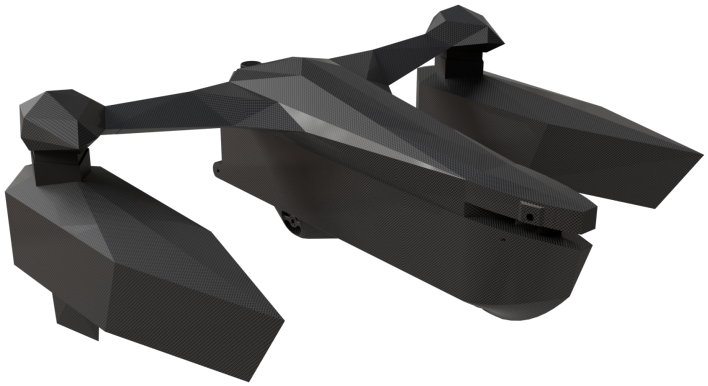
NJORD Challenge 2026
Technical Report

Team Ligmax, NTNU Department of Electronic Systems

June 2026



Ligmax: a roll- and pitch-stabilised autonomous trimaran (CAD render).

---

1 Introduction

Team Ligmax is a multidisciplinary group of students from the bachelor's and master's programmes at the NTNU Department of Electronic Systems (Elsys). The team has a strong track record of delivering large-scale technical projects together, and the NJORD Challenge is an ideal fit: a local competition that gives us a concrete engineering goal, a realistic platform to test our product, and the opportunity to learn alongside international student teams.

Our ASV is a roll- and pitch-stabilised, low-poly trimaran built around a stabilised sensor package. The central hull (*vaka*) carries the propulsion, power and compute, while two outriggers (*amas*) provide a wide, stable stance. The vessel actively counteracts wave motion using a sliding battery for pitch compensation and linear actuators for roll, keeping the sensor platform level for cleaner perception data and easier planing. Sensing is provided by two 360° lidars (one gimbal-stabilised, one rear-facing), two wide-angle cameras, sonar, GPS/compass and an IMU. Visual data processing runs on an NVIDIA Jetson Orin Nano Super, with a Raspberry Pi 5 controlling the remaining subsystems. A Holybro Pixhawk 6C flight controller handles real-time low-level control and communicates over MAVLink with the Pi 5. The Pixhawk, motor controllers and the battery management system share a CAN bus for real-time two-way communication.

This report follows the required content areas: a complete system overview (hull, propulsion, power, sensors, computing/communications and safety), our software approach, the operator GUI, our major design choices, and our headline innovation. Testing is woven into the relevant sections rather than isolated, and full-size diagrams and the complete component list are provided in the appendix.

2 System Overview

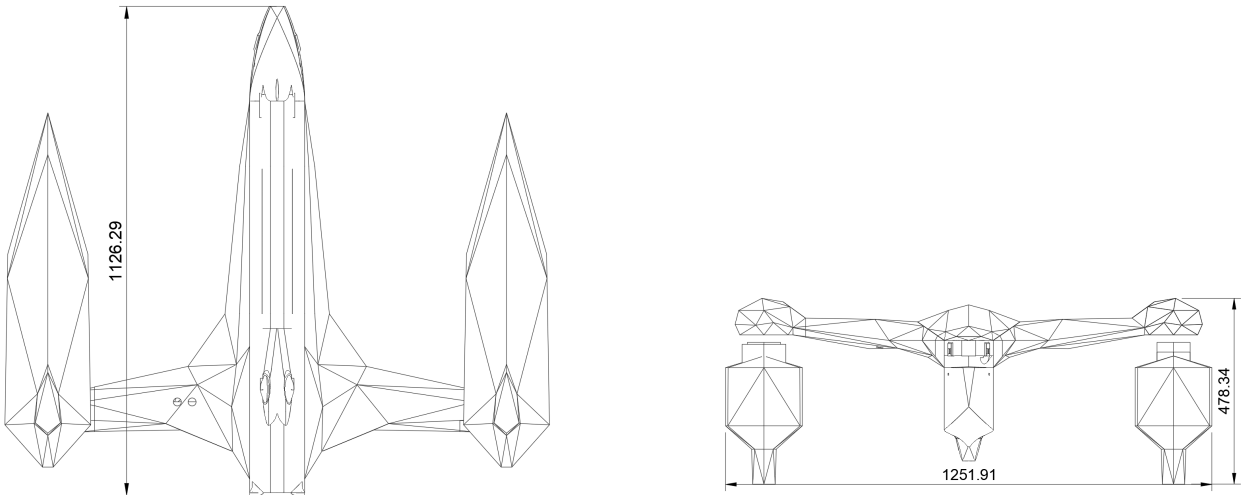
This section walks through every subsystem of the ASV and, just as importantly, how the subsystems connect to one another. Two diagrams anchor the overview: Figure 2 maps the hardware and its communication links, and the software

architecture is shown later in Figure 5. Together they account for every component and every interface on the vessel.

