AquaTux

Adam Suban-Loewen, Anthony De Caria, David Biancolin,
Victor Zhang, Ritchie Zhao, et al.
University of Toronto Mechatronics Design Association
email: mech.design@utoronto.ca

ABSTRACT

AquaTux is the fifth entry of the University of Toronto Mechatronics Design Association into the annual Autonomous Underwater Vehicle Competition. The vehicle is designed to autonomously complete the set of tasks described in the mission[1]. It is equipped with an IMU, a depth sensor, and two cameras that allow it to navigate in its environment. This year's frame was reused from previous years. We are proud to present new electronics, a new embedded system using an FPGA, and new software.

1. INTRODUCTION

The construction of the vehicle that our team will present at the 2014 Autonomous Underwater Vehicle (AUV) competition encompasses several new achievements. We explain the context of our realization by first describing the main objectives of the competition and how our team is organized.

1.1. Mission Overview

The objective of the competition is to build an autonomous underwater vehicle to perform various tasks involving image recognition, and passive sonar to interact with its environment. For the vision-guided tasks, the vehicle must pass through a validation gate, hit buoys, follow pipes, pass through set of pipes, drop an object in a box, launch a projectile through a hexagonal target, and manipulating a wheel. The sonar tasks consists of grabbing an object on top of an acoustic pinger and surfacing above it. The details of the required tasks for the vehicle are described in the mission specification on the AUVSI website[1]. Through the design and testing phases, emphasis was placed on performing a subset of these tasks well rather than attempting all tasks.

1.2. The Team

The team consists of twenty members that are divided into three subsystems: mechanical, electronics, and software. In order to maintain a high degree of integration throughout the project, members are encouraged to work in projects that span multiple subsystems. Our hardware is obtained through sponsorship and funding from our university (see Section 8).

Figure 1 illustrates the major components of the AUV and the reader should refer to it often as we describe the innovative aspects of those components in the following sections.

2. FRAME DESIGN AND CONSTRUCTION

The current frame design is mostly unchanged from previous years. Under the vehicle's operating conditions, we determined that the effects of fluid dynamics were negligible in terms of

[1]http://www.auvsifoundation.org/foundation/competitions/robosub/

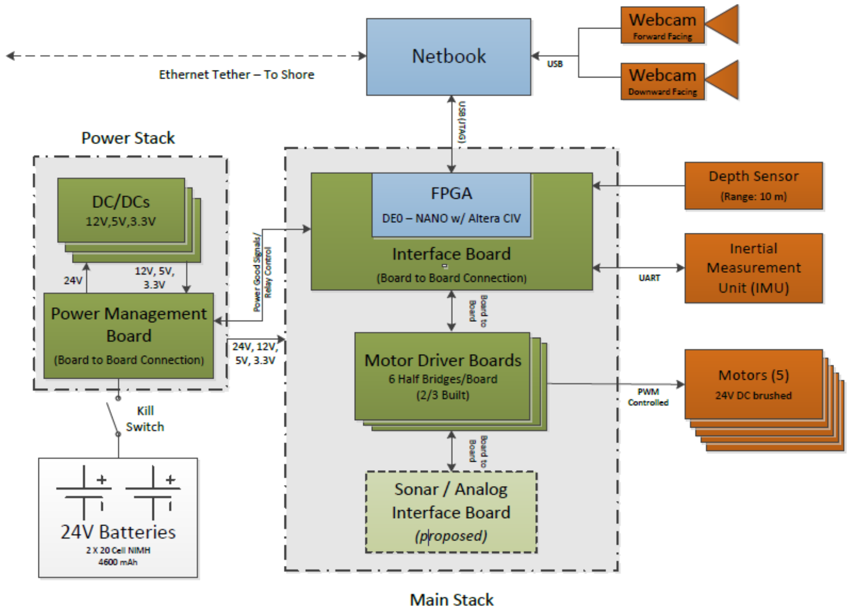


Figure 1: Organization of the AUV modules and their connectivity.

functionality. For this reason, a "design for manufacturing" development methodology was adopted: in efforts to minimize cost and machining time, the frame has been designed to take advantage of commercial off-the-shelf parts as well as capitalize on simple manufacturing processes.

