CHALMERS
UNIVERSITY OF TECHNOLOGY

Autonomous Fixed-Wing Drone Landing on Maritime Vessels

Development of guidance and navigation system enabling autonomous landing of fixed-winged drones on maritime vessels

Master's thesis in Complex Adaptive Systems

Ludvig Möller
William Almqvist

DEPARTMENT OF ELECTRICAL ENGINEERING E2

CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2026
www.chalmers.se

Master's thesis 2026

Autonomous Fixed-Wing Drone Landing on Maritime Vessels

Development of guidance and navigation system enabling autonomous landing of fixed-winged drones on maritime vessels

Ludvig Möller
William Almqvist

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Electrical Engineering
Division of Systems and Control
Chalmers University of Technology
Gothenburg, Sweden 2026

Autonomous Fixed-Wing Drone Landing on Maritime Vessels
Development of guidance and navigation system enabling autonomous landing of fixed-winged drones on maritime vessels
LUDVIG MÖLLER
WILLIAM ALMQVIST

Supervisor: Fredrik Falkman, Swedish Sea Rescue Society
Examiner: Jonas Sjöberg, Systems and Control - Electrical Engineering

Master's Thesis 2026
Department of Electrical Engineering
Division of Systems and Control
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Typeset in LaTeX
Printed by Chalmers Reproservice
Gothenburg, Sweden 2026

Autonomous fixed-wing drone landing on maritime vessels
Development of guidance and navigation system enabling autonomous landing of fixed-wing drones on maritime vessels
LUDVIG MÖLLER, WILLIAM ALMQVIST
Department of Electrical Engineering E2
Chalmers University of Technology

Abstract

To improve the effectiveness of maritime search and rescue (SAR) operations, the Swedish Sea Rescue Society (SSRS) is developing autonomous fixed-wing drones through its Eyes-On-Scene project. While these drones can autonomously launch and loiter over areas of interest, they currently lack automated recovery capabilities.

This thesis presents a guidance and navigation framework that enables fixed-wing drones to autonomously land on moving SSRS rescue vessels without requiring modifications to either platform. The system integrates AIS and camera-based tracking through an Extended Kalman Filter to estimate vessel position. A multi-stage landing algorithm is introduced, consisting of follow, descent, and diversion phases, designed to ensure safety under uncertain conditions. The algorithm enables controlled landing velocities and supports flexible approach angles for robust operations.

The complete system is tested in a simulated environment under various disturbances, including wind, position, altitude, and speed fluctuations. Simulation results demonstrate high landing accuracy, indicating the feasibility of autonomous landings in realistic SAR scenarios. The results highlight the potential to strengthen SSRS's autonomous drone capabilities and contribute toward more efficient recovery in future SAR operations.

Keywords: autonomous landing, fixed-wing drones, maritime search and rescue, SSRS, Eyes-On-Scene, AIS tracking, camera-based tracking, GNC, autonomous recovery, UAV

Acknowledgements

We would like to express our sincere gratitude to our supervisors, Fredrik Falkman and Alexander Sandström at SSRS, for their invaluable guidance and support throughout this thesis. We are also grateful to Professor Jonas Sjöberg for serving as our examiner and for providing insightful feedback that greatly improved the quality of this work.

Ludvig Möller, William Almqvist, Gothenburg, June 2025

List of Acronyms

Below is the list of acronyms that have been used throughout this thesis, listed in alphabetical order:

Acronym	**Description**
API	Application Programming Interface
AIS	Automatic Identification System
EKF	Extended Kalman Filter
EPP	Expanded Polypropylene
EOS	Eyes On Scene
GNC	Guidance, Navigation and Control
GNSS	Global Navigation Satellite System
GPS	Global Positioning System
KF	Kalman Filter
IMU	Inertial Measurement Unit
NED	North - East - Down
LOS	Line Of Sight
VHF	Very High Frequency radio
RMSE	Root Mean Squared Error
SAR	Search And Rescue
SSRS	Swedish Sea Rescue Society
TECS	Total Energy Control System
UAV	Unmanned Aerial Vehicle
UTC	Coordinated Universal Time

Nomenclature

Below is the nomenclature of indices, sets, parameters, and variables that have been used throughout this thesis.

Indices

Symbol	Description
s	Index denoting ship
d	Index denoting drone
L	Index denoting lookahead variable
t	Index for time step
$\{n\}$	Index denoting NED coordinate system
$\{b\}$	Index denoting body-fixed coordinate system
$\{s\}$	Index denoting seakeeping coordinate system
$\{o_a\}$	Origin of coordinate system $\{a\}$
$R_{x,\alpha}$	Simple rotational matrix, rotating any angle α around given axis x
R_b^a	Rotation matrix transforming coordinate vectors in $\{b\}$ to coordinate vectors in $\{a\}$

State Variables

Symbol	Unit	Description
z	m	Altitude of the drone
u	m/s	Surge velocity (forward, along x_b)
v	m/s	Sway velocity (sideways, along y_b)
w	m/s	Heave velocity (vertical, along z_b)
p	rad/s	Roll rate (rotation about x_b)
q	rad/s	Pitch rate (rotation about y_b)

r	rad/s	Yaw rate (rotation about z_b)
v_x	m/s	Speed of object x
ε_x	degree	Course of object x relative to north
ψ_x	degree	Heading of object x relative to north
ϕ	degree	Roll angle of drone
θ	degree	Pitch angle of drone
α	degree	Angle of attack
β	degree	Sideslip (drift) angle
Υ	degree	Flight path angle of drone
δ_t	–	Throttle control command (normalized 0–1)
δ_e	degree	Elevator control surface deflection
η	degree	Glide slope angle

Parameters

Symbol	**Unit**	**Description**
Gr	–	Glide ratio (unitless)
d_L	m	Look-ahead distance
t_L	s	Look-ahead time
R	m	Radius of the Earth
K_{speed}	-	Gain for speed controller (unitless)
$K_{altitude}$	-	Gain for altitude controller (unitless)

Variables

Symbol	**Unit**	**Description**
S_x	m/s, deg	Speed (m/s) and heading (deg) of object x
P_x	deg, m	Position of any point x (latitude, longitude in deg, altitude in m)
P_d	deg, m	Drone's position (latitude, longitude, altitude)
P_s	deg, m	Ship's position (latitude, longitude, altitude)
P_1	deg, m	Coordinate for the velocity setpoint when following ship
P_2	deg, m	Coordinate where the drone should start descent

P_3	deg, m	Landing point
$\hat{P}_{drone}$	deg, m	Drone's estimated position from Kalman filter
$\hat{P}_{boat}$	deg, m	Boat's estimated position from Kalman filter
D_{x-y}	m	Distance between two coordinates x and y
Q	–	Process noise covariance matrix
R	–	Measurement noise covariance matrix
K	–	Kalman gain

Disturbances

Symbol	**Description**
$w(t)$	Environmental disturbances
$n(t)$	Measurement disturbances

Contents

List of Figures

1
Introduction

This introduction provides an overview of the context, contributions, and structure of the thesis. Section 1.1 introduces the background and motivation behind the problem addressed. Section 1.2 outlines the key contributions of this work. Section 1.3 presents a high-level overview of the proposed solution. Section 1.4 discusses ethical and sustainability considerations relevant to the project. Finally, Section 1.5 describes the overall structure of the thesis.

1.1 Background

The Swedish Sea Rescue Society (SSRS) is a voluntary organization conducting search and rescue (SAR) operations in Swedish coastal waters. In 2024, SSRS volunteers responded to 6,426 calls, an increase from the previous year [1]. To improve the efficacy of these missions, SSRS has been developing the Eyes-On-Scene (EOS) project. This initiative aims to deploy fixed-wing drones (fig. 1.1) as first responders that autonomously carry out reconnaissance missions. These missions involve launching the drone, navigating to a rescue location, observing the scene while loitering, transmitting live video back to the rescue crew and finally returning to land to conclude the mission. By providing early visual insight into the situation, the system can support more informed decisions about which vessel and equipment to deploy, potentially improving the outcome of a mission.

The EOS drones are currently capable of autonomous launch and loitering around designated areas. However, long missions or distant locations can result in insufficient battery charge to return to shore, requiring the drone to ditch in the sea for later retrieval. Since the EOS project's long-term goal is to minimize human intervention, enabling the drone to autonomously land on an SSRS vessel would streamline recovery, reduce workload, and enhance operational safety, particularly in rough sea conditions.

To facilitate autonomous recovery, this thesis proposes adding the capability for fixed-wing drones to land on moving SSRS boats. If landing can occur while the boat is en route back to shore, it would further reduce the operational impact of using drones. However, landing a drone on a moving vessel presents several challenges. The drone's minimum airspeed is approximately 12 m/s, and SSRS recommends a maximum allowed impact speed of 3 m/s to avoid damage or injury. This implies that the boat must travel at least 9 m/s relative to the wind. Moreover, SSRS requires that its vessels not be modified with additional hardware such as landing

nets or tracking devices. A standard rescue boat measures 8x4 meters, with a wheelhouse 2.5 meters high and a stern deck of 2x4 meters, meaning the drone must achieve a landing accuracy of around 1.25 meters in both lateral and vertical dimensions. Adverse weather conditions are common during rescue operations and add further complexity.

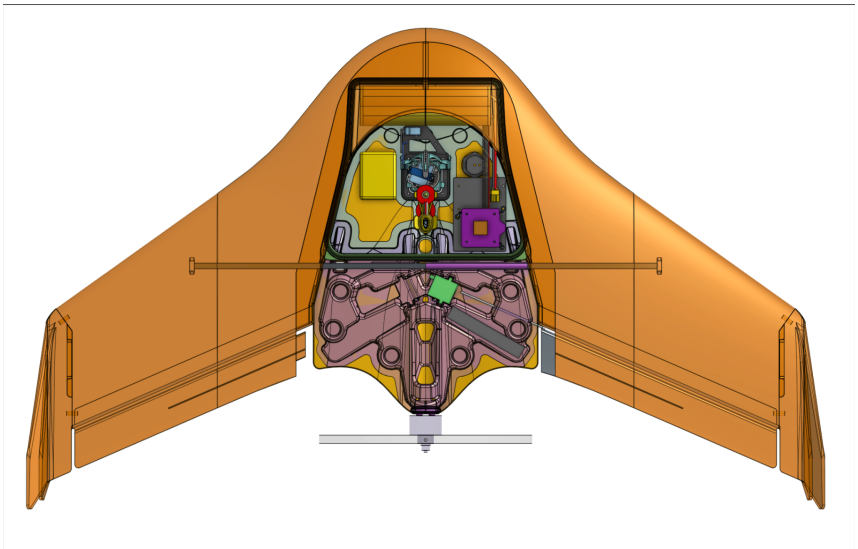


Figure 1.1: See-through image of the EOS drone

Ensuring safe operation of autonomous drones also involves legal and procedural considerations. Current regulations mandate a designated drone operator responsible for overseeing the mission. During the operation, this person monitors the drone's position and behavior and issues high-level commands as necessary.

1.2 Contributions

This thesis advances the autonomous capabilities of SSRS drones by enabling fixed-wing UAVs to land on moving boats autonomously. The primary contributions of this thesis are:

- **Autonomous landing strategy.** A landing procedure is developed, enabling fixed-wing UAVs to approach and land on a moving vessel with minimal human intervention.
- **Target tracking using sensor fusion.** An Extended Kalman Filter is implemented to estimate the boat's position and velocity by fusing AIS and camera-based detections, enabling allocentric target tracking.
- **Guidance for dynamic target following.** A guidance scheme is designed to maintain the UAV in proximity to the boat during approach, ensuring readiness to initiate landing.
- **Landing trajectory generation and guidance.** A trajectory planning system is implemented to compute interception points and guide the UAV through a controlled descent with respect to the boat's motion.

- **Validation through simulation.** The system is evaluated in simulation under varying wind conditions and boat trajectories. Performance is measured by landing accuracy and reliability.

1.3 System Overview

This section provides an overview of the general control flow involved in the drone's operation during the final phase of the reconnaissance mission. It begins by introducing the concept of Guidance, Navigation and Control (GNC) followed by a high-level mission flow chart that illustrates how commands from the drone operator are processed.

1.3.1 Guidance, Navigation and Control (GNC)

This thesis employs Guidance, Navigation, and Control (GNC) system which are commonly found in flight computers to manage vehicle positioning, trajectory planning, and control [2]. The following figure shows an overview of the feedback loop of this project, where GNC is a central subsystem.

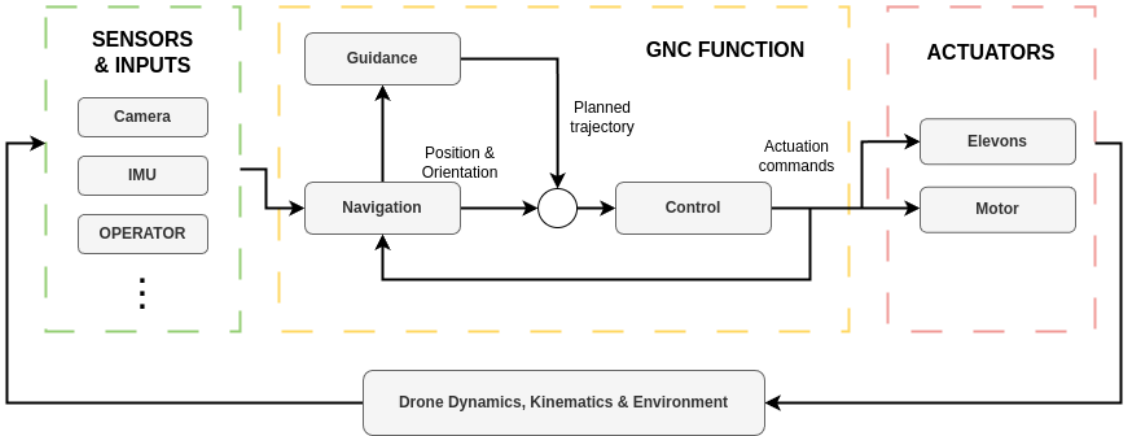


Figure 1.2: Simplified GNC feedback loop implemented in this thesis. Specifically, this thesis implements the *Guidance* and *Navigation* blocks as well as the processing of drone operator inputs.

Navigation is tasked with establishing where the craft is in relation to outer references, and determining what speed, position, and orientation (attitude) it currently has. It does so by processing sensor data through Kalman filters. The guidance system then takes this information and outputs a target trajectory toward the intended destination while considering vehicle dynamics, environmental factors, and mission constraints. The outputs from navigation and guidance are forwarded to the control system, which translates them into actuator commands to adjust control surfaces and throttle settings. The control system is also responsible for counteracting disturbances and ensuring stable handling. The responsibility of the *Camera* block from Figure 1.2 is to estimate the boat's position using a camera. This functionality is implemented in another, parallel master thesis.

1.3.2 Reconnaissance Mission Overview

The reconnaissance mission can be split up into three main parts; Launch, SAR and Recovery. As stated, this thesis focuses exclusively on the final recovery phase of the mission where the drone should autonomously land on a moving boat. Figure 1.3 shows the internal logic used by the *Guidance* block from Figure 1.2. This logic determines which high-level flight stage should be active at each moment.

The "end of mission sequence" begins when the operator issues a command to enter the *follow* stage, during which the drone tracks the boat's motion while awaiting further input. Once the drone is in position behind the boat the drone operator may initiate the *landing* stage, during which the drone will calculate trajectories to execute a landing maneuver. If at any point, the conditions are deemed too unsafe to continue, the drone will automatically go into the *diversion* stage to return to a safe position. All three of these stages are explained in more detail in Chapter 4.

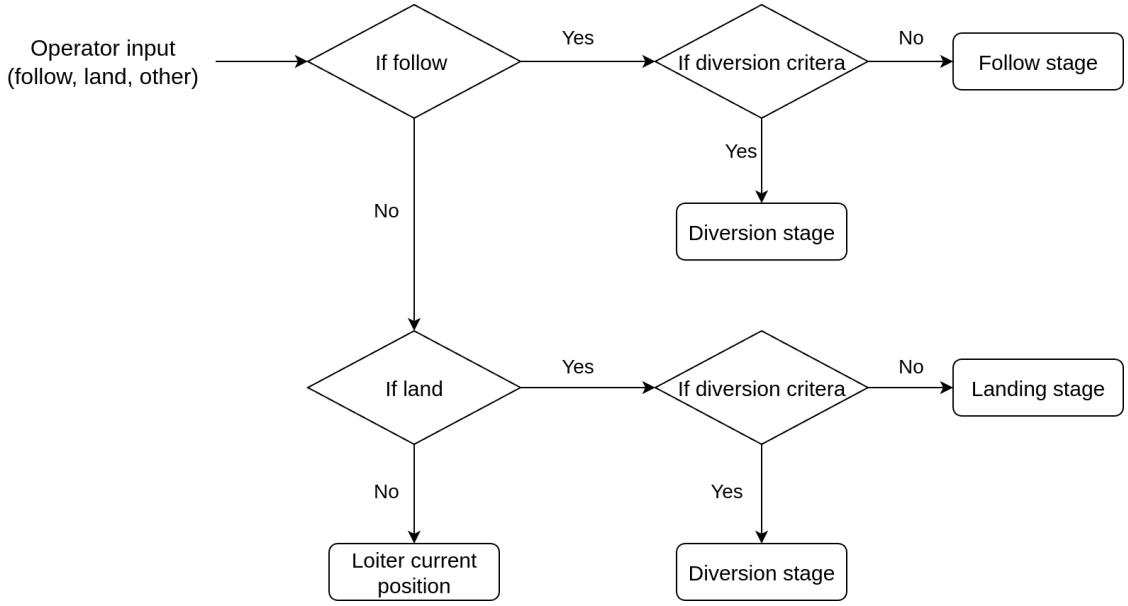


Figure 1.3: Flow chart of how the mission handler decides and keeps track of the current mission objective.

1.4 Ethics and Sustainability

SSRS is involved in 90% of all rescue operations in Swedish coastal waters. They are entirely funded by donations and have a total of 144,000 members as of 2024, contributing to their mission of making the Swedish coastal waters safer for everyone. Further developing their technical capabilities actively contributes to this pursuit.

Enabling UAVs to land in highly adverse locations, such as a moving vessel at sea, has broader applications than landing SAR UAVs. UAVs could autonomously deliver medical supplies, spare parts, or urgent goods to ships in transit, reducing the need for costly, less sustainable, and time-consuming docking. Drones could assist in

monitoring offshore wind farms and other maritime structures, reducing human risk and improving maintenance efficiency. Having another landing option can decrease the need to ditch a craft in the ocean and thus may prolong a unit's lifespan.

The technology could also be used in military applications with more questionable intent. The fundamental capability of landing a UAV on a moving platform has legitimate civilian and humanitarian uses. However, it could also be utilized for purposes such as autonomous weapon delivery, surveillance of adversaries, or intelligence gathering in contested areas.

The ethical implications depend on how the technology is employed. While it does not introduce new capabilities, it may lower the barrier for autonomous UAV operations in dynamic environments, potentially making certain military applications more accessible. This highlights the importance of responsible development and governance to ensure that the technology is primarily used for constructive and life-saving purposes.

1.5 Thesis Organization

The remainder of this thesis is structured as follows:

- **Chapter 2** provides an overview of the hardware and software used throughout the project.
- **Chapter 3** describes the proposed solution and system design for guidance and navigation. Preliminary results from each subsystem are also presented.
- **Chapter 4** details the simulation environment and testing procedures, including various test scenarios used to evaluate the system's robustness under adverse conditions.
- **Chapter 5** concludes the thesis with a discussion of key findings and offers recommendations for future development, testing and evaluation.

2
System Specifications

This chapter presents the pre-existing hardware and software used in this thesis. It begins by outlining the hardware that the drone is running, section 2.1, then the autopilot software, section 2.2, and finally the boat localization software, section 2.3 and 2.4.

2.1 EOS Hardware Specification

The hardware specifications for the EOS drone are the following:

- Expanding polypropylene (EPP) foam body construction
- Flight controller Control Zero H7 OEM G with built-in IMU and gyro. Running Ardupilot software.
- mRo SAM GPS receiver
- Magnetometer
- Raspberry Pi 4 companion computer
- 4G cellular connection
- remote.aero software for operator interface

Notable are the two different compute units, the Control Zero and the Raspberry Pi. The former is running the control software, Ardupilot [3], natively and has a carrier board that has inputs for sensors and outputs for control surfaces and motor controllers. The Raspberry Pi acts as a companion computer. It's running Linux and has higher computational capacity than the Control Zero. All the systems in this thesis are designed to be implemented on the Raspberry Pi. It also handles commands and communication with ground based systems, such as communication with the drone operator.

2.2 Ardupilot

This thesis will utilize Ardupilot, an open-source autopilot software that runs natively on the Control Zero flight controller of the EOS drone. It has a robust navigation system, utilizing an Extended Kalman filter (EKF) that can approximate the position, attitude, speed, and windspeed among others [4]. It has a control system that can take waypoints and navigate between them. Further, it can automatically guide the drone to given altitudes and speeds. Depending on the chosen settings, Ardupilot's guidance system can plan and execute a variety of tasks [5]. Landing on a moving vessel with a fixed-wing drone is however not one of the tasks currently supported by Ardupilots guidance system. Ardupilot can communicate with other

compute units via the communication protocol mavlink and thus offload certain tasks to another computer [6]. The companion computer of the EOS, a Raspberry Pi, will thus act as the guidance system by taking positional information from the navigation system in Ardupilot and calculating a suitable path. It then sends this path back to the control system in Ardupilot.

2.2.1 Control - L1 & Total Energy Control System (TECS)

ArduPilot employs an L1 nonlinear lateral guidance controller, primarily designed for fixed-wing aircraft, to follow predefined trajectories. The L1 controller calculates the lateral acceleration required to steer the drone towards the target trajectory by minimizing the cross-track error. This strategy ensures smooth turns by aliging the drones velocity vector with the target trajectory. A detailed explanation of the L1 algorithms and implementation can be found in [7].

For longitudinal control, ArduPilot uses the Total Energy Control System (TECS), which regulates both airspeed (kinetic energy) and altitude (potential energy) as they depend on each other. TECS computes the appropriate pitch and throttle commands to maintain the desired energy state. It comes with user-configurable parameters to change the behavior of TECS like weighing speed and altitude differently. Further technical details and tunable parameters are documented in [8].

2.3 Automatic Identification System (AIS)

The Automatic Identification System (AIS) is a communication system designed for automated tracking of vessels at sea. It is one of two systems used to track the target vessel's position in this thesis. AIS relies on transceivers that broadcast status information over a designated VHF radio band or via satellite communication, while simultaneously receiving transmissions from other nearby transceivers [9]. Originally developed as a supplement to marine radar, AIS now serves a wide range of purposes focused on monitoring the position and status of maritime vessels. There are two main classes of AIS transceivers: Class A and Class B, each with different capabilities. All SSRS vessels are equipped with Class A transceivers, which, in addition to standard AIS messages, can also transmit Search and Rescue (SAR) status messages.

The United States Coast Guard provides specifications for the various AIS message

types [10]. The messages relevant to this thesis include:

- **Message ID** – Identifies the message type.
- **SOG (Speed Over Ground)** – Indicates the vessel's speed relative to the ground.
- **Position Accuracy** – Indicates the accuracy of the positional report.
- **Longitude** – Vessel's reported longitude, typically with six decimal places.
- **Latitude** – Vessel's reported latitude, typically with six decimal places.
- **COG (Course Over Ground)** – The vessel's direction of travel.
- **True Heading** – The vessel's compass heading (direction the bow is pointing).
- **Time stamp** – UTC in whole seconds when the report was generated.

