S.O.N.I.A. AUV Technical Design Report

François Côté-Raiche, *Team Leader* Camille Sauvain, *Admin Team Leader* Alexandre Leblanc, *Treasurer* Alexandre Lamarre, *Mechanical Team Leader* Francis Alonzo, *Electrical Team Leader* Alexandre Desgagné, *Software Team Leader* Karine Blier, *Electrical Team* Rémi Blier, *Software Team* William Brouillard, *Electrical Team* Onur Cocelli, *Electrical Team* Florent Côté-Normandeau, *Mechanical Team* Martin Gauthier, *Software Team* Olivier Juteau-Desjardins, *Electrical Team* Lucas Mongrain, *Electrical Team* Antoine Pelchat-Fortin, *Software Team* Pierre-Yves Vincent-Tremblay, *Software Team*

***Abstract*—S.O.N.I.A. is a Canadian student club from École de Technologie Supérieure that involves 16 members who dedicate their knowledge to engineer an autonomous underwater vehicle (AUV). The guidelines of the club are innovation, efficiency. For our club, innovation goes particularly through the autonomy of our submarine. This is what makes our project so interesting and what sets us apart from others. The efficiency of the submarine is very important. However, we must not forget the efficiency of the processes. Therefore, in S.O.N.I.A., we place particular emphasis on the accessibility of our research. The club is divided into four departments: business, electrical, mechanical and software, each of which contribute to the achievement of the project. To facilitate the development of our submarine and the testing of our new technologies, we put a lot of effort into our development process. We believe that this strategy will allow us to perform in competitions in a more consistent way over the years. Since last year, team members want to bring both submarines to the competition, first to get the experience of sustaining two submarines in one competition and secondly to give us more training time and run time during the competition.**

***Index Terms*—Autonomous Underwater Vehicle, Technical Design Report, RoboSub, RoboNation.**

I. Competition Strategy

TEAM S.O.N.I.A.participated in the RoboSub competition for 21 years and look forward to compete for several more years. Therefore, our general strategy is based on the durability and the constant progression of our team. We put a lot of effort into the development of sustainable and flexible processes that will allow us to have constant improvement year after year and to avoid falling back.We understand the complexity that we add to certain tasks by using flexible systems. Choosing to use Deep learning as the main image detection technique instead of conventional vision or dead reckoning is a good example of that mentality. We consider that this choice allows more adaptability to the different hazards of the competition and will allow us to be more consistent in our results. Also, we think those complex systems give us better knowledge to carry over for the next competition year.

A. Mechanical Team strategy

This year, the mechanical department is a very good example of the consistency of the results we are looking for. Looking at the rankings of previous competitions, we found that some points were easily obtained by paying particular attention to the weight of our new submarine. We estimated a gain of 188 points which could be decisive in a tie-break situation. An indirect way of increasing consistency is the new dynamic model that allowed us to create a new 6-DOF model based controller and new simulation environment. This will give us a reliable way to test missions before the competition and to prepare for testing day. Finally, during the weight relief process, the department made sure that the maintainability of the platform would be optimal with the newly designed rack system. In short, assured points, reliable testing

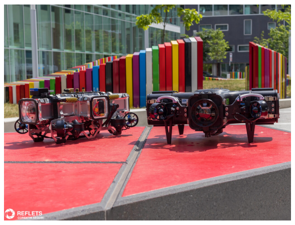


Fig. 1. AUV-8 & AUV-7

platform and easy and fast maintainability is what will give us an edge in the competition from our mechanical team.

B. Electrical Team Strategy

This year, the electrical department had one goal. Doing more with less. The main challenge for our electrical team was the major size reduction of our new submarine. The strategy to create the new platform was to adapt existing working design that proved its worth during the past years and optimize their size to use them in the new submarine. Since S.O.N.I.A. adopted enclosed submarines, we had some issues with our forward maximal speed. The newly improved power management will permit us to improve the overall speed of the submarine which should give us more time to execute tasks during the competition. Also, this newly gain of power coupled with lower inertia gives us the opportunity to get more style points with a rotation around the roll axis. Lastly, one of the most awarding point tasks in the competition is the random pingers which make it an attractive task to complete since it basically assured a spot in the final stage. The electrical department tried to develop a more accurate hydrophone system which will give us greater chances to detect the task with our deep learning system. We think that with the time saved during each movement of the submarine and a better chance of success at big points task, the electrical department will give us all the chance we need in the competition.

C. Software Team Strategy

This year, the software department's strategy was to normalize the development process. Starting with the continuity of the last year objective to dockerize our software platform. The goal was to make the development, the testing, and the learning of our software team as easy and efficient as possible. With our experience from previous years, we know that every task demands some kind of image recognition capabilities and a way to align to a target. Therefore, we put our main effort on making sure we make the control system and the recognition system as flexible and as efficient as we could for our future team members. For the testing of missions and new systems, we have made a new 3D simulation using Unity which will give use a more accurate testing environment when we do not have a pool available. Since activating a torpedo or closing a mechanical arm is basically the same for the software department, we think that focusing on the common core of every task instead of an individual one will give us more ways to adapt to a competition situation and to our recognition capabilities. This flexibility should let us choose the best options to adapt our strategy during the competition.

II. Design Creativity

A. Mechanical

1) AUV8 Outer shell: DINA is the 8th submarine designed by the SONIA team over the last 20 years. It features a single compartment hull with a slightly modified cross shape compared to our previous prototype. The watertight hull is composed with 7-part sealed with BUNA-N O-ring to ensure a seal down to at least 10m depth. The composition of hull is mainly in made of hard anodized aluminum and it also has four customs acrylic cap including one with an integrated dome. The cross shape allows quick access at each end of the vehicle to directly access the problematic component by minimizing the components to be removed. It is also a symmetrical shape which also allows us to keep the center of mass very close to the geometric center.