2.1. Design Considerations

The final design was conceived such that it could be fabricated without the need for outsourcing.

Below is the list of the desirable attributes for the frame that we set out to obtain based on the experience gained from past frame designs:

1. Minimize time required to access batteries and electronics;
2. Maximize modularity for easier transport and progressive revisions;
3. Optimize mass for transport, functionality and points;
4. Provide a secure removable internal rack for electronics to facilitate bench testing;
5. Obtain a a waterproof seal with a simple low tolerance assembly;
6. Create solid harness points and handles;
7. Facilitate movement through water and maximize battery life with a neutrally buoyant design;
8. Compensate for changes in mass distribution and motion calibration with adjustable thruster positions;
9. Design an attractive and professional looking vehicle.

2.2. Frame Overview

The AquaTux frame design can be divided into three subsections: external frame, hull, and internal rack. The goals based on above suggested the need to minimize the number of sealing surfaces and reduce the number of waterproof

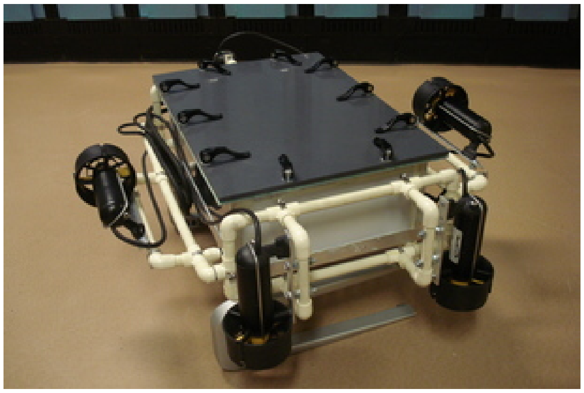


Figure 2: Adjustable PVC tube exoskeleton frame.

Figure 3: O-ring displaced to show the grooves and nylon dowel.

connectors. To achieve these objectives, a single enclosure was created (the hull) to encapsulate everything except for the markers to be dropped, the motor thrusters and the hydrophones. These external components would be connected to an exoskeleton type frame which allows for flexible mounting arrangements. Finally an internal rack was designed to hold all the electrical components including cameras and embedded computer.

2.2.1. External Frame

The external frame was designed with transportation and fabrication as the primary constraints. This exoskeleton is made entirely of commercial off-the-shelf parts, $\frac{1}{2}$ inch PVC plumbing pipe and connectors. The thruster positions are fully adjustable since they are attached to individual pipes fixed in place by standard gear clamps (Figure 2). The adjustability of the frame also facilitates transportation as the entire structure can be collapsed to fit a smaller envelope.

2.2.2. Hull

The hull design attempts to minimize the number of sealing surfaces while maintaining usable space inside the vehicle. In order to seal the hull, a lid was made with a special simplified O-ring groove (Figure 3). To avoid the need for precision milling or numerically controlled machines, an oversized rectangular groove was cut. The corners were then drilled out and replaced with nylon dowels to prevent the sharp corners from cutting into the O-ring. This design eliminated the need for specialized end mills or the need to mill curved patterns in the lid.

Past experience suggested the need for a quicker, safer method to access the internal components of the hull. A torque wrench, used with nuts and bolts provides a fast, foolproof means to compress the O-ring, creating a water-tight seal.