The accuracy of AIS measurements depends on the underlying sensors, which are not specified by the AIS standard and can therefore vary between devices [11]. Many transceivers apply internal processing to improve positional estimates. The Position Accuracy flag indicates whether the positional error is below or above 10 meters. The transmission frequency varies based on both the transceiver configuration and vessel speed, typically ranging from every 2 to 12 seconds [12].

In addition to shipborne transceivers, AIS infrastructure includes shore-based base stations equipped with both transmission and reception capabilities. Unlike mobile units, base stations have advanced capabilities defined in the AIS standards that enable them to perform system-level management tasks. Base stations are also able to request status reports from individual transceivers. SSRS have deployed their own base stations and have access to relatively low-latency information via an API call. Despite this, there is still a delay of 2-4 seconds from message generation until it's available via API.

2.4 Camera-Based Boat Detection

The drone is also fitted with a gimbal camera whose main purpose is to survey the designated area for a reconnaissance mission. This camera however, can also be used to help localizing the boat by giving a distance in x,y and z to the boat. This information allows the drone to follow and land on the boat without direct access to the boat's position.

3

Coordinate Systems and Kinematics

This chapter outlines the reference frames used throughout this thesis and establishes the mathematical relations needed to describe motion and transform between frames. Section 3.1 introduces the coordinate systems, while Section 3.2 presents the kinematic relations.

3.1 Reference Frames Overview

To model the motion of marine vessels and aerial vehicles, multiple coordinate frames are used. Figure 3.1 shows the two primary frames: the **North-East-Down (NED)** frame used for global positioning and the **Body-fixed** frame for motion and orientation modeling.

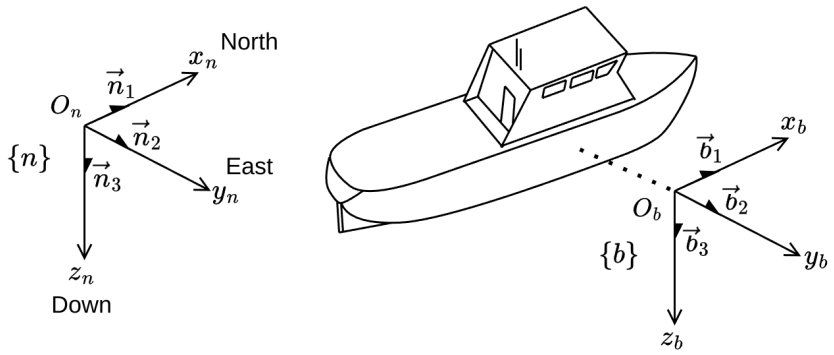


Figure 3.1: Body fix and local coordinate systems

North-East-Down (NED) Frame $\{n\}$ The NED frame is an Earth-fixed coordinate system defined relative to a local tangent plane of the Earth's surface. This is typically established using a global geodetic reference such as WGS 84 [13]. The origin of the frame is chosen as a fixed point, often near the vessel or region of operation. Its unit vectors are:

- $\vec{n}_1$ – North: tangential to Earth's surface, pointing towards geographic North.
- $\vec{n}_2$ – East: tangential to Earth's surface, orthogonal to North, pointing East.
- $\vec{n}_3$ – Down: normal to Earth's ellipsoid surface, pointing downward.

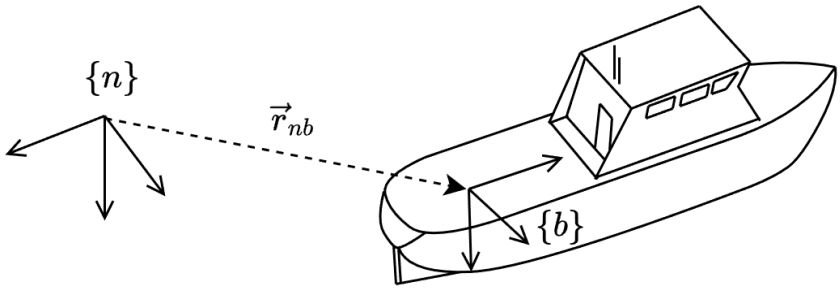


Figure 3.2: Relation between local and body-fixed coordinate systems

Body-Fixed Frame $\{b\}$ The body-fixed frame is attached to the moving object (ship or drone), rotating and translating with it. Its origin, o_b, is typically placed at the geometrical center or center of mass. The axes are commonly defined as:

- $\vec{b}_1$: Forward
- $\vec{b}_2$: Starboard (right)
- $\vec{b}_3$: Downward

The position of the body frame ($\{b\}$) in the NED system ($\{n\}$) is given by the translation vector $\vec{r}_{nb}$ as shown in Figure 3.2.

The kinematic variables for all rigid bodies are illustrated in Figure 3.3. The notation is adapted from Zhang & Wang, with minor modifications in accordance with Fossen [14], [15]. The same notation is used across all vehicles, with subscripts d and s denoting the local body-fixed frames b of the drone and ship, respectively.

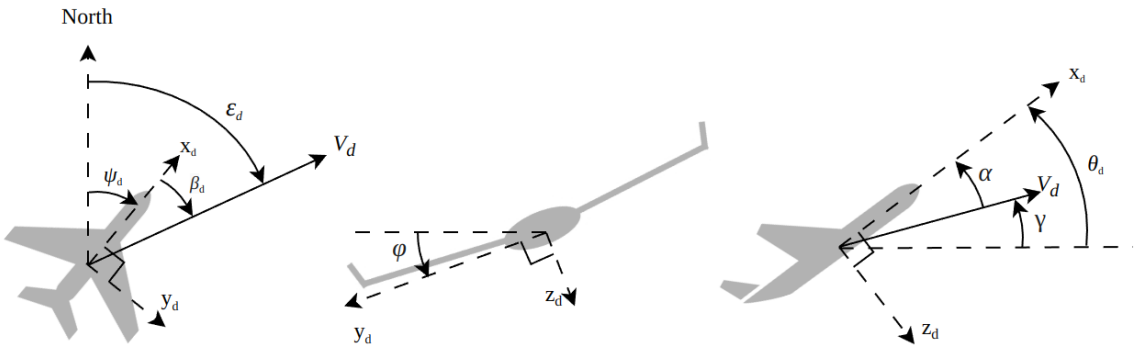


Figure 3.3: Kinematic variables in a body-fixed reference frame $\{b\}$ for Rigid-Body Vehicles.

3.2 Kinematic Quantities and Rotational Transformations

The motion of a vehicle is described by its translational, rotational and angular velocities in the body frame. The notation for this is described in detail below. The conversion between geodetic position and local frame distance is also covered.

3.2.1 Orientation and Rotation Matrices

The orientation of the body with respect to the NED frame is defined using the Euler angles.

- ϕ: Roll angle (tilt left/right)
- θ: Pitch angle (nose up/down)
- ψ: Yaw angle (heading angle from North)

Transforming coordinate vectors from one frame to another can be done with a rotational matrix $\mathbf{R}$. A simple rotation concerns rotation around only one axis and if applied on the NED frame, the rotation into any other frame can be computed [16].

- Rotation of an angle ϕ around the x axis

$$\mathbf{R}_{x,\phi} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix} \tag{3.1}$$

- Rotation of an angle θ around the y axis

$$\mathbf{R}_{y,\theta} = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \tag{3.2}$$

- Rotation of an angle ψ around the z axis

$$\mathbf{R}_{z,\psi} = \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.3}$$

Additional angles used to describe the relative direction of the velocity vector are:

- α: Angle of attack (between x_b and the projection of velocity in the xz-plane)
- β: Sideslip angle (between x_b and velocity projection in the xy-plane)
- γ: Flight path angle (between velocity vector and horizontal plane)
- ϵ: Course angle (velocity direction relative to North)

3.2.2 Translational and Angular Velocities

The linear velocity vector of the body in its own coordinate system is given by:

$$\vec{v}_b = \begin{bmatrix} u & v & w \end{bmatrix}^T \tag{3.4}$$

where:

- u: Surge (forward velocity, along x_b)
- v: Sway (sideways velocity, along y_b)
- w: Heave (vertical velocity, along z_b)

The body's rotation is expressed in terms of angular velocities:

$$\vec{\omega}_b = \begin{bmatrix} p & q & r \end{bmatrix}^T \tag{3.5}$$

where:

- p: Roll rate (rotation about x_b)
- q: Pitch rate (rotation about y_b)
- r: Yaw rate (rotation about z_b)

These variables together provide a full description of the six degrees of freedom of motion, applicable to both surface and aerial vehicles.

3.2.3 Geodetic Distance Conversion - Equirectangular Projection

To analyze spatial relationships between locations, it is necessary to compute distances between geodetic coordinates and convert local frame offsets to global positions. This thesis employs the equirectangular projection for these tasks—a method that approximates the Earth as flat over short distances by projecting latitude and longitude onto a local Cartesian plane. It assumes no curvature to the Earth, leading to errors over larger distances. However, for short distances, it is a simple method with good enough accuracy.

The distance between two coordinates is approximated using the following equation:

$$\begin{aligned} \text{lat}_{\text{mean}} &= \frac{\text{lat}_1 + \text{lat}_2}{2} \\ d &= R\sqrt{(\Delta\text{lat})^2 + (\Delta\text{lon} \cdot \cos(\text{lat}_{\text{mean}}))^2} \end{aligned} \tag{3.6}$$

Calculating "look-ahead points", that is, points at a given distance in a given heading from the current position, is defined in the equation below:

$$\begin{aligned} \text{new lat} &= \text{lat} + \frac{d\cos(\psi)}{R} \\ \text{new lon} &= \text{lon} + \frac{d\sin(\psi)}{R\cos(\text{lat})} \end{aligned} \tag{3.7}$$

Where R is the radius of the Earth, and Δlon is multiplied by $\cos(\text{lat}_{\text{mean}})$ to account for the change in longitude distance with latitude.

4

GNC Implementation

This chapter covers the methods and procedures used to implement the GNC system developed in this thesis. Section 4.1 and 4.2 provide an overview of the overall GNC implementation and logic flow. Following that, Section 4.3 details the implementation of the *Navigation* system, which is responsible for estimating the positions of the drone and the boat. Finally, Section 4.4 describes the implementation of the *Guidance* system, which determines where the drone should fly based on current mission objectives.

4.1 GNC Overview

This thesis implements a Guidance, Navigation and Control (GNC) system, as explained in Section 1.3.1. A full system overview is shown in Figure 4.2. This control diagram shows how the signals are processed and what inputs and outputs each sub-component has, as well as what is used as the control feedback.

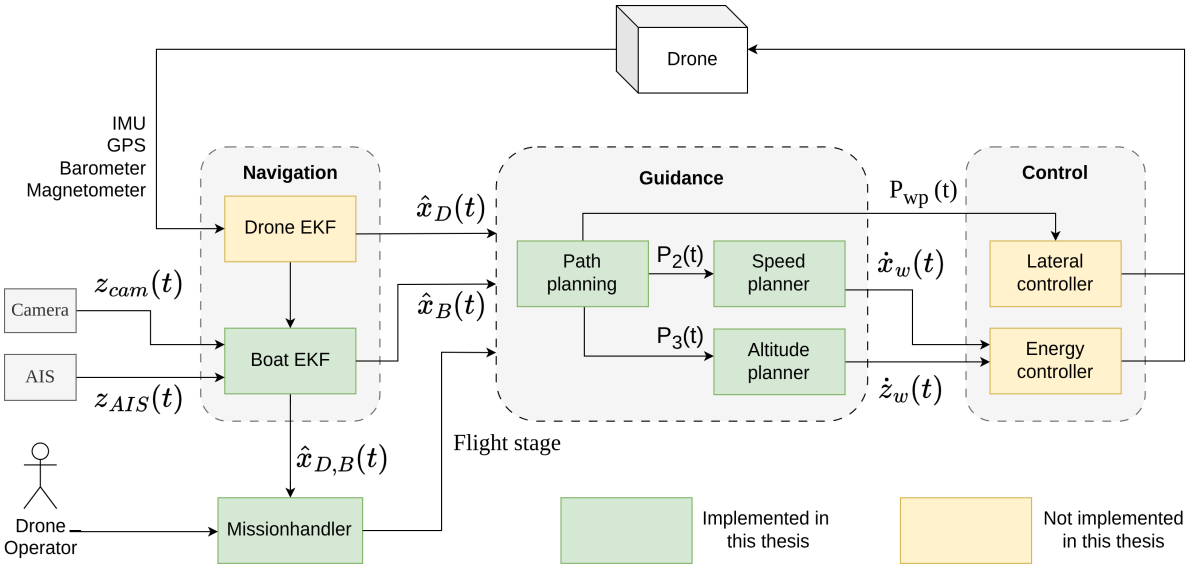


Figure 4.1: Detailed overview of the GNC implementation. Green boxes denote subsystems implemented in this thesis. Yellow boxes denote preexisting subsystems utilized in the GNC implementation.

4.2 Mission Handler

To ensure the correct execution of the landing algorithm, the system needs to be aware of its current flight stage. This coordination is handled by the mission handler which continuously communicates the current flight stage to the landing algorithm. It also decides whether diversions are required. Figure 4.2 presents a flow chart illustrating how the mission handler processes operator commands and manages transitions between flight stages. All terms in this figure are explained in detail in Section 4.4

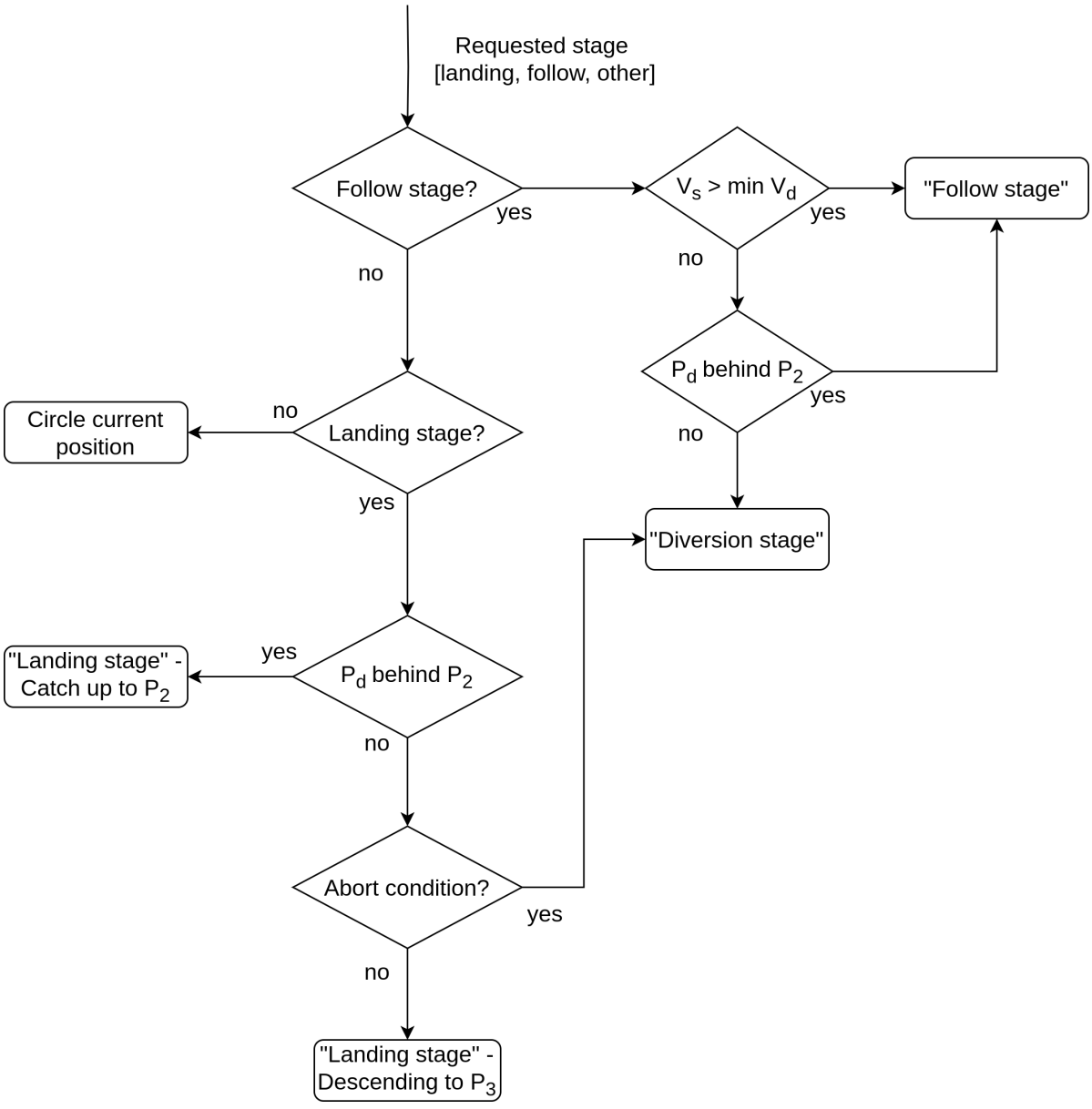


Figure 4.2: Detailed flow chart of how the mission handler decides and keeps track of the current objective

4.3 Navigation

The Navigation system is responsible for estimating the states of both the drone and the boat, as illustrated in Figure 4.3. Both estimators rely on EKFs, as described in Section A.3.

The boat's state estimation is based on measurements from AIS and the camera-based position estimates, as outlined in Chapter 2. Fusing these sources has the potential to produce a more accurate navigation solution than using either one independently. However, both sources are unsynchronized and subject to significant measurement delays. These challenges make Bayesian approaches, and Kalman filtering in particular, well-suited for this task, as discussed in Section A.

The following sections detail how the filter models the vessel dynamics, Section 4.3.1, how measurements are modeled, Section 4.3.2, and how uncertainties of the system are treated, Section 4.3.3. In addition, Section 4.3.4 describes how the filter handles the asynchronous and delayed measurements from the two different sources.

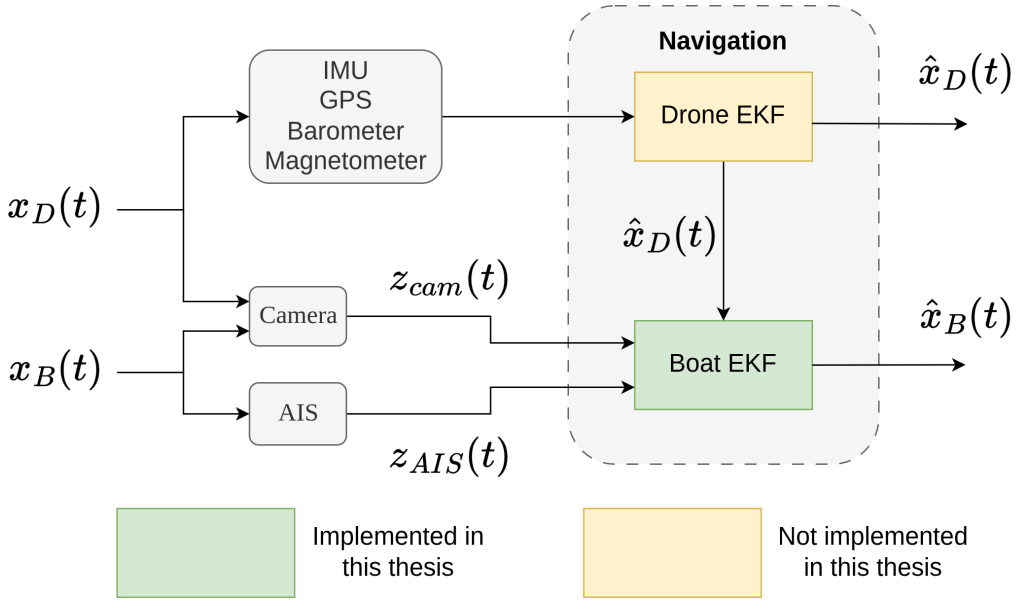


Figure 4.3: Overview of the navigation system. Sub-system of the larger system overview, Figure 4.2

4.3.1 System Modeling

An EKF is employed to track the vessel's planar motion in 2D, with a state vector incorporating position, heading, velocities, and accelerations. Motion and position not in the horizontal plane, such as roll, pitch, and heave, are assumed to have minimal influence on the planar dynamics. The motion is expressed in a body-fixed frame $\{b\}$ and referenced to a local NED frame $\{n\}$. These frames are consistent with the detailed description in Section 3.1. The state vector is defined as:

$$\mathbf{x} = [N, E, \psi, u, v, r, a_u, a_v]^T \tag{4.1}$$

Here:

- N, E: North and East positions in local frame, $\{n\}$
- ψ: yaw angle (heading), measured clockwise from magnetic north
- u, v: surge and sway velocities in body frame, $\{b\}$
- r: yaw rate
- a_u, a_v: surge and sway accelerations

Only the first derivative of each measurement is included to model dynamics while omitting higher-order terms like yaw acceleration and jerk. Similarly, since control inputs such as rudder angle or thrust are unobservable, their effects are implicitly captured by the process noise term.

The vessel's kinematics are expressed in a continuous-time nonlinear model as:

$$\begin{aligned} \dot{\vec{r}}_{nb} &= \mathbf{R}_{z,\psi}\vec{V}_d \\ \dot{\vec{V}}_d &= \vec{a}_d \end{aligned} \tag{4.2}$$

where:

- $\vec{r}_{nb} = [N, E]^T$ denotes the vessel's 2D position in the local NED frame $\{n\}$
- $\vec{V}_d = [u, v]^T$ is the velocity vector in the body frame $\{b\}$
- $\vec{a}_d = [a_u, a_v]^T$ is the acceleration vector in the body frame
- $\mathbf{R}_{z,\psi}$ is the yaw rotation matrix that transforms body-frame velocities to the NED frame as detailed in Section 3.2.

The continuous Equations in 4.2 can be discretized. A second-order Taylor series approximation is used to update the state at time k based on $\mathbf{x}_{k-1}$. For any state variable x, its Taylor expansion over time Δt is:

$$x(t + \Delta t) = x(t) + \dot{x}(t)\,\Delta t + \frac{1}{2}\ddot{x}(t)\,\Delta t^2 + \mathcal{O}(\Delta t^3) \tag{4.3}$$

Higher-order terms $\mathcal{O}(\Delta t^3)$ are omitted under the assumption that the third-order dynamics (jerk) are negligible and for the same observability issues present for $\dot{r}$.

Applying this to the position and heading terms leads to the discrete-time model:

$$
\begin{aligned}
N_k &= N_{k-1} + (u_{k-1}\cos\psi_{k-1} - v_{k-1}\sin\psi_{k-1})\,\Delta t \\
&\quad + \tfrac{1}{2}(a_{u,k-1}\cos\psi_{k-1} - a_{v,k-1}\sin\psi_{k-1})\,\Delta t^2 \\
E_k &= E_{k-1} + (u_{k-1}\sin\psi_{k-1} + v_{k-1}\cos\psi_{k-1})\,\Delta t \\
&\quad + \tfrac{1}{2}(a_{u,k-1}\sin\psi_{k-1} + a_{v,k-1}\cos\psi_{k-1})\,\Delta t^2 \\
\psi_k &= \psi_{k-1} + r_{k-1}\Delta t \\
u_k &= u_{k-1} + a_{u,k-1}\Delta t \\
v_k &= v_{k-1} + a_{v,k-1}\Delta t \\
r_k &= r_{k-1} \\
a_{u,k} &= a_{u,k-1} \\
a_{v,k} &= a_{v,k-1}
\end{aligned}
\tag{4.4}
$$

This formulation assumes constant acceleration over the time step Δt, which provides a more accurate position update than the Euler method, particularly when acceleration varies slowly or is small, which is common in naval applications [15]. The inclusion of the $\frac{1}{2}a\,\Delta t^2$ term aligns with Newtonian motion under constant acceleration and avoids the overestimation seen in simpler two-step update schemes.

