Mechatronics Design Association Design and Implementation of Tempest Autonomous Underwater Vehicle

Tasdiq Ameem, Shai Bonen, Chris Chan, Michael Chiou, Kathryn Darnell, Iryna Gomenyuk, Michael Gu, Lin Han, Xu Han, Ying Han, Albert Hsueh, Ervin Ibadula, Andrew Kim, Chang-Jie Lu, Richard Seo, Alex Simpson, Cathy Su, Chris Su, Jonathon Wong, Wentao Xu, Linda Yu

***Abstract*—Striving to learn and expand, the Mechatronics Design Association (MDA) has developed an autonomous underwater vehicle (AUV) named *Tempest* to compete in the RoboSub 2016 competition. The design focuses on robust essential functionalities and easy implementation of future expansions. Tempest features unique software architecture for controlling its movements autonomously in addition to a flexible electronics interface controlled by a field-programmable gate array (FPGA). Tempest interacts with its environment with six thrusters and senses its surroundings through cameras, depth sensors, and an inertial measurement unit (IMU). Tempest is a part of MDA's new rebuilt infrastructure designed to be more efficient at utilizing club resources.**

I. Introduction

The Mechatronics Design Association (MDA) is a team of undergraduate students at the University of Toronto (U of T). MDA presents Tempest, an autonomous underwater vehicle (AUV) designed to compete in the RoboSub 2016 competition. As the team's 5th entry into the RoboSub competition, Tempest is the accumulation of years of experience and hard work from individuals inspired by the field of robotics.

Tempest's design objective is to achieve basic functionality, build an infrastructure for future design improvements and club expansion. The team has designed an AUV that incorporates a well-organized software base, a spacious hull interior, a rugged frame for attachments and a modular circuit board design featuring programmable hardware in the form of a field-programmable gate array (FPGA).

II. Mechanical Systems

Tempest's mechanical design places an emphasis on robustness and modularity. The mechanical systems consist of an acrylic hull, two end caps, an aluminum frame and six thrusters. Each component is designed using SolidWorks and simulated using ANSYS FEA software to determine faults and weak points. Components are manufactured through processes such as water jet cutting, laser jet cutting, and 3D printing. Small and minor–components are machined by hand while larger and design-critical components are manufactured using CNC machinery to provide a cost-effective balance.

A. Frame

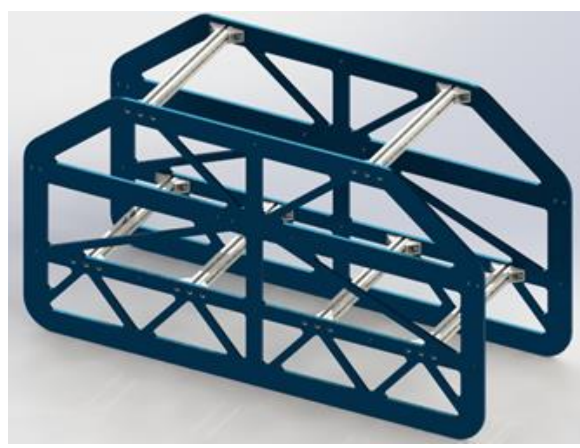

Fig 1. SolidWorks render of the aluminum frame

The frame consists of two ¼ inch aluminum 6061 water jet plates with six 80/20 aluminum extrusions connecting the two plates together. The plates provide a solid support structure to the AUV with a good weight-to-strength ratio relative to other design alternatives. The 80/20 extrusion cross beams allow secure and easily accessible mounting points for all of Tempest's mechanical components. The plate and support beams are designed for ease of assembly while simultaneously providing protection for fragile components such as the acrylic hull.

B. Hull

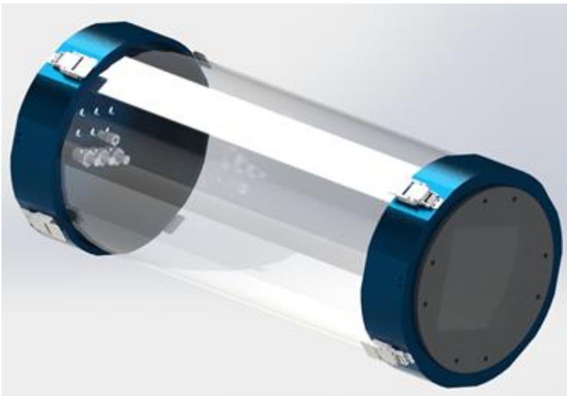

Fig 2. SolidWorks render of the hull with end caps attached.

The hull consists of an acrylic tube 1 foot in diameter and 2 feet in length with an end cap and support ring on each end. This tube's responsibility is to accommodate all necessary electronics and a netbook for operation. Each end cap and support ring, manufactured using water jetting, is attached on the ends of the hull to mitigate any non-hydrostatic stresses and provide adequate mounting points for the hull. The hull is secured in the two support rings using latches, and the entire assembly is mounted onto the frame.