2) AUV8 Inner shell: In the inner shell, the components are grouped by their functions. This creates a certain electrical workflow to reduce debugging time. Also, our AUV is equipped with a 3D printed clip rack system. This way all the elements are easily secured in place. It gives the possibility to take out all the components without any tools required. The only elements we did not use clips are the DVL and IMU because we need them to hang tight, we the submarine frame. Without these 2 components, assembling and connecting all the electronics takes less than 15 min compared to our previous one which could take up to 2 hours to assemble.

B. Electrical

1) Leak Sensors: Water ingress is one of the main dangers facing underwater vehicles. Leak sensors stand as the last line of defence against water intrusion. Their role is to detect a small quantity of water leaking inside the submarine, allowing the system to abort the mission before the water reaches critical systems. Off the shelve leak sensors already exist (notably BlueRobotics's SOS probes).

However, we want sensors that offers us more flexibility in their physical form and that could be reused after having been tripped (as opposed to the SOS probes that are one use only). For the design of the sensor, we choose to measure the resistance between two electrodes.If the electrode were to meet water, the resistance should be greatly reduced. To measure the resistance, we put the two electrodes at the bottom of a voltage divider. If the resistance between the electrode is very low (i.e., the probe has been tripped) 0 V should be read across the electrode. When the resistance is very high, almost 3.3V should be read across the electrode. The voltage is measured by an operational amplifier in comparator mode to provide the output. We chose to add hysteresis to the comparator by having a positive feedback. This way, the resistance of the two electrodes needs to be higher for the comparator to reset. As a result, the output of the comparator will stable even if the resistance oscillates around the initial setpoint of the comparator.

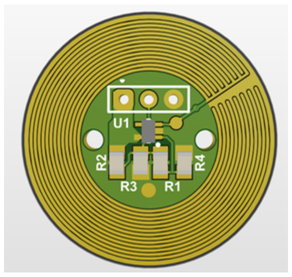


Fig. 2. Leak Sensor

2) Backplane: The first time we assembled our submarine, we found out that the space was not big enough to fit all the cables that were needed. We found that the power management was specially taking a lot of useful space. Since we already had the idea to give more power to our thrusters to give us the possibility to mark style points and travel faster to the different objectives giving us more time to do them, we used this opportunity to redesign the power management system.

The goal of this redesign was to free space in the submarine and to support a max current of 90% of the rated current of our thrusters. We used to have a desktop processor in our submarine that would only use a voltage of 12V. Now, with the Nvidia Xavier, no component in our submarine requires a single voltage input. That allowed us to completely remove the unnecessary converters and at the same time remove 4 printed circuit boards from the submarine. To reduce the cost without compromising the performance, we have chosen to remove the solder mask on the printed circuit board allowing us to add solder on the high current traces to reduce their electrical resistance. We are still using some thicker plating of the layers of the printed circuit board, but the removal of the solder mask gives us a possibility to improve our current ratings.

3) Direct Current to Direct Current: Modern electronic systems often require voltage regulation to work. In the case of SONIA's submarine, the voltage of the batteries (normally around 16 V) need to be decreased to a value more suitable for electronic components, for example 3.3V. In the past, DC-DC converters were integrated directly with the different PCBs, meaning that a module had to be redesigned for every project. With the addition of a FPGA (which requires multiple voltage to operate) the team decided to take a different approach by using board-mounted DC-DC converters. We originally intended to use off-the-shelf modules.However, we were unable to find a product that would meet all needs. Specifically, we were looking for a regulator with good efficiency, variable voltage output and low ripple voltage.

Additionally, we wanted the ability to turn the output on and off to meet the power sequencing requirement of our more complex processors (such as our FPGA). We also wanted a small board since space come at a premium in our submarine. To achieve all those goals, we decided to design the module ourselves. For a good efficiency, we decided to use a switching regulator as the efficiency is considerably greater compared to linear regulators. To simplify the design, we chose a controller with an integrated MOSFET, allowing us to reduce the number of components on the board. Thanks to the different steps we took to tightly integrate the converter, we managed to create an 11,3 mm X 20,04 mm board.

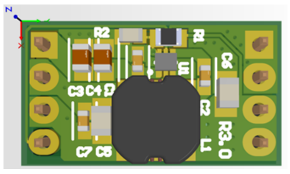


Fig. 3. DCDC Design

4) Hydrophone: The hydrophone is an important project for our submarine. We value this project a lot since it is a huge learning experience for our members and the acoustic source locations is a well rewarded task at the competition. At the 2019 competition, the hydrophones were not working, and we had a lot of noise issues on the signals. Those issues came from the noise of the thrusters at 35 kHz. Sadly, the documentation of our old acoustic system was not sufficient to maintain the project running.

Therefore, we had to redesign completely the hardware and software for the hydrophones. The objective for the hardware was to have a newer platform and an easier method to test the different components. For the software, the objective was to reduce the learning curve imposed by the complexity of the mathematics used in the algorithm. Also, we wanted to upgrade the precision of the hydrophone's detection to be around a meter from the exact position of the pinger.

To improve our hardware testing and revision process, we have implemented a quick replacement system for our filters that are made with two 20 pins connectors. The hydrophone main board has its own power converters, FPGA and UART communication with the on-board computer. Each filter has its own dedicated communication bus to make the main board design adaptable. Another advantage for the separate filter is that the layout is going to be identical for the 4 filters. We also updated the Spartan FPGA with the newest Spartan-7 series for more flexibility on the software.

To reduce the learning curve on the software, we have used Matlab and Simulink to generate some HDL code for the algorithm Time Difference of Arrival used to locate the pinger. We still had to create the core of the FPGA ourselves, but the learning is much easier. Also, by working with Matlab, we are certain that the code will be compatible between different versions of FPGA since we can choose the target FPGA when generating our HDL code.