An endplate was designed to provide a single replaceable surface to accept all the connectors between the interior and exterior of the vehicle. This endplate is sealed with a groove similar to that of the lid and is attached with self sealing screws from APM Hexseal. The waterproof connectors which pass through the endplate lead to two master-disconnects which allow for quick and easy installation of the internal rack (Figure 4). The waterproof connectors chosen are 8-pin Neoprene molded connectors from Impulse Enterprise. These connect the thrusters as well as the wireless tether and provide maximal flexibility due to their pin count as well as small profile. The pressure readings for the AUV are also gathered through the endplate. In order to sense depth, a MPX4115 pressure transducer is coupled to a segment of PVC tubing connected to a compression nut on the endplate. The depth of the vehicle corresponds to the pressure of the air trapped in the PVC tubing. The hull is fabricated from $\frac{1}{8}$ inch thick polycarbonate sheets, with waterproof epoxy bonding the joints together. To strengthen the bonds, aluminum angle brackets were added to the edges to increase the bonding area of the epoxy. The thin, clear polycarbonate walls of the hull allow peripherals such as the cameras, marker droppers and torpedo trigger to operate from within the vessel. Although the marker droppers and torpedo launcher did not make it into this year's submarine, the polycarbonate walls facilitate the addition of the peripherals and a method to actuate them.

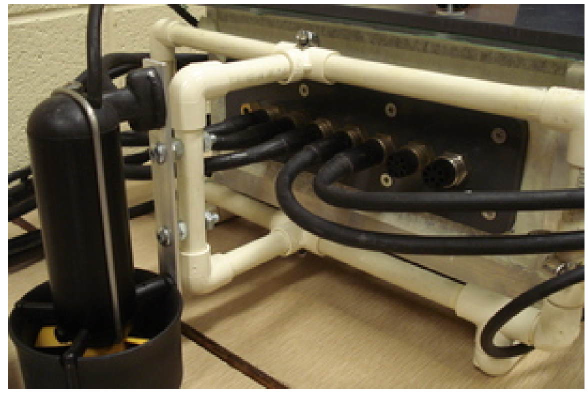
2.2.3. Internal Rack

The frame was designed to be reconfigured and usable in future years. The concept for the internal rack consists of base plates that accepts stacks of electronics. These plates can be removed as a single unit which facilitates wiring, and bench testing. Due to the buoyancy of the hull, stainless steel bars were used to add weight to the vehicle. These weights act as a method of balancing the vessel and allow the electrical components to be easily removed for debugging or modifications. The internal rack does not rest directly on the floor of the hull. In case of a leak, water would collect in the space below the electrical components and be detected with leak sensors prior to any damage.

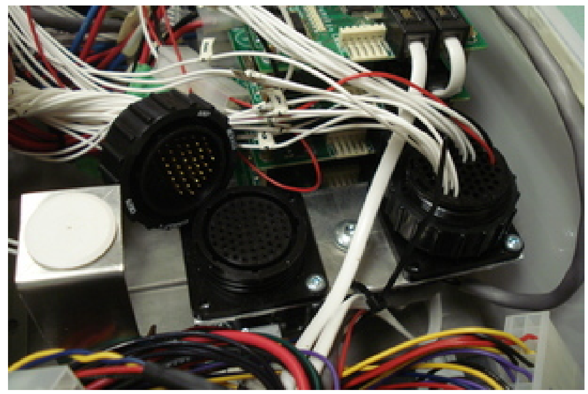
3. ELECTRONICS

Our electronics were designed to handle all low-level tasks that are required to move AquaTux, acquire sensor data from the IMU and depth sensors. The main goal of the electronics is to allow the AUV to be driven with only high-level heading and depth commands. The electronics are responsible for executing the commands and controlling their results (e.g. depth or heading) in a "black box" fashion. The electronics are managed by the embedded system described in the next section, giving access to each electronic peripheral.

Aquatux's electronics are arranged in two stacks of PCBs, one dedicated to power distribution and management, and the "main stack", which breaks out the on board FPGA's I/Os to any number of children.



(a) Endplate showing waterproof connectors.



(b) Master disconnects for the waterproof connectors on the inside of the vehicle.

