Njord Competition Technical Report
Team: Strath Voyager

August 13, 2023

Panagiotis Louvros, Wang Zhao, Paul Lee, Mauro Arcidiacono, Kaja Gorska, Charalampos Tsoumpris, Alexander Dixon, Nilesh Ohol, Marvin Wright, Moritz Troll, Muhammad Iqbal, Mohamed Adlan Ait Ameur, Evangelos Stefanou Isah Jimoh, Myo Zin Aung, Dr Saishuai Dai, Dr Laibing Jia, Prof Gerasimos Theotokatos

University of Strathclyde
Henry Dyer Building
Montrose St
Glasgow G4 0LZ
United Kingdom

1. Abstract

Starting from basic design and operational concepts on paper, as modular ASV was designed and built within approximately 12 months on a budget of around £5000. The boat participating in the Njord competition features a catamaran design comprised of two custom designed NPL hulls. The catamaran platform was designed to be fully modular and is constructed with 3D-printed hulls and carbon fibre decks. The platform is propelled using a static dual-thruster configuration. Computation is facilitated by an Nvidia Jetson, a Raspberry Pi 4B, and an Arduino MEGA. The system is powered by two custom-made 5S BMS 21V Li-Ion battery packs. The navigation sensor system consists of an Ouster OS32 3D Lidar with a built-in IMU, a Stereovision Zed Mini depth camera, an Adafruit Ultimate GPS Module PA1616D, and an Adafruit BNO055 Absolute Orientation Sensor. Further internal sensors, including voltage, current sensors, as well as a temperature sensor, are integrated. The ASV connects wirelessly to a remote control via an nRF2401 module, and to a laptop for operation monitoring via Holybro Telemetry Sik radio. Autonomous operation is enabled through sensor fusion of camera and Lidar information to identify environmental features, such as waypoints, and to generate virtual waypoints for the autonomous control system. The autonomous control system consists of an in-house developed deep reinforcement learning (DRL) algorithm that enables Line of Sight operation, as well as obstacle avoidance. As a backup, a general PID LOS waypoint tracking controller is also implemented in parallel. Extensive lab and pond testing has been carried out to develop and refine the system. Key features and innovations of the StrathVoyager ASV include:

- Fully modular 3D printed and carbon fibre ASV.
- Depth camera and 3D lidar sensor fusion for autonomous control.
- DRL-based autonomous control.

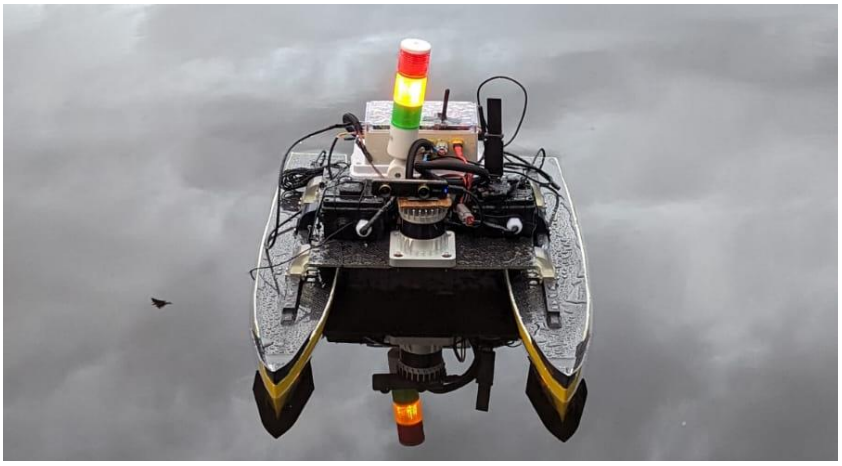


Figure 1 Strath Voyager ASV.

2. Autonomous Surface Vehicle Design & Manufacturing

2.1. Hull & Platform

2.1.1. Hull & Platform design

Central to this design was the concept of modularity. A modular design ensures ease of repairs, upgrades, and replacements of individual components, and simplifies transportation. Mechanical modularity was achieved through non-permanent fixtures between structural elements, such as the hull and the connecting platform. The platform connects to the hulls via four brackets that attach to a rail system on the hulls, allowing movement of the centre of mass in the boat's normal direction and adjustment of the boat's trim. The platform is a perforated carbon fibre plate designed for easy equipment mounting using nuts and bolts. Figure 2 shows a CAD model of the hull and platform without mounted electronics or the propulsion system.

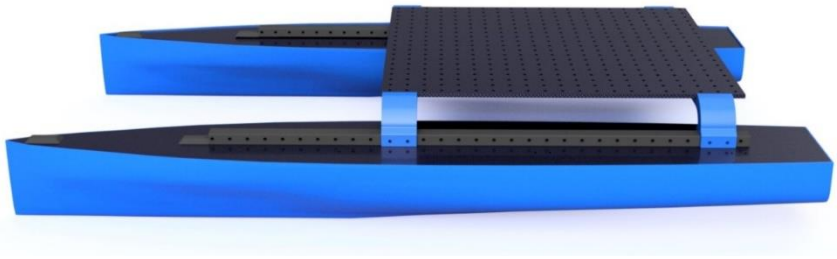


Figure 2 CAD model of modular hull.

A custom hull design was created based on the NPL Hull Series. Changes were made to align with the design's target dimensions and enhance hydrodynamic performance. Manual adjustments and optimization modifications to the S-NPL Hull 4b were carried out to create the ultimate hull design. Table 1 offers an overview of the final design characteristics for each of the catamaran hulls. The hulls half breadths, horizontal longitudinal hull sections were imported from Excel to Solidworks to create a CAD model. Figure 3 shows the imported hull surface points within SolidWorks.

Table 1 Overview of Hull design parameters**.**

Parameter	Symbol	Unit	Value
Length-displacement Ratio	$L = \nabla^{\frac{1}{3}}$	–	7.4
Length-to-Beam Ratio	L/B	–	9
Beam-to-Draught Radio	B/T	–	2
Block Coefficient	c_B	–	0.397
Approx. Mass Displacement	Δ_{full}	kg	7.5

Mass Displacement Demi-Hull	Δ_{demi}	kg	3.75
Fresh Water Density (15 deg c)	ρ	kg/m^3	999
Waterline Length	LWL	m	**1.150**
Beam	B	m	**0.128**
Draught	T	m	**0.064**

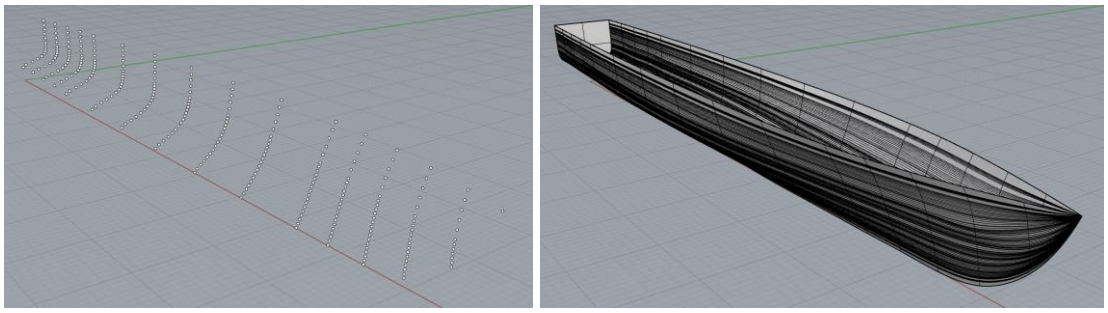


Figure 3 Hull half breadths and raw hull CAD model.

To attach the hull decks to the SLA printed hulls, a system of fused deposition modelling (FDM) printed bulkheads was manufactured. Three of these bulkheads were structural, and one was solid and watertight. This design aimed to ensure resilience and survivability in the event of a frontal hull impact and failure. The bulkheads were designed with pre-shaped features to secure the hull decks with M3 screws. The final bulkheads were attached to the hulls using epoxy resin.

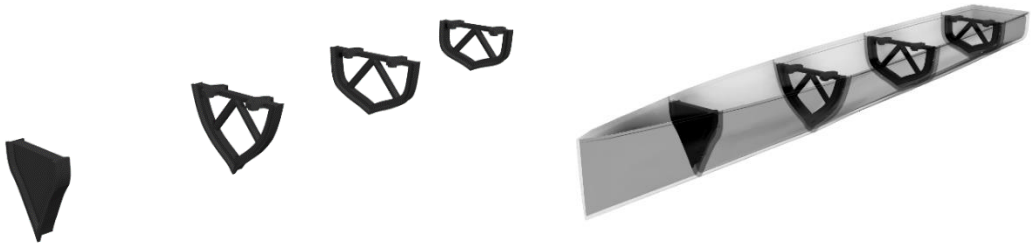
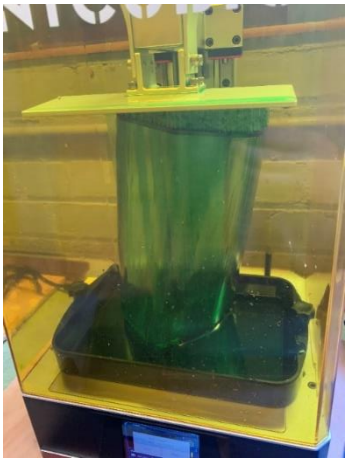


Figure 4 Hull bulkhead design CAD model.

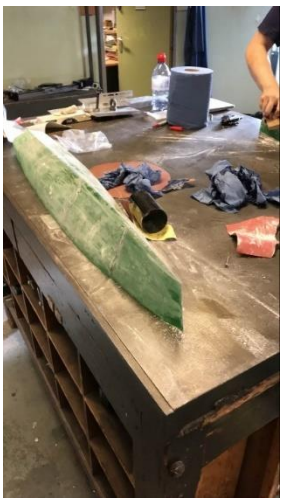
2.1.2. Manufacturing of hull and platform

The hulls were manufactured using Stereolithography with the ANYCUBIC MONO X printer and standard ANYCUBIC UV-sensitive resin. Due to the size of the hulls, they were divided into five sections. Each section is connected to its neighbouring sections via a flange and eight screws. Additional bonding was achieved through a two-component resin glue.



(a)

(b)



(c)

(d)

(e)

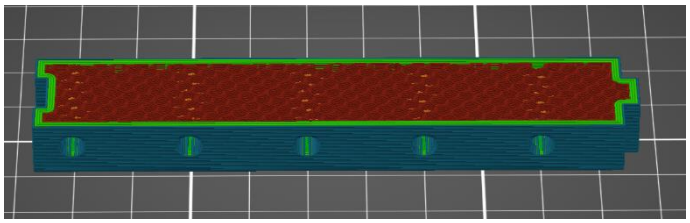
Figure 5 Hull manufacturing process (a) hull section SLA print (b) two 3D printed hull sections (c) hull sanding (d) hull coating (e) hull filled with expansion foam.

To minimize drag, the outer surfaces of the hulls were sanded smooth and coated with resin paint. The decision to use lightweight expanding foam came later in the assembly process. This foam provides weatherproofing for the craft by making the hulls resistant to water ingress, enabling the craft to operate in moderately adverse weather and water conditions.

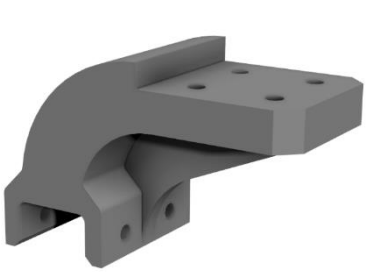
The three separate decks, consisting of two hull decks and one mission deck, were cut from 2mm carbon fibre board and 4mm carbon fibre composite board, respectively. These hull decks also featured cut-outs for the rail system increasing both the accuracy of construction and the resilience of the final ASV. The final components had numerous 3mm holes through which M3 machine screws could be used to secure them to the bulkheads within the hulls.

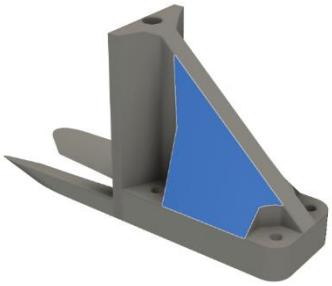
The mission deck was designed to maintain the recommended 2:1 ratio for vessel length to beam, providing directional stability at speed while not hindering mobility due to the twin thruster propulsion design. Circular cutouts were integrated into the mission deck, allowing various components to be attached as needed.

The system for mounting the mission deck to the hulls utilized a modular rail system, secured by SLA printed brackets to the mission deck itself. The final files were the result of an iterative process and various performance improvements. Figure 5 illustrates CAD models of the rail system, brackets, and thruster mount. A lightweight 3D printed mount/heatsink was created for the Lidar to replace the original heavy aluminium part, reducing the component's weight. Shown in Figure 6, thermally resistant PETG plastic with a GTT of 85°C was used. Additionally, 3D printed bulwarks were added to the hull's bow.



(a)





(b) (c)

Figure 6 (a) FDM modular rail element showing 3D print infill (b) SLA modular Rail-Platform bracket (c) FDM thruster mount.

