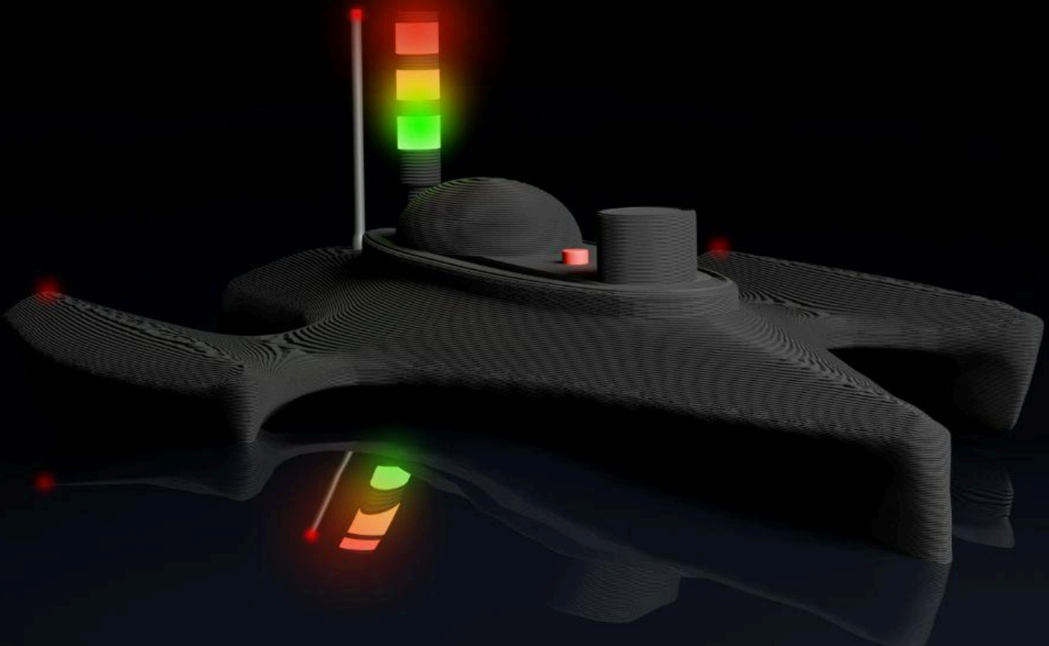
NODE

AUTONOMOUS SHIP

Progress Report 01|03|2025

Njord 2025
Autonomous Ship Challenge



IAAC ENGINIRING CLUB

Institute for advanced architecture of Catalonia BARCELONA SPAIN

Women in Aerospace

Progress Report for NJORD Challenge 2025

NODE Team

IAAC Institute of Advanced Architecture of Catalonia

Index

Explanation

NODE is a multidisciplinary student team composed of architects, designers and engineers, united by a shared passion for exploring robotic engineering, boat design, sustainable architecture, and digital fabrication. Based in Catalonia and formed at the Institute for Advanced Architecture of Catalonia (IAAC), our team brings a unique perspective shaped by an architectural background, stepping beyond our primary field to engage with engineering and marine autonomy challenges. As first-time participants in the Njord 2025 Physical Autonomous Challenge, we are eager to design and build an autonomous surface vessel (ASV) while exchanging knowledge and sharing experiences with other teams.

Our ASV features a custom-designed 3D-printed hull, leveraging additive manufacturing to enhance hydrodynamics and structural efficiency. The NODE team focuses on integrating advanced navigation, real-time sensor fusion, and robotic control to develop an efficient and adaptive ASV. While our foundation is in architecture, we embrace the challenge of marine robotics by continuously researching, testing, and refining our approach.

The following paper summarizes the main technical aspects of NODE’s ASV.

1.0 Vessel Design

1.1 Hull Design

Due to the absence of a provided hull design from the competition organizers—despite prior discussions—the **NODE team developed a custom hull design** optimized for stability, buoyancy, and modular fabrication.

NODE's **L1.0 x W0.7 x H0.3m trimaran** features a **1m x 0.3m central hull**, housing most hardware components, and **two 0.5m x 0.15m side hulls** that enhance stability and hydrodynamic efficiency. The hardware is **partially enclosed within the main hull**, while externally mounted components are strategically positioned for accessibility and functionality.

1.2 Fabrication Strategy

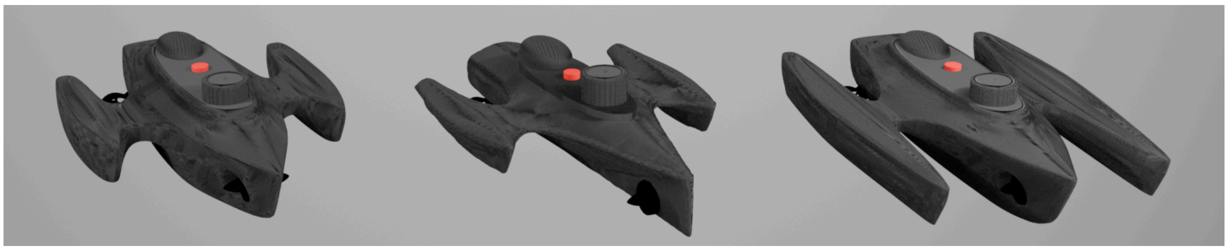
The hull is **3D printed in modular sections** using **PETG**, selected for its **high strength, UV resistance, and water resistance**, ensuring durability in marine environments. Given **facility constraints**, the boat is printed in multiple parts, with a design optimized for **fast and precise assembly**. Multiple **Bambu Lab 3D printers** are utilized to print sections simultaneously, which are later **assembled into a single vessel**.

