Office of Naval Research (ONR)

Final Report

December 1, 2019 to May 31, 2023

Grant: N00014-20-1-2092

Experimental Analysis of Advanced Control and Estimation Systems for Autonomous Ship Landing

Distribution Statement

DISTRIBUTION A. Approved for public release: distribution unlimited.

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.

Grant or Contract Number: N00014-20-1-2092

Date Prepared: May 31, 2023
Project Title: Experimental Analysis of Advanced Control and Estimation Systems for Autonomous Ship Landing

Principle Investigators:
Joseph F. Horn
Professor, Aerospace Engineering
Penn State University
814 865 6434, joehorn@psu.edu

Jack W. Langelaan
Professor, Aerospace Engineering
Penn State University
814 863 6082, jwl16@psu.edu

1. Overview of Project

Penn State has conducted scale experiments of a complete auto-landing system in collaboration with the Naval Surface Warfare Center Carderock Division (NSWCCD). These experiments have taken advantage of low-cost commodities, electronics, and multi-rotor aircraft to demonstrate automatic landing capabilities via landings on ship models at the Carderock wave tank facilities. These experiments have dealt with the development and implementation of trajectory generation methods, scalable flight control laws, deck motion prediction algorithms, vision-based sensing, and state estimation.

This project has provided three main contributions:

1. *The first rigorous model-scale experimental evaluation of autonomous landing guidance algorithms where reduction in scale was accounted for.* These tests required the development of scalable flight control laws, allowing aircraft closed loop dynamics to be related across test scales. Further, two guidance algorithms were developed as part of this effort: a baseline "deck tracking" method that follows deck motions while closing the gap between the deck and aircraft at a constant rate, and an advanced method that utilizes quadratic programming optimization to plan a landing path directly to the predicted deck state at touchdown. The results have provided insight into the feasibility of using typical deck motion prediction methods directly in landing path optimization, as well as the potential benefits of taking this approach.

2. *The demonstration of autonomous landings in scaled wave conditions with vision-based sensing and estimation methods used directly in the control loop.* The estimation method utilized measurements of fiducial markers ("AprilTags") that are specifically designed for enabling relative pose estimation, and the vision and IMU measurements were fused via an unscented Kalman filter.

3. *The establishment of a hardware and software configuration for performing autonomous flight tests in the Maneuvering and Seakeeping Basin (MASK).* This test setup has been thoroughly tested in over 200 autonomous flight tests and can be utilized for future autonomous landing and flight control research in the MASK facility.

2 Activities and Accomplishments

2.1 Hardware Design

Two hardware platforms were built: a coaxial hexacopter and a quadcopter (see figures 1 and 2). The vehicles were designed to perform ship landing flight tests in the MASK facility at the Naval Surface Warfare Center. The driving requirement for both vehicles was water resistance. Sensitive electronics are housed in waterproof enclosures, leaving only the motors exposed. Wiring and hardware interfaces were sealed using rubber gasket material and adhesives. The electronic speed controllers (ESCs) were fitted to aluminum heatsinks to exchange heat outside the enclosures. In the case of the hexacopter, an upgraded fan was used to prevent thermal throttling of the onboard computer. Initial flight tests showed both vehicles could perform back-to-back flights without overheating.

The enclosures provided additional challenges for testing visual navigation methods. To mitigate these, an optically clear acrylic dome was used to house the camera in the hexacopter. 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.

Both vehicles are equipped with a Pixhawk flight controller running the PX4 flight control firmware and an Odroid XU4 onboard computer. The Odroid uses the robotic operating system (ROS) to communicate with the Pixhawk and onboard camera, as well as the ground station computer. This allows for state estimation and control to be performed either completely onboard the vehicle or with the assistance of a ground station computer and/or an external motion capture system if desired.

Figure 1: Hexacopter UAV

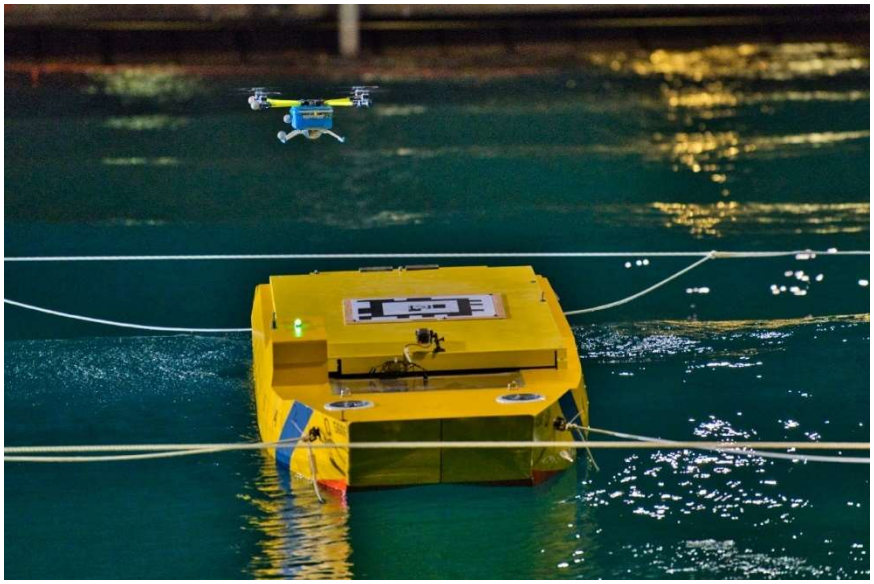


Figure 2: Quadcopter UAV

2.2 Vision System

A recursive nested AprilTag3 fiducial marker array from family tagCustom48h12 was used in vision-based sensing described here. The array is shown in figure 3. The array was printed on a 24 in square piece of Tyvek laminated on 1/8 in plywood and was affixed to the desired testing location.

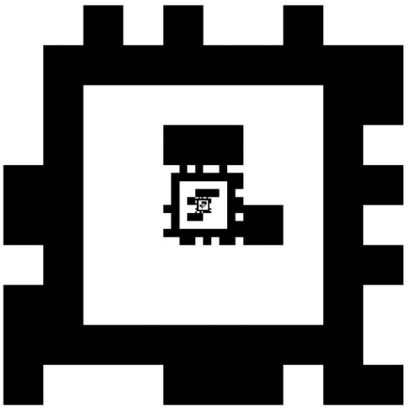


Figure 3: Recursive AprilTag marker array composed of four markers with IDs 0-4 from tag family tagCustom48h12.

The on-board Jevois smart camera was programmed to detect and estimate the position and orientation of the marker array based on a 640x480px image. The maximum advertised frame rate for the Onsemi AR0135 global shutter was 54Hz. Fiducials were detected, and array pose is estimated and streamed over USB to ROS running on the Odroid computer at 48 Hz.

Logic was imposed to only estimate the pose of the largest visible marker of all detected markers in the image. This reduced the computational requirements at some cost to accuracy and reliability of marker pose estimation, however it could be desirable to estimate and spherically interpolate all detected marker quaternions in addition to simply averaging poses.

2.3 State Estimation

An Unscented Kalman Filter (UKF) was used to estimate the state vector, **x,** and the diagonal of the estimated state covariance matrix, **P**. The estimation of state was performed in a recursive fashion and allowed for continued operation in the case where slight gaps occurred in the vision-based measurements (detections of the AprilTag). The estimated state vector consisted of the position of the deck in the helicopter frame in meters, 321 Euler angles from the helicopter to the deck in radians, the velocity of the deck in the helicopter frame in m/s, and the deck angular velocities in rad/s/s.

The measurement step utilized the relative position and orientation (pose) estimates of the fiducial array with respect to the camera frame streamed at 48Hz from the Jevois camera, transformed to the x-front, y-right, z-down vehicle body frame. This transformation allowed the use of a simple linear measurement model as the measurements are the first six estimated states:

$$z_{cam} = \left[r_d^{v\prime} \phi_d^v \; \theta_d^v \; \psi_d^v\right]' \tag{1}$$

The prediction step ran consistently at 78Hz and was calculated on a single thread from the incoming IMU data at 95Hz.

$$z_{IMU} = [\ddot{r}_v^h \; p \; q \; r \; \phi_v^h \; \theta_v^h \; \psi_v^h]' \tag{2}$$

The prediction step propagates the state estimates of the nonlinear dynamic model *F* with assumed known process noise, *n*. The measurement step corrects the estimate using the measurement model *G* with assumed known measurement noise.

$$\hat{x}_{k+1=F(\hat{x}_k,n)} \tag{3}$$

$$\hat{x}_{k+1=G(\hat{x}_k,v)} \tag{4}$$

The process model *F* was implemented in a fashion similar to that discussed in [1].

2.4 Control Design

2.4.1 Scaling Methodology

The UAV controller parameters used during ship landing experiments were determined by Froude scaling the closed loop dynamics of a typical full-scale manned rotorcraft similar in size to a UH-60. Therefore, prior to discussing the controller design, the proposed scaling methodology will first be discussed.

The closed loop aircraft dynamics were determined via Froude scaling. With this method, model scale distance is calculated by dividing full scale distance by the Froude number N_F. Model scale time is calculated by dividing full scale time by $\sqrt{N_F}$ and frequency follows the inverse of this.

Following these rules, we can derive the dynamic scaling laws shown in table 1 (note that we multiply by these scale factors to go from full to model scale).

Table 1: Froude Scaling Factors

Dimension	Scale Factor
Time	$1/\sqrt{N_F}$
Frequency	$\sqrt{N_F}$
Position	$1/N_F$
Velocity	$1/\sqrt{N_F}$
Acceleration	1
Jerk	$\sqrt{N_F}$
Angles	1
Angular Rates	$\sqrt{N_F}$
Weight	$1/N_F^3$
Inertia	$1/N_F^5$

The ratio of full to model scale hub-to-hub distance is usually taken as the Froude number when applying Froude scaling to multi-rotor aircraft control characteristics. Here, however, we propose using vehicle mass ratio to establish the scale factor. That is, our Froude number is calculated from

$$N_F = \left(\frac{M_{fs}}{M_{mfs}}\right)^{\frac{1}{3}} \tag{5}$$

where M_{fs} and M_{ms} represent full and model scale vehicle masses, respectively. The reason for using mass ratio is that, for tracking a trajectory, the UAV can be modelled as a point mass with a thrust vector. The bandwidth with which the thrust vector can rotate is prescribed in our attitude controller, and the bandwidth with which the thrust magnitude can vary is prescribed in our Z position controller. Additionally, the relative thrust to weight ratio can be particularly important when performing ship landings, as ship motions are often aggressive in heave. The downside to using mass ratio is that it is undesirable to have handling qualities requirements scale with vehicle mass since mass depends on payload, fuel, and other factors. Here, though, our goal is not to establish strict handling qualities requirements. Furthermore, the scaling factor is found from a cube root of mass ratio. Changes in the mass of either aircraft therefore do not have a large impact on the scaling factor unless they are very large compared to the nominal aircraft weight. Moving forward with our mass scaling law, the full-scale aircraft case that is considered here is a medium weight helicopter roughly the size of a UH-60. The UAV used in experimentation weighs 6.6 pounds, so taking the nominal full scale aircraft weight to be approximately 17,500 pounds we have:

$$N_F = \left(\frac{M_{fs}}{M_{mfs}}\right)^{\frac{1}{3}} = \left(\frac{17{,}500\ lbs}{6.6\ lbs}\right)^{\frac{1}{3}} \approx 13.8 \tag{6}$$

This scale factor of $N_F = 13.8$ was used for all scaling discussed in this report.

2.4.2 Explicit Model Following Controllers

The Explicit Model Following (EMF) control architecture is used to produce position-command position-hold and heading-command heading-hold controllers used by the autonomous control modes. A general EMF controller for a SISO plant is shown in figure 4. This architecture was chosen because, with an accurate model available for use in the feedforward inversion path, the closed loop transfer function will be approximately equivalent to the ideal response with the added time delay:

$$G_{CL}(s) \approx G_{ideal}(s)e^{-\tau s} \tag{7}$$

The goal of tailoring vehicle tracking bandwidths is therefore easily achieved through the specified ideal response model. Furthermore, both UAVs were shown to be well modeled through the use of simple decoupled transfer functions, which lend themselves nicely to the EMF control strategy. The transfer function models used in control design were obtained via the frequency domain system identification (ID) process presented in [2].

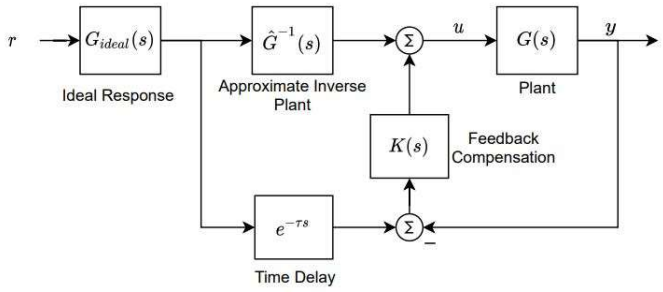


Figure 4: EMF controller for a general SISO plant.

A block diagram depicting the position control modes for the lateral and longitudinal axes is shown in figure 5. The architecture consists of inner roll and pitch attitude-command-attitude-hold controllers and an outer position control loop. The inner attitude loop inversion models are based off system ID results and the feedback compensator is a simple PI controller. The ideal attitude model is given as a generic second order system. For example, for roll, we use

$$G_{\phi,ideal}(s) = \frac{\omega_{\phi,ideal}^2}{s^2 + 2\zeta_{\phi,ideal}\omega_{\phi,ideal}s + \omega_{\phi,ideal}^2} \tag{8}$$

For designing the outer loop controllers, we consider roll and pitch attitudes as the driving inputs and neglect the higher bandwidth attitude dynamics. The resulting approximate models for X and Y heading frame accelerations are then given as

$$\ddot{X}^{hf} = -g\theta \quad \ddot{Y}^{hf} = g\phi \tag{9}$$

This results in the typical acceleration feedforward term with a gain of $1/g$, as seen in figure 5. The outer loop ideal models are also generic second order systems, and PID control is used for position feedback compensation. Note that the position command filter inputs are given in the inertial frame. The X and Y velocity and position tracking errors, as well as the integral of position tracking error, are also calculated in the inertial frame. The tracking errors are then transformed to the aircraft heading frame prior to multiplying by the feedback gains. The same process is applied

in the acceleration feedforward path. These transformations are necessary to produce appropriate pitch and roll commands, as our control gains are designed based off the heading frame model from equation 9.

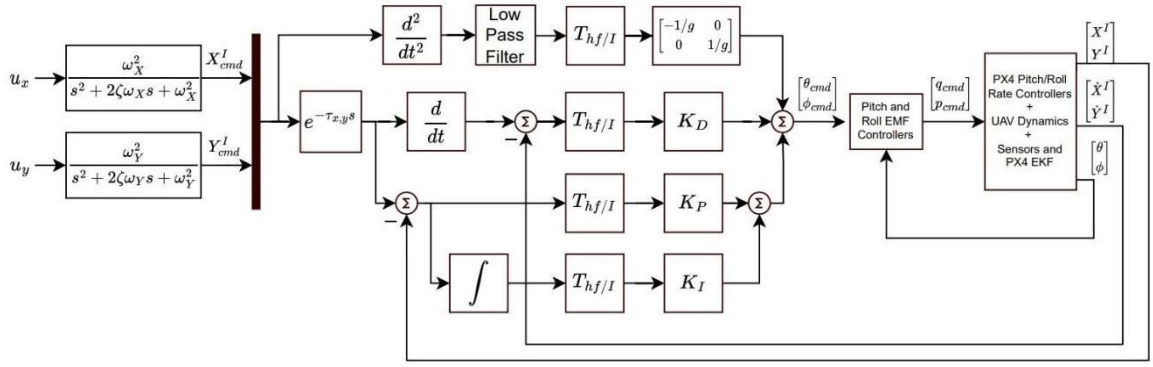


Figure 5: Lateral and longitudinal position control.

The yaw axis controller uses the same structure as the roll and pitch attitude control loops, with a second order command model and PI feedback compensation. The vertical axis control is designed similarly to the lateral and longitudinal control loops, but with the absence of an inner loop as the transfer function from throttle input to vertical position command output is used directly in the feedforward inversion model. There is also no need for frame transformations in the vertical position controller.

Note that this control architecture is being used in model scale experiments, where a portion of the intended tests includes varied reference tracking and disturbance rejection bandwidths. In order to have more meaningful carryover from model to full scale, the Froude scaling laws discussed in section 2.4.1 is used to scale the control laws. To exemplify how this scaling is performed, we will consider the longitudinal control axis. Say we have a full-scale model with a pitch tracking bandwidth denoted by $\omega_{\theta,fs}$. Following the Froude scaling framework, we simply scale this by $\sqrt{N_F}$ to arrive at the model scale pitch command filter frequency:

$$\omega_{\theta,ms} = \omega_{\theta,fs}\sqrt{N_F} \tag{10}$$

The outer loop position control bandwidth is then determined by dividing $\omega_{\theta,cf}$ by 5:

$$\omega_{X,cf} = \frac{\omega_{\theta,cf}}{5} \tag{11}$$

This factor of 5 is a good rule of thumb for ensuring adequate frequency separation between the inner and outer control loops.