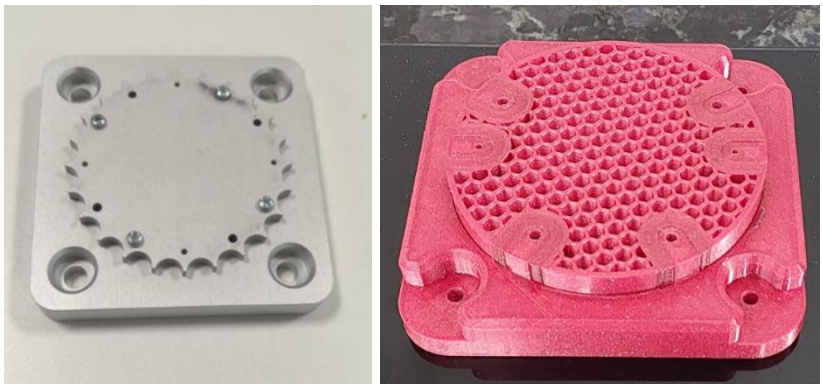
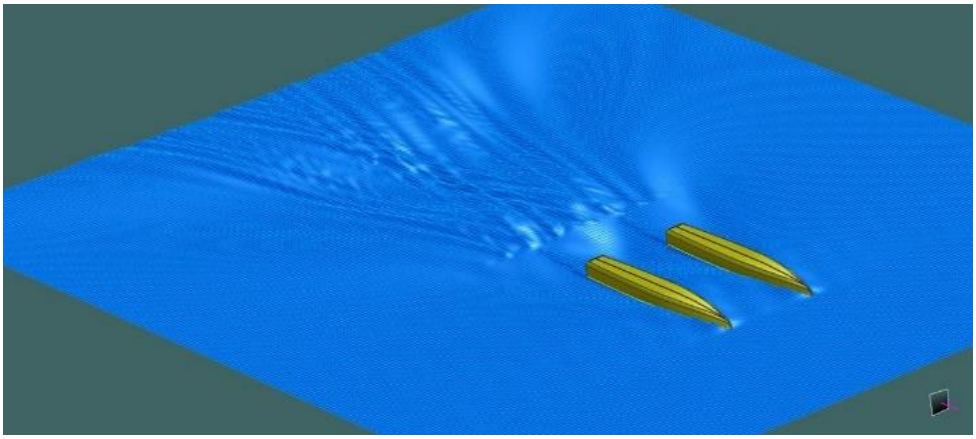


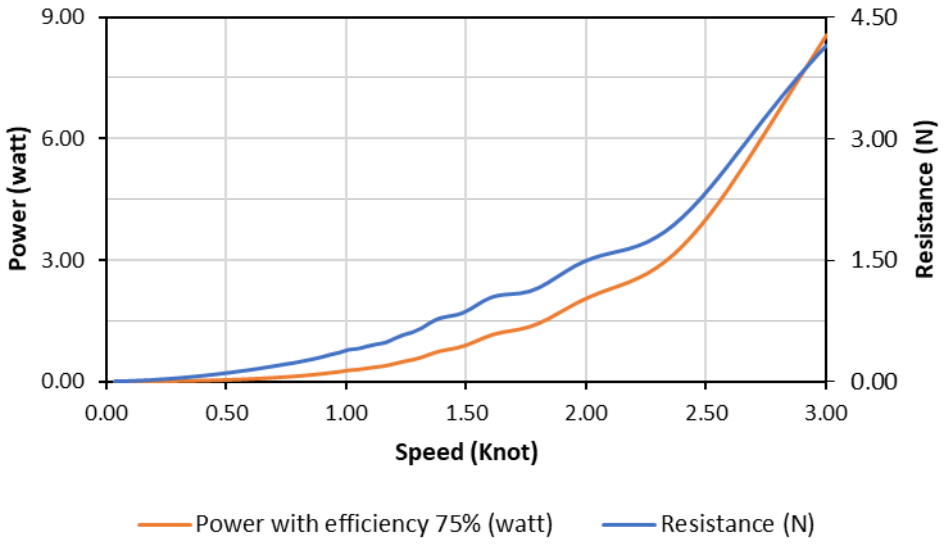
Figure 7 Original and light weight 3D printed LiDAR heatsink.

2.2. Propulsion Design

Hydrodynamic analysis was conducted to evaluate the propulsion requirements of the boat. A simplified model of the boat, representing the two hulls, was generated using the commercial software Maxsurf. The simulation model was configured using design estimates for payload and velocity. Figure 7 illustrates the boat's wake structure, along with the anticipated speed-to-resistance and speed-to-power curves.



(a)



(b)

Figure 8 Maxsurf simulation: (a) wake wave structure and (b) Power/ Resistance to speed curves.

Based on the simulation results, a dual fixed thruster configuration was selected. Propeller thrusters offer responsive manoeuvrability at anticipated lower speeds and outperform alternatives such as water jets. A propeller thruster is fixed at the stern of each hull and fully submerged within the water.

The distance towards the centre of mass allows the thruster configuration to generate momentum for agile turning. Two M060 thrusters from Full Depth were chosen. The thruster model fits within the overall budget and demonstrates sufficient thrust while maintaining acceptable power requirements. Each thruster can produce a maximum thrust of 0.9 kg at 12 Volts and 2 Amps.

A comprehensive explanation of the controller's electrical connection to the boat's circuitry was provided in section 2.5. In summary, each thruster is linked to an Arduino Mega through a motor controller. Thruster commands adhere to the PWM format and are restricted to 1100-1900 ms to facilitate forward and backward motion of the boat. Turning moment is achieved through a bias between the two thrusters, enabling on-the-spot turning. An empirical conversion relationship between the PWM signal and resulting speed was established during laboratory testing, along with further characterization of the ASV's manoeuvring model. Control of the thrusters is managed via the Arduino MEGA.

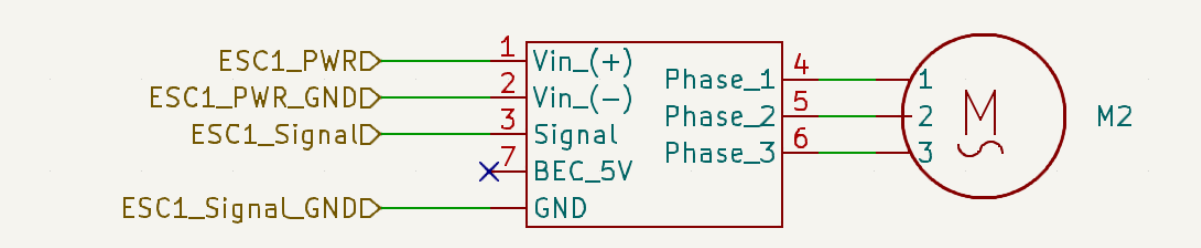


Figure 9. ESC and motor.

For forward rotation, Arduino generates a PWM pulse ranging from 1470 to 1900ms (pulse width), mapped to 0-100. Conversely, for aft thrust, the PWM pulse ranges from 1470 to 1100. The Arduino handles this conversion, having been equipped with values for the left or right motor falling within the range of -100 to 100. Motor commands originate from various sources, including radio communication, internal commands (e.g., halting motors or executing manoeuvres in case of signal loss), and autonomy provided by the Jetson.

2.3. Batteries

The ASV is powered by two custom-made Li-ion battery packs, which are stored in waterproof, removable containers on the mission deck. Each battery pack is linked to a power distribution board and can independently power the system. Figure 10 illustrates the electrical system topology.

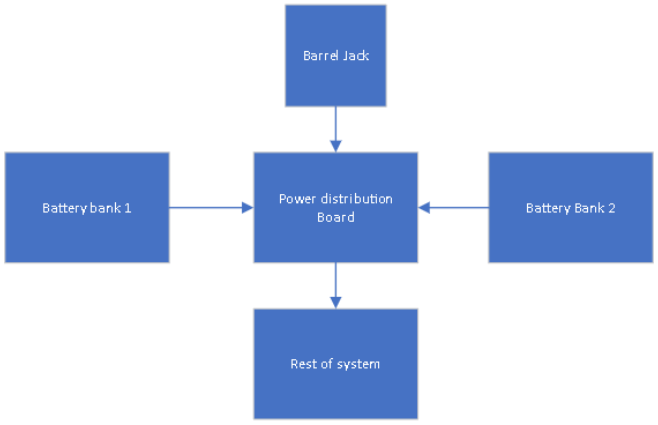


Figure 10. Battery electrical topology.

Each battery pack comprises a 5S Battery Management System (BMS) and five 18650 batteries, resulting in nominal and maximum voltages of 18V and 21V, respectively. The packs are equipped with waterproof DEUTSCH connectors, providing +V (positive), GND (negative), and 5V and signal connections.

Safety was a fundamental focus in the design of the battery system. Each battery pack features an integrated battery management system capable of protecting the batteries from various potential irregularities. Additionally, each pack is equipped with a 5A fuse installed in-line on the positive side. Voltage and temperature measurements complete the system. Table 2 provides an overview of the battery packs.

Table 2. Battery datasheet of a battery pack.

Number of cells and charge per cell	5 x 3350mAh nominally
Charging Voltage	21 V
Nominal Voltage	18 V
Charging Method	CC-CV (constant voltage with limited current)
Charging	Current Standard charge: 1,700mA. For cycle life: 1,020mA
Max. Charge Current	2,000mA (not for cycle life)
Max. Discharge Current	8,000mA (for continuous discharge) 13,000mA (not for continuous discharge)
Discharge Cut-off Voltage	13.65 V
Battery pack weight	393 g

2.4. Sensors

Sensing is a fundamental aspect of an autonomous system. The ASV's navigational sensors encompass a 3D LiDAR with an integrated IMU, a depth camera, GPS, and compass. Additional sensors facilitate system monitoring, encompassing voltage, current, power, and temperature sensors.

2.4.1. Lidar

The OUSTER OS0-32 Rev 6 LIDAR sensor serves as the primary information source for object localization. With a 90º vertical field of view (32 channels) and a horizontal resolution of 512, 1024, or 2048, this sensor weighs 447 g and holds an IP68 rating. While its maximum detection range reaches 50 m, for detecting small objects in the water, the maximum useful range was determined to be approximately 15 m. OUSTER provides an SDK in Python for sensor configuration, recording and reading lidar data, managing point clouds, and more. Additionally, the OUSTER lidar includes an IMU and the capability to connect to an external GPS module. Leveraging these features offers the advantage of achieving time synchronization across all captured data. Nevertheless, a different IMU and GPS sensors were used, primarily due to the fact that these alternatives are ready to use out-of-the-box, requiring minimal configuration.

2.4.2. Camera

The ASV is equipped with a Zed Mini camera with a field of view of 110 degrees. The camera can measure distances with a 0.1-15 m depth range and the data can be obtained through a Python API. The camera weights 60g and it has a USB 3.0 Type-C port. The system is equipped with an accelerometer and a gyroscope. The output resolutions from higher quality to lower quality are the following: 4416x1242 (2.2k), 3840x1080 (1080p), 2560x720 (720p), 1344x376 (WVGA).

2.4.3. GPS and Compass

There are two sensors dedicated for understanding the position and orientation of a ASV: Adafruit Ultimate GPS Module PA1616D and Adafruit BNO055 Absolute Orientation Sensor. Together, they provide all essential information for navigation, such as: latitude, longitude, speed over ground, course over ground, heading, linear acceleration, angular velocity.

2.5. Hardware Implementation

The hardware on the boat was organized with a total of four waterproof boxes that support the modularity of the system design. In addition to the two battery boxes, there are high-level and low-level computing boxes. The first computing box is centred around an Arduino MEGA, which acts as the host for all low-level functions. It is internally and externally connected to circuits through DEUTSCH connectors (total of 3 connectors). This box manages all low-level functions, including relays, power distribution and monitoring, RF communication, and more. This functionality is sufficient for remote boat control.

Figure 11 illustrates a circuit diagram, incorporating components like four main relays, the nRF2401 module, two electronic speed controllers for the BLDC thrusters, along with 18V (nominal), 5V, 3.3V, and common ground buses. The box also includes provisions for voltage, current, and temperature sensing within the box. Importantly, the box is fitted with a hardware kill switch and a software kill switch. Note that kill switch functions are handled by the Arduino and the relays, in addition to software commands.

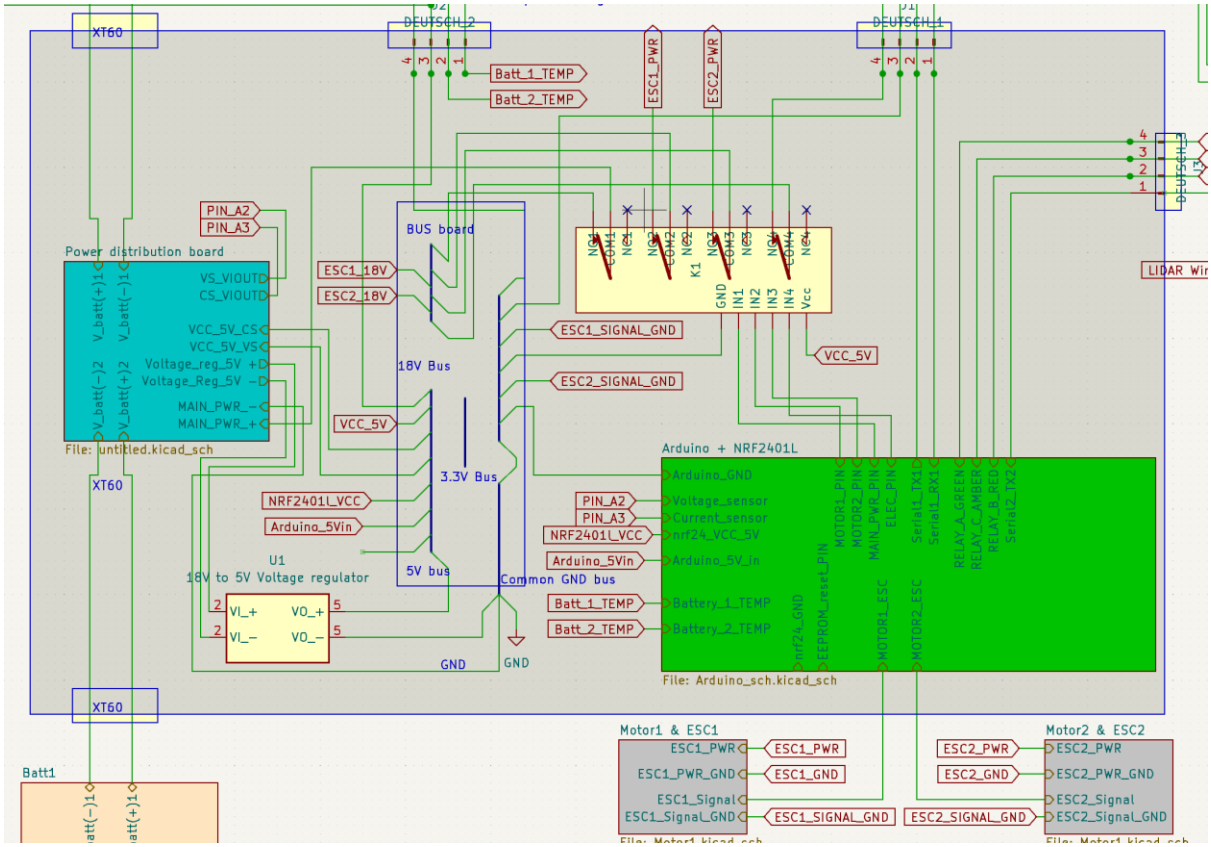


Figure 11. Circuit diagram cut-out of Arduino box.