1.4 Key Design Strategies

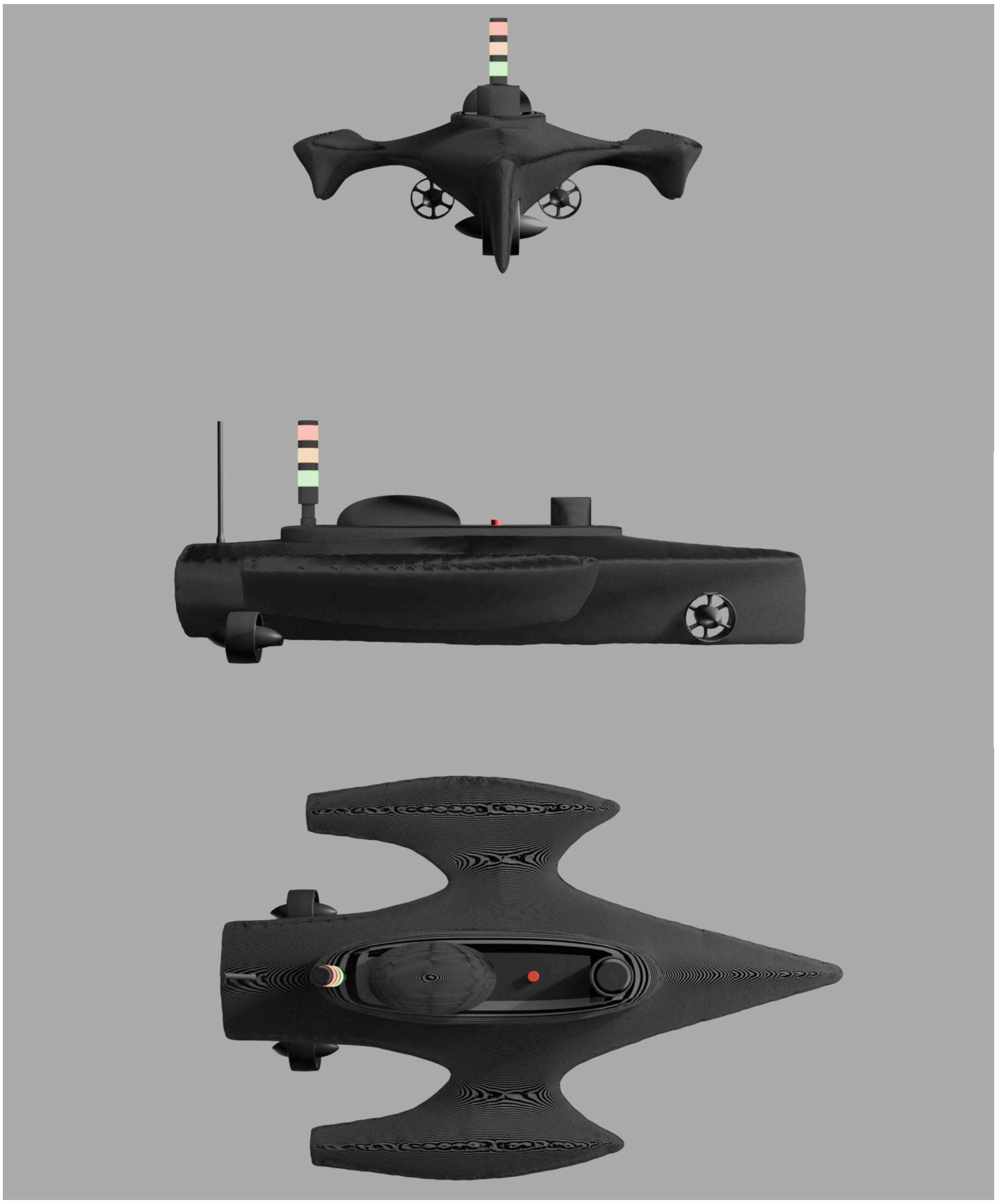
- **Watertight Seams** → Dovetail joints or overlapping edges for improved sealing.
- **Epoxy Reinforcement** → Marine epoxy coating on seams ensures full waterproofing.
- **Hollow Sections for Buoyancy** → Integrated air pockets enhance flotation.
- **Walls (~7mm+)** → Improves structural integrity and prevents leaks, maintaining light weight of the structure.

The design follows **competition regulations**, prioritizing **safety, reliability, and performance** while ensuring a structurally sound and watertight vessel.