The time delay included on position and velocity commands, denoted by $\tau_{x,y}$ in figure 5, is also calculated based off the attitude command filter. For position control, the attitude dynamics are neglected in the inversion, so we capture the phase of the attitude dynamics with this delay. Recall that the attitude tracking dynamics are forced to approximate the attitude command filters plus some small time delay. For pitch, we write

$$\frac{\theta_{UAV}}{\theta_{cmd}} = \frac{\omega_{\theta,cf}^2}{s^2 + 2\zeta\omega_{\theta,cf}s + \omega_{\theta,cf}^2} e^{-\tau_\theta s} \tag{12}$$

To approximate the phase of the attitude dynamics in equation 12, we calculate $\tau_{x,y}$ as follows:

$$\tau_{x,y} = \frac{1.65}{\omega_{\theta,cf}} + \tau_\theta \tag{13}$$

Note that this is valid with the attitude command filter damping ratio set to 0.8, which was done for all command filters. By scaling $\tau_{x,y}$ and $\omega_{X,cf}$ based off $\omega_{\theta,cf}$, the position and attitude tracking responses are consistently scaled based off just the full-scale attitude tracking bandwidth $\omega_{\theta,fs}$. Note that the above process is used for scaling the X and Y controllers, but for Z and yaw control we just need to scale command filter frequencies by $\sqrt{N_F}$ as was done for pitch in equation 10.

The above process illustrates how reference tracking bandwidths were scaled for model-scale testing. The disturbance rejection bandwidths were Froude scaled as well through systematic choice of feedback gains. The tests here focus on variations in reference tracking bandwidths, however, and aerodynamic disturbances were not included in the experimental setup. The details of disturbance rejection scaling are therefore not discussed here, but details on the disturbance rejection scaling method can be found in [3]. Note, though, that the scaled disturbance rejection bandwidths were set to meet Froude scaled level 1 handling qualities per ADS-33E, and therefore the UAV had good disturbance rejection capabilities for all tests discussed.

2.5 Trajectory Generation

Two separate trajectory generation and control algorithms have been developed for autonomous shipboard landings, both of which send position and heading commands to the same EMF control laws. The first guidance algorithm is the "baseline" algorithm that performs purely deck relative navigation, tracking deck motions while closing the gap between the UAV and deck at a constant rate. The second is a quadratic programming (QP) based control algorithm that incorporates deck motion predictions. The difference between these two strategies is illustrated in figure 6. If an aircraft has high control bandwidth and high maneuverability, a deck tracking approach is feasible, although it results in more maneuvering. In theory, if a deck motion forecasting strategy is used,

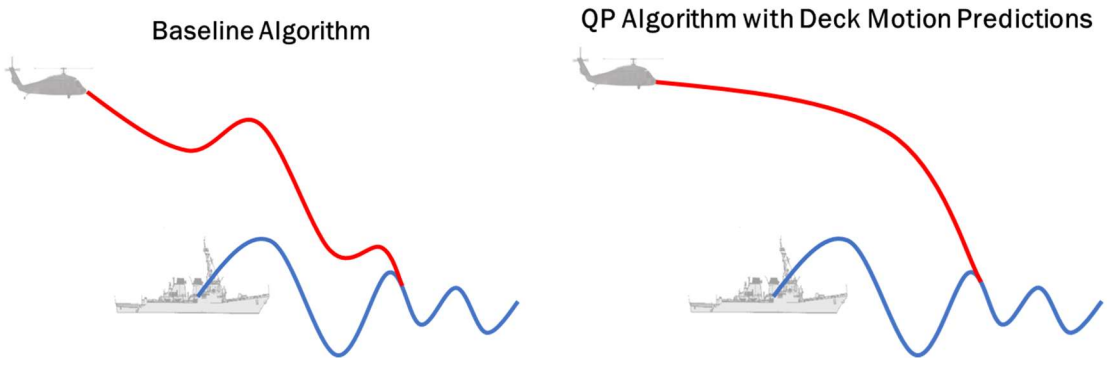


Figure 6: Illustration of baseline "deck tracking" algorithm versus QP optimization based algorithm that plans the landing path to a predicted deck state.

less maneuvering is required, and feasible trajectories can be found with lower bandwidth and more restrictive constraints on acceleration and jerk. This hypothesis was investigated via experiments in the MASK. Both autonomy algorithms will be described in the following sections, preceded by a description of the hardware integration necessary for performing autonomous navigation.

2.5.1 Hardware Integration

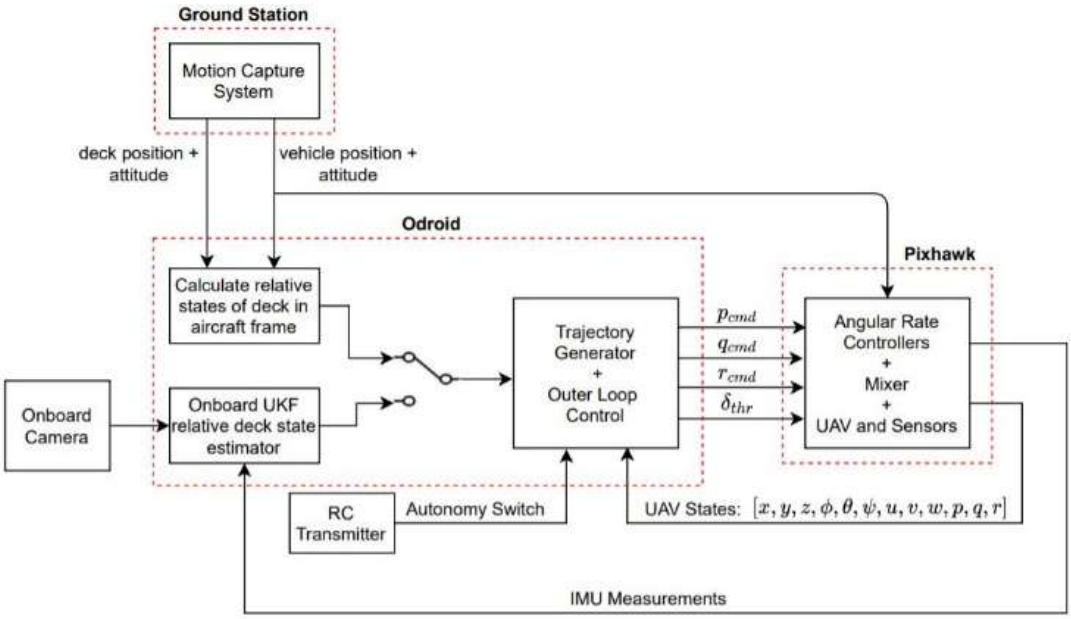


Figure 7: Implementation of trjectory generation and control algorithms on test hardware.

To perform autonomous navigation, it is necessary to have an established architecture for communication between all onboard processing units and the ground station computer. A diagram depicting the test hardware configuration used for our baseline control algorithm is shown in figure 7. Motion capture data is streamed over Wi-Fi to both the Pixhawk and the Odroid on board computer at 100 Hz. On the Pixhawk, the motion capture measurements of UAV pose are fused with IMU measurements by the PX4 extended Kalman filter. This allows PX4 to produce accurate position, velocity, and heading estimates. The PX4 state estimates are then sent to the Odroid to be used in path planning and control. The motion capture measurements of deck pose can be used directly in the path planning algorithms or can be substituted for using the onboard UKF. The onboard UKF estimates deck states relative to the UAV by fusing measurements sent to the Odroid from the onboard camera and Pixhawk IMU. With estimates of relative deck state and estimates of the inertial UAV pose produced by PX4, an estimate of the deck pose in the inertial frame can be calculated. For choosing to use either motion capture deck state information or deck state information from the onboard UKF, a flag can be set in the autonomy algorithm.

To trigger the autonomous flight modes a switch on the pilot's transmitter is used. Once this switch is selected, the trajectory generator begins sending commands to the UAV outer control loops. These outer loops are also run on the Odroid and will be discussed in detail in the control design section. The outer control loops produce angular rate and throttle commands, which are sent to the Pixhawk over a serial link at 100 Hz.

It should be noted that the architecture shown in figure 7 is for the baseline control mode. For the QP control mode with deck motion predictions, the setup is identical except that the QP based control mode does not currently incorporate the onboard UKF. This is due to the higher computational load of the QP based control algorithm, preventing the integration of the QP landing algorithm and vision-based state estimation on the current hardware. With recent advances in small form factor computers, however, an updated on-board processing unit would likely be capable of running both the QP trajectory generation and vision-based state estimation in real-time.

2.5.2 Baseline Trajectory

A graphical depiction of the baseline path planning algorithm is shown in figure 8. Referring to figure 8, the baseline control mode starts by navigating the UAV to some initial waypoint for starting the approach to the deck. The approach point navigation phase then begins, where the UAV is guided to the so-called approach point, which is a point specified relative to low pass filtered deck position and the deck heading. The low pass filters serve as an approximation of the mean deck motion, so that the approach navigation phase does not introduce unnecessary tracking of ship oscillations. Once the approach point is reached, a final descent to the deck is initiated. Note that for both the approach point navigation and final landing stages, the generated trajectories are dictated by the 45-degree angle between the camera lens and the UAV body frame since the deck needs to remain in view of the onboard camera. The waypoint, approach, and landing navigation phases are described in detail in the following subsections.

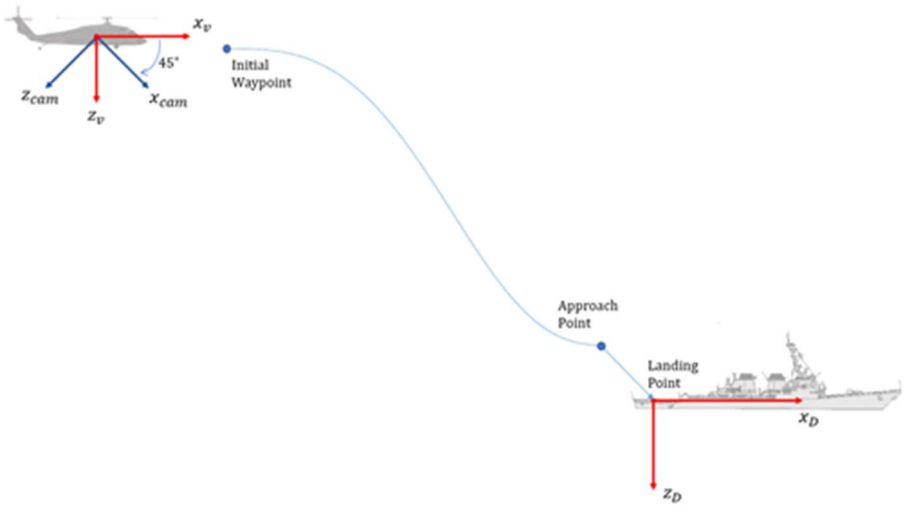


Figure 8: Landing sequence used during initial autonomous shipboard landings.

2.5.2.1 Baseline Waypoint Navigation

To navigate to the initial waypoint, the baseline control algorithm utilizes the high order tau guidance algorithm presented in [1] to generate position and heading commands. The tau guidance algorithm proposes a differential equation in terms of the distance to some goal and the velocity toward some goal, defined by x and $\dot{x}$ respectively. The value x is referred to as the action gap and does not need to represent a spatial distance but can represent the distance between the current state and desired state for any degree of freedom. The higher order tau guidance algorithm from [1] proposes the following differential equation for trajectory generation:

$$\frac{x}{\dot{x}} = \frac{k}{3}\left(t - \frac{T_g^3}{t^2}\right) \tag{14}$$

Here T_g represents the time until contact with the goal (i.e., $x(T_g) = 0$), t represents the time elapsed since the start of the trajectory, and k is a parameter that can be used to tune the trajectory. Solving equation (14) for x(t) results in

$$x(t) = \frac{x_0}{T_g^{3/k}}\left(T_g^3 - t^3\right)^{1/k} \tag{15}$$

where x_0 represents the initial action gap at the start of the trajectory. For an overview of the motivation behind equation (14) and the effects of k on the generated trajectories see [1].

For the approach navigation, four trajectories are produced: X, Y, and Z position trajectories and a heading trajectory. The distance between the actual and desired position and heading at the start of the trajectory are therefore taken as the initial gaps. For example, for the X position degree of freedom

$$x_{0,X}^I = X_{wp}^I - X_{0,uav}^I \tag{16}$$

where $x_{0,X}$ is the initial action gap for the X position trajectory, X_{wp} is the waypoint X position, and $X_{0,uav}$ is the X position of the UAV at the start of the waypoint navigation phase. The superscript I denotes that these values are all expressed in the inertial frame.

Note that equation (14) is designed to regulate the action gap to zero. The output of equation (15) therefore needs to be converted from values of x(t) for each degree of freedom to inertial position commands. This is done through the relation

$$\begin{bmatrix} X_{cmd}^I \\ Y_{cmd}^I \\ Z_{cmd}^I \end{bmatrix} = \begin{bmatrix} X_{wp}^I \\ Y_{wp}^I \\ Z_{wp}^I \end{bmatrix} - \begin{bmatrix} x_X^I \\ x_Y^I \\ x_Z^I \end{bmatrix} \tag{17}$$

which is formulated so that as the value of the action gap goes to zero, the commanded position approaches the waypoint. Here x_X^I, x_Y^I, and x_Z^I represent the action gap produced by equation (15) for the X, Y, and Z position trajectories respectively. For the heading command, the relation is analogous to the position command:

$$\psi_{cmd} = \psi_{wp} - x_\psi \tag{18}$$

In order to evaluate equation (15) the parameters k and T_g must be defined. For all four degrees of freedom the value of $k = 0.25$ is used, as this produces trajectories that end with zero velocity and zero acceleration (i.e., $\dot{x} = \ddot{x} = 0.$). Choosing a value of T_g is not quite as straight forward, as it is necessary to set values of T_g that guarantee commands will not be overly aggressive. Here this is ensured by solving for T_g for each of the four degrees of freedom based off a desired maximum magnitude of commanded acceleration over the course of the trajectory. Following this approach, the resulting expression for the time to contact is given in equation (19).

$$T_g = 2.888\sqrt{\frac{|x_0|}{\ddot{x}_{max}}} \tag{19}$$

Once equation (19) has been evaluated for all four degrees of freedom, the results are compared and the highest value is used as the time to contact for all trajectories. While the value of T_g does not need to be the same for all four degrees of freedom, this ensures that each degree of freedom is only commanded as aggressively as it needs to be to reach the initial waypoint without slowing convergence. During testing with this approach, it was found that values of $\ddot{x}_{max} = 0.5\ m/s^2$ for the position trajectories and $\ddot{x}_{max} = 50\ deg/s^2$ for the heading trajectory resulted in a smooth transition to the initial waypoint.

2.5.2.2 Baseline Approach Navigation

In the baseline algorithm's approach navigation phase, the UAV is commanded from the initial waypoint to a point that is near the deck, but that keeps the landing gear safely outside of the wave amplitude. For the tests discussed here this point was chosen to be about 0.66 meters behind and 0.66 meters above a low pass filtered deck position. The low pass filter serves as an approximation of the average deck motion. An example of this for the deck's inertial X position is shown in figure 9. For the first flight tests in the MASK, the filter parameters were set to $\omega_{lpf} = 2$ rad/s and $\zeta_{lpf} = 1$. The filter frequency was later reduced to $\omega_{lpf} = 0.5\ rad/s$. The heading was commanded to match that of the deck.

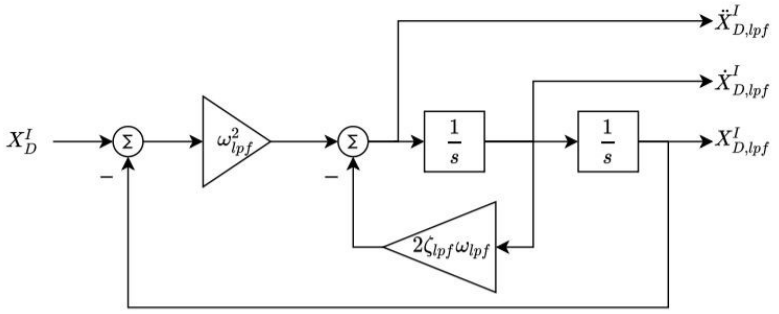


Figure 9: Deck position low pass filter.

In this phase it is important that the UAV position commands converge to an endpoint relative to the filtered deck position. This point will be referred to as the approach point, which is a constant point relative to the low pass filtered ship in the ship heading frame but is not a constant point in the inertial frame. The approach point is specified in the deck heading frame so that as the deck yaws the UAV will adjust to remain directly behind the ship. The deck roll and pitch angles, however, do not have any effect on the UAV position command. Similar to the initial waypoint navigation phase, commands are produced with the use of the higher order tau guidance algorithm. The main difference here is that we no longer wish to approach a constant point. Additionally, in this phase a tau-based trajectory is only needed for the three position degrees of freedom, as the heading is just commanded to match that of the deck. In order to generate trajectories that converge to a moving point, several updates need to be made to equation (16) from the waypoint navigation section. The initial gaps are now taken as the difference between the UAV position and the desired

position relative to the low pass filtered deck (referred to as the approach point) at the start of the approach phase. This is expressed as:

$$\begin{bmatrix} x_{0,X}^I \\ x_{0,y}^I \\ x_{0,z}^I \end{bmatrix} = \begin{bmatrix} X_{D,lpf}^I \\ Y_{D,lpf}^I \\ Z_{D,lpf}^I \end{bmatrix} + T_{I/D,hf} \begin{bmatrix} X_{appr}^{D,hf} \\ Y_{appr}^{D,hf} \\ Z_{appr}^{D,hf} \end{bmatrix} - \begin{bmatrix} X_{0,uav}^I \\ Y_{0,uav}^I \\ Z_{0,uav}^I \end{bmatrix} \tag{20}$$

where $X_{D,lpf}^I, Y_{D,lpf}^I$, and $Z_{D,lpf}^I$ represent the low pass filtered inertial deck position, $X_{appr}^{D,hf}, Y_{appr}^{D,hf}$, and $Z_{appr}^{D,hf}$ represent the approach point in the deck heading frame, and $T_{I/D,hf}$ is the transformation matrix from the deck heading frame to the inertial frame. To produce trajectories that converge to the approach point, the position, velocity, and acceleration command equations need to be updated to include the filtered deck states. The equation for position commands is

$$\begin{bmatrix} X_{cmd}^I \\ Y_{cmd}^I \\ Z_{cmd}^I \end{bmatrix} = \begin{bmatrix} X_{D,lpf}^I \\ Y_{D,lpf}^I \\ Z_{D,lpf}^I \end{bmatrix} + T_{I/D,hf} \begin{bmatrix} X_{appr}^{D,hf} \\ Y_{appr}^{D,hf} \\ Z_{appr}^{D,hf} \end{bmatrix} - \begin{bmatrix} x_X^I \\ x_Y^I \\ x_Z^I \end{bmatrix} \tag{21}$$

Note that the action gaps used to calculate the commands are still computed via equation (15) from the initial waypoint navigation section. $k = 0.25$ was again used in these equations and T_g was again calculated from equation (19) with $\ddot{x}_{max} = 0.5\ m/s^2$ for all three position trajectories.