C. Software

1) 6-DOF Model Based Controller: [4]

During the competition, all tasks require aligning the submarine with visual data obtained from the front and bottom cameras. These images or shapes will provide information on the orientation and position the submarine must take to line up. Proper alignment of the submarine is essential to complete those tasks. This requires that the submarine must move a long distance quickly without accumulating too much error. For the torpedoes task, a bad controller could mean the complete failure of the task and a big loss of time. For the octagon's task, it could be disastrous if we surface outside the octagon, which would end the run. Trajectories would also be used for movements between tasks. To perform this kind of task we need a suitable controller.

True to our philosophy, we wanted to innovate with our new controller. We also believe that knowledge sharing is very important and that is why we want to make this project an open-source project. The aim of this project is to design a modular control that can be used for our two current prototypes as well as our future prototypes. This control can be adapted by giving specific constants that represent the AUV. We also want to design a control that is easy to use and modify to allow other members as well as the next generation of S.O.N.I.A to use it as it should and that is why it will also be highly documented. We find it very important to document the development and the operations, but it is also very important for us to share our thoughts to leave a good traceability of our choices in relation to our project. The next members will then be able to get involved in this project in turn and continue to innovate in terms of control for the benefit of the S.O.N.I.A club as well as the community of creators of autonomous submarines.

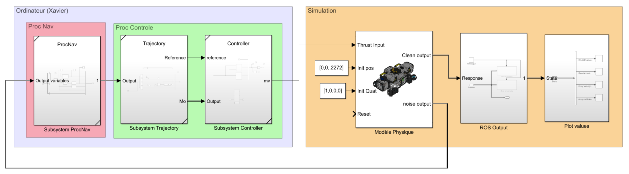


Fig. 4. Controller in MatLab

Attitude's Representation: [7] To represent the orientation of the submarine, we decided to opt for the unitary quaternion instead of the Euler angles as much for mathematical reasons as for implementation reasons. Even if the quaternion is not as intuitive as the Euler angles, it has some big advantages which explain its popularity.

From a mathematical point of view, since the quaternion has four parameters instead of three, it can define any rotation without any singularity. This phenomenon is better known under the name of Gimbal lock. In addition, the quaternion does not contain any discontinuity. With Euler angles, this problem is better known as wrap around. Regarding the implementation of our model, we have an advantage to use the quaternion, because it is more stable numerically.

Controller: [6] By default, the MPC command is a linear command. However, several tools are available to work with nonlinear systems. In our case, the submarine is a nonlinear system. However, the non-linearity does not vary greatly over a short period. It can therefore be said that the system is not highly nonlinear. In addition, the number of states and the sampling time will not change over time. In a case like this, we can usually use an adaptive MPC. This variation uses the Jacobians of the linearized model at the current operating point. However, we tested this method on our model shown with the quaternion, but the use of a Kalman filter with the quaternion was inconclusive. The error in the estimate was too great. To overcome this problem, we can use an external extended type Kalman filter (FKE) which better estimates the non-linearity. The estimation of states is nonlinear, but the control is linear. Although this type of MPC with an extended Kalman filter is more resource intensive and, our tests have shown notable results. To implement the MPC as part of our project, we used the MPC toolbox in Matlab and Simulink. This toolkit offers several tools to design MPC controllers (linear or non-linear).

Software Implementation: To do the software development of the submarine control, we opted to use Matlab and Simulink. We made this choice given the multitude of tools that Matlab and Simulink offer to develop complex software such as a specific ”toolbox” for robotics. This tool allows you to develop ROS nodes to communicate directly with all the modules of S.O.N.I.A. Another ”toolbox” also allows us to design an MPC that we used to make our controller. Finally, using Simulink, we can do a lot of simulations to test our physical model and the entire controller.

Another reason, less technical, also influenced our choice. Given the small number of members within S.O.N.I.A, we believe that using software such as Matlab and Simulink can simplify the development of an algorithm as complex as our controller. Also, most of the members who have worked on control in the past are enrolled in the Automated Production Engineering program at ETS and have used Matlab and Simulink in some of their courses. Such development directly in C ++ would have taken much longer to develop without the use of Matlab and Simulink. With these tools, we can directly deploy our code in C ++ for use on the submarine.

To generate the nonlinear state model of the submarine, we proceeded in several steps. First, we symbolically wrote dynamic equations using Matlab's Symbolic Math Toolbox. Next, we substituted the submarine-specific physical model constants with the numerical values. Finally, we were able to generate Matlab functions from the symbolic equations for use in Simulink. We generated a function for the nonlinear equation of the submarine considering disturbances, another function without considering disturbances and a function to generate the Jacobian matrices at an operating point. These Matlab functions could therefore be used inside Simulink to perform simulations.

Desktop Prototyping and Deployment: We want the deployment to be quick and easy to be able to make changes in our Simulink during pool testing and then directly test those changes. The first method that we present gives us precisely these criteria. Simulink, using the ”ROS Toolbox” [5] library, allows us to connect to our submarine via an SSH (Secure Shell Protocol) connection to directly deploy the generated codes and then execute it. This feature also makes it possible to see in real time the information in the ”Scopes” that we have placed in Simulink to know the information coming from the sensors. This feature is called “Monitor and Tune”. When our tests are done and we want to deploy the ROS nodes on the submarine, we will proceed with what we call going into production. We will particularly use this method to generate code tested using ”Monitor and Tune” and which is functional. The C / C ++ code generated by simulink will then be put in a Docker container provided for this purpose that can be used on the submarine to run the controller we developed.

Trajectories: Trajectories are a great way to give our controller a reference to get to a given position or positions and that is why we have opted for a trajectory generation system to ensure desired trajectory is followed. We wanted to allow anyone who uses the control to be able to give a list of points to generate a trajectory passing through these points, but also to give parameters to be respected for each of them.