The discrete-time model implicitly defines the nonlinear transition function $f(x_{k-1})$ used in the EKF prediction step detailed by Equation A.12. For linearization, the Jacobian of this function with respect to the state yields the system matrix A_k, which is computed at each time step as described in Equation A.13.

4.3.2 Measurement Modeling

This section defines how measurements from AIS and onboard cameras are incorporated into the EKF. All measurements are expressed in the local frame n with an origin fixed at a geodetic location, which simplifies fusion with the state vector by avoiding geodetic nonlinearities. To maintain local accuracy, the origin of n can be updated when necessary. Separate measurement models are defined for AIS and camera inputs.

4.3.2.1 Measurement Model - AIS

Measurements provided by the AIS API, as described in Section 2.3, are transformed to the local EKF frame before being used. The full measurement vector for AIS is defined as:

$$
\mathbf{z}^{\text{AIS}} = \begin{bmatrix} N^{\text{AIS}} & E^{\text{AIS}} & \psi^{\text{AIS}} & \epsilon^{\text{AIS}} & V^{\text{AIS}} \end{bmatrix}^T \tag{4.5}
$$

The position in North and East in frame $\{n\}$ is modeled as:

$$
\begin{aligned}
\hat{N}_k^{\text{AIS}} &= N_{k|k-1} + v_k^{\text{AIS-N}} \\
\hat{E}_k^{\text{AIS}} &= E_{k|k-1} + v_k^{\text{AIS-E}}
\end{aligned}
\tag{4.6}
$$

AIS provides the vessel's true heading (ψ) and course over ground (ϵ) following the convention of Figure 3.3. The true heading is used directly, while the course is calculated from the velocity components u and v using atan2(v, u). These measurements are modeled as:

$$\begin{aligned} \hat{\psi}_k^{\text{AIS}} &= \psi_{k|k-1} + v_k^{\text{AIS}-\psi} \\ \hat{\epsilon}_k^{\text{AIS}} &= \psi_{k|k-1} + \beta_{k|k-1} + v_k^{\text{AIS}-\epsilon} \\ &= \psi_{k|k-1} + \text{atan2}(v_{k|k-1}, u_{k|k-1}) + v_k^{\text{AIS}-\epsilon} \end{aligned} \tag{4.7}$$

AIS velocity measurement reflects the total ground speed magnitude. This is modeled as:

$$\hat{V}_k^{\text{AIS}} = \sqrt{(u_{k|k-1})^2 + (v_{k|k-1})^2} + v_k^{\text{AIS-V}} \tag{4.8}$$

The measurement model defines the nonlinear function $\mathbf{h}^{\text{AIS}}(\mathbf{x})$, which is used in the EKF correction step defined in Equation A.13. For linearization, the Jacobian of this function with respect to the state vector yields the matrix $\mathbf{H}_k^{\text{AIS}}$, which is evaluated at each time step. Only the relevant partial derivatives are shown; the remaining elements are zero:

$$\mathbf{H}^{\text{AIS}}(\mathbf{x}) = \frac{\partial \mathbf{h}^{\text{AIS}}}{\partial \mathbf{x}} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & \dots \\ 0 & 1 & 0 & 0 & 0 & \dots \\ 0 & 0 & 1 & 0 & 0 & \dots \\ 0 & 0 & 1 & \dfrac{-v}{u^2+v^2} & \dfrac{u}{u^2+v^2} & \dots \\ 0 & 0 & 0 & \dfrac{u}{\sqrt{u^2+v^2}} & \dfrac{v}{\sqrt{u^2+v^2}} & \dots \end{bmatrix} \tag{4.9}$$

4.3.2.2 Measurement Model - Camera

Camera measurements are obtained as relative positions (x, y, z) from the drone to the target vessel, expressed in the drone's body frame. Since the EKF models planar motion only, the z component is disregarded. The remaining x and y components are rotated into the local frame $\{n\}$ using the drone's yaw angle ψ_d, yielding a relative position in the EKF frame.
The measurement vector is defined as:

$$\mathbf{z}^{\text{cam}} = \begin{bmatrix} x^{\text{cam}} \\ y^{\text{cam}} \end{bmatrix} = \mathbf{h}^{\text{cam}}(\mathbf{x}) + \mathbf{v}^{\text{cam}} \tag{4.10}$$

This measurement function is modeled by:

$$\mathbf{z}^{\text{cam}}(\mathbf{x}) = \mathbf{R}_z^T(\psi_{d,k}) \begin{bmatrix} N_{k|k-1} - N_{d,k} \\ E_{k|k-1} - E_{d,k} \end{bmatrix} + \mathbf{v}_k^{\text{cam}} \tag{4.11}$$

Expanding each component gives:

$$\begin{aligned}\hat{x}_k^{\text{cam}} &= (N_{k|k-1} - N_{d,k})\cos(\psi_{d,k}) + (E_{k|k-1} - E_{d,k})\sin(\psi_{d,k}) + v_k^{\text{cam-x}} \\ \hat{y}_k^{\text{cam}} &= -(N_{k|k-1} - N_{d,k})\sin(\psi_{d,k}) + (E_{k|k-1} - E_{d,k})\cos(\psi_{d,k}) + v_k^{\text{cam-y}}\end{aligned} \tag{4.12}$$

Given the state vector in Equation 4.1, the Jacobian of $\mathbf{h}^{\text{cam}}(\mathbf{x})$ with respect to the state is:

$$\mathbf{H}^{\text{cam}}(\mathbf{x}) = \frac{\partial \mathbf{h}^{\text{cam}}}{\partial \mathbf{x}} = \begin{bmatrix} \cos(\psi_{d,k}) & \sin(\psi_{d,k}) & 0 & \dots \\ -\sin(\psi_{d,k}) & \cos(\psi_{d,k}) & 0 & \dots \end{bmatrix} \tag{4.13}$$

Only the relevant partial derivatives are shown; the remaining elements are zero. The Jacobian is evaluated at each time step using the current drone pose.

4.3.3 Uncertainty Modeling

Bayesian filtering relies on modeling uncertainties, which are captured in the process noise and measurement noise covariance matrices, denoted by Q and R, respectively. Since there are two separate measurement models, two separate measurement noise covariance matrices are defined: R^{AIS} and R^{cam}. Each matrix is detailed in the following sections.

4.3.3.1 Measurement Noise Covariance - AIS

AIS measurements are subject to several sources of error, which are captured in the measurement noise covariance matrix R^{AIS}. AIS messages are timestamped to the nearest second (see Section 2.3), introducing temporal quantization error. At higher speeds, this results in greater positional uncertainty. The standard deviation of the position error, σ_{xy}, is modeled as a function of the vessel's velocity and a base GPS accuracy error σ_{GPS}.

$$\sigma_{\text{xy}}(V) = \sigma_{\text{GPS}} + \delta t \cdot V \tag{4.14}$$

The term δt refers to the timestamp uncertainty, for example 0.5s when time is rounded to the nearest second. Velocity is assumed to be linear during δt. The North and East positional variances are assumed to be equal.

$$R_{\text{NN}}^{\text{AIS}} = R_{\text{EE}}^{\text{AIS}} = \sigma_{\text{xy}}(V^{\text{AIS}})^2 \tag{4.15}$$

At low speeds, the AIS-derived course is often noisier due to ambiguity in the vessel's true direction of motion. Figure 4.4 shows this behavior at standstill. It is addressed by using a sigmoid function to transition smoothly from a high expected angular error σ_{high} with low velocity to a lower error σ_{low} at higher velocity.

$$\sigma_{\text{deg}}(V) = \sigma_{\text{low}} + (\sigma_{\text{high}} - \sigma_{\text{low}}) \cdot \left(\frac{1}{1 + e^{-\zeta(V_0 - V)}}\right) \tag{4.16}$$

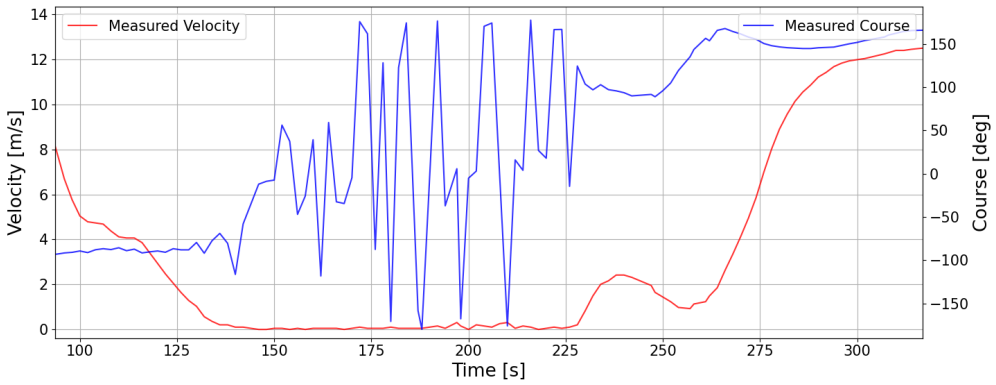


Figure 4.4: Plot with course and velocity while stationary. The plot displays the quickly changing behavior of the measurement when the vessel is close to standstill. Data stems from AIS gathered from the island ferry "Rivö" at sea in the archipelago outside Gothenburg.

The inflection point V_0 denotes the velocity at which the transition occurs and ζ controls the sharpness of this transition. The heading and course variances are modeled identically:

$$R_{\psi\psi}^{\text{AIS}} = R_{\epsilon\epsilon}^{\text{AIS}} = \sigma_{\text{deg}}(V^{\text{AIS}})^2 \tag{4.17}$$

Velocity measurement noise is assumed to be constant:

$$R_{\text{VV}}^{\text{AIS}} = \sigma_{\text{V}}^2 \tag{4.18}$$

No correlations between measurements are modeled, which means that the covariance matrix only has diagonal elements.

$$R^{\text{AIS}} = \begin{bmatrix} R_{\text{NN}}^{\text{AIS}} & 0 & 0 & 0 & 0 \\ 0 & R_{\text{EE}}^{\text{AIS}} & 0 & 0 & 0 \\ 0 & 0 & R_{\psi\psi}^{\text{AIS}} & 0 & 0 \\ 0 & 0 & 0 & R_{\epsilon\epsilon}^{\text{AIS}} & 0 \\ 0 & 0 & 0 & 0 & R_{\text{VV}}^{\text{AIS}} \end{bmatrix} \tag{4.19}$$

4.3.3.2 Measurement Noise Covariance - Camera

For vision-based distance estimation, measurement uncertainty increases with distance. The forward (Δx) and lateral (Δy) components of error are modeled separately. The forward error stems primarily from range estimation, while the lateral error is influenced by angular uncertainty. Assuming no correlation, the covariance matrix is diagonal and defined as:

$$d = \sqrt{(\Delta x)^2 + (\Delta y)^2 + (\Delta z)^2} \tag{4.20}$$

$$R^{\text{cam}} = \begin{bmatrix} (\sigma_x \cdot d)^2 & 0 \\ 0 & (\sigma_\alpha \cdot d)^2 \end{bmatrix} \tag{4.21}$$

4.3.3.3 Process Noise Covariance

The process noise covariance matrix Q captures uncertainty in the system's internal dynamics due to unmodeled forces and model imperfections. These include all unobserved control inputs, environmental disturbances, and higher-order motion components such as jerk.

Translational acceleration in both the x and y directions is modeled as white Gaussian noise $a_{\mathrm{x}}, a_{\mathrm{y}} \sim \mathcal{N}(0, \sigma_a^2)$. Assuming constant acceleration within each time step Δt, the resulting covariance matrix for position–velocity coupling in each axis is:

$$Q_{\text{pos-vel}} = \sigma_a^2 \begin{bmatrix} \frac{\Delta t^4}{4} & \frac{\Delta t^3}{2} \\ \frac{\Delta t^3}{2} & \Delta t^2 \end{bmatrix} \tag{4.22}$$

Yaw rate is modeled the same way, with angular acceleration modeled as white noise $\dot{r} \sim \mathcal{N}(0, \sigma_{\mathrm{AR}}^2)$.

$$Q_{\text{yaw}} = \sigma_{\mathrm{AR}}^2 \begin{bmatrix} \frac{\Delta t^4}{4} & \frac{\Delta t^3}{2} \\ \frac{\Delta t^3}{2} & \Delta t^2 \end{bmatrix} \tag{4.23}$$

Acceleration states a_x and a_y evolve according to a random walk driven by jerk, which is modeled as zero-mean white noise $\dot{a}_x, \dot{a}_y \sim \mathcal{N}(0, \sigma_{\mathrm{J}}^2)$. The resulting noise contribution is added to the diagonal of Q.

$$Q_{\text{acc}} = \sigma_J^2 \Delta t \tag{4.24}$$

Each component in Q is added in block-diagonal form, assuming no other correlations. This approach reduces the number of tuning parameters for the Q matrix to three $(\sigma_A, \sigma_{AR}, \sigma_J)$, simplifying the tuning process.

4.3.4 Out of Sequence Measurement Handling

Both measurement sources introduce non negligible time delays between the moment a measurement is taken and when it becomes available to the filter. AIS data typically has a delay between 2 to 4 seconds, while camera based measurements are delayed by approximately one second. Because of this, measurements often arrive out of sequence of each other.

This issue is solved with buffers for both states and measurements. When a delayed measurement arrives, the EKF rolls back to the closest prior state estimate and applies the measurement. The delayed measurement is inserted in chronological order into the measurement buffer. The filter then reapplies all subsequent measurements in sequence to propagate the state forward to the current time.

4.4 Guidance

This section focuses on the *Guidance* part from Figure 4.2, shown in Figure 4.5.

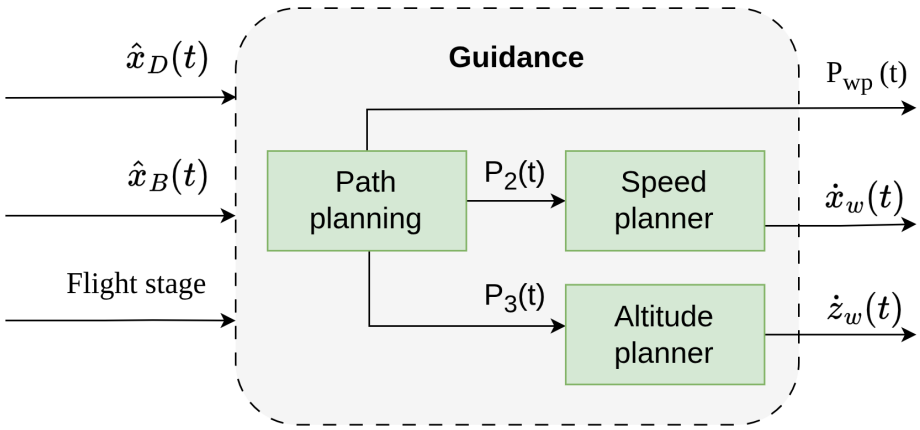


Figure 4.5: Overview of the *Guidance* system, sub-system of the entire system overview, Figure 4.2

It presents the developed system and the calculations needed to enable autonomous drone landing on a moving boat. The landing algorithm is structured into three flight stages: the *follow stage*, the *landing stage*, and the *diversion stage* which are described in Sections 4.4.1, 4.4.2, and 4.4.3 respectively, including the relevant equations, methodologies, and preliminary results obtained from simplified test scenarios. The objective of the *Path planning* block in Figure 4.5 depends on the current flight stage.

To aid in the explanation of all the flight stages, Figure 4.6 & Figure 4.7 illustrate the general flight behavior during landing. These figures will be referenced in the following subsections to clarify how waypoints are calculated in follow stage and landing stage. P_1 is the point from which the drone should follow the boat, P_2 is the starting point of the descent and P_3 is the final landing point.

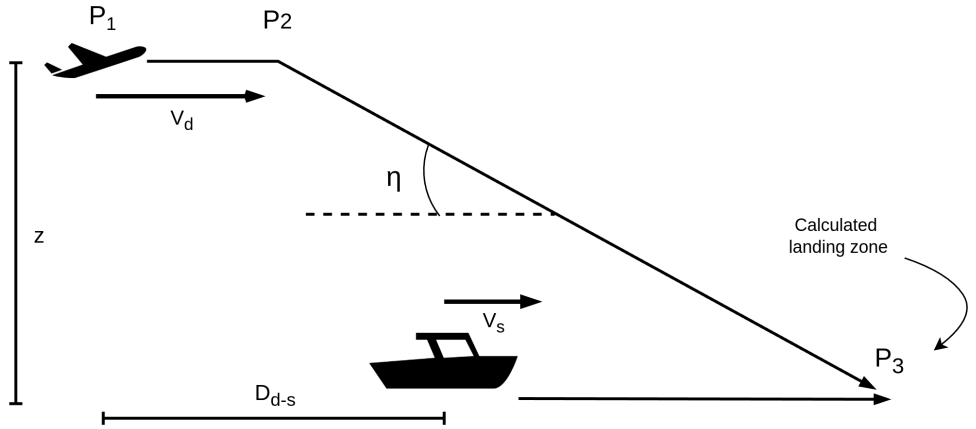


Figure 4.6: Visualization of how a landing works. η is the glide slope angle, while D_{d-s} represents the distance from the drone to the ship.

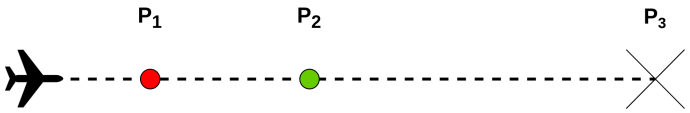


Figure 4.7: Top-down view of the landing path. P_1: speed setpoint, P_2: descent start point, P_3: landing point

4.4.1 Follow Stage

The follow stage refers to the stage of the mission in which the drone tracks the boat while attempting to keep a constant distance behind it. This process is handled by the *Path planning* and *Speed planner* blocks in Figure 4.2. This stage requires real-time decision-making to ensure that the drone aligns with the boat's current trajectory. This section presents several guidance methods in Section 4.4.1.1, then identifies the selected approach, and explains its implementation in Section 4.4.1.2. Additionally, a strategy for consistently maintaining the desired following distance is also outlined in Section 4.4.1.2.

4.4.1.1 Guidance Methods

The guidance methods are responsible for directing the drone along a desired trajectory to the boat. These methods typically rely on kinematic and geometric relationships between pursuer (drone) and target (boat). This subsection will explore a few common strategies to do this, each varying in complexity, performance and applications.

Proportional navigation The basic principle of proportional navigation is that the pursuer maintains a constant bearing to the target, resulting in an interception course [17], as shown in Figure 4.8

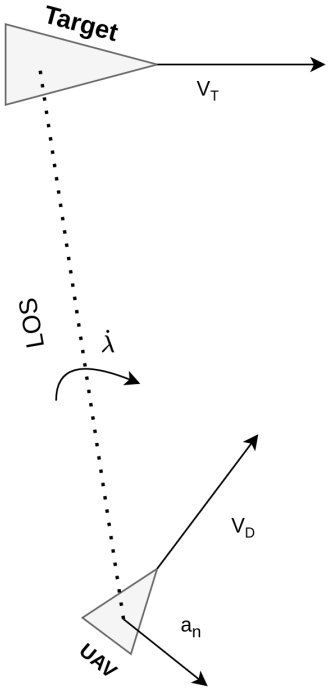


Figure 4.8: Visualization of the proportional navigation method.

This can be described as applying a lateral acceleration that is proportional to the rate of change of the line of sight (LOS) angle, denoted by $\dot{\lambda}$, as shown in the equation below [18]:

$$a_n = N\dot{\lambda}v_r \tag{4.25}$$

Here, a_n is the required lateral acceleration, perpendicular to the pursuer's velocity vector, $\dot{\lambda}$ is the LOS rate, N is the navigational ratio and v_r is the closing velocity defined as the difference between pursuer's and target's speeds $V_D - V_T$.

Proportional navigation is widely used in missile guidance systems due to its simplicity and effectiveness. It requires relatively little information about the target's motion compared to more complex methods [17].

Beam riding Beam riding is a three point guidance system that uses an external source to project a beam, typically a laser or radar, from a reference point to the target [19]. The pursuer attempts to remain within this beam, effectively following it straight from the reference point to the target as shown in Figure 4.9

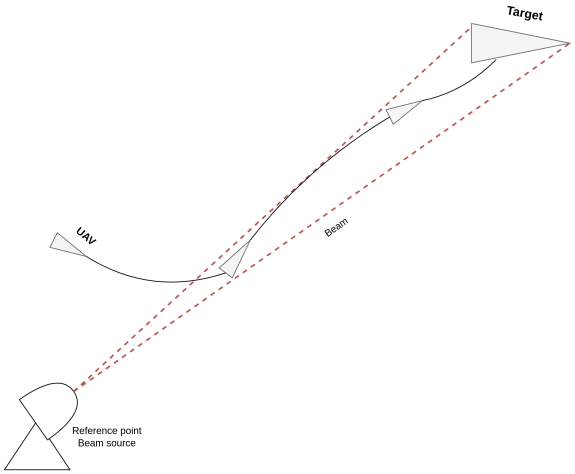


Figure 4.9: Visualization of the beam rider method

Pure pursuit Pure pursuit is a straightforward guidance method where the pursuer constantly steers toward the current position of the target [20]. In essence, it aligns its velocity vector with the LOS vector to the target, as shown in Figure 4.10.

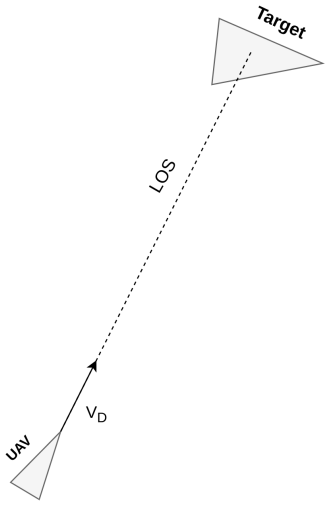


Figure 4.10: Visualization of the pure pursuit method

This method is intuitive and simple to implement especially if the target's position is available. However, because it reacts to the target's current position rather than predicting its future path, it can result in irregular trajectories, especially if the target is maneuvering.

To improve performance, a look ahead can be incorporated where the pursuer aims at a point in front of the target's current position. This helps smooth out trajectories and improves overall flight behavior [20].

4.4.1.2 Follow Strategy

When the drone operator activates the *follow stage*, the drone begins following the boat using the pure pursuit algorithm described in Section 4.4.1.1. This algorithm was chosen for its straightforward implementation and because it guides the drone directly towards the boat along a trajectory that can easily be improved by tuning the look-ahead distance.

In this stage, the drone flies towards the point P_2 in Figure 4.6 which is positioned directly behind the boat and marks the point where the drone should enter its landing glide path. The calculations needed for P_2 are explained in Section 4.4.2.2.

Although P_2 defines the intended point of initiating descent, the drone does not directly follow this point during the follow stage. Instead, a lookahead distance is applied in front of P_2 causing the drone to track a target slightly ahead. This approach helps the drone maintain a smooth trajectory, particularly when the boat is maneuvering or changing speeds.

To maintain a margin and avoid abrupt maneuvers, a secondary point P_1 is defined 20m behind P_2. This is the point that the drone will follow from and provides smoother transitions between stages and a decision-making buffer time for the control systems.