2.1 Hull

We selected a **low-poly trimaran**: a slender central hull (*vaka*) flanked by two outriggers (*amas*) on aluminium cross-beams (*akas*). The narrow central hull gives a low drag coefficient and planes easily, while roughly 60 % of the total buoyancy sits in the wide amas. A centred, low centre of mass combined with a wide buoyancy base produces strong passive roll stability, which our active system then augments (Section 6).

The shell is **3D-printed PLA** skinned and reinforced with **carbon-fibre cloth and epoxy** for a light, stiff, waterproof structure. Structural loads between the hulls are carried by a 2040 aluminium profile, and steel linear rails in the main hull, which both allow the battery to slide for trim (Section 6) and add structure to the long main hull. Principal dimensions, taken from the CAD model, are approximately **1126 mm (L) × 1252 mm (beam, ama to ama) × 478 mm (H)**; all measurements in millimetres.



(a) Top view **(b)** Front view

Figure 1: Hull sketches with principal dimensions (mm).

2.2 Propulsion

Propulsion is made possible through a **multi-propeller with differential thrust** design. Two **Flipsky 5085 140 KV (2500 W)** brushless outrunners drive the main thrusters, and steering is achieved by differential thrust between them, which removes the need for a rudder. A third motor, an **F2838 350 KV 24 V** unit, acts as a lateral **stern thruster** for precise station keeping and docking. The two main thrusters are driven by **VESC 6.7 (70 A, field-oriented control)** controllers on the vehicle CAN bus. The two main thrusters provide roughly 5 kW of combined output power. The stern motor is driven by a 30 A ESC over PWM.

sized for the docking, navigation and speed tasks of the course. The main thrusters and ESCs have been bench-tested directly from the companion computer's CAN interface.

2.3 Power

The vessel runs on a **12S12P Samsung INR18650-35E lithium-ion pack** (≈44.4 V nominal, ≈**1.8 kWh**). A **Daly 150 A BMS** with CAN telemetry manages the pack and reports state of charge, cell voltages and temperature to the operator. The pack is fully enclosed in a composite box combining a TIG-welded aluminium and steel frame with a carbon-fibre shell, which serves three roles at once: fireproof enclosure, rigid structural member, and the moving mass of the pitch-trim system.

Compute and sensors are fed through separate high- and low-voltage DC-DC buck converters. Because the pack is by far the heaviest component, it is mounted low and on the centreline for stability, and on linear rails so its position can be actively used for trim (Section 6). The pack is assembled and behaves as intended in testing.

