# Southern Polytechnic State University Autonomous Underwater Vehicle Design Rationale for Subzero

Douglas Allyson, Albert Cheng, Alex Evans, Andrew Geiman, Lorenza Hill, Taylor Martin, Ada Odey, Nikhil Ollukaren, Bennett Stedwell

June 27, 2014

**Abstract**

The Southern Polytechnic State University Subzero is a littoral-class autonomous underwater vehicle (AUV) built by undergraduate members of the SPSU AUV Team. The vehicle has been continuously modified and enhanced over the past several years, however the current configuration is a complete redesign from past vehicles, the product of a ten-month development period. The vehicle was designed almost completely using three-dimensional CAD and simulation in Dassault Systemes' Solidworks design software. Among the new design's features are redesigned camera housings, a new main housing, and a hexagonal, riveted 6061 aluminum exoskeleton to provide structural security while minimizing weight. The submarine is equipped with two cameras for challenge recognition and maneuvering computer vision tasks, a pressure sensor for active depth control, and an inertial measurement unit for orientation control.

| **Quick Facts** | | | |
|---|---|---|---|
| Dry Weight | 17.23 kg | Thrusters | 6x SeaBotix Brushless BTD150 |
| Dimensions (LWH) | 78.74 cm x 38.10 cm x 48.26 cm | Cameras | 2x Logitech Web Cams |
| Max Depth | 18.29 m | IMU | 3DM-GX3-25 |
| Degrees of Freedom | Yaw, Surge, Heave, Sway | Motor Drivers | 3x 2x5 Sabertooth Motor drivers |

# Software

## Operating System and Languages

To avoid unnecessary complication, a software stack was used to provide communication and interfacing with the sensors and cameras. This allowed for additional time to be spent addressing the challenges, rather than perfecting these utilities. These functions were provided by Robot Operating System (ROS). As explained on ROS's website, the Robot Operating System is a set of software libraries and tools that assist in building robot applications. ROS possesses open source drivers, state-of-the-art algorithms, and other powerful developer tools. As ROS releases a new version every six months, throughout the development period versions of ROS had to be updated, from "Fuerte" to "Groovy" to "Hydro". Fortunately, little code had to be rewritten between each update.

Using ROS narrowed down the options choices of operating system for the on-sub PC. Even though ROS can be run on a variety of Linux distributions and Windows, only one distribution is fully supported. Because it was the only distribution supported by ROS, it was decided to use Ubuntu. Code is not developed on the on-sub computer; instead, team members use personal laptops or lab computers, which have ROS installed, to develop any software for Subzero. One reason for this was for the concern that doing many compiles on the embedded PC could wear out the flash memory on the CF card, which is its only means of storage. It was decided to run Ubunut on both the development and on-sub machines, but with each installation customized for its purpose. The installations on the development have all the bells and whistles, a fancy desktop environment (Cinnamon), a browser, many editors, etc. The embedded PC installation, however, boots directly to the terminal with the option to start a GUI for testing, and little extra software aside from ROS and our code so that everything can fit on a 8 Gb CF card.

Because we are using ROS, our options for programming languages were also limited. The two options for languages that can use the ROS libraries are C++ and Python. Initially, the team decided to write in C++ for the speed of native code. After a bit of head scratching, the team switched to Python for the performance capabilities when vision processing tasks are running. When the code was ported to Python, there was no evidence of any performance lost on the other systems, so the team continued to move all current software code into Python.

## Architecture

We decided to use a closed layer approach while designing our Robot Operating System package. This architecture was chosen because we believe it is easy to comprehend and teach to students who are new at computer programming. We have four layers in our architecture: Control, Decision, Calculating, and Device. The nodes in each layer can only interact with nodes from the layers directly above or below, as well as nodes from its own layer.

## Control Layer

The control layer houses only one node: Intelligal_Artificialence. This node's only job is to run the mission objectives that are defined in the decision layer.

## Decision Layer

This layer makes all of the decisions the sub makes. Each task has its own node in this layer; even the intermediate tasks such as recovering from the previous activity. We used this approach to simplify updating code and to streamline the review and merging process for our content management system.

## Calculating Layer

This layer handles all of the heavy number crunching functions; such as locating the start gate in a frame. In addition to all of the visual recognition code, the nodes that detect collision and our swim computer reside here.