To stay a fixed distance behind the boat, the drone uses a proportional controller that controls its velocity depending on the distance to P_1. This control logic is defined by the equation below and illustrated in fig. 4.11

$$\begin{aligned} &\textbf{Distance behind P1: } D_{d-P1} = D_{d-s} - D_{P1-s} \\ &\textbf{Speed correction: } \dot{x}_{corr} = D_{d-P1} \cdot K_{speed} \\ &\textbf{Wanted speed: } \dot{x}_{wanted} = \dot{x}_{boat} + \dot{x}_{corr} \end{aligned} \tag{4.26}$$

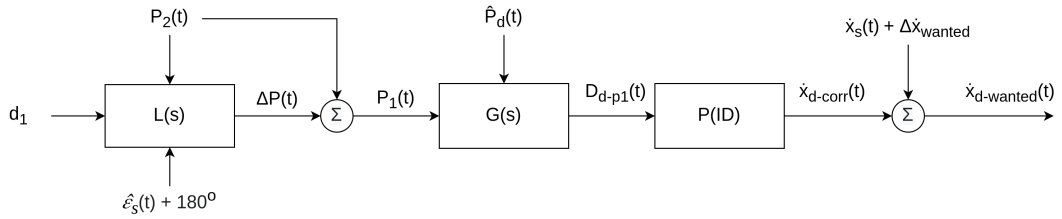


Figure 4.11: Control flow of how the *speed planner* block from Figure 4.5 calculates the wanted speed.

Figure 4.12 presents preliminary results demonstrating the performance of the follow strategy in a simple scenario where the boat performs minor maneuvers. As shown, the drone can successfully track the boat's trajectory when it is turning, maintaining accurate alignment for when a landing command is issued. Figure 4.13 shows the distance between the drone and the "Aft projection", a line from the boat straight back through P_2. This plot will be used as a reference to evaluate the different scenarios in chapter 5. Figure 4.14 shows the drone's distance behind the point P_1

over time. This plot indicates that the drone consistently maintains a distance close to zero behind P_1 which is the intended behavior in the follow stage. The minor fluctuations in distance are not critical, as the purpose of P_1 is to provide a buffer and margin to allow for such variations during the follow stage.

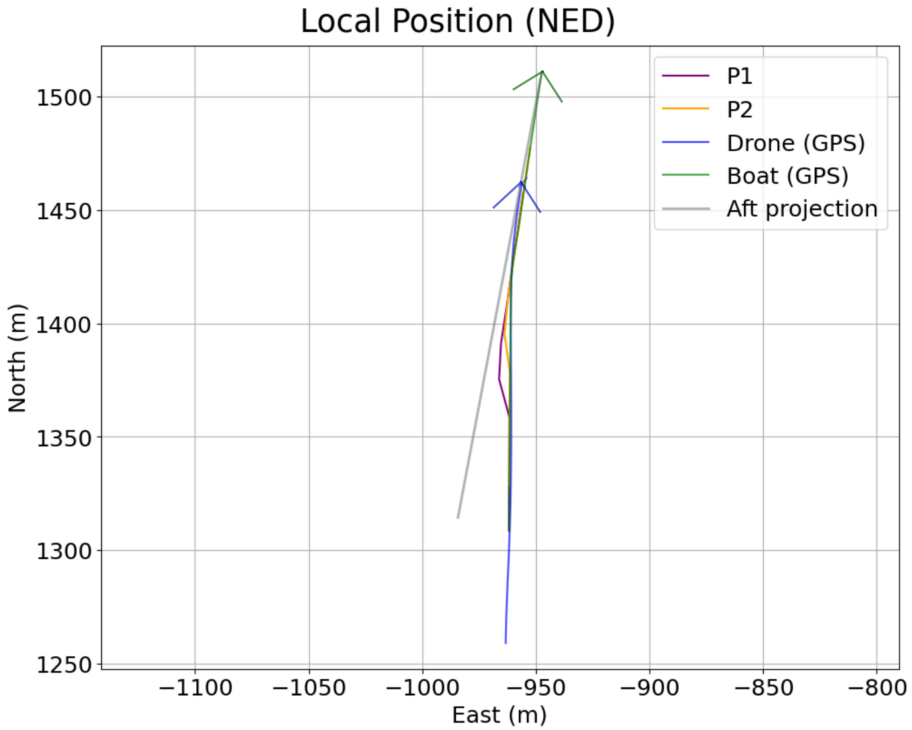


Figure 4.12: This figure shows the trajectories of the drone and the boat over time for the base case, under no disturbances. The current positions of both are marked with arrows. P1 and P2 indicate the speed set point and the descent start point, respectively. The "Aft projection" is a line extending straight back from the boat through P2.

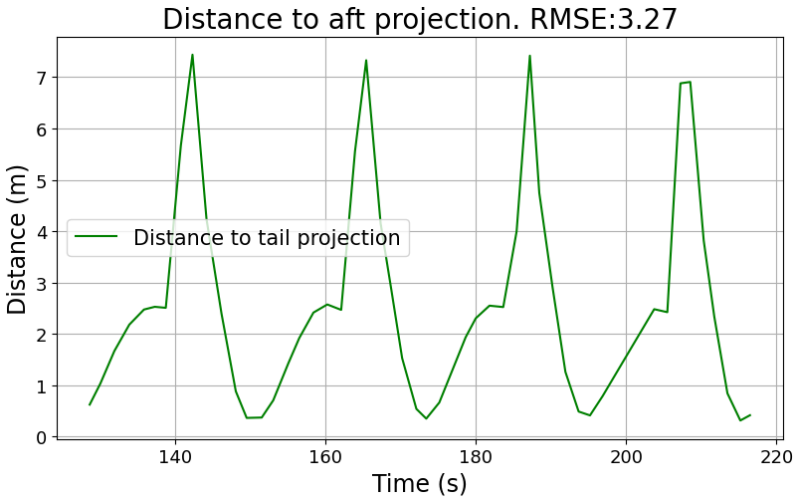


Figure 4.13: Shows the distance between the drone and the "Aft projection" over time as well as an average RMSE for this distance over the time steps included in the plot.

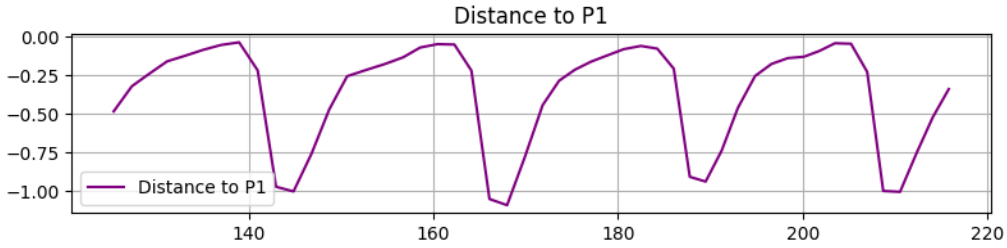


Figure 4.14: Drone's distance behind the speed setpoint, P_1 in meters over time in *follow stage*.

4.4.2 Landing Stage

The *landing stage* is responsible for intercepting and descending onto the boat. This process is handled by the *Path planner*, and *Altitude planner* blocks in Figure 4.5. Like the follow stage, it requires real-time decision making and continuous updates to ensure that the landing point and sink rate aligns with the current trajectories. This section begins by introducing the general landing strategy used, Section 4.4.2.1, followed by a detailed explanation of the calculations used to determine the landing point and required sink rate, Sections 4.4.2.2 and 4.4.2.3.

4.4.2.1 Landing Strategy

Landing normally consists of five distinct phases: base leg, approach, flare, touchdown, and after-landing roll which are visualized in Figure 4.15 [21, 22]. This section will describe the specific responsibilities associated with each of these phases.

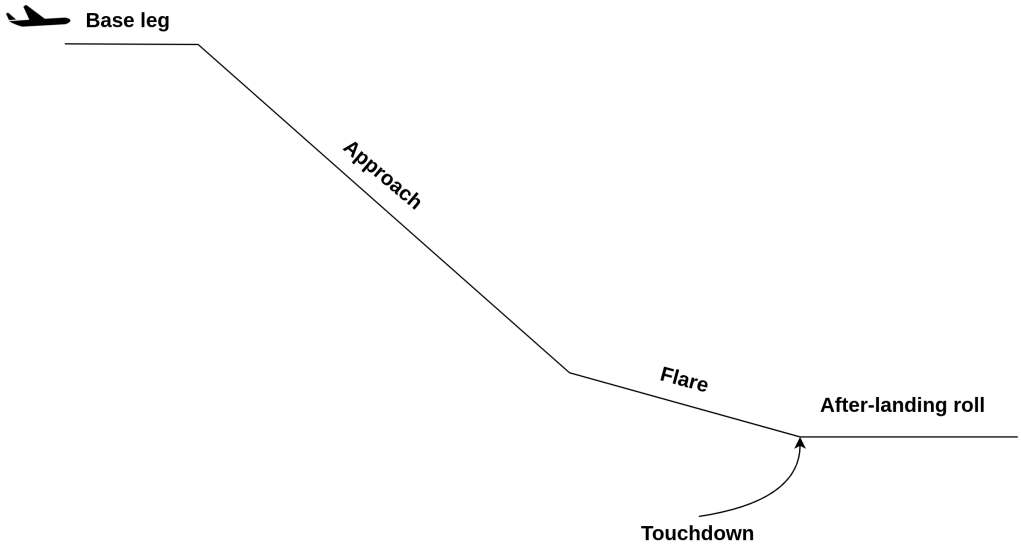


Figure 4.15: Visualization of the different phases during a landing.

The base leg phase includes alignment for the approach phase and catching up to the approach start point. Once at the approach start point, the approach phase begins. During this stage, the aircraft begins descending toward the touchdown point while following a defined glide slope which ensures a smooth and controlled descent. The glide slope is defined as the ratio of vertical distance (z-direction) to horizontal distance (x-direction) traveled.

The third phase, known as the flare, involves pitching up the aircraft just before touchdown to reduce vertical velocity and minimize impact force. This maneuver enables a smoother, less abrupt landing.

The touchdown phase marks the moment when the aircraft makes contact with the landing surface. Finally, in the after-landing roll phase, the aircraft decelerates and comes to a complete stop on the runway.

In this thesis, however, the landing takes place on a moving boat, where the impact speed can be defined and set to a low value. Thus, the after-landing roll phase becomes largely irrelevant due to the minimal relative motion. For this reason, a simplified four-phase maneuver used in this thesis is described below:

- **Base leg:** Aligning with the glide path behind the vessel while maintaining altitude.
- **Approach:** Descending toward the landing point along a glide slope.
- **Flare:** Pitching up to reduce descent rate at the end of the glide slope. (This is not done explicitly in this thesis, instead, Ardupilot's controllers will automatically slow down the descent as it gets close to the target altitude.)
- **Touchdown:** Making contact with the rear deck of the boat.

4.4.2.2 Landing Calculations

The position of the descent start point P_2 is determined by specifying a glide slope angle η and calculating its distance behind the boat using the following equations:

$$\begin{aligned} &\textbf{Glide ratio: } G_r = \tan(\eta). \\ &\textbf{Distance to descend drone: } D_{d-P3} = \frac{z}{G_r}. \end{aligned} \tag{4.27}$$

$$\begin{aligned} &\textbf{Equation of motion for the drone: } v_d = \frac{D_{d-P3}}{t}. \\ &\textbf{Equation of motion for the boat: } v_s = \frac{D_{s-P3}}{t}. \\ &\textbf{P2 distance behind boat: } D_{s-P_2} = D_{d-P3} - D_{s-P3}. \end{aligned} \tag{4.28}$$

The computed distance D_{s-P_2} is then used with the look-ahead Equation 3.7, to calculate the coordinates of P_2 based on the boat's current position and heading. This calculation is performed continuously during the follow stage, meaning P_2 is updated each iteration based on the latest states of the drone and boat. This process is illustrated in Figure 4.16, where the block $D(s)$ corresponds to Equations 4.27 and 4.28, and $L(s)$ refers to Equation 3.7.

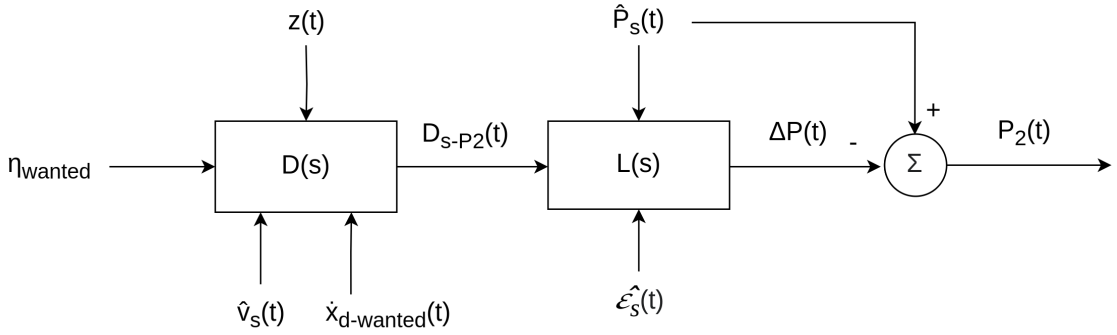


Figure 4.16: Control flow for how the *Path planner* block in Figure 4.5 calculates the descent start point, P_2

The coordinate for the landing point P_3 is calculated similarly, by calculating how far the boat will travel before the drone intercepts it horizontally. This step is independent of the drone's current altitude. Here, D_{d-s} is obtained from Eq. 3.6 and represents the drone's current horizontal distance to the boat. D_{s-P3} is then derived using Eq. 4.28 by substituting D_{s-P_2} for D_{d-s}. Once D_{s-P3} is obtained, it is combined with the boat's heading and coordinates to determine the position of P_3 using Eq. 3.7.

Similarly to P_2, the position of P_3 is calculated and updated each iteration of the landing stage to ensure that the touchdown point is aligned with the boat's current trajectory. This process is illustrated in Figure 4.17, where $G(s)$ corresponds to Eq. 3.6, and $E(s)$ refers to Equation 4.28.

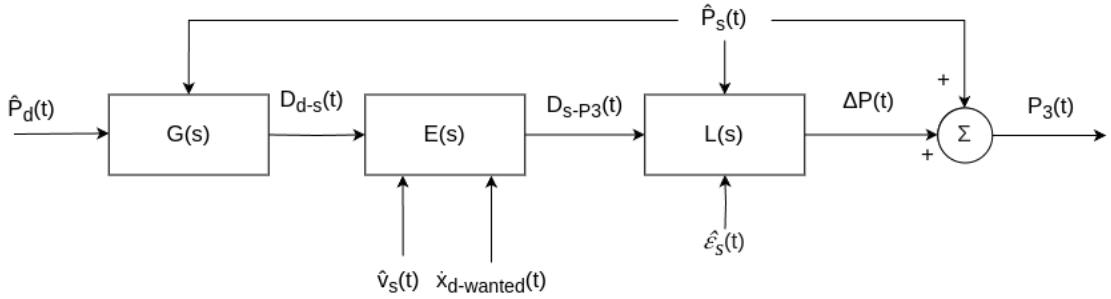


Figure 4.17: Control flow for how the *Path planner* block in Figure 4.5 calculates the landing point, P_3

The boat's heading and position are used for calculating both P_2 and P_3 instead of the drone's, since the drone's heading might not be aligned with the boat's trajectory.

4.4.2.3 Altitude Planner

During descent, the drone is guided towards the latitude and longitude of P_3 and an altitude equal to the boat for touchdown. To ensure a controlled descent, a control system is implemented that regulates the drone's vertical speed (sink rate), to keep it at the desired altitude throughout the descent. This ensures the drone follows a glide slope that allows it to approach the boat from above and avoid hitting the stern.

In practice, a **descent lookahead** is implemented. To compensate for internal delays in the system, a look-ahead distance is applied to P_2 to start descending slightly earlier. This helps the drone keep the desired altitude early on in the descent.

Calculation of desired altitude and glide ratio: At each time step, the *altitude planner* calculates a target altitude based on the previous glide ratio (Gr_{k-1}) and distance to previous P_3, $P_{3,k-1}$. This desired altitude is used along with the distance to the current $P_{3,k}$ to calculate a new glide ratio. This process is visualized in Figure 4.18

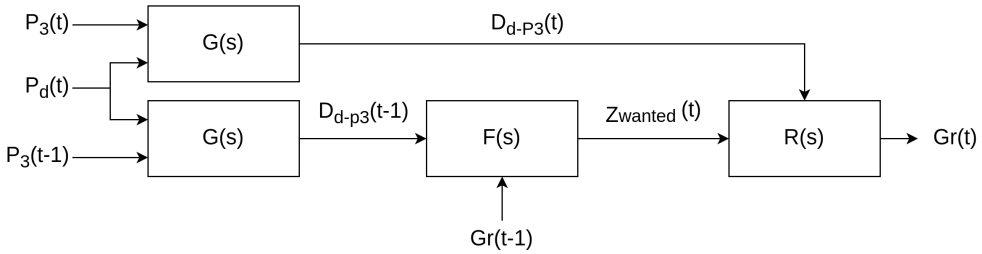


Figure 4.18: Control flow for how the *altitude planner* block in Figure 4.5 calculates the wanted altitude, z_{wanted} and the glide ratio, Gr.

Here, $G(s)$ is from Eq. 3.6, $F(s)$ performs calculations according to Eq. 4.29 and $R(s)$ uses Eq. 4.30

$$Z_{wanted}(t) = D_{d-P_3}(t-1) \cdot Gr(t-1) \tag{4.29}$$

$$Gr(t) = \frac{Z_{wanted}(t)}{D_{d-P_3}(t)} \tag{4.30}$$

This logic ensure that the glide ratio is only changed as a response to external factors such as winds or waves that affects the motion of either the drone or boat.

Correcting for the altitude error: As mentioned earlier, a control system is implemented to ensure the drone stays on the desired glide slope. This is implemented as a P-controller that adds the drone's altitude error multiplied by a gain to the desired sink rate. This process is defined by the equations below and illustrated in Figure 4.19.

$$\dot{Z}_{wanted}(t) = \dot{x}_{drone}(t) \cdot Gr(t) \tag{4.31}$$

$$\begin{aligned} e_{altitude}(t) &= Z_{drone}(t) - Z_{wanted}(t) \\ \dot{Z}_{needed}(t) &= \dot{Z}_{wanted}(t) + e_{altitude}(t) \cdot K_{altitude} \end{aligned} \tag{4.32}$$

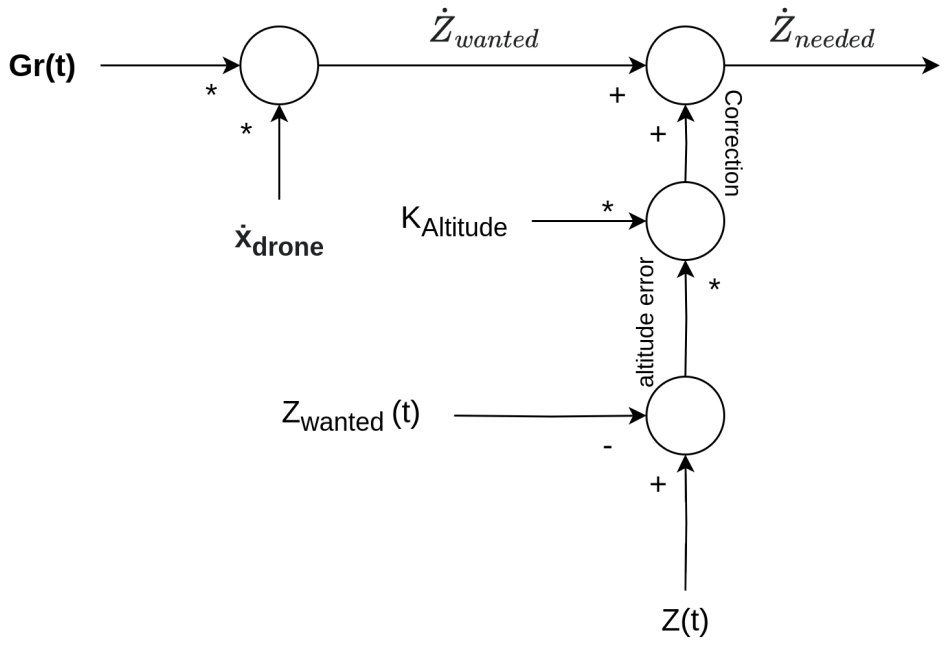


Figure 4.19: Control flow for how the *altitude planner* block in Figure 4.5 calculates the required sink rate, $\dot{Z}_{needed}$.

This method ensure that the drone adjusts its altitude to stay on the desired glide slope to maintain a safe descent path.

4.4.2.4 Preliminary Landing Strategy Performance

Figure 4.20 shows the distances between the drone and the boat in the different dimension along with the absolute distance to the boat. A landing is considered successful if the drone gets within 2 meters in x, 1.25 meters in y and 1.25 meters in z dimensions. These boundaries were determined based on the standard dimensions of SSRS sea rescue boats. The dotted line, "Min dist. point", marks the point of touchdown, when the drone had the closest absolute distance to the boat. Figure 4.21 shows that the drone is able to match its altitude to the desired altitude, although with a slight delay, using the P controller described in Section 4.4.2.3. These plots indicate that, in the absence of outside disturbances, the drone can accurately and precisely land on the boat.