Figure 4: Connection system to the outside of the vehicle.

3.1. Main Stack

At the heart of Aquatux's electronic subsystem is a Terasic DE0-Nano, a small form-factor FPGA development board, containing an Altera Cyclone IV FPGA. The DE0 mates directly to the "In-

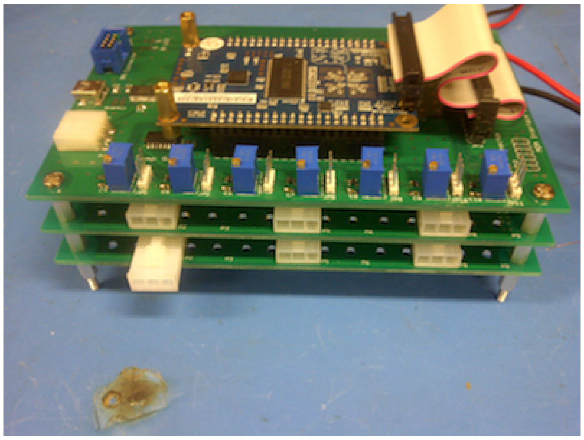


Figure 5: The main stack connector. Top, the interface board and DE0 nano. Below, two 6 channel motor drivers.

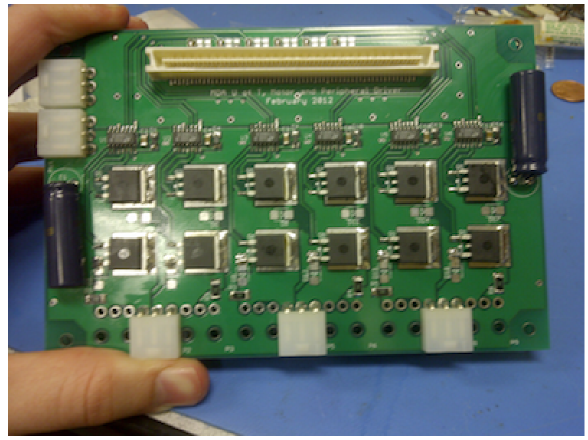


Figure 6: A motor board, configured with 3 full bridges.

terface Board" which breaks out all of the available I/0s to a 120 pin board to board connector (Figure 5). Any number of new PCBs can be mated, by providing the mating male and female connectors. These connectors form a wide bus between the FPGA and the mating boards, which when coupled with the reconfigurability of an FPGA, makes it very easy to incorporate new PCBs without having to physically rewire parts of the submarine – a Quartus recompile is all that's required. The interface board also connects to the IMU, a VectorNav VN-100 Rugged, which performs its own sensor signal processing to output attitude data to the FPGA.

3.2. Motor Board

The motor boards are used to control the Seabotix BTD150 Thrusters used on AquaTux and other actuators internal to the submarine. Each motor board (Figure 6) consists of 6 NMOS high-side half bridges driven by Linear Technologies LT1660 half-bridge drivers. Each half bridge has a sister with which it can be used a full bridge, by soldering the output connector in the correct position. Complementary PWM inputs to each driver can be sourced from one of three sets of I/Os provided via the main stack connector, allowing a multiplicity of boards to be mated to the stack. Presently, two of three boards are used, providing the five full bridges required to drive Aquatux's motors. The final board is provision to drive internal peripherals such as marker droppers in future competitions.

3.3. Power Distribution

Power for Aquatux's electronic systems, excluding the netbook, is provided by two 24V NiMH battery packs. The power distribution board switches the battery voltage with a relay, which is controlled by the FPGA's soft-processor. Throwing the kill switch (by removing the velcroed handle on the side of the hull) opens a reed switch which breaks the relay's coil current. Three DC/DC converters (3.3V, 5V, 12V) source from this voltage, and form a stack of PCBs with the power management board above them. LC output filters on power management board remove switching harmonics, before breaking out the rail voltages to the submarine through a series of copper busbars. Four pairs of analog comparators alert the FPGA when any of the rails over or under-volts by 5 percent of their nominal voltage. As a final baseline precaution, breakers have been inserted between the batteries and relay, and between the power management board and bus-bars for each rail.