C. End caps

The end caps are designed to provide both mounting points and waterproofing functionality. Each end cap is machined by hand from stock 12 inch aluminum rounds, and implements a double O-ring design to create a reliable waterproof seal. The front end cap incorporates a laser jet manufactured acrylic plate. This provides an unobstructed view for the front facing camera mounted on the interior of the hull. The back end cap acts as the electrical interface for internal and external electronics through Teledyne waterproof connectors.

D. Thrusters

Tempest uses six BTD150 SeaBotix thrusters to propel itself underwater. Two thrusters are mounted directly onto the frame facing forward, allowing forward and backward motion as well as yaw. The remaining four thrusters face downwards to allow vertical movement, roll, and pitch. These thrusters increase stability of the AUV by adjusting the speed of each thruster to compensate for weight distribution issues. The thruster configuration provides five degrees of freedom, sufficient to accomplish most tasks in the competition.

E. Interior

The interior has two shelves providing ample space to place and organize all electronic components. The interior is constructed using ¼ inch polycarbonate connected using angle brackets, nuts, and bolts to remain rigid under the weight of the heavier electronics. The interior structure is secured to the hull with drawer slides such that the shelves can slide out of the hull. This also provides easy access to the electronics for testing and repairing.

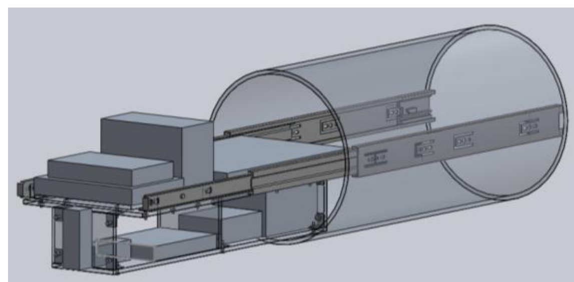

Fig 3. SolidWorks render of the interior design

III. ELECTRICAL SYSTEMS

The electrical system's objective is to power and control Tempest's peripherals and sensors. To achieve this, the electrical system is divided into several components: the power board, interconnect board, FPGA system-on-chip (SOC) board, netbook and two motor boards. The circuit boards are designed using the Altium printed circuit board (PCB) layout editor with modularity at its design focus in order to easily accommodate additional improvements in the future.

A. Power Source

Tempest utilizes custom-made rechargeable battery packs composed of twenty SY136T Sanyo nickel metal hydride (NiMH) batteries, which provide a total of 4100 mAh at a nominal 24V. These batteries power all electronics with the exception of the self-powered netbook.

B. Power Management Board

The unregulated voltage from the battery is routed to DC-DC converters and passes through LC filters to obtain various required voltage supplies. The motor controller boards’ voltage supplies are gated by a mechanical relay enabled by an FPGA controlled solid state relay (SSR) and a magnetic emergency kill switch. The power supply to the FPGA and sensors is not gated to allow continuous data collection after the kill switch has been activated. Each supply is split to their corresponding PCBs via copper bus bars with a manual enabling switch for each bus bar. Four pairs of analog comparators are used to detect voltage deviations greater than 5% in each rail. As a final precaution, breakers are inserted between the batteries and relay, and between the stack and bus bars.

C. Interconnect Board

The interconnect board is designed to interface the electrical systems with the FPGA controller in an organized manner. Unlike previous designs which used a main stack header, this year’s iteration features multiple smaller headers with individual wiring to allow other boards to be placed separately and have each wire routed individually within the AUV. This increases flexibility of the design by allowing new boards to be designed relatively independently from the existing systems. Another key decision was to design the interconnect board such that the FPGA SOC board can be mounted on top of the interconnect board to conserve space.

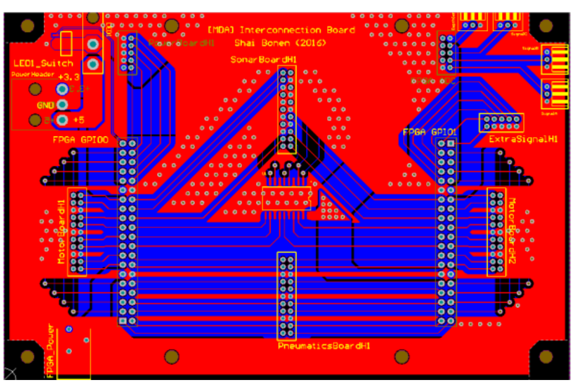

Fig 4. Layout of the interconnect board.

