

Diploma Project

Sailing Simulation

Ioanna Hadjiandreou

UNIVERSITY OF CYPRUS



DEPARTMENT OF COMPUTER SCIENCE

May 2026

UNIVERSITY OF CYPRUS
Faculty of Pure and Applied Sciences
DEPARTMENT OF COMPUTER SCIENCE

Sailing Simulation

Ioanna Hadjiandreou

Supervisor

Chrysanthou Yiorgos

The Thesis was submitted in partial fulfillment of the requirements for obtaining
the Computer Science degree at the Department of Computer Science
of the University of Cyprus

May 2026

Acknowledgements

I would like to express my sincere gratitude to my supervisor, Chysanthou Yiorgos, for his guidance, constructive feedback, and continuous support throughout the development of this thesis. His expertise and insights were instrumental in shaping both the technical and conceptual aspects of this work.

I am also grateful to the faculty of Computer Science of the University of Cyprus for providing the necessary resources and academic environment that facilitated the completion of this research.

I would like to extend my sincere thanks to Marios Michael, Scout Chief and captain of our boats, for his valuable assistance and for sharing his practical sailing expertise, which significantly contributed to the accuracy and realism of the simulation.

Finally, I would like to acknowledge all individuals who contributed, directly or indirectly, to the successful completion of this thesis.

¹I declare that this dissertation and any accompanying software are my own original work, conducted in full compliance with the University of Cyprus Code of Conduct for Research (<https://www.ucy.ac.cy/legislation/kodikas-deontologias-kai-politikes/>), and with full accountability on my part for the accuracy, integrity, and validity of all content.”

ABSTRACT

This thesis presents the design and implementation of a real-time interactive sailing simulation of the Ten-oared Whaleboat, a traditional vessel widely used in sea scout training across Cyprus and the Mediterranean region. The simulation is developed in Unity 6.3 and is grounded in physically-informed models of hydrodynamic and aerodynamic forces, with the goal of providing an accessible and educationally meaningful environment for novice sailors to develop foundational sailing skills.

The hydrodynamic model computes buoyancy, pressure drag, friction drag, and lateral resistance on a per-triangle basis from a dynamically generated underwater mesh, allowing the vessel's response to waves and motion to emerge from the geometry of the hull. The aerodynamic model derives lift and drag forces from camber-dependent polar curves interpolated in real time from the deformation of a cloth-simulated sail, and incorporates an induced drag term based on the finite-span characteristics of the sail geometry. The user controls the vessel through boom rotation and rudder deflection, with a real-time debug overlay providing immediate feedback on key simulation state variables.

The simulation is evaluated through a series of controlled experiments in which boom angle is varied across different points of sail and the resulting vessel speed is assessed against expected behaviour derived from sailing theory and practical knowledge of the Ten-oared Whaleboat. Two sailing performance tests were conducted: a downwind running test and a beam reach test. Both confirmed that the simulation produces physically plausible and consistent responses to sail trim inputs, with optimal boom angles correctly identified in the regions predicted by sailing theory. A task-based expert evaluation with five experienced Ten-oared Whaleboat sailors produced broadly positive results, with all six Likert-scale evaluation items rated above the neutral midpoint. The simulation's usefulness for sailing training and education received a unanimous rating of 7 from all participants. Qualitative findings identified the core sailing dynamics as credible, the controls as intuitive, and the UI feedback as valuable, while highlighting the vessel's unrealistic behaviour when in irons and the need for a visual wind direction indicator as the primary areas for improvement.

The work identifies a number of directions for future development, including a full VR-based human-computer interaction layer, an educator dashboard for scenario-based training, and the extension of the physical models to include Froude residuary resistance, added mass, hull aerodynamic drag, and a complete three-sail configuration.

Keywords: sailing simulation, real-time physics, sail aerodynamics, Unity, Ten-oared Whaleboat.

TABLE OF CONTENTS

ABSTRACT	iv
TABLE OF CONTENTS	v
1 Introduction	1
1.1 Motivation / Overview	1
1.2 Purpose of the Work	2
1.3 Methodology and Contributions	2
1.4 Outline / Document Structure	3
2 Background and Related Work	4
2.1 Fundamentals of Sailing	4
2.1.1 Basic Principles of Sailing	4
2.1.2 Forces Acting on a Sailing Vessel	4
2.1.3 Points of Sail	5
2.1.4 Sail Control and Trimming	6
2.1.5 Ten-oared Whaleboat Characteristics	6
2.1.6 Vessel Terminology	8
2.2 Hydrodynamics of Small Vessels	9
2.2.1 Buoyancy and Floating Stability	9
2.2.2 Hydrodynamic Resistance	11
2.2.3 Wave-Body Interaction	12
2.3 Aerodynamics of Sails	13
2.3.1 Wind and Relative Motion	13
2.3.2 Lift and Drag Forces on Sails	14
2.3.3 Resultant Force and Vessel Motion	16
2.3.4 Sail Trim and Angle of Attack	17
2.3.5 Rudder Hydrodynamics	17
2.3.6 Interaction with Hydrodynamic Forces	18
2.4 Computer-Based Simulation and Game Engines	18

2.4.1	Simulation-Based Training Environments	18
2.4.2	Real-Time Interactive Simulation	19
2.4.3	Game Engines as Simulation Platforms	19
2.4.4	Unity Engine	20
2.5	Related Work	21
2.6	Summary	22
3	Overview	23
3.1	System Overview	23
3.2	Modeling	24
3.2.1	Boat Hull	24
3.2.2	Mast, Boom and Sail	24
3.2.3	Ocean Environment	25
3.3	Simulation	25
3.3.1	Forces Acting on the Hull	25
3.3.2	Forces Acting on the Sail	25
3.3.3	User Controls	26
3.4	Human-Computer Interaction	26
4	Technical Section	27
4.1	Modeling	27
4.1.1	Boat Hull	27
4.1.2	Mast, Boom, Sail	28
4.1.3	Ocean Environment	29
4.2	Scene Managers	31
4.2.1	Wind Manager	31
4.2.2	Water Controller	32
4.2.3	Sail Wind Receiver	32
4.3	Hull Physics	33
4.3.1	Underwater Mesh Generation	33
4.3.2	Buoyancy and Damping	35
4.3.3	Pressure Drag	37
4.3.4	Friction Drag	38
4.3.5	Lateral Resistance	38
4.3.6	Rudder Force Model	40
4.4	Sail Aerodynamics	41
4.4.1	Apparent Wind	41
4.4.2	Angle of Attack	42

4.4.3	Camber Sampling	43
4.4.4	Polar Curves and Coefficient Blending	45
4.4.5	Induced Drag	46
4.4.6	Force Application and Center of Effort	47
4.5	User Controls	47
4.5.1	Boom Control	48
4.5.2	Sheet Controller	49
4.5.3	Rudder Control	50
4.5.4	Wind Manager	51
4.6	User Interface	51
4.6.1	Debug HUD	51
4.6.2	Compass Indicator	51
4.6.3	Vessel Trail	52
4.6.4	Camera System	53
5	Testing and Evaluation	54
5.1	Overview	54
5.2	Technical Testing	54
5.2.1	Downwind Sailing Test	54
5.2.2	Beam Reach Sailing Test	58
5.3	User Study	60
5.3.1	Study Design	60
5.3.2	Participants	61
5.3.3	Procedure	61
5.3.4	Data Collection	63
5.3.5	Analysis	63
5.3.6	Results	64
5.4	Summary	67
6	Conclusion and Future Work	69
6.1	Summary of Work	69
6.2	Evaluation Against Objectives	69
6.3	Limitations	70
6.4	Future Work	72
	BIBLIOGRAPHY	74

1 Introduction

1.1 Motivation / Overview

Over the past fifteen years, active participation in sea scouting has fostered a strong connection with the marine environment, particularly through traditional rowing and, subsequently, sailing with small wooden vessels.

As progression within the scouting framework led to the pursuit of formal qualifications required to assume the role of a captain of our boats, the significant responsibility associated with this position became evident. This responsibility is especially critical when leading a crew that includes younger and less experienced members. Safe and effective boat handling requires not only theoretical knowledge but also practical experience and confidence, both of which are typically developed through repeated exposure to real sailing conditions.

However, acquiring such experience presents several challenges. Sailing activities are inherently dependent on multiple external factors, including the availability of a full crew, suitable weather conditions, and the time-intensive preparation required to launch and operate a boat. These constraints limit the frequency and flexibility of training opportunities, particularly for individuals seeking to develop their skills independently.

In recent years, advancements in computer graphics, real-time physics simulation, and interactive development platforms have enabled the creation of highly realistic virtual environments. Modern game engines provide the capability to simulate complex physical systems with increasing levels of accuracy, making them suitable for training-oriented applications.

Motivated by these challenges and opportunities, this thesis explores the development of a sailing simulation tailored specifically to the characteristics of a traditional Ten-oared Whaleboat. The aim is to provide a realistic and accessible training environment that allows users to practice essential sailing skills, gain experience, and build confidence without the constraints imposed by real-world conditions.

1.2 Purpose of the Work

The primary purpose of this thesis is to design and develop a realistic sailing simulation that can serve as a training tool for individuals involved in small-boat sailing, particularly within the context of sea scouting.

The project focuses on accurately modelling the behavior of a traditional Ten-oared Whaleboat within a virtual environment. Emphasis is placed on replicating key physical interactions, including wind forces, sail dynamics, and boat response, in order to achieve a high level of realism.

By leveraging the capabilities of modern game development platforms, specifically the Unity engine, the simulation seeks to bridge the gap between theoretical instruction and practical experience. It aims to provide users with the ability to practice sailing independently, without the need for a full crew or specific environmental conditions.

Furthermore, the simulation is designed to reflect the operational characteristics and constraints of real-world sea scout vessels, ensuring relevance and applicability. Through this approach, the project aspires to support more effective training, enhance user confidence, and contribute to safer boat handling practices.

1.3 Methodology and Contributions

This work follows a physically-based simulation approach for modelling the behavior of a sailing vessel within a virtual environment. The system is developed using the Unity game engine, combining its built-in physics capabilities with custom modelling techniques to achieve realistic results.

The simulation incorporates key physical interactions, including the effect of water on the hull, the influence of wind on the sail, and the resulting motion of the boat. A modular design is adopted, allowing different components of the system, such as boat dynamics, sail behavior, and user interaction, to be developed and integrated effectively.

User control is an essential aspect of the system, enabling real-time manipulation of sail configuration through interactive inputs. This allows the simulation to reflect fundamental sailing principles and provides an engaging and practical training experience.

The main contributions of this work include the development of a sailing simulation tailored to a traditional Ten-oared Whaleboat, the integration of physically-based models for boat and sail behavior, and the design of an interactive environment aimed at supporting training and skill development. The system also establishes a foundation for further enhancements, such as more advanced environmental modelling and user interface integration.

1.4 Outline / Document Structure

The remainder of this thesis is organised as follows.

Chapter 2 presents the theoretical and technical background underpinning the simulation. It covers the fundamental principles of sailing, the hydrodynamic behaviour of small vessels, the aerodynamics of sails, and the role of game engines as real-time simulation platforms. Related work in sailing simulation is also reviewed.

Chapter 3 provides a high-level overview of the simulation system. It describes the three main components of the system modelling, simulation, and user interaction and illustrates the data flow and relationships between them, giving the reader a conceptual map before the technical detail is presented.

Chapter 4 presents the full technical implementation of the simulation. It covers the generation of the underwater mesh, the hydrodynamic force models acting on the hull, the aerodynamic force model acting on the sail, the user control systems, and the human-computer interaction layer implemented in this work.

Chapter 5 presents the testing and evaluation of the simulation. Two controlled sailing performance tests are conducted, examining the relationship between boom angle and vessel speed at a downwind running condition and at a beam reach, with results compared against the predictions of sailing theory. A task-based expert evaluation with semi-structured interviews is also conducted, in which experienced Ten-oared Whaleboat sailors assess the physical plausibility, control intuitiveness, and feedback quality of the simulation.

Chapter 6 concludes the thesis by evaluating the simulation against the objectives set out in this chapter, discussing the limitations of the current implementation, and identifying directions for future work, including the full HCI layer, additional resistance components, and multi-sail configurations.

2 Background and Related Work

2.1 Fundamentals of Sailing

2.1.1 Basic Principles of Sailing

Sailing is the process of converting wind energy into controlled motion of a vessel across the water. Unlike simple drifting, a sailing boat is capable of moving in various directions relative to the wind through the interaction between aerodynamic and hydrodynamic forces. By appropriately adjusting the orientation of the sail and the heading of the vessel, it is possible to generate forward motion even when travelling partially against the wind.

The motion of a sailing vessel is governed by the relationship between wind direction, sail configuration, and hull orientation. Effective sailing requires continuous adjustment of these parameters in order to maintain stability, efficiency, and control.

2.1.2 Forces Acting on a Sailing Vessel

The movement of a sailing vessel is the result of multiple forces acting simultaneously on different parts of the system. These forces can be broadly categorized into aerodynamic and hydrodynamic forces.

Aerodynamic forces act on the sail due to the interaction with the wind. These include lift, which is generated perpendicular to the wind direction, and drag, which acts in the direction of the wind flow. The combination of these forces produces a resultant force that propels the vessel forward [1, pp. 64-75] [2, pp. 185-241].

Hydrodynamic forces act on the hull as it interacts with the water. The most significant of these is buoyancy, which acts upward and allows the vessel to float, as described by Archimedes' principle [1, pp. 57-63]. In addition, water resistance, commonly referred to as drag, opposes the motion of the boat and depends on its velocity and shape. These forces contribute to both the stability and resistance of the vessel.

The overall motion of the boat is determined by the balance between aerodynamic propulsion and hydrodynamic resistance, as well as the distribution of forces acting on the vessel. A more detailed analysis of these forces is presented in the following sections.

2.1.3 Points of Sail

The term “points of sail” refers to the different directions in which a sailing vessel can travel relative to the wind direction. These positions determine the configuration of the sails, the efficiency of propulsion, and the overall behavior of the vessel.

Sailing directions are defined by the angle between the vessel’s heading and the true wind direction as shown at figure 2.1. When a boat attempts to sail directly into the wind, it enters a region known as the “no-go zone,” where effective propulsion cannot be generated due to the inability of the sail to produce sufficient aerodynamic force. Consequently, progress in this direction is achieved through a maneuver known as tacking, in which the vessel follows a zigzag path while alternating its heading relative to the wind.

At small angles to the wind, the vessel is said to be sailing close-hauled. In this configuration, the sails are trimmed tightly, and propulsion is primarily generated through lift forces. As the angle between the boat and the wind increases, the vessel transitions to a beam reach, where the wind approaches from the side. This point of sail is typically associated with high efficiency and speed, as both lift and drag contribute effectively to forward motion.

Further turning away from the wind results in a broad reach, where the wind approaches from behind at an angle. In this case, the sails are eased to capture a larger portion of the wind, and propulsion increasingly relies on drag forces. When sailing directly downwind, the vessel operates in a running configuration, where the sails are fully extended to maximize exposure to the wind [3,4] [1, pp. 53-55] [2, pp. 34-35].

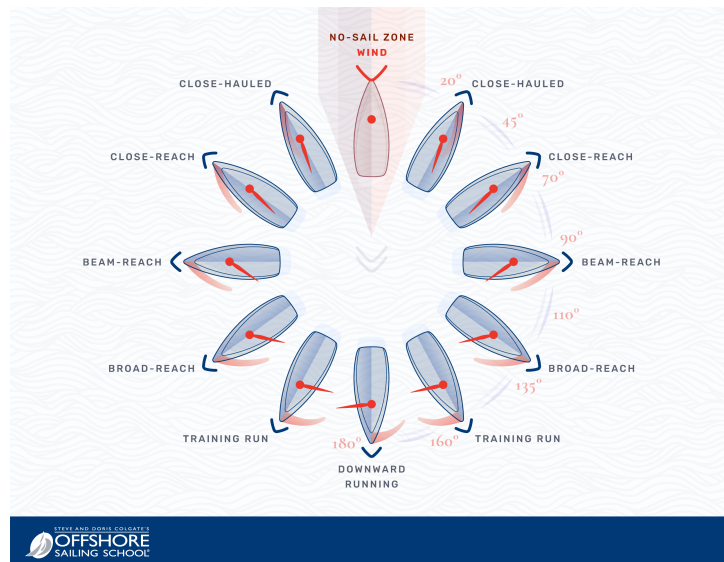


Figure 2.1: Points of sail relative to wind direction. (source: Offshore Sailing [3])

2.1.4 Sail Control and Trimming

Sail control, commonly referred to as trimming, is the process of adjusting the sail's orientation relative to the wind in order to optimize performance. This is typically achieved by controlling the boom and the tension of the sail.

Proper trimming ensures that the sail maintains an efficient angle with respect to the incoming wind, maximizing the generated aerodynamic forces. A tightly trimmed sail is used when sailing closer to the wind, while a looser configuration is more effective when sailing downwind.

Incorrect sail trimming can lead to a significant reduction in efficiency, loss of speed, and decreased control of the vessel. Therefore, continuous adjustment of the sail is essential for effective sailing.

The principles described in this section form the foundation for the modelling approach adopted in this thesis, particularly in the representation of sail forces and user-controlled sail manipulation within the simulation environment.

2.1.5 Ten-oared Whaleboat Characteristics

The Ten-oared Whaleboat is a traditional vessel historically associated with whaling ships, from which it derives its name. Its design is characterized by a long and narrow hull, optimized for both rowing and sailing operations. Due to its shape and relatively lightweight construction, the vessel exhibits a balance between maneuverability and stability, making it suitable for training and group-based maritime activities.

The boat is typically constructed either from wood or from Glass Reinforced Plastic (GRP). Wooden variants possess inherent buoyancy, which contributes to their ability to remain afloat even when flooded, provided that excessive additional weight is not present. In contrast, GRP versions are equipped with watertight compartments that ensure buoyancy in the event of flooding or capsizing.

In terms of dimensions, the Ten-oared Whaleboat has an overall length of approximately 8.20 meters, a maximum beam of 1.85 meters, and a loaded draft of approximately 0.85 meters. The total weight of the fully rigged vessel is around 900 kilograms, and it is designed to accommodate a crew of up to twelve individuals [1, p.20].

The crew structure of the vessel is well-defined and consists of ten rowers, a captain, a boatswain responsible for the correct handling of the rigging and associated equipment, and a lookout who continuously monitors the surrounding maritime environment. Steering is typically performed by a helmsman, a role that is often assumed by the captain. This distribution of responsibilities is essential for the safe and efficient operation of the vessel, particularly under sailing conditions

[1, pp. 47-50].

The vessel supports dual modes of propulsion: traditional rowing using oars, and sailing through the use of a three-sail rigging system. The sail configuration includes a jib, a mainsail, and a mizzen sail, with the latter two typically of the lateen type. This arrangement allows for flexible operation under varying environmental conditions [1, p.20].

From a sailing perspective, the characteristics of the Ten-oared Whaleboat introduce specific operational considerations. The relatively large crew size necessitates coordinated control, while the sail configuration requires careful trimming and adjustment to achieve efficient motion. Additionally, the hull geometry and weight distribution influence the vessel's response to wind and water forces.

In the simulation developed in this work, a simplified sail configuration is adopted. While the real Ten-oared Whaleboat utilizes a three-sail rigging system, the simulation models a single sail in order to reduce system complexity and enhance the clarity of the underlying sailing principles. This simplification allows for more focused analysis of sail behavior, user interaction, and force generation, without the additional complexity introduced by multiple interacting sails. A more detailed discussion of this modeling choice is provided in the methodology chapter. [1, p. 20] Similarly, the crew weight and its distribution across the vessel are not modeled in the simulation, the vessel mass is treated as constant and uniformly represented by a single Rigidbody with a fixed center of mass, as described in Section 4.3.



(a) Ten-oared Whaleboat with sails deployed.



(b) Ten-oared Whaleboat configured for rowing.

Figure 2.2: Configurations of a Ten-oared Whaleboat used in sea scouting activities.

2.1.6 Vessel Terminology

The following terms are used throughout this thesis to refer to specific parts of the vessel and its rigging. Figure 2.3 provides a visual reference for the key components described below.

Hull — the main body of the vessel, which sits in the water and provides buoyancy and structural support for all other components.

Bow — the front end of the vessel.

Stern — the rear end of the vessel.

Keel — the longitudinal structural element running along the centreline of the hull bottom, which provides lateral resistance and directional stability.

Rudder — a movable control surface mounted at the stern, below the waterline, used to steer the vessel by deflecting the water flow.

Mast — the vertical pole stepped into the hull that supports the sail and boom.

Boom — the diagonal spar that connects the sail to the mast and controls the orientation of the sail relative to the wind.

Sail — the fabric surface that captures wind energy and converts it into aerodynamic forces that drive the vessel.

Clew — the lower free corner of the sail, to which the sheet is attached. Adjusting the sheet tension at the clew controls the camber and trim of the sail.

