Meccatronica Design Report 2025

Gianluca Grinovero

July 29, 2025

1 Electronic and data acquisition

This year, the work on the mechatronic system was focused on improving the reliability and deployment of the system. The communication network has remained the same, while there has been an upgrade in the construction of the elements themselves. In addition, the power system went through different designs in order to accommodate an electrical actuation of the main foil. This year's objective is to have a completely electronic control system of the flight height of the main foil. In addition, with the data taken so far on the boat we aim to introduce a reliable state estimation technique to study the real world dynamics of the boat achieved with the design created last year, analyzing the performances of the new vessel with respect to it. Regarding the development of the control strategy, we first aim to reproduce the same control as the one achieved with the wand without the physical connection to the main foil (Fly-By-Wire) and then improve, through smarter control strategies, the actuation of the lateral dynamics to achieve the set height as fast as possible and stabilizing the boat in the air.

1.1 Overall Hardware

Last year project gave a solid base to improve. The creation of a reliable communication network was successful, allowing the system to be upgraded by adding new devices (Table 1) without the need to redesign the network.

	Old system	**Added devices**
Sensors	• Wand angle sensor • IMU • GPS	• Ultrasonic height sensor • Ultrasonic wind sensor
Actuators	–	• Servo motor (main foil)
Computer	• Raspberry Pi 4	• Fixed station computer • 2 LoRa modules (communication)
Power	• 21V Lithium battery (network) • Powerbank (computer)	• 11V Li-ion battery (computer) • 11V Li-ion battery (servo actuation)

Table 1: Hardware update in the system

1.2 Overall software

The main head of the mechatronics lies in the main-computer of the boat. This is composed of a Raspberry pi 4 with a linux operating system running ROS2. ROS stands for Robot Operating System which is an open source framework to maintain code easy to understand, modular and reusable in different scenarios. Where interchangeable hardware needs to be interfaced with always evolving software.

This allowed us to divide the software in packages maintained by different people and designed to solve a particular problem. The main software is composed of:

- a custom package to communicate with the bus and parse incoming data
- custom packages to preprocess the data coming from the sensors
- a custom package to handle different control strategies
- custom packages to handle communication with outside devices
- custom packages to visualize the data in real time

This framework enable the use of existing ready-made packages for different scenarios, and, if required, modify them for the specific case. An example of a ready-made package used is the robot-localization toolbox, providing programs to fuse different sensor readings (discussed more in the following sections) and the foxglove-bridge package for visualization, allowing us to see and plot the state of the boat in real time through a simple WI-FI connection.

1.3 Sensors and PCB design

Until now all electronics onboard was in the development phase, but in the final stage a printed circuit board was designed for each device to ensure the best reliability possible. As an additional benefit they are also volumetrically smaller making for easy integration in the already tight spaces inside the hull. After experimenting with various circuit designs and components on multiple breadboards, we reached an ideal system, which we used as a starting point for the creation of our printed circuit boards.

The software we used was Altium Designer, which allowed us, using its cloud-based infrastructure, to divide the workload between multiple people, streamlining the process of the creation of the PCBs.

Regarding the code running on each board. In addition to the programs necessary to read the raw and processed data coming from the sensors, they need to be plugged with the dedicated software needed for the transmission in accordance with the NMEA2000 standards [?], that was re-written for STM32 microcontrollers.

Figure 1: PCB render

1.4 Ultrasonic ride height sensor

To improve last year's estimate of the ride-height of the boat, a new type of sensor is used to allow redundancy and improve accuracy. 2 ultrasonic sensor (DYP-A02 model) are mounted at the end of the bowsprit at the same distance from the hull allowing for an accurate measure even for high roll angles of the hull. Measuring the distance between the sensor and the water through the time of propagation of an ultrasonic wave.

Some problems arisen during testing, such as unstable sensor reading frequency and presence of noise and outliers in the data. The first was fixed with the use of an additional sensor, allowing for redundancy in the incoming data making the system robust to loss of sensor readings. The latter was tackled through the use of filter on the incoming data, more precisely an Hampel Filter. The Hampel filter consists of a moving window, where the central element is considered an outlier if it deviates from the median by a fixed number of standard deviations. In that case the outlier is substituted by the value of the median, which unlike the mean, is an estimation of the sensor reading robust to outliers.

Both the size of the moving window and the number of standard deviations have been tuned after some experiments.

More formally, if $x_1, x_2, \dots, x_N$ are the heights registered in the moving window, then the Median Average Deviation (MAD) is

$$MAD = Median\{|x_1 - Median|, |x_2 - Median|, \dots, |x_n - Median|\} \tag{1}$$

and the standard deviation is defined as

$$\sigma = 1.4826 \times MAD \tag{2}$$

Therefore, the central x_C point is replaced with the median if:

$$|x_C - median| > \sigma \tag{3}$$

1.5 Control Algorithm

The control strategy for the main foil is developed in two sequences:

The first control strategy that was designed was meant to reproduce the dynamics of the well known mechanical system that the boat is equipped with. This allows to study the performances achievable with the manual system and to help the helm get used to the new hardware. The next step was to use a more sophisticated algorithm to have a better estimate of the state of the boat. This estimate is achieved by fusing different sensor readings (using Kalman filter, discussed in the next section), making the system always more robust to sensor faults and existing disturbances.