3.4. Power Source

The NiMH battery packs were custom designed by our team: each pack is composed of 20 SY136T Sanyo nickel metal hydride batteries. All batteries are connected in series to form a 24V nominal battery pack with a capacity of 4100mAh. Temperature sensors embedded in each pack allow the FPGA to kill the sub if any one pack overheats.

3.5. Leak Sensors

New, custom-etched leak sensors enable AquaTux to pre-emptively surface upon detection of a leak. The leak sensor design, and etching process also provided an excellent teaching tool for our younger members.

3.6. Sonar Board

The newly designed sonar board is used to pre-amplify, filter, and digitize the signals from our hydrophones. There are four pre-amplification and filtering channels and two analog to digital conversion channels (as each ADC is dual-channel). The digital output is buffered, before being sent to the main stack connector to communicate with our FPGA over SPI. Additionally, the board serves as an interface for our IMU connectors, as well as power monitoring and general analog input.

A bandpass filter was designed using the Sallen Key topology to attenuate low frequency noise generated from our thrusters, and provide a maximum gain at the frequencies of interest, i.e. the frequencies of the RoboSub SONAR pinger (Figure 7). Our ADC, Analog Devices AD7264, 1 Msample/s, 14-Bit, Simultaneous Sampling ADC with programmable gain provides a flexible means of digitizing and communicating with our FPGA.

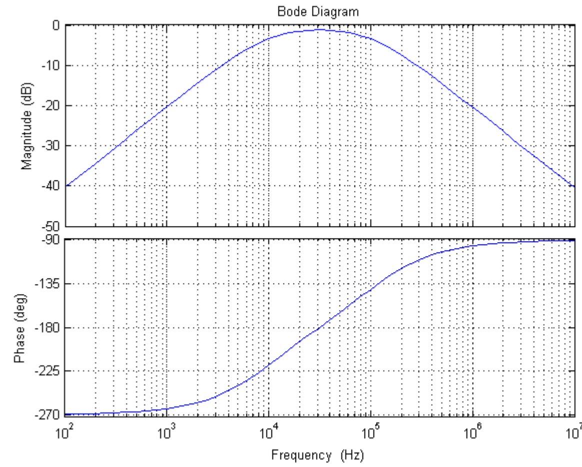


Figure 7: Bode plot of Sallen Key bandpass filter transfer function, note the bandwidth and centre frequency corresponding with the frequency of the pinger.

It was selected for it's high precision, high-speed sampling and communication interface, as well as for it's high input impedance, low signal-to-noise ratio, and programable gain amplifiers which allow us to scale our digital signal according to the distance from the pinger.

Placement and routing was optimized according to standard high-speed signal PCB design rules: we use power and ground layers, our high-speed analog and digital signals are kept far away from each other, and trace impedance is designed as small and consistent as possible to minimize noise and signal reflections (Figure 8).

4. EMBEDDED SYSTEM

The embedded system was designed to link the submarine's computer with all electronic peripherals, with a high-level interface to access groups of peripherals. The embedded system consists of a DE0-Nano FPGA, synthesizing custom circuitry to communicate with each peripheral and a soft processor to communicate with the com-