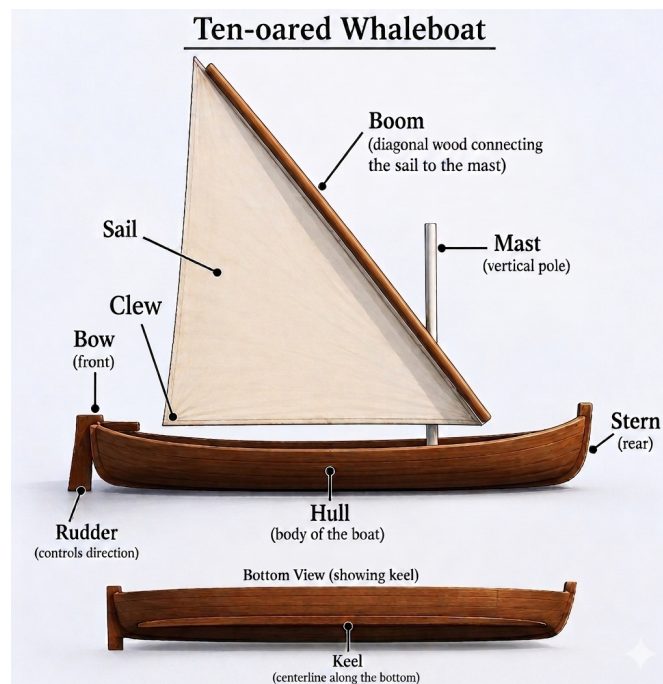


Figure 2.3: Key components of the Ten-oared Whaleboat referred to throughout this thesis.

2.2 Hydrodynamics of Small Vessels

Hydrodynamics plays a fundamental role in determining the motion and behavior of small vessels, as it describes the interaction between the hull and the surrounding water. The response of a vessel to external forces such as wind and waves is largely governed by hydrodynamic effects, which influence both its stability and resistance to motion.

2.2.1 Buoyancy and Floating Stability

2.2.1.1 Definition and Principles

Buoyancy is the fundamental force that enables a vessel to float and is governed by Archimedes' Principle. This principle states that any object, wholly or partially immersed in a fluid, experiences an upward force equal to the weight of the fluid it displaces. In the context of marine systems, this force determines the vessel's draft and trim, directly influencing its stability and pressure distribution along the hull [5, 6].

For small vessels, buoyancy plays a critical role in maintaining equilibrium under varying loading and environmental conditions. Variations in immersion, caused by waves or shifts in mass distribution, result in corresponding changes in the buoyant force, which in turn affect the vessel's motion and stability.

2.2.1.2 Mathematical Representation

The buoyant force F_b can be expressed as:

$$F_b = \rho g V$$

where ρ is the density of the fluid, g is the acceleration due to gravity, and V is the volume of displaced fluid. This formulation provides a global approximation of buoyancy and is commonly used in theoretical analyses of floating bodies [5].

2.2.1.3 Application in Simulation

In modern real-time simulations, particularly those involving dynamic water surfaces, buoyancy is often computed at a more localized level in order to capture the interaction between the vessel and waves. Instead of treating the hull as a single volume, the submerged surface is discretized into smaller elements, such as triangular mesh faces [7].

In this approach, the hydrostatic force acting on each submerged element can be approximated as:

$$F_{\text{hydrostatic}} = \rho A C g (\mathbf{n} \cdot \mathbf{y}) \mathbf{y}$$

where A represents the area of the submerged surface element, C is the vertical distance from the water surface to the element, $\mathbf{n}$ is the surface normal vector, and $\mathbf{y}$ is the upward unit vector [7].

This formulation accounts for both the depth of submersion and the orientation of the surface, allowing for a more accurate representation of buoyancy forces across complex hull geometries. As a result, the center of buoyancy can be dynamically updated as the vessel interacts with waves, leading to more realistic motion, including heave, pitch, and roll [6, 7].

Such discretized approaches are widely used in real-time applications and game-based simulations, where a balance between computational efficiency and physical realism is required [8]. By applying buoyant forces at multiple points on the hull, the simulation captures the distributed nature of hydrostatic pressure, improving the overall fidelity of vessel behavior under varying environmental conditions.

2.2.1.4 Center of Buoyancy and Stability

In addition to the magnitude of the buoyant force, its point of application plays a crucial role in determining the stability of a floating vessel. The buoyant force acts through the center of buoyancy (CB), which is defined as the centroid of the displaced fluid volume. The weight of the vessel, on the other hand, acts through its center of gravity (CG).

For a vessel in equilibrium, the buoyant force and the gravitational force are equal in magnitude and aligned along the same vertical line. When this condition is satisfied, the vessel remains in a stable floating state [5, 9].

However, when the vessel is disturbed for example by waves or uneven loading the relative positions of the center of buoyancy and the center of gravity may change. As the hull tilts, the submerged volume is redistributed, causing the center of buoyancy to shift. This shift generates a restoring moment that tends to return the vessel to its original upright position.

The relationship between these two points is fundamental to understanding the stability of floating bodies and plays an important role in the realistic simulation of vessel motion under dynamic conditions.

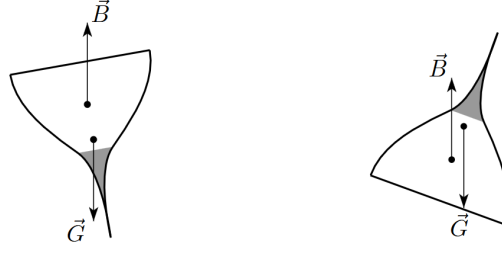


Figure 2.4: Illustration of the stability of a keel boat: a right-side-up configuration is stable because the center of gravity $\vec{G}$ is below the center of buoyancy $\vec{B}$, while the upside-down configuration is unstable [5].

2.2.2 Hydrodynamic Resistance

Hydrodynamic resistance refers to the forces that oppose the motion of a vessel as it travels through water. These forces arise due to the interaction between the hull and the surrounding fluid and play a critical role in determining the vessel's performance and efficiency [10, 11].

The total resistance of a vessel is commonly decomposed into several components, each arising from a different physical mechanism. In this work, three primary components are considered: pressure drag, viscous friction, and lateral resistance.

Pressure drag results from the asymmetric distribution of hydrodynamic pressure around the moving hull. As the vessel advances, water is displaced at the bow and reattaches toward the stern, creating a net pressure difference that opposes forward motion. For moderate to high velocities, this force is commonly approximated using a quadratic relationship [10, 12]:

$$F_D = \frac{1}{2} \rho C_D A v^2$$

where ρ is the fluid density, C_D is the drag coefficient, A is a reference area, and v is the velocity relative to the water. In this work, pressure drag is computed per submerged triangle of the hull mesh, by projecting the relative velocity onto each face normal and accumulating the resulting forces across the entire underwater surface.

Viscous friction arises from the interaction between the hull surface and the boundary layer that forms as water flows along it. The standard method for estimating this component in ship resistance calculations is the ITTC-57 correlation line, which expresses the frictional resistance coefficient C_F as a function of the Reynolds number Re [10]:

$$C_F = \frac{0.075}{(\log_{10} Re - 2)^2}$$

where $Re = vL/\nu$, with v the vessel speed, L the waterline length, and ν the kinematic viscosity of water. This formulation provides a well-established baseline for frictional resistance and is widely used in naval architecture. In this work, a simplified constant friction coefficient is applied per triangle instead of the full ITTC-57 line the implementation of the complete formulation is identified as future work [13].

Beyond frictional and pressure drag, a complete resistance model for a sailing vessel should account for residuary resistance, which includes wave-making resistance and form effects not captured by the friction line. The Froude number, defined as $Fr = v/\sqrt{gL}$, is the governing parameter for wave-making resistance, as it characterizes the ratio of inertial to gravitational forces acting on the hull [10]. At higher Froude numbers, the energy radiated into the wave system generated by the vessel becomes a dominant component of the total resistance. This component was studied but not implemented in the current simulation and is identified as future work.

A further effect relevant to the dynamics of a vessel in unsteady conditions is added mass, which represents the inertia of the surrounding fluid that is accelerated along with the hull. This added fluid inertia effectively increases the apparent mass of the vessel when it undergoes acceleration, and must be accounted for in a complete dynamic model [9, 10]. Added mass was not implemented in this work and is similarly deferred to future development.

For sailing vessels, lateral resistance is also of particular importance. Wind forces acting on the sails generate a sideways component of motion, known as leeway. The interaction between the hull and the water produces a resisting force that limits this drift, often enhanced by the presence of a keel or deep hull sections that generate lift-like forces and improve directional stability. In this work, lateral resistance is computed per triangle by isolating the component of relative velocity perpendicular to the vessel's forward axis and applying a resistance force opposing that motion.

2.2.3 Wave-Body Interaction

Ocean waves influence the motion of a vessel mainly by changing the surface of the water and the flow conditions surrounding it. As waves propagate, the elevation of the water surface varies over time, causing continued changes in the submerged volume of the hull. These variations directly affect the buoyant force acting on the vessel, leading to vertical motion (heave) and, rotational motions such as pitch and roll.

In addition to modifying buoyancy, waves can also generate hydrodynamic forces due to the relative motion between the water and the vessel. These wave-induced forces arise from local fluid velocities and pressure variations and may contribute to both vertical and horizontal motion components [9].

2.3 Aerodynamics of Sails

2.3.1 Wind and Relative Motion

In sailing, the forces acting on a vessel's sails depend not only on the ambient wind conditions but also on the motion of the vessel itself. For this reason, it is necessary to distinguish between true wind and apparent wind, as well as to understand their relationship through relative motion.

The true wind refers to the wind velocity measured with respect to a stationary observer on the Earth. In contrast, when a vessel is in motion, it experiences a different wind known as the apparent wind. This apparent wind results from the combination of the true wind and the velocity of the vessel [1, 9].

Mathematically, this relationship can be expressed as:

$$\mathbf{V}_a = \mathbf{V}_t - \mathbf{V}_b$$

where $\mathbf{V}_a$ is the apparent wind velocity, $\mathbf{V}_t$ is the true wind velocity, and $\mathbf{V}_b$ is the velocity of the vessel.

This relationship is often represented graphically using the wind triangle, where the three velocity vectors form a closed triangle. The apparent wind determines both the magnitude and direction of the airflow acting on the sails and is therefore the primary quantity used in aerodynamic force calculations.

Understanding the concept of apparent wind is essential for sailing, as it directly influences sail performance and trimming. Even in constant true wind conditions, changes in vessel speed or heading will alter the apparent wind, thereby affecting the forces generated by the sails and the resulting motion of the vessel.

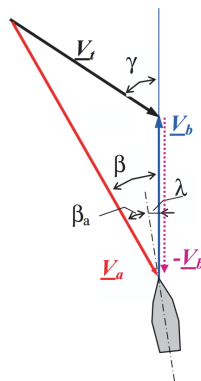


Figure 2.5: Wind Triangle [9]

2.3.2 Lift and Drag Forces on Sails

The interaction between the apparent wind and the sails generates a pressure distribution across the sailcloth. Following the principles of Bernoulli and Venturi, this pressure differential results in a total aerodynamic force $\mathbf{F}_{\text{total}}$ that can be decomposed into two primary components, lift and drag.

Lift (**L**) is defined as the force component that acts perpendicular to the apparent direction of the wind and is the main contributor to propulsion as it can be oriented forward through proper trimming of the sail. Drag (**D**) is the component acting parallel to the direction of the apparent wind, representing the resistance encountered by the sail from its movement through the air [1, 2].

The magnitude of these forces is typically expressed using non-dimensional coefficients:

$$F = \frac{1}{2} \rho A C V^2$$

where ρ is the air density, A is the effective sail area, V is the apparent wind speed, and C is a coefficient that depends on the sail shape and angle of attack [2, 9].

2.3.2.1 Aerodynamic Coefficient Polars and Sail Camber

The values of C_L and C_D are not constant but vary with the angle of attack and the shape of the sail. This dependency is typically represented through polar curves, which plot the lift and drag coefficients as continuous functions of the angle of attack for a given sail configuration [14]. Polar curves are fundamental tools in sail aerodynamics, as they characterize the full operating range of a sail from attached flow at low angles of attack through to stall and post-stall behavior.

An important geometric parameter that determines the shape of these polars is the camber of the sail, defined as the depth of the sail's curvature relative to its chord length. A cambered sail generates lift even at zero angle of attack and generally achieves higher peak C_L values than a flat sail, at the cost of increased base drag [14, 15]. In practice, the degree of camber is controlled by the tension applied to the sheet, and therefore changes continuously as the sailor adjusts the sail trim.

In this work, three discrete camber states are defined — flat, mid, and full — each associated with its own C_L and C_D polar curves derived from published aerodynamic data [14–16]. The instantaneous camber of the sail is measured in real time from the deformation of the cloth mesh, and the aerodynamic coefficients are obtained by continuously interpolating between the three polar curves based on this measured value.

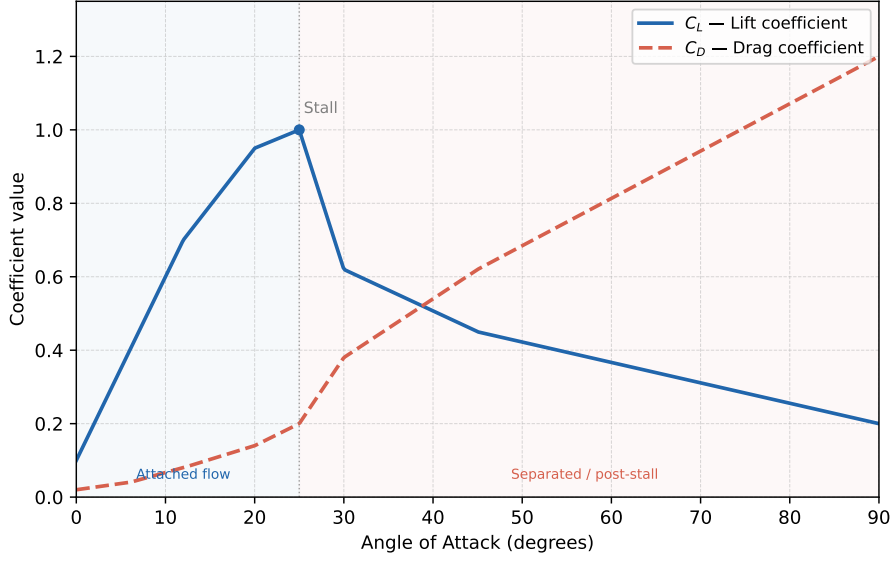


Figure 2.6: Representative C_L and C_D polar curves for a cambered sail (mid camber state), showing the linear attached-flow region, peak lift at the stall angle, and post-stall behavior. Based on aerodynamic data from [14, 15].

2.3.2.2 Induced Drag

In addition to the profile drag captured by the polar curves, a sail of finite span generates an additional drag component known as induced drag. This arises because lift production requires a pressure difference between the two sides of the sail, which causes air to flow around the tip from the high-pressure to the low-pressure side, forming trailing vortices. These vortices tilt the local lift vector rearward, effectively converting a portion of the lift into drag [13, 14].

For a finite wing or sail, induced drag is commonly expressed as:

$$C_{D_i} = \frac{C_L^2}{\pi \cdot AR \cdot e}$$

where AR is the aspect ratio of the sail, defined as the ratio of the span squared to the sail area and e is the Oswald efficiency factor, which accounts for deviations from the ideal elliptical lift distribution [13, 14]. A higher aspect ratio reduces induced drag for a given lift coefficient, which is why tall, narrow sails are aerodynamically more efficient than short, wide ones. In this work, the aspect ratio is measured dynamically from the sail geometry, and an Oswald efficiency factor of $e = 0.85$ is adopted. The induced drag coefficient is computed at each timestep and added to the profile drag obtained from the polar curves to give the total drag acting on the sail.

2.3.3 Resultant Force and Vessel Motion

The aerodynamic forces generated on a sail can be decomposed into components that directly influence the motion of the vessel. The lift and drag forces are combined to form a resultant aerodynamic force, which can be further resolved into a forward thrust component and a lateral force component.

Mathematically, this relationship can be expressed as:

$$F_r = L \sin \beta - D \cos \beta$$

$$F_l = L \cos \beta + D \sin \beta$$

where L is the lift force, acting perpendicular to the apparent wind direction, D is the drag force, acting parallel to the apparent wind direction and β is the apparent wind angle, defined as the angle between the vessel's heading and the apparent wind [2, 9].

The forward component contributes to the propulsion of the vessel in the direction of travel, while the lateral component induces sideways motion, commonly referred to as leeway. In practice, this lateral motion is counteracted by the hydrodynamic resistance of the hull and keel, which provides directional stability in the water.

The efficiency of this force conversion strongly depends on the angle between the sail and the apparent wind, known as the angle of attack. By adjusting the sail trim, sailors aim to maximize the forward component of the aerodynamic force while minimizing unwanted drag and excessive lateral forces.

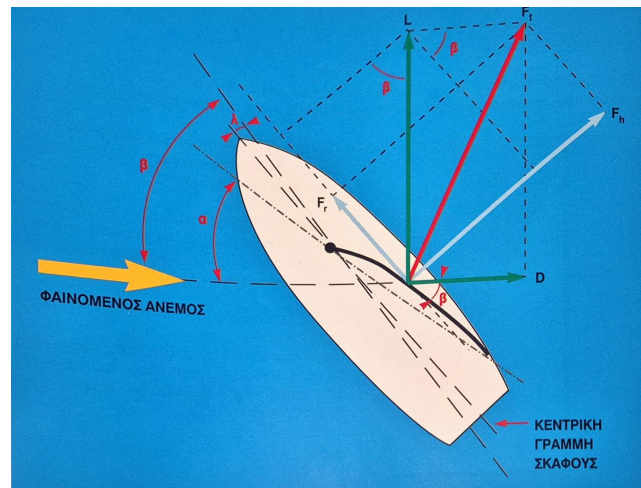


Figure 2.7: Aerodynamic forces on sail

2.3.4 Sail Trim and Angle of Attack

The aerodynamic efficiency of a sail is strongly influenced by its angle of attack (**aoa**), which is defined as the angle between the chord line of the sail and the direction of the apparent wind .

Sail trim refers to the adjustment of the sail's shape and orientation in order to control this angle. By tightening or loosening the sail, the curvature and orientation are modified, thereby affecting the lift-to-drag ratio for a given point. Proper sail trim allows the sail to operate efficiently by maintaining an optimal angle of attack, where lift is maximized and drag is minimized [2].

As the angle of attack increases, the lift coefficient (C_L) generally rises until it reaches a critical value known as the stall angle. Beyond this point, the airflow over the leeward side of the sail becomes separated, resulting in a significant reduction in lift and a corresponding increase in the drag coefficient (C_D).

Consequently, effective sail trimming is essential for maximizing propulsion efficiency and ensuring stable and controlled vessel motion under varying wind conditions.

2.3.5 Rudder Hydrodynamics

The rudder is a control surface mounted at the stern of the vessel, partially submerged below the waterline, that provides directional control by generating a hydrodynamic force when deflected. When the rudder is rotated about its vertical axis by an angle δ , it presents an angle of attack to the oncoming flow, generating a lift force perpendicular to the local flow direction [10]. This lift force acts at a distance from the vessel's center of mass, producing a yawing moment that rotates the vessel about its vertical axis and changes its heading.

The hydrodynamic forces acting on the rudder can be decomposed into lift and drag components, analogous to the aerodynamic forces acting on the sail. The lift force generated by the rudder is expressed as:

$$F_{L_r} = \frac{1}{2} \rho C_{L_r}(\delta) A_r V^2$$

where ρ is the water density, $C_{L_r}(\delta)$ is the rudder lift coefficient as a function of deflection angle, A_r is the rudder planform area, and V is the vessel speed through the water [10]. The rudder drag component similarly opposes the vessel's forward motion and increases with deflection angle. The effectiveness of the rudder therefore depends strongly on vessel speed, at low speeds the dynamic pressure $\frac{1}{2} \rho V^2$ is small, generating little force, while at higher speeds the rudder becomes increasingly responsive.

Beyond a critical deflection angle, the flow over the rudder separates and the lift coefficient drops sharply, a phenomenon analogous to stall in aerodynamics. Maintaining the deflection

angle within the attached flow regime is therefore important for effective directional control [10]. In the context of this simulation, the rudder is the primary means by which the user controls the vessel's heading. The implementation of a physically-based rudder force model is described in Section 4.5.3.

2.3.6 Interaction with Hydrodynamic Forces

The motion of a sailing vessel is governed by the interaction between aerodynamic forces generated by the sails and hydrodynamic forces acting on the hull. While the sails produce a resultant force that drives the vessel forward, a significant portion of this force acts laterally, tending to push the vessel sideways.

