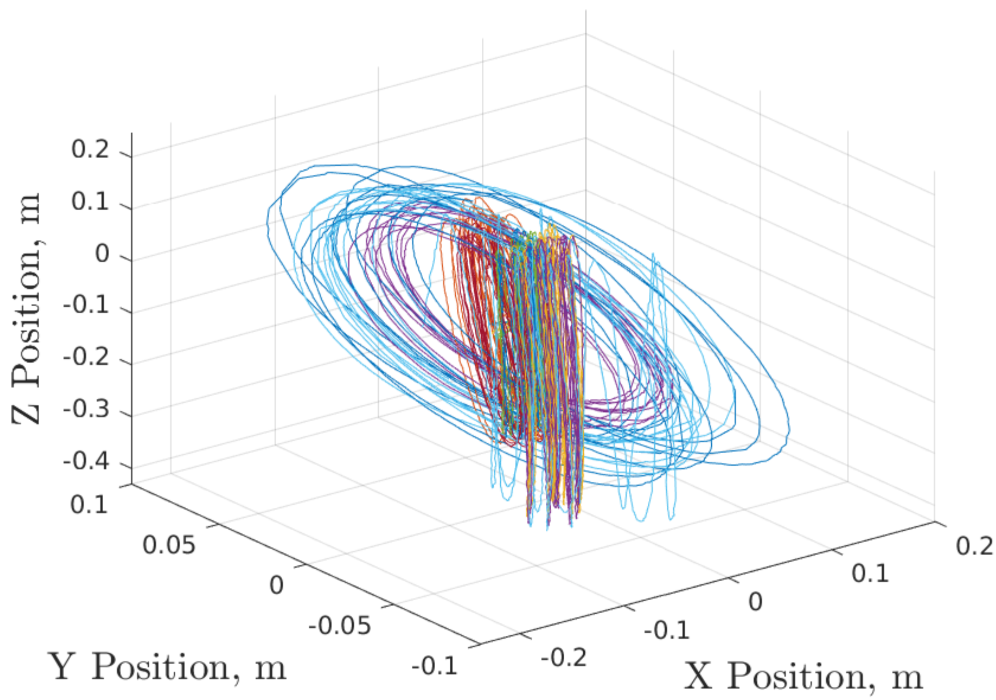


Figure 6.5: Optitrack measured Apriltag array positions for 15 estimator-based landings

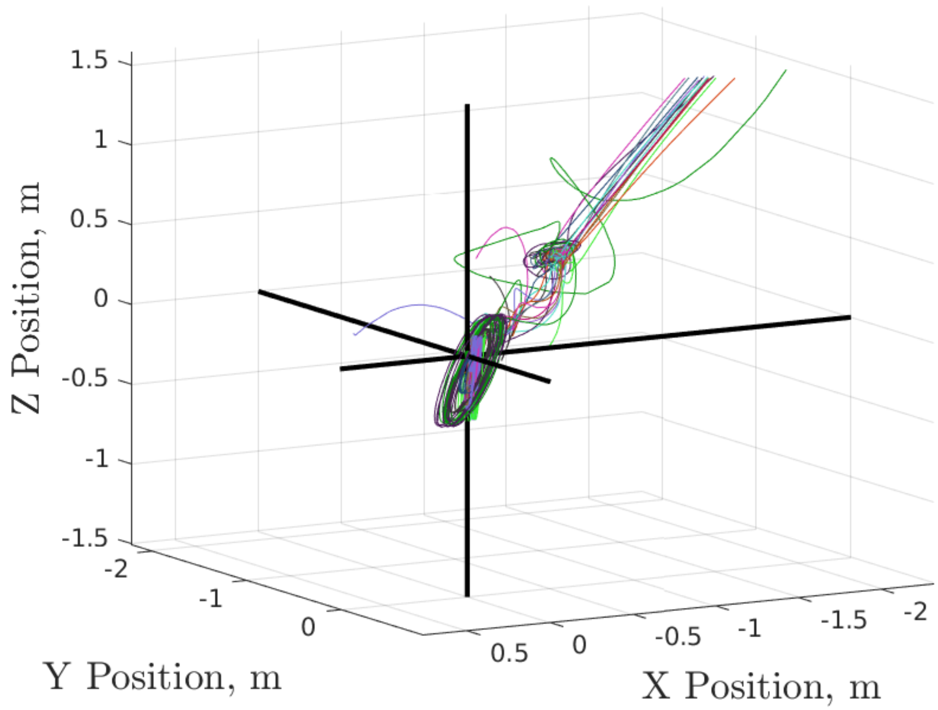


Figure 6.6: Optitrack measured vehicle and Apriltag array position time histories during the final approach to the deck for 15 estimator-based landings. The vehicle was commanded to hover 0.5 m above and behind the deck for 10 s before landing.

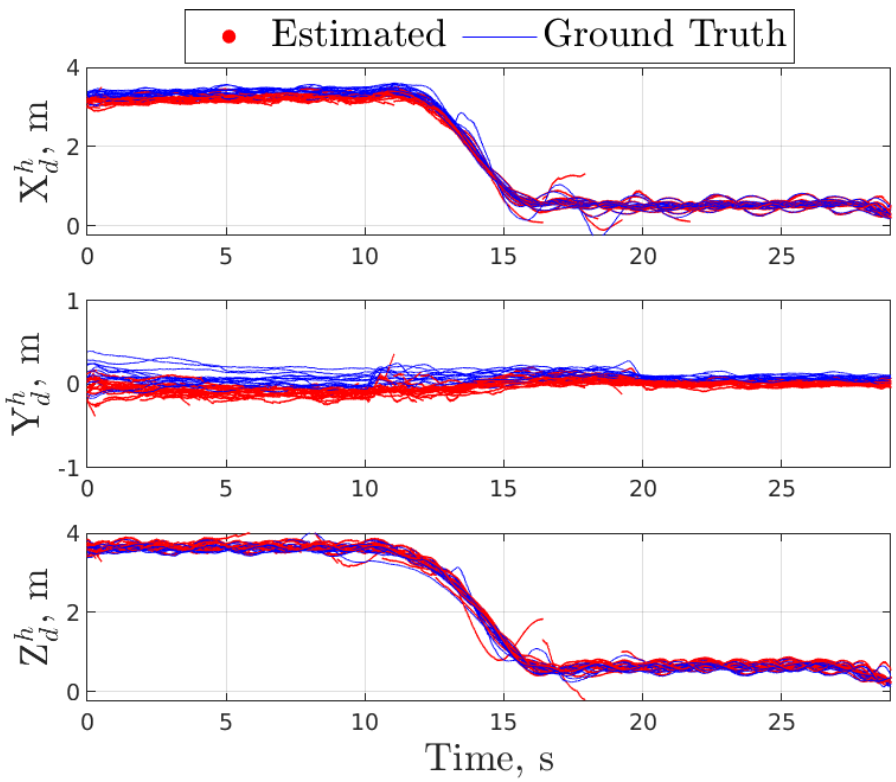


Figure 6.7: Components of both the estimated and Optitrack measured relative deck position in the vehicle frame for 15 vision-based landings