D. FPGA SOC Board

The Terasic DE0 Nano SOC board with its Altera’s Cyclone V FPGA serves as the connection between the netbook and the electrical systems. The FPGA is advantageous for interfaces due to its customizable and centralized hardware, which increases flexibility and allows for easier debugging. This board is chosen due to its large number of IO and gate arrays in addition to large program memory for loading more data onto the SOC. The SOC features a built-in LTC2308 analog to digital converter (ADC) which is used to interpret Tempest’s depth from a pressure sensor.

For issuing commands to the FPGA, the 11.6’’ Asus Vivobook X200MA netbook is chosen for its affordability, small size, and self-sufficiency in terms of power, cooling, and hardware. The FPGA and webcams are connected to the netbook by USB. The laptop is remote controlled through an Ethernet tether SSH X11 connection using an ARCFOUR protocol to enhance the transfer speed. The netbook sends Tempest’s target depth and orientation to the FPGA. Subroutines on the on-board NIOS-II soft processor schedule when to read inputs from the hardware and netbook, and uses control algorithms to map the target to actionable pulse-width modulation signals that drive motors.

E. Peripheral Motor Driver Boards

The purpose of the two motor driver boards is to control the six thrusters on Tempest. This is accomplished by using pulse-width-modulation (PWM) signals to control the speed and direction of each thruster. Each motor driver board consists of 8 NMOS high-side half-H-bridges driven by LT1160 half H-bridge drivers. Each half-H-bridge can be operated independently or form a full-H-bridge when combined. Future peripherals, such as additional thrusters, marker droppers, or torpedo launchers can be actuated with the same interface boards in this configuration.

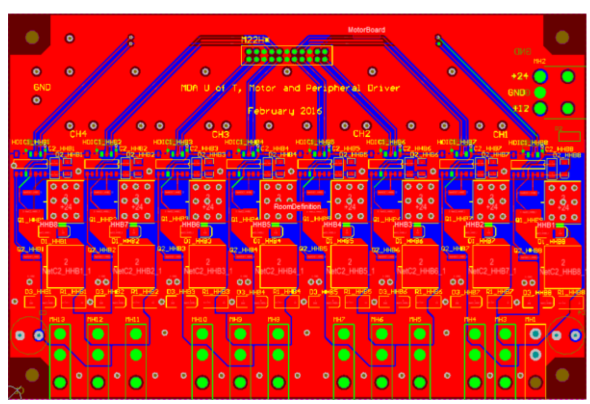

Fig 5. Layout of the peripheral motor driver board.

IV. SOFTWARE

Tempest's software is organized in a Model-View-Controller (MVC) framework built upon the OpenCV and Qt libraries. Coded in C++, it is designed to be adaptable and user-friendly using Agile development principles. Tempest also includes a custom simulator to allow for virtual testing of the software algorithms.

A. Code Framework

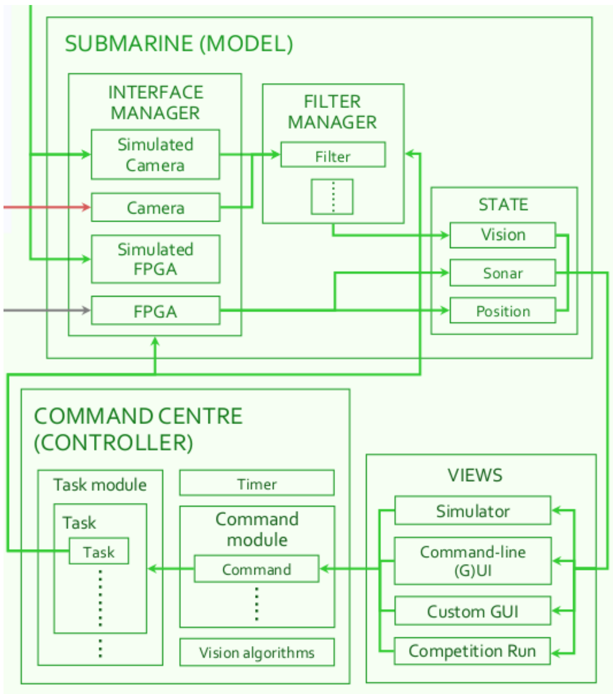

Fig 6. Brief MVC component diagram of Tempest on-board software

The MVC architecture allows for more modular code. The modularity decreases the learning curve for new developers–as existing code can be treated as black boxes. Furthermore, the structure of MVC inherently allows for clearer documentation.

The Model interfaces with external devices and stores their data. Separate interfaces for the real and simulated devices allow the Controller to execute tasks regardless of which environment it is working in, minimizing the changes needed to switch between real and simulated testing.

The View handles user interactions with a client-side graphical user interface (GUI), which improves testing efficiency by making it easy to issue commands.

The Controller schedules tasks to be run in series. A hierarchical view of commands is reinforced by having complex tasks contain simpler subtasks, thus increasing the overall reusability and modularity of the code.