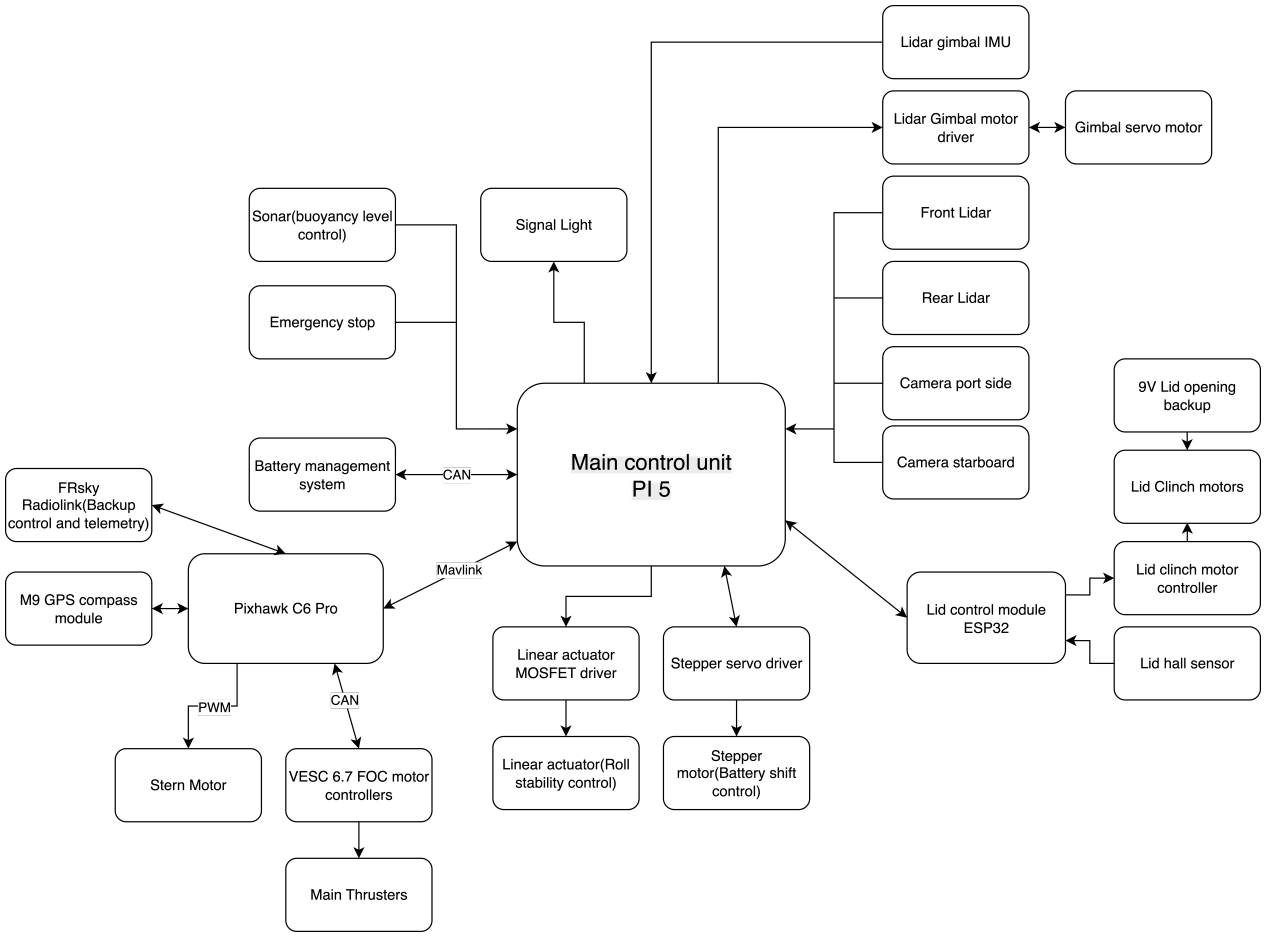


Figure 2: Labelled hardware and communications overview. The Raspberry Pi 5 is the central hub; CAN links the BMS and VESC controllers, MAVLink links the Pixhawk, and an ESP32 manages the locking lid.

2.4 Sensors

Sensors were chosen to give full situational awareness at low weight and power draw:

- **2× Slamtec RPLIDAR (360° 2D lidar)**, one front and one rear, give continuous range data around the entire vessel for obstacle and buoy detection. They are light, low-power and inexpensive, and the front unit is gimbal-stabilised (Figure 4) so its scan plane stays horizontal despite pitch and roll, which keeps detections clean.
- **2× 220° wide-angle cameras** (≈190° usable) provide colour, which the lidars cannot. Course tasks are colour-dependent (for example red and green channel buoys), so the cameras classify the colour of objects the front lidar detects. The very wide field of view means few cameras cover the whole forward arc.
- **Sonar** provides water-level and buoyancy feedback and doubles as an ingress indicator.
- **Quectel LC29HEA-based GPS/compass** supplies global position and heading for waypoint and line-of-sight navigation.
- **IMU.** The Pixhawk's redundant IMUs provide vehicle attitude for control, and a dedicated IMU on the front lidar drives the gimbal stabiliser.

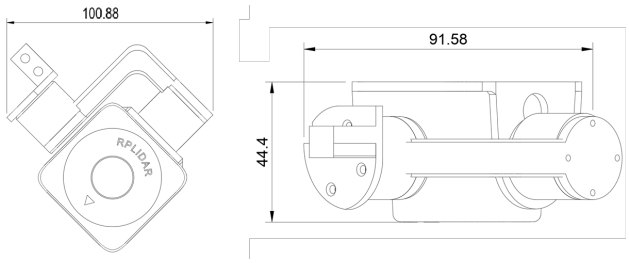
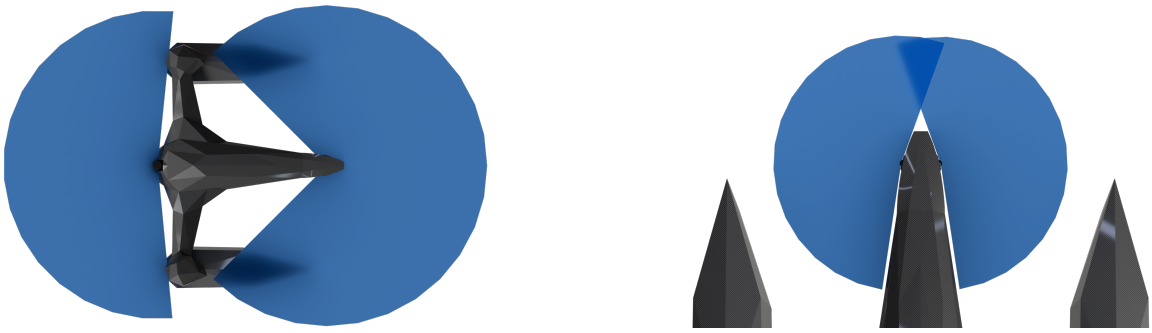