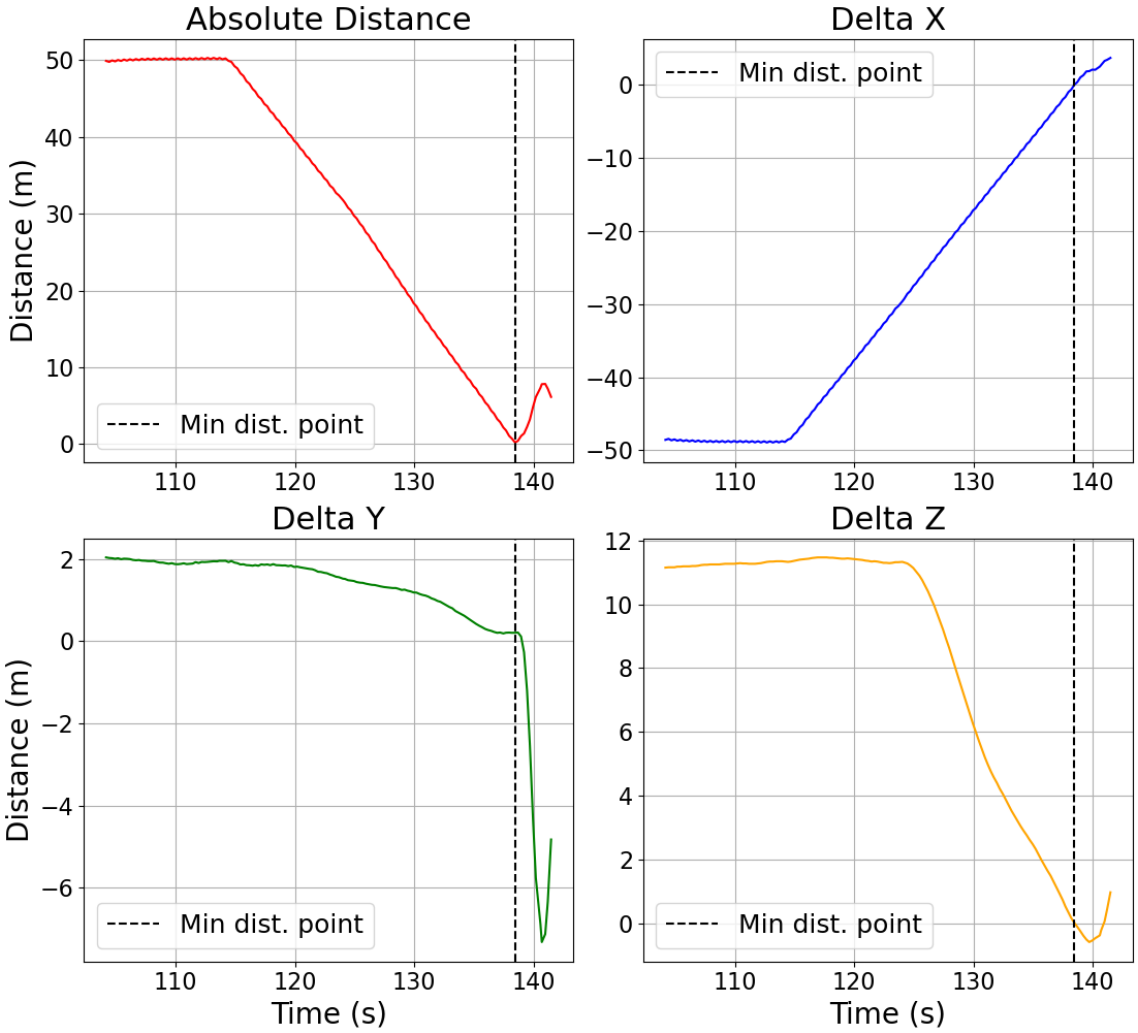


Figure 4.20: Distance data where the dotted vertical line indicates the identified moment of touchdown by looking at the closest absolute distance between the drone and the boat. Distances between drone and boat at touchdown: dx: 0.02m, dy: 0.20m, dz: 0.01m, absolute: 0.20m

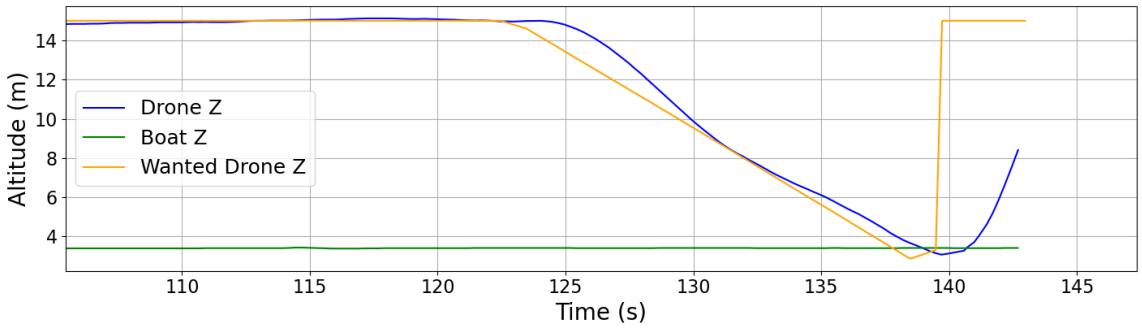


Figure 4.21: Shows the drone's and boat's altitude over time along with the drone's target altitude.

4.4.3 Diversion Stage

A diversion stage is implemented to prevent the drone from continuing in situations deemed unsafe. These situations are explained in Section 4.4.3.1 and the actions the drone needs to take in these situations are explained in Section 4.4.3.2. This stage is handled by the *Path planning* block in Figure 4.2

4.4.3.1 Activation Criteria

There are two main activation criterion for a diversion maneuver. The first criterion is triggered when the required speed in the follow stage becomes unachievable due to the drone's stall speed limit. If the drone cannot fly slowly enough and is too close to the boat, a diversion is required to increase the distance to the boat.

The second activation criterion is when the required glide ratio becomes to high. Apart from calculating the target glide ratio from the target altitude, the *altitude planner* also calculates the required glide ratio using the same Equation 4.30 but using the drone's actual altitude rather than the target one. If this glide ratio is too steep, the descent becomes too aggressive, increasing risk for damage and a diversion is needed to abort landing.

4.4.3.2 Diversion Strategy

The strategy during a diversion maneuver remains the same regardless of the trigger condition. When a diversion is initiated, the drone will set a target waypoint a distance $d_{diversion}$, 90 degrees offset from its current heading and then fly back to where it started the diversion maneuver. After the diversion is completed, the drone will go into follow stage regardless of what its previous stage was. This maneuver causes the drone to maintain its longitudinal (x) position for longer, thus increasing its distance to the boat. This process is visualized in Figure 4.22 & 4.23

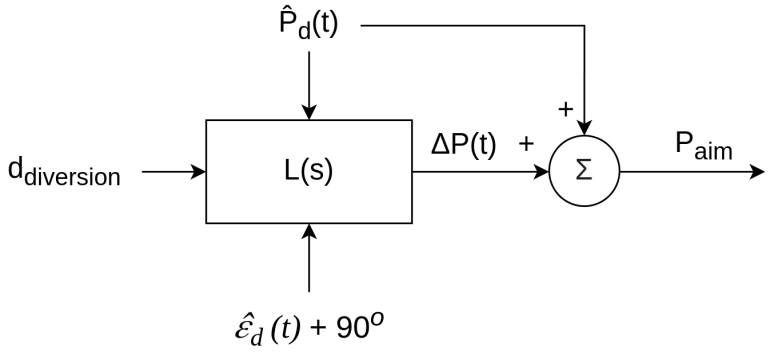


Figure 4.22: Control flow for how the *diversion stage* calculates the diversion path.

Where $L(s)$ is the lookahead Equation 3.7 also used in Section 4.4.2.2

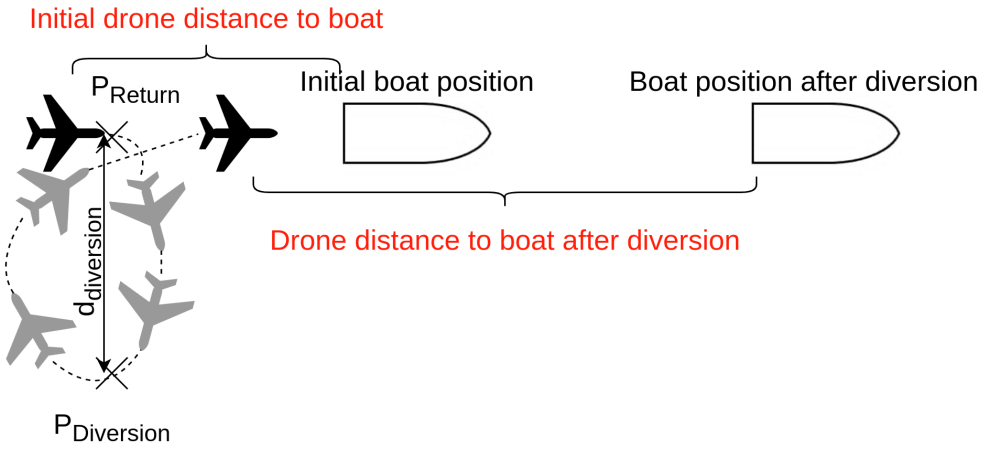


Figure 4.23: Visualization of how a diversion maneuver works from a top-down view.

As shown Figure 4.24 and 4.25, the diversion strategy works effectively in practice, successfully increasing the distance between the drone and the boat as intended. Although the drone does not immediately align with the boat's trajectory after the diversion maneuver, this is acceptable. The follow stage that comes after a diversion provides sufficient time and distance for the drone to realign itself.

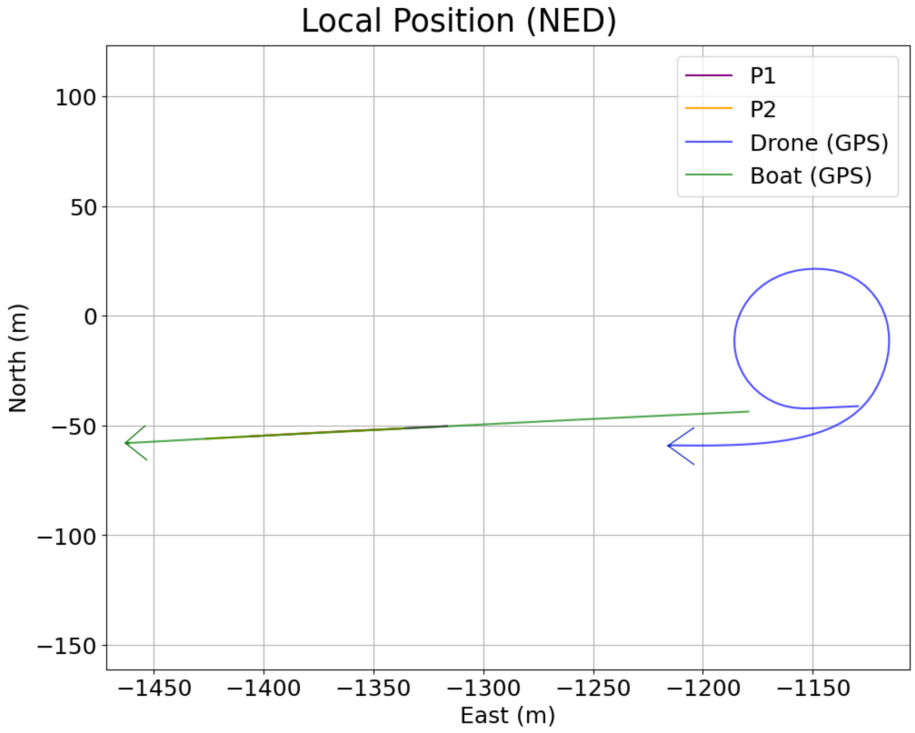


Figure 4.24: This figure shows the trajectories of the drone and the boat over time during a diversion maneuver. The current positions of both are marked with arrows. P1 and P2 indicate the speed set point and the descent start point, respectively.

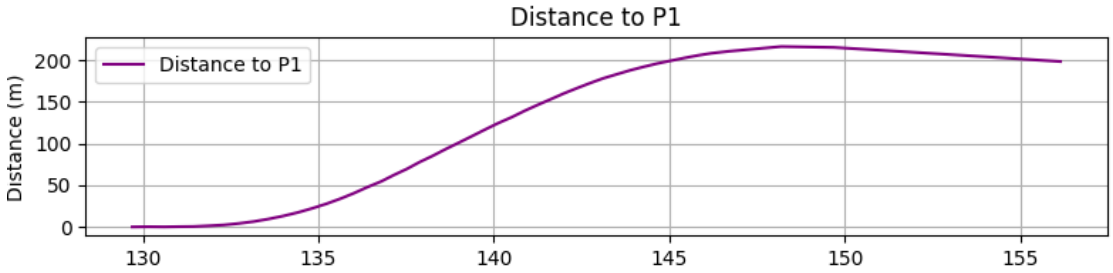


Figure 4.25: Drone's distance behind the speed setpoint, P_1 in meters over time in *diversion stage*.

5
Simulation & Evaluation

This chapter presents performance evaluation of the developed algorithms and the Kalman filter used for localizing the boat. It begins by describing the simulation based test environment in Section 5.1. The performances of the follow and landing algorithm are then evaluated across a range of scenarios, including variations in boat heading, Section 5.3, altitude, Section 5.4, and speed, Section 5.5, as well in windy scenarios, Section 5.2. Finally, this chapter gives an evaluation of the Kalman filter in Section 5.6.

5.1 Testing Environment

All testing was performed in a Gazebo simulation environment. The base scenario involves a boat traveling in a straight line, while a fixed-wing drone attempts to execute the implemented landing procedures. To simulate more realistic conditions, additional factors such as speed, heading, altitude and wind variations are introduced to emulate variations caused by environmental disturbances. This gives a more comprehensive evaluation of the implemented algorithms and gives a better idea of how they would perform in the real world. Each test scenario is designed to isolate and test one specific part of the algorithm. Detailed descriptions of these scenarios are given in the sections below.

A notable limitation in the simulation environment was the inability to directly control the drone's speed in Ardupilot-Gazebo plugin. The work around for this was to instead control the boat's speed, which gives the same relative velocity between the drone and the boat. Since only the relative motion affects the landing performance, this workaround did not invalidate the tests. This limitation is not persistent for the EOS drone.

Another limitation is the update frequency of the virtual connection to the simulation (MAVLink), which introduced several issues. In some cases, the delay results in plots that appear inconsistent, even though the drone actually landed precisely on the boat in the simulation. Specific instances of this discrepancy will be highlighted in the sections below.

The different scenarios that were tested are listed below:

- **Wind -** Wind was simulated by introducing a base wind speed and direction in the simulation environment, with an added sine wave and Gaussian noise

to replicate natural wind variability.

- **Movement fluctuation -** Random Gaussian noise was added to the boat's heading to simulate inconsistent motion. This noise is implemented as a random walk, where a random heading is added to the current heading incrementally. This gives a behavior where the boat can deviate from its original path over time.
- **Altitude fluctuation -** Variations in boat altitude were implemented as a sine wave with added Gaussian noise. The altitude changes impact the final landing point P_3 and ultimately the drone's landing trajectory. This mimics height variation caused by waves in the real world.
- **Throttle fluctuation -** Due to the limitations in changing the drone's speed directly, Gaussian noise was added to the drone's throttle instead of varying the speed of the boat. The boat then adjusts its speed to maintain the desired relative velocity.

The tests for the follow stage were considered successful if the drone's distance to P_1 remained within ± 20 meters, as P_1 is located 20 meters behind P_2. Additionally, a successful test requires that the root mean square error (RMSE) between the drone's and the "Aft projection" remains below 10, as above this value, the drone could not consistently stay behind the boat. For the landing stage, tests require that the drone successfully lands on the boat, which is evaluated by how close the drone gets to the boat. Specifically, the drone should be within (2, 1.25, 1.25)m in x, y and z directions, which is based on the standard dimensions of SSRS rescue boats. Not all SSRS vessels have exactly the same dimensions so lower distances is still always preferred. The target glide ratio was set to $\frac{1}{20}$ for all tests as that is close to the drone's natural glide ratio with the motors turned off.

5.2 Testing With Wind

Simulations were conducted with varying wind speeds but only the most severe conditions that the drone could handle while still completing its tasks are included. At higher wind speeds than the ones included, the drone could no longer effectively perform its designated tasks within the defined limits. The effects of wind were evaluated in both the follow and landing stage as wind impacts the performance of both.

5.2.1 Follow Stage

Figure 5.1 shows the wind speed and direction over time. Both parameters vary with a sinusoidal wave with added Gaussian noise around a base value. As shown in 5.2 and 5.4, the drone is capable of tracking the boat's trajectory within the boundaries, even when the boat is turning regularly. Figure 5.3 shows that the distance to P_1 is within acceptable boundaries but struggles to maintain a consistent distance in windy conditions. The drone's ability to maintain a consistent distance was the main limiting factor for the drone's performance in windy conditions, as the distance behind P_1 could vary by up to 60 meters under stronger wind conditions.

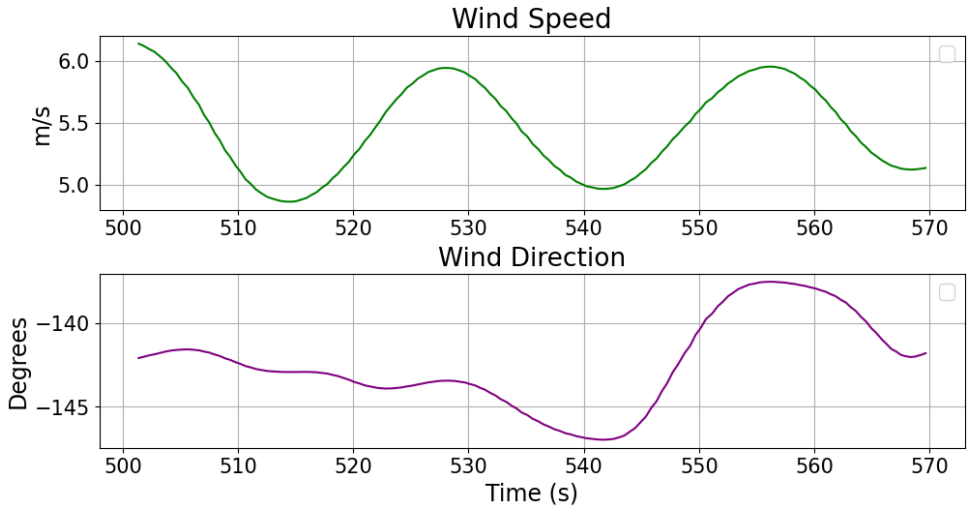


Figure 5.1: Wind speed and direction over time.

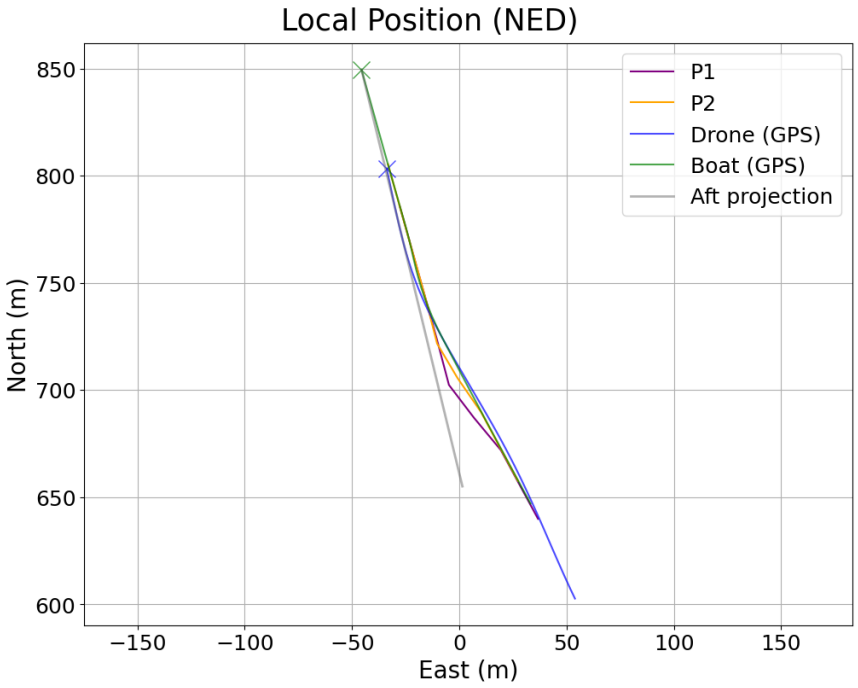


Figure 5.2: This figure shows the trajectories of the drone and the boat over time under wind disturbances. The current positions of both are marked with arrows. P1 and P2 indicate the speed set point and the descent start point, respectively. The "Aft projection" is a line extending straight back from the boat through P2. This figure can be compared to Figure 4.12, which shows the trajectories in the base case without disturbances.

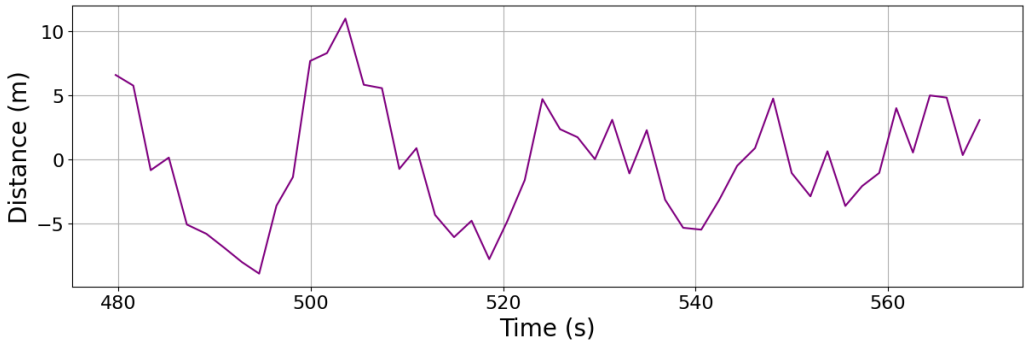


Figure 5.3: Drone's distance behind the speed setpoint, P_1 in meters over time in windy conditions.

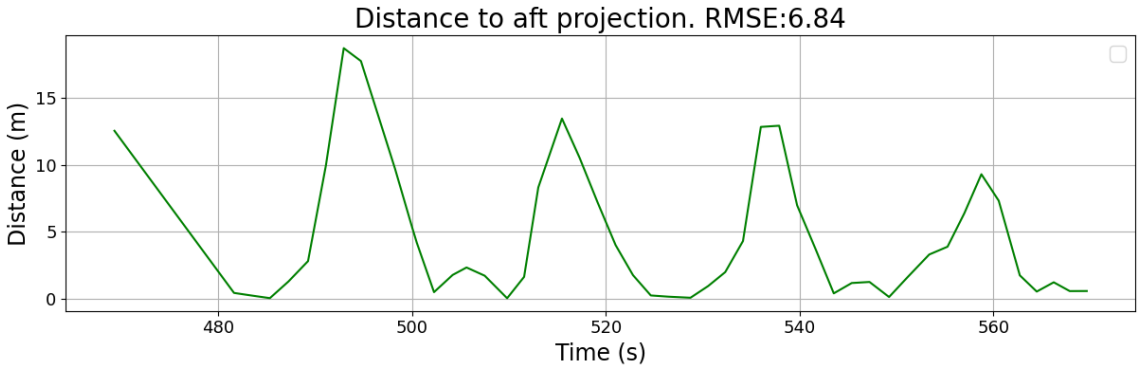


Figure 5.4: Shows the distance between the drone and the "Aft projection" over time as well as an average RMSE for this distance over the time steps included in the plot during windy conditions.

5.2.2 Landing Stage

Figure 5.6 shows the distances between the drone and the boat during landing. The drone could successfully land on the boat within the specified boundaries in these conditions. Figure 5.7 shows that the altitude controller is able to correct for the initial altitude error and approach the boat from above.

In more windy conditions, the drone occasionally struggled to simultaneously reach the target altitude and x-position to land on the boat. This difficulty lies in the fact that the changing winds continuously alters the velocities and positions of the drone and the boat, which in turn changes the location of P_3 according to Equations 4.28. Due to MAVLink's limited update frequency, this occasionally causes the drone to target an outdated P_3 that no longer aligns with the current state of the system, affecting the drone's landing accuracy.

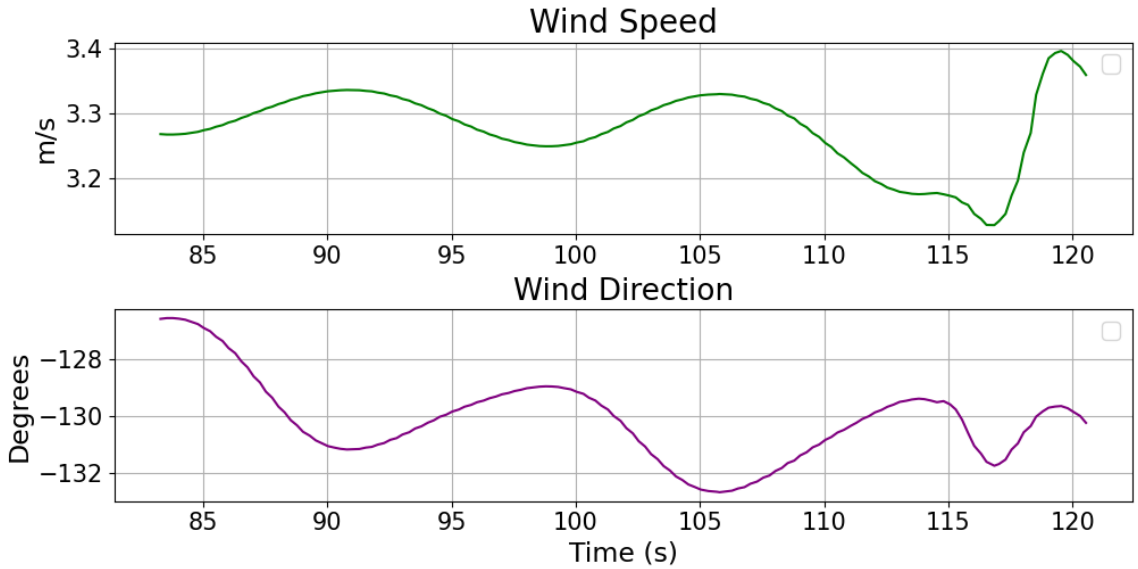


Figure 5.5: Wind speed and direction over time.