2) Dockbox: One of the challenges of the pool testing is to be the most efficient as possible and it all starts with a good setup of our different departments. We found that the software team have the longest and the more hazardous one. Therefore, we decided to ease the deployment of our department with a new dockbox. In the past, we needed to bring an old laptop to use it as a network bridge to access the internet and it was always a bit too long. We wanted a plug and play, easy to use dustproof, waterproof, and shockproof box to bring everywhere we needed that would protect our electronical devices.

This new dockbox is driven by an NVIDIA Jetson TX1. When powered on, this computer is used to bridge the network inside the dockbox to provide the internet to the software team and the AUV if needed. Also, a router and a network switch are installed inside the dockbox to allow the team to connect to the AUV and the internet. We installed many waterproof ethernet connectors and a waterproof power connector to keep the box, made from a Pelican Case. The team also use the dockbox as a file storage server using a Samba share which allows to store and access files during the tests. To access the internet, the network Eduroam [2] is used. Eduroam (education roaming) is an international roaming service for users in research, higher education, and further education. It provides researchers, teachers, and students easy and secure network access when visiting an institution other than their own. Authentication of users is performed by their home institution, using the same credentials as when they access the network locally.

3) Proc Image Processing: [4] Part of an autonomous submarine's challenge is to be efficient in the usage of the resources of onboard equipment. This includes the power drawn by the onboard computer of our submarine. One of the data processing modules we run on our Jetson AGX Xavier is performing basic transformation and detection from the cameras video feed.

Those algorithms based on the OpenCV2 library are presently running on the CPU of our onboard computer. Thus, to improve our efficiency, we can use advantage of the Jetson's GPU to run OpenCV image transformations on it instead of its CPU. Doing so, our submarine will use a more appropriate part of the hardware to compute images efficiently. To implement this change, we need to use a version of OpenCV with CUDA3 [3] enabled in it and incorporate it with the rest of our software architecture. This part of the project came with its own challenges. In the first place, the version of ROS we were using had its version of OpenCV shipped with it and was not allowing us to use CUDA. To solve that, we had to rebuild it with the proper configurations, and we took the opportunity to update the versions of the OpenCV used. In the process, we encountered another blocker when building the library OpenCV with CUDA. Since our workflow to build Docker images passes by the GitHub CI, the task would take over 6 hours and be shut down by GitHub for taking too long. To get around this one, we modified our workflow for this exception by building it ourselves and publishing the resulting image afterward. This was conceivable since we will need to rebuild the image only when we want to update the version of CUDA or OpenCV.

4) Deep Learning Training Automation: During the past years, a lot of effort has been put on the development of our deep learning and the process of training became increasingly complex. In fact, most of the project became a practical task which is difficult for newer members since it requires more than basic deep learning knowledge. Focusing only on external aspects of training, like identifying the most favourable neural network from TensorFlow, tuning hyper-parameters, and choosing the right dataset, allows for better results since more training jobs are completed as the workflow get easier. Giving access to this project to the first-year member will give us a massive advantage over the years as they will carry their expertise through more competitions. Therefore, the process of acquiring the data, preparing the data, preparing the model, and training the model was automated.

We used Apache Airflow to create pipelines that principally extract, transform, and load the data but also train the model. The groundwork for the training is carried out by multiple DAGs (directed acyclic graph), which are a collection of tasks in Airflow, who all have a key role in the orchestration of a machine learning job. The DAGs are separated entities but the process as a whole extract images from ROS bags, export images to Labelbox where the team labels them manually, build the necessary directory structure for training and import Labelbox projects into those directories. The model can then be trained with GCP and use their GPUs, or it can be trained locally. Depending on the training method local directory or GCP buckets will be used. Every DAG can also be run independently of the others, so certain steps do not have to be repeated every time we train a different model.

5) Web Telemetry: [4]

For many years, we used a homemade telemetry based on a ROS framework named RQT. Our telemetry had strong dependency with ROS. With our new software architecture, we had problems of portability because the implementation of the old telemetry inside a Docker container was hard to do. Therefore, we decided to create a new telemetry app using React, Typescript and Roslibjs.

This new web telemetry is portable and allows the team to connect from any device they want. (Windows, Mac, Linux, IOS and Android). This telemetry gives us a lot of modularity that fit every team member's needs since each user can choose which module to add to their web page and how many of each they want. To properly monitor the submarine's pose and speed in his environment, we decided to create a PFD (Primary Flight Display) inside our telemetry. It will take some time to get used to, but we think that it is a more efficient way to get the information about the submarine movement. To create this telemetry, we used React and Typescript to ensure portability of the code and ROSLib JS to communicate between ROS and the web app via an intermediate of ROSBridge. We used Material UI, an open-source project that features React components that implement Google's Material Design, to design our UI component modules. The architecture of our new telemetry uses a layout system that support module integration and ensure modularity and maintainability.

6) 3D Simulation: Testing is an important part of any development, especially when you try to innovate using less known technology like in our submarine. With the opportunities to test our physical platform reduced this past year, we realized how much important it is to have an accurate and fast way to test our work. Using our old telemetry on RQT, we had ways to test single modules by using RosBags, but it required recorded data prior to testing. Since they are recorded, this data cannot be changed. This can be problematic. For example, if we wanted to test a new alignment algorithm, the video feed used as input does not replicate the new movement asked by the tested algorithm and you need to analyze your output in your terminal which is not efficient at all.

We wanted to create a way to simulate the submarine that would permit every member of our team to test their advancement in a single environment. To do that, we decided to create our own simulation environment. This simulation needed to be the most accurate as possible for testing to be effective. We specially had two systems that we wanted to focus on which are the submarine controller and the cameras. We knew that gazebo had a very powerful built-in physical model and that the unity one is more on the arcade side, but the visual rendering of unity with its post-processing capabilities is what made us choose Unity.