To achieve remote control of the boat, information needs to be sent to the Arduino MEGA onboard, referred to as RX henceforth, wirelessly. There are several radio frequency modules that can be utilized for this. The used NRF2401 modules operate at 2.4GHz and have a theoretical range of +500 m. Connect is established with another Arduino onshore. In addition, a separate 433MHz Sik Radio is used to send telemetry data from Jetson to the shore at a separate computer. This radio also allows manipulation of the Jetson on board.

The second box accommodates the main computing unit, consisting of an Nvidia Jetson Nano and has arrangements for including a Raspberry Pi 4b 8GB (might not be used in final Implementation). It also houses the LIDAR interface box, GPS and IMU sensors, light relays, and their associated voltage regulators. The Jetson has its dedicated voltage regulator, while another regulator powers the relays and any additional devices that might be incorporated later. The light relays are both powered and controlled through three relays, which are managed by the Arduino in the other box. The LIDAR interface box is directly powered by the 18V bus. It's important to note that the power supply to this box can be switched off by the Arduino (corresponding to Relay 4). The full electric diagram and further component electric diagrams are added to the appendix.

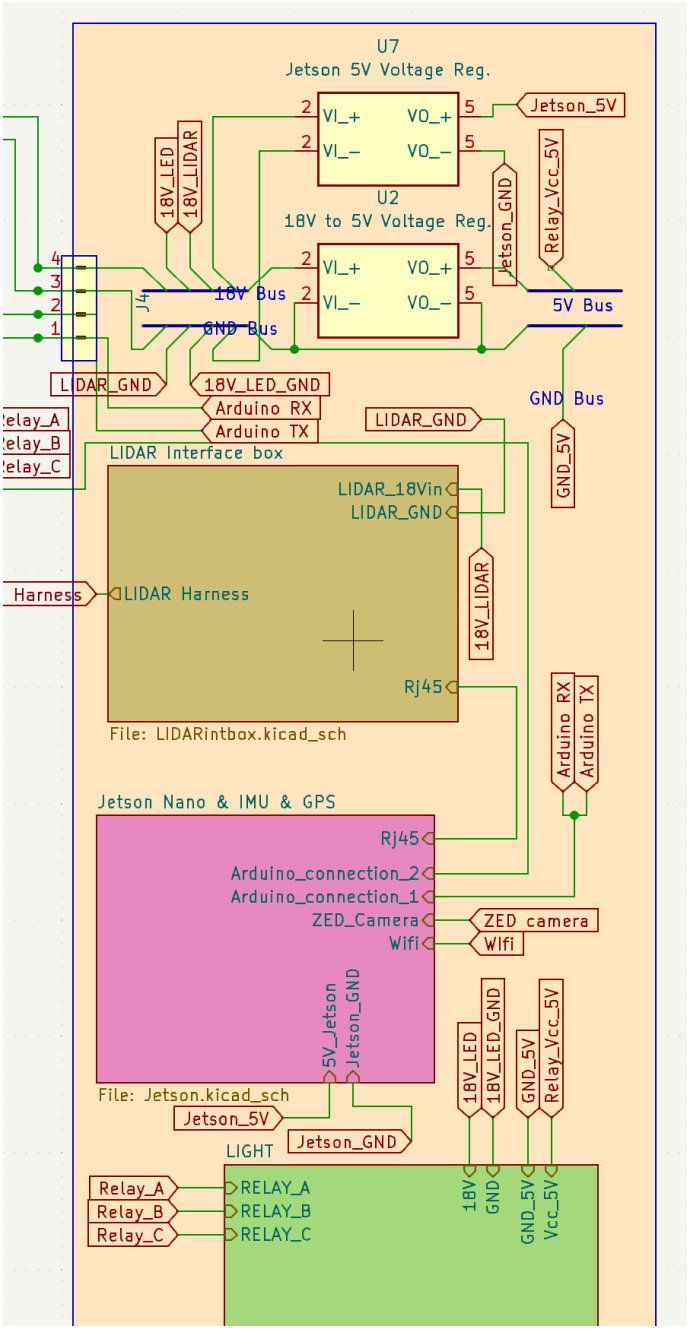


Figure 12. Electronics box circuit diagram.

2.6. Software Design

2.6.1. Arduino code

There are two Arduino controllers used for StrathVoyager; Arduino RX (onboard) and Arduino TX (shore). Below are some further information on the two codes:

Arduino RX

The script is 1000 lines long and generally accomplishes the following tasks:

1. Internally switches relays on/off to protect the circuits or
2. Actuates relays according to user input.
3. Measures voltage, current, temperature.
4. Packagers telemetry data and sends to shore.
5. Receives control data and acts accordingly.
6. Controls the motors.
7. Calculates State of Charge of batteries.
8. Activates light accordingly to situation.
9. Can switch between radio or autonomy control.

The script is written in a way to avoid blocking logics and has been tested in varying combinations of errors and exceptions to ensure the robustness of safety and safeguard systems. Such systems include:

1. a kill switch for all relays and reset of Arduino remotely,
2. radio signal timeout and kill switch activation after a stopping manoeuvre.
3. Overcurrent, undervoltage protection
4. Stopping of autonomy in cases of loss of signal or control.

Arduino TX

The TX Arduino is connected through USB to a laptop which runs a Processing script that writes to that Arduino. The Processing script itself is fed information via an Xbox controller connected to the same laptop. The TX Arduino then transmits the data in predetermined format to the RX Arduino. The RX Arduino acknowledges receipt of the data and every 2 seconds also returns Telemetry data that is readable at the laptop screen

The TX Arduino is fitted with an emergency stop button that will trigger a kill switch for the motors if pressed. This does not require the connection of the Xbox controller which can also actuate the kill switch. If the TX Arduino loses power or connection is lost for more than 5 seconds, the kill switch is activated as well.

2.6.1. Integrated System Architecture

The software of the ASV is managed using the Qt application manager. It handles tasks such as gathering and processing data from sensors and establishing a coherent communication network. A key aspect of this programming efforts lies in the collaboration between Qt and PyQt5. These two

components work together to shape the architecture and create an intuitive and dynamic control interface. PyQt5, a powerful GUI development framework, synergizes with the foundation of Qt to empower us in building an interactive control system. This system seamlessly integrates with the ASV's navigation functions. By utilizing event-driven mechanisms and ensuring compatibility across various platforms, PyQt5 ensures real-time responsiveness and consistent performance in different operational scenarios.

The software architecture of the ASV, shown in Figure 13, was designed to seamlessly integrate the capabilities of the NVIDIA Jetson, Arduino, and RF modules to enable efficient and robust autonomous navigation, control, and communication. The architecture was divided into several key components, each serving a specific purpose in the functioning of the ASV.

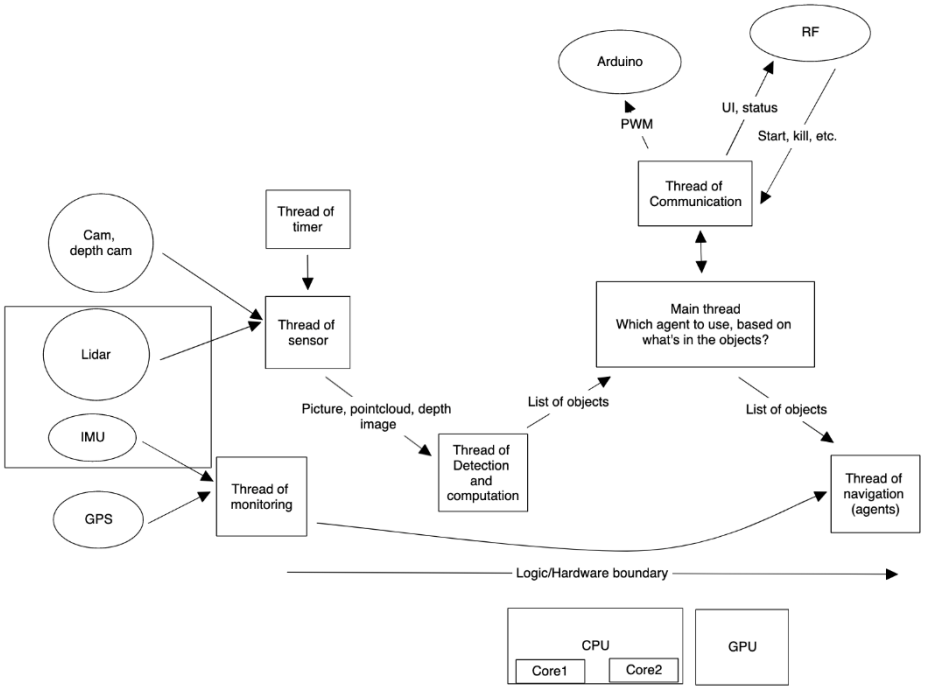


Figure 13 ASV software architecture.

Perception module

The Perception module is responsible for gathering data from various sensors onboard the ASV and processing it to generate an understanding of the ASV's environment. The NVIDIA Jetson, equipped with its powerful GPU, plays a central role in this module. It processes data from the camera, lidar, GPS and IMU sensors to create a detailed perception of the surrounding environment and the ASV's current

state. Object detection, point cloud processing, and sensor fusion methods are implemented in this module to provide a real-time representation of the ASV's environment.

Navigation module

The Navigation module takes the perception data and generates high-level decisions for the ASV's navigation. This module integrates DRL algorithms and PID control logic to interpret the surrounding and produces commands for the low-level control module based on learned behaviours or rule actions. Waypoint generation for path planning, obstacle avoidance, and berthing are all part of this module. The NVIDIA Jetson's computing capabilities enable rapid and accurate calculations, crucial for autonomous navigation.

Low-level control module

This module interfaces with Arduino, which acts as the low-level controller. The Arduino takes the high-level navigation commands and transforms them into precise motor control signals, adjusting the motor's rotational speed as needed. Hardware interface is USB to TTL board connecting to UART ports on Arduino.

Communication Module

The Communication Module establishes a reliable link between the ASV and the shore using RF modules. These modules facilitate bi-directional communication, enabling remote monitoring, control, and data exchange. Real-time telemetry data, such as the ASV's location, status, and sensor readings, are transmitted to the shore. Additionally, navigation commands and updates can be sent from the shore to the ASV if necessary.

Graphical User Interface (GUI) module

The Graphical User Interface (GUI) module offers an intuitive platform for operators to seamlessly interact with the ASV. This module dynamically generates real-time visual maps that portray the ASV's ongoing trajectory, its status, and the precise positions of detected objects. Furthermore, it provides details such as motor rotational speed, battery state of charge, voltage levels, and the system's current draw. By logging this information, the GUI module facilitates post-mission data analysis, serving as a valuable resource for in-depth processing and assessment once the mission concludes.

2.7. Autonomous control

The development of the autonomous control entails the employment of artificial intelligence (AI) algorithms, such as DRL agents, which are trained in digital twin environments to learn end-to-end decision-making from sensor data, such as the LiDAR, to the control of the actuators, such as the thrusters. The aforementioned development has been conducted in MATLAB 2021a, whose details are analyzed in the following sub-sections.

2.7.1. Digital twin environment

A high-fidelity digital twin environment is pivotal for the training and verification of AI-based systems prior to the full-scale deployment in real-world scenarios. The most critical digital twins developed are provided below.

2.7.1.1. Manoeuvring model

The manoeuvring model considers numerical simulation of the manoeuvrability of the vessel in time-domain. To derive the manoeuvring model, the Maritime Robotics Otter vessel was used as the initial reference, as provided by the Marine Systems Simulator package of Professor Fossen from NTNU, which considers a 6 degrees of freedom (DOF) manoeuvrability of a small-scale catamaran vessel. To derive the manoeuvring model that represents the manoeuvrability of the ASV, the model was optimised using experimental data from free running manoeuvring tests conducted at the Kelvin Hydrodynamic Laboratory. Particularly, a total of 40 manoeuvring tests were conducted to capture various straight path, zigzag, and turning circle manoeuvres using Qualisys with a broad range of PWM combinations [-15, 50], as shown in Figure 14.

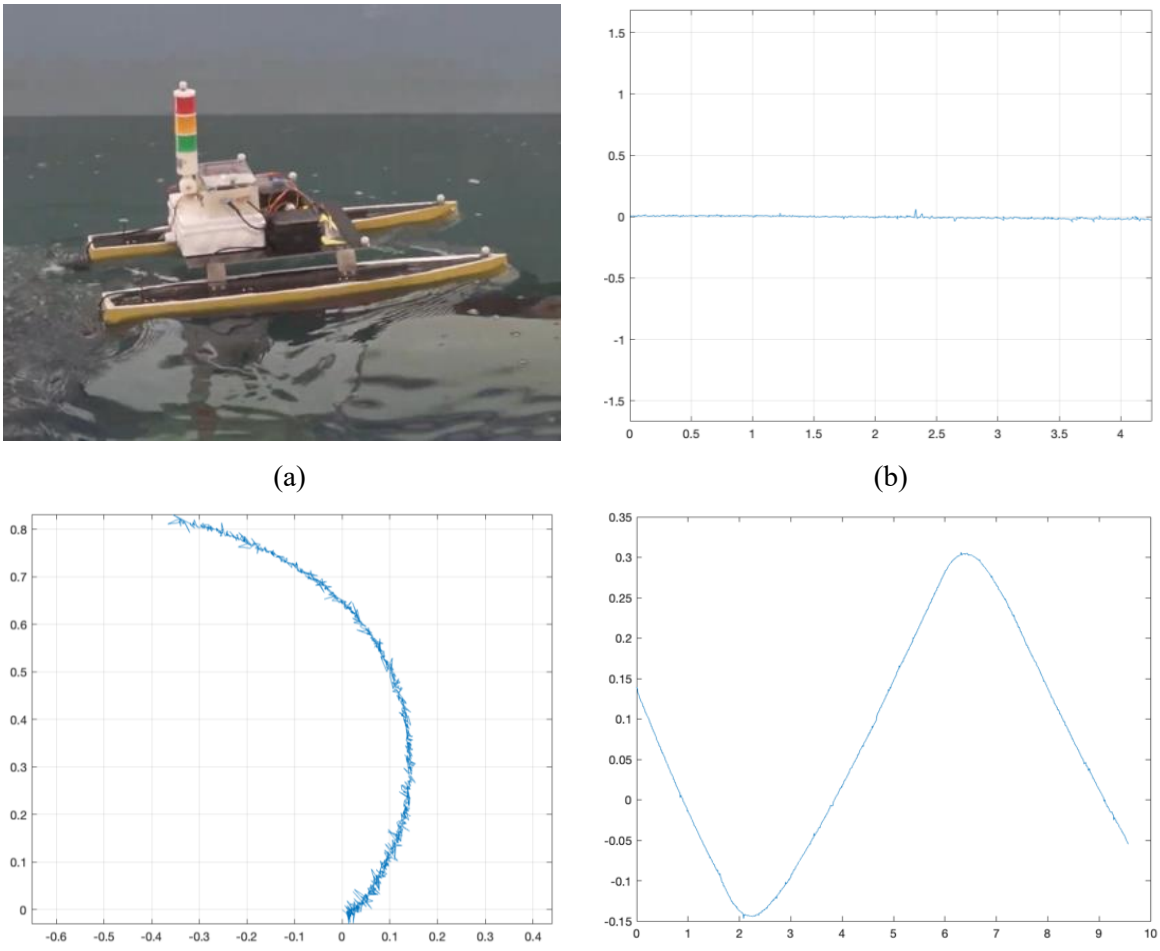


Figure 14 (a) Free running manoeuvring tests; (b) straight path manoeuvre; (c) turning circle manoeuvre; and (d) zigzag manoeuvre. Horizontal axis: x (m), vertical axis: y (m).