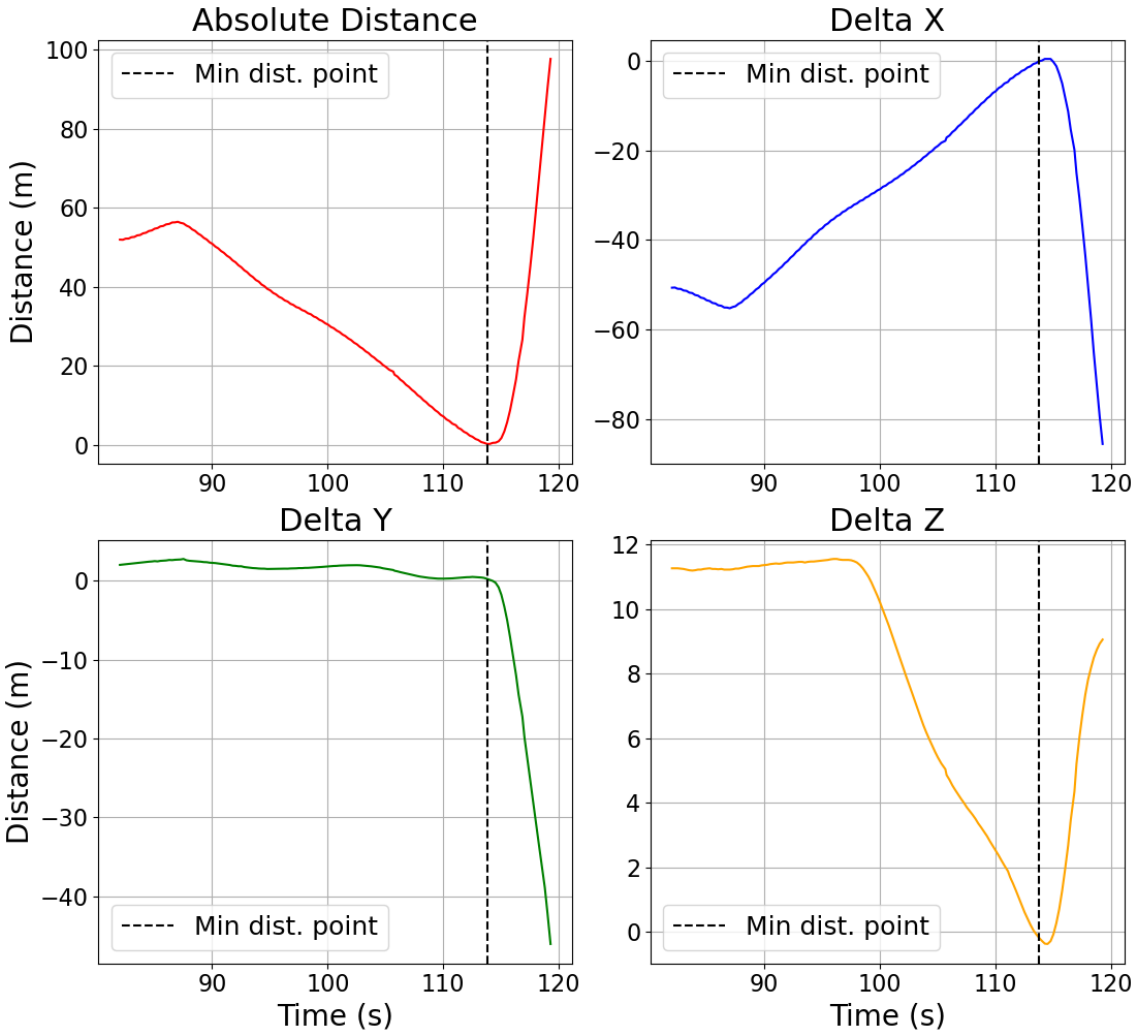


Figure 5.6: Distance data in windy conditions, where the dotted vertical line indicates the identified moment of touchdown by looking at the closest absolute distance between the drone and the boat. Distances between drone and boat when they were the closest: dx: 0.03m, dy: 0.25m, dz: 0.19m, absolute: 0.31m

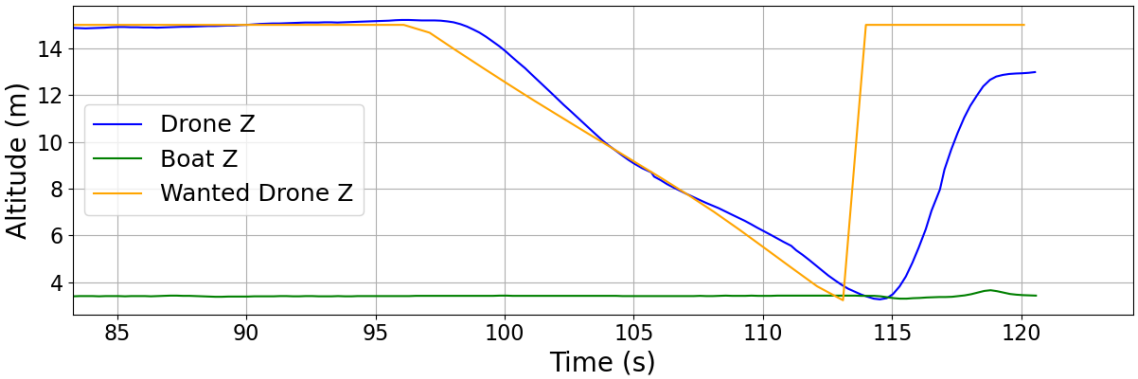


Figure 5.7: Shows the drone's and boat's altitude over time along with the drone's target altitude under windy conditions.

5.3 Testing With Movement Noise

Variations in boat movement were evaluated by introducing Gaussian noise to the boat's heading at each iteration as explained in Section 5.1. Similar to the wind tests, only the most severe variations that the drone could handle while still completing its tasks are included. Both the follow and landing stages were tested here as well.

5.3.1 Follow Stage

Figure 5.8 and 5.9 shows that the drone is successfully able to follow the boat's trajectory despite random heading fluctuations with zero mean and standard deviation of 30 degrees. Figure 5.10 shows that maintaining the desired distance to the boat becomes increasingly difficult with such erratic movements. At higher levels of variations, the drone would get too close to the boat, triggering repeated diversion maneuvers and failing to meet the success criteria.

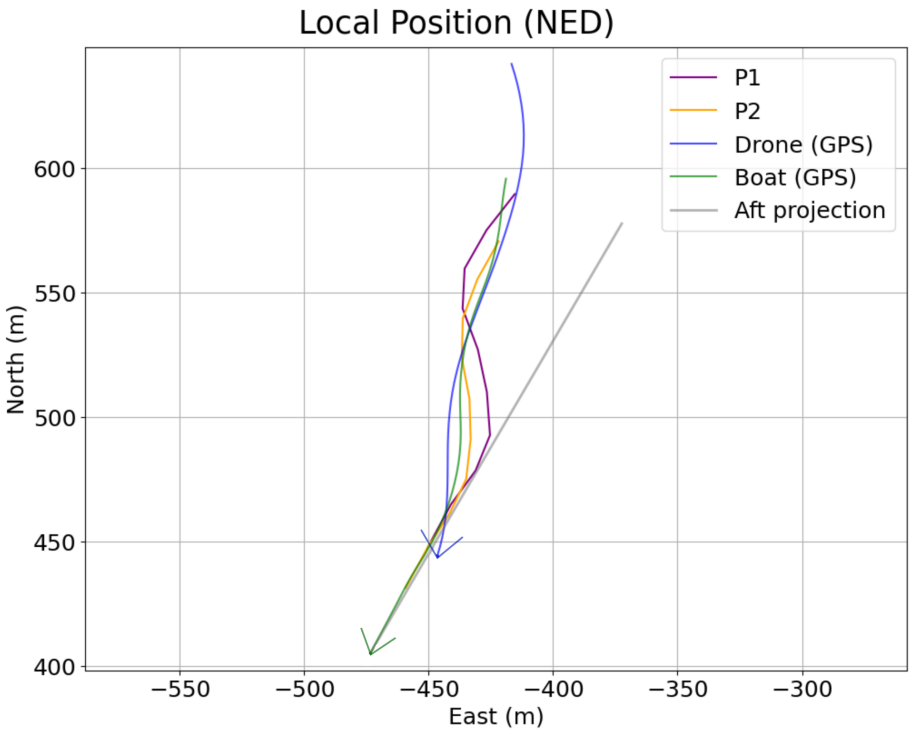


Figure 5.8: This figure shows the trajectories of the drone and the boat over time under boat movement disturbances. The current positions of both are marked with arrows. P1 and P2 indicate the speed set point and the descent start point, respectively. The "Aft projection" is a line extending straight back from the boat through P2. This figure can be compared to Figure 4.12, which shows the trajectories in the base case without disturbances.

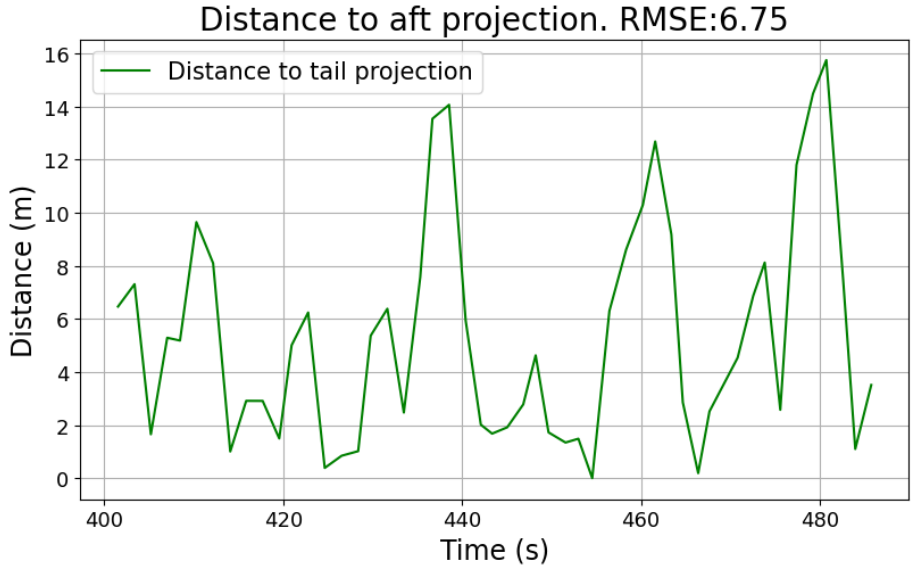


Figure 5.9: Shows the distance between the drone and the "Aft projection" over time as well as an average RMSE for this distance over the time steps included in the plot during boat movement variations.

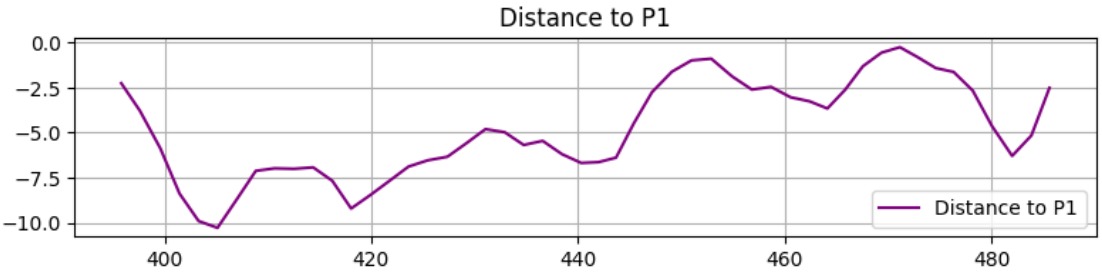


Figure 5.10: Drone's distance behind the speed setpoint, P_1 in meters over time with boat movement variations.

5.3.2 Landing Stage

Figure 5.11 shows that the drone is able to successfully land under heading variations with zero mean and a standard deviation of 3 degrees, though with limited margin. The system is particularly sensitive to variations in the boat's lateral position caused by the heading disturbances. This is evident from the altitude (z-axis) distance at the closest approach, which was 0.83 m, just below the limit of 1.25 m. During higher heading variations, the drone occasionally approached the boat from the side rather than from behind, failing to land on the boat. Figure 5.12 confirms that the drone was able to accurately maintain the desired glide slope. This result is expected as the heading variations primarily influence the lateral position of the boat, with minimal effect on the altitude control and the drone's ability to maintain a desired altitude.

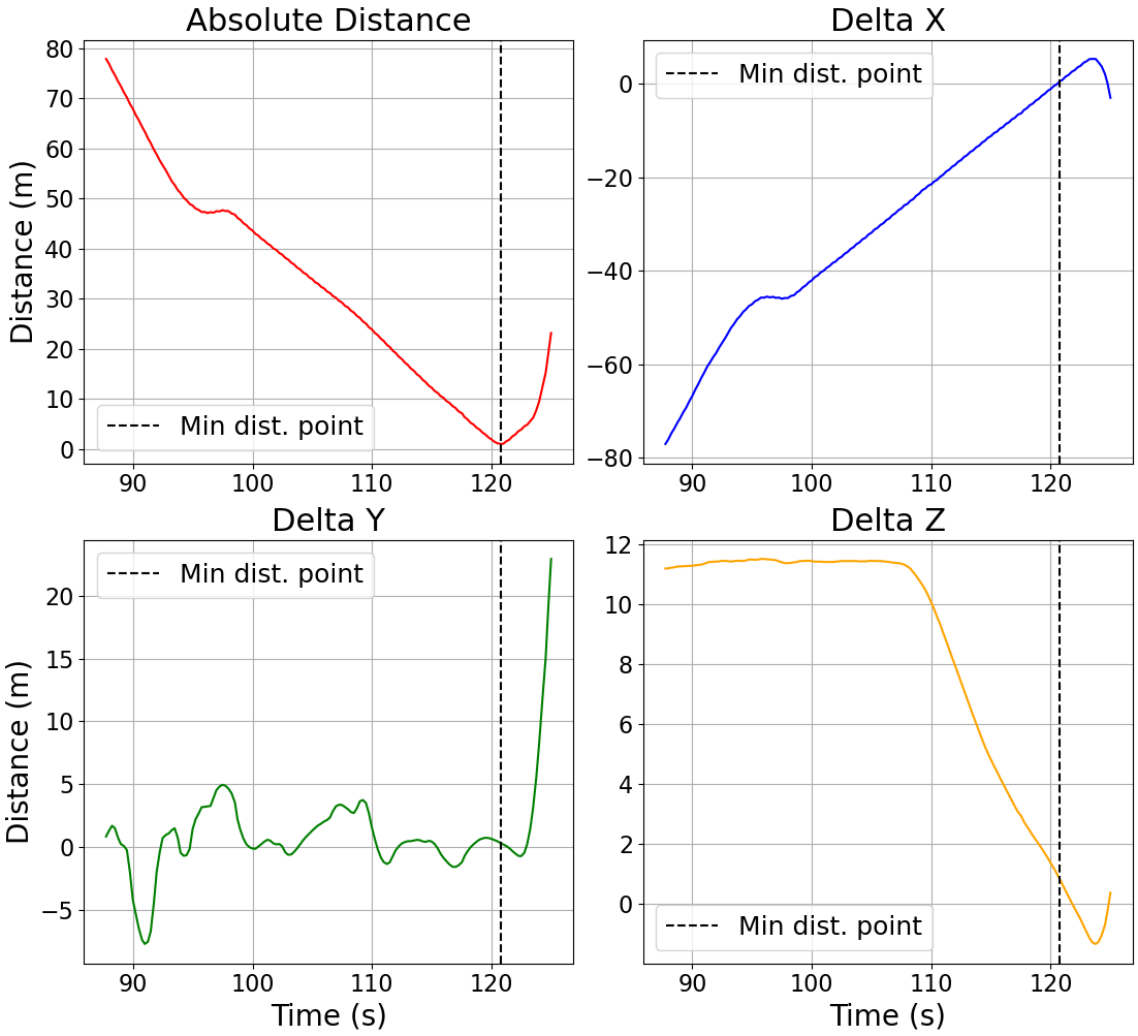


Figure 5.11: Distance data with boat movement variations, where the dotted vertical line indicates the identified moment of touchdown by looking at the closest absolute distance between the drone and the boat. Distances between drone and boat when they were the closest: dx: 0.46m, dy: 0.31m, dz: 0.83m, absolute: 1.00m.

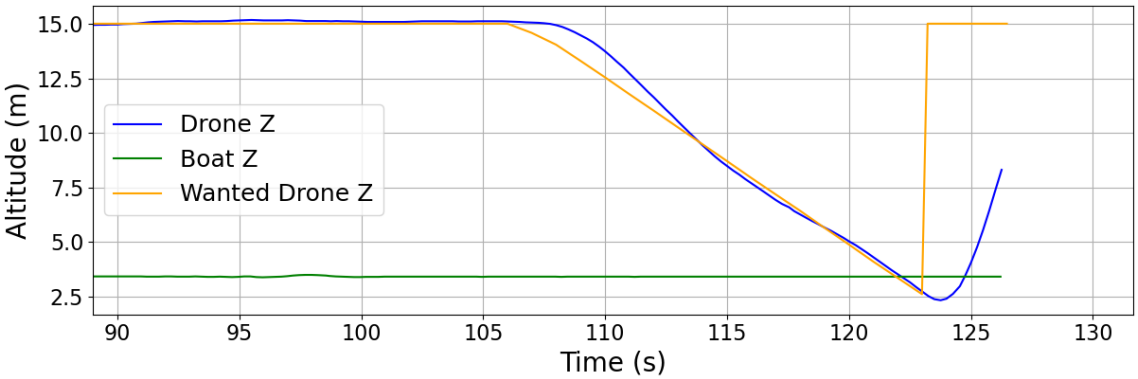


Figure 5.12: Shows the drone's and boat's altitude over time along with the drone's target altitude during disturbances to the boat's movement.

5.4 Testing With Altitude Noise

Altitude variations are introduced by adding a sine wave along with Gaussian noise to the boat's altitude to mimic wave behavior. Unlike previous tests which affected both the follow and landing stages, this test focuses solely on the landing stage. Since altitude variations do not affect the boat's trajectory, they are not expected to impact the drone's ability to follow the boat. Although the altitude does slightly affect the calculated position of P_1, the drone's ability to handle this is tested in the other sections.

5.4.1 Landing Stage

As shown in Figure 5.14, the drone struggles to maintain the desired glide slope when there are disturbances to the boat's altitude. This behavior is expected, as the altitude of P_3 is directly influenced by the altitude changes, causing continuous changes in the wanted altitude of the drone.

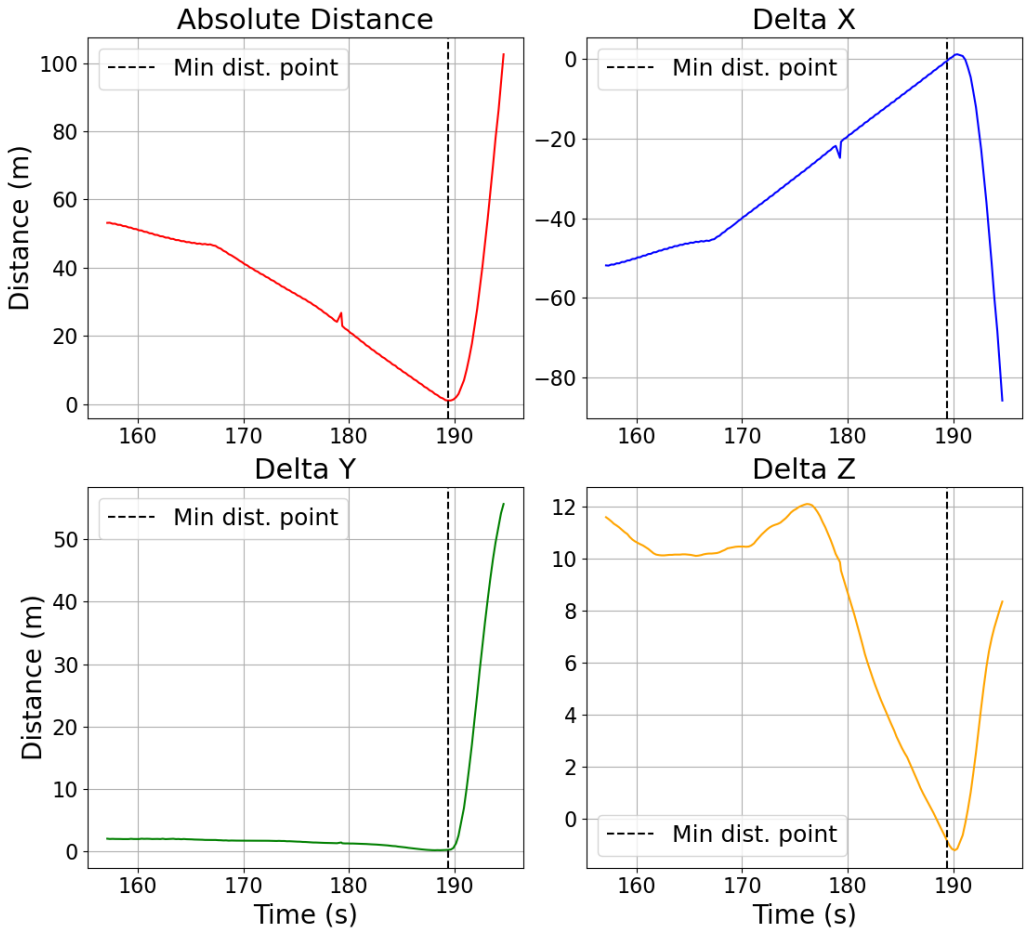


Figure 5.13: Distance data with boat altitude variations, where the dotted vertical line indicates the identified moment of touchdown by looking at the closest absolute distance between the drone and the boat. Distances between drone and boat when they were the closest: dx: 0.62m, dy: 0.22m, dz: 0.84m, absolute: 1.07m.

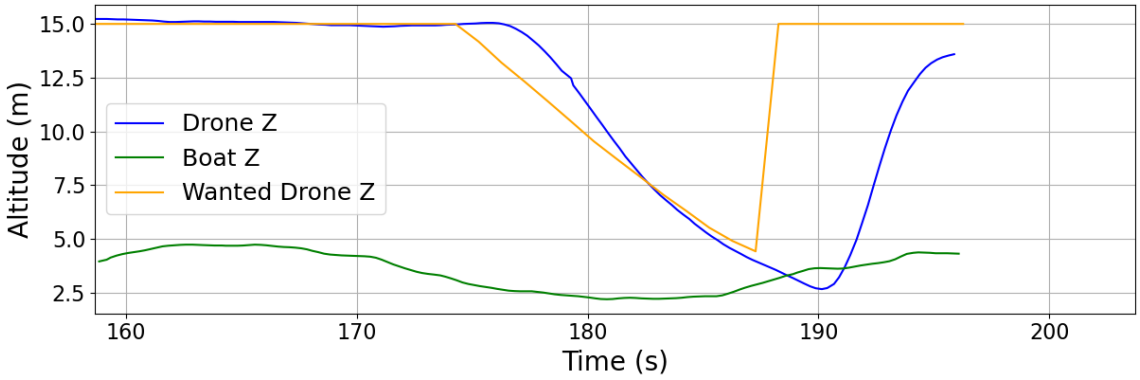


Figure 5.14: Shows the drone's and boat's altitude over time along with the drone's target altitude during disturbances to the boat's altitude.