This lateral component is counteracted by hydrodynamic resistance, particularly through the submerged surfaces of the hull and keel. These elements generate opposing forces that limit sideways motion, allowing the vessel to convert aerodynamic force into forward propulsion rather than drifting excessively. As a result, efficient sailing depends on the balance between aerodynamic lift and hydrodynamic resistance.

The coupling between these two force systems is fundamental to sailing dynamics. Changes in sail trim or wind conditions alter the aerodynamic forces, which in turn affect the hydrodynamic response of the vessel. This continuous interaction determines the overall motion, stability, and maneuverability of the boat [9].

In addition to the aerodynamic forces generated by the sails, the portion of the hull above the waterline is also subject to aerodynamic drag from the wind. This above-waterline surface, which includes the topsides, rigging, mast, and crew presents a bluff body to the airflow, generating a drag force that opposes or deflects the vessel's motion depending on the wind direction [13]. This component is typically modelled analogously to pressure drag, using a drag coefficient and a reference frontal area exposed to the wind. The magnitude of this contribution relative to the total aerodynamic force depends on the ratio between the sail area and the above-waterline residual area [13]. Hull aerodynamic drag was not implemented in the current simulation and is identified as future work.

2.4 Computer-Based Simulation and Game Engines

2.4.1 Simulation-Based Training Environments

Computer-based simulation has become an essential tool for modeling complex physical systems in a controlled and safe environment. In the context of maritime applications, simulation enables

the study and practice of vessel behavior under varying environmental conditions without the risks and logistical constraints associated with real-world operation.

The system developed in this work is based on physically-informed models of hydrodynamic and aerodynamic forces, while also supporting real-time user interaction. Rather than focusing solely on high-fidelity numerical accuracy, the objective is to provide an interactive environment in which the user assumes the role of the vessel's captain and can explore the effects of control inputs and environmental conditions on the boat's motion.

Such simulation-based environments are particularly valuable for training and familiarization purposes, as they allow repeated experimentation and skill development in a flexible and accessible setting.

2.4.2 Real-Time Interactive Simulation

Real-time simulation refers to systems in which the computational model is updated continuously at a rate that allows immediate user interaction. Unlike offline simulations, which prioritize numerical accuracy and may require significant computational time, real-time systems must balance physical realism with computational efficiency.

In interactive applications, the user is actively involved in the simulation loop, influencing the system through control inputs that produce immediate and observable responses. This requirement imposes constraints on the level of detail and complexity of the physical models, often leading to the adoption of simplified or approximate formulations that are computationally efficient while still capturing the dominant physical behavior.

For sailing simulations, real-time performance is essential, as the user must be able to adjust sail trim, heading, and other control parameters while receiving continuous feedback from the system. This interactivity enhances the learning experience and allows the user to develop an intuitive understanding of vessel dynamics.

2.4.3 Game Engines as Simulation Platforms

Game engines have evolved into powerful platforms capable of supporting real-time interactive simulations beyond entertainment applications. Modern engines provide a comprehensive set of tools, including rendering systems, physics engines, input handling, and scripting environments, which significantly reduce development complexity.

These features make game engines particularly suitable for simulation-based applications, where both visual realism and physical behavior are required. The built-in physics systems allow for the simulation of rigid body dynamics, while extensibility through scripting enables the imple-

mentation of custom physical models, such as those required for hydrodynamic and aerodynamic force calculations.

Furthermore, game engines support real-time user interaction, allowing users to directly influence the simulation and observe immediate feedback. This characteristic is especially important in training-oriented applications, where experiential learning and system responsiveness play a critical role. The use of interactive simulation environments for training and analysis has been widely recognized as an effective approach in complex systems [17].

As a result, game engines offer an effective balance between development efficiency, performance, and flexibility, making them a practical choice for real-time simulation environments.

2.4.4 Unity Engine

Unity is a cross-platform real-time development engine that has established itself as a practical platform for interactive simulation applications. Its architecture integrates a physics engine, a scripting environment, and a rendering pipeline within a unified development framework, reducing the overhead associated with building real-time simulation systems from separate components. For the purposes of this work, Unity was selected as the development platform due to its accessible development workflow, its support for real-time user interaction, and the availability of a built-in water simulation system compatible with the hydrodynamic modeling requirements of the sailing vessel.

Unity's physics system is built on the PhysX library, which handles collision detection and the numerical integration of forces and motion. Physics updates run on a fixed timestep, independent of the rendering frame rate, which ensures that the physical state of the simulation advances at a consistent rate regardless of how demanding the scene is to render. This is important for physically-informed simulations, where temporal consistency in force integration directly affects the accuracy of the results. In this work, the built-in physics system is extended through custom force application, allowing hydrodynamic drag, wave resistance, and buoyancy to be calculated and applied to the vessel's rigid body at each physics step [18].

Unity's component-based architecture allows the simulation to be organized into separate, self-contained scripts, each responsible for a distinct aspect of the system. In this work, hydrodynamic force calculation, aerodynamic lift and drag, sail control, and user input are each implemented as independent components, reflecting the physical decomposition of the problem and making each subsystem easier to develop and adjust in isolation. Beyond this organizational benefit, Unity's C# scripting environment allows custom physical models to be implemented and applied directly within the simulation loop, without needing to rely on external numerical tools such as MATLAB or OpenFOAM.

In addition to rigid body dynamics, Unity includes a cloth simulation system for modeling deformable surfaces. Rather than treating an object as a single rigid mass, the cloth system represents it as a mesh of vertices subject to physical constraints, damping, and external forces. In this work, this system is used to model the sail as a deformable surface whose shape responds to aerodynamic loading and control inputs. Each vertex can be assigned individual constraint parameters that limit how far it can move from its reference position, effectively controlling the local stiffness of the surface at runtime. This mechanism is the basis of the sheet trimming model, where the constraint at the clew vertex is adjusted to simulate the easing and hardening of the skotta, and is described in detail in the implementation chapter [19].

2.5 Related Work

Sailing simulation has been approached from a variety of perspectives in both research and commercial contexts. High-fidelity numerical approaches, such as Velocity Prediction Programs (VPPs) and Computational Fluid Dynamics (CFD) solvers, have been widely used in naval architecture to predict vessel performance by balancing aerodynamic and hydrodynamic forces across a range of wind conditions. While these methods offer a high degree of physical accuracy, they are generally oriented towards offline analysis and design optimization rather than real-time interactive use [20].

On the commercial side, sailing simulators such as eSail, Sailaway, and NauticEd provide interactive environments aimed at recreational and educational users. These platforms prioritize accessibility and visual experience, but their underlying physical models are not openly documented and are not tailored to specific vessel types or training objectives.

In the research literature, prior work has addressed the challenge of combining physically-based models with real-time performance, though the balance between the two remains difficult to achieve. Sueda and Salafi [21] developed an interactive sailing simulation system targeting novice sailors, demonstrating that simplified physically-based models can be effective in an educational context without requiring high computational overhead. More recently, game engines have been adopted as simulation platforms in maritime contexts, as they allow custom physically-informed models to be implemented directly on top of a built-in physics infrastructure, without sacrificing the real-time performance required for interactive use. Unity in particular has emerged as a practical choice for this kind of work, given its accessible physics interface, component-based architecture, and established use in academic simulation research. The present work follows this direction, combining simplified physically-informed models of hydrodynamic and aerodynamic forces with a game engine framework to deliver a real-time interactive sailing simulation.

The present work differs from existing approaches in several respects. Rather than targeting general sailing or optimizing for competitive performance, the simulation is developed specifically for the Ten-oared Whaleboat. The physical models are tailored to the characteristics of this vessel, and the interaction design is oriented towards the needs of novice sailors developing foundational skills in a structured educational context. To the authors' knowledge, no existing simulator addresses this specific vessel type or the pedagogical requirements of sea scout training programs.

2.6 Summary

This chapter has presented the theoretical and technical background underpinning the simulation developed in this work. The fundamental principles of sailing were introduced, covering the aerodynamic and hydrodynamic forces acting on a vessel, the points of sail, and the role of sail trim in controlling vessel motion. The physical characteristics of the Ten-oared Whaleboat were described, providing the vessel-specific context for the simulation. The hydrodynamic behaviour of small vessels was examined, with particular attention to buoyancy, hydrostatic stability, and wave-body interaction. Hydrodynamic resistance was discussed in detail, covering pressure drag, viscous friction via the ITTC-57 formulation, and lateral resistance, alongside residuary resistance and added mass, which were studied but identified as future work. The aerodynamics of sails were discussed, including the distinction between true and apparent wind, the generation of lift and drag forces through polar curve representations across varying camber states, induced drag arising from finite span effects, and the aerodynamic drag acting on the above-waterline hull, the latter also deferred to future work. The combined effect of these forces on vessel motion was examined, together with their interaction with hydrodynamic resistance. The role of computer-based simulation and game engines as practical platforms for real-time interactive applications was then established, with Unity identified as the development environment for this work. Finally, related work in sailing simulation was reviewed, positioning the present contribution relative to both high-fidelity numerical approaches and existing interactive simulators. The following chapter presents an overview of the simulation system developed in this work.

3 Overview

3.1 System Overview

This chapter presents a high-level description of the simulation system developed in this work, providing the reader with a conceptual map of its components and their relationships before the technical implementation is described in detail in Chapter 4.

The system is a real-time interactive sailing simulation of a simplified Ten-oared Whaleboat, developed in Unity 6.3. Its primary goal is to provide a physically-informed environment in which a user can assume the role of captain and experience the fundamental dynamics of sailing a traditional vessel. The simulation is built around three interconnected pillars: the modeling of the physical assets that constitute the vessel and its environment, the simulation of the forces acting on those assets, and the user interaction layer through which the sailor controls the vessel and observes its response.

The physical models are grounded in the hydrodynamic and aerodynamic theory presented in Chapter 2, and are implemented as a set of independent but coupled components that operate together within Unity’s physics loop at each timestep. The interaction between these components, hull forces responding to water, sail forces responding to wind, and user inputs altering the configuration of the vessel, produces the emergent sailing behavior that the user experiences.

Figure 3.1 illustrates the overall architecture of the system and the relationships between its main components.

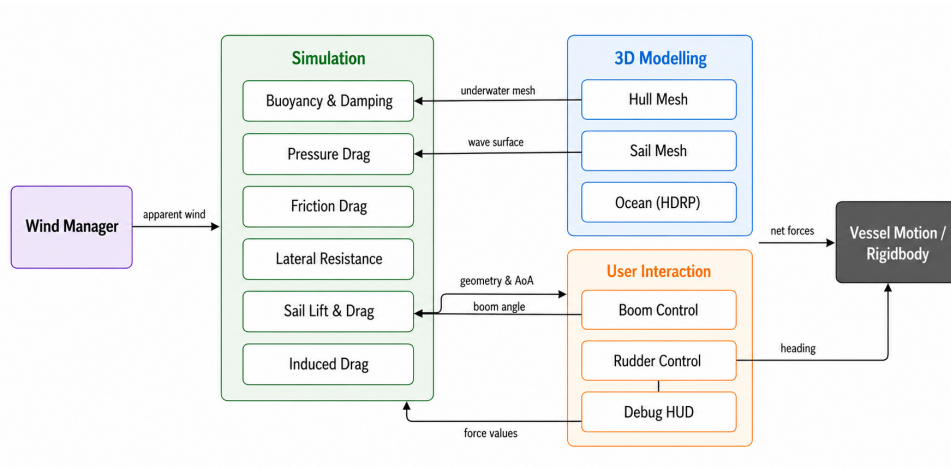


Figure 3.1: Overview of the simulation system architecture, showing the three main components and their relationships within the Unity physics loop.

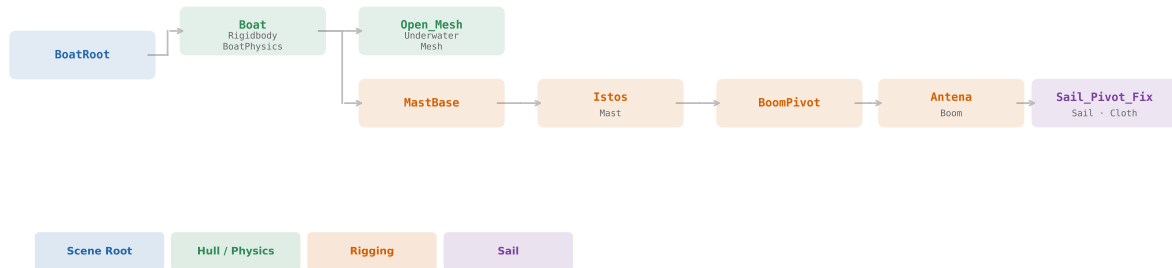


Figure 3.2: Simplified Unity scene hierarchy of the vessel GameObject structure.

3.2 Modeling

This section describes the three-dimensional assets that constitute the physical environment of the simulation. These assets form the foundation upon which the force calculations operate and define the geometry that the simulation interacts with at each timestep.

3.2.1 Boat Hull

The boat hull is modelled as a discrete three-dimensional mesh representing a simplified version of the Ten-oared Whaleboat. The hull GameObject is the root of the vessel’s parent-child hierarchy in Unity, and carries the Rigidbody component that accumulates all forces and moments acting on the vessel. Its mass and center of mass are configured to approximate the loaded weight and balance of the real vessel. At each physics step, the hull mesh is used to generate the underwater surface geometry that drives the hydrodynamic force calculations described in Section 3.3.1.

The rudder is modelled as a separate mesh integrated into the hull GameObject and serves as the steering surface of the vessel. It is controlled independently by the user through the rudder control system described in Section 4.3.6.

3.2.2 Mast, Boom and Sail

The mast, boom, and sail are modelled as separate GameObjects arranged in a parent-child hierarchy attached to the hull. The mast is fixed to the hull and defines the vertical span axis of the sail. The boom is a child of the mast and rotates around it in the horizontal plane, driven by the user’s control input. The sail is attached to the mast and boom and is modelled as a Unity cloth object, allowing its surface to deform in response to physical constraints. The geometry of the sail mesh including its span, chord, and resulting aspect ratio is sampled at runtime and used directly in the aerodynamic force calculations.

3.2.3 Ocean Environment

The ocean environment is implemented using Unity’s High Definition Render Pipeline (HDRP) water system. This system provides a dynamic water surface whose elevation varies over time according to a configurable wave model. The water surface height at any point on the hull is queried at each physics step, and the result is used to determine which triangles of the hull mesh are submerged and to what depth. This information feeds directly into the buoyancy and drag calculations. In addition to driving the force simulation, the HDRP water system provides a visually realistic ocean surface that contributes to the immersive quality of the environment.

3.3 Simulation

This section provides a high-level description of the force models that govern the physical behavior of the vessel. Each component operates independently but exchanges data with the others through the shared Rigidbody state and sampled geometry at each physics timestep.

3.3.1 Forces Acting on the Hull

The hydrodynamic forces acting on the hull are computed per triangle of the underwater mesh generated at each timestep. Three force components are calculated: buoyancy, which acts upward on each submerged face proportionally to its depth below the water surface, hydrodynamic drag, which is decomposed into a normal pressure component and a tangential friction component acting on each face relative to its motion through the water, and lateral resistance, which opposes sideways motion of the hull and provides the directional stability necessary for the vessel to convert sail forces into forward propulsion. A velocity-dependent damping term is also included in the buoyancy calculation to suppress unrealistic vertical oscillations. In addition, the rudder generates a hydrodynamic lift force when deflected, producing a yawing moment that controls the vessel’s heading. All forces are applied at the centroid of each triangle and accumulated on the Rigidbody at each fixed timestep.

3.3.2 Forces Acting on the Sail

The aerodynamic forces acting on the sail are computed at each physics step based on the apparent wind, the current angle of attack, and the instantaneous camber of the sail cloth. The apparent wind is obtained from the WindManager and combined with the vessel’s velocity to determine the effective airflow over the sail. The angle of attack is computed by projecting the apparent wind and the sail chord direction onto the plane perpendicular to the mast span axis. Lift and drag coefficients are then retrieved by interpolating between three camber-state polar

curves based on the measured sail deformation. An induced drag term is added to account for finite span effects. The resulting lift and drag forces are combined and applied at the sail's center of effort.

3.3.3 User Controls

The user interacts with the simulation through two primary control inputs. The boom can be rotated around the mast axis, which directly changes the sail's angle of attack relative to the apparent wind and therefore the aerodynamic forces generated. The rudder can be deflected to alter the vessel's heading. Together, these two inputs allow the user to explore the fundamental dynamics of sailing adjusting the sail to generate drive and using the rudder to maintain or change course. A debug overlay displays key simulation state variables in real time, including rudder and boom rotation, apparent wind speed and angle relative to the boat, and vessel heel angle, providing immediate feedback on the effect of control inputs.

3.4 Human-Computer Interaction

The full vision for the human-computer interaction layer of this simulation extends significantly beyond what has been implemented in the current work. The intended design is an immersive, VR-based environment in which the learner assumes the role of captain and is physically present on the vessel. Within this environment, the user would be able to look around and move across the boat, directly interacting with physical components such as the sheet attached to the clew, the boom control line, and the rudder. Navigational instruments modelled on the vessel, including a compass and a knot meter, would provide the user with real-time situational awareness. The acoustic environment would reinforce immersion through ocean ambience and the characteristic sound of sail luffing. A physical motion platform synchronized with the simulated vessel heel could further enhance the sense of presence, providing a kinesthetic feedback channel that mirrors the vessel's response to wind and sea conditions.

In addition to the on-board experience, the system design includes an educator dashboard accessible from a separate device, through which an instructor could modify environmental parameters such as wind speed, wind direction, and sea state in real time during a training session. This would allow structured scenario-based learning, where the educator progressively introduces more challenging conditions as the learner develops proficiency.

The current implementation focuses on the physical simulation layer and provides a basic desktop interface for development and validation purposes. The full HCI layer described above is identified as future work and is discussed further in Chapter 6.

4 Technical Section

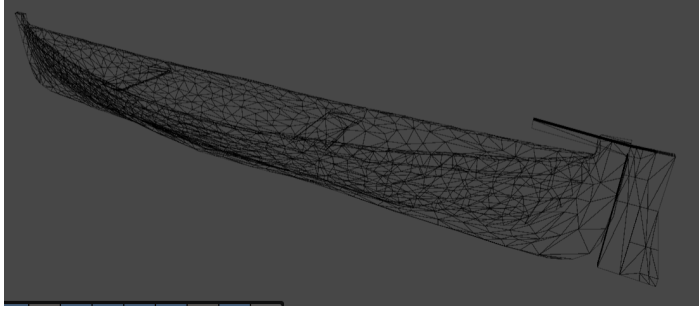
4.1 Modeling

4.1.1 Boat Hull

The boat hull is represented by a three-dimensional mesh originally sourced from Sketchfab [22], an online repository of three-dimensional models. The mesh was subsequently processed in Blender to reduce its polygon count, as the original geometry introduced performance issues incompatible with the real-time physics requirements of the simulation. The simplified mesh retains the essential hull form of the whaleboat while remaining computationally efficient. The rudder geometry was modeled separately in Blender and integrated into the hull mesh as a distinct sub-object. It serves as the steering surface of the vessel and is controlled independently by the user.

In the Unity scene, the hull `GameObject` serves as the root of the vessel hierarchy and carries the `Rigidbody` component through which all forces and moments are accumulated and integrated. The vessel mass is set to 1000 kg, approximating the loaded weight of the Ten-oared Whaleboat. The center of mass is offset by -1 m along the local vertical axis relative to the geometric center of the hull, lowering it toward the keel in order to improve the rotational stability of the vessel and reduce unrealistic roll responses. Linear and angular damping are set to zero, as all resistance to motion is handled explicitly by the hydrodynamic force models described in Section 4.3. Gravity is enabled, and collision detection operates in discrete mode, which is sufficient given the relatively low velocities involved in sailing dynamics.

The hull mesh serves a dual purpose in the simulation: it provides the visible geometry rendered to the user, and it acts as the input surface from which the underwater mesh is generated at each physics timestep, as described in Section 4.3.1.



(a) Wireframe view of the hull mesh in Unity, showing the simplified polygon structure and the rudder geometry at the stern.



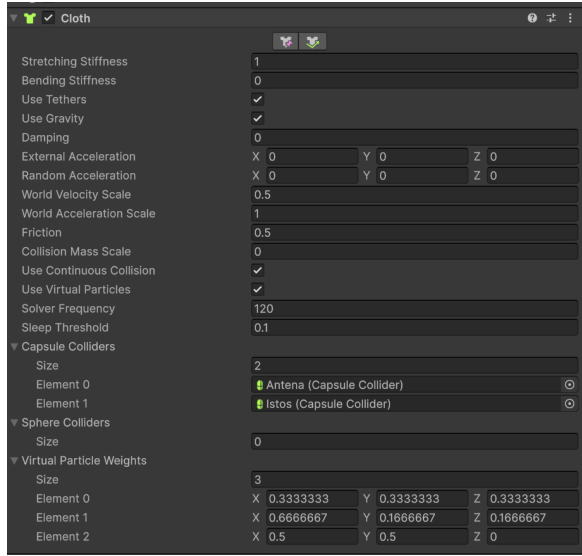
(b) Rigidbody component settings in the Unity Inspector.