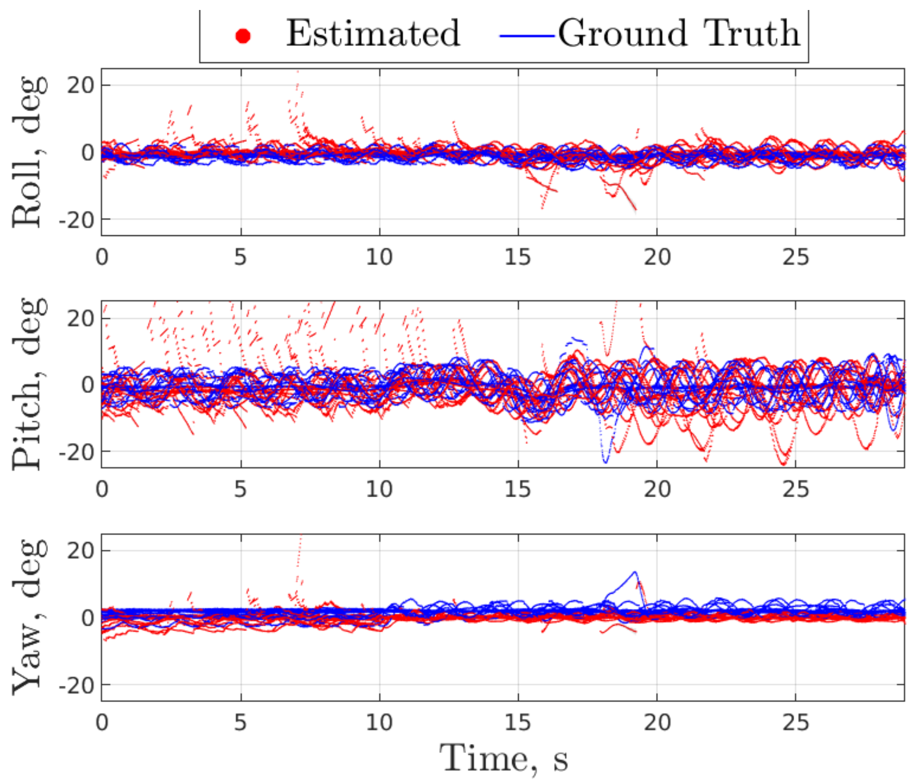


Figure 6.8: Euler angles describing successive intrinsic rotations from the FRD Vehicle frame to the deck frame for 15 vision-based landings

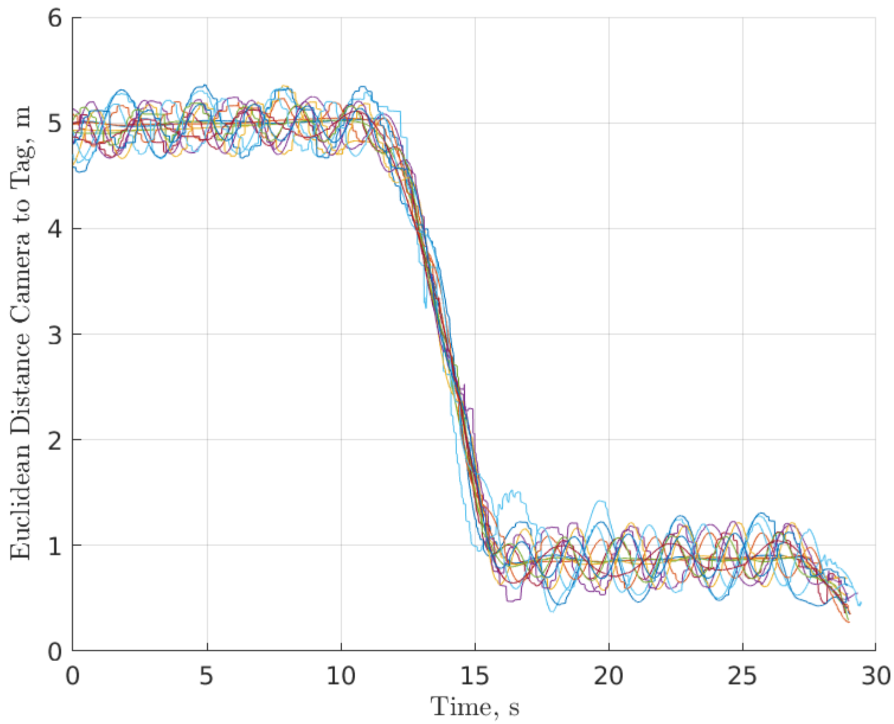


Figure 6.9: Euclidean distance from the camera sensor plane origin to tag center for 15 vision-based landings

Chapter 7

Conclusions and Future Work

7.1 Summary of Contributions

A completely integrated autonomous deck landing solution was implemented using a single monocular camera and low-cost IMU. The only ship augmentation required by the landing system was a printed fiducial marker. Solutions for deck localization, relative state estimation, trajectory generation and trajectory following were integrated on a newly constructed water-resistant sUAS platform.

Computer vision systems for landing area identification and relative pose estimation were evaluated in the context of maritime precision landing. A recursive array of AprilTag fiducial markers from family TagCustom48h12 was selected for testing. The resulting compound marker has significant advantages in precision landing applications because of its ability to produce accurate relative pose estimates from a wider range of distances than singular fiducial markers or other non-scalable visual features. This ultimately meant the UAV was able to maintain a high quality relative deck state estimate when it was both far away and right above the deck, or anywhere in between. A monocular camera with an embedded quad-core processor was used to detect and localize the marker array at a rate of 48 Hz.

The relative deck position, velocity, attitude, and angular velocity was estimated using an Unscented Kalman Filter that fused measurements from the vision system and IMU. All required sensors where carried on the vehicle and all algorithms for state estimation, trajectory planning, and control were hosted on an ODROID XU4 single board computer carried onboard the vehicle.

Two stages of flight tests were designed and completed. First, in-flight relative deck pose estimation accuracy was analyzed using motion capture in static (UAV hovering above stationary marker) and dynamic (UAV following landing trajectories to the ship deck in motion) conditions. The accuracy of vision based relative deck pose estimation was evaluated using motion capture system as ground truth. Error in relative deck pose estimates was found to be less than 23 cm translationally and 5° rotationally from distances less than 5 m. Second, the holistic capabilities of the landing were demonstrated in high sea states. 15 autonomous landings were performed on a small scale ship model in sea states as high as six (at scale representing the Bell Boeing MV-22 Osprey and DDG-51 Destroyer).