Figure 4: Front gimbal technical drawing



(a) 360° lidar coverage **(b)** Usable camera field of view

Figure 3: Sensor coverage: the two lidars cover every bearing, and the wide cameras cover the forward arc for colour classification.

The complete sensor chain has been integrated and bench-tested. The companion computer communicates with every sensor simultaneously, and live lidar scans and camera-based buoy detection both work (Figures 6a and 6b).

2.5 Computing and communications

Computing is distributed across three primary modules: two high-level companion computers and a real-time flight controller:

- **Vision computer: NVIDIA Jetson Orin Nano Super**. This module is dedicated entirely to visual computation, processing camera feeds, and running the primary perception models.
- **Companion computer: Raspberry Pi 5** (Broadcom BCM2712, quad-core Arm Cortex-A76 at 2.4 GHz, VideoCore VII GPU). Located in the rear of the vessel, it manages path planning, coordinates all general hardware subsystems, runs the operator/telemetry interface, and connects to the vehicle CAN bus through a CAN HAT.
- **Flight controller: Holybro Pixhawk 6C** (STM32H7, redundant IMUs). It executes real-time low-level control—including PID attitude and heading control, and thruster mixing—GNSS fusion, and manages the emergency backup control systems.

The interfaces between subsystems are deliberately simple and standard. **Internally**, a **CAN** bus links the Pixhawk, the VESC 6.7 ESCs and the Daly BMS; the companion computer and Pixhawk exchange commands and telemetry over **MAVLink**; the stern motor is driven over PWM; and an **ESP32** manages the locking lid. **Externally**, a **5G** router carries the primary low-latency operator link (a UDP hole-punched direct connection, described in Section 4), with an independent **FRSky long-range RC** link as a manual and telemetry fallback if 5G is lost. Every subsystem therefore has one defined path to the control core, and Figure 2 shows the complete set of links.

2.6 Safety

Safety follows the Team Handbook requirements for the firebox (§7.3.2) and waterproofing (§7.3.3):

- **Firebox (§7.3.2):** The battery is enclosed in a fireproof box whose frame and the mounting and structural parts of the battery weight-shift assembly are made of aluminium and 3 mm steel, with a carbon-fibre outer shell. A Daly 150 A BMS provides over-current, over- and under-voltage, over-temperature and short-circuit protection with live CAN telemetry displayed in the GUI.
- **Waterproofing (§7.3.3):** the central hull is sealed by a powered locking lid (ESP32 controller and clinch motors). A Hall sensor confirms the lid is closed before the vessel can arm, and a 9 V independent backup circuit can open the lid for recovery even if the main system is down. Sonar and water sensors flag any water ingress, put the vessel

in return-to-shore mode, and start three pumps that expel water from the hulls to keep the vessel operational even after water ingress.

- **Emergency stop and status:** a physical E-stop on the hull cuts power to propulsion via a physical contactor connected to a safety loop system. The loop runs through a smaller relay controlled by the main compute unit and through the physical E-stop in series, so both must be closed for the main contactors to close; if either opens, propulsion power is cut. A status signal light indicates the disarmed, manual, autonomous and fault states.

3 Software

3.1 Architecture, languages and packages

The software is split across the two compute units of Section 2. The **companion computer (Raspberry Pi 5)** runs the high-level stack in **Python**: sensor drivers, perception, the buoy/obstacle world model, the path planner, the active-stabilisation loop and the operator interface. The **Pixhawk 6C** runs open-source autopilot firmware for hard-real-time low-level control. The two halves communicate over **MAVLink**, while actuators and the BMS sit on a **CAN** bus. Open-source building blocks include **OpenCV** for image processing, **NumPy/SciPy** for the lidar clustering and geometry, the **MAVLink** stack for vehicle communication, and a compact CNN for buoy detection. Figure 5 shows how the modules interact.