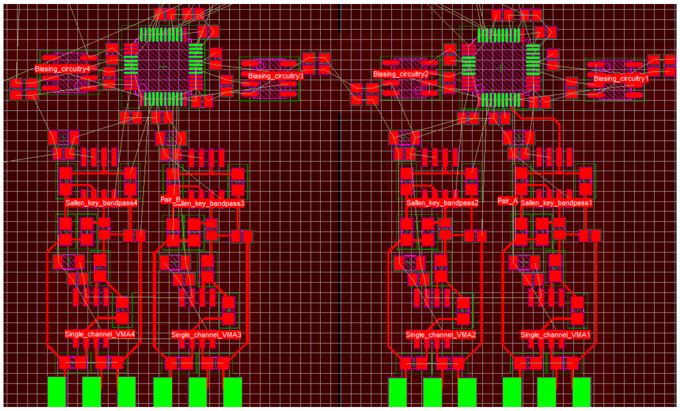


Figure 8: PCB layout of the Sonar Board, showing a channel for each hydrophone.

puter. The FPGA was chosen for its quantity of general-purpose I/O pins, allowing it to manage all electronic peripherals, and its compact size of 3x2 inches. The FPGA is the only programmable element in the submarine besides the computer, reducing the amount of communication between programmable elements.

4.1. FPGA Soft Processor

The FPGA Soft Processor exposes high-level commands to the computer to access each group of electronic peripherals. The IMU and depth sensor provide attitude readings and are used by the PID controllers and the computer interface. Power management continually monitors each voltage rail, and will power off the submarine if any voltage is invalid. The five motor drivers are controlled together by the soft processor's yaw, pitch, roll, and depth PID controllers. The computer interface can set target speed, yaw, and depth, which are inputs to the PID controllers. Speed is implemented as an offset because the submarine does not have velocity measurement. The PID controllers are implemented on the soft processor because they need immediate access to the motor drivers and attitude readings. The high-level commands are text-based, allowing manual control to test each peripheral individually. Future plans for sonar processing will be implemented on the FPGA, using the soft processor and built-in DSP elements.

4.2. FPGA Hardware

The FPGA also synthesizes custom hardware to communicate with peripherals through general-purpose pins. This flexibility can support any protocol to communicate with peripherals, including SPI and UART. Each peripheral protocol is a separate module, memory-mapped to the soft processor across Altera's Avalon Bus. The DE0-Nano has 85 general-purpose pins for digital inputs and 8 ADC inputs. The depth sensor is an analog peripheral, and all other peripherals communicate digitally. The hardware and soft processor run at 50 MHz.

4.3. Sonar Signal Acquisition and Processing System

A newly designed SPI controller module acquires digital signals from the Sonar Board. It uses high-speed serializers and deserializers, following a 34 MHz PLL clock, to transfer data from the SPI master-out-slave-in (MOSI) and master-in-slave-out (MISO) lines. Enables are generated with 16-bit counters, and set according to the ADC's serial interface. Despite SPI providing a multi-slave capable bus, we implement one SPI controller for each ADC, ensuring samples are not lost when an ADC is inactive (Figure 9).

The signal processing system uses the cross-correlation operation on each combination of hydrophone signals to determine the point at which they are most in phase. From here, we can generate the time delay of sound arriving at subsequent hydrophones. Given this information, and knowledge of the hydrophone array geometry - a simple 10 cm per edge square configuration - we determine the attitude towards the sonar pinger.

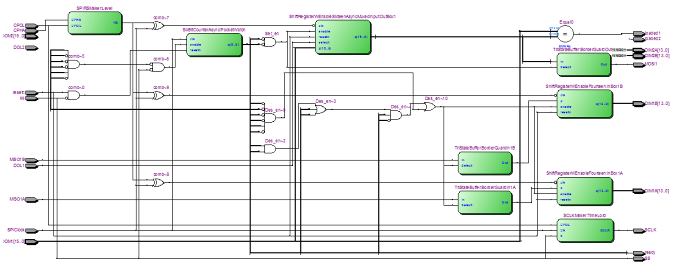


Figure 9: Schematic diagram of the SPI controller module, showing serializers and deserializers.