7.2 Lessons Learned

Doing all processing onboard the vehicle poses significant hardware and software implementation challenges. The UAV must carry and power computer hardware, which reduces practical flight times for electric aircraft.

Error in state estimates was underestimated due to the values of observation and process noise matrices selected. The values should be tuned to be large enough so that the estimated state covariance sufficiently encapsulates actual error in the estimated state.

7.3 Recommendations for Future Work

Deck motion forecasting is a key area that can be investigated to optimize landing performance in the moment of touchdown. It would be best to plan trajectories such that touchdown occurs during optimal periods of deck motion, or during wave peaks or troughs where deck velocity and angular velocity are small.

The vision system can be further optimized to be more robust to challenging lighting conditions. Shadows cast on the fiducial marker by the UAV may have caused some missed detections. Although this issue could be solved by utilizing a proper lighting installation on the deck or UAV, the camera sensor driver used can be tuned for better low light performance.

Further study of required marker size and camera resolutions for detection at different distances would be required to improve the maximum detection distance. Additionally, the ratio of marker sizes that make up the recursive array and its relationship with camera field of view could be worth analyzing.

Final trajectory phases could be improved further. The landings shown in this paper do not attempt to match attitude with the deck at the moment of impact. This would, in general, be possible using the vision system described here because the state estimate quality should not be significantly degraded at small distances due to the scalable nature of the fiducial employed.

The periods of waiting and hovering in the trajectory should be removed for further testing. The low pass filter on estimated deck position that is used to calculated vehicle commands resulted in inaccurate position tracking of the deck during the final approach phase.

References

[1] Ortega, T. J. (2014) "Photo Release – Northrop Grumman, MQ-8C Fire Scout Takes First Flight on USS Jason Dunham," *Northrop Grumman Newsroom*.

[2] Francisco J. Romero-Ramirez, R. M.-S. and R. Medina-Carnicer (2018) "Speeded up detection of squared fiducial markers," *Image and Vision Computing*, **76**, pp. 38–47.
URL `https://www.sciencedirect.com/science/article/pii/S0262885618300799`

[3] Krogius, M., A. Haggenmiller, and E. Olson (2019) "Flexible Layouts for Fiducial Tags," *IEEE International Conference on Intelligent Robots and Systems*, pp. 1898–1903.

[4] Gautam, A., P. B. Sujit, and S. Saripalli (2014) "A survey of autonomous landing techniques for UAVs," *2014 International Conference on Unmanned Aircraft Systems, ICUAS 2014 - Conference Proceedings*, (July 2019), pp. 1210–1218.

[5] Truskin, B. L. and J. W. Langelaan (2013) "Vision-based deck state estimation for autonomous ship-board landing," in *Annual Forum Proceedings - AHS International*, May, pp. 1693–1711.

[6] Holmes, W. K. and J. W. Langelaan (2016) "Autonomous ship-board landing using monocular vision," *Annual Forum Proceedings - AHS International*, **4**, pp. 3391–3405.

[7] Holmes, W. (2017), "Vision-Based Relative Deck State Estimation Used with Tau Based Landings," .

[8] Lumsden, B., C. Wilkinson, and G. Padfield (1998) "Challenges at the helicopter-ship dynamic interface," , pp. 15–18.

[9] Mohammadi, A., Y. Feng, C. Zhang, S. Baek, and S. Rawashdeh (2018) "Autonomous landing of a UAV on a moving platform using model predictive control," *Drones*, **2**(4), pp. 1–15.

[10] Ling, K., D. Chow, A. Das, and S. L. Waslander (2014) "Autonomous maritime landings for low-cost VTOL aerial vehicles," *Proceedings - Conference on Computer and Robot Vision, CRV 2014*, pp. 32–39.

[11] Lee, B., V. Saj, M. Benedict, and D. Kalathil (2020) "A vision-based control method for autonomous landing of vertical flight aircraft on a moving platform without using gps," *Vertical Flight Society's 76th Annual Forum and Technology Display*, 2008.05699.

[12] ——— (2021) "A Deep Reinforcement Learning Control Strategy for Vision-based Ship Landing of Vertical Flight Aircraft," in *AIAA AVIATION 2021 FORUM*, American Institute of Aeronautics and Astronautics, Reston, Virginia.
URL https://arc.aiaa.org/doi/10.2514/6.2021-3218

[13] Lee, B., V. Saj, and M. Benedict (2021) "Machine learning vision and nonlinear control approach for autonomous ship landing of vertical flight aircraft," *77th Annual Vertical Flight Society Forum and Technology Display, FORUM 2021: The Future of Vertical Flight.*

[14] Bhargavapuri, M., A. K. Shastry, H. Sinha, S. R. Sahoo, and M. Kothari (2019) "Vision-based autonomous tracking and landing of a fully-actuated rotorcraft," *Control Engineering Practice*, **89**(May), pp. 113–129.
URL https://doi.org/10.1016/j.conengprac.2019.05.015

[15] Roderick, W. R., M. R. Cutkosky, and D. Lentink (2017) "Touchdown to take-off: At the interface of flight and surface locomotion," *Interface Focus*, **7**(1).

[16] Lee, D. N., R. J. Bootsma, M. Land, D. Regan, and R. Gray (2009) "Author's update: General Tau Theory: Evolution to date," *Perception*, **38**(6), pp. 837–858, pMID: 19806967.
URL https://doi.org/10.1068/pmklee

[17] Gajjar, B. I. and J. Zalewski (2004) "On-ship landing and takeoff of Unmanned Aerial Vehicles (UAV'S)," *IFAC Proceedings Volumes (IFAC-PapersOnline)*, **37**(20), pp. 42–46.
URL http://dx.doi.org/10.1016/S1474-6670(17)30568-2

[18] Pravitra, J., J.-P. B. Clarke, and E. N. Johnson (2019) "Landing a Fixed-Wing UAV on a Moving Platform: A Pseudospectral Optimal Control Approach," in *AIAA Aviation 2019 Forum*, American Institute of Aeronautics and Astronautics, Reston, Virginia.
URL https://arc.aiaa.org/doi/10.2514/6.2019-3257

[19] Pravitra, J. and E. N. Johnson (2020) "Adaptive Control for Attitude Match Station-Keeping and Landing of a Fixed-Wing UAV onto a Maneuvering Platform,"

in *AIAA Scitech 2020 Forum*, American Institute of Aeronautics and Astronautics, Reston, Virginia.
URL https://arc.aiaa.org/doi/10.2514/6.2020-1082

[20] Pravitra, J. (2021) *Shipboard UAS Operations with Optimized Landing Trajectories*, Ph.D. thesis, Pennsylvania State University.