One of the challenges using Unity is the connection to ROS. We chose to go with the 2020 version to use the new Unity robotic package [1] which includes a ROS integration. This package lets us create our own message and communicate with the ROS master to listen or publish messages and services like our submarine would do. By using Unity, we prioritize the visual rendering to the physical model knowing that with the new control we were developing, we already had to create our own physical model. By using our own model, we can use positions output by our controller to give the position we want the submarine to be in Unity. One thing that can be overview is the portability of Unity. Since it can be compiled for any platform, anyone can use it without being on Linux or needing to have ROS installed.

III. Experimental Results

A. Mecanical

1) Rigid body constant: We have decided to use a more theoretical approach to define most of the constraint needed because we can not have a proper feedback the small pool. For the rigid body constraint, we weighed all the components, and we entered the results in the properties of the respective cad in Solidwork, then we can use the mass property feature of solid work, to determine the Mass, the center of mass, the volume, and the inertia tensor. To make sure our data makes sense, we then weighed the assembled sub to make sure it matches the mass calculated by solidworks and we are less than 500g away. This is mainly caused by the fact that the cables are not modelled. Therefore, we were able to check if we have respected the consideration and the strategies that we have defined previously. If we start with the volume, compared to our previous submarine, Dina is half the volume with 27L compared to 55L. Next, if we look at the inertia tensor and more precisely at the main inertia moment and compared it to our previous prototype. Data has shown that we have succeeded in reducing the inertia about roughly 72% around surge axis, 15% around sway axis and 47% around heave axis.

B. Electrical

1) Backplane: We have done a stress test of the components of our first revision to detect potential thermal issues. With a current of 15 amps, we have seen a temperature rise to 83.1°C after a full minute of load. To get 90% of our thrusters, we would need new parts with a lower internal resistance and a better thermal management.

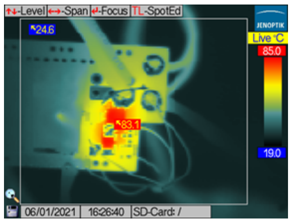


Fig. 5. Heat testing result after 1 min

2) Leak Sensors: The circuit has been simulated using MicroCap. The result of the simulation is presented below. The upper graph presents the output of the comparator, and the graph below represents the resistance of the electrodes. From those graphs,

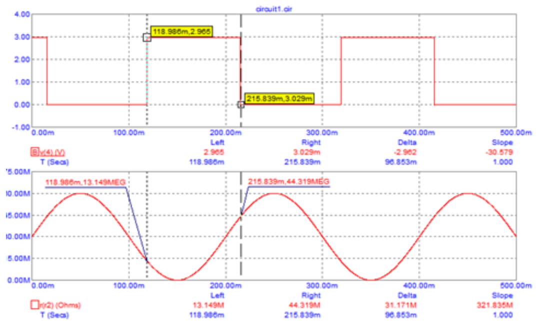


Fig. 6. Leak Sensor

we can see the hysteresis of the comparator as the output switches to high when the resistance is around 13.15 Mohms and the output resets only when the resistance is around 44.32 Mohms. Those resistance values are much higher than those that we expect to use in our final design. We intend to find better suited values experimentally once our first prototype is done.

3) Direct Current to Direct Current: In our original design, we had grounding problems which resulted in higher-than-expected ripple voltage. By reducing the size of the ground loop, we managed to reduce the ripple voltage from 122 mV to 64.8 mV. Although this is a stark improvement, we are still looking at refining the layout and the component selection to reduce the ripple voltage.

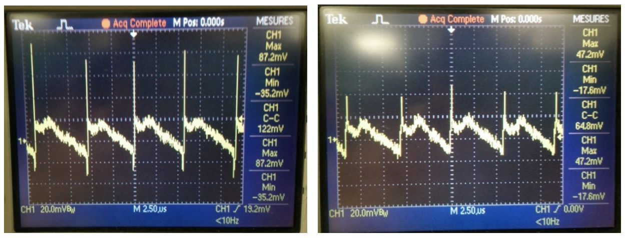


Fig. 7. Ripple Voltage Testing

4) Hydrophone: Since we only have one working filter for now, we could not test the localization algorithm since the full four filters are needed to locate the source. On the other hand, we were able to test the reception of data from our pinger in miniature pool environment. For future pool tests, we would want to start defining the threshold of the signal to separate the desired ping and the unwanted noise. This process has been estimated in the simulation, but we found inconstancy in the results produced so far. After defining the threshold, we will be able to focus on the accuracy of the algorithm. For now, we have done some simulation with the algorithm used on the FPGA and we have achieved the precision of 0.79 metres when the submarine is at 15 metres of the source with a Gaussian noise with a variance of 0.01. The data used to generate the signal was the data collected at the TRANSDEC during the 2019 competition.

C. Software

1) 6-DOF model based controller [4]: Following, we will present the results of a simulation that we performed using Simulink. To create the trajectory used for this simulation, we proceeded in several steps. First, we plotted this path in the SolidWorks modelling tool. This is what the modelled path looks like. As you can see in the following figure, we have chosen to raise the submarine while turning. We wanted to generate a trajectory where the submarine moves on 4 degrees of freedom simultaneously. Then, we chose some points on the trajectory to give them to our trajectory generator using a Python script. Here is the trajectory that was generated by our trajectory generator. In the figure, we can observe the points generated as well as the orientations for each of them in a 3D perspective.The Figure 8 shows the generated points and orientations that will be sent as a reference to the MPC. The Figure 9 are the graphs of the linear positions and the quaternion of the trajectory that we generated. The Figure 10 compare the outputs (the states in our case) to their respective references. The Figure 11 shows the command of the motors coming from the MPC.