2.5.2.3 Landing Phase

For the baseline guidance algorithm, the goal of the landing phase is to have the UAV contact the deck at the deck origin while remaining on a roughly 45-degree line in the ship $X - Z$ plane (as was shown in figure 8). This approach path is designed to keep the deck tag (located at the deck origin) in view of the onboard camera.

To ensure the touchdown was not overly aggressive, the UAV was commanded to a constant vertical descent rate relative to the deck. This is calculated as

$$\dot{Z}_{cmd}^I = \dot{Z}_{uav/D} + \dot{Z}_D^I \tag{22}$$

where $\dot{Z}_{uav/D}$ is the desired relative descent rate between the UAV and the deck. Here the value $\dot{Z}_{uav/D} = 0.25\ m/s$ was used. The X and Y position commands were then calculated using equation (23).

$$\begin{bmatrix} X_{cmd}^I \\ Y_{cmd}^I \end{bmatrix} = \begin{bmatrix} X_{D,lpf}^I \\ Y_{D,lpf}^I \end{bmatrix} + \begin{bmatrix} \cos(\psi_D) & -\sin(\psi_D) \\ \sin(\psi_D) & \cos(\psi_D) \end{bmatrix} \begin{bmatrix} Z_{uav}^I - Z_D^I \\ 0 \end{bmatrix} \tag{23}$$

In equation (23) the inertial X and Y commands are updated to maintain an X distance relative to the deck that is equivalent to the Z distance relative the deck when the relative X distance is expressed in the deck heading frame. In other words, equation (23) commands the UAV to follow a 45-degree line relative to the deck in the ship $X - Z$ plane.

The final aspect of the landing sequence is throttling down the UAV. Here the UAV was commanded to throttle down and contact the deck once the UAV landing gear achieved a relative Z distance of 5 centimeters or less from the deck.

2.5.3 Quadratic Programming Based Trajectory with Deck Motion Prediction

A high-level view of the QP guidance algorithm is shown in figure 10. The algorithm solves quadratic programming problems in real time to produce X, Y, and Z position commands, with the trajectory planned to match a future deck state prediction derived from autoregressive (AR) time series models. The deck forecasts are updated continuously during the approach, and therefore become more accurate as the landing progresses. This approach has the potential to reduce the required maneuvering, allowing feasible trajectories to be found with lower bandwidth and more restrictive constraints on acceleration and jerk. Here the MATLAB function “mpcActiveSetSolver" was used to perform the QP optimizations. Note that no optimization solver was used to command UAV heading. Instead, the heading was commanded to track low-pass filtered deck heading, as the deck yaw motion was mild.

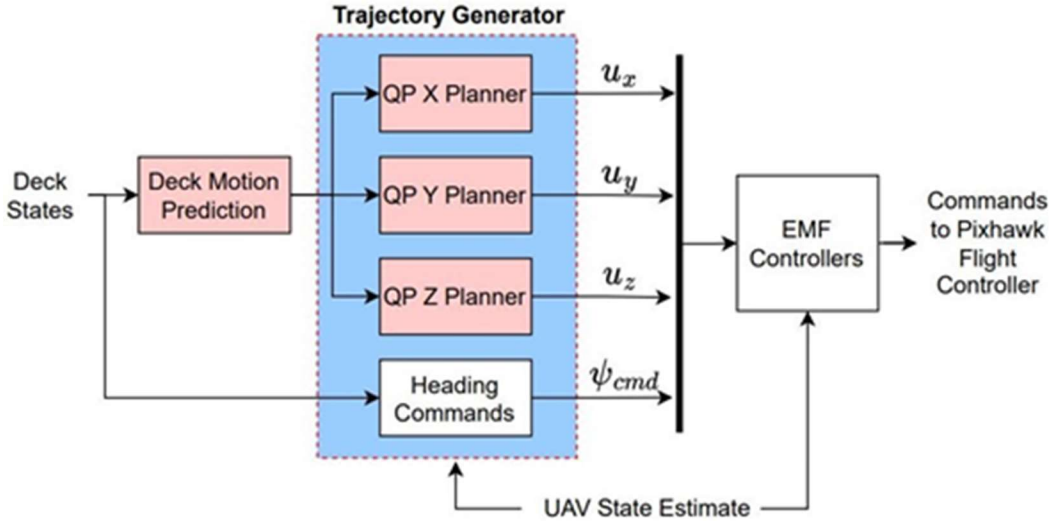


Figure 10: High-level view of QP guidance algorithm.

2.5.3.1 QP Trajectory Generation

The method used to formulate the QP trajectory optimization problem is essentially the same as that used in the so-called “direct shooting" approach to linear MPC. The main difference here is that the prediction horizon is variable in length, vanishing during the final stages of the landing sequence. In both cases, the first step is to define a discrete linear time-invariant (LTI) state space model. Here the inertial X, Y, and Z translational dynamics are approximated by the theoretical ideal response with EMF control. The models are then discretized assuming a zero-order hold and accounting for time delays. For example, for X trajectory generation, discretizing and converting to state space form gives

$$G_X(s) = \frac{\omega_{X,cf}^2}{s^2 + 2\zeta\omega_{X,cf}s + \omega_{X,cf}^2} e^{-\tau_X s} \quad \rightarrow \quad \begin{matrix} \vec{x}_{X,k+1} = A_X\vec{x}_{X,k} + B_X u_{X,k-\tau_d} \\ \vec{y}_{X,k} = C_X\vec{x}_{X,k} + D_X u_{X,k-\tau_d} \end{matrix} \tag{24}$$

where subscript k represents the current time step, τ_d represents the number of full sample period delays, and fractional portions of the continuous time delay are absorbed into the state space model

coefficients. Since the command filters are uncoupled, this approach allows separate QP optimization solvers to be run for each control axis. This boosts computational efficiency as the QP solvers can be run simultaneously on different CPU cores.

Since the position command filters are second order, the position, velocity, and acceleration can be included in the state space model output vector and can therefore be included in the QP optimization penalty and constraint functions. But, to account for the fact that an aircraft cannot accelerate instantaneously, the QP algorithm also penalizes and constrains jerk. Jerk is therefore approximated by a back difference on the acceleration output.

Once a discrete time model is obtained, the recursive nature of the state space equations is used to solve for the output and jerk at each point in the prediction horizon while also accounting for time delays in the discrete model. The resulting equations can then be combined with a quadratic cost function to transcribe a standard QP optimization problem of the form

$$\begin{aligned} J &= \frac{1}{2}\bar{U}^T H \bar{U} + F^T \bar{U} \\ s.t. \quad & A_c \bar{U} \leq b_0 \end{aligned} \tag{25}$$

where J represents the quadratic objective function, $\bar{U}$ represents the decision variables (in this case the controls), and H and F are constant matrices. In addition, the constant matrix A_C and constant vector b_0 define linear inequality constraints placed on the decision variables. The linear constraints are used to limit velocity, acceleration, and jerk during the landing, and also to constrain UAV altitude to be greater than the predicted deck altitude prior to the target land time. For more details on the QP transcription, refer to [3].

When transcribing the optimization problem, the cost function was written as

$$\begin{aligned} J = \sum_{k=0}^{N-1} \left[\left(\vec{y}_{ref,k} - \vec{y}_k\right)^T Q \left(\vec{y}_{ref,k} - \vec{y}_k\right) + u_k^T R u_k \right] + \sum_{k=0}^{N-1} \left[j_k^T Q_\Delta j_k \right] \\ + u_N^T R u_N + N \left[\left(\vec{y}_{ref,N} - \vec{y}_N\right)^T S \left(\vec{y}_{ref,N} - \vec{y}_N\right) + j_N^T S_\Delta j_N \right] \end{aligned} \tag{26}$$

where N represents the prediction horizon and Q, Q_Δ, S, S_Δ, and R are diagonal weighting matrices. This equation includes penalties on the deviation of position, velocity, and acceleration outputs from a reference trajectory $\vec{y}_{ref}$, as well as penalties on jerk and control inputs. For the output reference errors and jerk, separate weights are used for the terminal and running costs. This allows for emphasis to be placed on the final point of the trajectory, which is particularly important. Also note that the terminal cost associated with jerk and output reference error is weighted by the number of points in the prediction horizon N. This is done because the horizon is variable in length. When the horizon is at a maximum, the running cost is higher due to more points in the summation, so the terminal weighting factor is also increased to avoid its impact being diluted.

Since the prediction horizon itself is not included in the optimization, the time duration of the maneuver must be assigned prior to running the QP solver. Here, an initial land time was calculated with the equation

$$t_{land} = \max\left\{\begin{array}{l} 1.444\sqrt{\frac{\left|X^I_{UAV,0} - X^I_{d,0}\right|}{a_{max,X}}} \\ 1.444\sqrt{\frac{\left|Y^I_{UAV,0} - Y^I_{d,0}\right|}{a_{max,Y}}} \\ 1.444\sqrt{\frac{\left|Z^I_{UAV,0} - Z^I_{d,0}\right|}{a_{max,Z}}} \end{array}\right\} \tag{27}$$

where subscript 0 denotes the UAV and deck positions at the start of the landing and $a_{X,max}$, $a_{Y,max}$, and $a_{Z,max}$ are the QP planner acceleration limits. This equation was motivated by the tau guidance method used in the baseline algorithm and was found to produce reasonable land times that scale well with different initial distance from the deck.

The QP algorithm also has the option to update the target land time during the approach based off the predicted deck states. The update is allowed to occur when there is between 1.5 and 3 seconds left in the maneuver. The lower bound was set at 1.5 seconds to avoid shifting the land point without leaving enough time for the UAV to react. The 3 second upper limit was chosen because deck predictions are known to be inaccurate at longer prediction horizons. When in this window, the algorithm can shorten the remainder of the maneuver by up to 3 time steps (0.3 seconds in this case) or extend the land time out to 3 seconds. Each time point in the considered landing window is assigned a cost, and the point with the lowest cost is selected as the new land time. The cost for each time point is given by

$$J_K = w_Z\left(Z^I_{dp,k} - \bar{Z}^I_{dp}\right) - w_{\dot{Z}}\dot{Z}^I_{dp,k} + w_\phi\left|\phi_{dp,k}\right| + w_\theta\left|\theta_{dp,k}\right| + w_{\Delta t}\left|t_{land,k} - t_{land}\right| \tag{28}$$

where dp denotes a predicted deck state, and the weighting factors were set to $w_Z = 1$, $w_{\dot{Z}} = 0.5$, $W_{\Delta t} = 0.15$, and $w_\phi = w_\theta = 0.06 \cdot 180/pi$. The first two terms of equation (28) were designed to target a point where the deck heave is near a high point but beginning to descend (note that Z is positive down in equation (28)). This was done to discourage landing when the deck is heaving upward significantly, or near the bottom of an oscillation and about to heave upward, which is a harder state to match at landing. The second two terms of equation (28) were included to discourage landing at a time with an extreme deck attitude and the last term was included to penalize changes from the current land time, so that large changes in land time are not made due to small changes in the other terms. This simple scheme for adjusting the targeted land time during the approach was found to work well in simulations, but as will be discussed in the experimental results section, had a negative effect at times during the experimental landings.

Note that the QP guidance algorithm limits the horizon length to $N_{max} = 30$ (corresponding to 3 seconds with an update rate of $\Delta t_{QP} = 0.1$ seconds) to keep the optimization problem small enough for real-time execution. The calculated maneuver duration is generally greater than 3 seconds, however. To address this, when there are more than 3 seconds remaining in the landing sequence, the reference trajectory included in the cost function is a simple straight line toward the predicted deck state at landing. This brings the UAV closer to the deck until less than 3 seconds are remaining. Once the time remaining is under 3 seconds, the reference trajectory is changed so that the target positions and velocities match the predicted deck positions and velocities at landing. The target

final accelerations were set to motivate matching the predicted deck attitude at landing assuming the UAV heading frame accelerations can be approximated by

$$\ddot{X}^{hf} = -g\theta \quad \ddot{Y}^{hf} = g\phi \tag{29}$$

Additionally, the time horizon begins being computed as $N = \frac{time\ remaining}{\Delta t_{QP}}$, keeping the desired land time as the final point in the optimization.

2.5.3.1 Deck Motion Predictions

The deck motion predictions define the terminal deck state that we wish to match with the QP trajectory optimization. These predictions are produced by autoregressive (AR) time series models. The use of a time series method is appealing as it requires no model of the ship dynamics and computational cost can be kept reasonable. Additionally, past works have found this class of methods to converge to reasonable predictions with sufficient lead time for performing autonomous landings. Other time series methods have been applied to deck motion prediction in the literature, but the results appear comparable. The AR method was therefore chosen here due to its ease of implementation.

The AR model formulation assumes that the current output can be described by a linear combination of lagged outputs plus additive zero mean white noise. This is expressed as

$$\vec{y}_k = \alpha_1 \vec{y}_{k-1} + \alpha_2 \vec{y}_{k-2} \cdots + \alpha_{N_{lag}} \vec{y}_{k-N_{lag}} + \vec{v}_k \tag{30}$$

where output vectors $\vec{y} \in R_m$, matrices $\alpha \in R_{m \times m}$, noise vector $\vec{v} \in R_m$, and N_{lag} represents the number of lagged outputs. Here $N_{lag} = 15$ was used, as increases in accuracy leveled off when adding additional parameters. Transposing equation 30 and writing in matrix form, we have

$$\vec{y}_k^T = \begin{bmatrix} \vec{y}_{k-1}^T & \vec{y}_{k-2}^T & \cdots & \vec{y}_{k-N_{lag}}^T \end{bmatrix} \begin{bmatrix} \alpha_1^T \\ \alpha_2^T \\ \vdots \\ \alpha_{N_{lag}}^T \end{bmatrix} + \vec{v}_k \tag{31}$$

or, in a more compact form,

$$\vec{y}_k^T = \bar{Y}_{lag} \underline{\alpha} + \vec{v}_k \tag{32}$$

This is then separated into one equation for each output:

$$\begin{aligned} \vec{y}_1^T &= \bar{Y}_{lag} \underline{\alpha}_1 + \vec{v}_1 \\ \vec{y}_2^T &= \bar{Y}_{lag} \underline{\alpha}_2 + \vec{v}_2 \\ &\vdots \\ \vec{y}_m^T &= \bar{Y}_{lag} \underline{\alpha}_m + \vec{v}_m \end{aligned} \tag{33}$$

Here α_i represents the i^{th} column of α. Each column vector α_i is then estimated using a standard recursive least squares (LS) algorithm. While the problem could be formulated as one larger recursive LS problem, breaking the parameter update into m smaller problems was found to be more computationally efficient due to smaller matrix inversions in the recursive update. With the

estimate of α obtained, we propagate the AR model into the future to cover the desired prediction horizon:

$$\begin{aligned} \vec{y}_{k+1}^T &= \begin{bmatrix} \vec{y}_k^T & \vec{y}_{k-1}^T & \cdots & \vec{y}_{k-N_{lag}+1}^T \end{bmatrix} \underline{\alpha} \\ &\vdots \\ \vec{y}_{k+N}^T &= \begin{bmatrix} \vec{y}_{k-1+N}^T & \vec{y}_{k-2+N}^T & \cdots & \vec{y}_{k-N_{lag}+N}^T \end{bmatrix} \underline{\alpha} \end{aligned} \tag{34}$$

Note that for predictions we are interested in the expected future output, so the zero mean white noise term is dropped.

The above equations define the general process of estimating and propagating AR models for forecasting a time series. For forecasting deck states, two separate AR models are used: one for longitudinal and vertical states and one for lateral states. The output vectors used for each AR model are

$$\begin{aligned} \vec{y}_{long} &= \begin{bmatrix} X_d^{dhf} & \dot{X}_d^{dhf} & \theta_d & Z_d^I & \dot{Z}_d^{dhf} \end{bmatrix}^T \\ \vec{y}_{lat} &= \begin{bmatrix} Y_d^{dhf} & \dot{Y}_d^{dhf} & \phi_d & \psi_d \end{bmatrix}^T \end{aligned} \tag{35}$$

Deck vertical position and velocity are included with the longitudinal outputs as the correlation between ship pitch motion and ship deck heave was found to improve the AR predictions. Including all outputs in a single vector, however, was found to cause less reliable predictions at times.

In equation 35 X_d, Y_d, $\dot{X}_d$, and $\dot{Y}_d$ are expressed in the averaged deck heading frame. This frame is defined by the heading of the deck averaged over the lagged time history stored in the AR model. The transformation of X_d and Y_d from the inertial frame to the averaged deck heading frame is computed as

$$\begin{bmatrix} X_d^{dhf} \\ Y_d^{dhf} \end{bmatrix} = \begin{bmatrix} \cos(\bar{\psi}_d) & \sin(\bar{\psi}_d) \\ -\sin(\bar{\psi}_d) & \cos(\bar{\psi}_d) \end{bmatrix} \begin{bmatrix} X_d^I \\ Y_d^I \end{bmatrix} \tag{36}$$

where $\bar{\psi}_d$ is the averaged deck heading. $\dot{X}_d$ and $\dot{Y}_d$ are transformed in the same manner. Once the AR model predictions are produced in this frame, they are transformed back to the inertial frame for use by the trajectory generator.