Design Exploration:

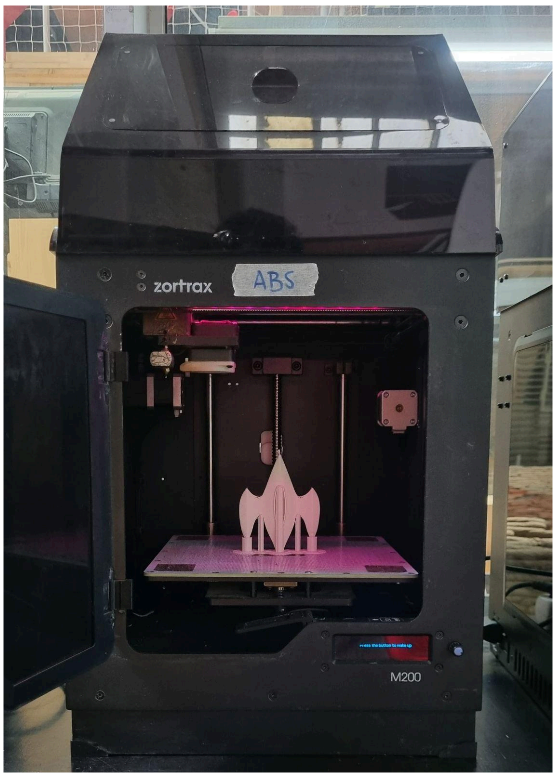


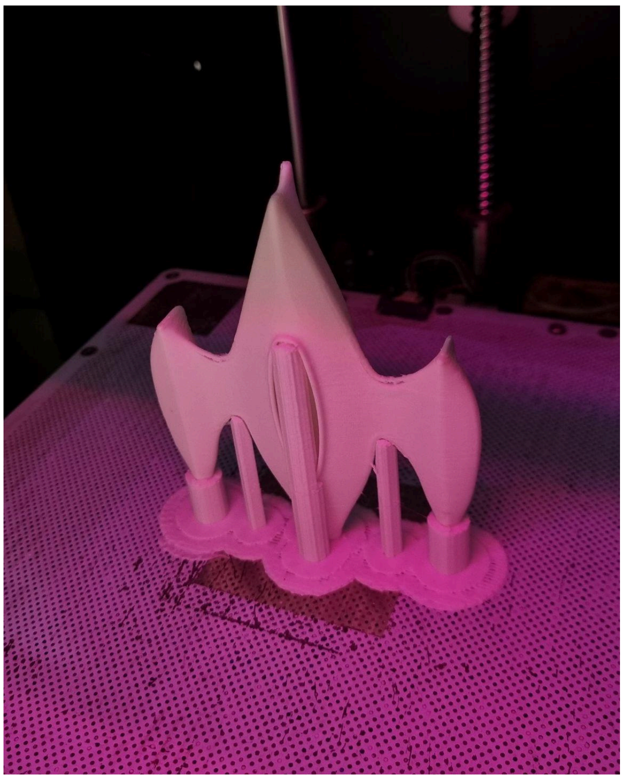
Final Design:

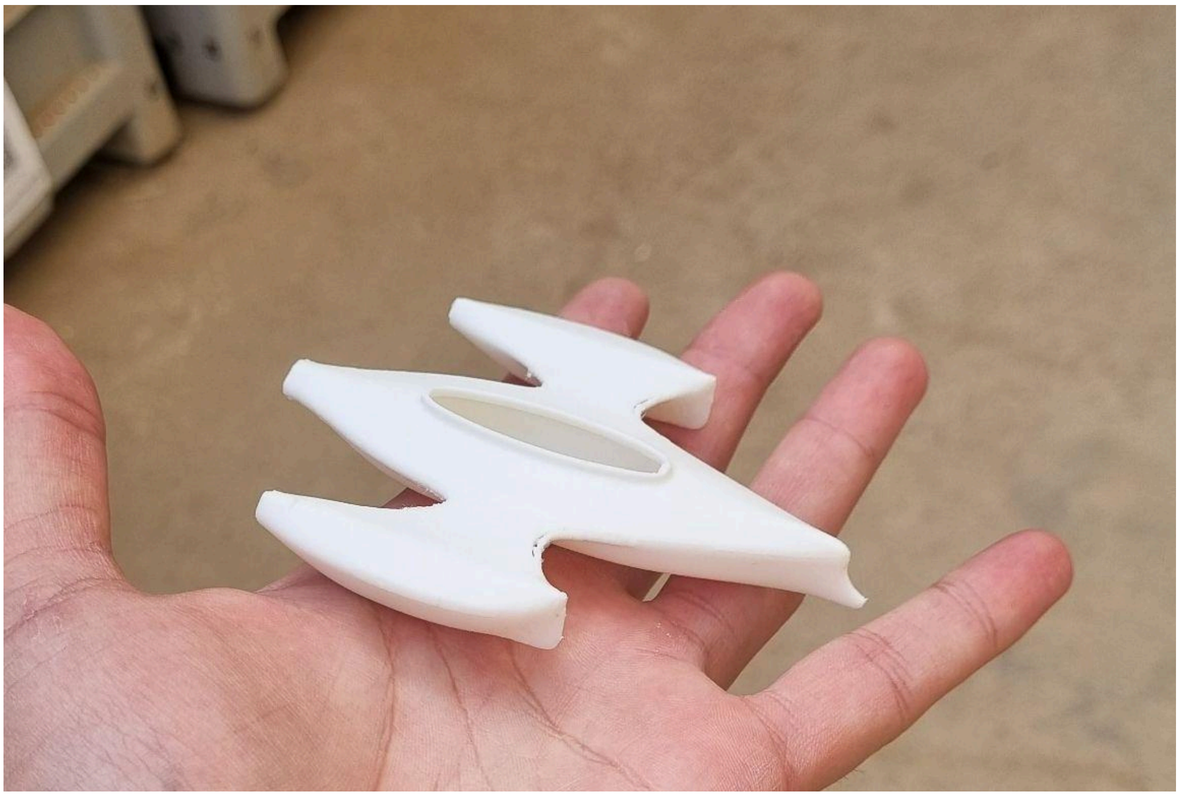


1.5 Testing

Fabrication







ml 1400

2.0 Hardware

2.1 Propulsion

- **Motors**

For the **propulsion system** of our boat, we have chosen motors that will be placed directly **in the water**. There will be a total of **three motors**: **two positioned** at the rear to propel the boat **forward and steer** by adjusting their power output. Additionally, a **third motor** will be placed at the **front center of the boat**, **perpendicular** to the direction of movement, allowing for sharper and faster rotations. That third motor helps us maintain speed while turning, but we also see it as useful for keeping the boat stationary, preventing it from drifting out of its designated position.

These motors are from the brand **APISQUEEN**, specifically the "**U2 MINI 1.3KG UNDERWATER THRUSTER 16V 130W**" model. We chose these motors for their affordability and the necessary power to propel the boat at a suitable speed.

APISQUEEN U2 MINI 1.3Kg Underwater Thruster 16V 130W

Each motor operates within a **voltage range of 12V to 16V**, with a maximum current draw of **8A**. At full thrust, each motor provides **1.3 kg of thrust**. Based on reviews and considering our boat's hydrodynamic shape, we believe this **thrust will be sufficient.**

We have also ensured that these motors are suitable for use in **saltwater.**

- **ESC (Electronic Speed Controller)**

Since the motors have a maximum current draw of **8A at full thrust**, we plan to use **20A ESCs** from **APISQUEEN** to go with the motors. We prefer to use the ESCs from the same company as the motors to ensure perfect compatibility and maximize the chances of avoiding issues.

2.2 Batteries

For **power supply**, we will use a **4S 8000mAh LiPo battery**, which should be sufficient to run all the motors, the Raspberry Pi 5 and the other sensors. However, if the Raspberry Pi or any of the sensors consume more power than expected or if additional electronic components are required for boat control, we may need an additional battery.

To ensure proper charging and battery health, we use a **compatible charger** that allows us to monitor the battery's charge state. This will help us maintain its **longevity** and check whether its **capacity is sufficient** over time.

2.3 Sensors

For this project, we have selected several **essential sensors** to enable **autonomous navigation**. We prioritized **affordable options** to stay within budget, and some sensors were provided for free by members of our school. The sensors include:

- **LIDAR**: Livox Mid-360 - For obstacle detection and environmental mapping, the Robotics lab at our university can provide us with this LIDAR.
- **GPS Module**: We will use the Holybro M9N since it works out of the box with the PixHawk, it is also fairly accurate and affordable.
- **Accelerometer**: The PixHawk 4 already comes with an MCU that helps us understand the movements and positioning of our vessel.
- **2.4Ghz Dipole Antenna**: For receiving the signal from the controller in manual mode.
- **Ultrasonic Sensor**: Essential for underwater obstacle detection and water depth measurements, we will use the DFRobot underwater ultrasonic sensor with a range of 6m, since it has very complete documentation and is affordable.
- **Cameras**: For Computer Vision based object detection, for now we plan to use two regular webcams we have at the Faculty. One will be in the front and one in the back.

2.4 Hardware Implementation

Apart from the **sensors**, several other hardware components are crucial for our project:

- **Raspberry Pi 5**: It supports Python and can run ROS2, which is essential for our software implementation. We chose it because we have one at the faculty, but if it cannot provide the required computing power that our software needs, we will have to have the option to fall back on an Intel NUC or a Jetson Nano X.
- **PixHawk 4:** There is a lot of Documentation on making PixHawk 4 work together with the Raspberry Pi 5. The PixHawk 4 is a well tested Flight Controller and provides us with a good Plug-and-Play starting point to build our electronic circuits around.
- **Signal Light Tower**: For visual status indication.
- **Emergency Stop Button:** To shut off the vessel completely in case of an emergency
- **Waterproof Box**: One for the battery and another for the electronic components.
- **Fireproof Bag**: For additional safety inside the Battery Box.
- **Remote Control**: To control the boat manually, we will use a Taranis QX7.
- **16V to 5V Step-Down Converter**: To power components requiring lower voltage.
- **BMS:** We are planning to use a FrSky telemetry BMS alongside a temperature sensor to monitor the battery for safety purposes.