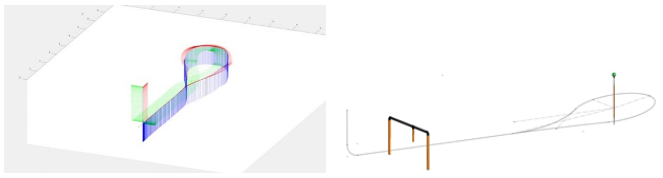


Fig. 8. First Trajectory Generated

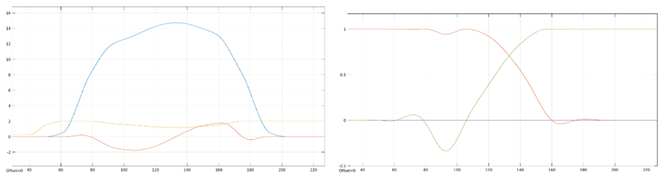


Fig. 9. Linear Positions and Quaternion Generated

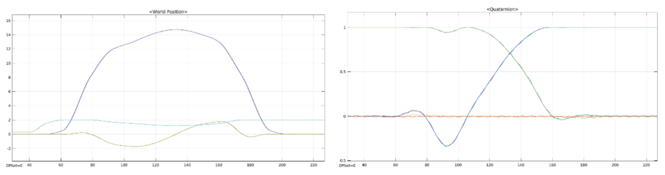


Fig. 10. Output States

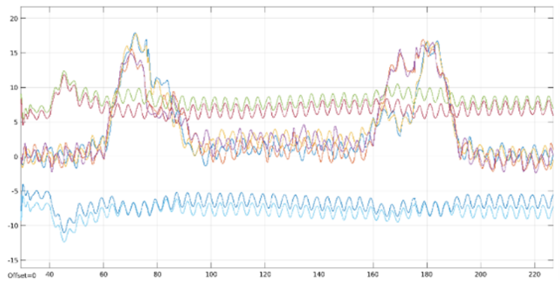


Fig. 11. Command from MPC

According to those results, we can notice that the controller is able to follow the trajectory that has been imposed on him while respecting the constraints that we have imposed. We also notice that we have a zero error in steady state and that if an overshoot is observed, it is negligible. The points that we sent to create the trajectory resulted in a trajectory as we wanted it. The generated trajectory is not completely smooth and has some imperfections, but it will be improved in the future.

Although we had the opportunity to run simulations with Simulink as well as in our simulator, we would have liked to be able to perform tests in the pool. As soon as the sanitary situation allows us, we will be able to perform more realistic tests in the swimming pool to see our work in action in the water and do the work necessary so that our control can be officially deployed on the submarine.

2) Dockbox: During the year, we had the opportunity to test the dockbox many times. Each of our mini pool test, we tried to connect multiple persons on the box without any issues. We made large downloads and uploads to test if all was good too. We are confident with our new dockbox to be a reliable way to improve our deployment efficiency and we are using it every test we have.

3) Proc Image Processing [4]: We will be able to measure the gain in performance by looking at the number of frames per second we can process with our solution. A reduction of the CPU load of our onboard computer should be noticed. We could also measure the difference in power consumption between the old execution of filter chains on the CPU and now on the GPU.

Furthermore, we looked at the modifiability of the module at the same time. Since we need to modify this module often by adding, modifying or even retiring filters following the tasks needed to be performed in the competition, we wanted to be sure the effort required to do so was minimal. To help us keep track of this important quality attributes, we integrated a new tool to our workflow named Sonarcloud. This tool will help us keep track of the quality of the module's code and will help members identify fixes for bugs, vulnerabilities, code smells, code duplications and test coverage.

4) Deep Learning Training Automation: The implementation of Airflow pipelines has already been useful. The newest members interested in artificial intelligence have been able to train models and improve their abilities on this subject without external help of the most experienced representatives.

5) Web Telemetry [4]: We recently tested our telemetry with AUV8 during one of our tests. We were able to test the image viewer and the PFD. We will certainly add more modules inside our telemetry and improve the ones already created. Each time we modify a module, we will make sure to do the proper tests with the submarines to validate the new functionalities added.

6) 3D Simulation: Testing the testing environment is a crucial step toward its development. Since we could not do it in a pool big enough to get credible data, we cannot be assured that the physical part of our simulation is accurate. On the other part, we used the 2019 competition recorded RosBag to test the visual rendering of Unity. We are happy with the result between the simulated environment and the Transdec footage, but we know it can still be improved for more realism. We were not able to test a newly trained AI in unity since we had some performance issues with the image publisher message, but this is the next step of testing. We look forward to the end of the testing because we would want the simulation to be part of our new web telemetry giving us 3D representation in real time of the submarine action during the pool testing.

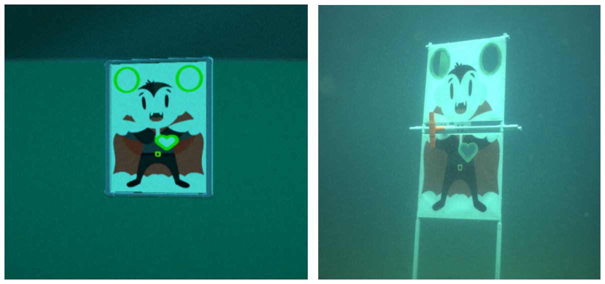


Fig. 12. Vampire in Unity vs Vampire at Transdec

ACKNOWLEDGMENT

We wanted to highlight the involvement of Mr. Alain April. For several years, he has been helping our team members with their projects. He also allowed us to undertake the deep learning automation project thanks to the external GPU that he offered to our club.

We wish to thank Mr. Rachid Aissaoui who supervised us throughout the new controller project. He greatly helped us in our reflections, our choices as well as the acquisition of technical knowledge. He offered us invaluable support for the full realization of our project.

We wanted to thank members that dedicated a course to help our team:

To start the Docker projet: A special thanks to Marc-Antoine Couture, Martin Gauthier and Kevin Charbonneau supervised by Alain April.

For our new Web Telemetry: A special thanks to Laurent Boisvert, Hervé Carol and Anthony Frezzato supervised by Alain April.