2.6 Experimental Setup in the MASK

For the autonomous landing flight tests, the MASK facility OptiTrack motion capture was used to track the ship deck and UAV, with the motion capture software running on the ground control station (GCS). The unmanned ship vessel (USV) used was approximately 20 feet long and was loosely tethered in the wave pool. Tethering prevented the ship's heading and position from drifting significantly but allowed significant oscillations in position and attitude. A launch and recovery (L&R) zone was designated off the shore of the basin for UAV takeoff and landing between test runs. A picture of the test setup in the MASK facility is shown in figure 11 and a close-up of the model-scale ship can be seen in figures 1 and 2.

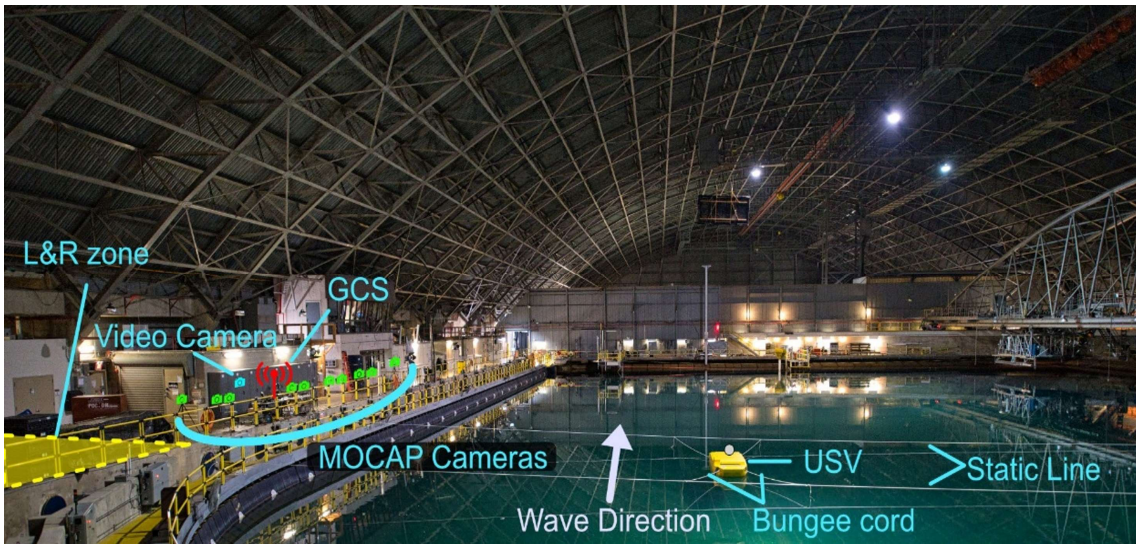


Figure 11: Test setup in the MASK.

2.7 Experimental Results

The first flight tests in the MASK were conducted over a two-week period in August and September of 2021. These two weeks were focused on establishing communications between the UAV and the MASK facility motion capture system, on validating the baseline trajectory generation method and EMF control laws, and ultimately on evaluating the vision-based state estimation system through vision-based landings.

Extensive flight testing was also performed over an eight-day test period in October 2022. During this test period the focus was on the evaluation of the autonomous landing trajectory generation methods. To achieve this, 149 autonomous landings were performed using both the baseline and QP optimization based trajectory generation algorithms. During testing the control laws were used to artificially degrade reference tracking bandwidths and constraints on acceleration and jerk, with the objective of studying the ability of both trajectory generation methods to cope with reduced maneuverability.

2.7.1 Evaluation of Vision-Based Deck State Estimator in the MASK

2.7.1.1 Validation of Deck State Estimator with Hovering Aircraft

The first week of testing in the MASK was used to enable operation of the Optitrack motion capture system and gather vision-based state estimates with a moving deck. For these tests, the vehicle was commanded to hover for one-to-two-minute periods over the deck with the vision system and state estimator running. Motion capture data and vision-based estimates were logged simultaneously, allowing for evaluation of the vision estimates using the motion capture measurements. These tests were performed in six separate wave conditions, which are reported in table 2. Note that regular waves refers to a pure sinusoidal wave form with frequency as shown in table 2. Irregular waves refer to stochastic wave forms with the peak in the wave spectrum occurring at the frequency shown in table 2. The wave angle of 0 degrees indicates that the waves approached from the front of the ship, directly at the bow.

Table 2: Wave conditions for initial deck state estimator evaluation.

Condition	Wave Amplitude [m]	Wave Frequency [Hz]	Wave Angle [deg]
No Waves	0	0	-
regular	0.06	0.500	0
regular	0.12	0.500	0
regular	0.18	0.250	0
regular	0.24	0.500	0
irregular	0.12	0.315	0
irregular	0.12	0.280	0

Deck position and orientation estimates for one flight and wave condition are shown in figures 12 and 13. The error in the position and attitude estimates is shown in figures 14 and 15. These plots show good agreement between the UKF estimates and the ground truth, though there are some constant biases in the estimates. These biases are generally small, however, and are likely attributed to slight errors in the nominal orientation of the camera in the vehicle body frame.

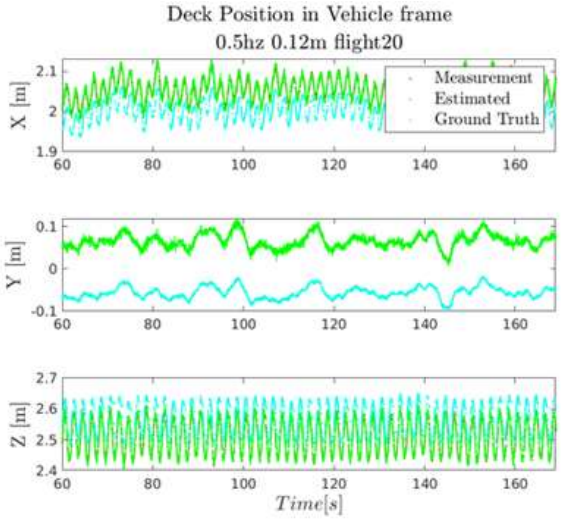


Figure 12: Sample plot of UKF position estimates compared to the ground truth.

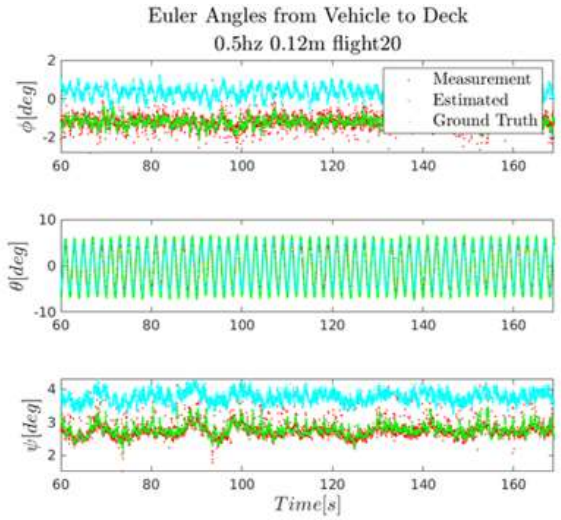


Figure 13: Sample plot of UKF attitude estimates compared to the ground truth.

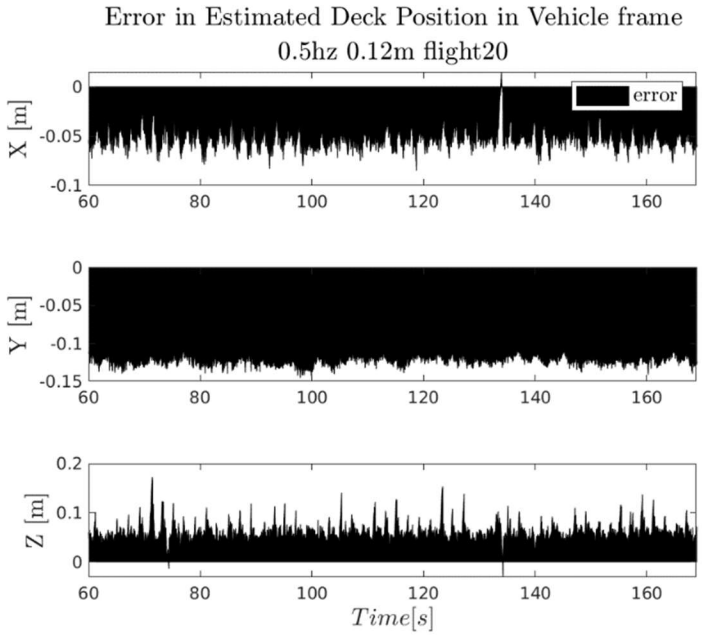


Figure 14: Sample plot of UKF position estimation error.

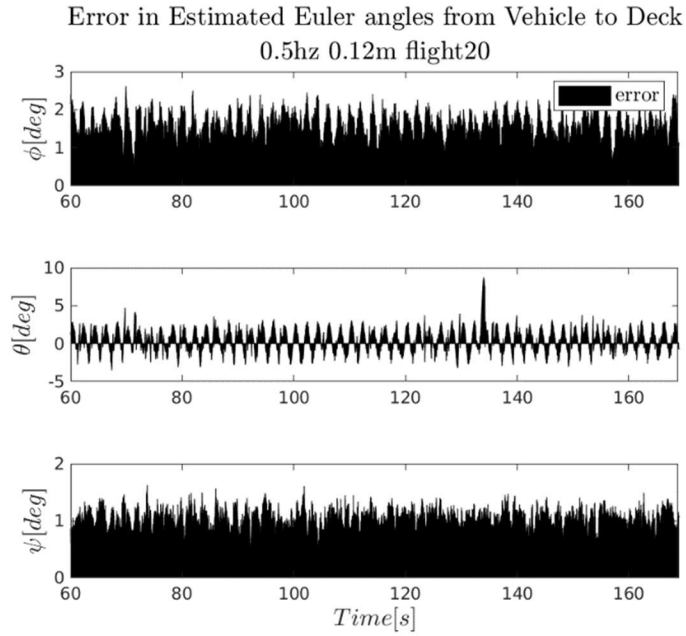


Figure 15: Sample plot of UKF attitude estimation error.

Figure 16 shows the error in the relative deck position state estimate versus time for eight different flights with the hexacopter hovering around 2.5 meters above the deck, with the deck subject to the wave conditions listed in table 1. The results show that the errors in the position estimates for each flight are consistently grouped within about 15 centimeters of one another and are largest in the Y degree of freedom. This is because the heading angle of the camera has a slight misalignment with the UAV body frame. Due to this, the error decreases with decreased distance from the deck.

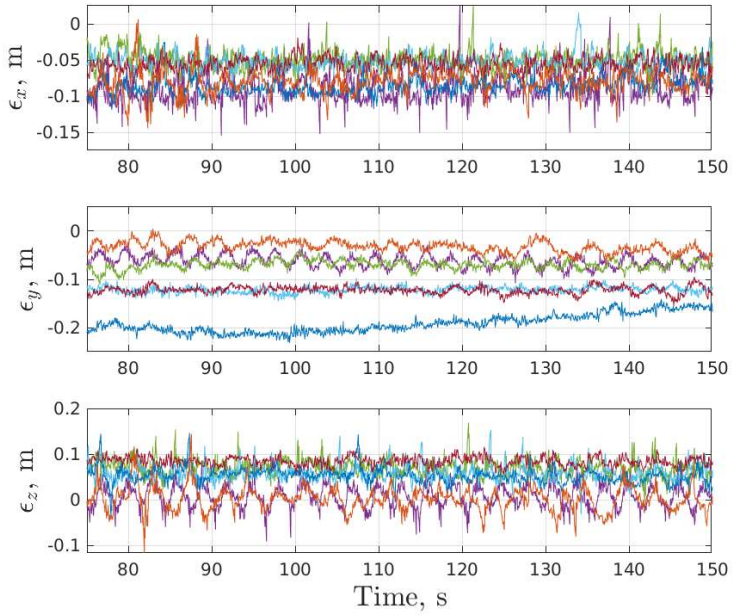


Figure 16: UKF position error for eight different flights.

2.7.1.2 Estimator-Based Autonomous Landings in the MASK

Fifteen autonomous landings were executed using the baseline guidance algorithm with the estimated relative deck states in the control loop. The wave motion parameters for these landings are listed in table 3. Note that the wave angle of 60 degrees in table 3 refers to waves approaching the deck from 60 degrees off the bow to starboard.

Table 3: Wave conditions for estimator-based landings.

Condition	Wave Amplitude [m]	Wave Frequency [Hz]	Wave Angle [deg]	Number of Tests Performed
No Waves	0	0	-	3
regular	0.15	0.33	0	2
regular	0.20	0.33	0	2
regular	0.20	0.50	0	2
regular	0.10	0.25	60	1
regular	0.10	0.50	60	2
regular	0.15	0.50	60	3

Figure 17 shows the position of the vehicle relative to the deck for each of the 15 autonomous landings. Vision-based state estimates are plotted in red while the relative position calculated from the motion capture measurements is shown in blue. Figure 18 shows both the estimated and Optitrack-measured Euler angles describing the attitude of the vehicle relative to the deck. These results show that the Optitrack and estimated measurements generally agree quite well during the landing procedure.

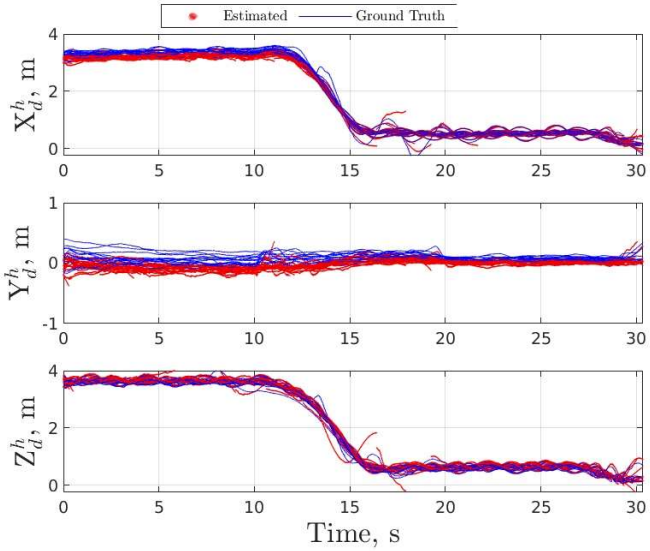


Figure 17: Position estimates compared to ground truth for all flights.

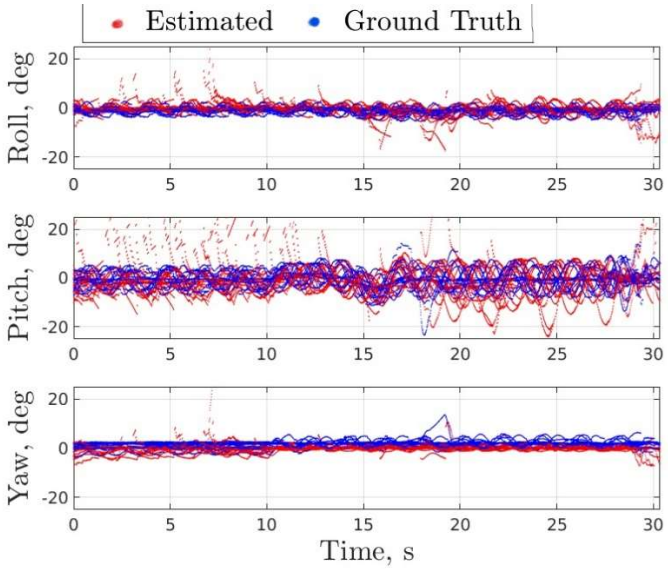


Figure 18: Pitch estimates compared with ground truth for all flights.

Figure 19 shows the estimated and true standard deviation of error for the first six state estimates, (x_d^h y_d^h z_d^h ϕ θ ψ), for all 15 on-board vision system and deck state estimator-based landings.

$$\boldsymbol{\sigma}_{True} = \sqrt{(\hat{\boldsymbol{x}} - \boldsymbol{x})(\hat{\boldsymbol{x}} - \boldsymbol{x})^T} \tag{43}$$

$$\boldsymbol{\sigma}_{Estimated} = \sqrt{Trace(\boldsymbol{P})} \tag{44}$$

Figure 20 shows the average of the estimated and true standard deviations of error for the first six state estimates for the same 15 landings vs Euclidean distance between the vehicle IMU and the marker array center. The large standard deviation of the error in the beginning of these plots is due to the fact that the camera does not yet have the tag in view. The large standard deviation at the end

of the plots is because once the vehicle lands the camera again loses sight of the tag. At this point the landing has already been completed, however, and the deck state estimates no longer matter. Overall, these results, combined with the fact that all 15 landings were completed successfully, give confidence in the developed vision-based sensing and estimation algorithms.

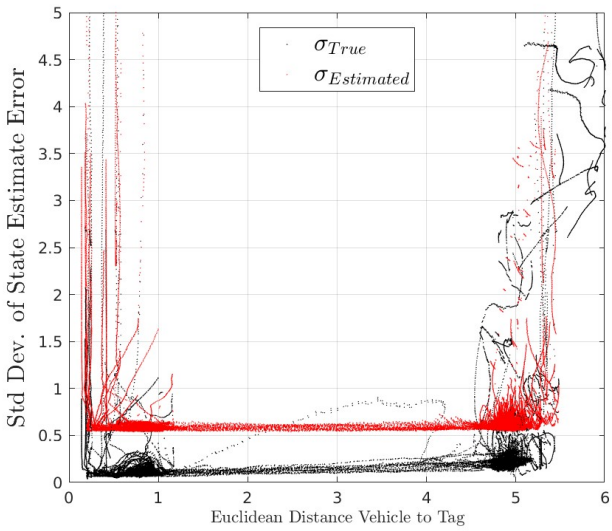


Figure 19: Estimated vs true average standard deviation of error in the UKF state estimates.