Figure 4.1: Hull mesh geometry and Rigidbody configuration of the vessel.

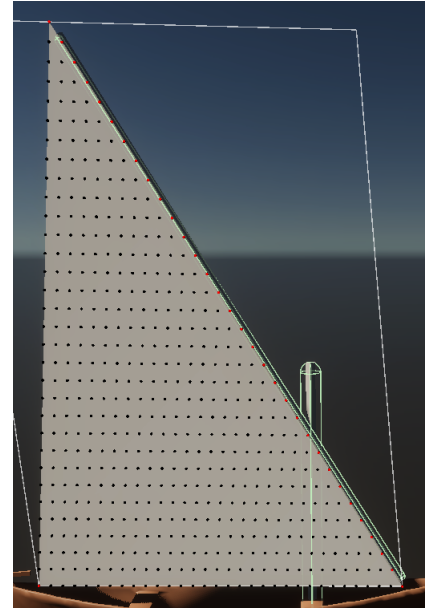
4.1.2 Mast, Boom, Sail

The mast, boom, and sail are modelled as separate GameObjects arranged in a parent-child hierarchy attached to the hull, as illustrated in Figure 3.2. All meshes in this assembly were created in Blender. The MastBase serves as the attachment point to the hull. The Istos (mast) is a child of MastBase and defines the vertical span axis used in the angle of attack calculation described in Section 4.4.2. The BoomPivot is a child of Istos and defines the rotation axis around which the boom (Antena) rotates in the horizontal plane, driven by the PodeonasController described in Section 4.5.1. A CenterOfEffort transform is defined within the sail hierarchy as the point at which the resultant aerodynamic force is applied to the Rigidbody.

The sail is modelled using Unity's Cloth component [19], which represents the sail surface as a mesh of vertices subject to physical constraints, damping, and external forces. This approach was chosen because it allows the sail to deform realistically in response to its boundary conditions, the leech vertices constrained to the boom (Antena) at a maximum distance of zero, while the remaining vertices are free to deform under the cloth simulation, producing a cambered shape. The Cloth component is configured with a stretching stiffness of 1 and bending stiffness of 0, allowing the sail to stretch minimally while remaining free to bend. The solver frequency is set to 120, and continuous collision detection is enabled to prevent the sail mesh from tunnelling through the rigging colliders at high velocities. Two capsule colliders, assigned to the Antena (boom) and Istos (mast), prevent the sail mesh from intersecting the rigging geometry. Gravity is enabled on the cloth, contributing to the natural drape of the sail under its own weight.



(a) Cloth component settings in the Unity Inspector.



(b) Cloth vertex constraints — red dots indicate vertices fixed at maximum distance zero along the leech.

Figure 4.2: Cloth component configuration and vertex constraints of the sail.

The Cloth component requires a Skinned Mesh Renderer [23] in order to operate, as Unity’s cloth simulation is built on top of the skinned mesh deformation pipeline. The sail mesh is a flat plane created in Blender, rigged with two bones (Podeonas and Skotta) with Podeonas serving as the root bone of the Skinned Mesh Renderer. The Skotta bone was used by the skotta controller to manipulate the clew vertex constraint dynamically, simulating sheet trimming. This controller was deactivated in the current implementation due to stability issues and is identified as future work.

4.1.3 Ocean Environment

The ocean environment is implemented using Unity’s High Definition Render Pipeline (HDRP) water system [24]. Several alternatives were considered before this choice was made. A full fluid dynamics simulation was ruled out due to the computational cost incompatible with real-time performance requirements. Gerstner wave-based implementation was explored, where the water surface is displaced using a sum of sinusoidal wave functions, but this approach required manual integration with the physics queries and offered limited visual fidelity. The Crest Ocean System [25], a widely used third-party ocean asset, was also considered but was not available in a free HDRP-compatible version at the time of development. The HDRP water system was ultimately selected as it provides both a physically-inspired dynamic surface and a built-in scripting

interface for querying surface height and current velocity at runtime, which is directly required by the hull physics system.

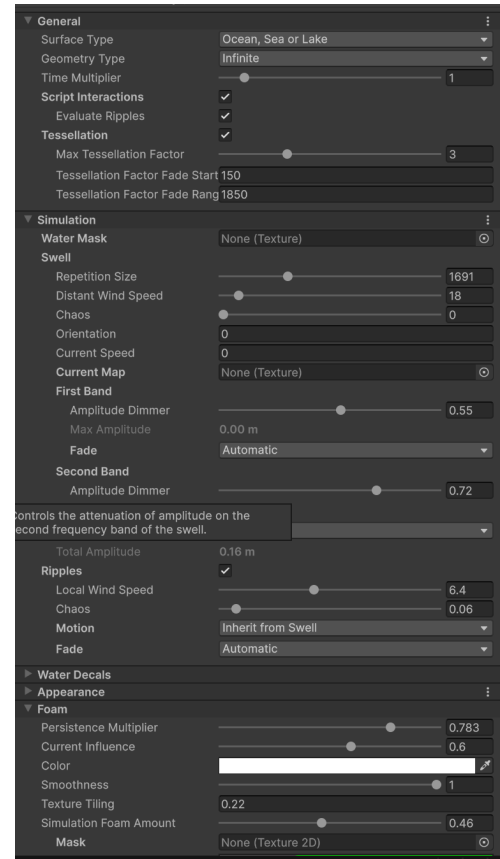
The HDRP water system simulates wave motion through two components: distant wind, which produces large swell-type waves, and local wind, which generates smaller surface ripples [24]. The surface type is configured as Ocean, Sea or Lake with an infinite geometry, providing an unbounded water plane around the vessel. Script Interactions and Evaluate Ripples are enabled [26], allowing the physics system to query accurate surface height including ripple contributions at each timestep. The swell is configured with a default distant wind speed of 18 m/s and a repetition size of 1691, producing moderate open-sea wave conditions. Ripples are enabled with a local wind speed of 6.4 m/s, adding smaller surface detail. The total swell amplitude is 0.16 m. Tessellation is enabled with a maximum tessellation factor of 3, improving the geometric resolution of the water surface near the vessel.

At each physics timestep, the WaterController queries the HDRP water surface to retrieve the surface elevation and current velocity at each triangle centroid of the underwater mesh, feeding this data directly into the buoyancy and drag calculations described in Section 4.3. The wave parameters are not static but are updated at runtime in response to the wind conditions set by the WindManager. The wind speed is mapped to the swell distant wind speed and ripple local wind speed, and the wind direction is mapped to the swell and ripple orientation. The swell band amplitudes are scaled proportionally with wind speed, and the chaos parameter increases as wind speed rises to produce choppier conditions. This coupling ensures that the visual and physical state of the ocean remain consistent with the aerodynamic environment experienced by the sail at all times.

A Water Excluder is applied to the interior of the hull to prevent the HDRP water surface from rendering inside the boat [27]. Without this, the water surface would appear inside the vessel as a visual artifact when the hull is partially submerged. The excluder is implemented as a separate mesh (the BoatLid) shaped to fit the open top of the hull, and is a child of the Boat GameObject so that it moves with the vessel at all times. The Water Excluder only affects the water surface rendering from above and does not interfere with the physics queries used for buoyancy and drag calculations.



(a) The simulation environment showing the vessel on the HDRP ocean surface.



(b) HDRP Water Surface component settings.

Figure 4.3: Ocean environment configuration in Unity HDRP.

4.2 Scene Managers

The simulation relies on three scene-level manager components that operate independently of the vessel hierarchy and provide shared services to the rest of the system. These are the WindManager, which maintains the global wind state and computes apparent wind, the WaterController, which interfaces with the HDRP water surface and synchronises its parameters with the wind state, and the SailWindReceiver, which translates the apparent wind into a physical acceleration applied to the cloth sail.

4.2.1 Wind Manager

The WindManager is a singleton MonoBehaviour that maintains the global wind state for the simulation. It exposes two primary parameters configurable at runtime: wind speed and wind direction, the latter expressed in degrees relative to the world forward axis. A turbulence model is implemented using Perlin noise [28, p. 289], which introduces subtle frame-to-frame variation

in both wind speed and direction. The turbulence offset is randomly initialised at startup to ensure variation between simulation runs. At each frame, the actual wind direction and speed are computed by adding the Perlin noise contribution to the base values, and the result is converted into a world-space wind vector using a rotation around the vertical axis.

The WindManager also exposes a `GetApparentWind()` method, which returns the apparent wind vector by subtracting the vessel's current linear velocity from the true wind vector. This method is called at each physics step by both the SailForceController and the SailWindReceiver, ensuring that all sail-related computations use a consistent apparent wind value.

4.2.2 Water Controller

The WaterController is a singleton MonoBehaviour that serves as the interface between the simulation's physics system and the HDRP water surface. It exposes a `GetWaveData()` method, which projects a given world-space position onto the water surface using Unity's `ProjectPointOnWaterSurface()` [29] API and returns a `WaterSearchResult`[30] containing the projected surface position and current velocity at that point. This method is called at each physics timestep by the hull physics system for every submerged triangle centroid, providing the surface height and water current velocity required by the buoyancy and drag calculations.

In addition, the WaterController synchronises the HDRP ocean parameters with the current wind state at each frame through the `UpdateWind()` method, as described in Section 4.1.3. A `DistanceToWater()` utility method is also provided, which returns the signed vertical distance between a given position and the water surface, positive above and negative below.

4.2.3 Sail Wind Receiver

The SailWindReceiver is a MonoBehaviour attached to the sail GameObject that translates the apparent wind into a physical force acting on the cloth simulation. At each frame, it retrieves the apparent wind vector from the WindManager and applies it as an external acceleration to the Unity Cloth component via the `externalAcceleration` property. The wind vector is transformed from world space into the local space of the sail before application, ensuring that the cloth deformation responds correctly to changes in sail orientation as the boom rotates. Additionally, the WindManager's turbulence value is applied as a random acceleration on the cloth, producing subtle flutter in the sail surface that varies with wind conditions. This component is responsible for driving the visible deformation of the sail mesh, which is in turn sampled by the SailCamberSampler to determine the instantaneous camber state used in the aerodynamic force calculations.

4.3 Hull Physics

4.3.1 Underwater Mesh Generation

At each frame, before any hydrodynamic forces are computed, the hull mesh is analysed to determine which portions of it lie below the water surface. This process is implemented in the `ModifyBoatMesh` class and produces a list of `TriangleData` structures representing the submerged geometry of the hull, which is then consumed by the force calculations in `BoatPhysics` at each fixed timestep.

4.3.1.1 Vertex Distance Computation

The first step is to transform all hull mesh vertices from local object space into world space and query the signed distance from each vertex to the water surface. A positive distance indicates that the vertex is above the water surface, and a negative distance indicates that it is below. This is performed by calling `WaterController.DistanceToWater()` for each vertex, which projects the vertex position onto the HDRP water surface and returns the difference between the vertex height and the surface height at that horizontal position:

$$d_i = y_i - y_{\text{surface}}(x_i, z_i)$$

where d_i is the signed distance of vertex i , y_i is its world-space height, and $y_{\text{surface}}(x_i, z_i)$ is the height of the water surface directly below or above it. Because the HDRP water surface is dynamic, this query accounts for the instantaneous wave elevation at each vertex position, ensuring that the submerged geometry is updated correctly as the vessel moves through waves.

4.3.1.2 Triangle Classification

Once the signed distance is known for every vertex, the algorithm iterates over all triangles of the hull mesh. Each triangle is defined by three vertices, and its submersion state is determined by the signs of the three corresponding distance values:

- 1: **for** each triangle (v_1, v_2, v_3) in hull mesh **do**
- 2: $d_1, d_2, d_3 \leftarrow$ signed distance to water surface for each vertex
- 3: **if** $d_1 > 0$ **and** $d_2 > 0$ **and** $d_3 > 0$ **then**
- 4: **discard** triangle ▷ entirely above water
- 5: **else if** $d_1 < 0$ **and** $d_2 < 0$ **and** $d_3 < 0$ **then**
- 6: **add** triangle as-is ▷ entirely submerged
- 7: **else**

- 8: **clip** triangle at waterline ▷ straddles waterline
 9: **add** resulting submerged sub-triangles

4.3.1.3 Waterline Clipping

When a triangle straddles the waterline, new intersection points are computed along the edges that cross the surface. The intersection point along an edge between a submerged vertex P_L (with distance $h_L < 0$) and an above-water vertex P_H (with distance $h_H > 0$) is found by linear interpolation:

$$t = \frac{-h_L}{h_H - h_L}, \quad P_I = P_L + t \cdot (P_H - P_L)$$

where $t \in [0, 1]$ is the interpolation parameter and P_I is the resulting intersection point on the water surface. This produces geometrically accurate clipping that places the new triangle edges exactly at the waterline, regardless of the local wave elevation.

Two sub-cases arise depending on how many vertices are above the waterline:

One vertex above water: the above-water vertex H is cut off, and two intersection points I_M and I_L are computed along the two edges connecting H to the submerged vertices M and L . This produces two new triangles below the waterline: (M, I_M, I_L) and (M, I_L, L) .

Two vertices above water: only the single submerged vertex L remains, and two intersection points J_H and J_M are computed along the edges connecting L to the two above-water vertices. This produces one new triangle below the waterline: (L, J_H, J_M) .

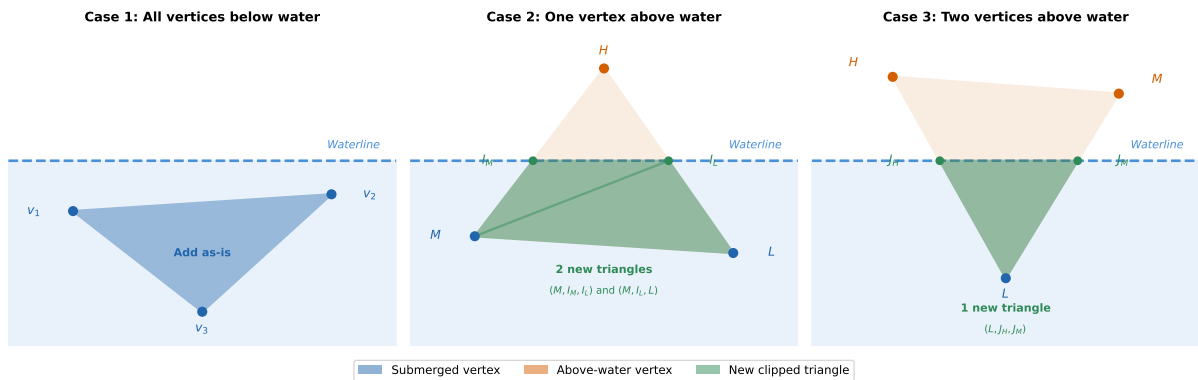


Figure 4.4: The three triangle classification cases in the underwater mesh generation algorithm. Case 1: all vertices submerged, triangle added as-is. Case 2: one vertex above water, triangle clipped into two new submerged triangles. Case 3: two vertices above water, triangle clipped into one new submerged triangle. Green triangles represent the clipped submerged geometry.

4.3.1.4 Triangle Data

Each submerged triangle (whether original or clipped) is stored as a `TriangleData` struct containing all quantities required by the force calculations. These include the three vertex positions, the centroid, the outward surface normal, the triangle area, the average submersion depth, and the center of pressure. The area is computed from the cross product of two edge vectors:

$$A = \frac{1}{2} |\vec{e}_1 \times \vec{e}_2| = \frac{ac \sin B}{2}$$

where a and c are the lengths of two edges and B is the angle between them. The average submersion depth is computed as the mean of the absolute depths of the three vertices, clamped to zero for above-water contributions:

$$\bar{d} = \frac{\max(0, -d_1) + \max(0, -d_2) + \max(0, -d_3)}{3}$$

The center of pressure is computed as a depth-weighted centroid of the three vertices, giving more weight to the portions of the triangle that are deeper below the surface:

$$\mathbf{P}_{\text{cp}} = \frac{h_1 \mathbf{p}_1 + h_2 \mathbf{p}_2 + h_3 \mathbf{p}_3}{h_1 + h_2 + h_3}$$

where $h_i = \max(0, -d_i)$ is the absolute submersion depth of vertex i . This ensures that the buoyancy force, which is applied at the center of pressure, acts at a physically meaningful point that reflects the pressure distribution across the triangle rather than simply at its geometric centroid.

4.3.2 Buoyancy and Damping

Buoyancy and hydrodynamic damping are computed together for each submerged triangle and applied at the triangle's center of pressure. The two forces act along the vertical axis and are accumulated on the RigidBody at each fixed timestep.

4.3.2.1 Buoyancy Force

The buoyancy force acting on each submerged triangle is derived from Archimedes' principle, approximating the upward hydrostatic pressure acting on the triangle face as a function of its average submersion depth and area. For each triangle, the buoyancy magnitude is computed as:

$$F_b = \rho g \bar{d} A$$

where $\rho = 1027 \text{ kg/m}^3$ is the density of seawater, g is the magnitude of gravitational acceleration, $\bar{d}$ is the average submersion depth of the triangle computed from the three vertex depths as described in Section 4.3.1, and A is the triangle area. The resulting force acts vertically upward and is applied at the center of pressure of the triangle rather than its geometric centroid, ensuring that the buoyant moment acting on the hull reflects the depth-weighted pressure distribution across the face.

This per-triangle formulation distributes the total buoyant force across the submerged surface of the hull, allowing the net upward force and its point of application to shift dynamically as the vessel pitches, rolls, and heaves through waves. This is essential for realistic stability behaviour, as the restoring moment that returns the vessel to upright depends on the lateral shift of the center of buoyancy as the hull tilts.

4.3.2.2 Hydrodynamic Damping

In addition to the hydrostatic buoyancy force, a velocity-dependent damping force is applied to each submerged triangle to suppress unrealistic vertical oscillations that would otherwise arise from the discrete timestep integration of the buoyancy force. Without damping, the vessel would bounce indefinitely on the water surface, as the buoyancy force at each step slightly overshoots the equilibrium position.

The damping force is proportional to the vertical component of the relative velocity between the triangle and the local water surface, and to the triangle area:

$$F_{\text{damp}} = -v_{\perp} \cdot k_d \cdot A$$

where $v_{\perp} = (\mathbf{v}_{\text{point}} - \mathbf{v}_{\text{water}}) \cdot \hat{y}$ is the vertical component of the relative velocity between the triangle centroid and the water current at that point, $k_d = 800$ is the damping coefficient, and A is the triangle area. The negative sign ensures that the damping force opposes the vertical motion. The water current velocity at each triangle is obtained from the WaterController query, so the damping correctly accounts for wave-induced vertical water motion and not just the absolute velocity of the hull.

Both the buoyancy and damping forces act along the vertical axis and are summed before being applied to the RigidBody at the triangle's center of pressure:

$$\mathbf{F}_{\text{hydrostatic}} = (F_b + F_{\text{damp}}) \hat{y}$$

4.3.3 Pressure Drag

Pressure drag arises from the asymmetric distribution of hydrodynamic pressure around the moving hull. As the vessel advances through the water, higher pressure builds at faces oriented toward the direction of motion and lower pressure develops at faces oriented away from it. The net effect is a force that opposes the motion of the hull.

In this implementation, pressure drag is computed per submerged triangle by projecting the relative velocity between the triangle and the water onto the triangle's outward surface normal. This isolates the component of motion that acts perpendicularly to the face, the component responsible for pressure buildup, from the tangential component, which is handled separately as friction drag in Section 4.3.4.

For each submerged triangle, the relative velocity between the triangle centroid and the local water current is first computed:

$$\mathbf{v}_{\text{rel}} = \mathbf{v}_{\text{point}} - \mathbf{v}_{\text{water}}$$

where $\mathbf{v}_{\text{point}}$ is the velocity of the triangle centroid on the RigidBody, obtained via `GetPointVelocity()`, and $\mathbf{v}_{\text{water}}$ is the current velocity at that position returned by the WaterController query. The normal component of the relative velocity is then extracted:

$$v_n = \mathbf{v}_{\text{rel}} \cdot \hat{n}, \quad \mathbf{v}_n = v_n \hat{n}$$

where $\hat{n}$ is the outward surface normal of the triangle. The pressure drag force is then computed using the standard quadratic drag formulation applied to the normal velocity component:

$$F_{\text{pressure}} = \frac{1}{2} \rho |\mathbf{v}_n|^2 A C_D$$

where $\rho = 1027 \text{ kg/m}^3$ is the water density, A is the triangle area, and C_D is the drag coefficient, set to 1.0 in the current implementation. The force acts in the direction opposite to the normal velocity component, opposing the penetration of the face into the water:

$$\mathbf{F}_{\text{pressure}} = F_{\text{pressure}} \cdot (-\text{sign}(v_n) \hat{n})$$

This formulation ensures that only faces actively moving into the water contribute a pressure drag force. Faces moving away from the flow direction produce no pressure drag contribution, which is consistent with the physical behaviour of separated flow around a bluff hull. The force is applied at the geometric centroid of each triangle and accumulated on the RigidBody at each

fixed timestep.