[21] Yu, G., Y. Hu, and J. Dai (2020) "TopoTag: A Robust and Scalable Topological Fiducial Marker System," *IEEE Transactions on Visualization and Computer Graphics*, (3), pp. 1–12, 1908.01450.

[22] Fiala, M. (2005) "ARTag, a fiducial marker system using digital techniques," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 2, pp. 590–596 vol. 2.

[23] Nitsche, M., T. Krajník, P. Čížek, M. Mejail, and T. Duckett (2015) "WhyCon: An Efficent, Marker-based Localization System," in *IROS Workshop on Open Source Aerial Robotics*.

[24] Stevens, B. L., F. L. Lewis, and E. N. Johnson (2015) *Aircraft control and simulation: Dynamics, controls design, and autonomous systems: Third edition*.

[25] Garrido-Jurado, S., R. Muñoz-Salinas, F. Madrid-Cuevas, and R. Medina-Carnicer (2016) "Generation of fiducial marker dictionaries using Mixed Integer Linear Programming," *Pattern Recognition*, **51**, pp. 481–491.
URL https://www.sciencedirect.com/science/article/pii/S0031320315003544

[26] Romero-Ramirez, F. J., R. Muñoz-Salinas, and R. Medina-Carnicer (2021) "Tracking fiducial markers with discriminative correlation filters," *Image and Vision Computing*, **107**, p. 104094.
URL https://www.sciencedirect.com/science/article/pii/S0262885620302262

[27] Olson, E. (2011) "AprilTag: A robust and flexible visual fiducial system," *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 3400–3407.

[28] Wang, J. and E. Olson (2016) "AprilTag 2: Efficient and robust fiducial detection," in *Proceedings of the EEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

[29] Zakiev, A., K. Shabalina, and E. Magid (2019) "Pilot Virtual Experiments on ArUco and AprilTag Systems Comparison for Fiducial Marker Rotation Resistance," *Proceedings of International Conference on Artificial Life and Robotics*, **24**, pp. 132–135.

[30] Zakiev, A., T. Tsoy, K. Shabalina, E. Magid, and S. K. Saha (2020) "Virtual Experiments on ArUco and AprilTag Systems Comparison for Fiducial Marker Rotation Resistance under Noisy Sensory Data," *Proceedings of the International Joint Conference on Neural Networks*.

[31] Richardson, A., J. Strom, and E. Olson (2013) "AprilCal: Assisted and repeatable camera calibration," *IEEE International Conference on Intelligent Robots and Systems*, pp. 1814–1821.

[32] Wu, Y. (2006) "PnP problem revisited," *Journal of Mathematical Imaging and Vision*, **24**(1), pp. 131–141.

[33] Persson, M. and K. Nordberg (2018) "Lambda Twist: An Accurate Fast Robust Perspective Three Point (P3P) Solver," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, **11208 LNCS**, pp. 334–349.

[34] Julier, S. J. and J. K. Uhlmann (1997) "New extension of the Kalman filter to nonlinear systems," *Signal Processing, Sensor Fusion, and Target Recognition VI*, **3068**, p. 182.

[35] Cai, Z. and D. Zhao (2006) "Unscented Kalman filter for non-linear estimation," *Geomatics and Information Science of Wuhan University*, **31**(2), pp. 180–183.

[36] Owens, E. H. (1984) *Beaches and Coastal Geology, Sea conditions*, Springer US, New York, NY, pp. 722–722.
URL `https://doi.org/10.1007/0-387-30843-1_397`

[37] Hendrick, C. (2022) "Scaled Experiments in Flight Control Design for Autonomous Landing in High Sea States," in *AIAA AVIATION 2022 FORUM*, American Institute of Aeronautics and Astronautics, Chicago, Illinois.

Appendix A
ROS TF Tree

This appendix summarizes the transformation tree as configured in ROS as used by the deck landing system presented in this paper. The output in Figure A.1 was generated using the tf2 package:

```
$ rosrun tf2_tools view_frames.py
```

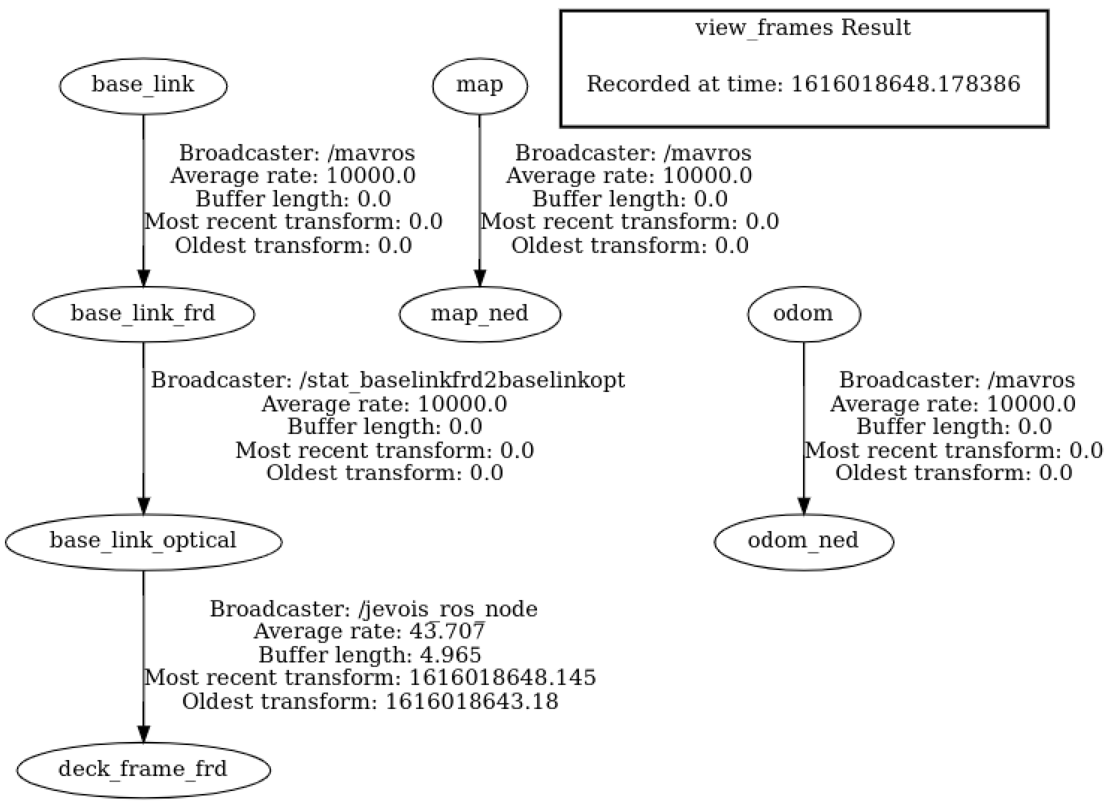


Figure A.1: The Transformation (TF) tree created in and used by ROS to keep track relative object poses. The chain on the left shows the pose transformation between the camera and vehicle body frame.