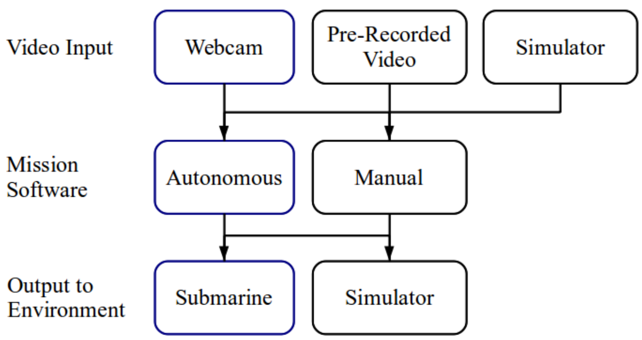


Figure 10: Software module organization.

5. SOFTWARE

The submarine's software is aware of the competition objectives, and performs computer vision to command the submarine to move through the embedded system. The software is organized so that it is easy to test the core vision and control systems with different sources of video input and to command the submarine or the simulator to move.

5.1. Software Organization

The software is broken down into 3 components (Figure 10): (i) the video input; (ii) the mission software gives physical commands to move the sub; and (iii) the environment that reacts to the physical commands. The video input

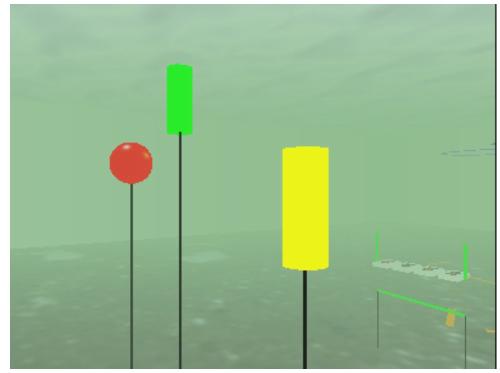


Figure 11: Example image from the simulator showing pipe segments and the buoys in turbid water.

can come from the submarine's webcams, prerecorded video, or the simulator (Figure 11). The mission software can be the autonomous algorithms that drive the submarine or a manually controlled user interface. The environment that reacts to physical commands can be the submarine or the simulator. The manual user interface allows us to act as a passive observer, constantly requesting the state of the vehicle, or as a controller sending commands to the AUV.

5.2. Simulator

The simulator is a piece of software written in C++ using the OpenGL library. Its purpose is to simulate the competition grounds so the software and vision algorithms can be tested with greater ease (Figure 11). The simulator can accept commands to move the camera viewpoint, so that we can properly simulate video feedback when the submarine is commanded to move.

5.3. Software Control

Our control algorithms are run the submarine runs autonomously. The entire mission is broken down into tasks, such as the gate or buoy task, which are implemented in separate modules. The control code queries the vision system for circles, lines or rectangles, and translates the vision result into a movement command. While the submarine is reacting to a movement command, the video input is ignored to prevent overshooting or oscillating the target. Each control task can be tested separately.

6. VISION

The machine vision system is defined as all software and hardware which converts images (the input to the system) into numerical data (the output). This data refers primarily to the existence, position, orientation, and color of various objects in the image, such as the starting gate or a colored buoy. This information is handed off to the rest of the software, which acts on it to determine the correct course of action for the submarine. Because the internal representation of an image is simply a large matrix of numbers, the vision system's job can be thought of as filtering this large volume of mostly extraneous data to a few pieces of useful data, which the control system can then analyze.

6.1. Vision Hardware

The hardware for the vision subsystem consists of two cameras and a small netbook. Both cameras are Logitech Webcam 9000 Pro webcams, one of which points forward, and the other of which points down. These cameras were chosen for their price (about $50 each) and the availability of Linux driver support. The netbook used is an HP Atom netbook. This model was chosen for its small size and low cost.

6.2. Vision Software

All of our image processing code is written in C++, using the OpenCV library. A large number of algorithms were tried and discarded during the development of our final flow. Due to the scope of this document only the final algorithms used will be described.