The Qualisys data were used to derive the empirical relation between the PWM and thrust of the ASV, as shown in Figure 15. For the final optimisation, the model was treated as a non-linear grey-box model to estimate the hydrodynamic added mass and linear damping terms using the System Identification Toolbox.

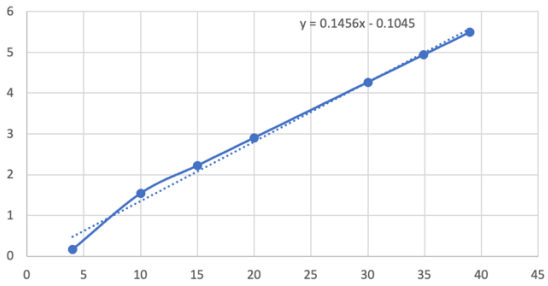


Figure 15 Empirical relation of thrust as a function of PWM for [5,40]. Horizontal axis: PWM (%), vertical axis: Thrust (N).

2.7.1.2. Training scenarios

To train the decision-making agents, virtual training scenarios were developed based on the competition's criteria. In order to increase the robustness of the agent against unforeseen scenarios, random training scenarios were generated in every new training episode as shown Figure 16, Figure 17, and Figure 18. The training scenarios for manoeuvring and navigation task entails randomness of the angle between each buoy pairs, for the docking the docking location, and for the collision avoidance, the relative angle of the target vessel.

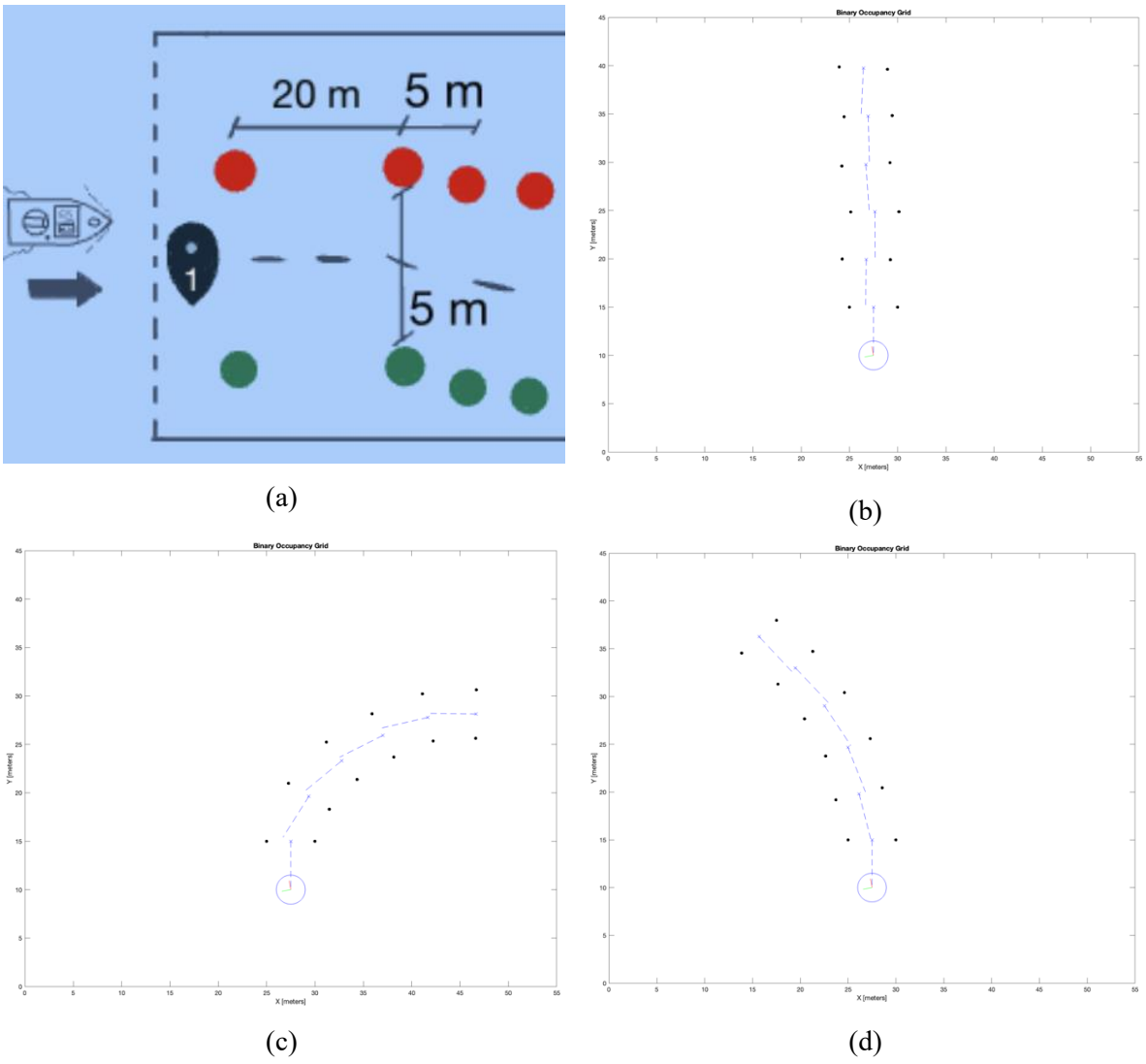
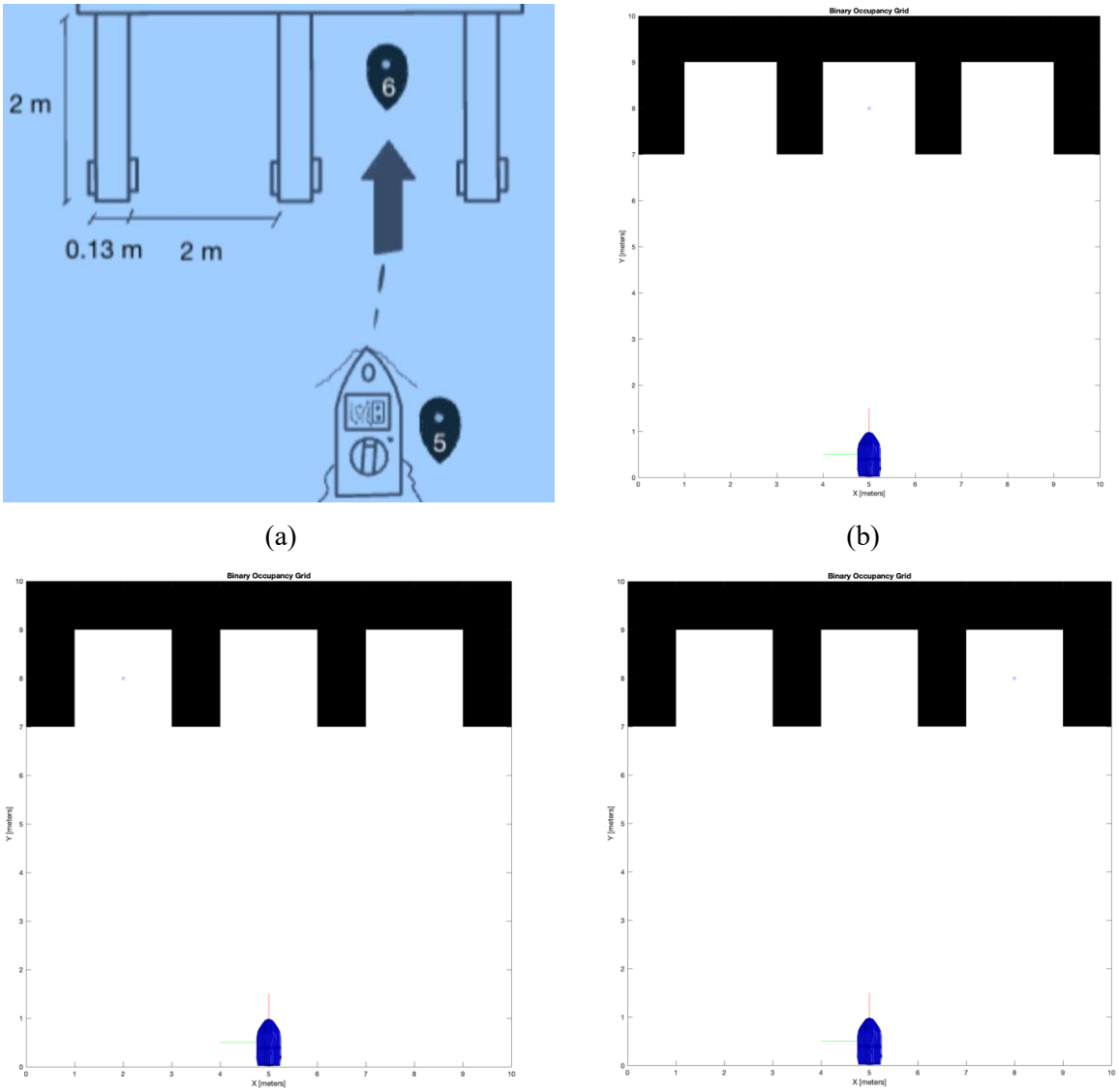


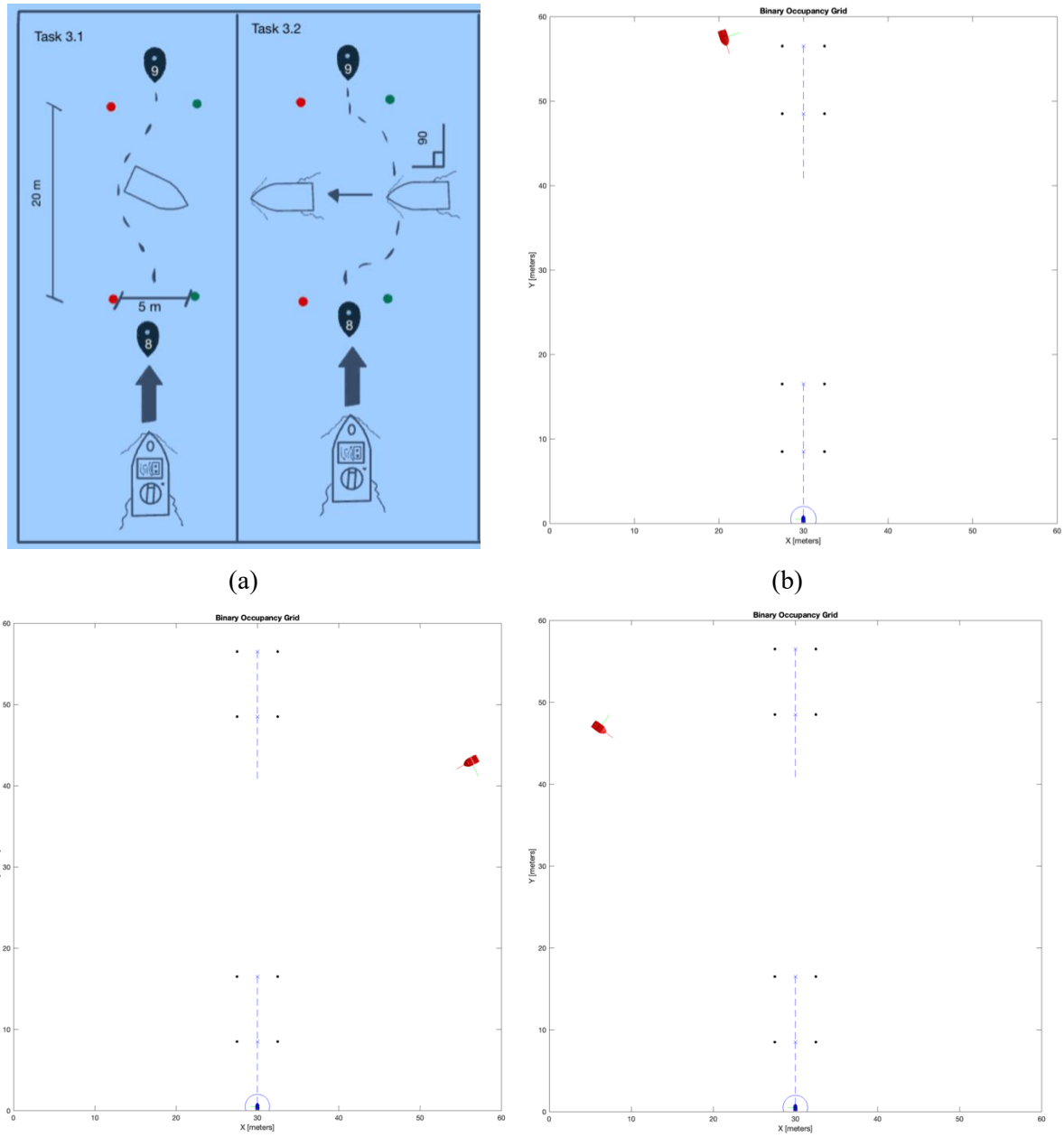
Figure 16 (a) Schematic representation of the navigation task provided by the competition; and (b-d) randomly generated training scenario.



(a) (b)

(c) (d)

Figure 17 (a) Schematic representation of the docking task provided by the competition; and (b-d) randomly generated training scenario.



(a) (b)

(c) (d)

Figure 18 (a) Schematic representation of the collision avoidance task provided by the competition; and (b-d) randomly generated training scenario.