Appendix B

Generating a new AprilTag Family

Bash commands to generate AprilTag family tagCustom24h13 are as follows:

```
ant

java -cp april.jar april.tag.TagFamilyGenerator \
    custom_wwwwwwwww \
           wbbbbbbbw \
           wbdddddbw \
           wbdddddbw \
           wbddxddbw \
           wbdddddbw \
           wbdddddbw \
           wbbbbbbbw \
           wwwwwwwww 13
```

```
ant
```

```
java -cp april.jar april.tag.TagToC april.tag.TagCustom24h13
```

```
java -cp april.jar april.tag.GenerateTags \
    april.tag.TagCustom24h13 tagCustom24h13
```

Appendix C

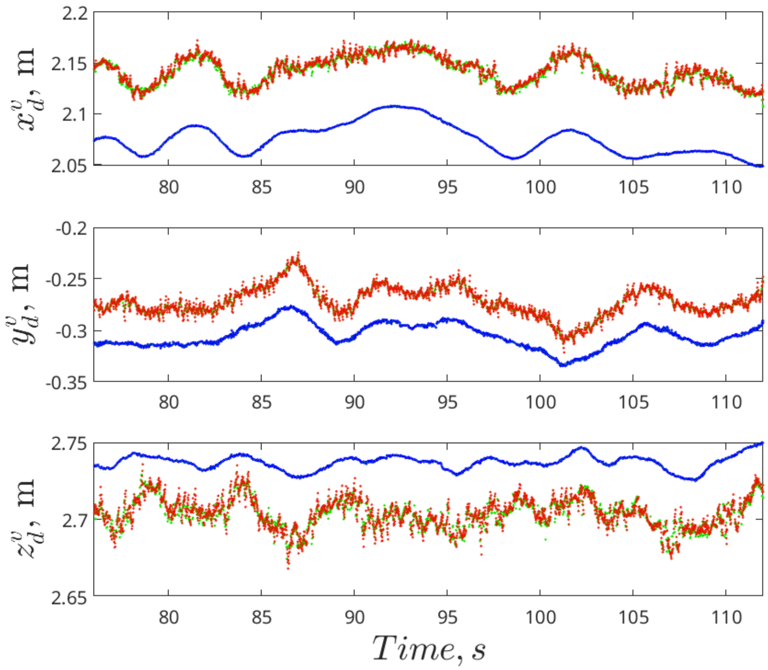
Full Flight Data for Deck State Estimator Hover Tests

Each paired chart in this appendix shows the position and orientation of the deck with respect to the vehicle from three sources:

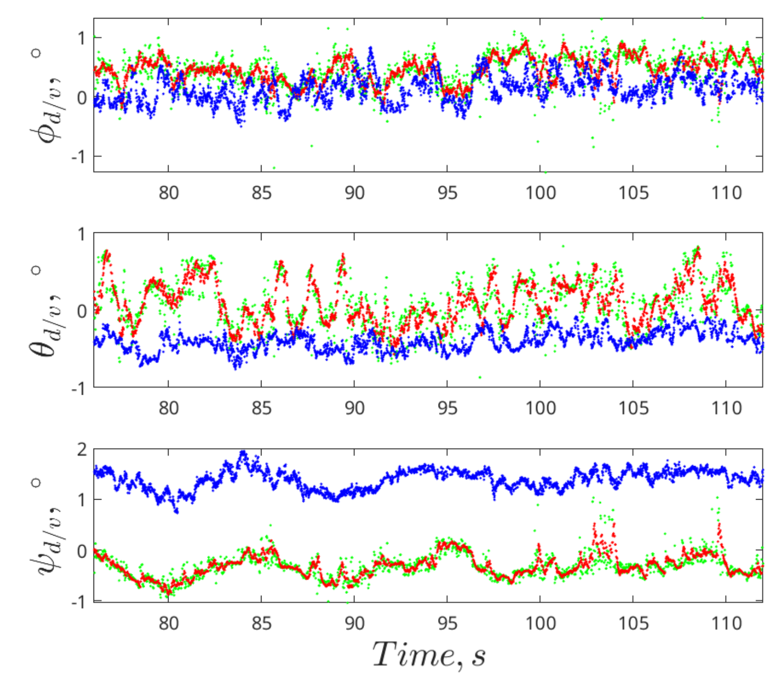
1. Green dots are deck pose measurements directly from the vision system, transformed to the body frame

2. Blue dots are the measurements of relative deck pose calculated from Optitrack MOCAP data

3. Red dots are components of the UKF estimated relative deck pose

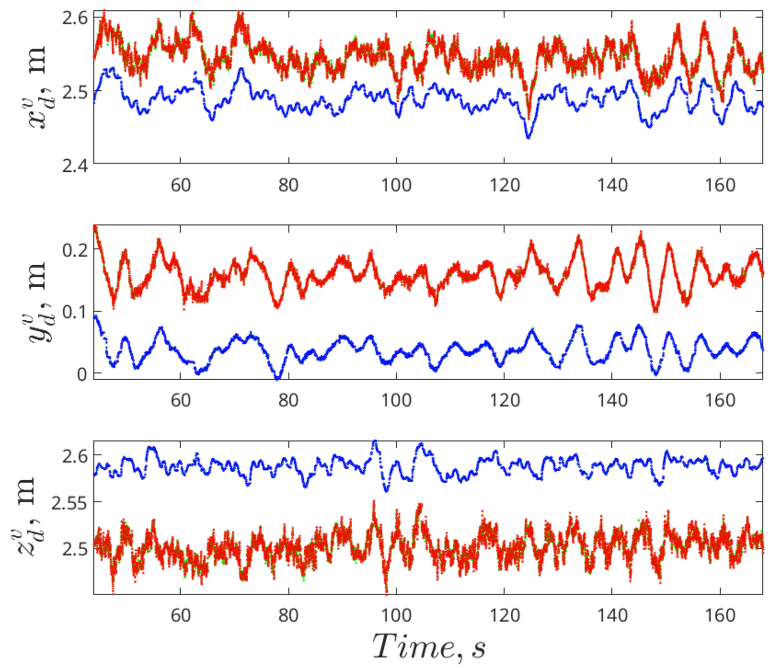


(a)