The image processing flow can be broken up into three distinct parts: image segmentation (separating a 3-channel image into 1-channel pieces based primarily on color), object recognition (classification of each segment as background or target) and property calculation (deriving the object's properties such as location or color). The algorithm used for each step is distinct and will be discussed below.

Image segmentation is performed using OpenCV's implementation of the Watershed algorithm. The algorithm works iteratively from a collection of segmented pixels (seeds) and at each step associates the most similar pixel adjacent to a segmented pixel with that segment. It returns a grayscale image, with different gray levels corresponding to different segments. Each seed pixel has a value associated and two seeds of the same value will contribute to a single segment.

The performance of the Watershed algorithm is heavily dependent on the quality of the seeding. Optimally we want at least one seed for each different object in the image and at least one seed for the background. A single object should not have two different valued seeds, as this will cause two segments to emerge. To seed the algorithm, we take a random sample of pixels from the source image, and run K-Means clustering to cluster the samples based on color similarity. The results of the clustering is a minimum collection of pixels which are representative of the color space of the image. The clustering results used to assign values to the seeds - similarly colored seeds will get the same values (Figures 12 and 13).

Object recognition is made simple due to the fact that only circles and rectangles need to be recognized for the tasks we are aiming to do. OpenCV can find the minimum enclosing circle and (rotated) box for any segment, and a simple comparison of area, perimeter, and various symmetry properties can be performed to check if the segment is similar to its enclosing shape. For example, a the minimum enclosing circle of a non-circular segment will leave large amounts

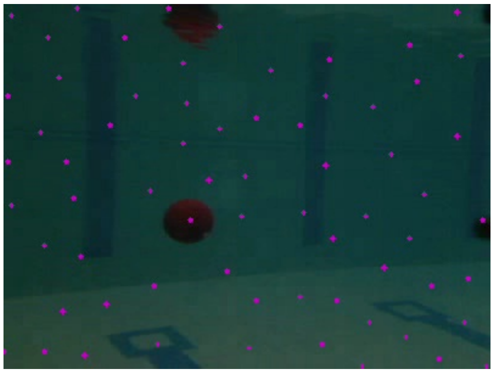


Figure 12: Example image at a test pool, with seed locations overlayed.

Figure 13: The test image segmented.

of blank space, and will therefore have a very different area compared to the segment.

Property calculation is done on a target-by-target basis. For most targets (the buoy, for example), it is sufficient to know the position and characteristic dimension in pixel space. We can then use these numbers, along with the camera's field-of-view and the physical length to calculate the position and range of the object in real space. For the pipes at the bottom of the pool, the position angle is also needed. All of the pixle properties are easily extracted from the enclosing shape derived by OpenCV. To identify an object's color, we convert the BGR color of the object into HSV, which is more robust against changes in brightness, and use the resulting hue value to determine the color of the object.

Finally, in order for the vision code to return stable results and be robust against noise, we store the objects identified in the current frame begin processed as well as a number of frames prior. We use this to average and filter the numerical information from multiple frames. This allows us some room for error - a single frame with poor segmentation or a stray object will not destroy the system's performance.

7. CONCLUSIONS

This year's team took on the challenge of building new electronics, digital hardware and software. This AUV is composed of a Netbook that is linked to a single FPGA that drives the electronic peripherals. The Netbook runs mission-aware vision software, which uses the OpenCV library. Circuit board fabrication, plastic machining, embedded programming, and software engineering are some of the skills that many of our members were able to acquire. More rational design techniques and project management skills are also derived from the our previous competitions and they had a significant effect on maximizing the productivity of all the man-hours contributed to the project.

8. ACKNOWLEDGMENTS

We would like to thank the staff of the University of Toronto for supporting our project.

We would also like to thank all our sponsors who have helped make this project possible: The University of Toronto Engineering Alumni Association, The Edward S. Rogers Department of Electrical and Computer Engineering, The Division of Engineering Science, the University of Toronto Engineering Society, and Altera.