## Device Layer

This layer contains software from the ROS community that pulls the data from the hardware and converts it into a format we can use. This layer contains the rosserial nodes which we use to communicate with the two Arduinos, one to control the thrusters and the other reads data from various sensors.

## Swim Computer

The swim computer is an idea we received from our colleagues in SPSUs Aerial Robotics Team. Their Unmanned Aerial Vehicles (UAVs) have a flight computer; which has a built-in Inertial Measurement Unit (IMU) and uses it to stabilize the aircraft. They recommended that we put an actual flight computer in our vehicle, so we could send it basic commands and have it handle stabilization and correcting for drift. However time constraints prevented us from using this idea, so we created our own in the software.

## Vision Hardware

For this year, our vision system will be using two USB cameras. One of the cameras will be facing forward and the other downward. We are using the downward camera to find the path on the bottom of the pool, while the frontward camera will be used for vision targets such as the start gate. The cameras will be placed each in their own enclosure made out of PVC, with clear plastic on one end, and a waterproof connected for a cable leading to the main housing to communicate with the PC on the other end.

For much of the time we were working on vision programming, we were using cameras connected by FireWire. To connect these to the PC, we used a FireWire to USB cable. These cameras took us a while to get working, though. These cameras gave us great images at a detailed resolution. But, sometimes the cameras would return an image that was half the correct image and half the before mentioned image, flipped and in black and white. This was too great a risk to have happen at competition where the sub would need vision to navigate, so we swapped the FireWire cameras with USB cameras to resolve the issue. Thankfully ROS made it easy to use different cameras using our existing code with only minor revisions to the launch file.

## Chaos

This past year we lost of our embedded PCs (EPC) through power source malfunctions. However due to the popularity of ROS and our choice of Python as a programming language, we were able to easily port most functionalities on to a Raspberry Pi, we had planned to use two Pies who split the work and use ROSs networking functionality to have the two communicate. Fortunately, we received our third EPC and had it up and running two weeks before we were to ship the vehicle. We currently have a contingency plan in place where we use the same model of pc that Aerial uses in favor of the Pies if something were to happen to the third EPC.

The EPC is set up to wirelessly connect to a debugging module that is secured near the main housing when it is in use. This module contains a Raspberry Pi with a Wi-Fi adapter that is tethered to a computer on the surface. This allows us to reduce the risk corrosion in an exposed connector.

## Plans for Next Year

Currently each task is split in multiple files; the main logic has its own node, but any supporting logic, such as vision, in in another node in the calculating layer. Next year we plan to have all code in one file, but have the supporting nodes import any needed functions from the main node into its own.

We also plan to switch over to Hardkernels ODROID U3 as our embedded pc. This is the same computer the Aerial team uses on their UAV. We will make this change because the ODROIDs footprint and weight is a fraction of the current PC, with comparable specs. The major downside to this idea is that the U3 contains 2 GB of RAM versus the current 4 GB. However if RAM becomes an issue, we plan on expanding to two U3s on the vehicle.

Next year we also plan on testing Aerials suggestion of installing a flight computer built for RC vehicles to correct for drift and to stabilize the vehicle.

# Electrical

## Safety

The Kill Switch is a magnetic reed switch similar to a door sensor for home security systems. When a magnet is brought close enough, the switch will change states. The switch is then hooked up to an Arduino and a 30 amp relay. When the switch changes state, the Arduino sends a signal to the embedded PC informing the vehicle of which state the motors are in. The relay is connected between the thruster batteries and the motor drivers. When the switch is in the off position, no current can pass through the relay.

A magnetic switch was chosen in order to reduce the number of possible places the vehicle could leak. A magnet can be attached to a brightly colored float so the safety diver can quickly de-energize the vehicle in the event of an emergency.

## Propulsion

The submarine utilizes six SeaBotix thrusters for maneuverability. They are brushed DC motors encased in a waterproof housing. They are able to produce a peak thrust of 2.9 kg force and are controlled by three Sabertooth motor drivers that communicate by simple serial. These drivers give users the ability to control the rotational speed and direction of the thrusters.