4.3.4 Friction Drag

Friction drag arises from the viscous interaction between the hull surface and the water flowing along it. Unlike pressure drag, which acts perpendicular to the hull face, friction drag acts tangentially, opposing the component of relative motion that runs parallel to the surface. It is associated with the formation of a boundary layer along the hull and is the dominant resistance component at low speeds.

The tangential component of the relative velocity is obtained by subtracting the normal component from the total relative velocity:

$$\mathbf{v}_t = \mathbf{v}_{\text{rel}} - \mathbf{v}_n = \mathbf{v}_{\text{rel}} - (\mathbf{v}_{\text{rel}} \cdot \hat{n}) \hat{n}$$

where $\mathbf{v}_{\text{rel}}$ is the relative velocity between the triangle centroid and the local water current, and $\hat{n}$ is the outward surface normal. The friction drag force is then computed using a quadratic formulation analogous to pressure drag, but applied to the tangential velocity component:

$$F_{\text{friction}} = \frac{1}{2} \rho |\mathbf{v}_t|^2 A C_f$$

where $\rho = 1027 \text{ kg/m}^3$ is the water density, A is the triangle area, and $C_f = 0.02$ is a constant friction coefficient. The force acts in the direction opposing the tangential motion:

$$\mathbf{F}_{\text{friction}} = F_{\text{friction}} \cdot (-\hat{v}_t)$$

The friction coefficient $C_f = 0.02$ is a simplified constant value used in place of the ITTC-57 correlation line, which would express C_f as a function of the Reynolds number. As discussed in Section 2.2, the full ITTC-57 formulation is identified as future work. The constant value adopted here provides a reasonable approximation of viscous resistance for the speed range expected in the simulation, while keeping the implementation computationally straightforward. The friction drag force is applied at the geometric centroid of each triangle and accumulated on the RigidBody together with the pressure drag at each fixed timestep.

4.3.5 Lateral Resistance

Lateral resistance is the hydrodynamic force that opposes sideways motion of the hull through the water. In sailing, the aerodynamic forces generated by the sail include a significant lateral component that tends to push the vessel sideways, a motion known as leeway. Without sufficient

lateral resistance, the vessel would drift downwind and be unable to convert sail forces into forward propulsion. The hull geometry, particularly the deep sections near the keel, resists this sideways motion by generating a lift-like opposing force.

In this implementation, lateral resistance is computed per submerged triangle by isolating the component of relative velocity that acts perpendicular to the vessel's forward axis. The forward axis of the vessel is defined as the negative local right vector of the Rigidbody transform. The lateral velocity component is extracted by removing the forward projection from the total relative velocity:

$$\mathbf{v}_{\text{lat}} = \mathbf{v}_{\text{rel}} - \text{proj}_{\hat{f}} \mathbf{v}_{\text{rel}} = \mathbf{v}_{\text{rel}} - (\mathbf{v}_{\text{rel}} \cdot \hat{f}) \hat{f}$$

where $\hat{f}$ is the unit vector along the vessel's forward axis and $\mathbf{v}_{\text{rel}}$ is the relative velocity between the triangle centroid and the local water current. This isolates the purely sideways component of motion, independent of the vessel's forward speed.

The lateral resistance force is then computed using a quadratic formulation, weighted by the degree to which the triangle face is oriented sideways relative to the vessel:

$$F_{\text{lat}} = \frac{1}{2} \rho |\mathbf{v}_{\text{lat}}|^2 A C_k \alpha$$

where $\rho = 1027 \text{ kg/m}^3$ is the water density, A is the triangle area, $C_k = 2.0$ is the lateral resistance coefficient, and α is an alignment factor defined as:

$$\alpha = |\hat{n} \cdot \hat{f}_{\perp}|$$

where $\hat{n}$ is the outward normal of the triangle and $\hat{f}_{\perp}$ is the vessel's transverse axis. This alignment factor ensures that only faces oriented sideways, such as the deep hull sections and keel surfaces, contribute significantly to lateral resistance, while faces oriented toward the bow or stern contribute little. The lateral resistance force acts in the direction opposing the lateral velocity:

$$\mathbf{F}_{\text{lat}} = F_{\text{lat}} \cdot (-\hat{v}_{\text{lat}})$$

The lateral resistance force is accumulated alongside the pressure and friction drag forces and applied at the geometric centroid of each triangle at each fixed timestep. Together, these three drag components constitute the total hydrodynamic resistance acting on the submerged hull surface.

4.3.6 Rudder Force Model

The rudder force model is implemented in the `RudderController` component and computes the hydrodynamic lift and drag forces generated by the rudder at each physics timestep, applying them to the RigidBody at the rudder pivot position to produce a yawing moment that changes the vessel's heading.

4.3.6.1 Lift Force

The rudder lift coefficient is computed using thin aerofoil theory, which relates the lift coefficient to the angle of attack through a linear relationship valid for small deflection angles [10]:

$$C_{L_r} = 2\pi \sin(\delta)$$

where δ is the rudder deflection angle in radians. This formulation captures the approximately linear increase of lift with deflection angle in the attached flow regime. The coefficient is clamped to the range $[-1.5, 1.5]$ to prevent unrealistic values at large deflection angles where flow separation would occur in practice. The resulting lift force is:

$$F_{L_r} = C_{L_r} \cdot q \cdot A_r$$

where $q = \frac{1}{2}\rho V^2$ is the dynamic pressure, $\rho = 1027 \text{ kg/m}^3$ is the water density, V is the vessel speed, and $A_r = 1.0 \text{ m}^2$ is the rudder planform area. The lift force acts in the forward direction of the rudder pivot transform, which rotates with the rudder deflection, ensuring that the force direction correctly reflects the current rudder orientation.

4.3.6.2 Induced Drag

As with the sail, the rudder generates induced drag as a consequence of lift production. The induced drag coefficient is computed using the same finite-wing formulation applied to the sail in Section 4.4.5:

$$C_{D_i} = \frac{C_{L_r}^2}{\pi \cdot AR \cdot e}$$

where $AR = 2.5$ is the rudder aspect ratio and $e = 0.85$ is the Oswald efficiency factor. A baseline parasitic drag coefficient of 0.01 is added to account for skin friction on the rudder surface, giving the total drag coefficient:

$$C_{D_{\text{total}}} = 0.01 + C_{D_i}$$

The drag force opposes the vessel's velocity vector:

$$\mathbf{F}_{D_r} = C_{D_{\text{total}}} \cdot q \cdot A_r \cdot (-\hat{v})$$

where $\hat{v}$ is the unit vector of the vessel's linear velocity.

4.3.6.3 Force Application

The total rudder force is the vector sum of lift and drag, scaled by a configurable force scale factor and applied to the Rigidbody at the rudder pivot position:

$$\mathbf{F}_{\text{rudder}} = (F_{L_r} \cdot \hat{l}_r + \mathbf{F}_{D_r}) \cdot k_s$$

where $\hat{l}_r$ is the lift direction defined by the rudder pivot forward vector and k_s is the force scale factor. Applying the force at the rudder pivot position rather than at the center of mass ensures that the force generates the correct yawing moment about the vessel's vertical axis, turning the bow in the direction of rudder deflection. The force calculation is skipped entirely when the vessel speed falls below 0.1 m/s, as the rudder generates negligible force at very low speeds and the computation would introduce numerical noise.

4.4 Sail Aerodynamics

This section describes the aerodynamic force model acting on the sail, implemented in the `SailForceController` component. The model computes lift and drag forces at each physics timestep based on the apparent wind, the instantaneous angle of attack, and the camber state of the cloth sail. The resulting forces are applied to the Rigidbody at the sail's center of effort.

4.4.1 Apparent Wind

As established in Section 2.3, the aerodynamic forces acting on a sail depend not on the true wind but on the apparent wind, the wind experienced by a moving vessel. At each physics timestep, the apparent wind vector is retrieved from the `WindManager` via the `GetApparentWind()` method, which computes it as:

$$\mathbf{V}_a = \mathbf{V}_t - \mathbf{V}_b$$

where $\mathbf{V}_t$ is the true wind vector in world space and $\mathbf{V}_b$ is the current linear velocity of the vessel's Rigidbody. This ensures that the aerodynamic force calculation always reflects the combined

effect of wind and vessel motion, and that changes in vessel speed automatically update the aerodynamic loading on the sail.

If the apparent wind speed falls below a threshold of 0.05 m/s, the force calculation is skipped entirely for that timestep, as the resulting forces would be negligibly small and could introduce numerical instability.

4.4.2 Angle of Attack

The angle of attack is the angle between the sail chord direction and the apparent wind vector, measured in the plane perpendicular to the mast span axis. It is the primary parameter that determines which region of the aerodynamic polar curves is active at any given moment.

4.4.2.1 Span Axis Projection

Because the mast is not always perfectly vertical (the vessel heels under sail forces) the angle of attack must be computed in the plane of the sail rather than in the horizontal plane. This is achieved by projecting both the apparent wind vector and the sail chord direction onto the plane perpendicular to the mast span axis before computing the angle between them:

$$\mathbf{w}_{\text{flat}} = \mathbf{V}_a - (\mathbf{V}_a \cdot \hat{s}) \hat{s}, \quad \mathbf{c}_{\text{flat}} = \mathbf{c} - (\mathbf{c} \cdot \hat{s}) \hat{s}$$

where $\mathbf{V}_a$ is the apparent wind vector, $\hat{s}$ is the unit vector along the mast span axis (the local up direction of the Istos transform), and $\mathbf{c}$ is the chord direction of the sail defined as the local right vector of the BoomPivot transform. Both projected vectors are normalised before the angle is computed.

4.4.2.2 Signed Angle Computation

The signed angle of attack is computed between the projected chord direction and the projected wind direction using `Vector3.SignedAngle()` about the world up axis:

$$\alpha = \angle(\mathbf{c}_{\text{flat}}, \mathbf{w}_{\text{flat}})$$

The sign of the angle indicates on which side of the sail the wind is acting, which in turn determines the direction of the lift force. The angle is then normalised to the range $[-90^\circ, +90^\circ]$ using the following mapping:

$$\alpha \leftarrow \begin{cases} 180^\circ - \alpha & \text{if } \alpha > 90^\circ \\ -180^\circ - \alpha & \text{if } \alpha < -90^\circ \end{cases}$$

This ensures that angles beyond $\pm 90^\circ$, which correspond to wind coming from behind the sail, are folded back into the valid operating range of the polar curves. The absolute value $|\alpha|$ is used to index the polar curves, which are defined over the range $[0^\circ, 90^\circ]$, while the sign of α is retained separately to determine the direction of the lift force vector.

4.4.2.3 Aspect Ratio from SailEdgeSampler

The `SailEdgeSampler` component identifies the three corner vertices of the sail mesh, head, tack, and clew, at runtime by analysing the triangle topology of the cloth mesh. Corner vertices are identified as those shared by the fewest triangles in the mesh. The head is the highest corner in world space, and the clew is distinguished from the tack by its greater distance from the boom pivot anchor.

The effective mast height is estimated from the distance between the head and tack vertices, corrected by a factor of $1/0.71$ to account for the non-uniform world-space scale applied to the sail mesh in the Unity scene:

$$h_{\text{mast}} = \frac{|\mathbf{p}_{\text{head}} - \mathbf{p}_{\text{tack}}|}{0.71}$$

where the divisor corresponds to the lossy scale factor of approximately 0.71.

The aspect ratio is then computed once at startup as:

$$AR = \frac{h_{\text{mast}}^2}{A_{\text{sail}}}$$

where A_{sail} is the sail area set in the inspector. This value is passed to the `SailForceController` and used in the induced drag calculation described in Section 4.4.5.

4.4.3 Camber Sampling

The camber of the sail determines which polar curves are active in the aerodynamic force calculation. Rather than using a fixed camber value, the simulation measures the instantaneous camber of the cloth sail at each physics timestep by analysing the deformation of the cloth mesh vertices. This is implemented in the `SailCamberSampler` component.

4.4.3.1 Slice Selection

The sampler operates by taking a horizontal slice of the sail mesh at a configurable height fraction along the sail. A height fraction of 0.25 is used, meaning the slice is taken at one quarter of the sail height from the foot, a position recommended for triangular sails where the chord is wide enough to provide a reliable camber measurement. The slice thickness is set to 30% of the sail height, collecting all vertices whose height falls within that band.

4.4.3.2 Chord and Depth Measurement

Within the slice, the luff and leech vertices are identified as the vertices with the minimum and maximum projection onto the chord axis respectively. The chord length is the world-space distance between these two points. The depth profile of the sail is then sampled at ten equally spaced positions along the chord. At each sample position, the nearest vertex in the slice is found, and its depth, measured along the sail's forward axis relative to the straight chord line connecting luff to leech, is recorded:

$$d_i = \left| (\mathbf{v}_i \cdot \hat{d}) - \left[(1 - t_i)(\mathbf{p}_{\text{luff}} \cdot \hat{d}) + t_i(\mathbf{p}_{\text{leech}} \cdot \hat{d}) \right] \right|$$

where $\hat{d}$ is the depth axis (the local forward direction of the sail transform), $t_i \in [0, 1]$ is the normalised chord position of sample i , and $\mathbf{p}_{\text{luff}}$, $\mathbf{p}_{\text{leech}}$ are the luff and leech vertex positions.

4.4.3.3 Smoothness Correction

The raw camber ratio is computed as the mean depth across the ten samples divided by the chord length. However, a poorly shaped or partially collapsed sail may produce an irregular depth profile with high variance. To suppress this, a smoothness factor is computed from the coefficient of variation of the depth profile:

$$CV = \frac{\sigma_d}{\mu_d}, \quad k_s = \text{clamp}(1 - CV, 0, 1)$$

where σ_d is the standard deviation and μ_d is the mean of the depth samples. The effective camber ratio is then:

$$\gamma_{\text{eff}} = \frac{\mu_d}{c} \cdot k_s$$

where c is the chord length. A high coefficient of variation reduces the effective camber, reflecting the fact that an irregular sail shape is aerodynamically less efficient than a smooth one.

4.4.3.4 Temporal Smoothing and Normalisation

The effective camber is smoothed over time using an exponential moving average with a smoothing factor of 0.85, reducing frame-to-frame jitter caused by cloth simulation noise:

$$\gamma_{\text{smooth}}^{(t)} = (1 - \alpha) \cdot \gamma_{\text{eff}}^{(t)} + \alpha \cdot \gamma_{\text{smooth}}^{(t-1)}, \quad \alpha = 0.85$$

The smoothed camber ratio is finally normalised to the range $[0, 1]$ using configurable minimum and maximum camber ratio bounds, producing the CamberT value:

$$\text{CamberT} = \text{clamp} \left(\frac{\gamma_{\text{smooth}} - \gamma_{\text{min}}}{\gamma_{\text{max}} - \gamma_{\text{min}}}, 0, 1 \right)$$

where $\gamma_{\text{min}} = 0.0$ and $\gamma_{\text{max}} = 0.15$ are the normalisation bounds. A value of 0 corresponds to a flat sail and a value of 1 corresponds to a fully cambered sail.

4.4.4 Polar Curves and Coefficient Blending

The aerodynamic lift and drag coefficients are determined at each timestep by evaluating a set of pre-defined polar curves and interpolating between them based on the instantaneous camber state of the sail.

4.4.4.1 Three-State Polar Curves

Three pairs of polar curves are defined, each corresponding to a distinct camber state: flat ($\gamma = 0$, $\text{CamberT} = 0$), mid ($\gamma = 6^\circ$, $\text{CamberT} = 0.5$), and full ($\gamma = 12^\circ$, $\text{CamberT} = 1.0$). Each curve is implemented as a Unity `AnimationCurve` that maps angle of attack in degrees to a coefficient value. The curve data is based on published aerodynamic data for cambered sail sections [14–16], covering the range from 0° to 90° angle of attack. All three states share a common stall angle of approximately 25° , beyond which C_L drops sharply and C_D increases rapidly, consistent with separated flow behaviour reported in the literature.

4.4.4.2 Camber Blending

The instantaneous coefficients are obtained by linearly interpolating between adjacent camber states based on the CamberT value provided by the `SailCamberSampler`:

$$C_L(\alpha, \gamma) = \begin{cases} \text{lerp}(C_L^{\text{flat}}(\alpha), C_L^{\text{mid}}(\alpha), 2 \cdot \text{CamberT}) & \text{if } \text{CamberT} \leq 0.5 \\ \text{lerp}(C_L^{\text{mid}}(\alpha), C_L^{\text{full}}(\alpha), 2(\text{CamberT} - 0.5)) & \text{if } \text{CamberT} > 0.5 \end{cases}$$

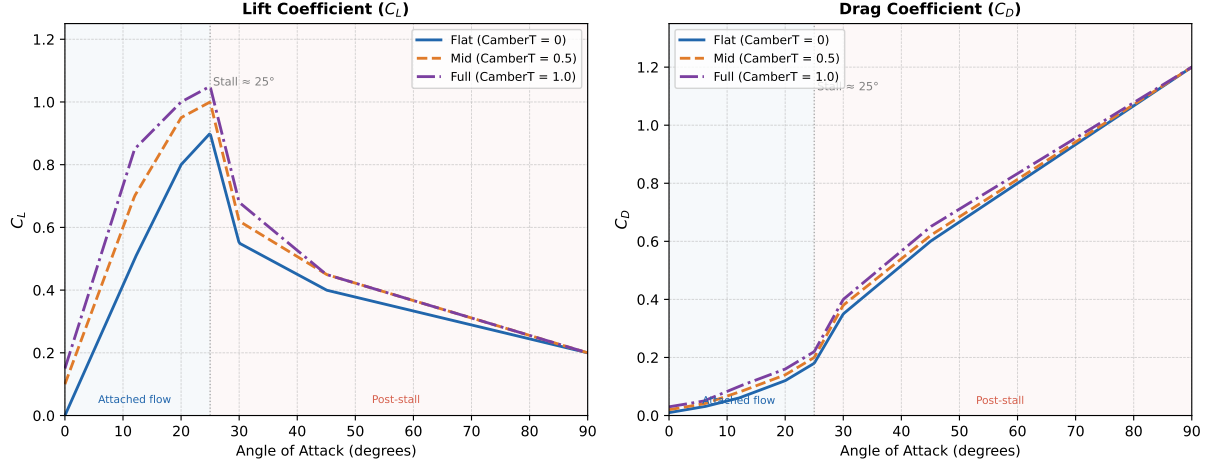


Figure 4.5: Lift coefficient C_L (left) and drag coefficient C_D (right) as functions of angle of attack for the three camber states implemented in the simulation. All states share a stall angle of approximately 25° , beyond which C_L drops sharply and C_D increases rapidly.

The same blending is applied to C_D . This piecewise linear interpolation ensures smooth and continuous variation of the aerodynamic coefficients as the sail camber changes, without requiring a full polar curve for every intermediate camber state.

4.4.5 Induced Drag

As described in Section 2.3, a sail of finite span generates induced drag as a consequence of lift production. The induced drag coefficient is computed at each timestep from the instantaneous lift coefficient, the sail aspect ratio, and the Oswald efficiency factor:

$$C_{D_i} = \frac{C_L^2}{\pi \cdot AR \cdot e}$$

where AR is the aspect ratio computed by the `SailEdgeSampler` as described in Section 4.4.2, and $e = 0.85$ is the Oswald efficiency factor, which accounts for deviations from the ideal elliptical lift distribution [14]. The induced drag coefficient is added to the profile drag coefficient obtained from the polar curves to give the total drag coefficient:

$$C_{D_{\text{total}}} = C_D + C_{D_i}$$

This total coefficient is then used in the drag force calculation described in Section 4.4.6.

4.4.6 Force Application and Center of Effort

Once the lift and drag coefficients and the apparent wind speed are known, the aerodynamic forces are computed using the standard dynamic pressure formulation. The dynamic pressure is first calculated as:

$$q = \frac{1}{2} \rho V_a^2$$

where $\rho = 1.225 \text{ kg/m}^3$ is the air density and V_a is the apparent wind speed. The lift and drag forces are then:

$$\mathbf{F}_L = C_L \cdot q \cdot A \cdot \hat{l}, \quad \mathbf{F}_D = C_{D_{\text{total}}} \cdot q \cdot A \cdot \hat{w}$$

where A is the sail area, $\hat{w}$ is the apparent wind direction, and $\hat{l}$ is the lift direction, defined as the cross product of the apparent wind direction and the mast span axis, with its sign determined by the sign of the angle of attack:

$$\hat{l} = \text{sign}(\alpha) \cdot (\hat{w} \times \hat{s})$$

This ensures that lift acts perpendicular to the apparent wind and on the correct side of the sail depending on which tack the vessel is on.