5.5 Testing With Throttle Noise

Speed variations were evaluated by adding Gaussian noise to the drone's base throttle. The drone's base throttle was set to 1800 which corresponds to roughly 16.6 m/s under windless conditions. The boat attempted to align its speed with the drone's using a simple speed controller identical to the one that would be used by the drone from Section 4.4.1.2, but with negated values. This gives the same relative velocity between the two vehicles which is ultimately what is evaluated in these test. These tests were conducted in both the follow and landing stages. To isolate the throttle effects, the boat was commanded to move in a straight line for these tests.

5.5.1 Follow Stage

Figure 5.15 shows the boat's velocity over time, while 5.16 shows the drone's velocity under throttle fluctuations. Figure 5.17 displays the distances to P_1, P_2 and P_3 over time. For this test, Gaussian noise with zero mean and standard deviation of 200 was applied to the drone's throttle. For reference, a throttle of 1600 corresponds to 11.75 m/s and a throttle of 2000 corresponds to 19.0 m/s in conditions without wind.

The results show that the boat responds to the drone's velocity changes with a slight delay and occasional overshoot which is due to the use of a basic proportional controller. Despite these throttle variations, the drone is able to maintain an acceptable distance to P_1. Larger throttle perturbations led to more significant deviations and exceeded the limit of maintaining 20m distance to P_1. Beyond this threshold, the drone also frequently slowed down to the point of stalling, dropping significantly in altitude.

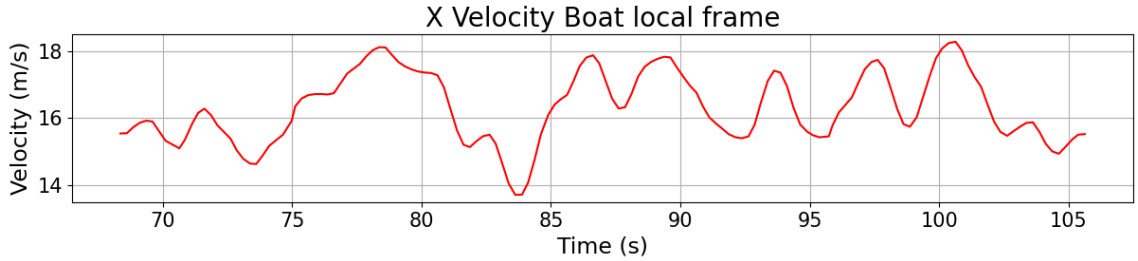


Figure 5.15: Boat velocity over time when trying to match the drone throttle variations using the speed planner.

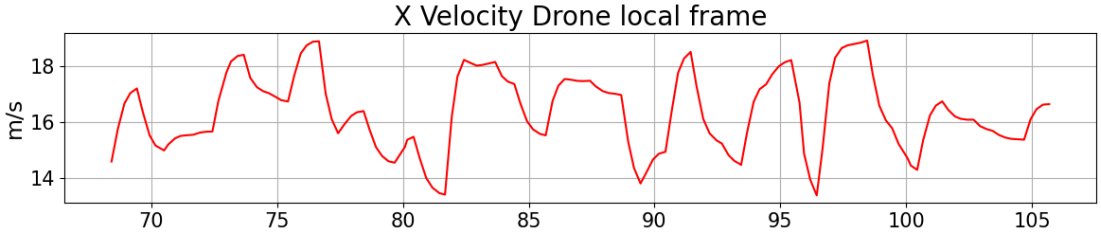


Figure 5.16: Drone velocities over time when subject to throttle variations

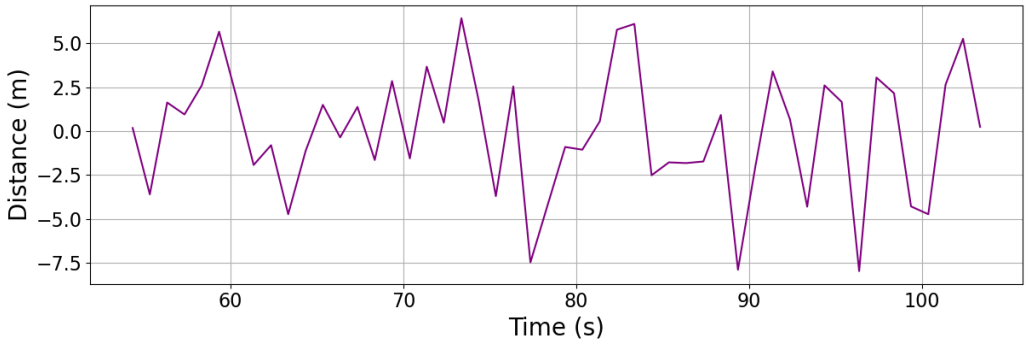


Figure 5.17: Drone's distance behind the speed setpoint, P_1 in meters over time with drone throttle variations.

5.5.2 Landing Stage

Figure 5.19 illustrates that the drone has some difficulties maintaining the desired glide ratio and altitude during the landing stage. Changing the drone's throttle instead of the boat's introduces two key effects:

- **Changes in P_3:** Changing the speed of the drone will influence the calculations of the landing point P_3, which in turn affects the target altitude and glide slope and ultimately the commanded sink rate.
- **Altitude changes:** Sudden throttle changes also temporarily affect the drone's lift before Ardupilot's control systems can respond, leading to momentary altitude changes.

Despite these challenges, Figure 5.18 shows that the drone is still able to perform a successful landing under throttle variations. The base input throttle was 1800 with

a standard deviation of 60. This corresponds to a flight speed between 15.1 m/s to 18.0 m/s within one standard deviation. Figure 5.19 shows that the altitude varies a bit up and down which is due to effect on lift that throttle changes has. Overall, this demonstrates that the system can tolerate speed variations while still completing the landing phase consistently. Additionally, Figures 5.21 and 5.20 shows how the speed of the drone changes with the throttle variations and how the boat attempts to match this according to the impact speed of 2 m/s.

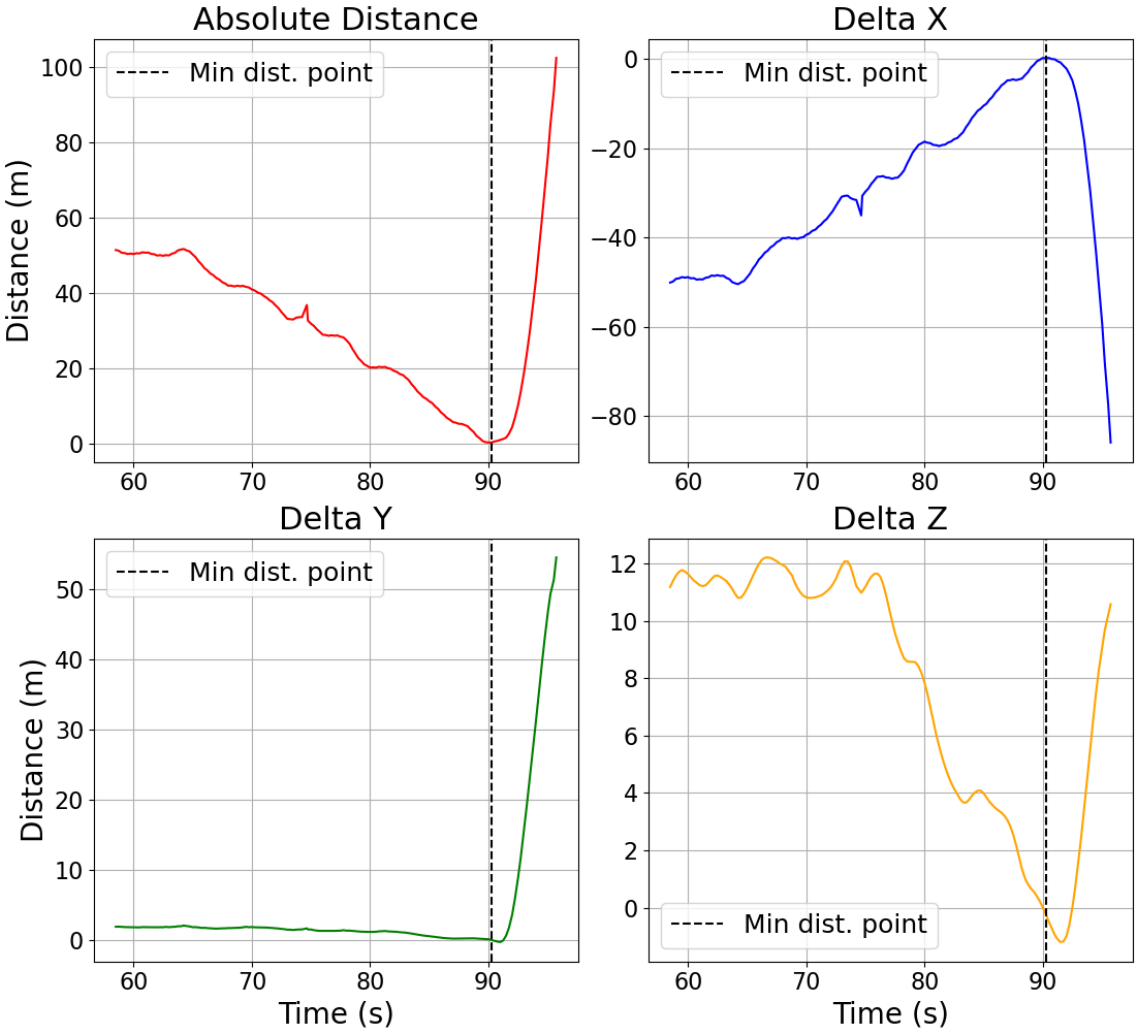


Figure 5.18: Distance data with drone throttle variations, where the dotted vertical line indicates the identified moment of touchdown by looking at the closest absolute distance between the drone and the boat. Distances between drone and boat when they were the closest: dx: 0.14m, dy: 0.10m, dz: 0.22m, absolute: 0.28m

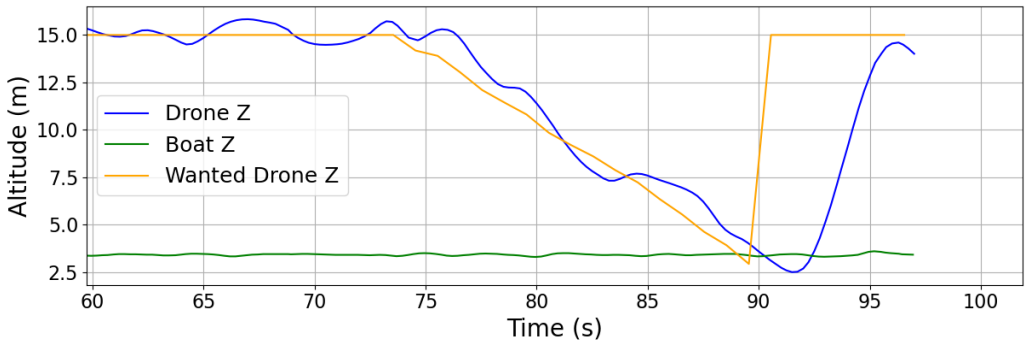


Figure 5.19: Shows the drone's and boat's altitude over time along with the drone's target altitude during disturbances to the drone's throttle.

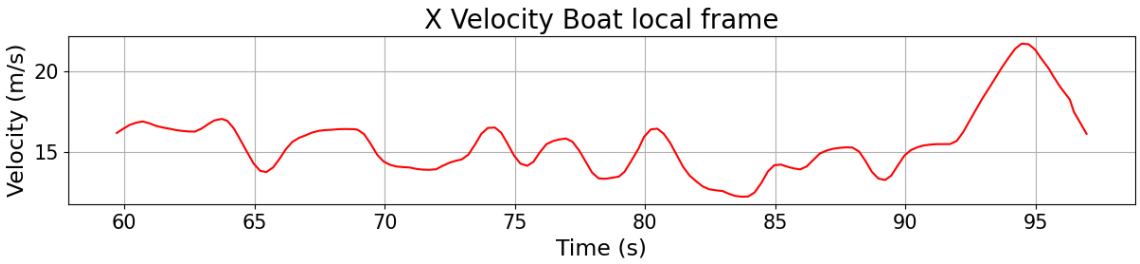


Figure 5.20: Boat velocity over time when trying to match the drone throttle variations according to the target impact speed of 2 m/s

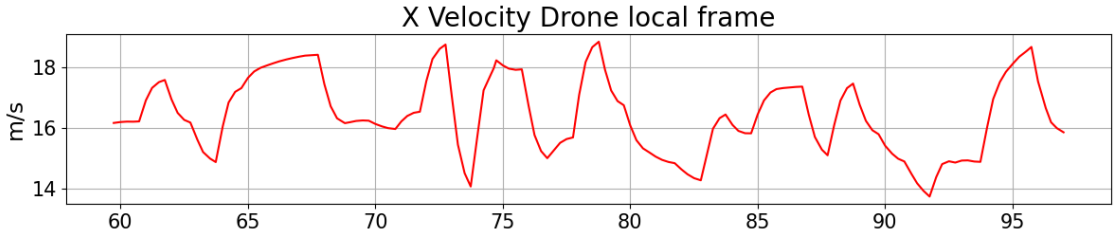


Figure 5.21: Drone velocities over time when subject to throttle variations

5.6 Kalman Filter Evaluation

Qualitative testing of the boat positioning EKF was done using two sets of AIS data, both described separately in the following sections. The first dataset is recorded AIS data from the SSRS SAR vessel *Josephine*, a 13-ton displacement sea rescue vessel. Data was collected under calm conditions, with minimal wave disturbance. The second set of data is an AIS recording of a sea-going passenger vessel, *Rivö*, traveling in the archipelago west of Gothenburg. The weather was moderate and the vessel was traveling a standard passenger route.

Due to the absence of ground truth data, a strictly quantitative assessment of the EKF's accuracy is not possible. However, qualitative analysis using realistic AIS

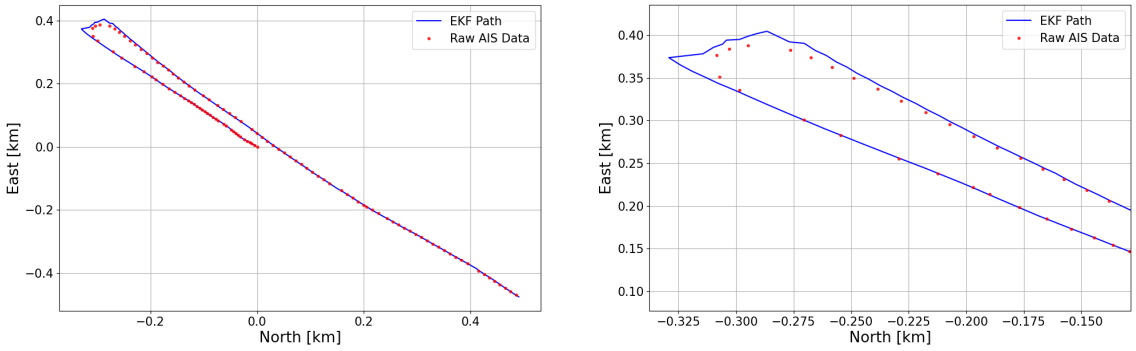
data from actual vessel operations provides meaningful insights into the filter's behavior under dynamic conditions. This allows for preliminary validation of the model's robustness and its ability to capture key vessel dynamics such as positioning, heading-course discrepancies, velocity tracking, and maneuvering behavior.

5.6.1 SAR Vessel - Qualitative Analysis

During data gathering, the vessel maintained a steady pace and attempted to hold a consistent heading, mimicking expected behavior during an autonomous landing scenario, where the captain would assist by maintaining a stable heading.

The test route traversed the Göta Älv river delta, where a steady current introduced a consistent deviation between the vessel's heading and its course over ground. This discrepancy presents a relevant challenge for testing the EKF's ability to reconcile heading and velocity measurements under environmental disturbance.

To further test the filter's performance under nonlinear dynamics, the dataset includes a sharp turn performed at relatively high speed. Although no ground truth reference was available, the evaluation offers a preliminary validation of the EKF's ability to track states in realistic maritime conditions.



(a) Measured positions over time and filter trajectory.

(b) Zoomed-in plot focused on the sharp turn.

Figure 5.22: Filter trajectory compared with measured AIS data. Left: Full path. Right: Zoomed-in view showing behavior in a sharp turn.

Figure 5.22 displays the measured coordinates overlaid on the filter's path. It displays the capability to track the trajectory in real time. In the sharp turn, there are some discrepancies between the output from the EKF and the measured coordinates.

Tracking the velocity is important to be able to smoothly match the vessels speed. Figure 5.23 shows how well the velocity tracks throughout the measured period. The EKF tracks forward as well as lateral velocity, u and v. Splitting up the absolute velocity from Figure 5.23 into its components gives the plot in Figure 5.24. Here it is visible that the lateral velocity component is larger in the beginning than at the end, where the absolute velocity is higher. This is consistent with the fact that the vessel would need to compensate for a greater discrepancy in course and true

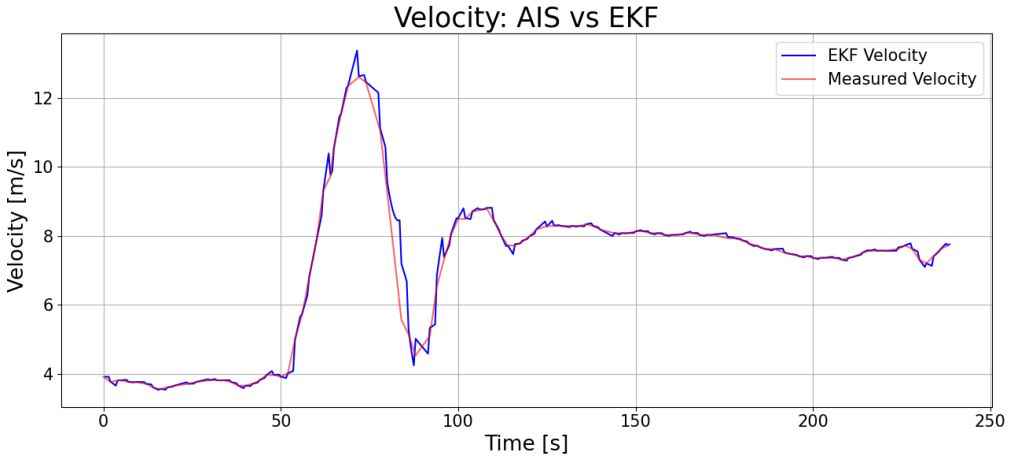


Figure 5.23: Filtered vs measured velocities

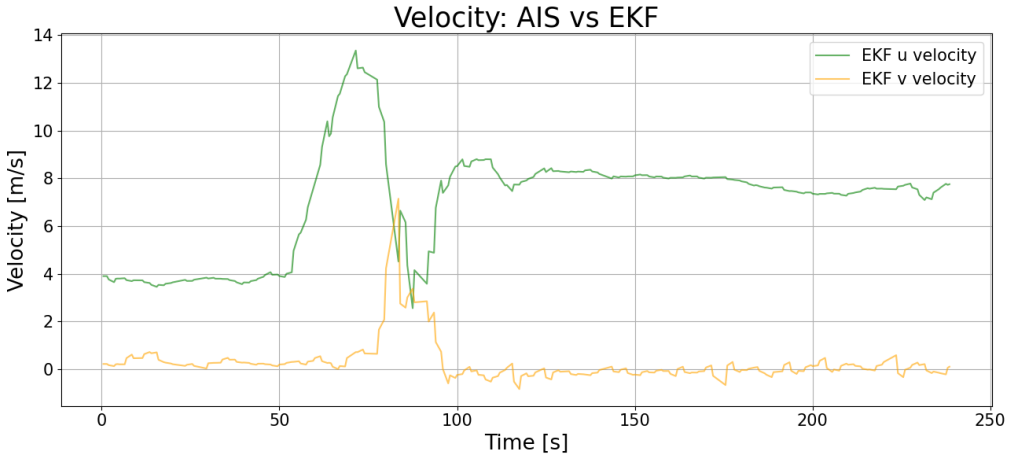


Figure 5.24: Filtered longitudinal and lateral velocity, u and v

heading due to the constant current. This discrepancy is also visible in the course and heading derived by the EKF. Figure 5.25 shows this clearly in the initial phase. The unobserved states (x and y acceleration, and yaw rate) are modeled as Gaussian random walks and assumed constant between measurements. Despite being unmeasured directly, these states influence the system dynamics and contribute to the model's performance. The estimated accelerations are shown in Figure B.1, and the yaw rate is shown in Figure B.2.

5.6.2 Passenger Vessel - Qualitative Analysis

The data from the passenger vessel is useful to demonstrate the robustness of the EKF in real-world conditions. Robustness here refers to the filter's ability to maintain stable state estimates despite frequent changes in vessel dynamics, stops at multiple ports, and gaps in AIS updates. Figure 5.26 shows the filter's position over a time period of 47 minutes.

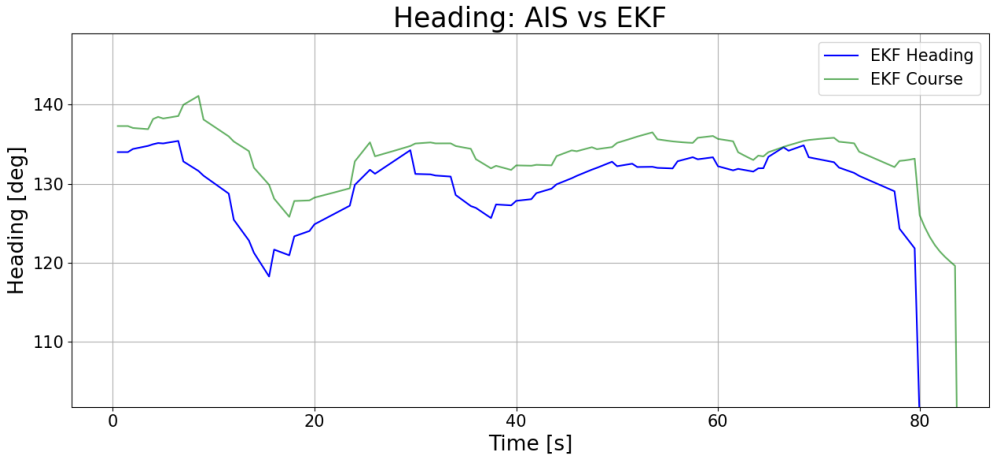


Figure 5.25: Zoomed in plot of heading vs course from the EKF

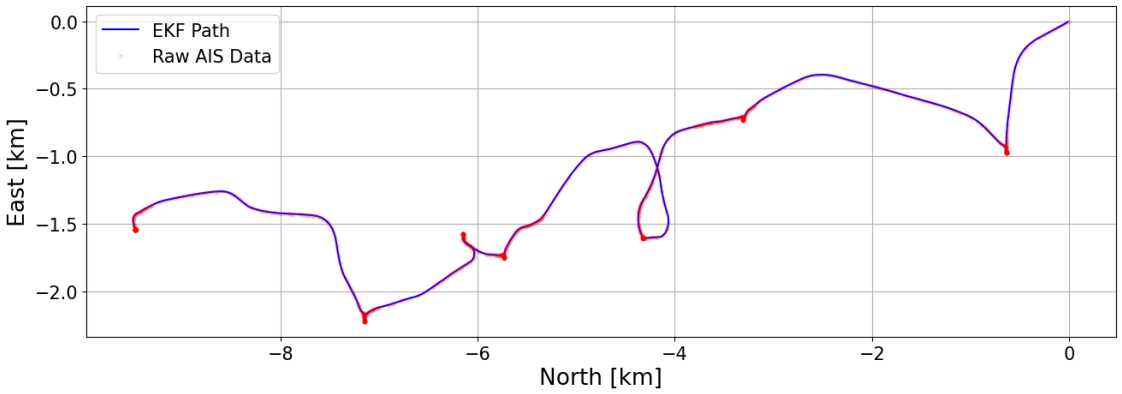
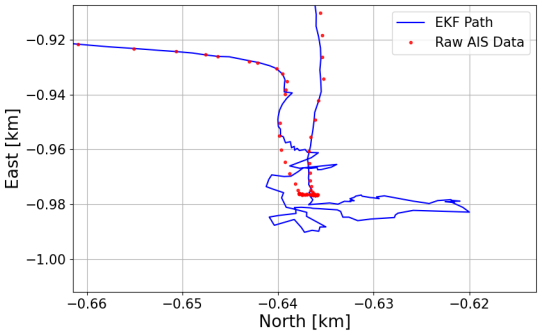
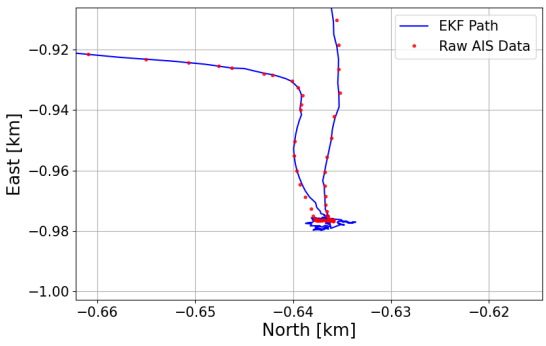


Figure 5.26: Raw AIS positional data and filters trajectory. Passenger vessel.