2.7.2. Decision-making agent

A DRL based deep deterministic policy gradient (DDPG) algorithm was developed for the decision-making. Particularly, two hidden layers were used for both the Critic and Actor Networks, whose number of neurons and hyperparameters were tuned accordingly depending on the complexity of the decision-making task. The flowchart of the DDPG agent's training is presented in Figure 19.

For each task, different decision-making objective were set. Specifically, the objectives for the manoeuvring and navigation task were: 1) sailing at nominal speed, 2) minimisation of heading error from the line-of-sight path and the heading of the ASV, and 3) minimisation of the cross-track error from the line-of-sight path. The objectives for the docking task were: 1) sailing at target speed that linearly reduces based on the distance from the final docking location, and 2) distance margin from docking using LiDAR detection. Finally, the objectives for the collision avoidance task were: 1) sailing at nominal speed, 2) minimisation of heading error from the line-of-sight path and the heading of the ASV, 3) minimisation of the cross-track error from the line-of-sight path, 4) COLREG rules based relative angle penalty, and 5) maximisation of the distance from the ASV to the target vessel. The training progress and verification of the agent for each task are presented in Figure 20 and Figure 21, respectively.

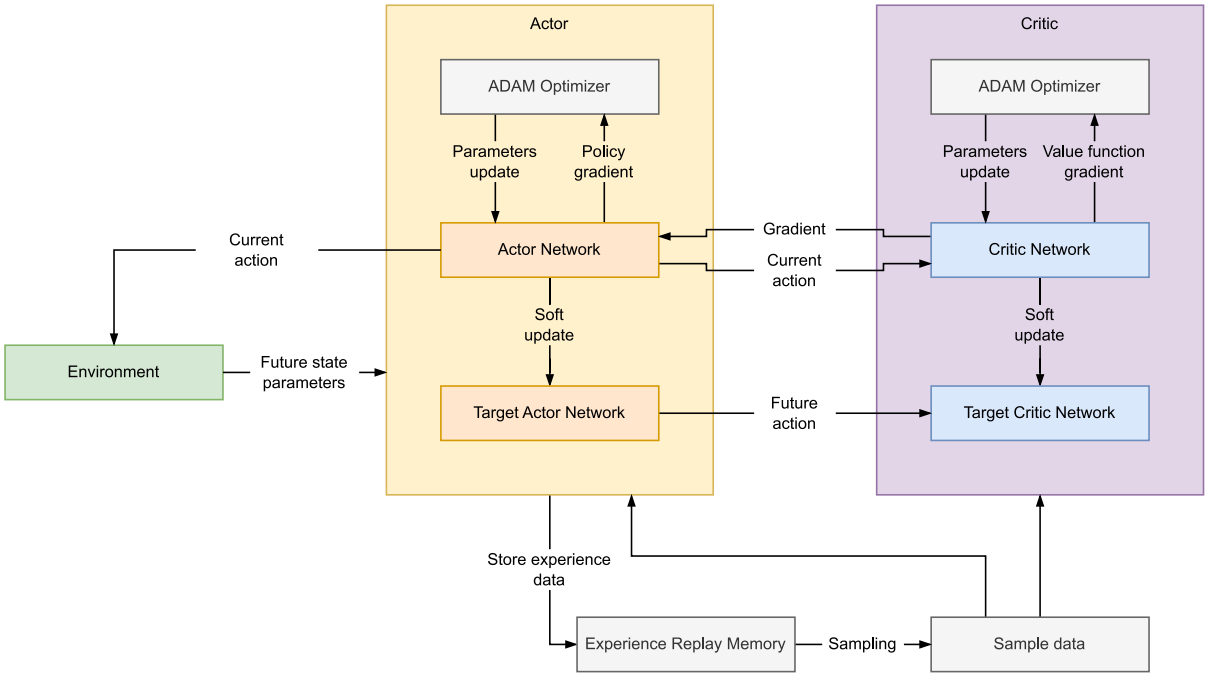


Figure 19 Flowchart of the training process of DDPG algorithm.

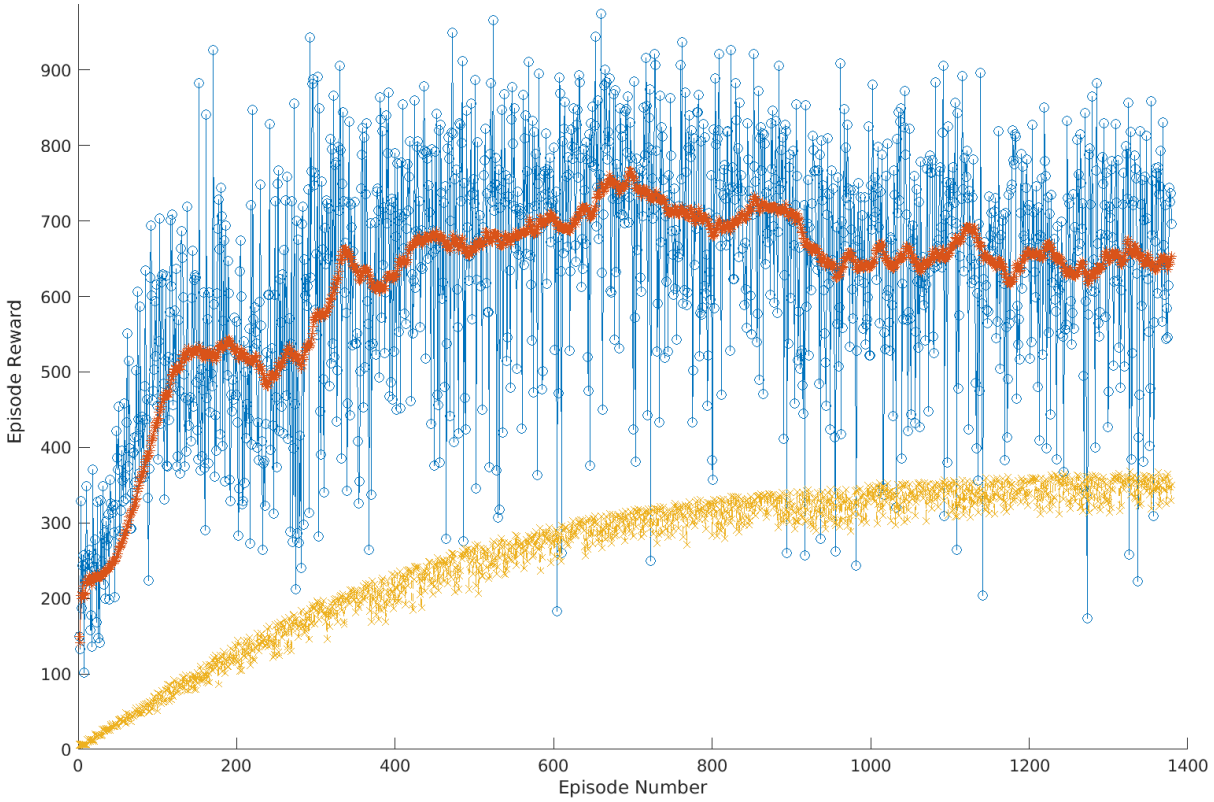


Figure 20 Training progress graph. Blue: reward of each training episode; red: average 50 training episodes; yellow: Q0 value.

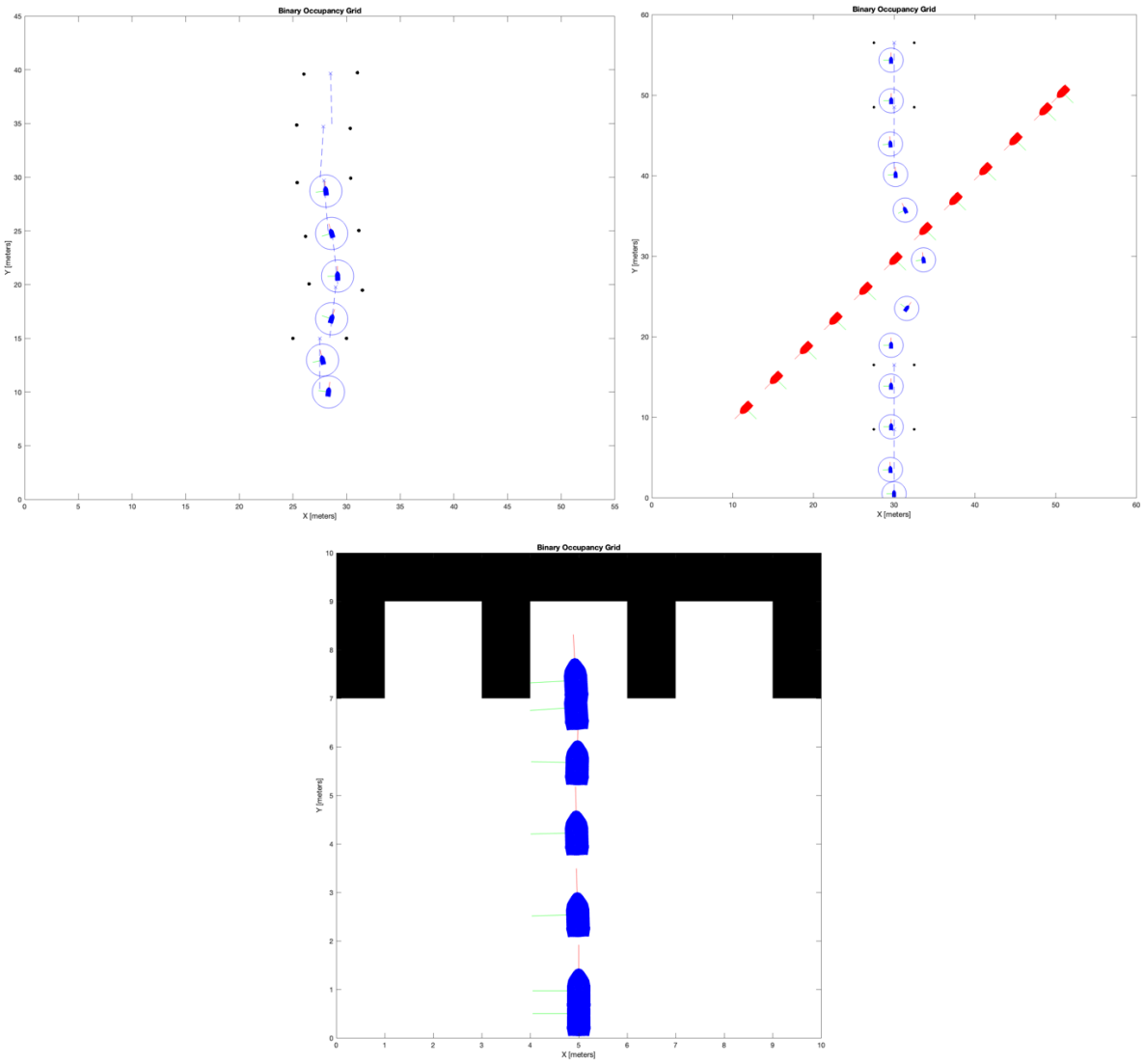


Figure 21 Verification of the trained agents in each task.

2.7.3. PID Control

A Proportional Integral and Derivative (PID) controller is a popular linear feedback controller applied in many processes. In parallel to the implemented deep learning algorithm, a PID LOS waypoint tracking controller was implemented. The implementation follows the information “Handbook of marine craft hydrodynamics and motion control.” published in 2011 by Professor Fossen. Required controller tuning was supported by a Simulink Model also shared online by Professor Fossen. Manual fine tuning was done during the integrated system testing.

2.7.4. Sensor fusion

The machine vision system consisted of a YOLOv7 (You Only Look Once) model that was trained with the provided pictures of the competition. At first, the model was implemented with its standard version, but the computational power available was found to be not sufficient. Therefore, a decision was made to proceed to try to train the tiny version, which is a reduced architecture with faster inference times. Even though this solution improved inference times, it was not fast enough. Therefore, the model was deployed using TensorRT in Nvidia Jetson, taking advantage of the Nvidia board and the improved inference times. The YOLOv7 architecture provides the centre of the detected object in the image as X,Y coordinates and the height and length of the bounding box. To calculate the angle, the following algorithm was followed:

1. Store the field of view (FOV) and resolution of the image (resolution_x, resolution_y).
2. Calculate the pixel in the X, Y coordinate that represents the centre of the detected object. To do so, multiply the X value provided by the detection software by resolution_x.
3. Calculate the distance between the pixel and the centre pixel in pixel units.
4. Obtain the object horizontal angle by multiplying the distance calculated in 3. by FOV/resolution_x.

This method does not consider the lens distortion. The information provided by YOLOv7 was then completed with the angle and distance found by lidar. First, one scan of the surrounding was captured. Then, the point cloud was cropped with a 3D box, to delete all the points exceeding the specified range. Further processing involves clustering points with a DBSCAN, after which objects' positions are calculated. This was done by taking the arithmetic mean of x and y coordinates (i.e., horizontal position only) of each point in the cluster. Depending on the needs, cartesian coordinates can be easily converted into polar: angle and distance. Parameters for DBSCAN were determined empirically after analysing previously captured point clouds. For each object detected by YOLOv7 corresponding lidar object was found. Lidar object with the smallest angle difference to the angle calculated in 3.3.3. was declared to be the same object, and the fusion was completed.

3. Testing

3.1. Hardware Testing

All hardware tests were conducted within the Kelvin Hydrodynamics Laboratory. The hull and platform were thoroughly tested for general watertightness, stability and safety. A ballasting test confirmed the maximal payload weight. Strongly supported by the modularity of the system fine adjustments to the ballasting could be made and components were placed symmetrically on the platform. The boat trim was tested and adjusted via the modular rail system. Early remote-control tests confirmed the dynamic stability of the boat as well as its high manoeuvrability, including on the spot turning. A Bollard tests was conducted to measure the maximum forward and backward thrust and relating input PMW signal to thrust output.

3.2. Software test

Taking advantage of the modularity of the code, the software was tested module after module in a systematic way. Each module has a high level of cohesion and low coupling. Therefore, while the system was being developed, some other parts of it were being tested. Finally, integration tests were performed to ensure the proper interaction between modules. The unit tests consisted primary in using multiple inputs to each function. Then, the output values were validated by applying logical constraints. The integrated tests were performed by creating requests to each module from the coordinator and sending data from module to module. The Software system was tested in a “dry test” using images of buoys and actual buoys and moving the camera and lidar in front the of ASV, as shown in Figure 22.