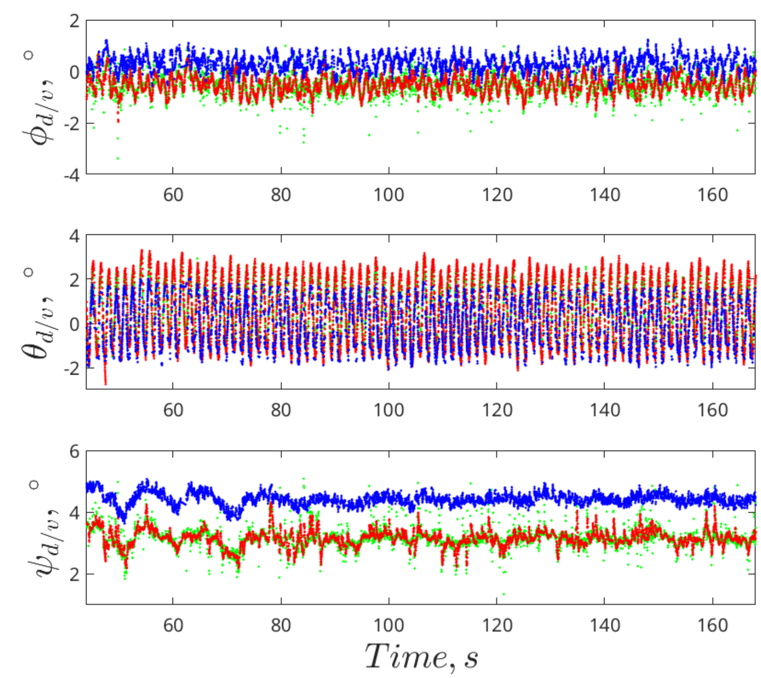


(b)

Figure C.1: Static deck (a) Position (b) Orientation

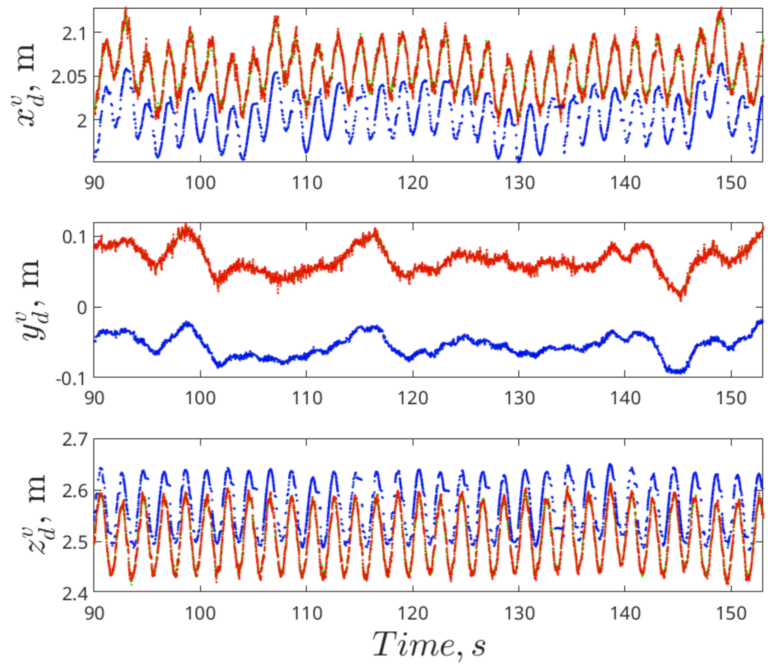


(a)

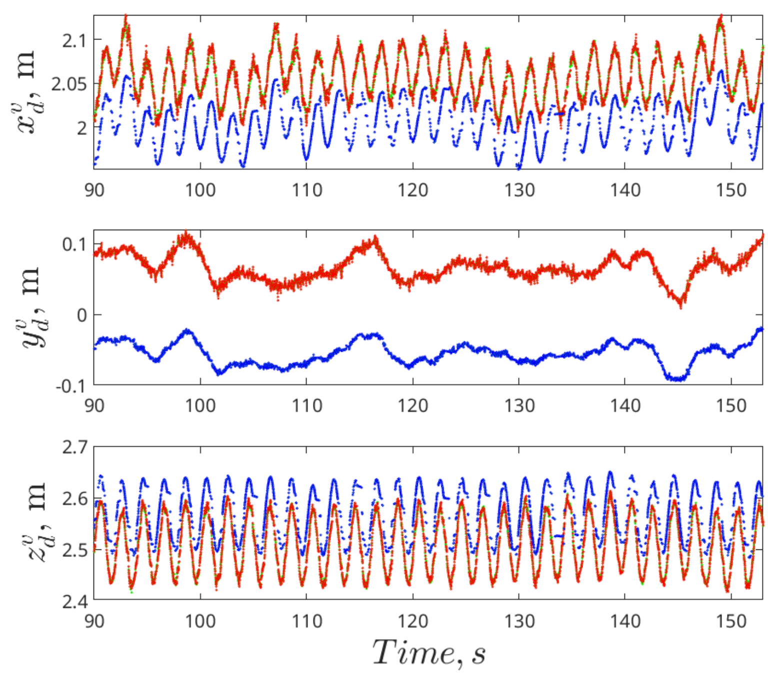


(b)

Figure C.2: Regular periodic waves with 0.66 Hz, 0.06 m (a) Position (b) Orientation

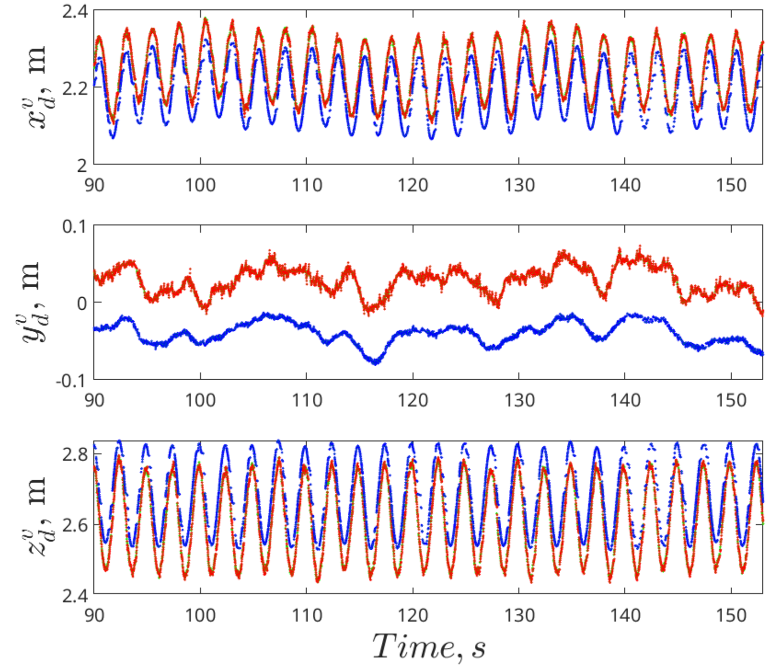


(a)