2.5 Hardware Diagram and Locations

Inside the hull

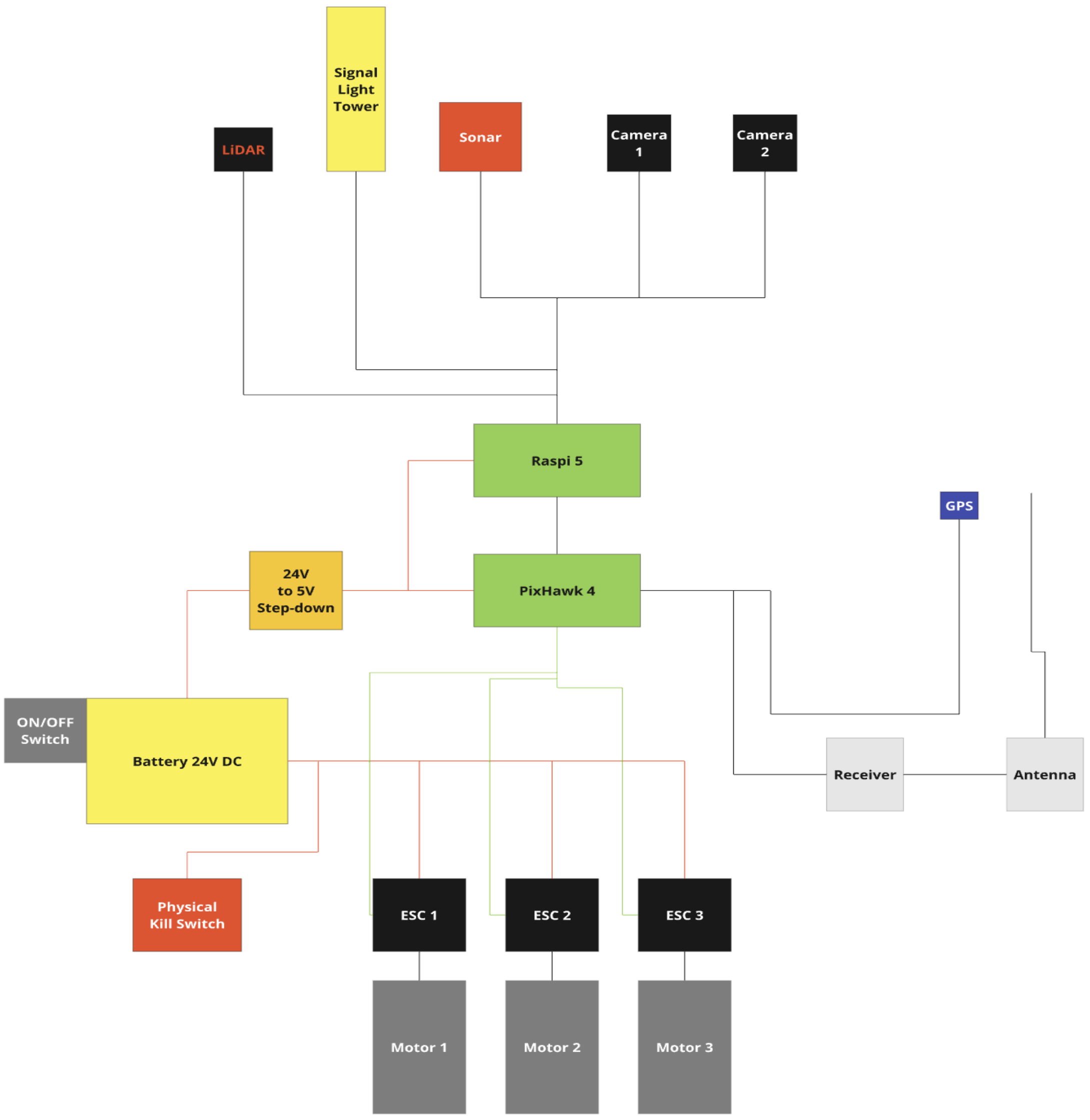
The Battery Box will contain the 4S LiPo Battery inside a fire-retardant bag. From there, we will connect the PixHawk4 and the Raspberry Pi 5, which are located inside the waterproof component box through a 16V to 5V Step-down to provide a stable power supply for our sensors and on-board computers. The ESCs as well as the receiver will end up in the component box.

Upper part

On the top of the boat, we will place the Antennas and the GPS naturally for reception. The Lidar will also have to go on top at the center of the boat and we will try to place it as close to water level as possible.
The two webcams will also find their homes on the top of the boat, one in the front, one in the back.
Safety features like the Killswitch and the Signal Light Tower will also be placed on top.

Below part

On the bottom of our boat we will place the motors and the sonar, facing straight down.