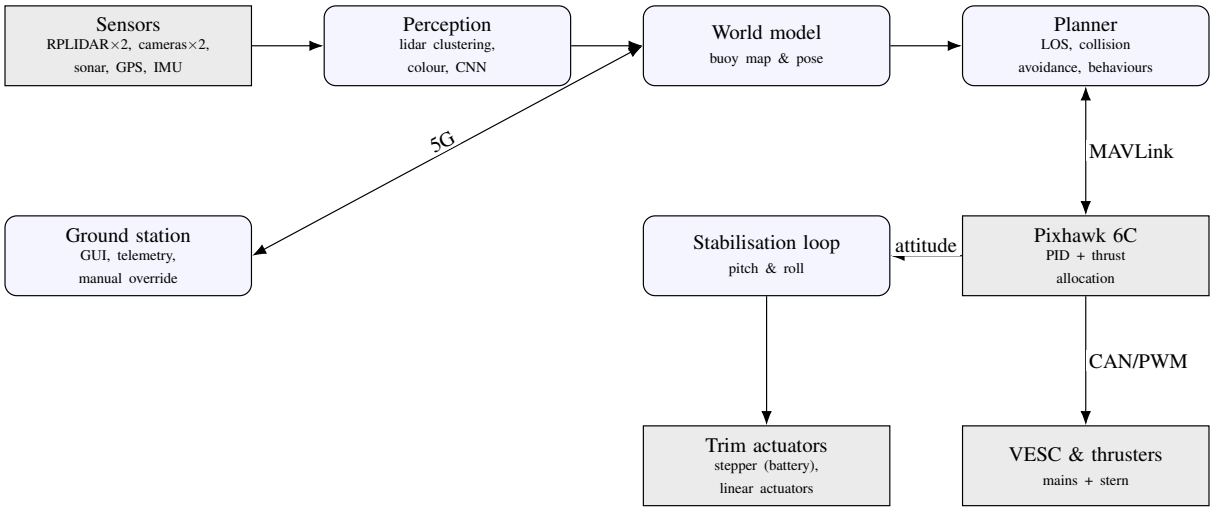


Figure 5: Software architecture. Perception feeds a world model and planner on the Pi; setpoints go to the Pixhawk over MAVLink; the Pixhawk drives the thrusters and feeds attitude to the stabilisation loop. The ground station connects over the 5G link.

3.2 Low-level control

Low-level control runs on the Pixhawk. PID loops regulate heading and speed, and a **thrust-allocation/mixing** stage maps the desired surge and yaw onto the two main thrusters (differential thrust) and maps lateral/station-keeping demands onto the stern thruster. Motor commands are issued to the VESC 6.7 controllers over CAN, where field-oriented control delivers smooth, efficient torque. The autopilot's hardware failsafes (link loss, low battery, arming checks) run here, independent of the higher-level software.

3.3 High-level control and autonomy

On the companion computer, perception turns raw sensor data into a world model, and the planner turns that into setpoints:

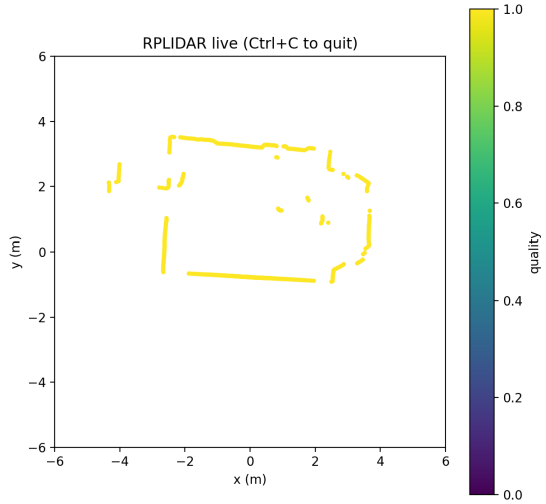
- **Lidar (geometry):** a deterministic **Euclidean clustering** algorithm groups 2D scan returns into candidate objects, giving each a range, bearing and size. Being deterministic, it is fast and fully explainable, which matters for a safety-critical vehicle.
- **Camera (colour):** a non-neural **probabilistic colour classifier** labels the clustered objects (e.g. red vs. green channel markers) by projecting each lidar object into the camera frame and sampling its colour.
- **CNN (cross-check):** a compact convolutional network detects buoys directly in the image as a fallback and cross-reference, improving robustness in difficult lighting (Figure 6b).

- **Planning:** the planner performs **line-of-sight (LOS)** waypoint following with reactive **collision avoidance** around detected objects, plus task-specific behaviours (navigation channel, station keeping, docking). Setpoints are sent to the Pixhawk over MAVLink.