The total aerodynamic force is the vector sum of lift and drag:

$$\mathbf{F}_{\text{aero}} = \mathbf{F}_L + \mathbf{F}_D$$

This force is applied to the RigidBody at the Center of Effort, a fixed transform defined within the sail hierarchy at a point representative of the aerodynamic centroid of the sail. Applying the force at this point rather than at the vessel's center of mass ensures that the aerodynamic loading generates the correct heeling and yawing moments on the hull, contributing to realistic vessel response under sail.

4.5 User Controls

This section describes the input systems through which the user interacts with the simulation. Two primary controls are available: boom rotation, which adjusts the sail's angle of attack relative to the apparent wind, and rudder deflection, which controls the vessel's heading. Wind settings are also adjustable at runtime through the WindManager inspector interface.

4.5.1 Boom Control

Boom rotation is handled by the `PodeonasController` component, which reads user input and applies it as a rotation to the `BoomPivot` transform. The component uses Unity's Input System [31] to read a two-axis input action (`Sailing.AdjustPodeona`), where the horizontal axis controls the rotation of the boom around the mast (yaw), and the vertical axis controls a small vertical tilt of the boom (pitch).

4.5.1.1 Camera-Relative Input

The horizontal rotation input is made camera-relative, so that pushing the input in a given direction always rotates the boom in the visually expected direction regardless of the camera's orientation around the vessel. This is achieved by comparing the camera's forward direction to the boat's forward direction:

$$\delta_{\text{input}} = \begin{cases} +\delta_x & \text{if } \hat{c} \cdot \hat{f} \geq 0 \\ -\delta_x & \text{if } \hat{c} \cdot \hat{f} < 0 \end{cases}$$

where $\hat{c}$ is the horizontal component of the camera forward vector, $\hat{f}$ is the boat forward vector, and δ_x is the raw horizontal input value. When the camera is behind the vessel the input direction is preserved; when the camera is in front it is inverted, preventing the control from feeling reversed.

4.5.1.2 Rotation Limits

The boom rotation angle is accumulated over time and clamped to a configurable range of $[-80^\circ, +80^\circ]$ around the mast axis, preventing the boom from rotating through the hull. A small vertical tilt is also permitted within the range $[-5^\circ, +20^\circ]$ to allow the boom to be raised or lowered slightly. The resulting rotation is applied directly to the `BoomPivot` local transform as a combined Euler rotation:

$$\mathbf{R}_{\text{boom}} = \text{Euler}(0^\circ, \theta_y, \theta_z)$$

where θ_y is the accumulated horizontal angle and θ_z is the accumulated vertical tilt. Since the `BoomPivot` is a child of the `Istos` (mast) transform, this rotation is automatically expressed relative to the current mast orientation, correctly accounting for any heel or pitch of the vessel.

4.5.1.3 Effect on Sail Aerodynamics

Rotating the boom directly changes the chord direction of the sail, which in turn changes the angle of attack computed by the `SailForceController` at each physics step. This creates the primary control loop of the simulation: the user adjusts the boom to find the angle of attack that maximises forward drive for the current point of sail, and the physical response of the vessel provides immediate feedback on the effect of that adjustment.

4.5.2 Sheet Controller

The sheet controller (`SkottaController`) was designed to give the user direct control over sail camber by simulating the easing and hardening of the mainsheet (the rope that connects the clew of the sail to a fixed anchor point on the hull). Several approaches were explored before the fundamental limitation of Unity's cloth constraint system was identified.

4.5.2.1 Approach 1 — Bone Movement

The first attempt moved the Skotta bone directly in world space toward a target position computed from the wind direction and the current rope length, with the intention that the clew of the sail would follow the bone. This approach failed because Unity's cloth simulation does not follow bone movement once the Cloth component is active, cloth vertices are driven entirely by the cloth solver and are not affected by skeleton transforms at runtime.

4.5.2.2 Approach 2 — Boom Angle Restriction

The second approach modelled the sheet as a geometric constraint on the boom rotation. If the distance between the rope anchor and the sail tip exceeded the current rope length, the boom rotation angle was clamped to prevent further rotation away from the anchor. While this produced a physically plausible boom behaviour, it had no effect on sail camber, the cloth shape remained entirely unaffected by rope length.

4.5.2.3 Approach 3 — Bone Movement Combined with Per-vertex Constraint

The third approach combined bone movement with dynamic updates to the `maxDistance` cloth constraint on a single hardcoded corner vertex, with the constraint value mapped proportionally to the rope length. This was closer in principle to the correct approach, but still relied on bone movement which does not affect cloth vertices, and the hardcoded vertex index made it fragile to mesh changes.

4.5.2.4 Approach 4 — Pure Per-vertex maxDistance Constraint

The fourth and most technically developed approach abandoned bone movement entirely and used only Unity's `ClothSkinningCoefficient.maxDistance` [19,32] constraint, applied exclusively to the automatically identified clew vertices. The `maxDistance` parameter defines the maximum distance each vertex is allowed to move from its rest position (the position it would occupy if the cloth were driven purely by the skeleton). Setting it to zero pins the vertex in place, increasing it allows progressively more freedom of movement. By mapping the user's rope input to this constraint value, tightening the sheet corresponded to reducing `maxDistance` and easing it corresponded to increasing it.

This approach functioned correctly when the boom was at its default rotation, where the clew vertices' rest positions coincided approximately with the sheet anchor on the hull. However, a fundamental limitation was encountered when the boom was rotated: because `maxDistance` is always measured relative to the vertex's rest position in the bone's local space, rotating the boom displaced the rest positions of the clew vertices away from the anchor point. As a result, constraining the `maxDistance` no longer pulled the clew toward the anchor, it constrained it relative to the rotated rest position, which bore no geometric relationship to the anchor. The sheet control therefore produced incorrect behaviour at any boom angle other than the default, and the controller was deactivated.

The correct solution would require either a position-based constraint that operates in world space rather than relative to the bone rest position, or a custom cloth solver that directly enforces a distance constraint between the clew vertex and the anchor transform at each timestep. This is identified as future work.

4.5.3 Rudder Control

Rudder deflection is handled by the `RudderController` component, which reads user input and applies it as a rotation to the rudder pivot transform. The component uses Unity's Input System [31] to read a single-axis input action (`Sailing.SteerRudder`), where a positive value deflects the rudder to starboard and a negative value deflects it to port.

The deflection angle is accumulated over time and clamped to a maximum of $\pm 35^\circ$, preventing the rudder from over-rotating beyond the physically plausible range. The rudder pivot transform is updated each frame to reflect the current deflection angle, and the resulting rotation drives the hydrodynamic force calculation described in Section 4.3.6. The rudder force is applied at the rudder pivot position at each fixed physics timestep, generating a yawing moment that rotates the vessel about its vertical axis and changes its heading.

A real-time debug overlay displays the current rudder deflection angle, the instantaneous lift and

drag forces generated by the rudder, and the vessel yaw rate, providing the user with immediate feedback on the effect of rudder inputs.

4.5.4 Wind Manager

In the current implementation, wind conditions are configured through the WindManager inspector interface rather than through an in-simulation user control. The user can set the base wind speed and wind direction before or during a simulation run by modifying the exposed fields in the Unity Inspector. A wind direction arrow indicator in the scene provides visual feedback of the current apparent wind direction relative to the vessel, as described in Section 4.6. The development of a runtime wind control interface accessible during play mode is identified as future work, as part of the broader educator dashboard described in Chapter 3.

4.6 User Interface

The user interface of the simulation consists of four components: a debug overlay that displays real-time simulation state variables, a compass indicator that shows the vessel's heading and apparent wind direction, a camera system that follows the vessel during simulation, and a persistent trail renderer that records the vessel's path through the water.

4.6.1 Debug HUD

The debug HUD is implemented using Unity's OnGUI system [33] and renders a set of real-time simulation state variables as text labels in the top-left corner of the screen. The following quantities are displayed at each frame:

- Apparent wind speed and vessel speed in m/s
- Vessel heel angle in degrees
- Boom rotation angle in degrees
- Rudder deflection angle

This overlay serves as the primary feedback channel during development and testing, allowing the user to observe the immediate effect of control inputs on the aerodynamic state of the sail and the physical response of the vessel.

4.6.2 Compass Indicator

The compass indicator provides the user with a visual reference for the vessel's heading and the apparent wind direction. It is implemented as a secondary orthographic camera (UIArrowCamera)

positioned far above the scene, pointing downward at a 3D arrow GameObject rendered on a dedicated ArrowOnly culling layer. This camera renders only the arrow object and composites it over the main view in the top-right corner of the screen via its viewport rectangle settings.

The WindArrowRotator component rotates the 3D arrow each frame to point toward world north relative to the vessel's current heading. The heading angle is computed by transforming the world forward vector into the boat's local space and extracting the signed angle about the vertical axis:

$$\theta_{\text{heading}} = \text{atan2}(f_x^{\text{local}}, f_z^{\text{local}}) \cdot \frac{180^\circ}{\pi}$$

where f_x^{local} and f_z^{local} are the local-space components of the world forward vector. The arrow rotation is smoothed using spherical linear interpolation (Slerp) to avoid abrupt changes as the vessel turns. The current heading angle is displayed as a numeric label below the arrow. In addition, the apparent wind angle relative to the vessel and the apparent wind speed are displayed as a text label (AW: angle° speed m/s) adjacent to the arrow, giving the user immediate awareness of both heading and wind state in a single compact indicator.

4.6.3 Vessel Trail

A visual trail is rendered behind the vessel using Unity's built-in TrailRenderer component [34], attached to the hull GameObject. The trail records the path followed by the vessel through the ocean surface, providing a persistent visual record of the vessel's trajectory that helps the user understand how the boat responds to control inputs over time, for example, how the heading changes in response to rudder deflection or how the vessel curves when tacking.

The trail is configured with a constant width of 1.0 unit and a cyan colour. The time parameter is set to Infinity, meaning the trail persists for the entire duration of the simulation session without fading. A minimum vertex distance of 0.1 metres ensures that new trail points are only added when the vessel has moved a meaningful distance, keeping the trail geometry smooth without excessive vertex density at low speeds.

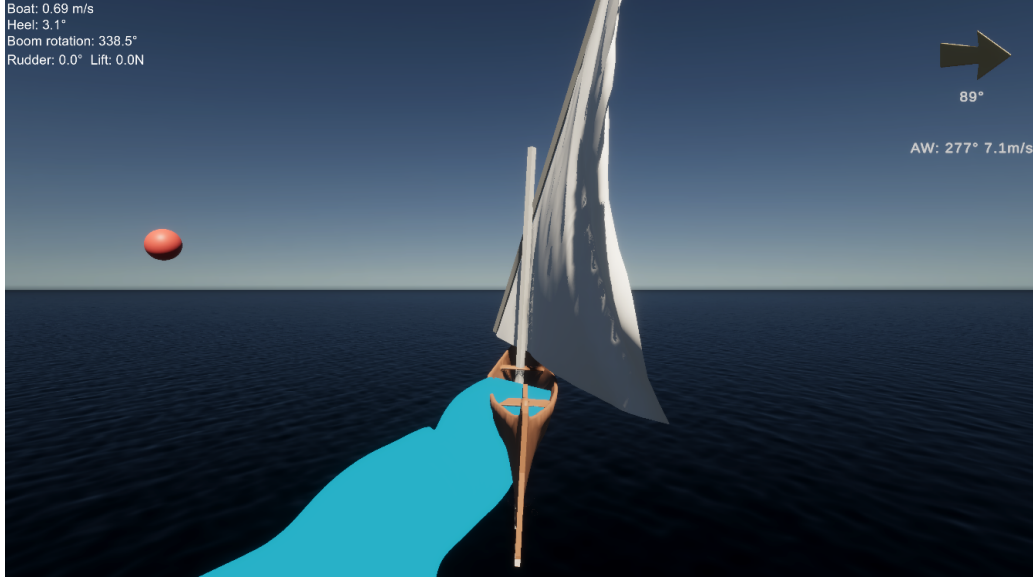


Figure 4.6: The simulation interface showing the debug HUD in the top-left corner and the compass and apparent wind indicator in the top-right corner.

4.6.4 Camera System

The main simulation camera follows the vessel using the `CameraFollow` component, which maintains a fixed offset from the vessel while rotating to track its heading. At startup, the initial world-space offset between the camera and the vessel is computed and stored. During each late update, this offset is rotated by the vessel's current yaw angle to produce the desired camera position:

$$\mathbf{p}_{\text{cam}} = \mathbf{p}_{\text{boat}} + R_y(\psi) \cdot \mathbf{d}_0$$

where $\mathbf{p}_{\text{boat}}$ is the vessel position, $R_y(\psi)$ is a rotation matrix about the vertical axis by the vessel's yaw angle ψ , and $\mathbf{d}_0$ is the initial offset vector. The camera position is smoothly interpolated toward the desired position using linear interpolation, and the camera orientation is smoothly interpolated toward a rotation that keeps the vessel centred in the frame while locking the vertical tilt to its initial editor-configured value. This prevents the camera from rolling or pitching as the vessel heels, maintaining a stable and readable view of the simulation at all times.

5 Testing and Evaluation

5.1 Overview

This chapter presents the testing and evaluation of the sailing simulation developed in this work. The evaluation is structured in two parts. The first part consists of technical testing, in which the physical behaviour of the simulation is assessed against expected behaviour derived from sailing theory. Two sailing performance tests are conducted: a downwind running test that examines the relationship between boom angle and vessel speed at an apparent wind angle of 180° , and a beam reach test that examines the same relationship at an apparent wind angle of approximately 90° . The second part consists of a task-based expert evaluation with semi-structured interviews, in which five to six experienced Ten-oared Whaleboat sailors interact with the simulation and assess its physical plausibility, control intuitiveness, and feedback quality.

The objective of the technical tests is not to validate the simulation against real vessel performance data, which would require instrumented sea trials, but to assess whether the simulation produces physically plausible and internally consistent responses to control inputs that align with the qualitative predictions of sailing theory. The user study complements this by providing qualitative and quantitative assessment from the perspective of domain experts with practical sailing experience.

5.2 Technical Testing

5.2.1 Downwind Sailing Test

5.2.1.1 Objective

The objective of this test is to determine whether the simulation produces a physically plausible relationship between boom angle and vessel speed under downwind sailing conditions. In a running configuration, where the wind approaches from directly behind the vessel, sailing theory predicts that an optimal boom angle exists at which the forward drive component of the aerodynamic force is maximised. Boom angles that are too small relative to the wind direction result in a sail that is insufficiently open to capture the wind, while angles that are too large cause the sail to stall or present excessive drag without useful forward drive. The test assesses whether the simulation reproduces this behaviour.

5.2.1.2 Test Setup

The test was conducted using an automated test harness implemented in the `SailTestHarness` script, which controls the boom angle programmatically, maintains a fixed vessel heading using a PID autopilot integrated into the `RudderController`, and records vessel speed at each boom angle step. The wind was configured in the `WindManager` with a speed of 7 m/s and a direction set such that the apparent wind angle relative to the vessel was approximately 180° , placing the vessel in a running configuration throughout the test. Wind turbulence was kept at its default value.

The PID autopilot used a pure integral controller with $K_i = 0.02$ and $K_p = K_d = 0$, which maintained the target heading within a window of approximately $\pm 5^\circ$ throughout each run. The target heading was set to 0° and the vessel was reset to zero velocity at the start of each boom angle step to ensure consistent initial conditions.

5.2.1.3 Methodology

The boom angle was swept from -90° to $+90^\circ$ in steps of 10° , producing 19 test configurations. At each step, the vessel was allowed to accelerate from rest for 30 seconds. Speed was sampled only during the final 5 seconds of each run and averaged to produce a single representative speed value for that boom angle, reducing the influence of the initial transient acceleration phase. The average speed in both m/s and knots and the apparent wind angle were recorded for each step and saved to a CSV file.

The recorded apparent wind angle at each step was used as a reference to verify that the downwind running condition was maintained throughout the sweep. In addition, a theoretical optimal boom angle was computed at each step as half the apparent wind angle, derived from the principle that for maximum drive the sail chord should bisect the angle between the apparent wind direction and the vessel's heading [1, 2]. The results are evaluated against both this theoretical prediction and the qualitative guidance of sailing theory, which states that when sailing downwind the sail should be eased out to a position approximately perpendicular to the vessel's centreline in order to maximise the projected area exposed to the wind and capture the maximum aerodynamic force [1, 2]. The optimal boom angle is therefore expected to be in the region of $\pm 80^\circ$ to $\pm 90^\circ$ from the centreline for a pure running configuration, consistent with a theoretical optimal of approximately 90° derived from an apparent wind angle of 180° .

5.2.1.4 Results

Table 5.1 presents the average speed recorded at each boom angle step, and Figure 5.1 illustrates the speed profile across the full sweep.

Table 5.1: Sailing performance test results — average speed at each boom angle under downwind conditions (wind speed 7 m/s, AWA $\approx 180^\circ$).

Boom Angle ($^\circ$)	Avg Speed (m/s)	Avg Speed (knots)	AWA ($^\circ$)	Theoretical Optimal ($^\circ$)
-90	0.944	1.835	180.2	90.1
-80	0.910	1.769	180.7	90.3
-70	0.793	1.541	180.8	90.4
-60	0.745	1.448	179.8	89.9
-50	0.658	1.278	181.0	90.5
-40	0.640	1.244	182.8	91.4
-30	0.555	1.079	182.3	91.2
-20	0.440	0.856	178.8	89.4
-10	0.264	0.512	181.7	90.8
0	0.071	0.138	181.0	90.5
+10	0.196	0.380	182.7	91.3
+20	0.425	0.825	180.4	90.2
+30	0.549	1.066	180.4	90.2
+40	0.597	1.160	181.7	90.8
+50	0.668	1.298	180.5	90.3
+60	0.744	1.447	179.4	89.7
+70	0.815	1.585	182.0	91.0
+80	0.916	1.781	180.6	90.3
+90	0.933	1.814	180.3	90.1

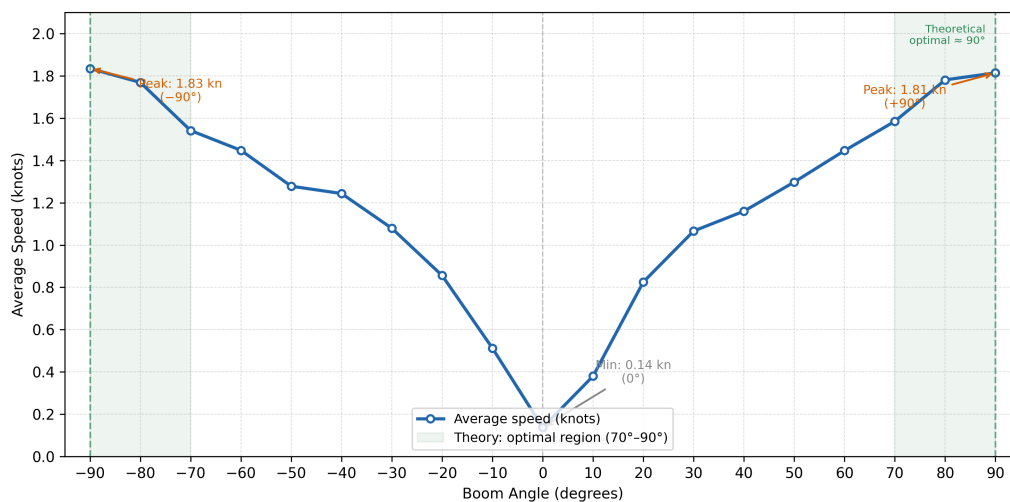


Figure 5.1: Average vessel speed as a function of boom angle under downwind conditions. The green shaded regions indicate the theoretically optimal boom angle range of 70° to 90° predicted by sailing theory. Speed peaks at $\pm 90^\circ$ and reaches a minimum at 0° .

5.2.1.5 Analysis

The results demonstrate a clear and consistent relationship between boom angle and vessel speed that is in strong agreement with the predictions of sailing theory. Speed increases monotonically as the boom is rotated away from the centreline in either direction, reaching a peak of 1.83 knots at -90° and 1.81 knots at $+90^\circ$. The minimum speed of 0.14 knots occurs at 0° , where the sail is aligned with the vessel's centreline and presents minimal projected area to the wind.

The apparent wind angle remained consistently close to 180° throughout the sweep (varying between 178.8° and 182.8°), confirming that the downwind running condition was maintained reliably by the PID autopilot across all boom angle steps. The theoretical optimal boom angle, computed as half the apparent wind angle, is consistently approximately 90° across all steps, in direct agreement with the qualitative prediction of sailing theory that the sail should be eased out to a position perpendicular to the vessel's centreline for maximum drive in a running configuration [1, 2].

The simulation results match this prediction well, the highest speeds are recorded at boom angles of $\pm 80^\circ$ and $\pm 90^\circ$, which fall squarely within the theoretically optimal region. The speed profile is smooth and monotonically increasing with boom angle magnitude on both sides, with no unexpected discontinuities or reversals, suggesting that the aerodynamic and hydrodynamic force models interact consistently and without numerical artefacts across the full range of tested angles.