For our new Controller: A special thanks to Alexandre Desgagné and Alexandre Lamarre supervised by Rachid Aissaoui.

For the new proc image processing: A special thanks to Pierre-Yves Vincent-Tremblay, Guillaume Tremblay, William Tang and David Sauvageau supervised by Alain April.

We wanted to thank the club ReflETS for the incredible photos of our submarines as well as our team photos. We also wanted to thank the club Cedille for hosting and maintaining our server.

We wanted to thank our sponsors:

Diamond: Bruël & Kjaer, Altium, SolidWork and Cegep du Vieux Montréal

Platinium: Lojiq, Lenovo, AEETS, Cegep St-Jérome, Usinage Villneuve and Fond de développement ÉTS

Gold: Teledyne Marine, Labelbox, Caisse Desjardins and Drillmex

Silver: Travis CI, Parc Jean-Drapeau, Vectornav, Tritech, Samtec and Digi-Key electronics

Bronze: Nvidia, Connect tech In., Blue robotics, Laser AMP, Anodisation Expert, Groupe Rivest, Simplify 3D, General Dynamics, Attaches Richard

REFERENCES

[1] Unity Technology: Unity Robotics, `https://unity.com/solutions/automotive-transportation-manufacturing/robotics`
[2] Eduroam, `https://eduroam.org/`
[3] CUDA ToolKit, Nvidia Developer, `https://developer.nvidia.com/cuda-toolkit`
[4] S.O.N.I.A, Special project and Capstone project, `https://wiki.sonia.etsmtl.ca/en/software/projects`
[5] Castro, S., (2017). Getting Started with Matlab, Simulink, and ROS, `https://blogs.mathworks.com/racing-lounge/2017/11/08/matlab-simulink-ros/`
[6] Islam, M., Okasha, M., Sulaeman, E., (2019). *A Model Predictive Control (MPC) Approach on Unit Quaternion Orientation Based Quadrotor for Trajectory Tracking*. International Journal of Control, Automation and Systems: KIEE and Springer, 14 p.
[7] Fjellstad, O., Fossen, Thor I., (1994). *Position and Attitude Tracking of AUVs: A Quaternion Feedback Approach*. Norwegian University of Science and Technology: Department of Engineering Cybernetics, 15 p.

APPENDIX A
COMPONENT SPECIFICATIONS (AUV7)

Component	Vendor	Model/Type	Specs	Cost	Status
Buoyancy Control	-	Dead mass	Brass plates	-	installed
Frame	Homemade	CNC aluminium system	6061-T6 CNC machined and anodized	-	installed
Waterproof Housing	Homemade	Carbon Fiber and CNC aluminium system	6061-T6 CNC machined and anodized	-	installed
Waterproof Connectors	TE Connectivity	Seacon connector	Wet Mate	-	installed
Thrusters	Blue Robotics	T200 (x8)	0.02 kg f	-	installed
High Level Control	Homemade	4DOF PID	-	-	installed
Actuators	Homemade	-	65N (spring for torpidoes)	-	installed
Battery	Multistar	4S 16000mAh	14.8V	-	installed
CPU	Nvidia	Jetson AGX Xavier	16GB RAM	-	installed
Internal Comm Network	Homemade	RS485	2 twisted pairs Ethernet cables	-	installed
External Comm Network	ConnectTech	XDG016	1000 Mbps Switch	-	installed
Inertial Measurement Unit	MicorStrain	3DM-GX3-25	-	-	installed
Doppler Velocity Log (DVL)	Nortek	DVL500	300m	-	installed
Vision	Flir	Chameleon 3 USB	55FPS, 3.2MP	-	purchased
Acoustics	Brüel & Kjaer	8103	0.1 to 180kHz	-	broken
Manipulator	BlueRobotics	Newton Subsea Gripper	modified to open up to 10cm	-	broken
Algorithms: vision	OpenCV	-	-	-	installed
Algorithms: acoustics	Homemade	-	10kHz to 50kHz	-	installed
Algorithms: autonomy	FlexBe	Finite-state-machine	-	-	in testing
Open source software	OpenCV, FlexBe, AirFlow, TensorFlow, ROS, Unity Robotics, Docker, React, WikiJS, Github				installed
Team Size	16				
Expertise ratio	10/6				
Testing time: simulation	Simulation development still in progress				
Testing time: in-water	0 hours				
Inter-vehicle communication	Water Linked AS	MODEM M64	64 bits, omnidirectional	2000$	purchased
Programming Languages	C/C++, C#, Python, React JS, Matlab				

APPENDIX B
COMPONENT SPECIFICATIONS (AUV8/DINA)

Component	Vendor	Model/Type	Specs	Cost	Status
Buoyancy Control	Homemade	-	Foam	-	in design
Frame	Homemade	CNC aluminium system	6061-T6 CNC machined and anodized	-	installed
Waterproof Housing	Homemade	CNC aluminium system	6061-T6 CNC machined and anodized	-	installed
Waterproof Connectors	MacArtney	Subconn connector	Wet Mate	-	installed
Thrusters	Blue Robotics	T200 (x8)	0.02 kg f	-	installed
Motor Control	GetFPV	Bullet 30A ESC (x8)	30A	12.99$	installed
High Level Control	Homemade	MPC Controller	-	-	installed
Battery	MaxAmps	4S 16000mAh	14.8V	-	installed
CPU	Nvidia	Jetson AGX Xavier	32GB RAM	-	installed
Internal Comm Network	Homemade	RS485	2 twisted pairs Ethernet cables	-	installed
External Comm Network	ConnectTech	XDG016	1000 Mbps Switch	-	installed
Inertial Measurement Unit	VectorNav	VN-100 Rugged IMU/AHRS	Standard calibration +25	-	installed
Doppler Velocity Log (DVL)	Teledyne	Pathfinder	600kHz, 140m	-	installed
Vision	Flir	Chameleon 3 USB	55FPS, 3.2MP	-	purchased
Acoustics	Brüel & Kjaer	8103	0.1 to 180kHz	-	installed
Algorithms: vision	OpenCV	-	-	-	installed
Algorithms: acoustics	Homemade	-	10kHz to 50kHz	-	in testing
Algorithms: autonomy	FlexBe	Finite-state-machine	-	-	in testing
Open source software	OpenCV, FlexBe, AirFlow, TensorFlow, ROS, Unity Robotics, Docker, React, WikiJS, Github				installed
Team Size	16				
Expertise ratio	10/6				
Testing time: simulation	Simulation development still in progress				
Testing time: in-water	15 hours				
Inter-vehicle communication	Water Linked AS	MODEM M64	64 bits, omnidirectional	2000$	purchased
Programming Languages	C/C++, C#, Python, React JS, Matlab				