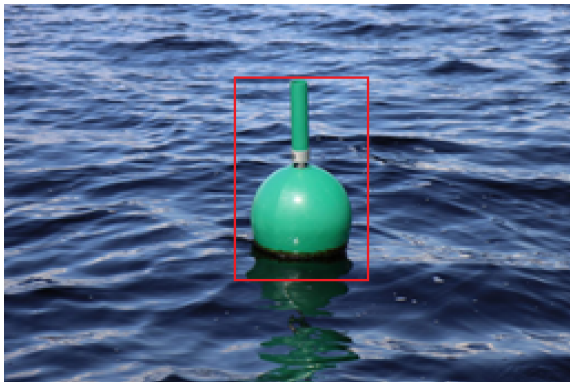
Fusing colourless lidar geometry with camera colour, cross-checked by the CNN, gives detections that are both accurate in position (lidar) and correct in class (camera). This approach has worked well for us on earlier projects.

3.4 Testing and validation

Modules have been validated individually and as an integrated system. The live lidar tool produces clean 2D scans of a known environment (Figure 6a), and the buoy detector reliably boxes channel buoys on real on-water imagery (Figure 6b). At the system level, the companion computer has been verified communicating with every sensor and the full CAN bus (BMS telemetry and VESC motor commands) simultaneously. Remaining work is field and integration testing of the fully assembled vessel: early trials in indoor pools in Trondheim, followed by full course testing against real buoys in both Trondheim and Oslo, across the light and water conditions we expect at the competition.



(a) Live RPLIDAR 2D scan of the workshop.



(b) CNN buoy detection on on-water imagery.

Figure 6: Perception testing: lidar geometry (left) and camera-based buoy detection (right).

4 Operator GUI / Human-Robot Interface

Our operator interface is **custom-built**, adapted from a proven solution our team developed for a 5G-controlled UGV. It has two parts: a native low-latency program for live video and manual control, and a web-based telemetry dashboard.

4.1 Design choices

The guiding principle is **minimum latency with maximum clarity**. The link uses **UDP hole-punching** to establish a *direct* connection between the vessel and the ground-station laptop over 5G, giving the lowest latency physically achievable rather than routing video through a relay or cloud. We chose a **custom native client** over a generic HMI (such as Mission Planner or Neptus) for two reasons: it minimises video latency, and it lets us tailor the displays to exactly the data an ASV operator needs. Alongside it, a lightweight **Flask web server** presents telemetry in a browser; this gives an at-a-glance dashboard, allows several clients (judges and team) to connect at once, and acts as an independent safety fallback if the native client has a problem. Both the direct link and the Flask dashboard are taken from earlier projects, so they are mature and well tested.