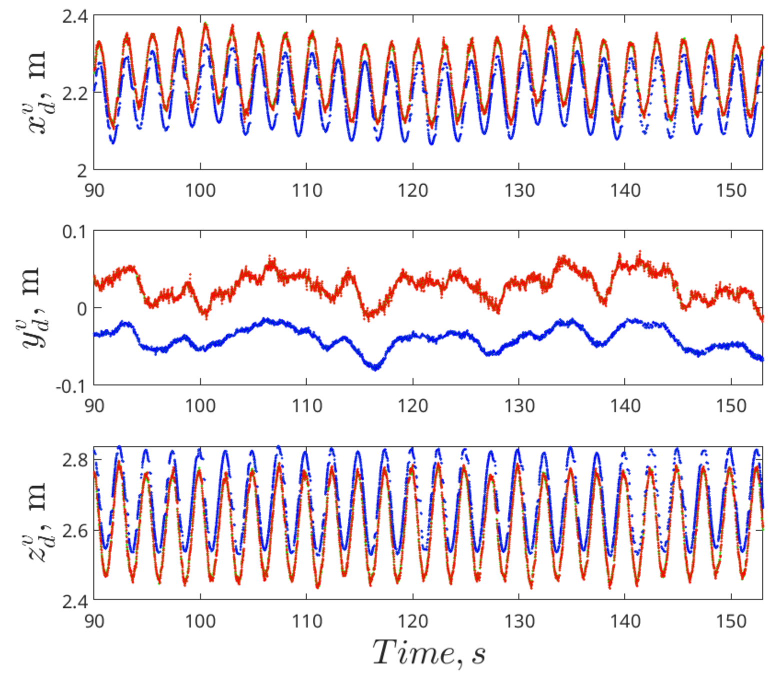


(b)

Figure C.3: Regular periodic waves with 0.5 Hz 0.12 m (a) Position (b) Orientation

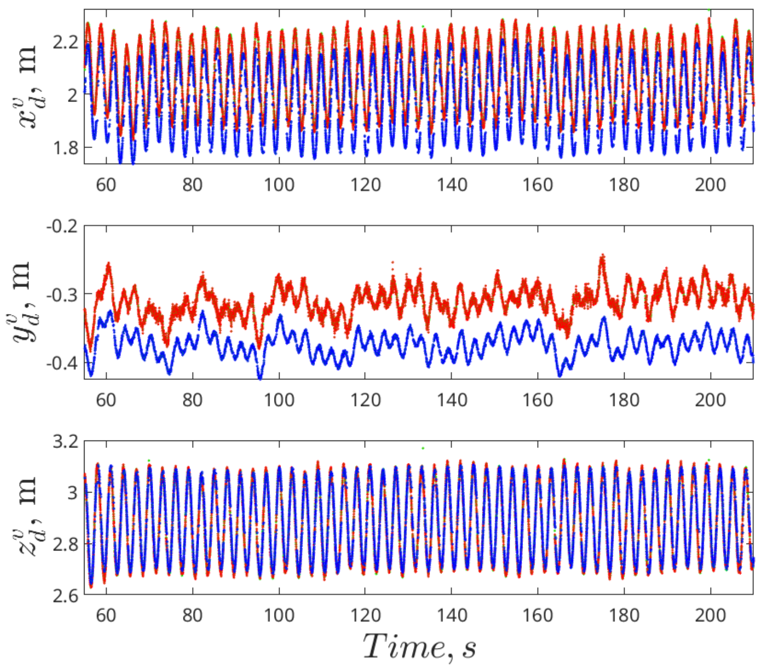


(a)

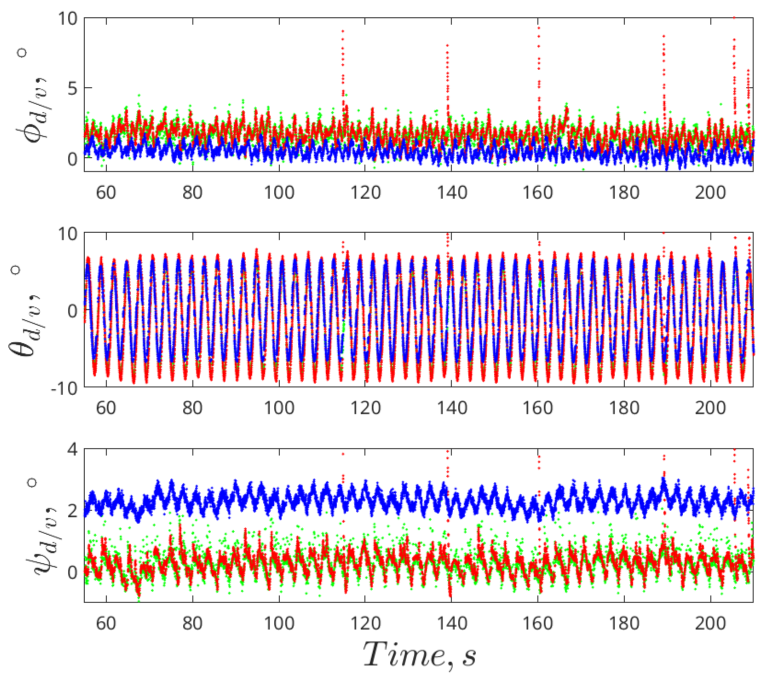


(b)

Figure C.4: Regular periodic waves with 0.4 Hz frequency, 0.18 m amplitude (a) Position (b) Orientation