A. Competition Inter Quebec

While having a second online edition of RoboSub keeps our mind sharpened and focused on producing a good work for a competition and fully understanding the decision of the organization, we still had a feeling we would be missing out something if we could not have a physical experience with other teams. Most of our team members had the chance to participate in the 2019 RoboSub competition so we know the kind of experience this event provides and we know not many of us will have the chance to come back in San Diego next year. We did not want to leave the team unprepared for the next competition. We are also missing the community ambiance that can only be found at the Transdec or the hotel with all the other teams. All these reasons led us to organize our own small competition with teams from Quebec.

We will have this competition between McGill, Trois-Rivières and us. All these teams have already participated in at least one RoboSub competition. We knew McGill team from previous collaborations such as pool test sharing and we had met Trois Rivières in 2019 when they came in San Diego for the first time. We want to use this competition as a way to create stronger bonds between our teams and start to collaborate more with each other. We intend to recreate a format like RoboSub with test runs and a final run. The tasks will be similar to RoboSub as we also want to create this competition in prevision for the next years. The long-term goal is to have this competition a few months after RoboSub in order to show to the new members what will be required, fix what went wrong at the Transdec and share what went right to the other teams so everybody can improve.

The fact that it's a local competition will help us with the sanitary rules, for now the evolution of the situation lets us think there won't be a problem this autumn when we want to have the competition. We were thinking of having the competition in a natural location to recreate the environment of the Transdec, specially for the water color and the shining of the sun which are two influent components when we are in San Diego. But finding a perfect spot with all the facilities close is a hard task and we had to resolve ourselves to pick an indoor diving pool. At least we'll be safe from any unpredictable event.

The competition is set to happen this fall so we don't have any result to provide yet. But as for establishing stronger bonds between the teams, we can proudly say it is already a success. We had more contact with these teams in the past four months than in the last three years and we are all eager to meet each other at the competition.

B. Data Sharing

A recurrent situation when using Artificial Intelligence is the lack of data. We are all facing challenges in multiple departments of our teams and we don't have necessarily enough time to get all the data we need to train or test our models, and this even more true for new teams. When we entered SONIA, one of the first lesson was that we all do this for the community and being open source is a pride. This is why when Hitesh and Julianna came to us talking about starting a Data Sharing Platform, we accepted immediately.

On the Data Sharing Platform, any team registered in RoboSub, RoboBoat or RobotX competition will be granted an access to the data uploaded by the teams in the previous and present year. This is not just about training and test data for AI models, the teams can also share their mechanicals, electricals and software designs to help the community.

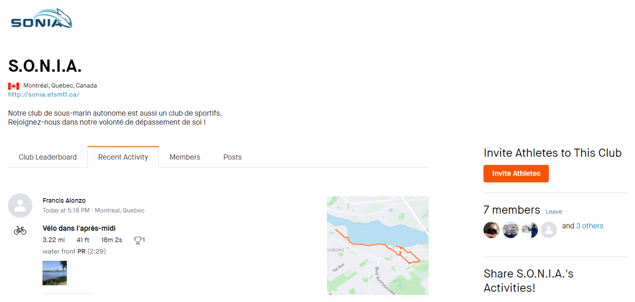
The Data Sharing Comity has been gathering almost every week for one year to design the rules and the organization of the platform. We made test on who could share, who could only download and we decided that the teams could upload and download files from there generic account. And they could share the access to the platform for their members that would get an access to just download the files. This is in order to make sure the platform remains as clean as possible. The comity will be here to maintain the platform and help the teams with their problems. The comity is composed of members of teams interested in data sharing and will be renewed as members leave so this is really a community driven effort.

The platform had a beta testing phase in the Fall, based on the returns from the participants and make the adjustments needed. We are now in the soft launch phase because this is a year without a competition in person and with limited testing and data recording opportunity. This is not a bad thing since this will let time for the teams to visit and get comfortable with the platform. We have already seen some requests for data so this is encouraging. The real test will happen next year before the competitions season.

C. Team building activities

With the pandemic and the remote school, it has at times been difficult to connect with other members of our club outside of work on the submarine. It was important for us to keep the atmosphere of our club even from a distance. To do this, we participated and organized several small activities allowing us to keep in touch and have fun together.

1) Strava: Some of our actual and old members decided to form a Strava group to encourage members to exercise. It's a great way to keep in touch and to feel together even from a distance.



S.O.N.I.A.

Montréal, Quebec, Canada
http://sonia.etsmtl.ca/

Notre club de sous-marin autonome est aussi un club de sportifs.
Rejoignez-nous dans notre volonté de dépassement de soi !

Francis Alonzo
Today at 5:18 PM · Montreal, Quebec
Vélo dans l'après-midi
3.22 mi 41 ft 16m 2s 1
water front PR (2:29)

Fig. 13. Strava Group

2) Multi-Gaming Competition: At our school, a group organized a multi-gaming competition in 6 stages. We decided to participate with a team made up of members of S.O.N.I.A. It was very nice to have a little competition every month. We are in a battle for the top position and we are waiting for the latest results