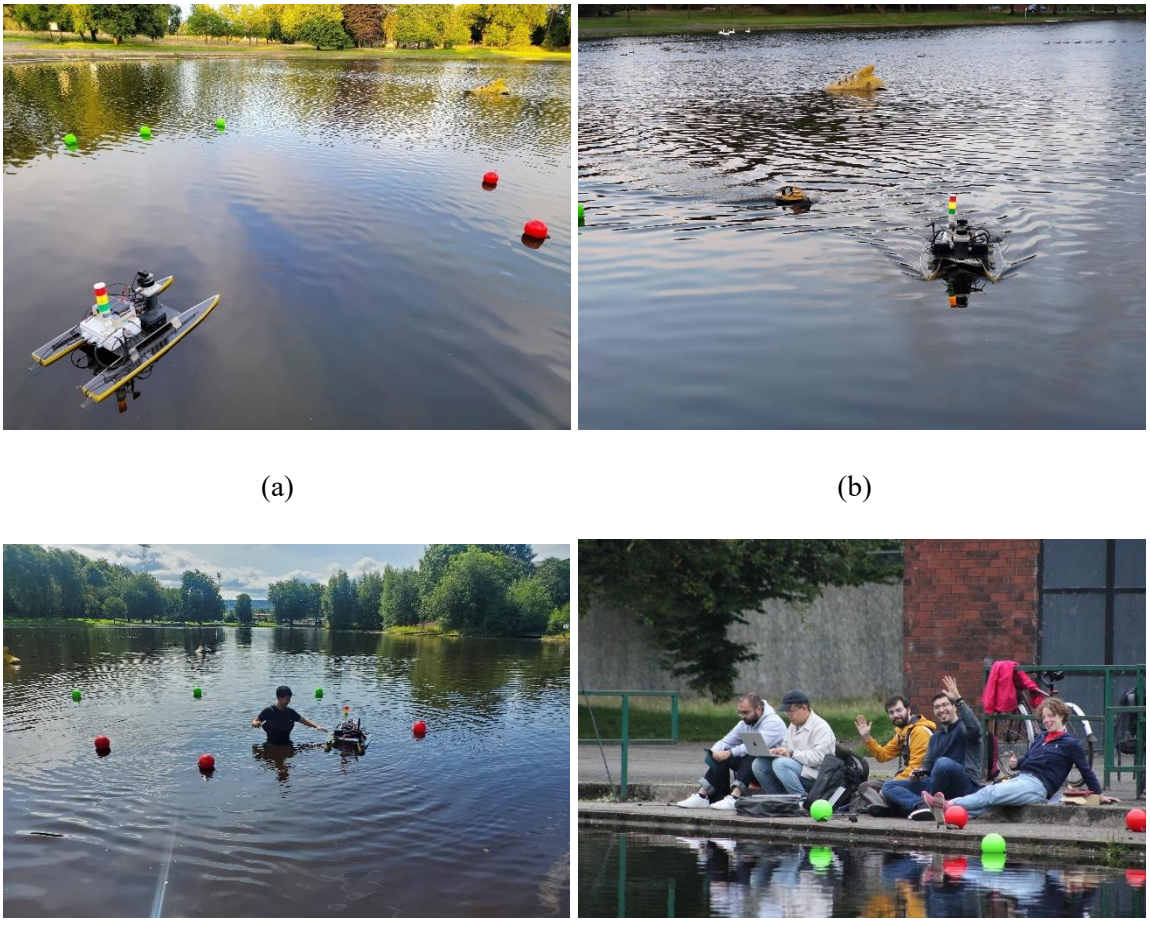
(a) (b)

(c) (d)

Figure 22 Testing sensor collection and correct data processing and control command (a) buoys image in front camera (b) students moving the boat checking dynamic data collection (c) camera captured image (d) 3D lidar point cloud.

3.3. Integrated System testing

Full system tests were done at a pond close to the University. Tests were organised to systematically check the boats performance within an outdoor environment. First tests were conducted to assess the wireless connection distance and controls from shore, battery life and other basic hardware functions. Following, individual sensor data collection quality in the open environment were checked and optimised. Within the available budget, elements and scenario of a parkour similar to the expected Njord parkour which included the creation of gates using red and green buoys as well as using a remote controlled second boat as an obstacle. Photos of these tests are shown in Figure 23.



(a) (b)

(c) (d)

Figure 23 Outdoor tests on (a) buoy gate passing (b) obstacle avoidance (c) recovering boat during testing (d) students debugging at the pond.

4. Acknowledgement

Financial Support:

The team received financial support from the Strathclyde Alumni Fund and NAOME NAOME and EU's Horizon 2020 AUTOSHIP project (agreement No 815012) to cover travel and hardware costs.

The team would like to express special thanks to:

1. Kelvin Hydrodynamics Laboratory Team
2. Dr Zhiming Yuan
3. Dr Byongug Jeong
4. Magnus Macintosh
5. Alexander Brown
6. Eleanor Buxton
7. Hongbo Hou
8. Morven Pennie
9. Yi Huang

The team would like to thank the Strathclyde University Alumni Fund for generous financial contribution to the team enabling us to build the ASV and to travel to Norway. Also, the team would like to acknowledge the sponsorship of ARCHIE-WeSt for providing HPC core hours for the training of DRL agents, Holybro for providing a discount on the RF modules, DIGIFLEC for providing a discount on the Ouster LiDAR and Full Depth for providing a discount on the thrusters.

5. APPENDIX

5.1. Additional Electronic Drawings

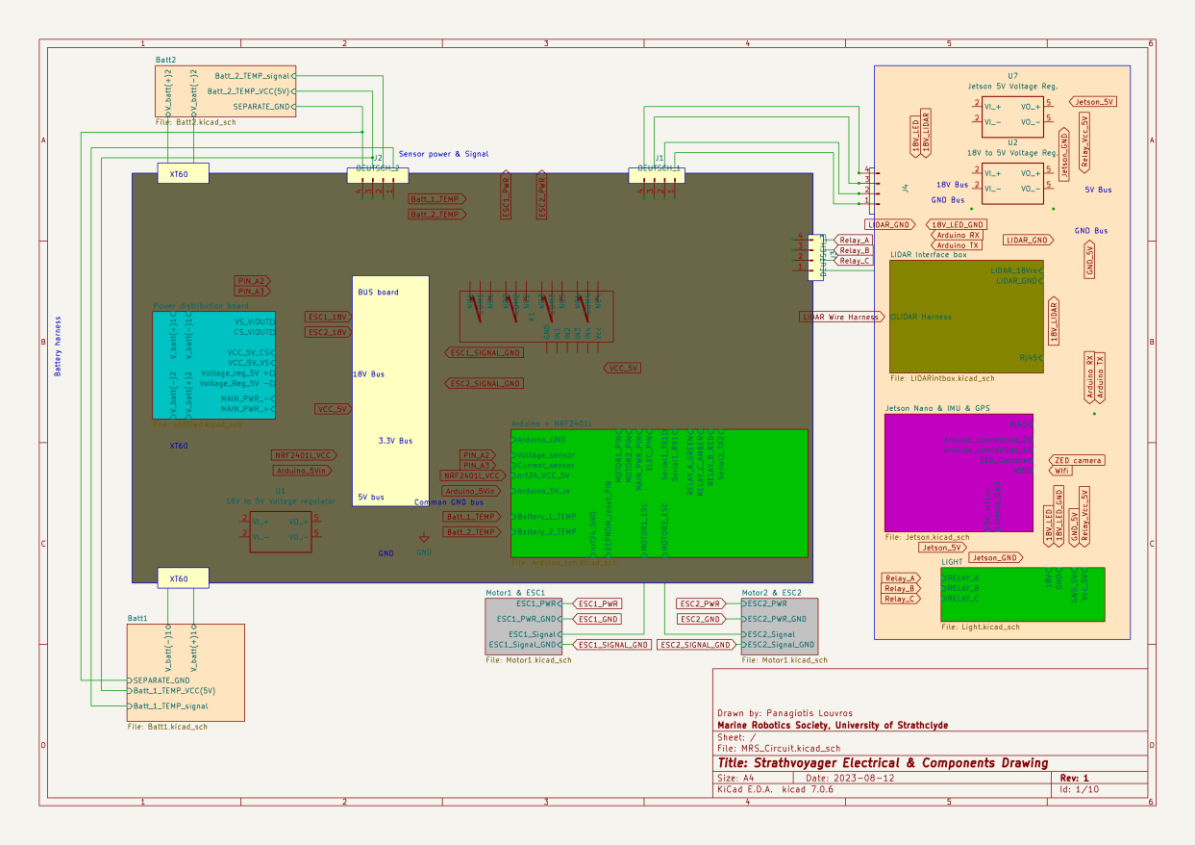


Figure 24 Root hierarchical sheet.

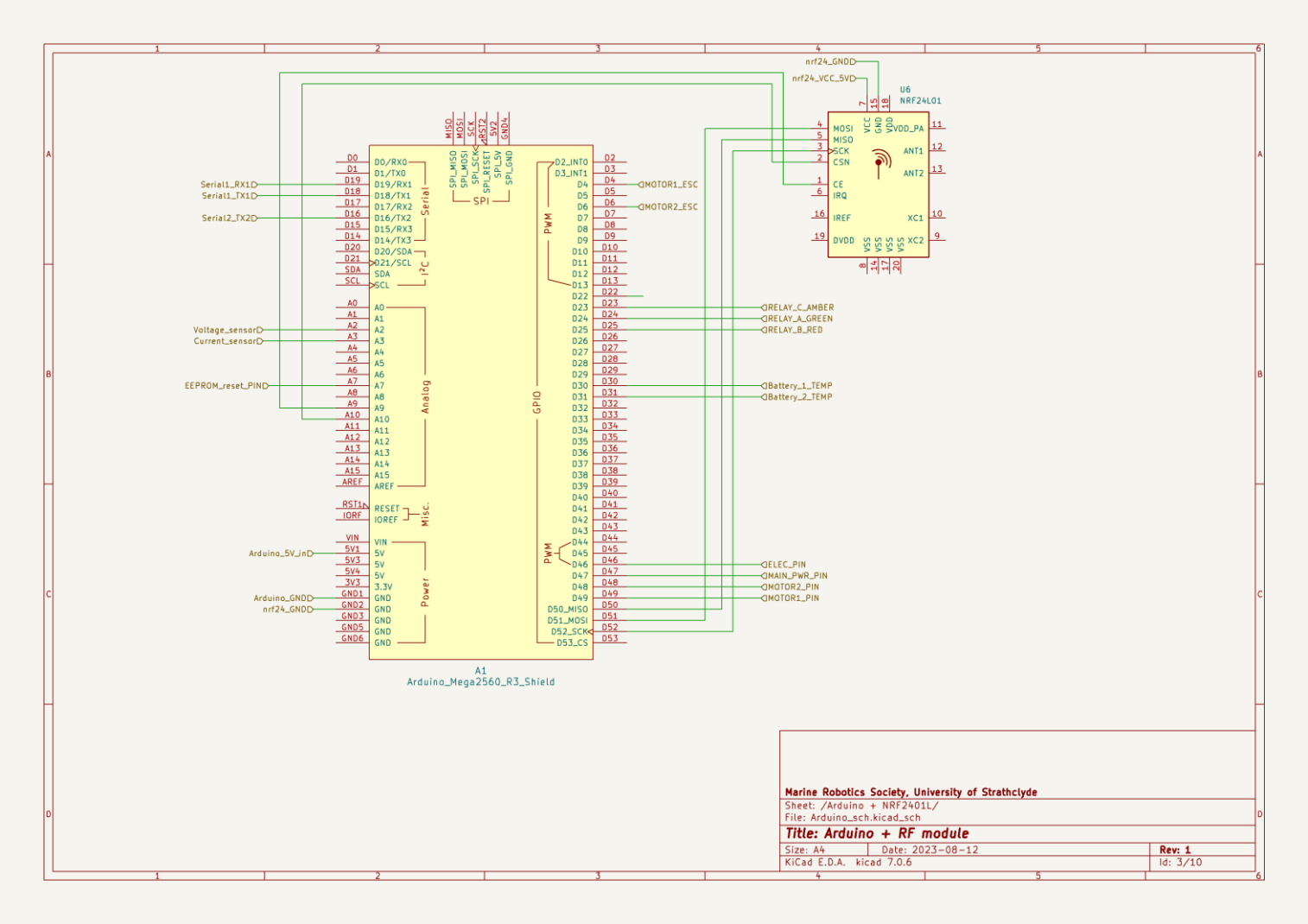


Figure 25 Arduino + RF module circuit diagram.

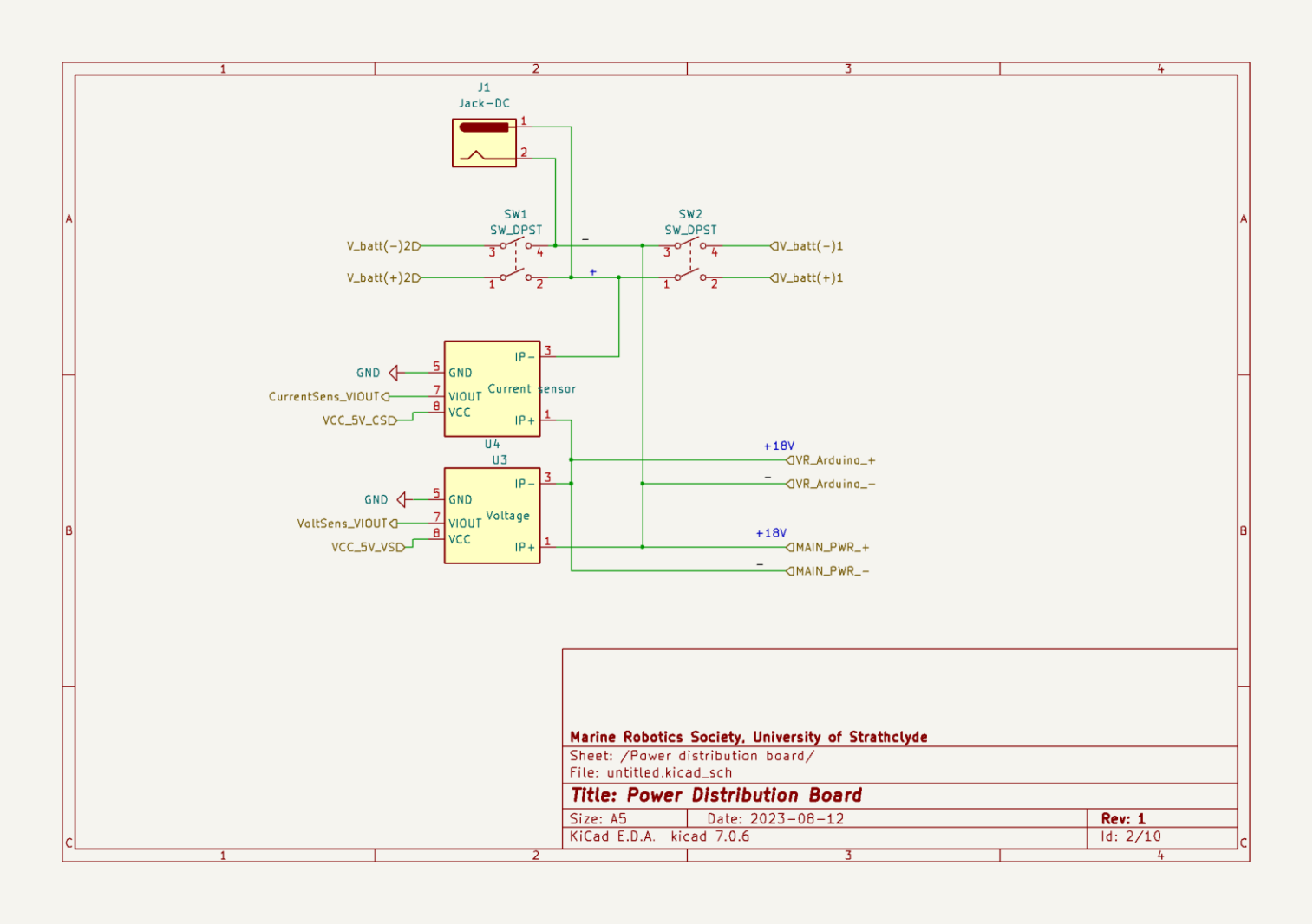


Figure 26 Power distribution board circuit diagram.