## Sensor Suite

The number of sensors utilized was limited by the events the team chose to compete in; as such the submarine is currently set up in a "bare bones" configuration. It has two cameras: one facing forward to locate and work through the challenges, and one facing the floor to handle line following for the movement to each successive challenge. The decision was made to use "off the shelf parts for the cameras, i.e. web cams, to minimize expenses and design complexity. The cameras are both Logitech brand, one being a ball-jointed type and the other a monitor- mount type. The primary sensors are a PX26 series pressure transducer, which acts as a depth gauge, and the IMU, used for underwater navigation. The pressure sensor is used to keep the submarine within the proper range of the tank floor, ensuring the sub does not breach the surface unexpectedly. The Inertial Measuring Unit (IMU) detects changes in the vehicle's orientation in three major axes: pitch, roll, and yaw, and is used for orientation of the vehicle as necessary for the tasks as well as stabilization during movement.

## Power Distribution Network

Initially the power system was housed separately from all the electronics; however this was found to significantly increase the weight of the overall submarine, with little additional benefit. The current fully integrated setup is simple and organized, with the focus of keeping power system maintenance as easy as possible. Motor power is provided by two 14.8 volt lithium-polymer batteries connected in parallel. The maximum voltage rating on the motors is 19 volts, so a parallel setup was used to stay under the limit while benefiting from increased mission time. The power from the batteries moves through a 30 A relay which functions as the "kill switch", and then gets passed on to three motor drivers to individually power the motors. The main power for the computers is a lithium-iron-phosphate motorcycle battery. Originally all power was to be stepped down from the motor battery setup; however this would have required additional circuitry to the submarine and thus another point of possible failure. The 12 volt motorcycle battery worked well without the need for additional circuits, making powering the computer as simple as plugging it in. All other electronics aboard the submarine e.g. the Arduino, IMU, Cameraetc. are powered through the computer.

### Connectors and Ports

The vehicle has seven Fischer connectors and two water proof glands. Four of the Fischer Connectors are two pin 105 series connectors with an IP64 rating and are rated to 30 amps per pin. There are two six pin 104 series connectors also with an IP64 rating, but are only rated to 5 amps per pin. Finally, the last Fischer connector is four pin connector of the same type and rating as the six pin. The cable glands IP64 rated, but because they are glands, they do not have an electrical rating. The cable passes through the gland and a nut tights the gland around the cable sealing it again water penetration.

## Structure

### Main Housing

At the beginning of the year a sketch by the mechanical lead showed how a cylinder would be able to hold the components of the main housing. The majority of the electrical components are housed in this design. The housing consists of clear PVC tube and two machined aluminium end caps. The end caps were designed in SolidWorks so that they could be made by CNC machine at a very high level of precision. The rationale for choosing aluminium for the end caps was that it has relatively low weight in comparison to the other metals, as well as a high rate of heat transfer. With this in mind it was decided to mount the embedded PC to one of the the end caps. The other end cap was designed to allow for multiple waterproof connectors to pass and connect to the electronics inside the tube. There were a total of 16 waterproof connectors passing through the housing. The two end caps utilize an O-ring channel so that the O-rings will be able to compress against the side wall of the tube, forming the watertight seal. The advantage of designing all the parts in SolidWorks was that it became possible to utilize the simulation and analysis suites available with SolidWorks. These tools enabled the Structures team to simulate the maximum pressure that the main housing could sustain before being destroyed.

### Main Housing Snafu

The main housing that is on the submarine was not supposed to be used this year. It has many disadvantages in comparison to the one that had been designed. However, it does have sufficient volume to hold all of the equipment needed. The reason for reusing this older housing was a logistics mishap; the watertight connectors were ordered, but the order was never completed. By the time the issue was discovered, it was too late to get the correct connectors in.