As described in section 4.3.3.1, when the vessel has a low velocity, the measurement model is adapted to the changing course characteristics. Figure 5.27 displays the effect this change in expected standard deviation has.



(a) No change in σ_{deg} at low velocities.



(b) Increased σ_{deg} for velocities below 2 m/s.

Figure 5.27: Comparison of filter trajectories with and without adaptive covariance based on vessel velocity.

6
Conclusion

The goal of this thesis was to expand the autonomous capabilities of SSRS's fixed-wing drones by developing and evaluating a system capable of autonomously landing on a moving sea vessel. This work is a step towards fully autonomous maritime drone operations, reducing reliance on human operators, improving safety and performance in difficult conditions. To achieve this, an EKF was implemented to estimate the boat's position under noisy or irregular measurements. An additional algorithm was developed to track and follow the boat while awaiting suitable landing conditions. Finally, a landing algorithm was also developed that guides the drone towards a computed landing point using a specified glide ratio and impact speed.

The developed system shows strong potential for both the localization and landing tasks. The EKF can be tuned for real world deployment, improving the frequency and reliability of landing algorithms updates in situations of noisy or irregular measurements. The system demonstrates successful following and landing behavior, indicating that the approach works in theory. By enabling autonomous mid sea recovery, this system contributes directly to the EOS project's aim to execute streamlined and fast reconnaissance missions in time critical SAR scenarios.

The simulation environment allowed for isolated and controlled testing of realistic SAR situations, including various disturbances. The system consistently managed simulated disturbances in wind, boat movement, boat altitude and speed to a practical extent. These results suggest that the system can operate consistently and reliably in a range of real world scenarios.

The system still remains sensitive to large or abrupt variations. Beyond certain thresholds, the drone can't consistently perform the designated tasks. For a real world context, this corresponds to the drone being unable to handle strong winds, large waves, or unpredictable currents that significantly disturb the boat's or drone's movement. Addressing these challenges is essential for robust deployment in all SAR situations.

Future work

A next step could be to implement the developed system on the real drone hardware. Specifically. This involves deploying the algorithms to the onboard companion computer and ensuring proper communication with the flight controller, navigation sensors, and external systems such as AIS. Integrating the system with the camera-

based boat localization developed in parallel to this thesis is also required to enable fully independent autonomous landings that don't rely on pre-supplied information about the target vessel.

Since the Kalman filter was not tuned in this thesis, future work should also include parameter tuning of the filter based on live measurements. Gathering data with ground truth is needed to improve estimations and benchmark the performance of the filter.

Real world testing in representative flying conditions is critical to validate the system's performance in actual wind, wave and current disturbances. Such testing will provide insight into operational challenges and help in guiding further parameter tuning of the algorithms and improve robustness and consistency.

The current speed and altitude planners are implemented as basic proportional controllers. These could be extended to include an integral and a derivative part to improve responsiveness and precision in managing the horizontal speed and the descent rates during the follow and landing stages.

Finally, future work could focus on improving the system's robustness in more extreme or unpredictable environmental variations like wind, waves and current, extending its use cases and reliability in more demanding SAR situations.

Bibliography

[1] Sjöräddningssälskapet. *Fler söker hjälp i tid: statistiken från 2024*. Accessed: 2025-01-21. 2024. URL: https://www.sjoraddning.se/artiklar/fler-soker-hjalp-i-tid-statistiken-fran-2024-0.

[2] NASA. *Guidance, Navigation, and Control (GNC)*. Accessed: 2025-02-24. 2023. URL: https://www.nasa.gov/smallsat-institute/sst-soa/guidance-navigation-and-control/?utm_source=chatgpt.com.

[3] ArduPilot Development Team. *ArduPilot: Open Source Autopilot*. https://ardupilot.org. Accessed: 2025-05-12. 2025.

[4] ArduPilot Development Team. *Extended Kalman Filter — ArduPlane documentation*. Accessed: 2025-02-24. 2023. URL: https://ardupilot.org/plane/docs/common-apm-navigation-extended-kalman-filter-overview.html.

[5] ArduPilot Development Team. *Flight Modes — Plane documentation*. Accessed: 2025-01-22. 2023. URL: https://ardupilot.org/plane/docs/flight-modes.html#flight-mode-list.

[6] MAVLink Development Team. *MAVLink Developer Guide*. Accessed: March 14, 2025. 2025. URL: https://mavlink.io/en/.

[7] ArduPilot Dev Team. *L1 navigation overview*. Accessed: March 13, 2025. 2024. URL: https://ardupilot.org/plane/docs/navigation-tuning.html.

[8] ArduPilot Dev Team. *TECS (Total Energy Control System) for Speed and Height Tuning Guide*. Accessed: March 13, 2025. 2024. URL: https://ardupilot.org/plane/docs/tecs-total-energy-control-system-for-speed-height-tuning-guide.html.

[9] MarineTraffic Support. *FAQ and Troubleshooting*. Accessed: April 24, 2025. 2025. URL: https://support.marinetraffic.com/en/collections/11599008-faq-and-troubleshooting.

[10] U.S. Coast Guard Navigation Center. *AIS Class A Reports*. Accessed: April 24, 2025. 2025. URL: https://www.navcen.uscg.gov/ais-class-a-reports.

[11] U.S. Coast Guard Navigation Center. *How AIS Works*. Accessed: April 24, 2025. 2025. URL: https://www.navcen.uscg.gov/how-ais-works?pageName=vhfnb.

[12] Martin Redoutey et al. "Efficient Vessel Tracking with Accuracy Guarantees". In: Dec. 2008, pp. 140–151. ISBN: 978-3-540-89902-0. DOI: 10.1007/978-3-540-89903-7_13.

[13] Lantmäteriet. *WGS 84*. Accessed: April 22, 2025. 2025. URL: https://www.lantmateriet.se/sv/geodata/gps-geodesi-och-swepos/Referenssystem/Tredimensionella-system/WGS-84/.

[14] D. Zhang and X. Wang. "Autonomous Landing Control of Fixed-wing UAVs: from Theory to Field Experiment". In: *Journal of Intelligent & Robotic Systems* 88 (2017), pp. 619–634. DOI: 10.1007/s10846-017-0512-y. URL: https://doi.org/10.1007/s10846-017-0512-y.

[15] Thor Fossen. *Handbook of Marine Craft Hydrodynamics and Motion Control*. Wiley, Apr. 2021. ISBN: 978-1119575047. DOI: 10.1002/9781119994138.

[16] Tristan Perez and Thor I. Fossen. "Kinematic Models for Manoeuvring and Seakeeping of Marine Vessels". In: *Modeling, Identification and Control* 28.1 (2007), pp. 19–30. DOI: 10.4173/mic.2007.1.3.

[17] U.S. Shukla and Pravas Mahapatra. "The Proportional Navigation Dilemma—Pure or True?" In: *Aerospace and Electronic Systems, IEEE Transactions on* 26 (Apr. 1990). Accessed: 2025-02-13, pp. 382 –392. DOI: 10.1109/7.53445.

[18] Mirza Hodžić and Naser Prljaca. "Missile Guidance using Proportional Navigation and Machine Learning". In: *Journal of Engineering Research and Sciences* 3 (Mar. 2024). DOI: 10.55708/js0303003.

[19] Qian Peng, Jianguo Guo, and Jun Zhou. "Integrated guidance and control system design for laser beam riding missiles with relative position constraints". In: *Aerospace Science and Technology* 98 (2020). Accessed: 2025-02-13, p. 105693. ISSN: 1270-9638. DOI: https://doi.org/10.1016/j.ast.2020.105693. URL: https://www.sciencedirect.com/science/article/pii/S1270963819305358.

[20] R. Craig Coulter. *Implementation of the Pure Pursuit Path Tracking Algorithm*. Technical Report CMU-RI-TR-92-01. Carnegie Mellon University, Robotics Institute, 1992. URL: https://www.ri.cmu.edu/pub_files/pub3/coulter_r_craig_1992_1/coulter_r_craig_1992_1.pdf.

[21] Federal Aviation Administration. *Course Notes – Normal Landing*. n.d. URL: https://www.faasafety.gov/files/gslac/courses/content/35/562/Course%20Notes%20-%20Normal%20Landing.pdf (visited on 04/15/2025).

[22] Solomon Gudeta and Ali Karimoddini. "Design of a Smooth Landing Trajectory Tracking System for a Fixed-wing Aircraft". In: (July 2021). Accessed: 2025-04-15. DOI: 10.48550/arXiv.2107.05803.

A

Statistical Filtering

When the full system state cannot be directly observed, statistical filtering provides a robust approach for estimating it from noisy and partial measurements. It allows the fusion of multiple probability densities, producing a unified estimate from various sensors. This chapter introduces Bayesian filtering and its linear Gaussian counterpart, including the Kalman filter and its nonlinear extension, the Extended Kalman Filter.

A.1 Recursive Bayesian Estimation

The aim is to calculate the probability distribution of the hidden state x given a set of observations gathered over time $\mathbf{z}_{0:k} = \{z_0, z_1, ..., z_k\}$ that are all dependent on the hidden state. For a discrete-time dynamic system, the goal is to estimate

$$p(x_k|z_{1:k}) \tag{A.1}$$

The process can be described as a hidden Markov model, which gives Bayesian filtering two key properties

Markov property - The current state depends only on the immediate previous state and is independent of all other earlier states:

$$p(x_k \mid x_{0:k-1}, z_{0:k-1}) = p(x_k \mid x_{k-1}) \tag{A.2}$$

Conditional independence of measurements - Each measurement depends only on the current state and is independent of all prior states

$$p(z_k \mid x_{0:k}, z_{0:k-1}) = p(z_k \mid x_k) \tag{A.3}$$

Using these assumptions, we can recursively compute the posterior $p(x_k|z_k)$ by applying Bayes' rule

$$p(x_k|z_k) = \frac{p(z_k|x_k), p(x_k|z_{k-1})}{p(z_k|z_{k-1})} \tag{A.4}$$

Here, $p(x_k|z_{k-1})$ is a prior obtained by integrating over the previous state

$$p(x_k|z_{k-1}) = \int p(x_k|x_{k-1}), p(x_{k-1}|z_{k-1}), dx_{k-1} \tag{A.5}$$

This yields the two-step recursive filtering process

$$\begin{aligned} \text{Prediction:} \quad p(x_k|z_{k-1}) &= \int p(x_k|x_{k-1}), p(x_{k-1}|z_{k-1}), dx_{k-1} \\ \text{Update:} \quad p(x_k|z_k) &= \frac{p(z_k|x_k), p(x_k|z_{k-1})}{\int p(z_k|x_k), p(x_k|z_{k-1}), dx_k} \end{aligned} \tag{A.6}$$

The prediction step uses the Chapman-Kolmogorov equation to propagate the belief about the state forward in time, while the update step incorporates the latest measurement using Bayes' rule. Together, these steps allow for the recursive estimation of $p(x_k|z_{0:k})$as new data becomes available.

A.2 Kalman Filter

Recursive Bayesian filtering provides a general method for estimating the state of a system over time. When the process and measurement models are linear and the noise is Gaussian, the filter equations have an exact and optimal (closed-form) solution. This special case is known as the Kalman filter, which performs Bayesian filtering in a way that minimizes the average error in the state estimates under these conditions.

The system is modeled as follows

$$\begin{aligned} x_k &= Ax_{k-1} + Bu_{k-1} + w_{k-1} \\ z_k &= Hx_k + v_k \end{aligned} \tag{A.7}$$

Here, measurements are denoted z_k, x_k is the system state, u_{k-1} is the control input, and w_{k-1} is the process noise. A is the nxn state transition matrix for the linear system that is, in the absence of control input and noise, able to map the previous state x_{k-1} to the current state x_k. Likewise, B represents the optimal control input matrix, mapping control inputs to state changes.

The noise terms, w and v, are modeled as independent, zero-mean Gaussian distributions with known covariances

$$\begin{aligned} w_{k-1} &\sim \mathcal{N}(0, Q_{k-1}) \\ v_k &\sim \mathcal{N}(0, R_k) \end{aligned} \tag{A.8}$$

This model satisfies the assumptions of recursive Bayesian estimation presented in Equation A.6 with the specific notations for the update cycle shown in fig A.1. At each timestep, it maintains an estimate of the state $\hat{x}_k$ and its associated uncertainty represented by the covariance matrix P_k.

Prediction Step
This step estimates the state at time k using the previous estimate $\hat{x}_{k-1}$ and control input u_{k-1}

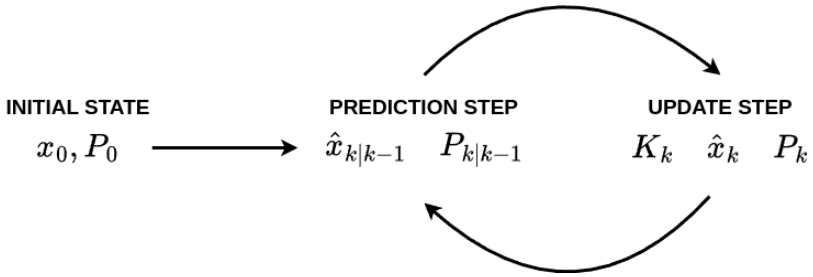


Figure A.1: Update cycle with standard Kalman filter equations.

$$\begin{aligned} \hat{x}_{k|k-1} &= A\hat{x}_{k-1} + Bu_{k-1} \\ P_{k|k-1} &= AP_{k-1}A^\top + Q \end{aligned} \tag{A.9}$$

$\hat{x}_{k|k-1}$ is the prior state estimate before incorporating the new measurement. $P_{k|k-1}$ is the predicted error covariance, which accounts for uncertainty from the previous state and the process noise.

Update Step
In the update step, the filter adjusts the predicted state based on the new measurement z_k

$$\begin{aligned} S_k &= HP_{k|k-1}H^\top + R \\ K_k &= P_{k|k-1}H^\top S_k^{-1} \\ \hat{x}_k &= \hat{x}_{k|k-1} + K_k(z_k - H\hat{x}_{k|k-1}) \\ P_k &= (I - K_kH)P_{k|k-1} \end{aligned} \tag{A.10}$$

The innovation covariance S_k captures the combined uncertainty of the prediction and the measurement. The Kalman gain K_k determines the weighting between the predicted state and the measurement. This gain is computed to minimize the posterior covariance, or expected squared error. This is what ensures a closed-form solution when possible. The state estimate is then updated using the measurement residual, and the error covariance P_k is reduced to reflect the improved estimate accuracy.

Together, the prediction and update steps allow the Kalman filter to recursively estimate the system state over time, incorporating new measurements to continuously refine its estimate.

A.3 Extended Kalman Filter

The kalman filter provides an optimal recursive estimator under the assumption of linear dynamics and Gaussian noise. However, most real-world system exhibit nonlinear behavior either in the dynamics or in the observational model. This is

especially true for maritime tracking, where the tracked vessels are subject to turbulent wave dynamics, complex hydrodynamical effects and wind. Control inputs generally also give rise to nonlinear behavior in the vessel. To handle such systems, the Extended Kalman Filter (EKF) introduces a generalization. It does so by locally linearizing the nonlinear functions around the current estimate, enabling the use of Kalman filtering in nonlinear applications.

The system is now described by nonlinear functions f and h that determine the state transition and measurement models, respectively:

$$\begin{aligned} x_k &= f(x_{k-1}, u_{k-1}) + w_{k-1} \\ z_k &= h(x_k) + v_k \end{aligned} \tag{A.11}$$

Here $f(x_{k-1}, u_{k-1})$ contains the process model that now can have complex dynamics that cannot be captured with a fixed transition matrix A. The term $h(x_k)$ contains the observation model, which uses the current state to predict the incoming observations. The noise models are still Gaussian with zero mean and uncorrelated as described in Equation A.8.

The EKF still maintains the recursive Bayesian estimation structure outlined in Equation A.6. It approximates the belief propagation by linearizing the nonlinear functions using a first-order Taylor expansion about the current mean estimate. The linearization enables the filter to approximate the posterior estimate as a Gaussian. The posterior is characterized by its mean and covariance, which change over time through the prediction and update steps.

Prediction step In the EKF, the prediction step evaluates the nonlinear process model at state estimate $k-1$. The predicted covariance is computed by propagating uncertainty through the linearized dynamical model:

$$\begin{aligned} \hat{x}_{k|k-1} &= f(\hat{x}_{k-1}, u_{k-1}) \\ F_{k-1} &= \left.\frac{\partial f}{\partial x}\right|_{\hat{x}_{k-1}, u_{k-1}} \\ P_{k|k-1} &= F_{k-1} P_{k-1} F_{k-1}^{\top} + Q \end{aligned} \tag{A.12}$$

What differs from the linear filter is that the Jacobian matrix F_{k-1} serves as a linear approximation of the nonlinear dynamics, linearized around $\hat{x}_{k|k-1}$.This replaces the fixed A matrix from the KF. As in the Kalman filter, the predicted covariance accounts for both prior uncertainty and process noise

Update step The update step also linearizes the measurement model around the predicted state:

$$
\begin{aligned}
H_k &= \left.\frac{\partial h}{\partial x}\right|_{\hat{x}_{k|k-1}} \\
S_k &= H_k P_{k|k-1} H_k^\top + R \\
K_k &= P_{k|k-1} H_k^\top S_k^{-1} \\
\hat{x}_k &= \hat{x}_{k|k-1} + K_k(z_k - h(\hat{x}_{k|k-1})) \\
P_k &= (I - K_k H_k) P_{k|k-1}
\end{aligned}
\tag{A.13}
$$

The innovation term, $z_k - h(\hat{x}_{k|k-1})$, represents the difference between the observed and predicted measurements. The Jacobian H_k replaces the fixed observation matrix H, now describing how much each measurement would change in response to small changes in the state, evaluated at the current predicted state. The EKF adjusts the predicted state estimate by multiplying the measurement residual with the Kalman gain, shifting the estimate toward the observed measurement. At the same time, it updates the state covariance to reflect the reduced uncertainty resulting from the new information.

The EKF keeps the Bayesian recursive structure. While the filter is no longer optimal in the statistical sense due to the linearization, it is still effective when the system's nonlinearities are somewhat smooth. The quality of the estimations depends on how well the linear approximation describes the system behavior, and performance may worsen if the linearization is poor or the prediction is far from the true state.

B

Plots

B.1 Kalman Filter Rvaluation

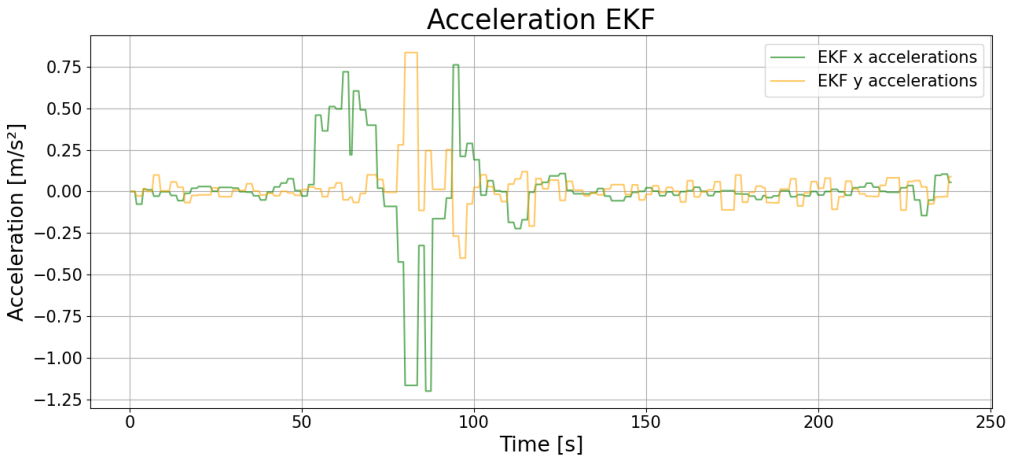


Figure B.1: SAR vessel testing data showing acceleration states over time.

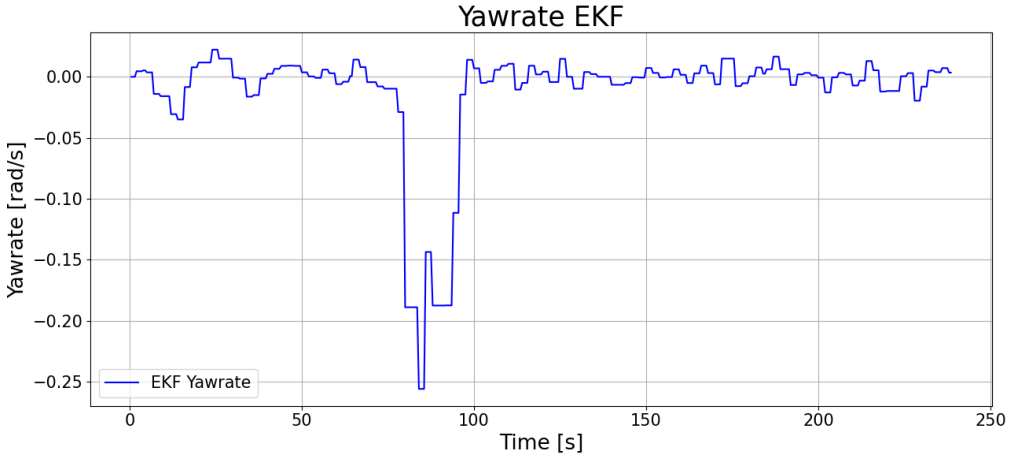


Figure B.2: SAR vessel testing data showing yaw rate over time

DEPARTMENT OF ELECTRICAL ENGINEERING E2
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden
www.chalmers.se

CHALMERS
UNIVERSITY OF TECHNOLOGY