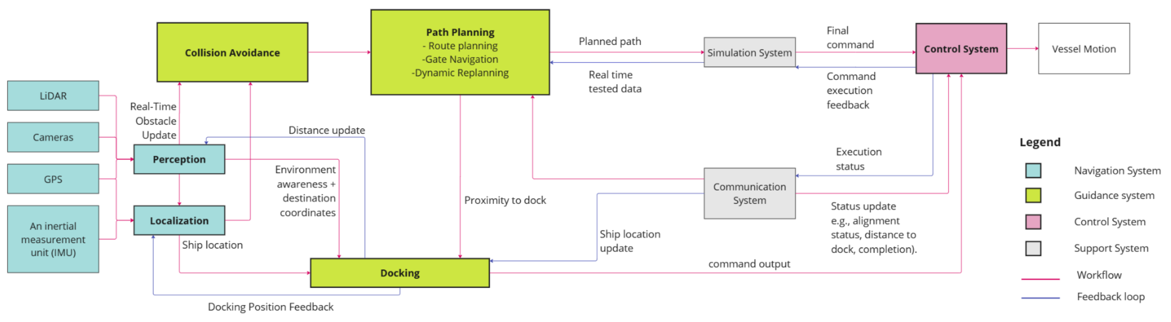
3.0 Software

3.1 Software Design

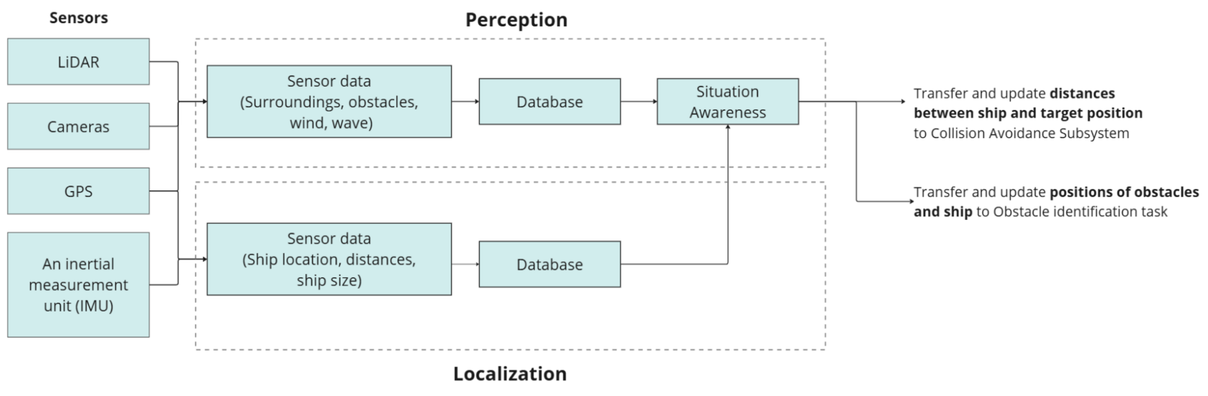
Languages used: Python, ROS

Autonomous surface vessels (ASVs) rely on precise navigation, real-time sensor data processing, and efficient control mechanisms to operate independently. To achieve these functions, a structured framework is required. The Robot Operating System (ROS 1/2) serves as the middleware for managing communication between different components, while Python with rospy provides a flexible programming environment for implementing control algorithms.