### Exoskeleton

One of the most important issues we faced was having to fabricate the submersible ourselves. Our first step was to gather information and brainstorm ideas, we did research into the standard types of raw materials that were available on the market. Home depot was our first stop, it was important that we get a physical hold of the metal, and judge what could be done with it. With this being done our next mission was to find a cheaper source that would give a price break for to student or larger quantities. However a supplier of this kind was not found until later in the competition season. Our research spanned the Internet as well and this widened our knowledge. With a sizable knowledge of the types of raw materials that are on the market each member of the team was tasked with submitting a rough design for the competition. We felt that a small competitive project would bring out the best in every one and yield a few good design features. This turned out to be a success with two separate designs being turned into a hybrid that was superior to all of the others. The next stage in the process was to order the tools and raw materials needed to construct a prototype. The prototype exoskeleton helped the team to improve, streamline and, increase the efficiency of our manufacturing process. We learned that Brake press bending machines are more accurate and better suited to preform bends, while the Chinese pipe bending press can perform the larger degree bends. This knowledge was very useful in creating the structural frame and then the support framing for the Exoskeleton. Getting access to the pieces of equipment turned out to be free and cheap, which is always a plus. The team also gained access to a small milling machine, this allowed for the time consuming and inaccurate of each and every hole on the structure, of the then four hole patterned joint. The joints were also tested and deemed unnecessarily large, because of this we switched to a two-hole pattern. With this design change we still felt it necessary to build a prototype, so we fabricated just one ring to see if the change compromised or structural integrity. This smaller and cheaper build allowed us to do deformation and impact testing which proved successful. Now that we have a general plan, the materials, the tools to get it done , and practice we were ready for the final build. We used three types of raw materials during our construction of the final exoskeleton. The raw materials we used include one inch by a sixteenth of an inch thick aluminum bar stock, this was used as light weight support structure and gave us the advantage of being easy to work and cheap. Camera housings and motors can be attached to the main frame work using this. The next is one inch by an eighth of an inch this bar stock, we used this for the construction of the ring, these were still very light but much tougher than the thinner stuff. With the thickness of the piece being increased we calculated that the increase in strength from bending would help the joints be stiffer and not deflect as much. The final piece this cage is the T-beam, this formed the crucial structure for our frame and allowed us to use lighter supports around it because of it superior geometry. An advantage to the current frame is that we have to freedom to put more on or design, we can change configurations easily and it is very light weighting only seven pounds. This was right at our design goal but it is my opinion that the Exoskeleton would not have met this goal if we had used bolts instead of rivets. The use of rivets was a big debate amongst the team with both sides having good reasons for and against there use. They are lighter and tougher than a bolt of the same material; they are also cheap with a hundred costing only a few dollars. Compared to bolts which need washers, lock washers and bolts to be a fastener, however rivets require tools and the proper know how. Rivets won in the end and proved to be the good choice.

## Ballast

We chose to go with a cheap and easy design for the ballast tanks leaving only the material choice and two dimensions up for debate. We put to common plumbing caps on both ends of common piping; we just had to choose material and length. We determined in our first pool test with the sub that three PVC two inch diameter pipes almost works, this was done with old materials laying around the lab. We planned on using what we had first and make new ones with the data we received for the pulmonary testing. When our test showed that the sub just barley sinks with the not streamline and bulky old ballast we then started the construction of the final tanks calculating in the proper size increase to make the submarine five percent buoyant. The new tanks we chosen to be made of 2 thin walled copper tubing. We received a donation of this material and with the walls being able to support a pressure much higher than the sub will ever encounter we did not have to worry about them failing. We also created a new PVC set that has smooth rounded ends instead of the bulky treaded end caps. This set was constructed in case the copper ones where not fabricated on time. In all cases we used common hose clamps to hold everything in position, this allowed quick attachment and removal along with a weight that is negligible.

## Pressure Housing

We needed to design a simple waterproof housing that would be affordable to manufacture in house and would be durable enough to last multiple completion seasons. Keeping these requirements in mind we decided to use an aluminum body which would be machined (CNC) out of raw stock which kept our material costs down to approximately $12 per housing. Using SolidWorks we were able to depth test our design for up to 60 feet of water and also gain insight as to how the internal electronics would be spaced out. Once the design was vetted for any potential problems the material was bought and machined. The pressure sensor was pressure fitted in place alongside the electronics and sealed up. A rubber gasket and silicon putty was used to ensure a watertight seal and physically depth tested numerous times at 3.66 m of water.

### Camera Housings

The camera housings were a slight modification from a previous iteration of the vehicle. They are constructed from 3 inch diameter PVC piping and fittings. One end has a straight fitting with a Lexan lens, and the other has a threaded fitting. The threaded fitting allows for easy access to the cameras for maintenance or replacement. Attached to the cap is a plastic shelf that the cameras sit on, that way when the cap is removed, the cameras come out of the housing.

## Acknowledgments

The team would like to thank our academic sponsor, Scott Tippens, for putting up with us through out the year. It has been a rather chaotic year for the team, but he has helped us through it and made sure we have made it as far as we have. We would also like to thank all of our various sponsors for providing us with equipment, time, and space in order for us to construct our vehicle.