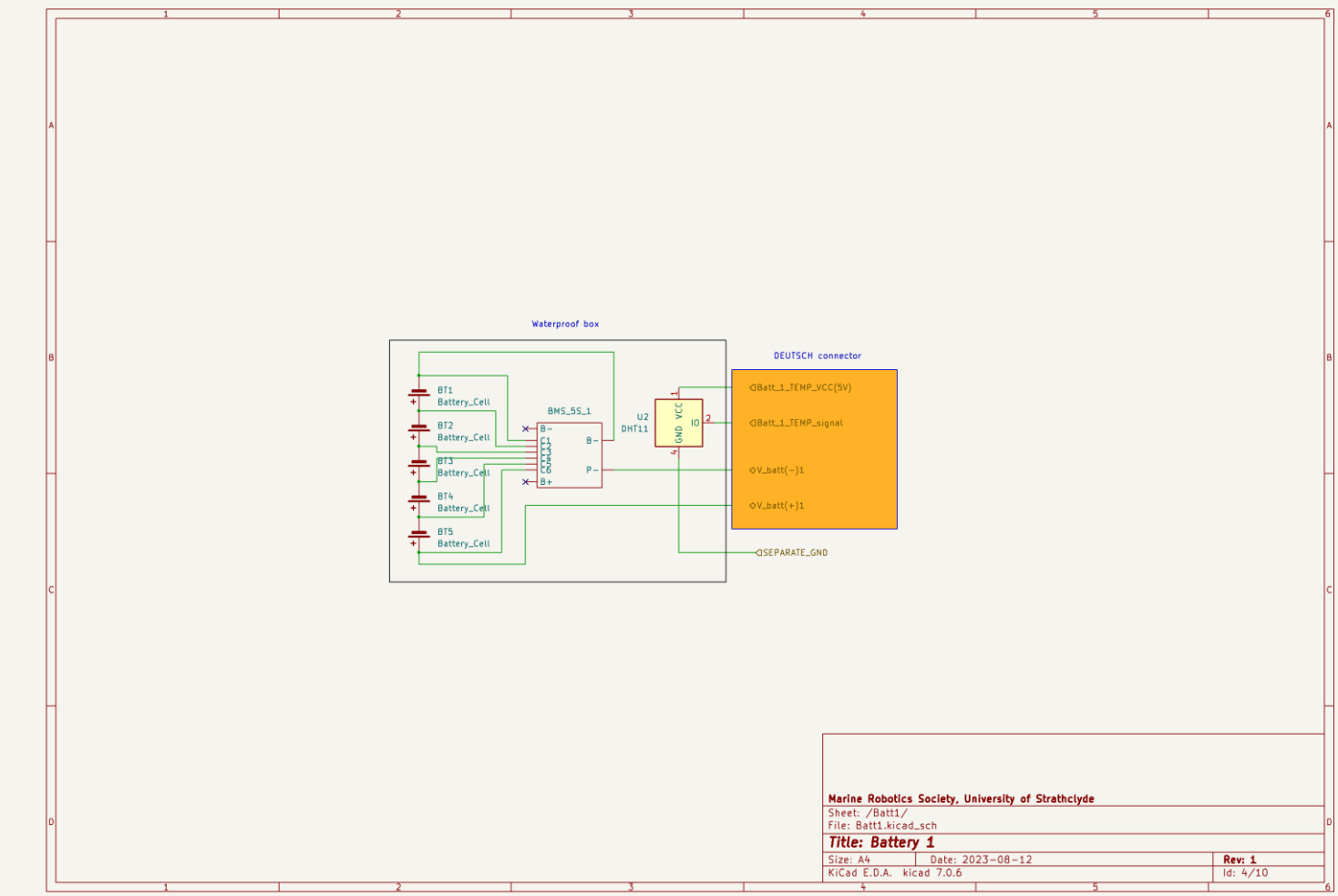


Figure 27 battery 1 circuit diagram.

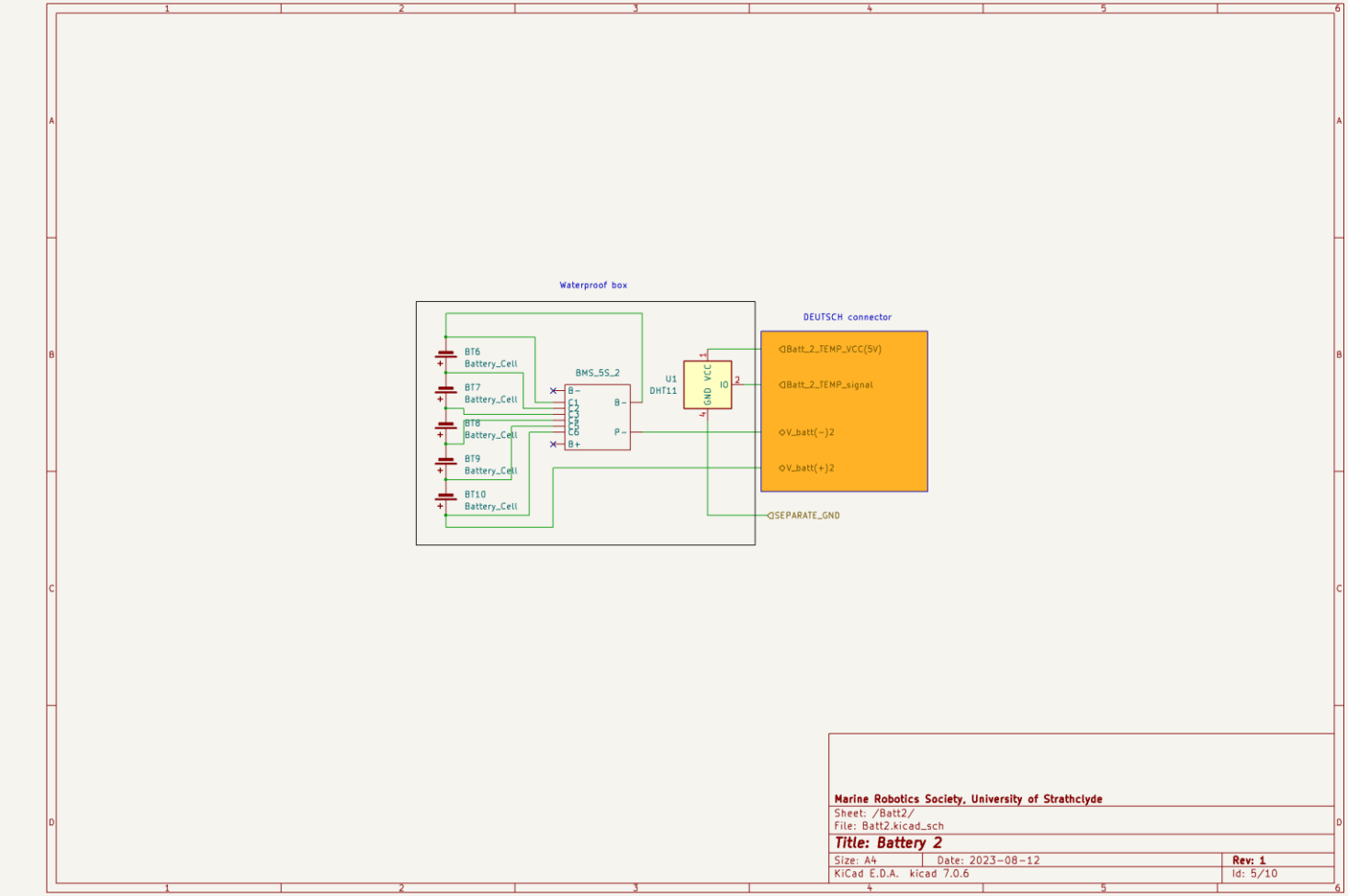


Figure 28 battery 2 circuit diagram.

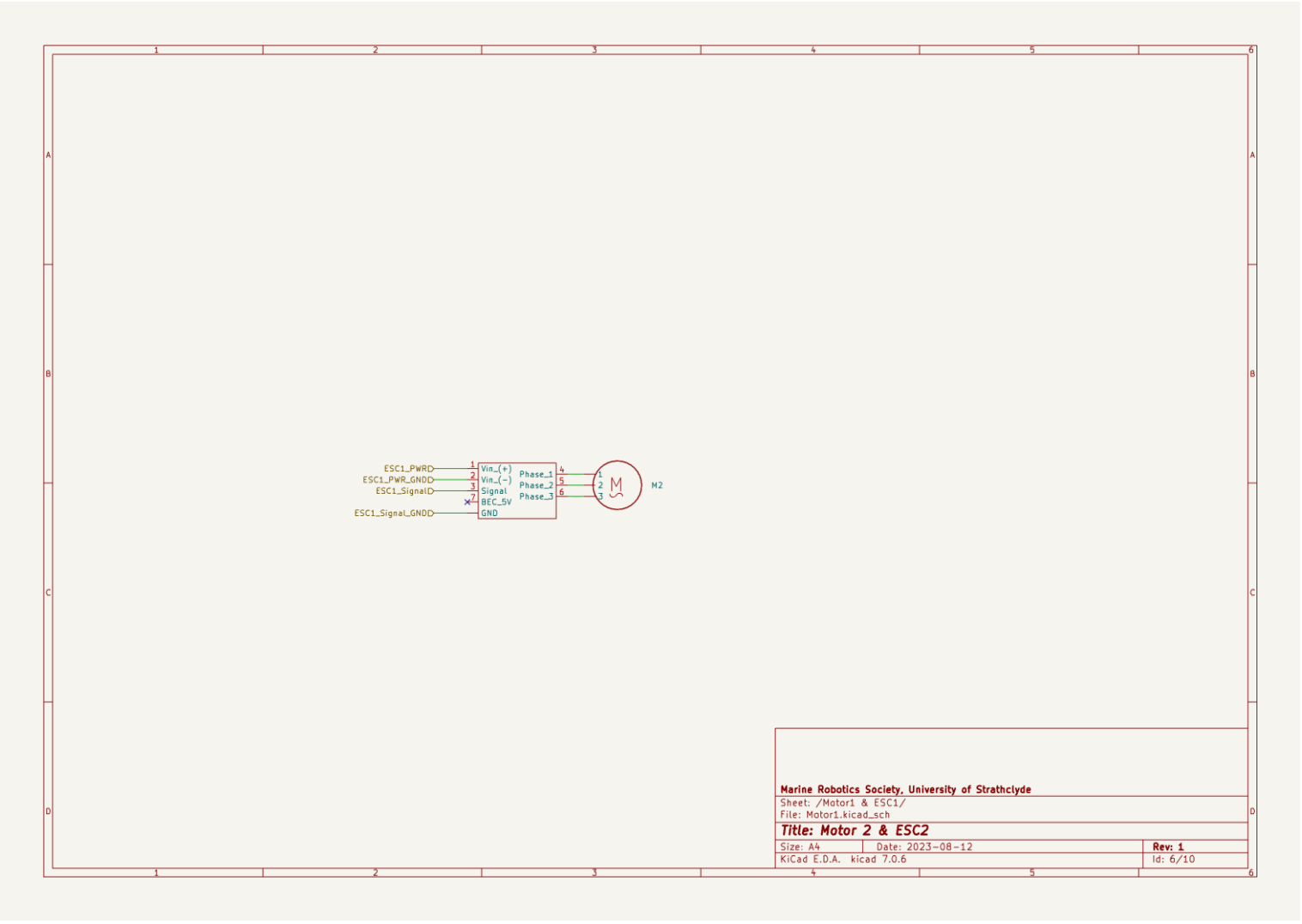


Figure 29 motor 1 & ESC 1 circuit diagram.