The symmetric speed profile observed between port ($-$) and starboard ($+$) boom angles is physically expected for a downwind running condition at $AWA \approx 180^\circ$. At this point of sail the wind approaches from directly behind the vessel, making the geometry mirror-symmetric between the two tacks (rotating the boom to $+80^\circ$ or -80° presents the same sail area to the wind on opposite sides). This symmetry is a property specific to the running configuration and would not be expected at other points of sail, where the wind approaches from one side only and useful drive is generated only on the correct tack.

A small residual asymmetry is nonetheless observed, for example 1.83 knots at -90° versus 1.81 knots at $+90^\circ$. This is attributable to the slight offset in the apparent wind angle, which averaged approximately 180.5° rather than exactly 180° , placing the wind marginally more favourably on the port side. This is a consequence of the residual heading error of the PID autopilot rather than a physical asymmetry in the simulation model.

5.2.2 Beam Reach Sailing Test

5.2.2.1 Objective

The objective of this test is to assess whether the simulation produces physically plausible behaviour at a beam reach point of sail, where the apparent wind approaches from the side of the vessel at approximately 90° relative to the bow. Sailing theory predicts that at a beam reach the optimal boom angle is approximately half the apparent wind angle, in this case approximately 45° on the leeward side, where the sail generates the maximum combination of lift and forward drive [1, 2].

5.2.2.2 Test Setup and Methodology

The test was conducted using the same automated test harness and PID autopilot configuration described in Section 5.2.1. The wind was configured to approach from the port side of the vessel at an apparent wind angle of approximately 270° (port beam), with a wind speed of 7 m/s. The same sweep parameters were used: boom angles from -90° to $+90^\circ$ in steps of 10° , with a 30-second run duration and 5-second sampling window at the end of each step.

However, due to the known limitation of the aerodynamic polar curve model, in which angles of attack beyond 90° are folded back symmetrically rather than modelled accurately, the analysis is restricted to the port tack boom angle range of -90° to 0° . On the port tack with wind approaching from the left, negative boom angles place the sail on the leeward side where it can generate useful aerodynamic drive. Positive boom angles would place the sail on the windward side, and in the current model would produce physically incorrect results due to the polar curve symmetry limitation described in Section 6.3.

5.2.2.3 Results

Table 5.2 presents the average speed recorded at each boom angle step for the port tack range, and Figure 5.2 illustrates the speed profile.

Table 5.2: Beam reach sailing test results — average speed at each boom angle, port tack only (wind speed 7 m/s, AWA $\approx 275^\circ$).

Boom Angle ($^\circ$)	Avg Speed (m/s)	Avg Speed (knots)	AWA ($^\circ$)
−90	0.361	0.701	273.1
−80	0.435	0.846	273.5
−70	0.710	1.379	276.5
−60	0.878	1.706	278.1
−50	0.903	1.755	277.7
−40	0.783	1.522	276.2
−30	0.734	1.426	275.9
−20	0.646	1.257	274.8
−10	0.530	1.030	274.5
0	0.453	0.880	274.0

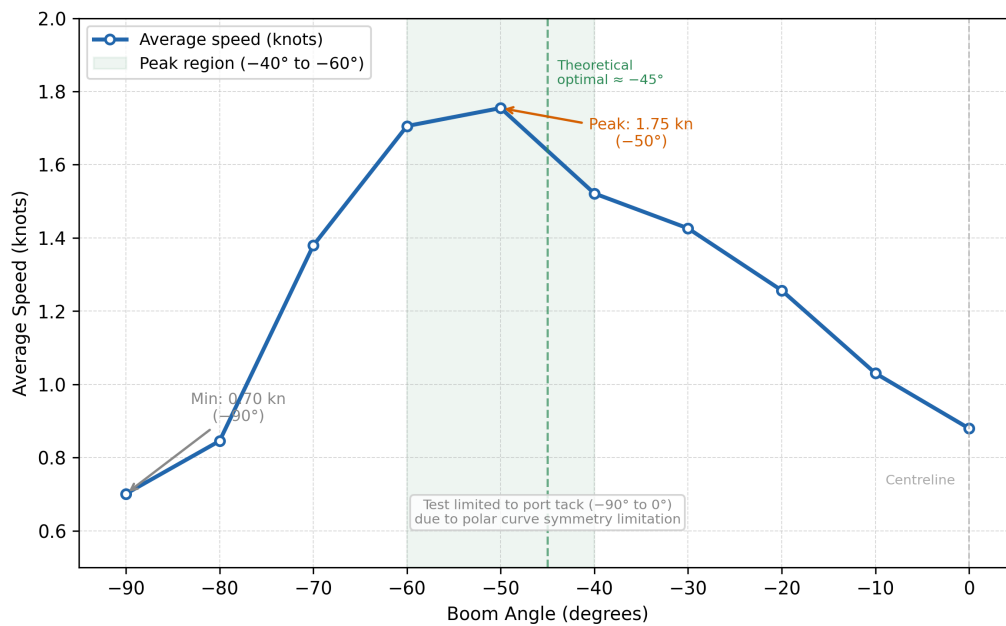


Figure 5.2: Average vessel speed as a function of boom angle for a port tack beam reach (AWA $\approx 275^\circ$). The green shaded region indicates the peak performance range around -50° . The test is limited to the port tack range (-90° to 0°) due to the polar curve symmetry limitation.

5.2.2.4 Analysis

The results show a clear peak in vessel speed at a boom angle of -50° , reaching 1.75 knots, with speed dropping off on either side of this optimum. This behaviour is consistent with the theoretical prediction that at a beam reach the sail should be trimmed to approximately half the apparent wind angle on the leeward side. With an average AWA of approximately 275° , corresponding to an effective beam reach angle of 85° relative to the port side, the theoretical optimal boom angle is approximately -42° to -45° , in good agreement with the observed peak at -50° .

The speed profile shows a clear and physically meaningful shape: speed is low at -90° where the sail is fully extended and stalled, rises to a peak at -50° where lift and forward drive are maximised, and then decreases toward 0° where the sail is insufficiently trimmed relative to the wind direction. This pattern is qualitatively consistent with the expected aerodynamic behaviour of a sail at a beam reach, where the interplay between lift generation and drag minimisation produces a well-defined optimal trim angle [1, 2].

The apparent wind angle varied between 273.1° and 278.1° throughout the test, confirming that the beam reach condition was maintained consistently by the PID autopilot. The small variation in AWA has a negligible effect on the theoretical optimal angle and does not affect the validity of the comparison.

Overall, the beam reach test further supports the conclusion that the simulation produces physically plausible sailing behaviour across different points of sail, with the optimal boom angle correctly identified in the region predicted by sailing theory.

5.3 User Study

5.3.1 Study Design

A task-based expert evaluation with semi-structured interviews was conducted to assess the simulation from the perspective of experienced sailors. The study was designed to address four research questions:

- RQ1 How realistic do experienced sailors perceive the simulated sailboat behaviour to be?
- RQ2 How effectively do the boom and rudder controls support meaningful sailing interaction?
- RQ3 How useful is the real-time feedback UI for understanding sailing dynamics and boat behaviour?
- RQ4 What aspects of the simulation reduce immersion, realism, or learnability?

Table 5.3: Participant demographic profile, including sailing experience, prior exposure to simulations, and technology familiarity.

ID	Age	Gender	Sailing Exp.	Simulation Exp.	Sailing Sim. Exp.	Tech Familiarity
P1	26–35	Female	8–15 years	None	None	Not familiar
P2	36–45	Male	15+ years	Limited	None	Very familiar
P3	36–45	Male	15+ years	Limited	Limited	Moderately familiar
P4	36–45	Female	15+ years	None	None	Very familiar
P5	26–35	Female	4–7 years	None	None	Very familiar

5.3.2 Participants

Five participants were recruited from experienced captains and sailors of the Ten-oared Whaleboat, the specific vessel targeted by the simulation. This selection criterion ensures that participants have direct practical knowledge of the vessel’s sailing characteristics and can provide informed assessments of the simulation’s physical plausibility. Table 5.3 presents participants’ demographic.

The group consists of three females and two males, with ages ranging from 26 to 45. Four of the five participants have 15 or more years of sailing experience, and one has between 4 and 7 years. Prior exposure to simulations is limited across the group, with three participants reporting no previous simulation experience and two reporting limited experience. None of the participants had prior experience with sailing simulators specifically, with the exception of one participant who reported limited experience. Technology familiarity varied, with four participants describing themselves as very or moderately familiar with digital systems and one reporting no familiarity.

5.3.3 Procedure

The study followed a structured three-phase procedure for each participant.

Introduction (2 to 3 minutes). The purpose of the study was briefly explained to the participant, and the demographic questionnaire was completed. Participants were asked to sign a consent form before proceeding.

Tutorial phase (4 to 5 minutes). Each participant was introduced to the simulation controls and interface. They were given time to freely interact with the simulation without any specific objectives, in order to familiarise themselves with the boom and rudder controls and the feedback overlay before the task phase began.

Task phase (10 to 15 minutes). Participants were asked to complete four sailing tasks, presented in randomised order to counterbalance any learning or fatigue effects across participants. The tasks were:

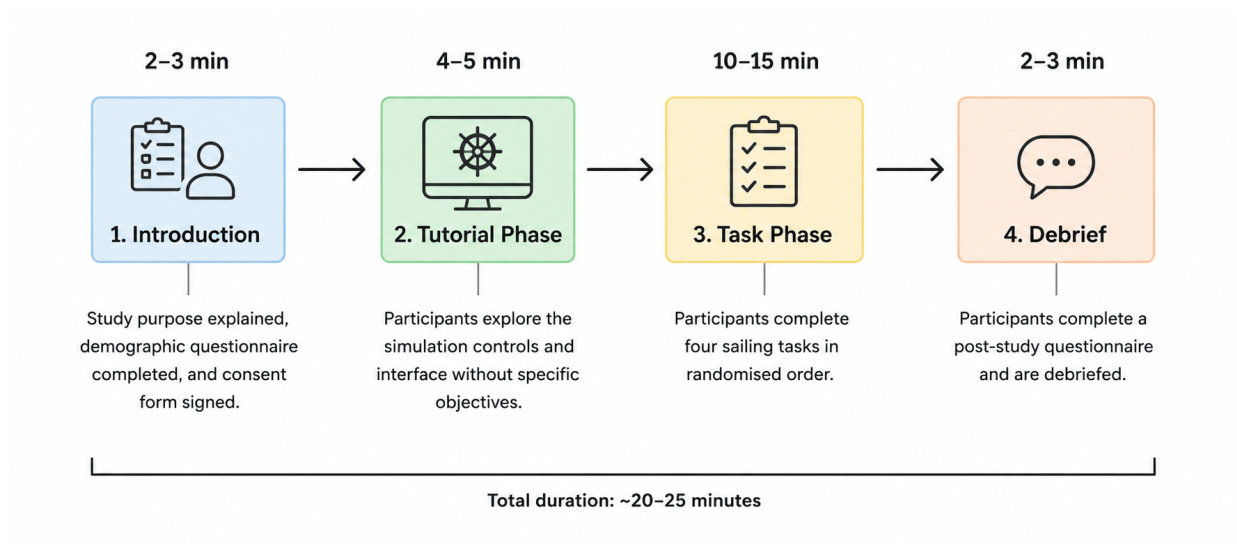


Figure 5.3: Overview of the user study procedure, showing the four sequential phases conducted with each participant: introduction and consent, tutorial and familiarisation with the sailing simulator, task completion, and post-task interview/debrief. Approximate duration for each phase is indicated.

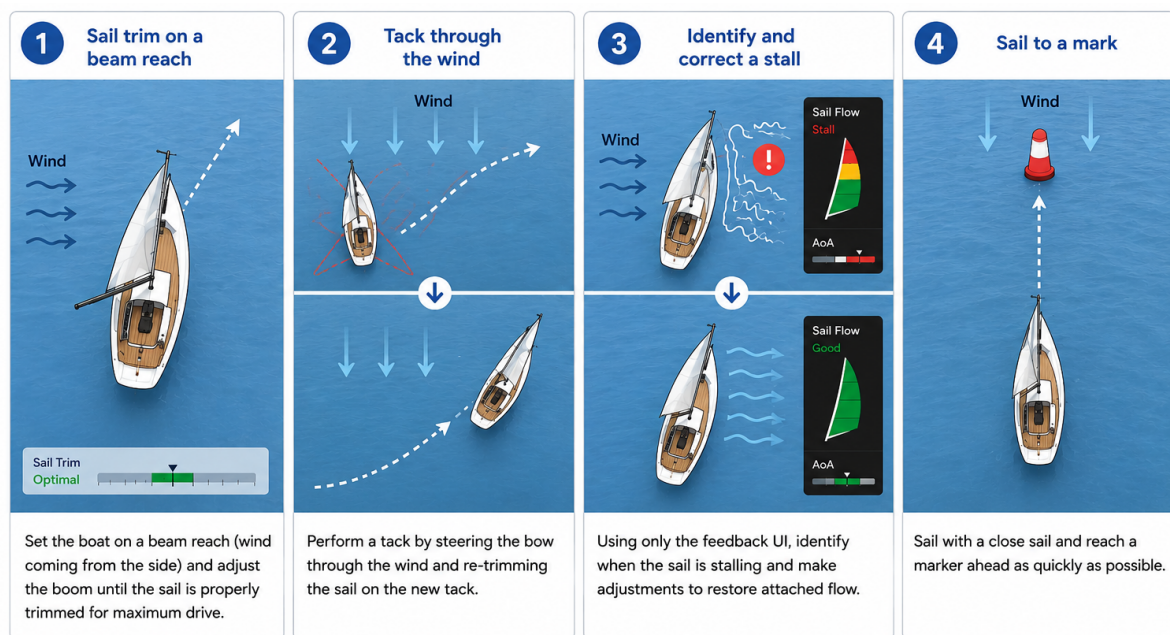


Figure 5.4: The four tasks presented to participants during the task phase, shown in the order they were administered. Tasks were presented in randomised order to counterbalance learning and fatigue effects.

- **Task 1 (Sail trim on a beam reach):** Set the boat on a beam reach (wind coming from the side) and adjust the boom until the sail is properly trimmed for maximum drive.
- **Task 2 (Tack through the wind):** Perform a tack by steering the bow through the wind

and re-trimming the sail on the new tack.

- **Task 3 (Identify and correct a stall):** Using only the feedback UI, identify when the sail is stalling and make adjustments to restore attached flow.
- **Task 4 (Sail to a mark):** Sail with a close sail and reach a marker ahead as quickly as possible.

Following the task phase, a semi-structured interview was conducted covering four thematic areas: realism of the sailing behaviour, intuitiveness of the controls, usefulness of the UI feedback, and comparison with real sailing or other simulators. Participants were also invited to suggest improvements to realism and additional mechanics they would expect.

5.3.4 Data Collection

Data was collected through two complementary instruments. The semi-structured interview provided qualitative data on participant perceptions across the four research question themes. A structured questionnaire collected quantitative ratings on six Likert-scale items (1 = Strongly Disagree, 7 = Strongly Agree):

1. The boat movement felt realistic.
2. The sail behaviour matched my expectations.
3. The controls were intuitive.
4. The simulation helped me understand sailing dynamics.
5. The User Interface (UI) feedback was useful.
6. I would consider this simulation useful for sailing training or educational purposes.

5.3.5 Analysis

5.3.5.1 Quantitative Analysis

Quantitative data from the six Likert-scale items was analysed descriptively. For each item, the mean and standard deviation were computed across the five participants. Results are presented as a bar chart with error bars indicating standard deviation, allowing visual comparison of ratings across evaluation dimensions. Given the small sample size, no inferential statistical tests were conducted.

5.3.5.2 Qualitative Analysis

Semi-structured interview responses were audio-recorded with participant consent and subsequently transcribed using the automated transcription service Rev.com. The resulting transcripts

Table 5.4: Likert-scale results (1 = Strongly Disagree, 7 = Strongly Agree). Mean and standard deviation across five participants.

Item	Mean	Std Dev
The boat movement felt realistic	5.40	0.80
The sail behaviour matched my expectations	5.40	0.49
The controls were intuitive	6.20	0.40
The simulation helped me understand sailing dynamics	5.00	0.63
The UI feedback was useful	6.40	0.80
I would consider this simulation useful for training or education	7.00	0.00

were analysed using thematic analysis following the six-phase framework of Braun and Clarke [35]. The analysis was conducted inductively, allowing themes to emerge from the data rather than being imposed by the research questions.

5.3.6 Results

5.3.6.1 Quantitative Results

Table 5.4 presents the mean and standard deviation for each of the six Likert-scale items across the five participants, and Figure 5.5 illustrates the results as a bar chart.

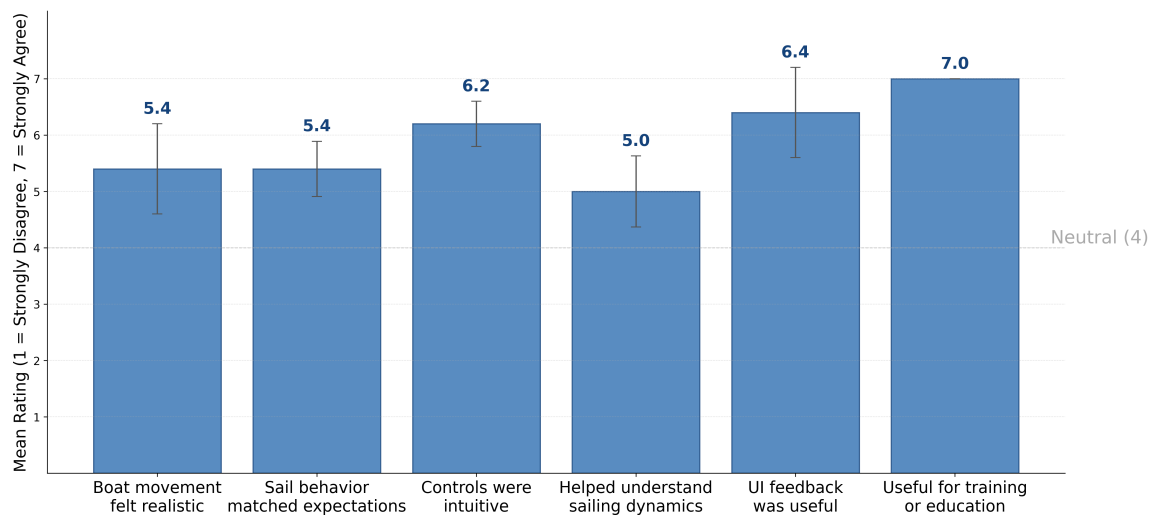


Figure 5.5: Mean Likert-scale ratings across five participants. Error bars indicate one standard deviation. All items scored above the neutral midpoint of 4.

All six items received mean ratings above the neutral midpoint of 4, indicating a generally positive assessment across all evaluation dimensions. The highest-rated item was the usefulness of the simulation for sailing training or educational purposes, which received a unanimous rating of 7 from all five participants (mean 7.00, std 0.00). The UI feedback was also rated highly (mean 6.40, std 0.80), and the intuitiveness of the controls received strong agreement (mean

6.20, std 0.40). Realism-related items received moderate to good ratings, with boat movement realism and sail behaviour both scoring a mean of 5.40. The item regarding the simulation's contribution to understanding sailing dynamics received the lowest mean rating of 5.00, though this remains above the neutral midpoint.

5.3.6.2 Qualitative Results

The thematic analysis revealed 5 themes related to Perceived Realism, Control Naturalness, UI Feedback and Information Design, Training Value, Desired Improvements.

T1: Perceived Realism of Sailing Behaviour. All five participants judged the core sailing dynamics of the simulation to be credible. The relationship between wind angle, sail trim, and boat motion was consistently described as realistic, though participants qualified their endorsements within the constraints of the single-sail configuration relative to the three-sail rig they sail in practice.

“It was quite realistic. It meets my expectations — overall experience, very impressive.” (P2)

“It was very, very realistic, I’d say.” (P5)

The most significant realism failure, raised independently by three participants, was the vessel's behaviour when in irons (head-to-wind), the simulation produces an unrealistic backwards drift with excessive speed accumulation, causing the boat to spin around. P3 diagnosed the root cause as the aerodynamic force model failing to switch to the correct side when the vessel crosses through the no-go zone, causing speed to accumulate additively rather than reset.

“It builds too much speed — it’s like putting a car in reverse and pulling the hand-brake. It spins your boat around because of that.” (P3)

A secondary realism concern was the mismatch between the simulation's symmetric hull form and the broader transom stern of the real Sea Scout dinghy, which was noted as contributing to the unexpected backwards behaviour and reducing contextual authenticity for this user group.