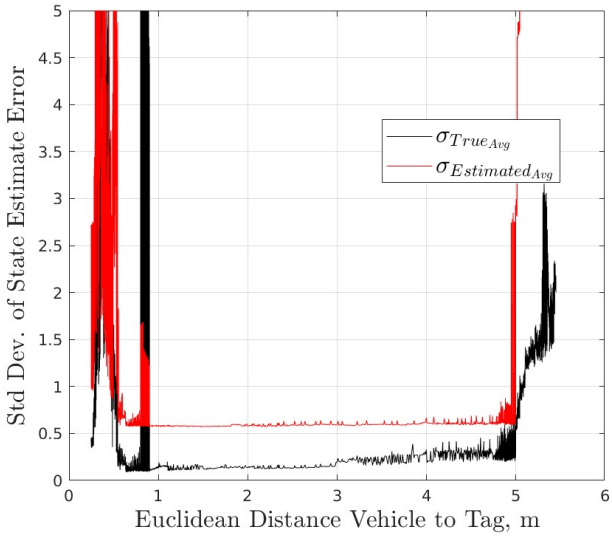


Figure 20: Estimated vs true average standard deviation of error in the UKF state estimates.

2.7.2 Evaluation of QP and Baseline Trajectory Generation Methods

The tests conducted in the MASK in October 2022 focused on the evaluation of autonomous landing trajectory generation methods. The remainder of this section will discuss the test setup and results for these particular experiments.

2.7.2.1 Wave Conditions

The guidance algorithms were evaluated via landings in three separate stochastic wave conditions. The wave conditions are referred to as stochastic because a single sinusoidal wave form was not used. Rather, the waves are produced to match a realistic wave spectrum. The standard deviation (STD) and maximum difference from the mean are tabulated for each degree of freedom at each wave condition in table 4. Ship heading and heading rate were excluded from this table as oscillations in heading were mild for these tests. Note that ω_p in table 4 is the frequency corresponding to the peak in the wave amplitude spectrum, and ψ_w is the wave heading. For wave conditions 1 and 2 the waves approached the bow of the ship directly, but for wave condition 3 the waves approached at an angle of 45 degrees to starboard.

Wave condition 1 was the most mild, which is reflected by the deck motion statistics shown in table 4. Wave conditions 2 and 3 were similar, but the 45 degree wave angle for condition 3 led to significantly more aggressive roll and sway motion and less aggressive surge motion. Wave condition 3 was found to be the most difficult for the autonomy algorithms to perform landings in, though it should be noted that this may not be entirely due to larger lateral motions. The wave makers in the MASK dissipate less energy with a 45 degree wave heading than with 0 degree wave heading, which leads to slightly more aggressive motions on average.

Table 4: Ship motion properties for MASK tests performed in October 2022.

	Cond. 1 ω_p = 2.8 rad/s ψ_w = 0 deg	**Cond. 2** ω_p = 2.3 rad/s ψ_w = 0 deg	**Cond. 3** ω_p = 2.3 rad/s ψ_w = 45 deg
DOF	STD/Max	STD/Max	STD/Max
Roll	1.46/5.24	1.89/8.04	3.74/16.20
Roll Rate	5.16/15.51	5.74/17.79	12.38/50.36
Pitch	2.82/10.32	3.82/15.58	2.30/10.96
Pitch Rate	9.20/25.76	10.66/35.42	8.71/30.94
Surge	0.11/0.41	0.15/0.64	0.10/0.41
Surge Rate	0.09/0.27	0.17/0.58	0.13/0.45
Sway	0.07/0.24	0.08/0.36	0.14/0.72
Sway Rate	0.05/0.17	0.09/0.28	0.15/0.54
Heave	0.05/0.23	0.11/0.44	0.11/0.41
Heave Rate	0.15/0.42	0.26/0.93	0.27/0.94

[a]Angles and angular rates are shown with units of deg and deg/sec. Translation and translational rates are shown with units of meters and meters/sec.

To give a qualitative feel for deck motion, a segment of deck heave at wave condition 2 is plotted in figure 21. Only heave is plotted here because the deck motion was most aggressive in heave for all wave conditions, though significant oscillations in surge and sway were present as well. Referring to figure 21, two important aspects of the deck motion can be noted. First, the motions are irregular and difficult to predict, which is essential for accurate evaluation of the QP guidance algorithm as it incorporates deck motion predictions. Also, it should be noted that the deck regularly undergoes motions that are quite aggressive for the scale the tests were conducted at. At times the deck goes through oscillations with amplitudes around 35 to 40 centimeters, meaning the deck goes

through a total throw of 70-80 centimeters, and this occurs in a time span of 1 to 2 seconds. Relative to the UAV, a throw of 70 centimeters is around 125 percent the UAV lateral hub-to-hub distance (which is approximately 53 centimeters).

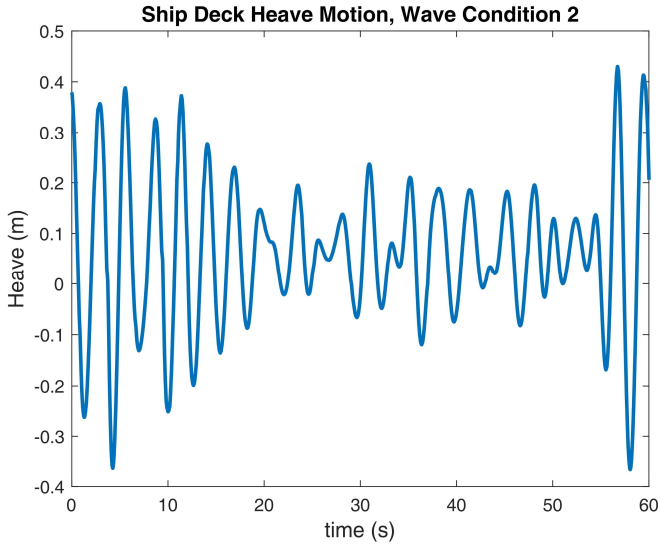


Figure 21: Sample deck heave motion time history.

2.7.2.2 Test Cases

A total of 149 autonomous landings were performed (79 with the QP guidance algorithm and 70 with the baseline algorithm), with 10 different flight controller configurations used. The controller configurations are summarized in table 5. At least 10 landings were performed for each combination of control case and wave condition. However, in instances where the UAV battery had enough capacity remaining to perform additional landings before switching test cases, additional landings were recorded.

Table 5: Landing test cases.

Control Case	Guidance Algorithm	Roll and Pitch[a] Tracking BW (rad/s)	X and Y Tracking BW (rad/s)	Heave Tracking[a] BW (rad/s)	X,Y/Z[b] Jerk Limit (m/s^3)	Wave Conds. Tested (No. of landings)[c]
1	QP	High	2.23	High	9/9	1 (10), 2 (12), 3 (11)
2	QP	Med	1.67	High	7/9	2 (11)
3	QP	Low	1.11	High	5/9	2 (12)
4	QP	High	2.23	Med	9/7	2 (10)
5	QP	High	2.23	Low	9/5	2 (12)
6	baseline	High	10	30	NA	1 (10), 3 (10)
7	baseline	High	2.23	30	NA	2 (10)
8	baseline	High	2.23	15	NA	2 (10)
9	baseline	High	2.23	6	NA	2 (10)
10	baseline	High	2.23	High	NA	2 (10), 3 (10)

[a] Refer to table 5 for high, medium, and low values.
[b] Jerk limits in QP planner are reported as "X,Y limit" / "Z limit." For example, 7/9 for control case 2 means the X,Y jerk limit was $\pm 7\ m/s^3$ and the heave jerk limit was $\pm 9\ m/s^3$ for this case.
[c]The number in parentheses is the number of landings performed at that wave condition. For example, "2 (12)" for control case 3 means that 12 landings were performed at wave condition 2 for control case 3.

Referring to table 5, control cases 1 - 5 used the QP algorithm, but with varied pitch, roll, and heave tracking bandwidths to simulate reduced aircraft mobility. The values corresponding to high,

medium, and low cases are reported in table 6, which were chosen by Froude scaling realistic values for an aircraft similar to a UH-60.

Table 6: Tracking bandwidth cases for QP algorithm.

	Model Scale Bandwidth			Full Scale Equivalent		
	High	**Med**	**Low**	**High**	**Med**	**Low**
$\omega_{\theta,cf}$	11.14	8.36	5.57	3.00	2.25	1.50
$\omega_{\phi,cf}$	11.14	8.36	5.57	3.00	2.25	1.50
$\omega_{Z,cf}$	3.71	1.86	0.74	1.00	0.50	0.20

Note: All entries in units of rad/s

Control cases 6-10 used the baseline algorithm, with pitch and roll tracking bandwidth fixed to the high configuration and heave tracking bandwidth varied. Note that control cases 6-9 used Z bandwidths that were significantly higher than the “High" configuration used for the QP algorithm. For these cases, the bandwidth is reported directly in table 5. This was done because the deck relative commands produced by the baseline guidance algorithm were passed directly into the UAV altitude controller command filter, and the UAV lags the deck motion increasingly as command filter frequency is reduced. The baseline controller was therefore first tested with a very high Z bandwidth (30 rad/s) to simulate a case where the aircraft can track the deck with minimal lag. The command filter frequency was then progressively reduced until the lag was too great to perform soft landings. This occurred by the time the Z bandwidth was reduced to the “High" case used for QP landings (i.e., control case 10).

For all control cases, except for control case 6, the X and Y bandwidths were assigned by dividing the pitch and roll bandwidth by 5, ensuring adequate frequency separation between the inner and outer loops. For control case 6 the X and Y bandwidths were set 10 rad/s, giving a faster response, but with a higher potential for significant overshoot.

For cases with the QP algorithm, jerk limits were also varied with tracking bandwidth. For example, when heave was in the high bandwidth configuration, the maximum Z axis jerk constraint is set to $\pm 9\ m/s^3$. When heave was in the medium and low bandwidth configurations this was reduced to $j_{Z,max} = \pm 7\ m/s^3$ and $j_{Z,max} = \pm 5\ m/s^3$, respectively. The X and Y jerk limits were tied to the pitch and roll bandwidths in the same manner. To simulate the QP algorithm controlling an aircraft with reduced maneuverability some restriction of output constraints is necessary, because if the constraints are too loose the optimal control algorithm can overdrive the command filter input to compensate for the reduced tracking bandwidth. For the UAVs used during testing this would not be problematic as the reduced bandwidth is artificial, but for a full-scale aircraft, bandwidth limits are related to real physical limitations such as actuator rate saturations. Here jerk was restricted because it was found to be the most common constraint to approach a limit, and also is related to actuator rate saturation. The acceleration and velocity output constraints were fixed to $V_{max} = \pm 7\ m/s$ and $a_{max} = \pm 3.5\ m/s^2$.

2.7.2.3 Deck Motion Prediction Accuracy

Prior to assessing the performance of each landing algorithm, discussion of the deck state prediction accuracy is pertinent as the predictions are used directly in the QP guidance algorithm. To give a qualitative feel for the deck prediction accuracy, a representative sample of deck heave predictions

produced during flight is given in figure 22. This shows that the predictions are very accurate at 0.5 seconds ahead, but the accuracy drops off quickly. At just 1.5 seconds ahead, errors in both magnitude and phase of the predicted heave are significant. By 2.5 seconds ahead, the heave magnitude is regularly under-predicted, and in some cases the prediction does not deviate far from the mean of the heave oscillations.

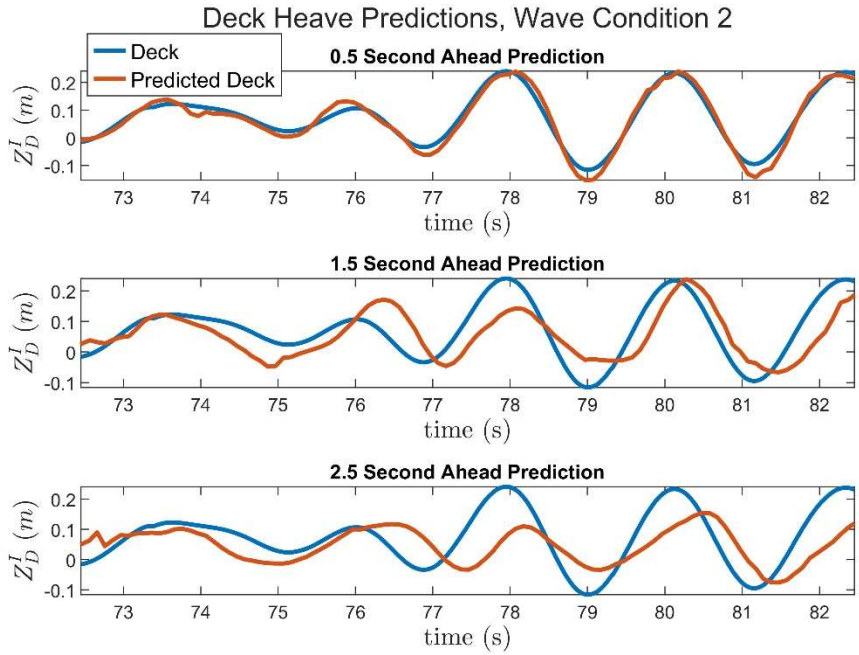


Figure 22: Model scale deck heave predictions.

These observations are supported by the prediction accuracy statistics calculated across trials shown in figure 23. The errors for a single “trial" were computed as follows:

1. A time point in a given landing flight test was chosen.
2. The deck motion predictions produced at the chosen time point were saved.
3. The saved predictions were compared to the actual deck motion over the next three seconds.

The time points chosen to evaluate the predictions were spaced out by at least 3 seconds in all cases. This ensured that directly adjacent time points were not used when calculating prediction errors, providing trials taken at a more diverse set of conditions. The number of trials used to calculate the prediction error statistics was increased until the mean and standard deviation converged.

Looking at the wave condition 2 prediction errors shown in figure 23, it can be seen that X and Y prediction errors start converging to the true deck value at slightly over 1.5 seconds out and the heave predictions begin to converge between 1 and 1.5 seconds out. On average, the long-term position prediction errors are slightly under the mean value of deck motion. For example, the average deck heave deviation from the mean is about 9 cm, while the average heave prediction error at 2.5 seconds is about 6 cm, though in many cases the long-term prediction errors are more than double this value. The attitude errors also follow the expected trend, decreasing in the final second of the prediction horizon. The same is true for the velocity predictions, but the velocity predictions also exhibit a number of sharp spikes in prediction error, which sometimes occur with

under 1 second remaining in the prediction horizon. This is related to occasional data buffering that occurred when sending deck measurements from the ground station to the UAV over wifi. When this happens, small step-like inputs are passed through the filters used to estimate deck velocity, resulting in spikes in the velocity estimates. Since the AR model calculates future outputs based off past outputs, this can result in sharp spikes appearing in the predictions. Despite these issues, it will be shown that QP algorithm was able to perform well with the quality of deck predictions obtained here.

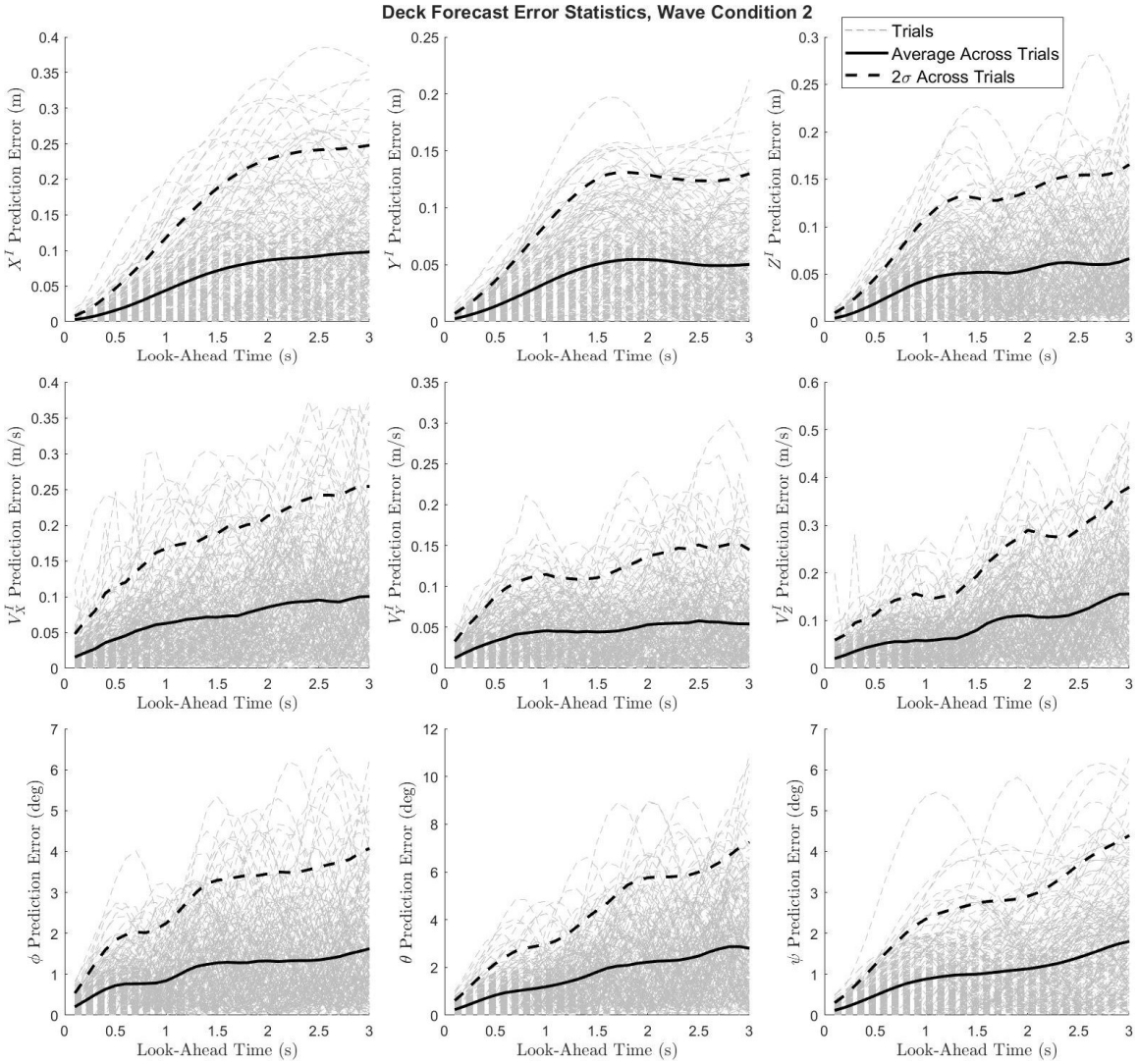


Figure 23: Position, velocity, and attitude prediction error statistics from landings performed at wave condition 2 (see tables 5 and 6).