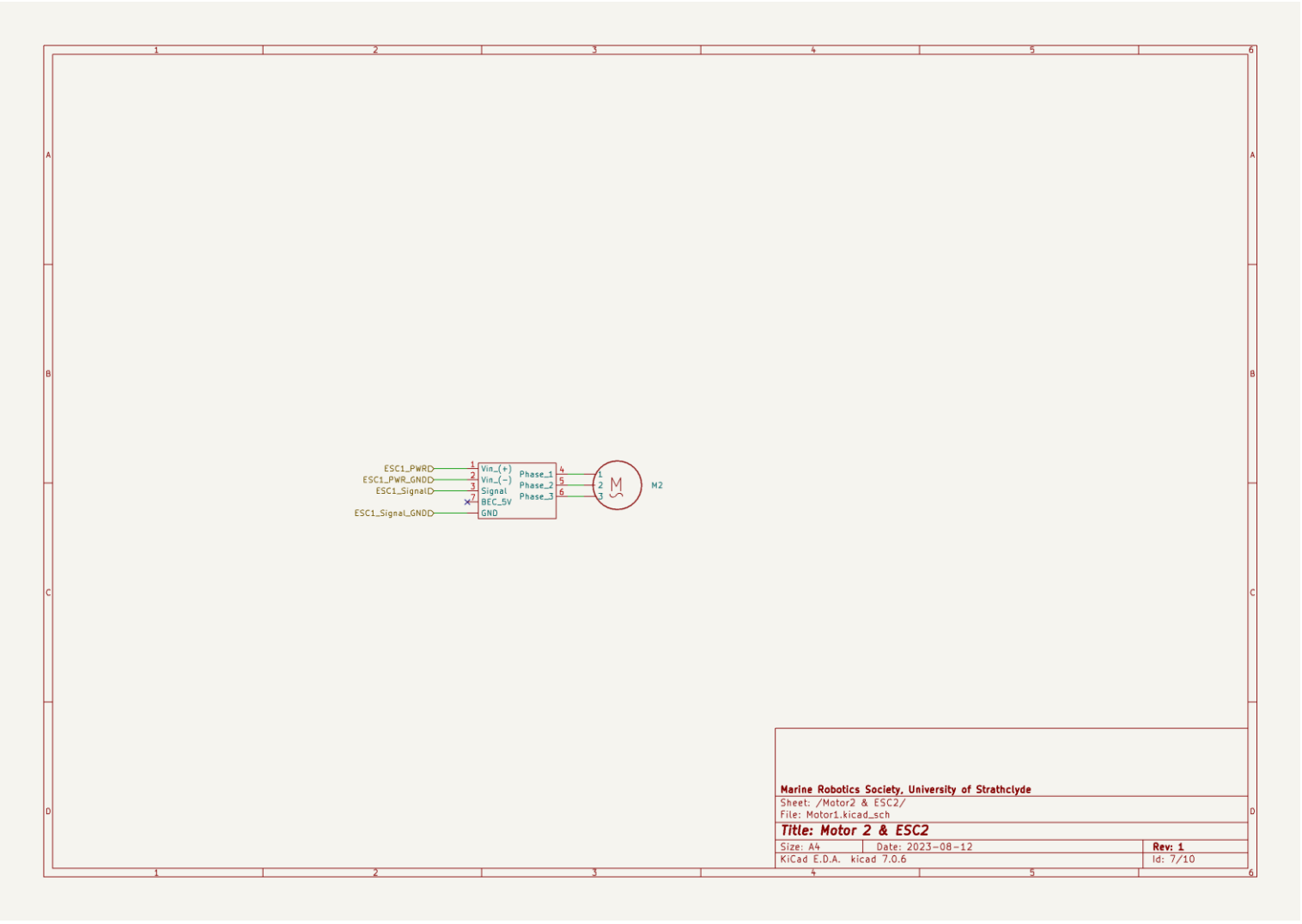


Figure 30 Motor 2 & ESC 2 circuit diagram.

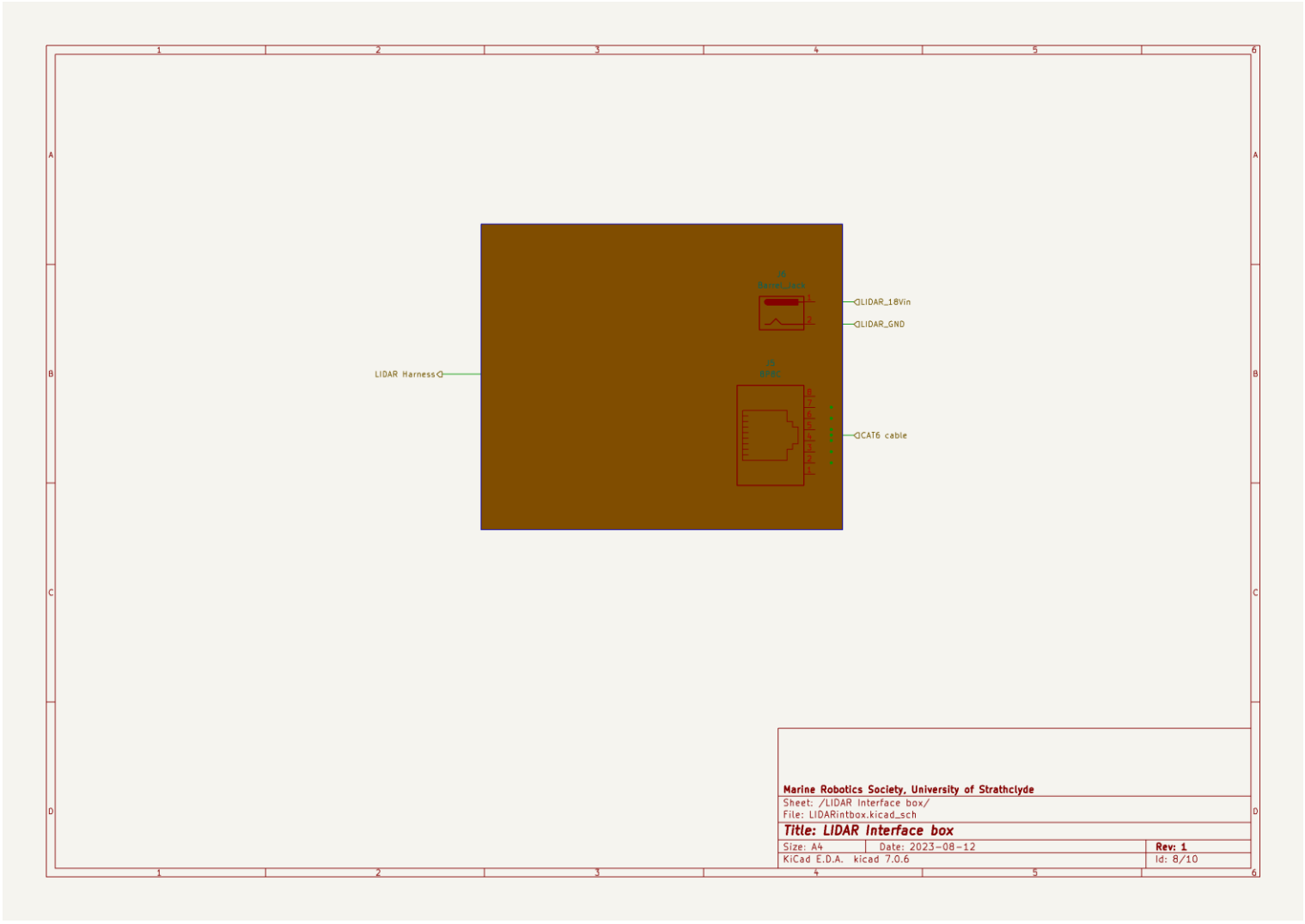


Figure 31 Lidar Interface box circuit diagram.

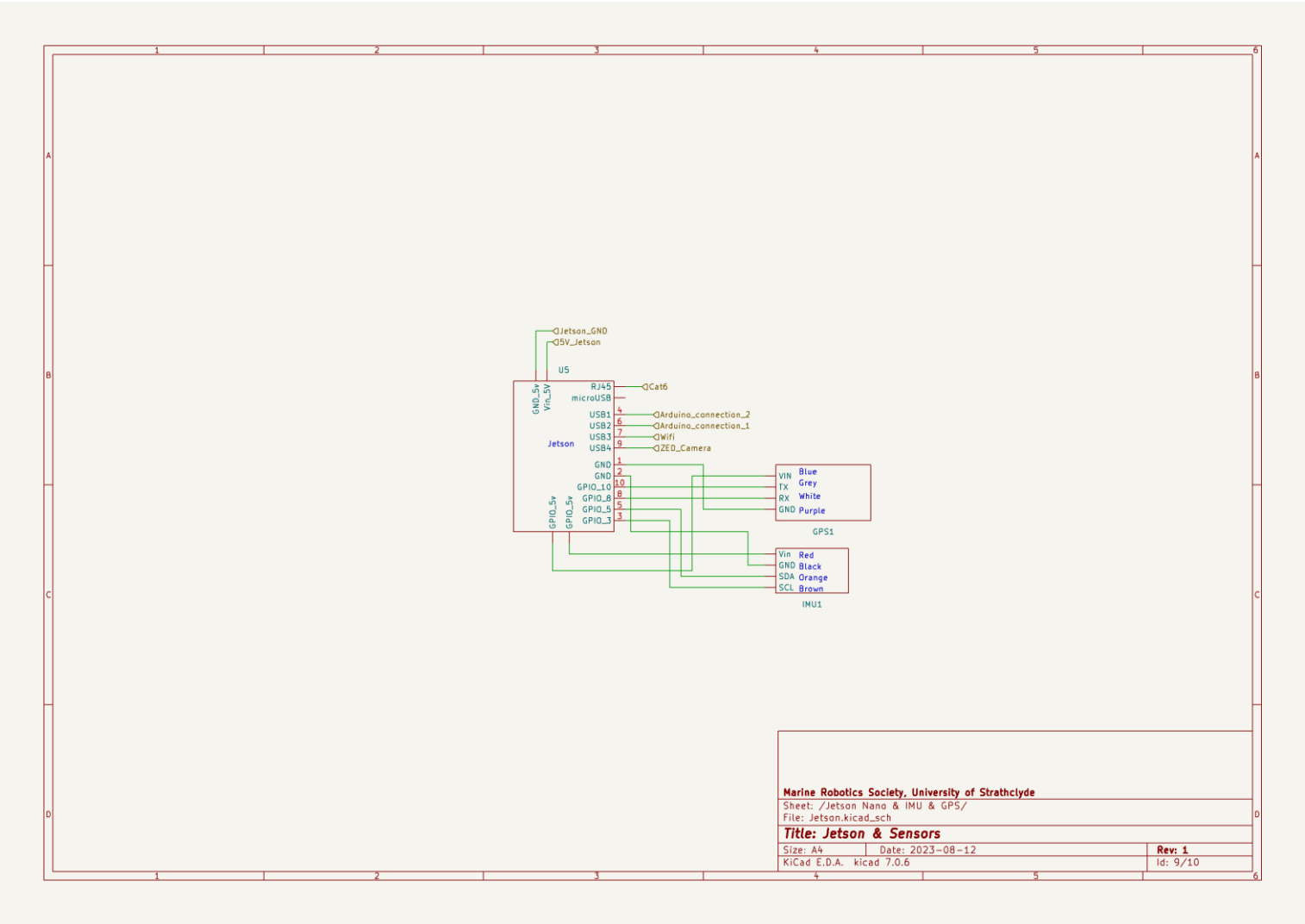


Figure 32 Jetson & sensors circuit diagram.

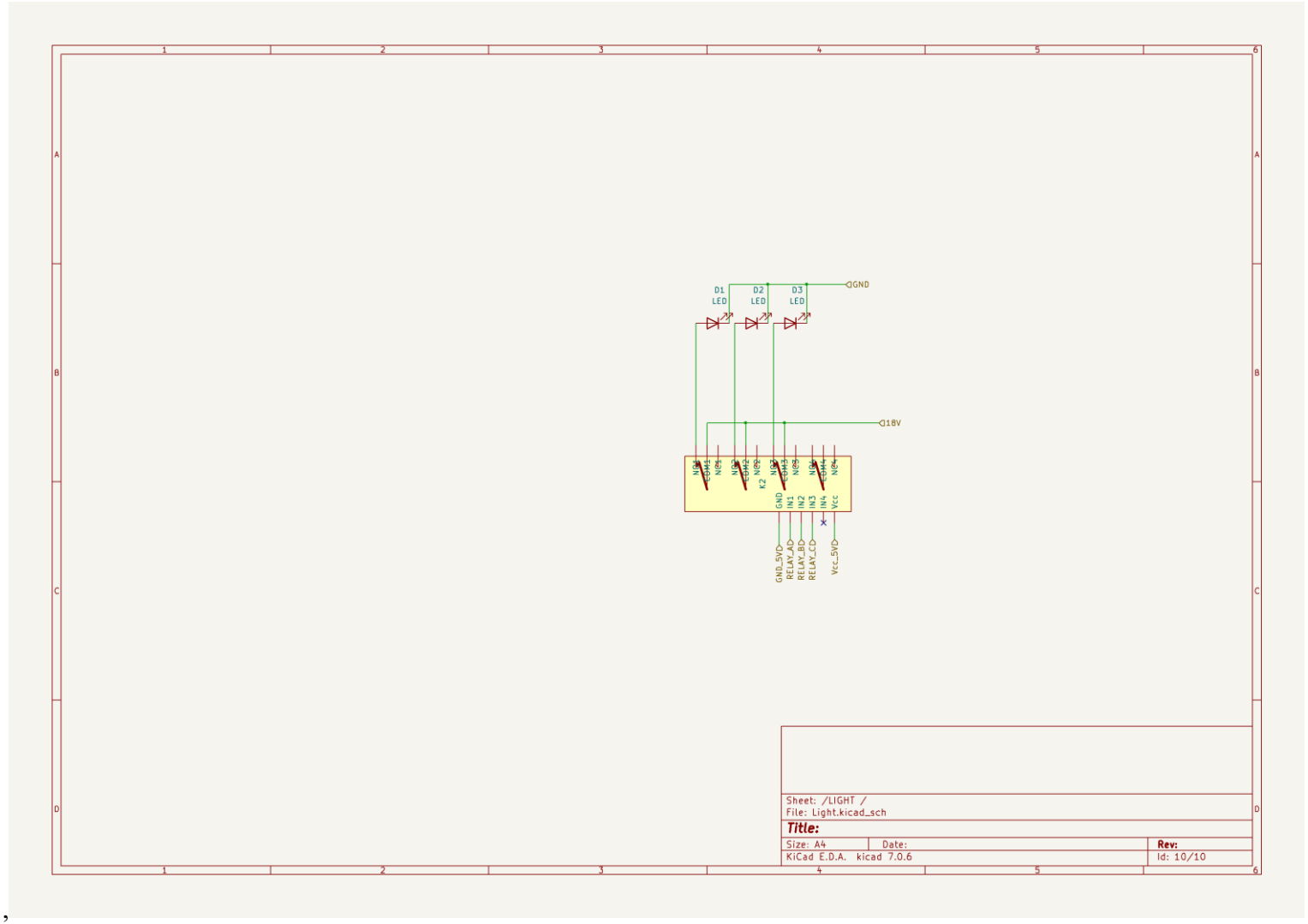


Figure 33 Lighting system circuit diagram.