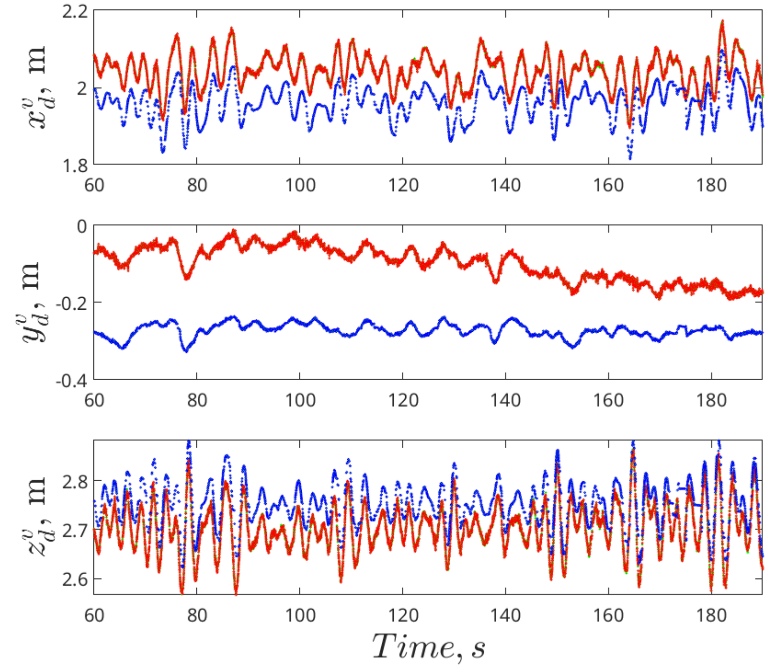


(a)

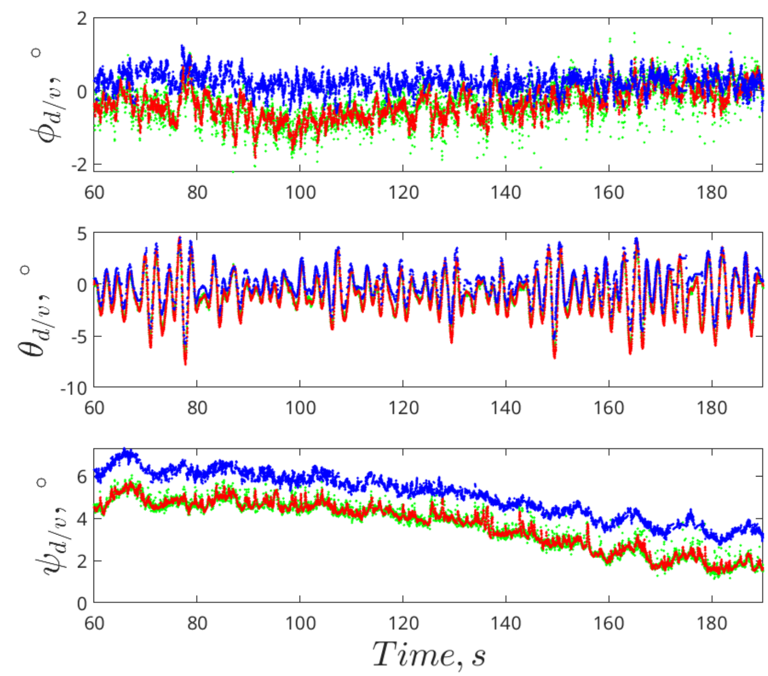


(b)

Figure C.5: Regular periodic waves with 0.337 Hz frequency, 0.24 m amplitude (a) Position (b) Orientation

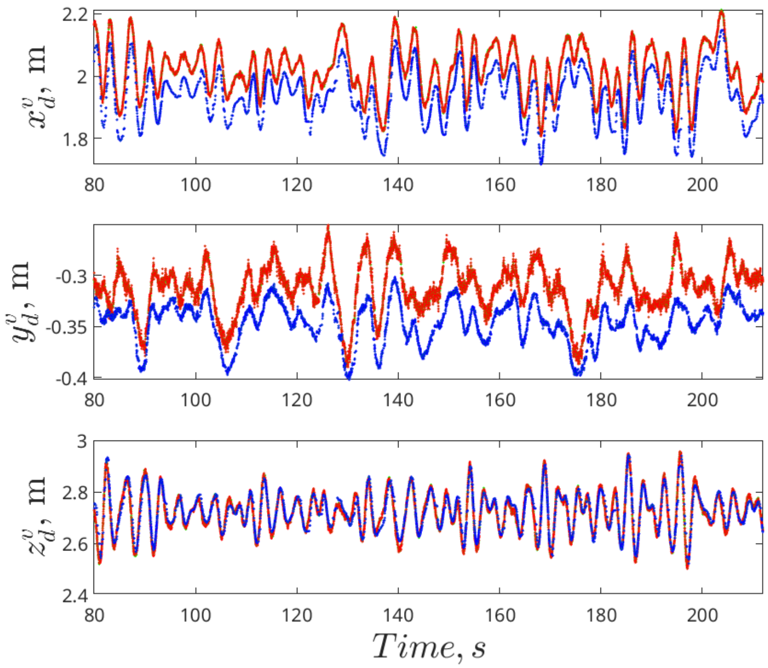


(a)

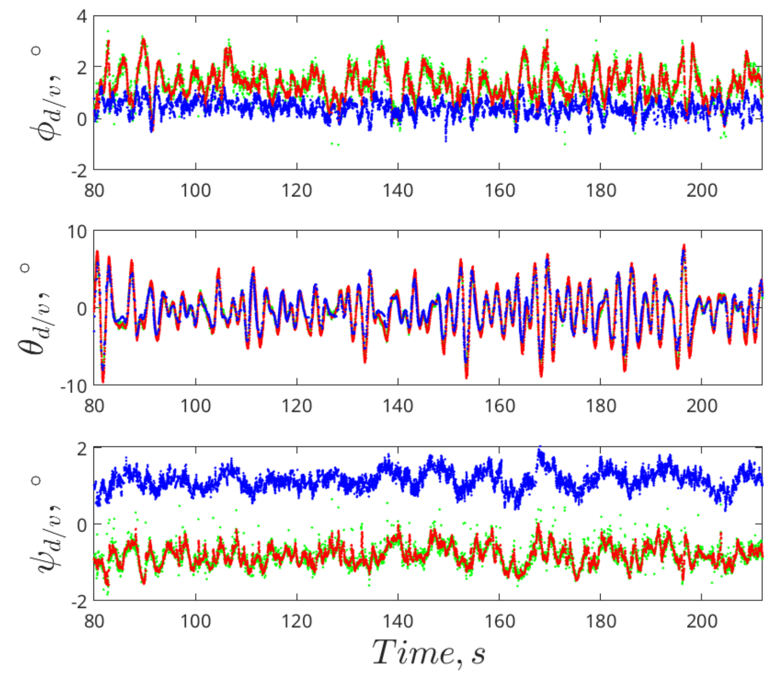


(b)

Figure C.6: Irregular waves dominant mode of 3.17s period, 0.22 m amplitude (a) Position (b) Orientation



(a)



(b)

Figure C.7: Irregular waves with dominant mode of 3.56 s period, 0.33 m amplitude (a) Position (b) Orientation