It should also be noted that the prediction quality is dependent on both the frequency and magnitude of deck motion. If the prediction error statistics shown in figure 23 scaled following the Froude scaling method used for determining model-scale UAV bandwidths, then the model-scale predictions converging at 1.5 seconds ahead would translate to full-scale predictions converging at slightly over 5 seconds ahead. A formal study would need to be conducted to understand the validity

of this scaling, but full-scale simulations with similar time-series based prediction methods have shown 5-second ahead predictions that appear at least qualitatively similar to this assumption. The effect of deck motion aggressiveness on prediction performance can be seen from figure 24, where position prediction errors at wave condition 1 (the most mild wave condition) are notably lower than the errors calculated at wave conditions 2 and 3.

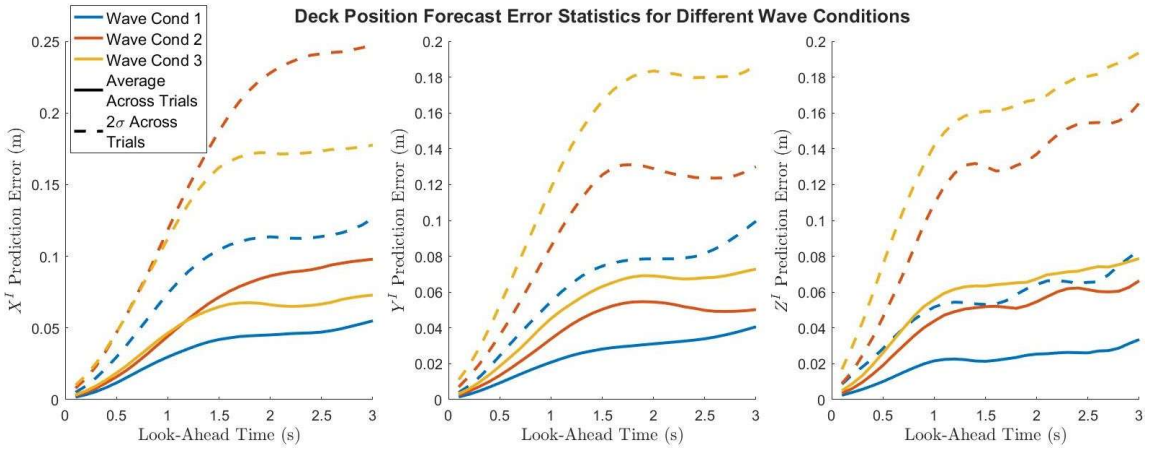


Figure 24: Position prediction errors for each wave condition.

2.7.2.4 Sample Landing Time Histories

A sample landing at wave condition 2 with the baseline guidance algorithm is shown in figure 25 and a time lapse corresponding to this trajectory is shown in figure 26. Similarly, a sample landing at wave condition 2 with the QP guidance algorithm is shown in figure 27, and a time lapse of this landing is shown in figure 28. In both cases, a successful landing was completed with low terminal deck-relative positions and velocities.

For the baseline landing case, the relative descent rate at touchdown was very close to the $0.25\ m/s$ descent rate bias commanded in the baseline guidance algorithm. This is not surprising as the Z bandwidth for control case 7 was high and the UAV could effectively track the deck. The baseline algorithm also matched X and Y velocity well in this instance, but this is somewhat coincidental as the X and Y command filter for control case 7 was low enough to introduce significant lag in X and Y translation. Also, note that the sharp changes in attitude that occur shortly after deck contact are due to a combination of rigid landing gear on the UAV and the throttle being cut several centimeters above the deck.

For the QP landing case, similar relative velocities were obtained at landing but deck oscillations that would have been followed using the baseline guidance algorithm were largely ignored. This illustrates how the QP algorithm can reduce the required maneuvering, though the baseline "deck tracking" method is simple and effective when the aircraft has the required control authority. Also, for the case shown in figure 27, the land-time update scheme included in the QP algorithm shifted the land time from a point with significant upward heave to a point near a peak in the oscillations. This prevented the UAV from needing to change direction to match heave motion at landing. It will be shown in the following subsections, however, that the land time update harmed performance in many other cases. The QP algorithm also manages to match deck attitude in this case, though deck attitude was small at landing. The capability of the landing algorithms to consistently match

position, velocity, and attitude will be clarified through the statistics presented in the following subsections.

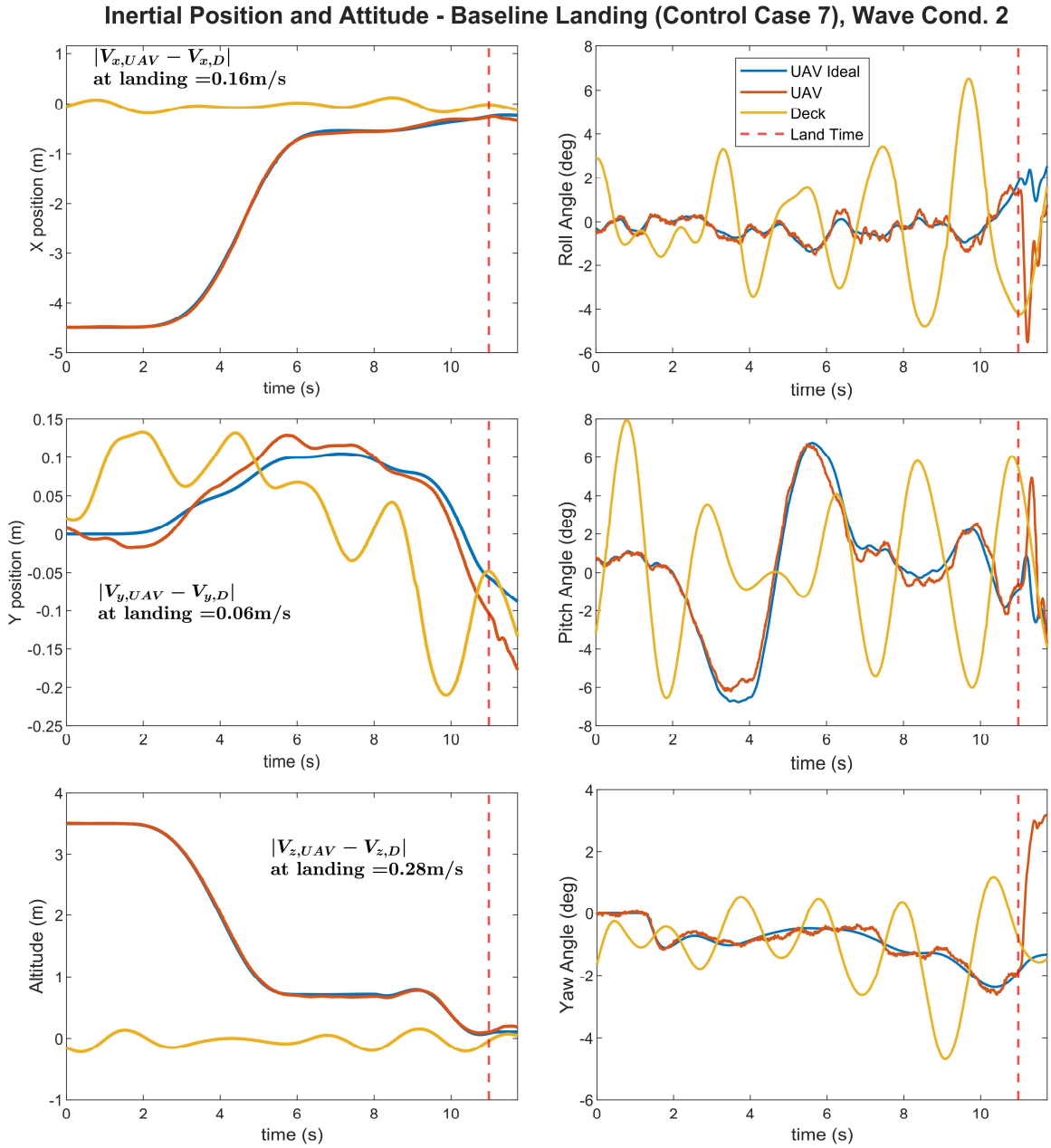


Figure 25: Sample landing time history with baseline guidance algorithm.

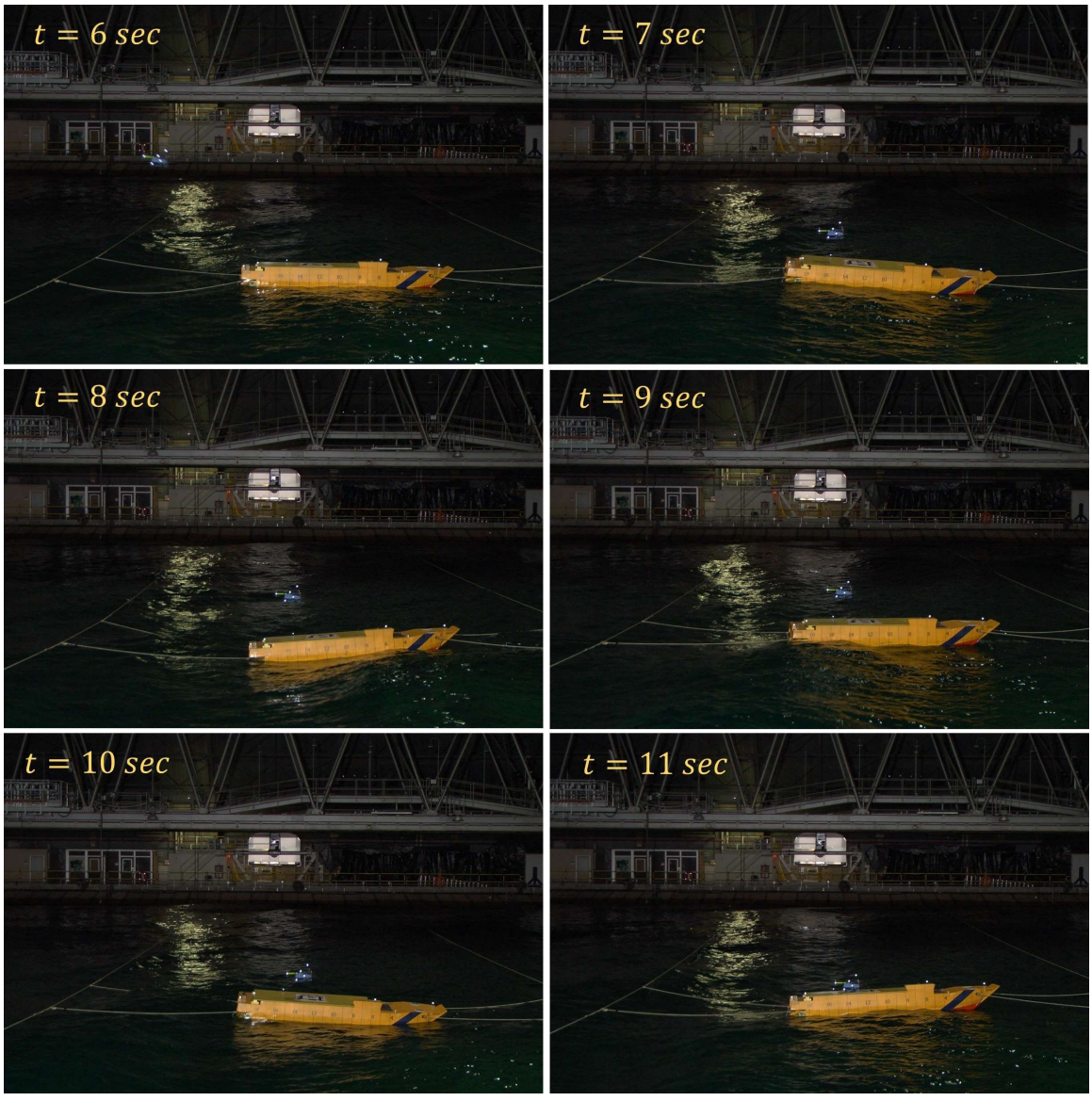


Figure 26: Time-lapse corresponding to sample the baseline landing shown in figure 25.

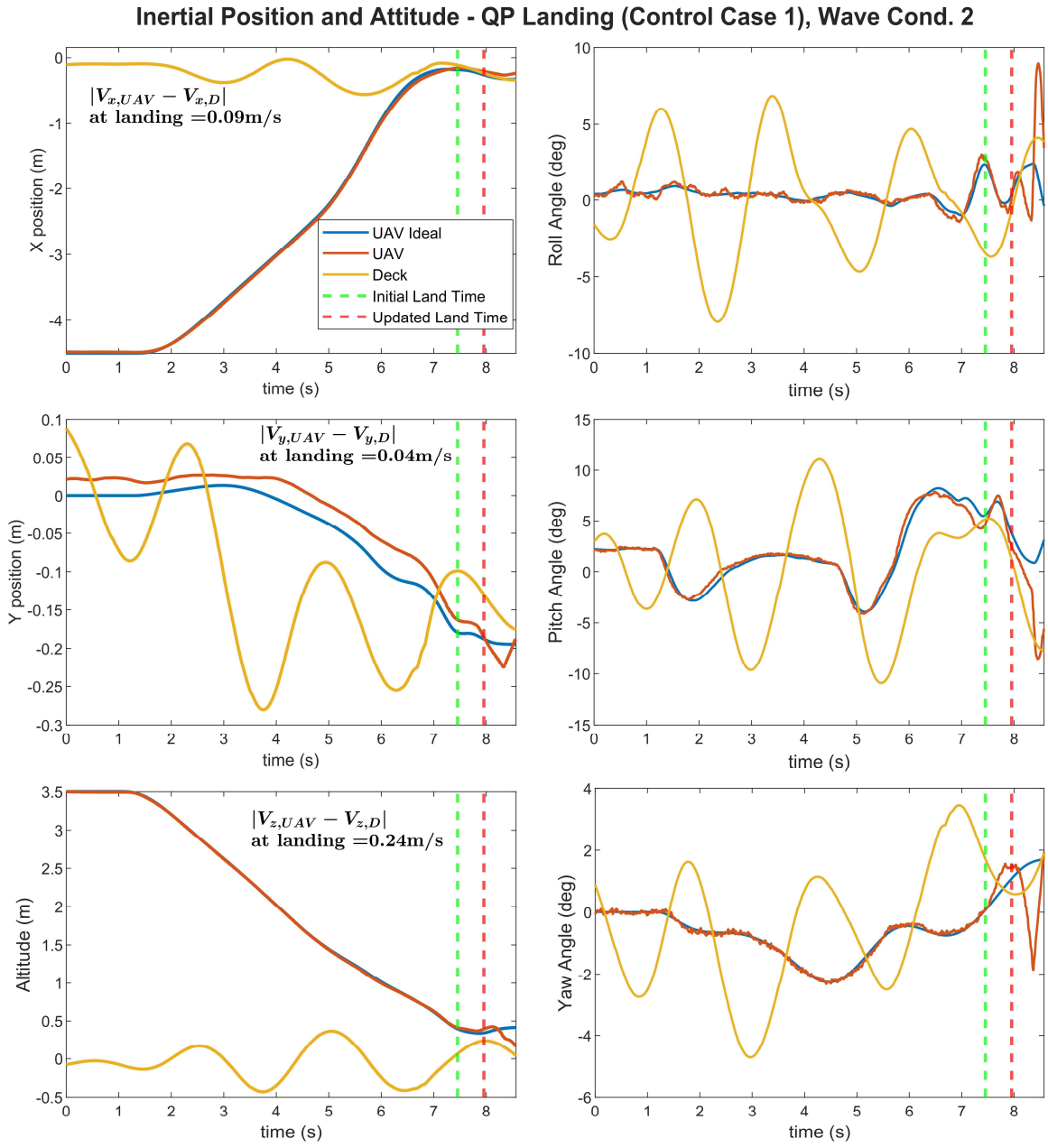


Figure 27: Sample landing time history with QP guidance algorithm.