C. Task Algorithms

The task algorithms are designed to control the movement of the submarine. They receive video feeds from the front and bottom cameras as well as data from the FPGA. The video passes through a hue saturation value (HSV) filter, which isolates features that are within preset color thresholds. The filtered video is then passed to other filters to undergo shape recognition.

The task algorithms use the geometry and position of the detected shapes as well as the data from the FPGA to execute motor commands.

The figures below present the process of completing the path task.

Fig 7. The HSV filter converts the original image (left) into the resulting image (right)

Fig 8. The red rectangle is detected by the shape filter and overlaid on top of the original image

D. GUI Features

Fig 9. Users can trigger commands and check the IMU through the GUI

Clicking on the GUI buttons prompts the Controller to execute tasks. The user interface is designed to assist in task testing at all levels within both realistic and simulated environments. Testing of high level tasks and simple motor controls is isolated for convenience. Interactive features on the GUI are made using Widgets in the QT 5.5 library.

V. Testing Methods and Procedures

A. Mechanical Testing

The primary objectives of mechanical design testing are to ensure waterproofing and to correct weight distribution for ballasting. The waterproof test consists of three stages: an end cap test, a shallow test and a depth test. The hull goes through an iterative process of having a non-curing sealant applied to the locations of the known leaks, being placed underwater to test for other leaks, and then progressing to the next stage if no additional leaks are found. The multi-stage method allows the team to perform waterproof tests more frequently without having to reserve a pool, which is expensive and difficult to schedule. The multi-stage testing also reveals leaks that occur only at greater depths.

Once the interior is guaranteed to be waterproof, the electronics are placed in the hull to determine their weight distribution. External weights are attached to the frame until Tempest is neutrally buoyant at the target depth.

B. Electrical Testing

The electronics are tested through debugging and iterative reliability testing. The IMU, used by Tempest to determine orientation, is tested by holding the unit in different orientations and reading the respective values it outputs. The depth sensor is first dry tested by observing changes in the values on the ADC when the sensor is subjected to varying pressures. The sensor is then tested in the water by confirming the linear pressure increase with depth.

The motor drivers are tested extensively in order to guarantee reliability in the water by running them repeatedly in different speed and direction combinations. Testing revealed that the previous method of switching between forward and backward drives on the motor driver H-bridge was unreliable. In response, the new method of switching between the desired directions and braking was developed to increase reliability.

C. Software Testing

Tempest's software testing system incorporates an in-house logger which allows for customizable output for easy debugging, and a simulator which creates a virtual environment to test the task algorithms.

The logger can be treated as an augmented print statement with additional features such as timestamps, logging levels, and writing to file. Logging also helps with code readability, which is an industry expectation.

The simulator runs on a thread separate from the GUI to provide visual feedback in a virtual environment. The team attempts to mirror the pool environment with first-person-like camera views, a dimension-appropriate mock arena, and simulated FPGA for motor control. The simulator is built using the 3D game engine Irrlicht. Irrlicht is chosen due to its C++ libraries, allowing for straightforward integration with the rest of the Tempest code. Irrlicht is also lightweight, allowing it to run fairly fast on less powerful laptops.

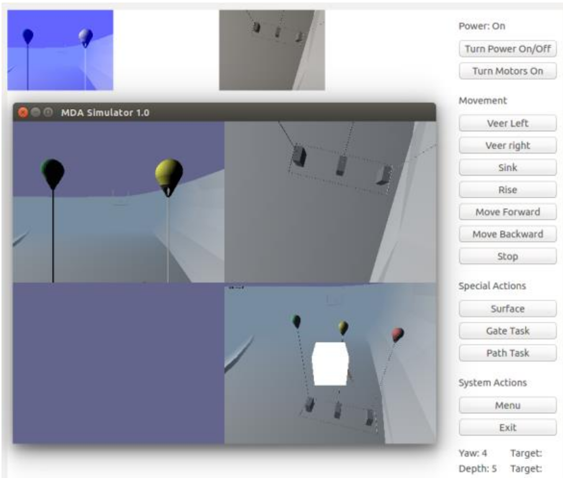

Fig 10. Simulator and GUI exchanging interactions, changes are reflected on GUI's filter views (top) and Simulator's cameras

VI. Conclusion

Tempest is a modular AUV developed by U of T MDA. Following its predecessor, SubZero, the overall goal was to develop a robust, spacious, and modular design that is adaptable and easily incorporable into next year's design cycle.

Future members will be able to focus on improving the AUV by building new features on top of the foundation built by Tempest to better accomplish each task in the competition. The valuable experience in complex mechanical, electrical, and software engineering design has been a challenging yet rewarding team-building experience for MDA members who hope to participate in the next RoboSub competition with its new AUV iteration.