These qualitative observations are consistent with the quantitative ratings, where boat movement realism and sail behaviour received mean scores of 5.40 ($\sigma = 0.80$) and 5.40 ($\sigma = 0.49$) respectively, moderate to good ratings that reflect genuine endorsement tempered by the identified limitations.

T2: Control Naturalness and Expressiveness. The boom and rudder controls were broadly well-received across all five participants, who found them responsive and intuitive. Participants who identified physics-level issues clearly distinguished the input mechanism from the underlying simulation behaviour.

“Very good. Very good. [The controls]” (P3)

“Yes, they worked correctly.” (P5)

The one specific mechanical limitation raised was the rudder’s angular range. P2 estimated the simulation restricts the rudder to approximately 35°, whereas real rudders should reach 60° to 70°, limiting the expressiveness of sail-rudder combinations.

“The rudder should have had a bit more freedom of movement. The more flexible your rudder is, the more manoeuvrable the boat — otherwise the combination of sail handling and rudder handling becomes limited.” (P2)

Requests for additional controls, including a mainsheet and a full three-sail rig, were raised by multiple participants, though these represent requests for expanded mechanics rather than critiques of the existing control model.

This positive reception is reflected in the quantitative data, where the intuitiveness of the controls received the second highest mean rating of 6.20 ($\sigma = 0.40$), indicating strong and consistent agreement across participants.

T3: UI Feedback and Information Design. All five participants engaged with and valued the on-screen feedback. However, a consistent finding was that numeric-only displays impose interpretive overhead, requiring participants to mentally translate numbers into sailing concepts. Wind direction was the most universally flagged issue, with four of five participants requesting a visual directional arrow to replace or supplement the numeric wind angle display.

“I would have preferred seeing the wind direction with an arrow — so I don’t have to interpret every time what 180 degrees means in terms of point of sail.” (P2)

“It’s easier for me to see than to think. If this [the wind angle] were shown as an arrow, it would definitely help me.” (P1)

P3 suggested labelling the current point of sail by name alongside the numeric angle to reduce the translation step, and P1 requested minimal, well-grouped indicators to reduce cognitive load. These findings point toward a need to design the UI for a range of user literacy levels.

“What would be more understandable is to also write the name of the point of sail — write ‘beam reach’ for me.” (P3)

Despite the qualitative concerns about interpretive overhead, the UI feedback item received a mean rating of 6.40 ($\sigma = 0.80$), suggesting that participants valued the information provided even while identifying ways in which its presentation could be improved.

T4: Perceived Training Value. Four of five participants endorsed the simulation as a genuinely valuable training tool, and this theme arose unprompted. The common argument was that the simulation addresses a specific access problem: learners who cannot sail regularly still need to build conceptual and procedural knowledge between sessions, and video or theory cannot replicate the act of doing.

“It’s a very important tool for someone who can’t go out on a boat every day. Because it’s one thing to see it on paper or in videos, and another to do it yourself.”
(P2)

“I would definitely use it — to give people an introduction, so that when they actually get on the boat, they already have some understanding.” (P5)

Participants consistently recommended a staged pedagogical approach: introduce one sail first, validate that model, then progressively add rigging complexity. A VR version was raised by four participants as a desirable future direction, particularly for restoring the physical sensation of heel and wind, which experienced sailors described as instinctive rather than cognitive inputs.

“The feeling of the wind is crucial. After some years, some things become instinctive — the feeling of the wind helps you understand where you’ll actually go.” (P5)

This strong endorsement is directly supported by the quantitative data, the item regarding the simulation’s usefulness for sailing training or educational purposes received a unanimous rating of 7 from all five participants (mean 7.00, $\sigma = 0.00$), the highest and most consistent rating across all Likert items.

T5: Desired Improvements. Participants raised a number of specific improvements, ordered by how many participants requested them. The most broadly requested change was a visual wind direction arrow (4 of 5 participants), to replace the numeric wind angle display with a dynamic arrow in the helmsman’s reference frame. The most critical physics fix identified was the force-transition behaviour in irons (3 of 5 participants). Additional requests included point-of-sail labels alongside numeric angles (2 of 5), a colour-coded heel indicator showing remaining margin before capsize (2 of 5), an extended rudder angular range to approximately 70° (1 of 5), visible sail flutter at stall (1 of 5), a corrected hull shape reflecting the broader transom stern of the real vessel (1 of 5), and a navigable environment with visible waypoints and track recording (1 of 5). A VR version with wind and heel haptics was expressed as a longer-term direction by four participants.

5.4 Summary

This chapter presented the technical testing and user evaluation of the simulation. The downwind sailing test demonstrated that the simulation produces a physically plausible and monotonically increasing relationship between boom angle and vessel speed under running conditions, with peak performance correctly identified at boom angles of $\pm 80^\circ$ to $\pm 90^\circ$, consistent with the predictions of sailing theory. The beam reach test confirmed that a well-defined optimal trim angle exists at approximately -50° on the port tack, in good agreement with the theoretical prediction of approximately -45° derived from the apparent wind angle.

The user study, conducted with five experienced Ten-oared Whaleboat sailors, produced broadly positive results across all evaluation dimensions. All six Likert-scale items received mean ratings above the neutral midpoint, with the simulation's usefulness for training and education receiving a unanimous rating of 7 from all participants. The qualitative analysis identified five themes, perceived realism of sailing behaviour, control naturalness, UI feedback and information design, training value, and desired improvements. Core sailing dynamics were judged credible by all participants, with the controls rated as responsive and intuitive. The primary limitation identified was the vessel's unrealistic behaviour when in irons, attributed to the aerodynamic force model failing to switch sides when crossing through the no-go zone. UI feedback was valued but participants consistently requested a visual wind direction arrow and point-of-sail labels to reduce interpretive overhead. Training value was strongly affirmed, with participants recommending a staged one-sail approach before expanding to the full three-sail rig.

6 Conclusion and Future Work

6.1 Summary of Work

This thesis has presented the design and implementation of a real-time interactive sailing simulation of the Ten-oared Whaleboat, developed in Unity 6.3. The work was motivated by the absence of any existing simulator targeting this specific vessel type and the pedagogical needs of sea scout training programmes in Cyprus and the broader Mediterranean region.

The simulation was built around three interconnected pillars. The first is a physically-informed hydrodynamic model that computes buoyancy, pressure drag, friction drag, and lateral resistance on a per-triangle basis from a dynamically generated underwater mesh, allowing the vessel's response to waves and motion to emerge from the actual geometry of the hull. The second is an aerodynamic model that derives lift and drag forces from camber-dependent polar curves interpolated in real time from the deformation of a cloth-simulated sail, incorporating an induced drag term based on the finite-span characteristics of the sail geometry. The third is a user interaction layer through which the user controls the vessel via boom rotation and rudder deflection, with a real-time debug overlay and compass indicator providing immediate feedback on the simulation state.

The system was developed with a deliberate balance between physical accuracy and real-time performance. Simplifications were made where full implementation would have introduced disproportionate complexity relative to the educational objectives of the simulation, and these are clearly identified throughout the thesis and in the future work section below.

6.2 Evaluation Against Objectives

The primary objective of this work was to develop a real-time interactive sailing simulation that produces physically plausible vessel behaviour in response to user control inputs and environmental conditions. The following objectives were set out in Chapter 1 and are evaluated here:

- **Physically-informed hydrodynamic model** — achieved. Buoyancy, pressure drag, friction drag, lateral resistance, and rudder hydrodynamics are all implemented and operate on a per-triangle basis from a dynamically generated underwater mesh.
- **Physically-informed aerodynamic model** — achieved. Lift, profile drag, and induced drag are computed from camber-dependent polar curves sampled in real time from the cloth sail deformation, and applied at the center of effort.

- **Real-time interactive control** — achieved. The user can adjust boom angle and rudder deflection at runtime and observe immediate physical responses in the vessel’s motion.
- **Vessel-specific modelling** — achieved. The simulation is built around the geometry and characteristics of the Ten-oared Whaleboat, with mass, center of mass, sail area, and aspect ratio configured to approximate the real vessel.
- **Validation against sailing theory** — achieved. Controlled sailing tests confirmed physically plausible behaviour at two points of sail, with optimal boom angles matching theoretical predictions in both cases. Expert evaluation by five experienced Ten-oared Whaleboat sailors rated all simulation dimensions above the neutral midpoint, with training value receiving a unanimous score of 7. The primary limitation identified was unrealistic vessel behaviour when in irons, identified as a priority fix for future development.

6.3 Limitations

Several limitations of the current implementation should be acknowledged.

Hydrodynamic model simplifications. The friction drag model uses a constant friction coefficient rather than the ITTC-57 correlation line, which would express the coefficient as a function of the Reynolds number. Froude residuary resistance, which accounts for wave-making effects and becomes significant at higher vessel speeds, was not implemented. Added mass, which affects the vessel’s response to acceleration, was also not included. These simplifications mean that the resistance model is most accurate at moderate speeds and less reliable at the extremes of the vessel’s operating range.

Single sail configuration. The real Ten-oared Whaleboat uses a three-sail rigging system comprising a jib, mainsail, and mizzen. In this simulation, a single sail configuration is deliberately adopted. The primary objective of the simulation is to provide a clear and focused environment for novice sailors to develop a foundational understanding of sail trimming and vessel response. Introducing multiple interacting sails at this stage would add significant complexity to both the aerodynamic model and the user interaction, potentially obscuring the fundamental principles that the simulation is designed to teach. The extension to a full three-sail configuration is identified as a natural next step once the single-sail model has been validated and the user interaction layer has been developed.

Sheet controller. The sheet trimming system, which was designed to give the user direct control over sail camber by simulating the easing and hardening of the mainsheet, was developed through four successive approaches but ultimately deactivated due to a fundamental limitation of Unity’s cloth constraint system. The `maxDistance` constraint, which defines the maximum distance each vertex can move from its rest position in bone-local space, does not operate in

world space. As a result, rotating the boom displaces the clew vertex rest positions away from the sheet anchor, causing the constraint to pull the clew toward the wrong point rather than toward the anchor. A full description of the development process and the identified limitation is provided in Section 4.5.2. As a result, sail camber varies passively in response to apparent wind acceleration and gravity acting on the cloth, and the polar curve blending system responds to these passive camber changes, but direct user control over sheet tension is not available in the current implementation.

Hull aerodynamic drag. The aerodynamic drag acting on the above-waterline surfaces of the hull, rigging, mast, and crew was not modelled. This component can be significant in strong wind conditions and its absence means the simulation overestimates the net driving force available from the sail. To partially compensate for this, a configurable force scale factor is applied to the total aerodynamic sail force, allowing the effective force magnitude to be tuned downward to approximate the net effect of the missing resistance components. This is an empirical correction rather than a physically-based model, and the implementation of a proper hull aerodynamic drag model is identified as future work.

In-irons behaviour. When the vessel crosses through the no-go zone (head-to-wind), the simulation produces an unrealistic backwards drift with excessive speed accumulation, causing the boat to spin around rather than stalling and losing way as would occur in practice. This behaviour was independently identified by three of five expert participants during the user study as the primary realism failure of the simulation. The root cause has not been definitively identified and likely involves a combination of factors in the aerodynamic force model, including the handling of angles of attack beyond 90° and the force direction when the apparent wind crosses the sail from the wrong side. A systematic investigation of the force transition behaviour in the no-go zone is identified as the highest-priority fix for future development.

Crew weight and distribution. The mass of the crew is not modelled. The vessel is treated as a single rigid body with a fixed mass and center of mass, which does not account for the effect of crew movement on trim and stability.

Human-computer interaction layer. The full VR-based immersive interface described in Chapter 3 was not implemented. The current interface is a desktop simulation with a debug overlay, which is suitable for development and testing but does not provide the immersive experience intended for training purposes.

6.4 Future Work

The limitations identified above directly motivate the following directions for future development:

ITTC-57 friction and Froude resistance. Replacing the constant friction coefficient with the ITTC-57 correlation line and adding a Froude-based residuary resistance component would improve the accuracy of the resistance model across the full speed range of the vessel.

Added mass. Implementing added mass would improve the accuracy of the vessel's dynamic response to acceleration and deceleration, particularly in unsteady wind conditions.

Multi-sail configuration. Extending the aerodynamic model to include the jib and mizzen sails, along with their interaction effects, would produce a more complete and realistic representation of the Ten-oared Whaleboat under sail.

Sheet controller. The sheet controller requires a world-space position constraint solution that enforces a maximum distance between the clew vertex and the sheet anchor transform at each timestep, independent of bone rest positions. This could be achieved either through a custom cloth solver extension or by replacing the Unity cloth constraint system with a position-based dynamics approach for the clew vertex specifically. Resolving this would restore direct user control over sail camber, completing the intended sail trim interaction model.

Hull aerodynamic drag. Adding a bluff-body aerodynamic drag model for the above-waterline hull surfaces would improve the accuracy of the net force balance, particularly in strong wind conditions.

In-irons force transition. A systematic investigation of the aerodynamic force model behaviour when the vessel crosses through the no-go zone is required to resolve the unrealistic speed accumulation and spin-out identified during expert evaluation. This should examine the handling of angles of attack beyond 90° , the force direction switching mechanism when the apparent wind crosses the sail from the wrong side, and whether extending the polar curves to cover the full $[0^\circ, 180^\circ]$ range would produce physically correct behaviour in this regime.

User interface improvements. The user study identified several concrete directions for improving the feedback layer of the simulation. The most broadly requested change, raised by four of five participants, is the addition of a visual wind direction arrow displayed in the helmsman's reference frame, replacing the current numeric wind angle display which imposes an interpretive overhead for users unfamiliar with degree conventions. Participants also requested point-of-sail labels alongside numeric angles (for example displaying "Beam Reach" next to the angle value), a colour-coded heel indicator showing remaining margin before capsize, and visible sail flutter when the sail is at or beyond stall. P1 additionally highlighted the need for a minimal, spa-

tially grouped display layout to reduce cognitive load for users with reading difficulties. These improvements are identified as near-term development priorities ahead of the full HCI layer described below.

VR human-computer interaction layer. The most significant longer-term direction for future development is the implementation of the full immersive interface described in Chapter 3. This would include a VR environment in which the learner is physically present on the vessel, direct interaction with sail and rudder controls, navigational instruments, acoustic feedback, and a physical motion platform for kinesthetic feedback. The educator dashboard for runtime scenario control would also be developed as part of this layer. Four of five expert participants expressed strong interest in a VR version, specifically for restoring the physical sensation of heel and wind, which experienced sailors described as instinctive rather than cognitive inputs.

BIBLIOGRAPHY

- [1] E. I. Lada, *To εγχειρίδιο του αρμενιστή*, 1st ed. Ιδρυμα Ευγενίδου, 2012.
- [2] P. G. Strouza, *Ιστιοπλοΐα και Ναυτική Τέχνη*, 4th ed. Ιδρυμα Ευγενίδου, 2010.
- [3] D. Colgate, “What are the points of sail?” <https://www.offshoresailing.com/sailing-basics-understanding-the-points-of-sail/>, Seaworthy Academy’s Offshore Sailing School, 2024.
- [4] “Points of sail,” <https://www.mallorcasailing.com/articles/points-of-sail>, Mallorca Sailing Academy, 2026.
- [5] J. Feldman, “Buoyancy,” Course Notes, 2003.
- [6] P. S. Şerban, “Numerical simulation of shallow waters effects on sailing ship ”mircea” hull,” *Scientific Bulletin of Naval Academy*, vol. 19, 2016.
- [7] A. A. Vekinis and S. Perantonis, “Aeolus ocean – a simulation environment for the autonomous colreg-compliant navigation of unmanned surface vehicles using deep reinforcement learning and maritime object detection,” 2023.
- [8] J. Kerner, “Water interaction model for boats in video games,” <https://www.gamedeveloper.com/programming/water-interaction-model-for-boats-in-video-games>, 2015.
- [9] J. J. Slooff, *The Aero- and Hydromechanics of Keel Yachts*. Springer International Publishing, 2015.
- [10] A. F. Molland, S. R. Turnock, and D. A. Hudson, *Ship Resistance and Propulsion*. Cambridge University Press, 2011.
- [11] P. Cieślak, “Stonefish: An advanced open-source simulation tool designed for marine robotics, with a ros interface,” in *OCEANS 2019 - Marseille*, 2019, pp. 1–6.
- [12] F. M. White, *Fluid Mechanics*. McGraw-Hill, 2011.
- [13] P. Izaguirre Alza, “Numerical and experimental studies of sail aerodynamics,” PhD Thesis, Universidad Politécnica de Madrid, 2012.
- [14] F. Fossati, *Aero-hydrodynamics and the performance of sailing yachts*. Adlard Coles Nautical, 2009.
- [15] A. Z. Tleubergenova, N. K. Tanasheva, K. M. Shaimerdenova, S. S. Kassymov, A. R. Bakhtybekova, N. N. Shuyushbayeva, S. Z. Uzbergenova, and G. A. Ranova, “Mathematical modeling of the aerodynamic coefficients of a sail blade,” *Advances in Aerodynamics*, vol. 5, p. 14, 2023.

- [16] J. Zhang, G. Bertrand, M. Rabaud, B. Augier, and M. Fermigier, “Flying shape and aerodynamics of a full-scale flexible olympic windsurf sail,” 2025. [Online]. Available: <https://arxiv.org/abs/2501.13254>
- [17] X. Z. Wenfeng Hu, Zhouqing Qu, “A new approach of mechanics simulation based on game engine,” in *Fifth International Joint Conference on Computational Sciences and Optimization*, 2012.
- [18] Unity Technologies, “Unity manual: Physics,” 2025. [Online]. Available: <https://docs.unity3d.com/6000.3/Documentation/Manual/PhysicsSection.html>
- [19] —, “Unity manual: Cloth,” 2025. [Online]. Available: <https://docs.unity3d.com/6000.3/Documentation/Manual/class-Cloth.html>
- [20] P. Robin, A. Leroyer, D. de Prémorel, and J. Wackers, “Tackling modern sailing challenges with a cfd-based dynamic vpp,” *Journal of Sailing Technology*, 2024. [Online]. Available: <https://doi.org/10.5957/jst/2024.9.1.1>
- [21] T. Salafi, “Smartsail mobile: a system for novice sailors using mobile computing,” 11 2014.
- [22] Unknown, “Boat hull 3d model,” Sketchfab, 2025, original source model, retrieved November 2025. Author details unavailable.
- [23] Unity Technologies, “Skinned mesh renderer — unity manual,” <https://docs.unity3d.com/6000.3/Documentation/Manual/class-SkinnedMeshRenderer.html>, 2025, accessed: May 2026.
- [24] —, “Water system simulation — high definition rp,” <https://docs.unity3d.com/Packages/com.unity.render-pipelines.high-definition@14.0/manual/WaterSystem-simulation.html>, 2025, accessed: May 2026.
- [25] Wave Harmonic, “Crest ocean system — unity asset store,” <https://assetstore.unity.com/packages/tools/particles-effects/crest-ocean-system-hdrp-164158>, 2024, accessed: May 2026.
- [26] Unity Technologies, “Water surface properties — high definition rp,” <https://docs.unity3d.com/Packages/com.unity.render-pipelines.high-definition@14.0/manual/WaterSystem-Properties.html#scriptinteractions>, 2025.
- [27] —, “Exclude part of a water surface — high definition render pipeline,” <https://docs.unity3d.com/Packages/com.unity.render-pipelines.high-definition@17.3/manual/water-exclude-part-of-the-water-surface.html>, 2026, accessed: May 2026.
- [28] K. Perlin, “An image synthesizer,” in *Proceedings of the 12th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*, 1985, pp. 287–296.

- [29] Unity Technologies, “Watersurface.fillwatersearchdata — high definition render pipeline api,” <https://docs.unity3d.com/Packages/com.unity.render-pipelines.high-definition@17.3/api/UnityEngine.Rendering.HighDefinition.WaterSurface.html>, 2026.
- [30] —, “Watersearchresult — high definition rp api,” <https://docs.unity3d.com/Packages/com.unity.render-pipelines.high-definition@14.0/api/UnityEngine.Rendering.HighDefinition.WaterSearchResult.html>, 2025.
- [31] —, “Input system — unity manual,” <https://docs.unity3d.com/Packages/com.unity.inputsystem@1.19/manual/index.html>, 2025.
- [32] —, “Clothskinningcoefficient — unity scripting api,” <https://docs.unity3d.com/6000.3/Documentation/ScriptReference/ClothSkinningCoefficient.html>, 2025.
- [33] —, “Monobehaviour.ongui — unity scripting api,” <https://docs.unity3d.com/6000.3/Documentation/ScriptReference/MonoBehaviour.OnGUI.html>, 2025.
- [34] —, “Trail renderer — unity manual,” <https://docs.unity3d.com/6000.3/Documentation/Manual/class-TrailRenderer.html>, 2025, accessed: May 2026.
- [35] V. Clarke and V. Braun, “Thematic analysis,” *The journal of positive psychology*, vol. 12, no. 3, pp. 297–298, 2017.