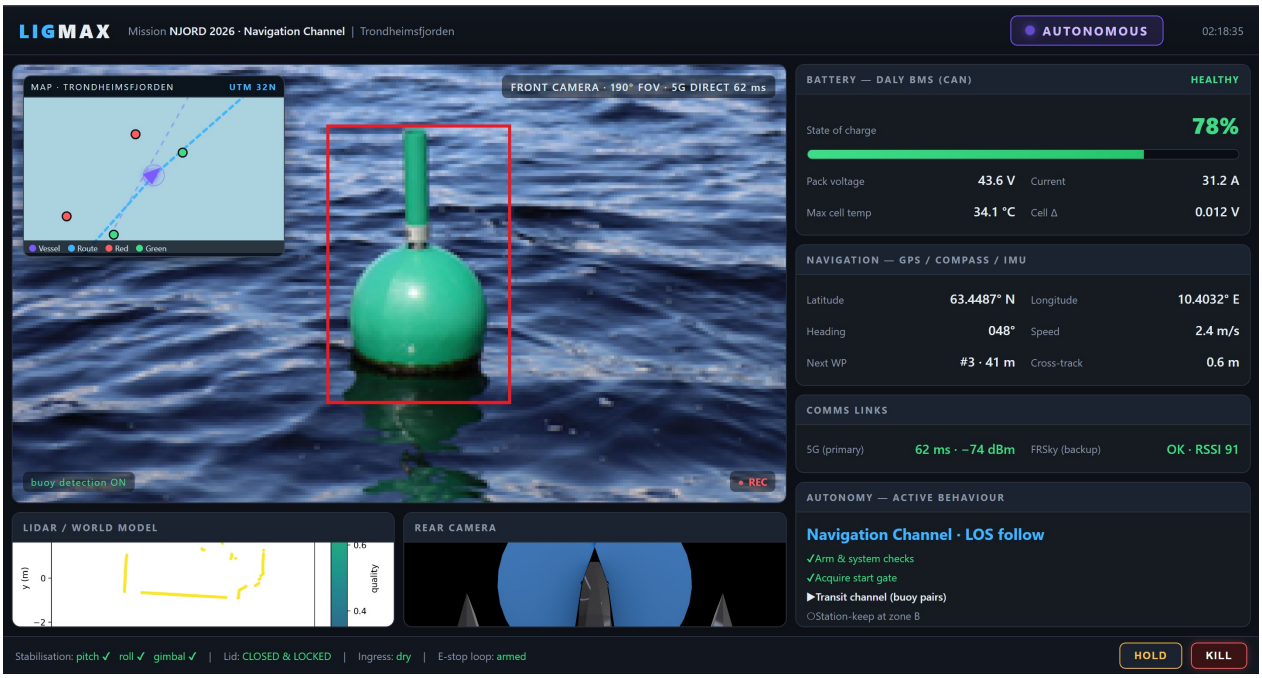


Figure 7: The operator console: a large front-camera feed with buoy detection, a Trondheimsfjorden minimap (vessel pose, planned route and detected channel buoys) inset top-left, and the safety-critical telemetry (battery, navigation, comms links, active autonomy behaviour) along the right. The mode indicator and HOLD/KILL controls are always visible. Manual takeover is instant via a game controller over the direct 5G link.

4.2 How to use it

The operator works from a single screen showing the **live camera feeds** and a **map** with the vessel's position, heading, planned route and detected buoys. The mode (disarmed, manual, autonomous, fault) and key status indicators are always visible, mirroring the on-board signal light. To intervene, the operator simply picks up the **game controller** and takes over manual control instantly over the direct link, with no menu navigation required, so a non-technical operator can drive immediately. A clearly marked **kill / hold** control stops the vessel, and the same command is available on the FRSky backup link.

4.3 Which data is shown and why

The dashboard surfaces the data an operator needs to keep the vessel safe and on task: **battery state** (state of charge, pack voltage, temperature) streamed from the Daly BMS over CAN, so the operator can spot a power problem early; **position, heading and speed** from the GPS/compass and IMU for navigation; **link status** for both the 5G and FRSky links, so loss of comms is immediately obvious; and the **current mode and active behaviour**, so the operator always knows what the autonomy is doing. Safety-critical items (battery, link, mode) are given the most visual weight, in line with the GUI/HMI requirements.

5 Major Design Choices

We describe three decisions that shaped the vessel, and for each we explain not only what we did but why we chose it over the alternatives we considered.

5.1 A low-poly trimaran rather than a monohull or catamaran

We needed a platform that is both fast and stable. A monohull is efficient but rolls, while a wide catamaran is stable but carries more wetted area and drag. The **trimaran** resolves the tension between the two: a slender central planing hull keeps drag and the resistance penalty low, while two outriggers on wide cross-beams give a large righting moment and a stable sensor platform. The low-poly form was a practical choice as well as an aesthetic one. Flat facets are

straightforward to 3D-print in sections and to skin with carbon-fibre cloth, which let us iterate the hull quickly and cheaply while still ending up with a light, stiff, waterproof shell.

5.2 A split Pixhawk and Raspberry Pi compute architecture

Autonomy and safety-critical control have very different requirements, so we deliberately separated them. A single companion computer would have to run perception and the tight real-time control loops on the same processor, where an operating-system scheduling hiccup could destabilise the vehicle. Instead, the **Pixhawk 6C** owns hard-real-time control (PID, thruster mixing, GNSS fusion and hardware failsafes) on a dedicated microcontroller running a mature, field-proven autopilot, while the **Raspberry Pi 5** runs the compute-heavy, non-deterministic perception and planning. MAVLink between them is a clean, standard interface. The payoff is safety and modularity: the vessel keeps stable low-level control and its failsafes even if the autonomy stack stalls or is restarted.

5.3 Dual 360° lidar with camera fusion, instead of a single 3D lidar or cameras alone

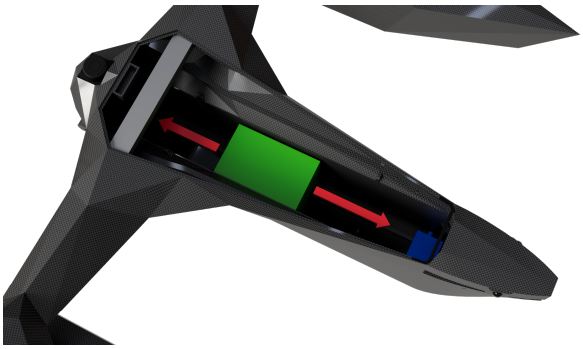
The course demands accurate geometry and reliable colour. A camera-only suite struggles with dependable ranging and with changing light on the water, while a single 3D lidar is heavy, power-hungry and expensive, and still provides no colour. We instead pair **two inexpensive 360° 2D lidars** (front and rear) for full-coverage, low-power, explainable range data with **wide-angle cameras** for colour. Geometry comes from the lidar and class from the camera, with a compact CNN as a cross-check. Stabilising the front lidar on a gimbal (Section 6) holds its scan plane horizontal and recovers much of the robustness we would otherwise look to a 3D sensor for, at a fraction of the cost, weight and power.