The control algorithm is then designed setting a height reference the boat should aim at at all times, commanding the flap angle to achieve the goal as fast as possible and maintain the boat in the air. This algorithm make use of the PID architecture which is an easy and reliable method that doesn't require a detailed model of the dynamics of the boat. It is composed of three components: proportional, integrative, derivative. The proportional component adjusts the output to match the current error between the estimated height of the boat and the reference; the integrative component accumulates past errors to eliminate steady-state offset; and the derivative component predicts future error trends by evaluating its rate of change. (Fig: 2).

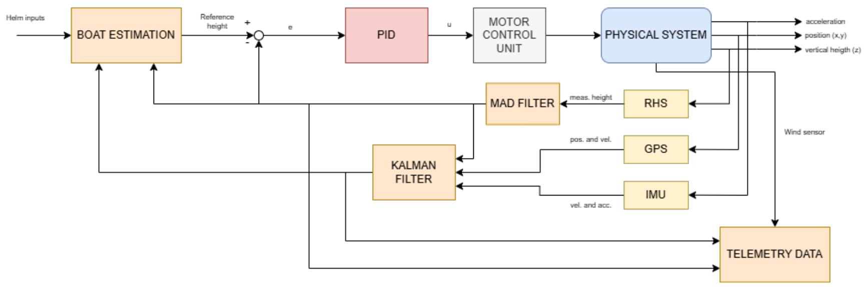

Figure 2: PID control

The height reference and control parameters can be changed real time from the support boat, just by connecting to the onboard computer through WI-FI.

For future development, a different control algorithm could be developed. This would improve the resilience to errors in the measuring system or give the boat a faster and more precise transient during manoeuvrers and take-off. A possible example would be to us a *LQ* control. Given the fact that is closely related to the Kalman filter structure it could be implemented in a short period of time. However, the downside, is that any error in the mathematical model would make more unreliable that the simpler structure that was adopted.

1.6 Boat state estimation

To optimally control the dynamics, as described in the previous chapter, and to assess the overall performance of the boat, we employ various sensors to monitor its movement in space. However, since each sensor measures the boat's state differently, we fuse their measurements to obtain the real-time estimate of the boat's position and velocity. To achieve this, we use Kalman filters. More specifically, an extended Kalman filter (EKF) that fuses data from multiple sensors without requiring an exact model of the boat's dynamics.

The overall estimation process utilizes two filters: one for local measurements and another for global measurements. The local filter fuses estimates derived from the wand angle, ultrasonic height sensor, and IMU to accurately determine the boat's roll, pitch, and height. This filter is particularly important, as its output is fundamental for controlling. The global filter fuses readings from the GPS and IMU to derive the boat's overall trajectory and speed. It is designed to provide real-time position estimates by compensating for the slower GPS updates through integration with accelerometer data and orientation readings. Note that the GPS provides readings in UTM (Universal Transverse Mercator) coordinates along with latitude and longitude, which must be converted into the world frame used for the overall state estimation.

This estimation framework yields valuable performance data, including an assessment of the boat's leeway, a critical parameter during foiling, and facilitates a more accurate interpretation of wind sensor readings by distinguishing between true wind and apparent wind.

Overall, the integration of these components is schematically illustrated as follows (Fig: 3).

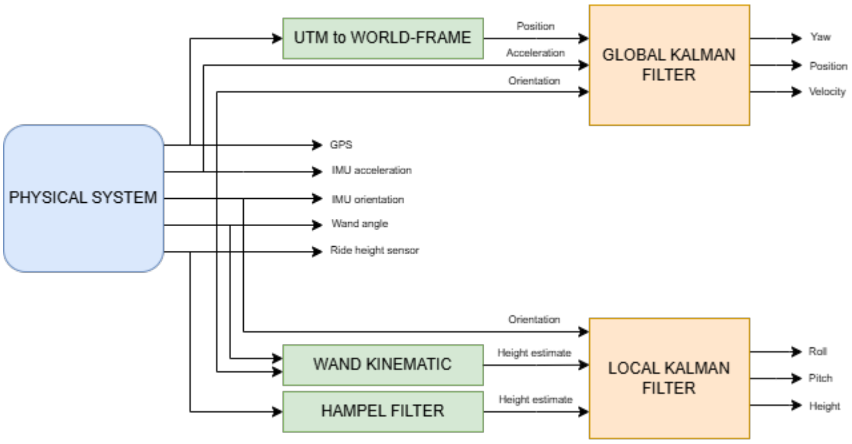

Figure 3: Kalman filter schematics

All of this is achieved through custom made programs and through the use of the open source robot-localization package in ROS2. With this package 2 extended Kalman filters can be setup to achieve the estimates described and then everything can be visualized real time on the computer like shown in figure 4