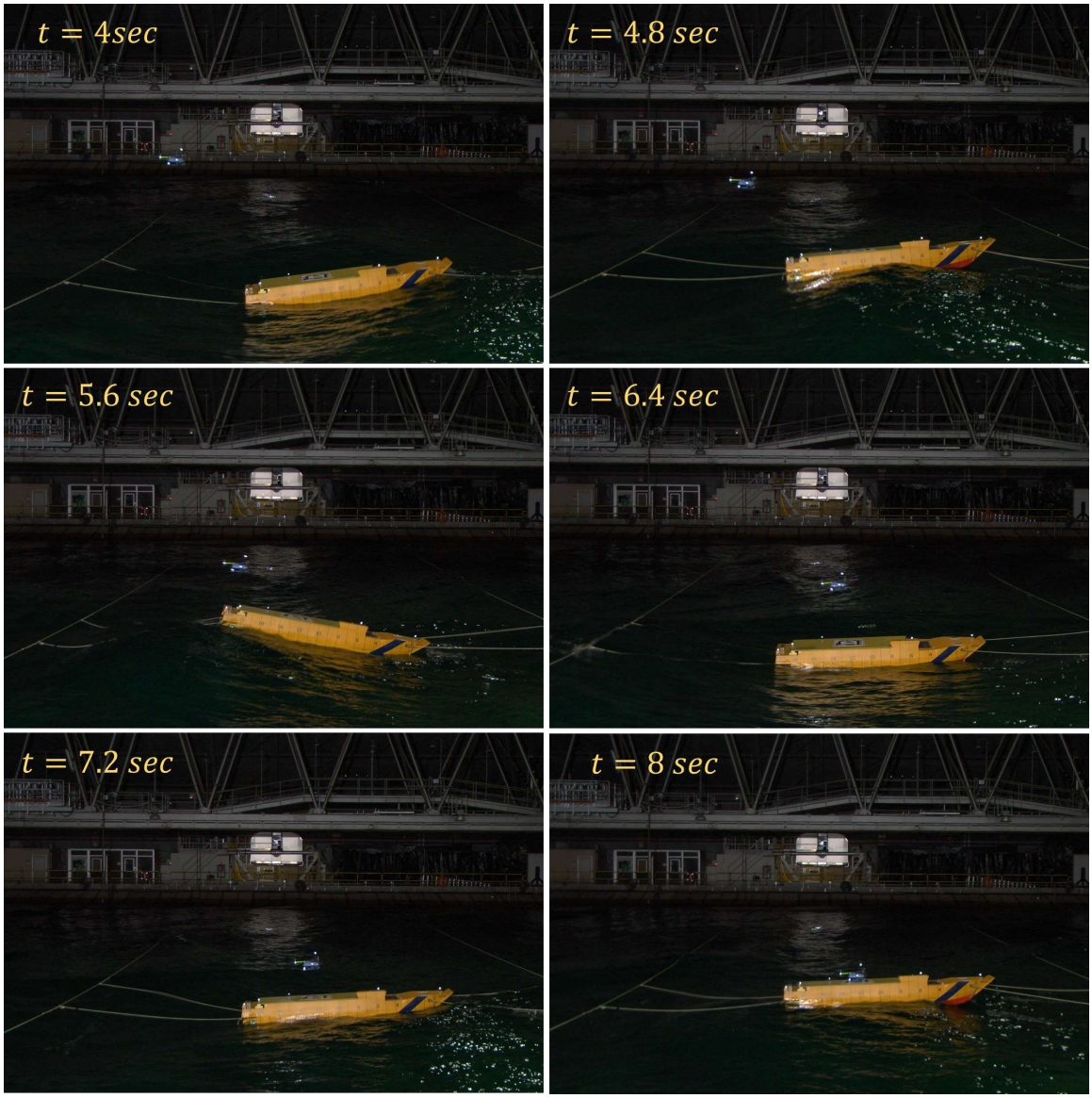


Figure 28: Time-lapse corresponding to the sample QP landing shown in figure 27.

2.7.2.5 Position, Velocity, and Attitude Landing Errors

The deck-relative position and velocity errors are shown for all flight tests in figures 29 and 30. Focusing first on the baseline landing algorithm, the results show the simple deck tracking method to provide low deck-relative descent rates for all high bandwidth conditions tested in wave conditions 1 and 2. For example, control case 6 in wave condition 1 (the most mild wave condition) and control case 8 in wave condition 2 (the "moderate" wave condition, though deck motion was still quite aggressive) both led to deck-relative descent rates of under 0.4 m/s for all landings, with averages close to the 0.25 m/s Z velocity bias commanded in the baseline landing algorithm. For control case 7 in wave condition 2 the results are similar, though two landings did end with relative descent rates of around 0.45 m/s. While testing, however, landings that were no longer visibly perceived as "soft" did not occur until relative descent rates of around 0.6 m/s. One such landing

occurred for control case 6 in wave condition 3 (the most aggressive wave condition), though the other landings at this test condition had impact velocities of around 0.4 m/s or less.

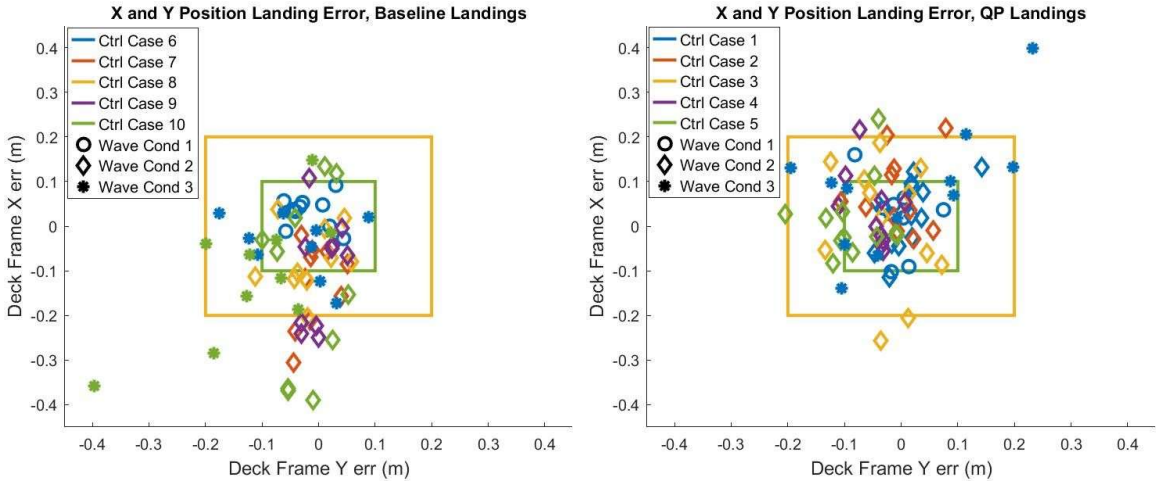


Figure 29: X and Y position landing errors with baseline (left) and QP (right) guidance algorithms.

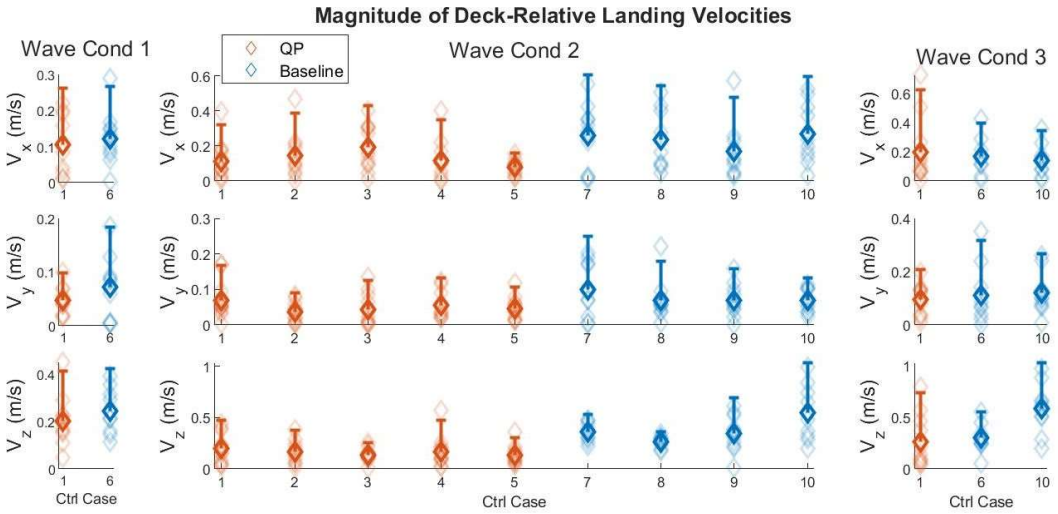


Figure 30: Deck-relative landing velocities at touchdown. The faded markers are values from individual landings, the bold markers show the average value, and the one-sided error bar represents 2σ of the deck-relative landing velocity magnitudes.

The X and Y position and rate errors were also typically low for control case 6, both in wave condition 1 and the more aggressive wave condition 3. All landings performed with control case 6 led to X and Y position errors of less than 18 cm and X and Y velocity errors under 0.42 m/s. For control cases 7-10, however, there were instances with X velocity errors of around 0.6 m/s and the position errors shown in the left plot of figure 29 are clearly inflated as well. This difference in performance was expected as, referring back to table 5, X and Y bandwidth was highest for control case 6. Further, despite the lack of frequency separation between the inner and outer loops for control case 6, the frequency of deck motion was low enough to avoid significant overshoot resulting from commanding the UAV to track the deck. If roll and pitch bandwidths are not high, however, simply using a high outer loop command filter frequency may not be a tractable solution.

The position and velocity errors reported in figures 29 and 30 also clearly illustrate the effect of reduced Z bandwidth on baseline algorithm performance. For example, moving from control case 8 to control case 9 at wave condition 2, the Z bandwidth was reduced from 15 rad/s to 6 rad/s. This effectively adds an additional delay on the deck relative commands: the command filter frequency of $\omega_z = 15\, rad/s$ corresponds to a delay of about 0.11 seconds, while $\omega_Z = 6\, rad/s$ translates to a delay of about 0.27 seconds. As expected, the reduced bandwidth leads to increased impact velocities, demonstrating that deck tracking can be sensitive to increased lag. When moving to control case 10, ωz = 3.71 rad/s was used (equivalent to the "high" bandwidth case for the QP landings), leading to a large delay of about 0.44 seconds and resulting in excessively high impact velocities. Additionally, with the UAV and deck altitudes out of phase, X position errors also increased due to the UAV contacting the deck prior to the expected land time.

Examining the results reported in figure 30 for the QP landings, on the other hand, shows the QP landing algorithm is insensitive to reduced bandwidth. In wave conditions 1 and 2, control cases 1-5 all provide low velocity errors on average, though there is one outlier for control case 4 at wave condition 2 which resulted in a Z velocity error of close to 0.6 m/s. There is also a slight increase in the average X velocity error when moving from control case 1 to 3 at wave condition 2. This is likely due to the decreased X and Y bandwidths and restricted jerk constraints for control cases 2 and 3, but the total deviation in average velocity error is under 0.1 m/s when comparing control cases 1 and 3. The X and Y position landing errors follow a similar trend, with errors being roughly consistent across test cases and remaining around 20 cm or less for the majority of cases in wave conditions 1 and 2.

There were several cases where the QP algorithm did not perform well, however. Looking at the velocity errors for control case 1 at wave condition 3, hard landings resulted in two cases, one of which also terminated with a very large position error. The QP algorithm also performed poorly in terms of attitude matching at touchdown. This is shown by the results in figure 31, where the QP landings terminated with deck-relative pitch attitudes of close to 15 degrees in several instances. The pitch attitude errors were often greater than those obtained with the baseline algorithm, despite

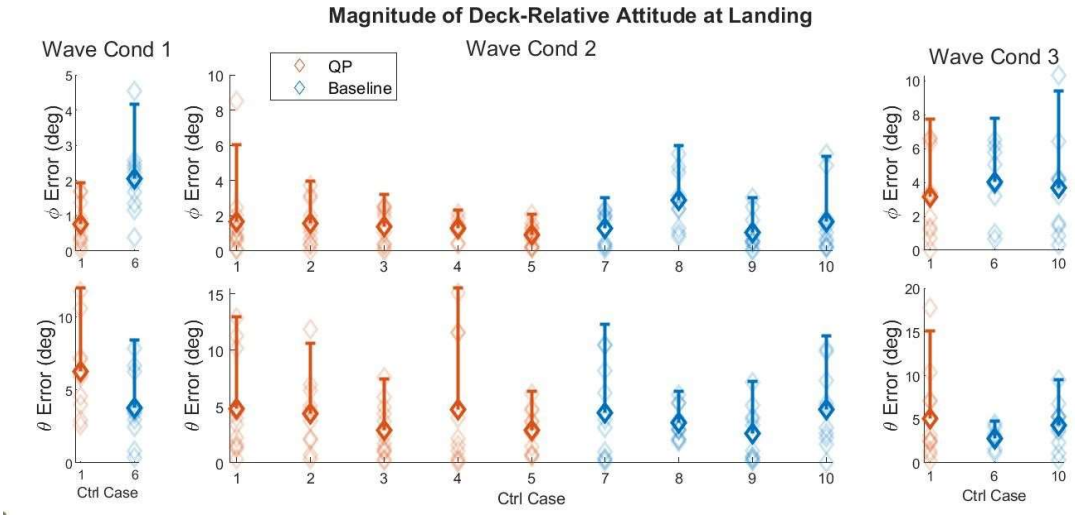


Figure 31: Deck relative landing attitude. The faded markers are values from individual landings, the bold markers show the average value, and the one-sided error bar represents 2σ of deck-relative landing attitudes.

the fact that the QP algorithm included some motivation for attitude matching in the cost function (albeit through a simplified relationship between attitude and acceleration). It will be shown in the following subsection, however, that the vast majority of cases where the QP algorithm performed poorly likely would have been averted with a change to the land-time update algorithm.

Despite the poor attitude matching performance, these results illustrate one of the major advantages of using a guidance algorithm like the QP method presented here. That is, integrating deck motion predictions and an optimal control solver allow the guidance algorithm to plan ahead for the aircraft dynamics and expected deck motions. Here, this allowed the QP algorithm to adequately match deck positions and velocities at lower bandwidths than the baseline landing algorithm. With a sequential loop closure architecture used to control X and Y translation, this in turn allows the outer loop controllers to be designed assuming adequate frequency separation. With the baseline algorithm, on the other hand, the outer loop command filter frequency was set to balance a trade-off between speed of response and overshoot.

The QP algorithm also has the potential to plan ahead for more restrictive output constraints. For the QP algorithm, this is different than planning for reduced bandwidth. The highly weighted terminal cost used to motivate matching deck state at touchdown outweighs the cost placed on control inputs. Without constraints included, the optimal control algorithm may therefore just increase the magnitude of control inputs to compensate for reduced bandwidth. With stringent output constraints included, however, the optimization algorithm may need to plan the landing path with the control action spread out over a larger time range, which also requires that deck motion predictions converge with enough lead time to avoid a rapidly changing desired terminal state at the end of the trajectory. Here, tests with restricted jerk constraints were included to study the ability of the QP algorithm to plan ahead for reduced mobility. The results showed the reduced jerk constraints to have little impact on performance, but the jerk constraints may simply have not been restricted enough to have a large impact. To better understand the extent to which the QP algorithm is capable of planning ahead for more stringent output constraints, additional testing with further restricted acceleration and jerk limits would need to be conducted.

2.7.2.6 Accelerations During Landing

Another hypothesized advantage of the QP algorithm was that accelerations during landing could be reduced, as deck motions do not need to be tracked. This proved not to be the case, however. Referring to the results shown in figure 32, the baseline algorithm did result in large Z accelerations around 2-2.5 seconds before landing, but this is an artifact of the baseline guidance logic. It was around this time in the landing sequence where the baseline algorithm would switch from tracking low pass filtered deck motion to tracking actual deck motion while descending. This introduces a small step input, resulting in sharp throttle commands and more aggressive accelerations. This behavior could have been averted by phasing in the switch between the approach and landing stages of the baseline algorithm, but this does illustrate an advantage of the QP algorithm: the QP algorithm results in smoothly regulated accelerations without the need to manage transitions between flight modes.

Looking at later time points in the landing sequence, the baseline and QP algorithms resulted in comparable Z acceleration magnitudes, though the baseline landing accelerations are defined by those of the deck and more aggressive deck motion may change this result. The QP guidance algorithm also generally resulted in higher X accelerations, but this was due to the X command filter frequency resulting in heavily filtered deck- relative commands for all baseline tests

conducted at wave condition 2. It should also be noted that the QP accelerations remained within the specified 3.5 m/s^2 constraint for all cases but one case. This one instance was due to the landing gear contacting the deck slightly harder than usual at touchdown, resulting in a spike in the acceleration in the final 0.2 seconds of the plotted trajectory.

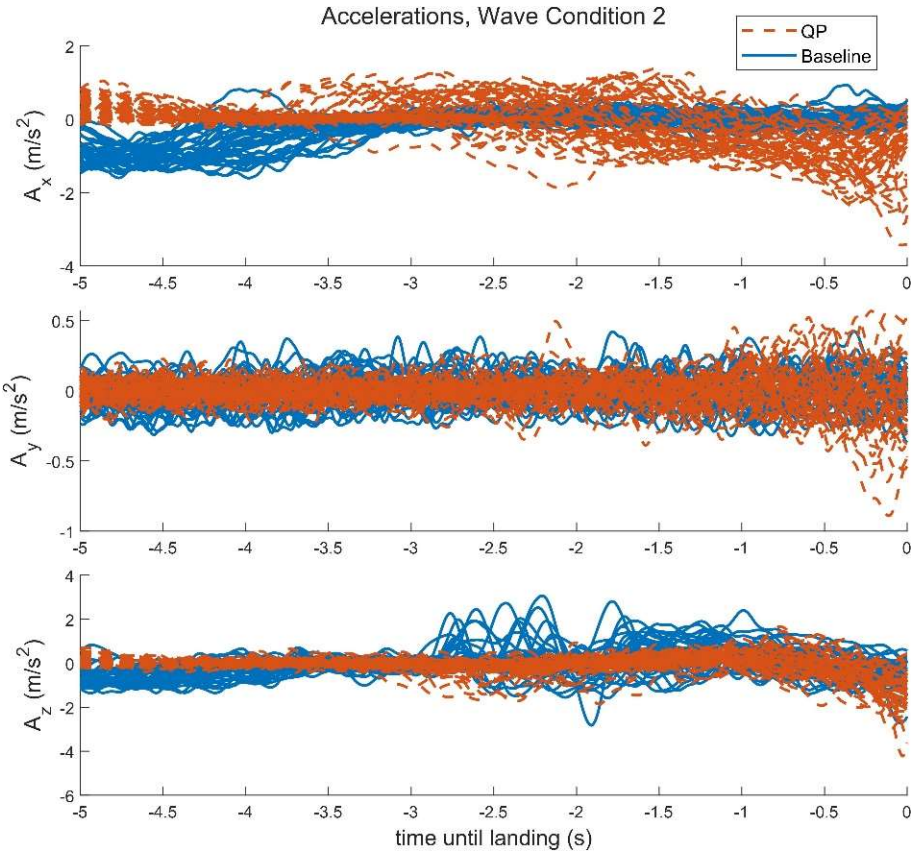


Figure 32: Accelerations during the last 5 seconds of landing for flights conducted at wave condition 2.