6 Innovative Aspect: Active Stabilisation with a Stabilised Sensor Package

6.1 What it is

Our headline innovation is a fully **active stabilisation system** that keeps the vessel, and therefore its sensors, level in waves instead of relying on hull shape alone. It has three coordinated parts:

- **Pitch trim by weight shift.** The 1.8 kWh battery, the heaviest single component, sits on two linear rails and is driven fore and aft by a dual-shaft **Nema 23** stepper and belt, moving the centre of mass to control pitch.
- **Roll trim by active outriggers.** Linear actuators in the amas change the geometry of the aka connections, adjusting the righting moment and the immersion of the main hull to counter roll.
- **A gimbal-stabilised front lidar.** An IMU-driven gimbal holds the front lidar's scan plane horizontal regardless of residual hull motion.



(a) Pitch trim: the battery (green) slides on rails driven by the Nema 23 (blue).

(b) Roll trim: linear actuators extend the aka connections.

Figure 8: The active stabilisation mechanisms for pitch (left) and roll (right).

6.2 How it improves performance

Keeping the platform level pays off in three ways. First, it gives **cleaner perception**: a horizontal lidar scan plane and a stable camera horizon mean detections are not smeared or lost as the boat pitches and rolls, so buoy ranging and colour classification stay reliable. This is the reason the sensor package is described as stabilised. Second, it improves

efficiency: shifting the heaviest mass fore and aft trims the vessel to the right attitude to plane, which reduces wetted area and drag at speed. Third, it adds **safety and steadiness**: active roll control keeps the vessel composed in wake and chop, which matters for docking and station-keeping tasks.

6.3 Technical implementation

All three loops run as closed-loop controllers using the IMUs of Section 2. The pitch loop reads attitude from the Pixhawk and commands the stepper, through its driver, to a rail position. The roll loop commands the ama linear actuators through a MOSFET driver. The gimbal loop runs off its own dedicated IMU for fast, local correction of the front lidar. A neat consequence of the design is that the **battery does double duty**: it is both the energy store and the trim mass, so the stabilisation costs almost no dead weight, only the rails and the stepper.

6.4 Comparison with conventional approaches

Most student ASVs use a fixed hull and rigidly mounted sensors. They accept scan-plane tilt and motion blur, try to clean it up in software, and rely on passive stability alone. By **mechanically** holding the boat and its sensors level, we attack the problem at the source. We obtain the clean data of a much more expensive stabilised or 3D sensor rig from inexpensive 2D lidars and cameras, and we gain a planing-efficiency benefit that a passive hull cannot.

6.5 Evidence

The individual building blocks are validated. The lidar produces clean 2D scans (Figure 6a), buoy detection works on real water imagery (Figure 6b), and the rail and stepper assembly and the roll actuators have been built and exercised (Figure 8). The end-to-end benefit of the stabilisation loops, namely the quantified reduction in scan-plane tilt and the planing-drag improvement, is the focus of our upcoming on-water tests in the Trondheim pool and on the buoy courses in Trondheim and Oslo.

7 Component overview

Figure 9: The majority of the project hardware, laid out. Item numbers correspond to Table 1.

Table 1: Main component list.

No.	Component
01	PLA hull component
02	Battery management system (Daly 150 A)
03	Main thrusters (Flipsky 5085)
04	Carbon-fibre cloth
05	Epoxy resin
06	Epoxy hardener
07	Li-ion battery cells (Samsung 35E)
08	Linear actuators (roll trim)
09	Lidars (Slamtec RPLIDAR)
10	Aluminium TIG weld sticks
11	DC-DC buck converter, low voltage
12	DC-DC buck converter, high voltage
13	Battery for system testing
14	5G router
15	Pixhawk power distribution board
16	Holybro Pixhawk 6C (flight controller / IMU)
17	VESC 6.7 FOC motor controller
18	Sonar
19	Lidar gimbal motor
20	IMU vibration-damping mount
21	Dual-shaft Nema 23 with belt pulleys (battery shift)
22	Idle belt pulleys
23	Screws
24	Pixhawk cables and PWM breakout
25	Gimbal structural part
26	Stepper-motor driver
27	Raspberry Pi 5 (companion computer)
28	GPS antenna (u-blox M9)
29	CAN communication Pi 5 HAT
30	Linear rails (battery shift)
31	Emergency-stop button

Additional components not pictured: stern thruster, 2040 aluminium profile beam, lid locking mechanism, lights, battery aluminium plate, and mounting hardware (belts, bolts, misc.).