3.2 Software Architecture

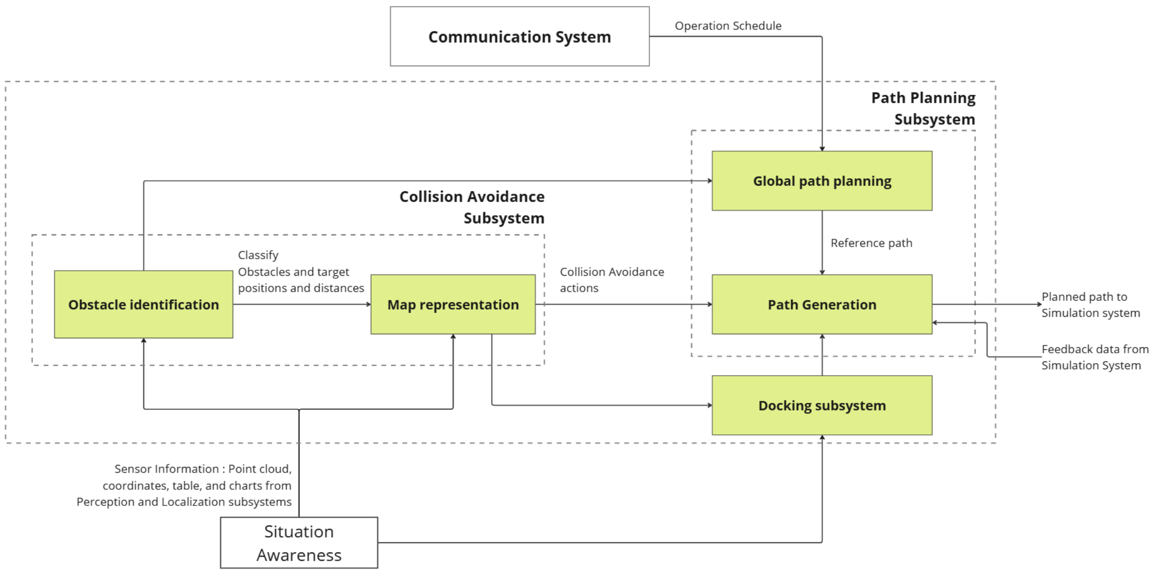


- Navigation System

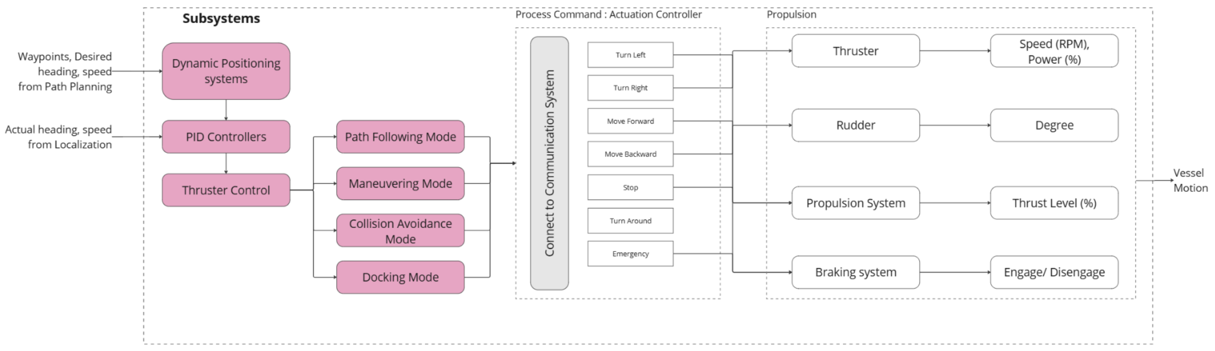


Sensors
LiDAR
Cameras
GPS
An inertial measurement unit (IMU)
Perception
Sensor data (Surroundings, obstacles, wind, wave)
Database
Situation Awareness
Sensor data (Ship location, distances, ship size)
Database
Localization
Transfer and update distances between ship and target position to Collision Avoidance Subsystem
Transfer and update positions of obstacles and ship to Obstacle identification task

- Guidance System


Communication System
Operation Schedule
Path Planning Subsystem
Global path planning
Collision Avoidance Subsystem
Reference path
Classify Obstacles and target positions and distances
Collision Avoidance actions
Obstacle identification
Map representation
Path Generation
Planned path to Simulation system
Feedback data from Simulation System
Docking subsystem
Sensor Information : Point cloud, coordinates, table, and charts from Perception and Localization subsystems
Situation Awareness

- Control System



3.2.1 ROS as a Middleware

ROS Kinetic is an open-source framework designed for robotic systems, offering a modular structure that simplifies system development. It provides essential tools for autonomous boat operation, including:

Node-Based Communication: The roscore service manages the communication between multiple processes running on the system, ensuring efficient data exchange between navigation, sensor processing, and motor control nodes.

Localization with robot_localization: The robot_loc alization package utilizes an Extended Kalman Filter (EKF) to combine sensor data from GPS and an Inertial Measurement Unit (IMU), improving the accuracy of position estimation.

Autonomous Navigation Using move_base: The move_base package processes waypoints and obstacle data to compute an optimal path while adjusting velocity commands for safe travel.

Simulation and Testing with Gazebo: The Gazebo simulation environment provides a physics-based platform to test control algorithms and sensor configurations before deployment in real-world conditions.

3.2.2 System Control

Python, through the rospy library, is widely used for writing ROS nodes due to its simplicity and compatibility with robotics applications. Key areas where Python enhances ASV functionality include:

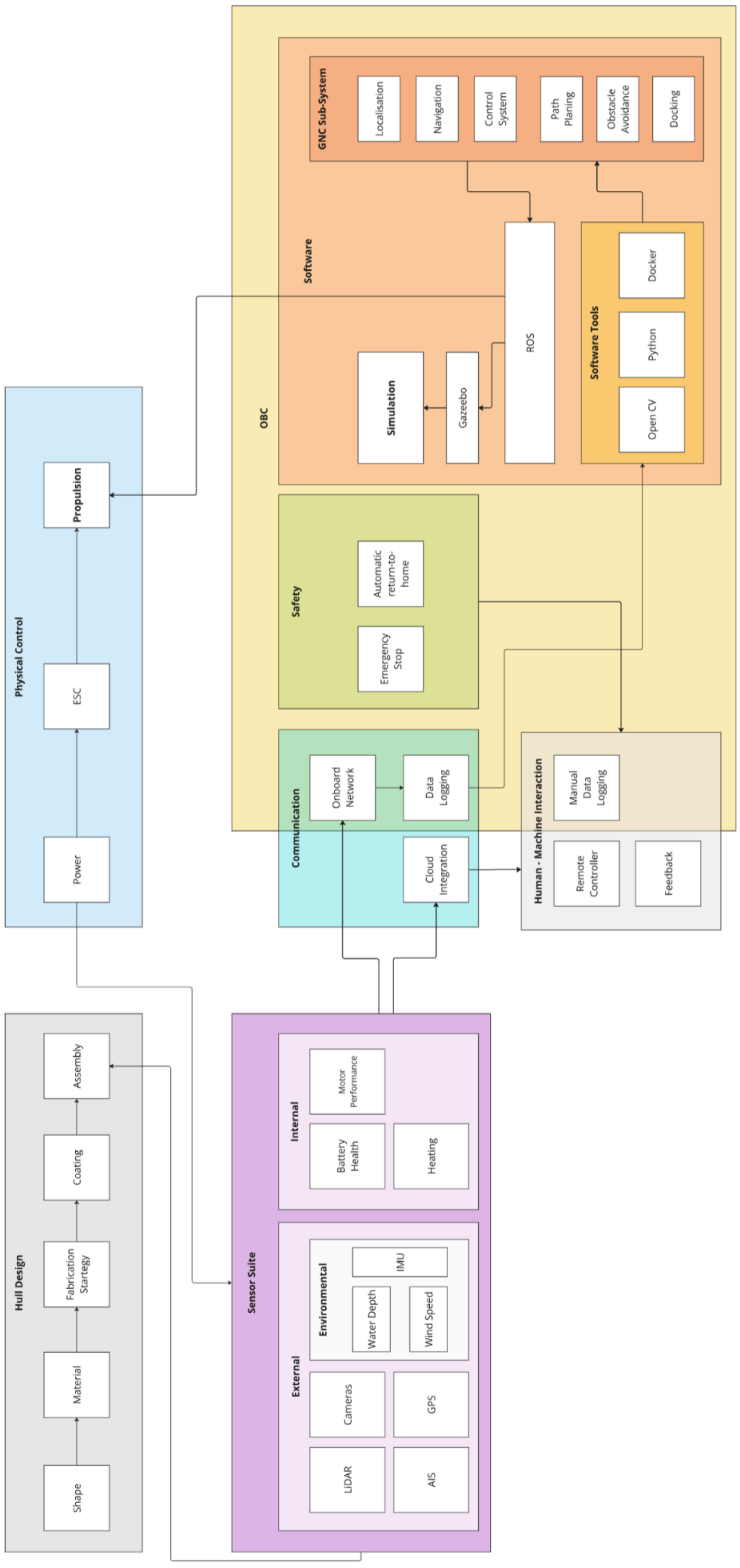
Command Transmission with rospy.Publisher: The rospy.Publisher class is used to send velocity and steering commands in the form of geometry_msgs/Twist messages, allowing real-time control of the boat's propulsion system.

Sensor Data Processing with rospy.Subscriber: The rospy.Subscriber class receives incoming data from GPS and IMU sensors using the sensor_msgs/NavSatFix message type, enabling continuous monitoring of the vessel's position.

Object Detection with OpenCV: The cv_bridge package connects ROS image topics with OpenCV, enabling real-time processing of camera feeds for detecting obstacles in the boat's path.

System Monitoring with rospy.loginfo: The rospy.loginfo() function provides a logging system to track sensor readings, command outputs, and error messages, which helps in debugging and optimizing performance

4.0 System Architecture



5.0 Funding

The *Engineering Club* has been a student-led initiative of students across the different MSc programmes offered by the Institute for Advanced Architecture of Catalonia (IAAC). IAAC has allowed the Engineering Club to freely use the facilities and resources. This includes space for the team to develop the project, access to fabrication tools (3D printers, laser scanners, CNC machines, woodworking workshop), and borrowing of hardware (cameras, lidar sensors, electronics tools, batteries), and material for the hull fabrication. Additionally, IAAC's Robotics Lab has donated 500 EUR (~6000 NOK) for purchasing the project-specific hardware.

This initiative was born out of the urge of IAAC's students, primarily coming from an architectural background, to get hands-on experience through the development of an engineering project and delve deeper into STEM. Marita Georganta, mentor of the Engineering Club and Project Coordinator of Women In Aerospace - Europe, has achieved a collaboration between Engineering Club and Women In Aerospace -Europe and a donation of 350 EUR for additional expenses.

Foreseeing the costs of the competition, including the shipping of materials, participation, and other fees, we are in discussions with Catalan entities to cover these costs.

6.0 Contributions

Team

Alexandra Kraeva - Team Leader

Mert Özkılıç - System Architect

Maximilian Severin Becht - Hardware Engineer

Auxence Daillen - Mechanical Engineer

Siddharth Aiyappa Nambiar - Hardware Architect

Arun Prasad - Software Engineer

Chanakan Chormai - Software Engineer

Geetham Pasumarty - Hardware/Software Integration

Yuvraj Ajay Shirke - Computational Designer + Fabrication

Acknowledgements

Marita Georganta, IAAC Robotic Sensing Expert, mentor and coordinator

Huanyu Li, IAAC Robotics Researcher, software mentor

Pit Siebenaler, IAAC Research Assistant, documentation

Alex Dubor, IAAC Robotics Lab Director, mentor and logistics

We are looking forward to hearing from you!