2.7.2.7 Factors Affecting QP Algorithm Performance

To understand the robustness of the QP algorithm, it is necessary to understand why the QP algorithm performed poorly in some instances. Here it was found that there were three contributing factors: the quality of deck motion predictions, the aggressiveness of deck motion at landing, and the land time update scheme included in the QP algorithm.

The individual impact of each of these factors is clarified by studying the plots shown in figure 33. Figure 33a plots the velocity norm at landing against the relative-descent rate at touchdown for all QP landings. As the deck motion was dominated by translational oscillations, the velocity norm is representative of the aggressiveness of deck motion. This plot shows that the cases with the three highest impact velocities all landed at times where deck velocity norm is large, which is an expected result.

More interestingly, figure 33a also reveals the impact of the land time update scheme. The markers plotted as a red "x" in figure 33a represent cases where the land time update scheme elected to

shorten the land time by a slight amount (the target land time was not reduced by more than 0.3 seconds in any case), and the black "*" markers represent all other QP landings. If the cases with shortened land times are all discarded, then every landing where the deck velocity norm was under 0.6 m/s but the relative descent rate was over about 0.3 m/s would be removed. Further, only one landing with a relative descent rate of greater than 0.5 m/s would be retained. This landing is marked by "Flight 59" on figure 33a and, by comparison with figure 33b, it is seen that abnormally poor deck heave predictions occurred during the final 0.5 seconds of the landing. Further, the deck was moving aggressively at touchdown for this case. Also, it should be noted that while removing the QP landings with shortened maneuver duration would remove slightly over 50 percent of the QP landing cases, the removed cases are not disproportionately biased toward cases with high deck velocity norms at landing. For example, 5 of 8 cases with a deck velocity norm greater than 0.5 m/s would be removed, and 5 of 10 cases with a deck velocity norm of greater than 0.45 m/s would be removed. The removed landings are heavily biased toward cases with high impact velocities, however.

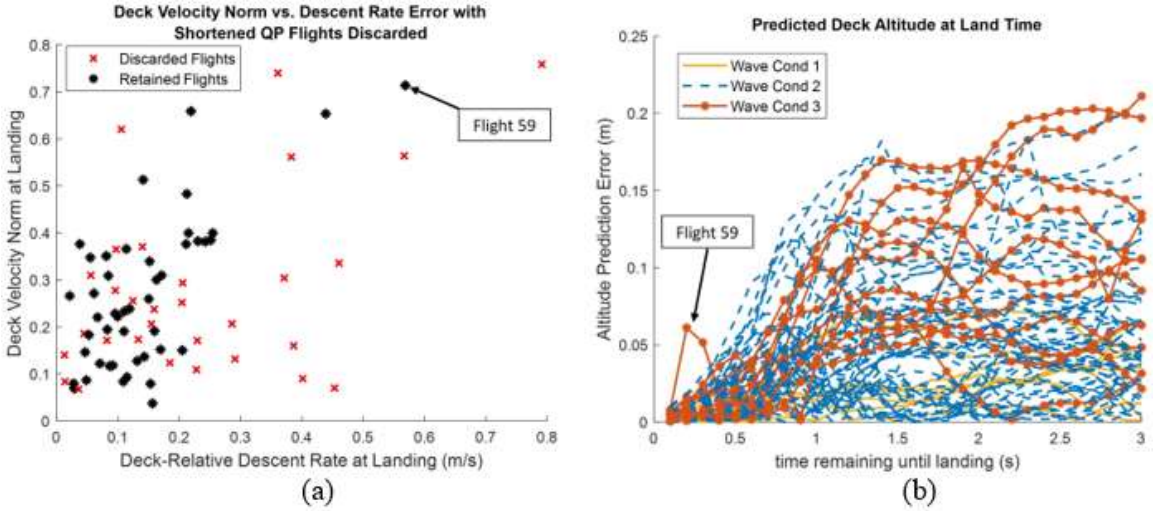
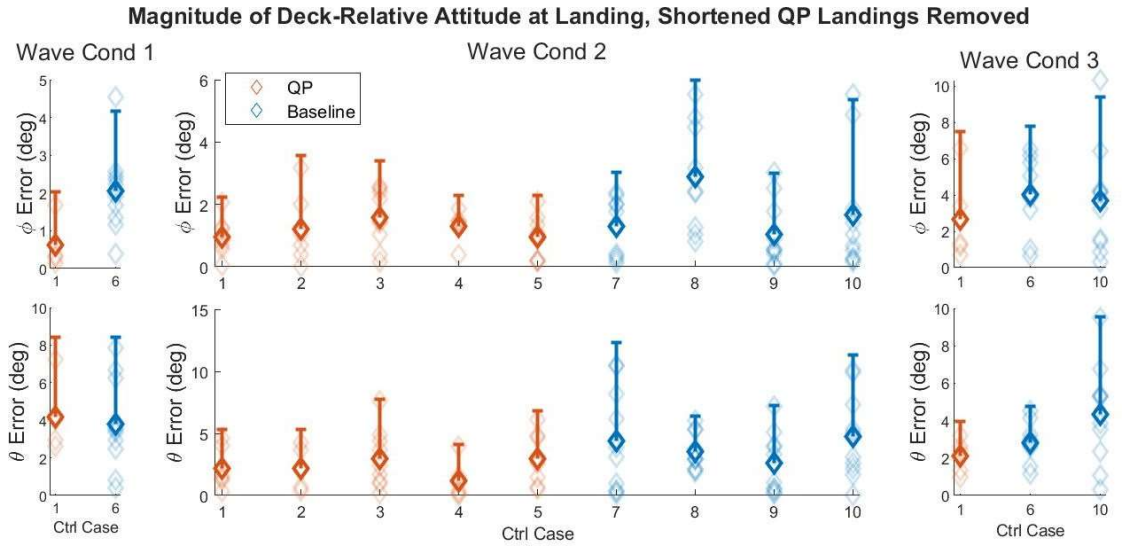
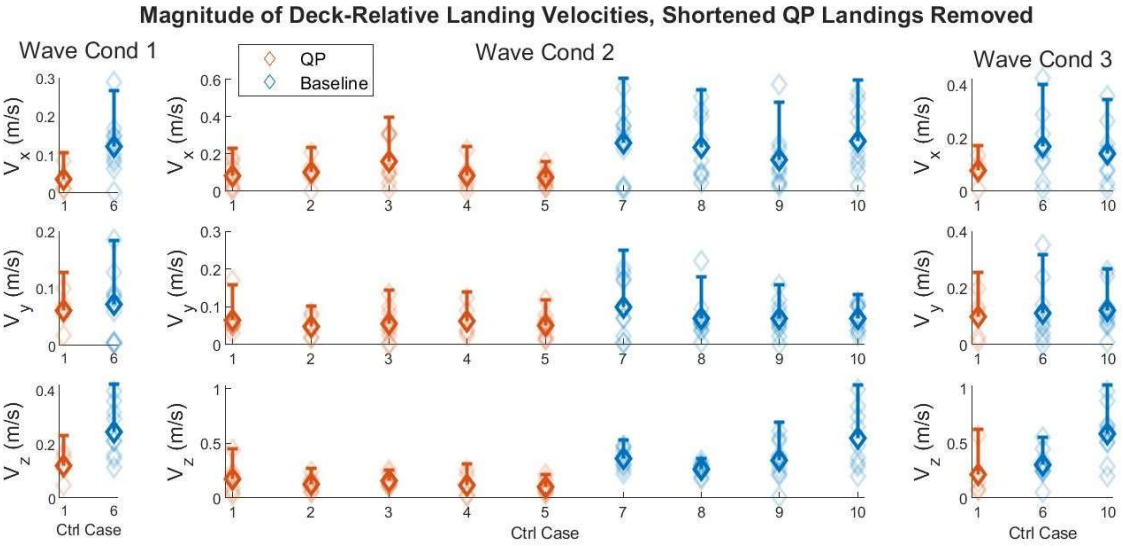


Figure 33: (a) Deck velocity norm vs. deck-relative descent rate at landing with shortened QP flights removed. (b) Predicted deck altitude at landing during final 3 seconds of all QP flights.

The land time update scheme also negatively impacted QP algorithm attitude errors at touchdown. This can be observed by comparing figure 34a and figure 31. In figure 34a, the shortened QP flights are removed and all QP landings with a pitch attitude error greater than about 7.5 degrees are removed. The reason for this is that the slightly shortened land time requires a more aggressive trajectory to be planned, resulting in the UAV flaring pitch attitude at the end of the trajectory to match deck surge velocity. The flared pitch attitudes often resulted in the rear landing gear contacting the deck, increasing the relative pitch attitude at touchdown. This also inflates the velocity errors at touchdown, which is clearly seen by comparing figure 34b and figure 30. With the shortened QP landings removed, the cases with the largest X and Z velocity errors are removed. These results are interesting, as a major point of concern prior to testing was the fidelity of the deck motion predictions. In the vast majority of cases, however, poor deck motion predictions proved not to be the culprit of poor performance.



(a) Deck-relative landing attitude.



(b) Deck relative landing velocities.

Figure 34: Deck-relative landing velocity and attitude with shortened QP flights removed. The faded markers are values from individual landings, the bold markers show the average value, and the one-sided error bar represents 2σ of the deck-relative error magnitude.

3. Findings and Conclusions

The first two-week test period in the MASK facility focused on the evaluation of the vision-based state estimation system. As part of this effort, a monocular camera-based deck state estimation algorithm was developed and demonstrated in model-scale autonomous landing experiments. The estimation algorithm utilizes an unscented Kalman filter to fuse IMU measurements with measurements of the pose of a recursive array of ApriTag fiducial markers. During these tests, the estimation scheme was also integrated with the baseline trajectory generation algorithm with the

purpose of validating the EMF control laws that have been developed while testing the use of the deck state estimator in the control loop. Fifteen successful autonomous landings were performed with the vision-based estimates used in the control loop. The results showed the UKF estimator performs well and can provide precise deck-relative state estimates that enable autonomous shipboard landings. The estimator performance could be improved through further optimized measurement and process noises, however.

The second test period in the MASK focused on the evaluation of advanced guidance algorithms for shipboard landings. As part of this effort, 149 autonomous ship landings were performed with two separate guidance algorithms: a baseline algorithm that commands a simple deck-relative path, and a quadratic programming based algorithm that plans the landing path to a predicted future deck state. Both guidance algorithms provided inertial position and heading commands to a Froude scaled EMF control law, and the control laws were used to perform tests with progressively reduced reference tracking bandwidths and jerk constraints. Based on the results of these experiments, the following observations and conclusions can be made:

1. The baseline "deck tracking" landing strategy is both simple and effective when the UAV has the bandwidth required to track deck motions. This is supported by the low average velocity errors obtained with the baseline algorithm in high- bandwidth configurations. The deck tracking method is sensitive to additional sources of lag, however.

2. The inclusion of deck state predictions and the ability to plan for the system dynamics allowed the QP algorithm to plan more direct landing paths, and also to perform landings with significantly lower tracking bandwidths than the baseline algorithm. But, in order to better understand the ability of the QP algorithm to plan for more restrictive acceleration and jerk constraints, additional testing with more stringent output constraints would be required.

3. Long term deck state predictions produced by the AR models were often unreliable, with the predictions typically beginning to converge toward the true deck state with 1.5 seconds or less remaining in the landing. Additionally, the velocity predictions often exhibited intermittent points with spikes in the prediction errors. This was due to intermittent buffering issues that occurred when sending ship measurement data over WiFi to the UAV.

4. Despite poor long term deck state predictions, the QP landing algorithm was able to match position and velocity well in the majority of cases. There were, however, several outlier cases with deck-relative landing velocities greater than 0.5 m/s. Additionally, there were a handful of cases where the QP algorithm led to deck-relative pitch angles of greater than 10 degrees at landing.

5. It was hypothesized that robustness to poor deck state predictions would be a limiting factor for the QP algorithm. These results, however, indicate that the land time update scheme was the culprit in the majority of cases where the QP algorithm showed poor performance. Removing cases where the maneuver duration was slightly decreased was found to remove almost all instances where a high deck-relative pitch attitude or velocity occurred at landing. The one exception to this was a case where deck altitude prediction errors were exceedingly poor during the last 0.5 seconds of the maneuver, and it is believed that this

was related to data buffering issues that occurred occasionally during testing. These results give confidence in the feasibility of incorporating deck motion predictions directly in path planning, provided the deck state measurements used in the prediction scheme are not low quality.

4. Transitions and Impacts

The development of deck state estimation, trajectory generation, and control algorithms provides an opportunity for technology transition to our partners at the U.S. Naval Surface Warfare Center Carderock Division for use in autonomous shipboard landing solutions.

5. Collaborations

We are working this project in collaboration with U.S. Naval Surface Warfare Center Carderock Division, Sea-Based Aviation and Aeromechanics Branch, Code 882. Dr. Anish Sydney is our main point of contact. Experimental testing was conducted at the Maneuvering and Sea Keeping (MASK) basin at the NSWCCD.

6. Personnel

Principal investigators:

Joseph F. Horn, 1.5 person months
Not a National Academy Member

Jack W. Langelaan, 1.5 person months
Not a National Academy Member

7. Students

Duncan Nicholson, M.S. Student (graduated August 2022), 24 person months

Christopher Hendrick, Ph.D. Student, 28.5 person months

Emma Jaques, M.S. Student, 21 person months

8. Technology Transfer

None to report.

9. Products, Publications, Patents, License Agreements, etc.

Conference Publications:

1. Christopher M. Hendrick et al. “Scaled Experiments in Flight Control Design for Autonomous Landing in High Sea States”. In: AIAA AVIATION 2022 Forum. DOI: 10.2514/6.2022- 3280. eprint: https://arc.aiaa.org/doi/pdf/10.2514/6.2022-3280. URL: https://arc.aiaa.org/doi/abs/10.2514/6.2022-3280.

2. Nicholson, D. V., Hendrick, C. M., Jaques, E. R., Langelaan, J. W., Horn, J. F., and Sydney, A. J., "Scaled Experiments in Estimation for Autonomous Landing in High Sea States," AIAA Aviation 2022 Forum, AIAA, Chicago, Illinois, 2022.
3. Hendrick, C. M., and Horn, J. F., "Analysis of Flight Control and Trajectory Planning for Autonomous Ship Landing Using Small-Scale UAVs," 48th European Rotorcraft Forum, ERF 2022, Winterthur, Switzerland, September 6-8, 2022.
4. Hendrick, C.M, Jaques, E. R., Langelaan, J. W., and Horn, J. F. "Evaluation of Autonomous Ship Landing Systems at the Maneuvering and Seakeeping Basin," Vertical Flight Society 79th Annual Forum, West Palm Beach, Florida, May 2023.

Journal Papers

1. Hendrick, C. M., and Horn, J. F., "Scalable Guidance and Control Laws for Model-Scale Analysis of Autonomous Ship Landing Systems," submitted to *CAES Aeronautical Journal*, currently under review.

Plan to submit an updated version of "Evaluation of Autonomous Ship Landing Systems at the Maneuvering and Seakeeping Basin" to the Journal of the American Helicopter Society.

MS Thesis

Title: Scaled Experiments in Vision-Based Approach and Landing in High Sea States
Author: Duncan Nicholson
Institution: The Pennsylvania State University
Date Completed: May 2022
Acknowledgement of Federal Support: Yes

Expect another Ph.D thesis by Christopher Hendrick by December 2023 and another M.S. thesis by Emma Jaques by December 2023.

10. Point of Contact in Navy

Dr. Anish Sydney
Naval Surface Warfare Center Carderock Division
Sea-Based Aviation and Aeromechanics Branch, Code 882
301-227-1482

Date of Last Contact: April 10, 2023

References

[1] Holmes, W. "Vision-Based Relative Deck State Estimation Used with Tau Based Landings", M.S. thesis, Pennsylvania State University. State College, PA 2017.

[2] Christopher M. Hendrick et al. "Scaled Experiments in Flight Control Design for Autonomous Landing in High Sea States". In: AIAA AVIATION 2022 Forum. DOI: 10.2514/6.2022- 3280. eprint:https://arc.aiaa.org/doi/pdf/10.2514/6.2022-3280. URL:https://arc.aiaa.org/doi/abs/10.2514/6.2022-3280.

[3] Christopher M. Hendrick and Joseph Horn. “Analysis of Flight Control and Trajectory Planning for Autonomous Ship Landings Using Small-Scale UAVs”. In: 48th European Rotorcraft Forum, ERF 2022. Zurich University of Applied Sciences (ZHAW), 2022.