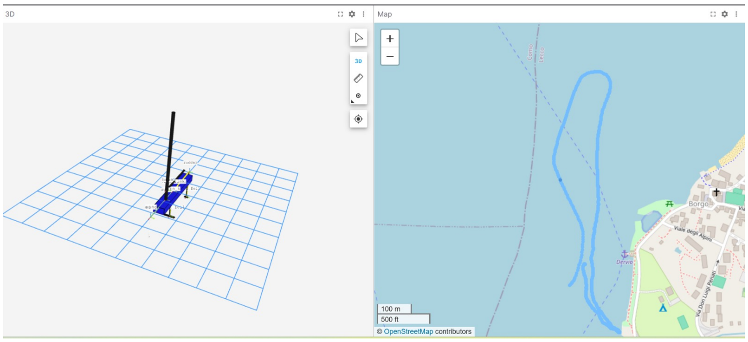

Figure 4: GPS map

1.7 Communication with the bus

Getting the data from numerous sensors to a computer and allowing the computer to send data as well to the same network is not as straightforward as it might seem, and since the interface to communicate with the bus was developed by us, it was a problem we needed to solve.

The use of the same port to receive and send data from the bus to the main computer requires the use of either a synchronous but time-consuming communication or a asynchronous but fragile way of sending data packets between the computer and bus interface.

The solution might be trivial but the hardware used posed a challenge. The synchronous solution needed the implementation of more complicated communication protocol that was not trivial to interface between a ST microcontroller and a linux computer.

Although the asynchronous solution would have been the easiest to implement, the implementation of a multiple-port solution was not possible on the current computer without sacrificing other functionalities. So a single port asynchronous solution is proposed with the use of a packet identification protocol called COBS.

Consistent Overhead Byte Stuffing (COBS) (Fig: 5) is an algorithm for encoding data bytes that results in efficient, reliable, unambiguous packet framing regardless of packet content, thus making it easy for receiving applications to recover from malformed packets. It employs a particular byte value, typically zero, to serve as a packet delimiter (a special value that indicates the boundary between packets). When zero is used as a delimiter, the algorithm replaces each zero data byte with a non-zero value (typically the position of the next supposed zero value) so that no zero data bytes will appear in the packet and thus be misinterpreted as packet boundaries.

This allows the computer and the micro-controller to interpret packages correctly without the need of synchronization.

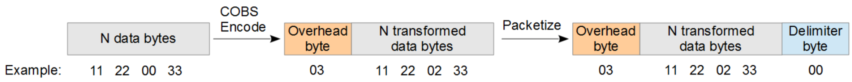

Figure 5: COBS

1.8 Ultrasonic wind sensor

The wind sensor is produced by LCJ Capture and uses a combination of an emitter and receiver of ultrasonic waves to measure the intensity and direction of the wind. For a clean sampling, the mounting point chosen is the forward beam of the spreaders. Given its hard to reach position a different approach has been chosen to deliver the data to the communication network. The sensor will be connected to a board with Bluetooth capability, an identical board will be present on board, and will be connected to the communication network. So the data package will firstly be transmitted via Bluetooth and then will enter the main system via the same network for the computer to analyze. Given its precarious attachment it will be difficult to use the data for any control, but its main utility will be for telemetry and training scenarios.

1.9 Remote monitoring

The remote monitoring system enables real-time telemetry of a ROS2-based platform via a long-range, low-power LoRa link. An onboard computer with an EBYTE LoRa E220 module (868 MHz) transmits structured telemetry data at 1 Hz, using a Python node that subscribes to key ROS2 topics. On the ground, a Raspberry Pi with a similar LoRa module receives and decodes the data, allowing identical algorithms to run remotely. This setup ensures reliable monitoring in areas lacking conventional network infrastructure, such as lakes.

2 Innovative Elements

A central innovation is the adoption of a fully parametric design workflow for both the hull and the appendages. This approach allowed the team to iterate quickly across a wide range of geometric configurations, leveraging genetic algorithms to optimize foil sections and placements for improved hydrodynamic efficiency. The integration of simulation-based tools, such as CFD for hull drag estimation and structural solvers for load validation, enabled a multi-objective optimization process that tightly coupled performance and feasibility. As a result, the final geometry reflects not only performance gains but also manufacturability and material efficiency.

In parallel, the team pursued strategic core material reduction in both the hull and the gantry. By analysing load paths and stress distribution, localized areas of high and low stress were identified, allowing for selective removal of core material without compromising structural integrity. This optimization contributed to a measurable reduction in weight and resin usage, aligning with the team's broader sustainability goals.

The spreaders underwent a topological optimization process, which allowed for a redesigned geometry that retains stiffness while eliminating excess material. Starting from a solid baseline model, constraints and boundary conditions were defined to reflect real-world load cases, leading to a structure that is lighter and more efficient.

Moreover, the development of a fully integrated electronic control system marks a significant advancement in the control mechanisms for these boats. It enables more refined, customizable, and precise control over both the flying and riding experience, offering improved performance and adaptability to different conditions and user preferences.

Together, these innovations represent a step forward in combining performance oriented naval architecture with responsible engineering practices. They illustrate how digital workflows, structural optimization, and sustainable materials can converge into a coherent design philosophy suitable for high-performance sailing and beyond.