Design and Implementation of Cubeception 3

Donovan Drews, Jason Ganton, Ryan Ganton, Rahul Keyal, Jason Ma, Patrick Paxson, Robert Quitt, and Rahul Salvi

***Abstract*—Cubeception 3 is the 2015-2016 iteration of San Diego Robotics 101's cube-shaped autonomous unmanned underwater vehicle (UUV) designed for the 19th International RoboSub competition. Cubeception 3 aims to take advantage of its modular nature and complete a wide variety of tasks given a limited budget and low construction time.**

Notable improvements over previous designs include a reworked core design allowing for separate removal of electronics and the battery, increased optimization of space, multiple specialized Raspberry Pis communicating via Ethernet, and expanded navigation capabilities due to the addition of sonar, computer vision, and improved algorithms.

I. Introduction

San Diego Robotics 101 is a highly interdisciplinary team consisting of members across multiple schools at the university and high school level with the purpose of building autonomous UUVs. This allows for an exploration of the advantages and disadvantages of a cube-shaped underwater vehicle, particularly the highly modular nature and quick build time afforded by this design. San Diego Robotics 101 is also interested in the potential applications of its design to real world underwater vehicles. The process of construction involved CAD modeling, board prototyping, experimental analysis, software testing, simulations, and assembly as well as real-world trials, all of which were done in an approximately 8 month build cycle. San Diego Robotics 101 participates in the annual AUVSI Foundation RoboSub Competition, which takes place in late July and is hosted at the SSC Pacific TRANSDEC in San Diego, CA. This competition puts autonomous UUVs like Cubeception 3 through a wide variety of missions which simulate tasks performed by professional UUVs. These tasks range from passing through a gate to touching buoys, locating pingers, surfacing within octagons, and handling objects, requiring precise maneuverability, mission prioritization software, and navigation capabilities. To achieve as many of these tasks as possible, San Diego Robotics 101 is divided into Mechanical, Electrical, Navigation, Sonar, and Computer Vision subteams.

II. Design Overview

A. Design Strategy

Due to the decreased distance that Cubeception 3 needs to travel, more emphasis was placed on capability and maneuverability over movement speed in its design. As such, Cubeception 3 features a more extensive set of sensors including hydrophones for sonar and cameras for computer vision. This allows Cubeception 3 to more accurately determine its location relative to missions and attempt a wide variety of tasks, including passing through gates, touching buoys, detecting the pinger, surfacing in the octagon, and moving along marked paths. At the beginning of the design cycle of Cubeception 3, some consideration was also given towards including a separate, smaller robot or a claw for attempting tasks involving grabbing objects, although San Diego Robotics 101 determined that sonar and computer vision were higher priority capabilities. Much of San Diego Robotics 101's time was spent researching and simulating sonar and computer vision to determine whether they were viable options on Cubeception 3, and the tests ultimately concluded that sonar was effective in bringing Cubeception 3 within close proximity of tasks, while computer vision would allow Cubeception 3 to precisely navigate through them, making a sonar and computer vision combination the most viable addition to Cubeception 3's design in terms of added capability and point scoring.

The additional complexity introduced with sonar and computer vision capabilities prompted San Diego Robotics 101 to implement a Raspberry Pi network in the hopes of increasing robustness and modularity. Furthermore, significant changes in the sealing of the main core were introduced to reduce failure points as well as human error. The batteries need to be removed relatively often compared to the electronics core, so having a separate battery enclosure with a clear lid for the operator to visually inspect the seal allows for faster and more reliable waterproofing.

San Diego Robotics 101 also has relatively limited manpower and funding compared to other autonomous UUV building teams, so many design strategy choices were made with that in mind, optimizing capability and reliability with a small amount of time and funding.

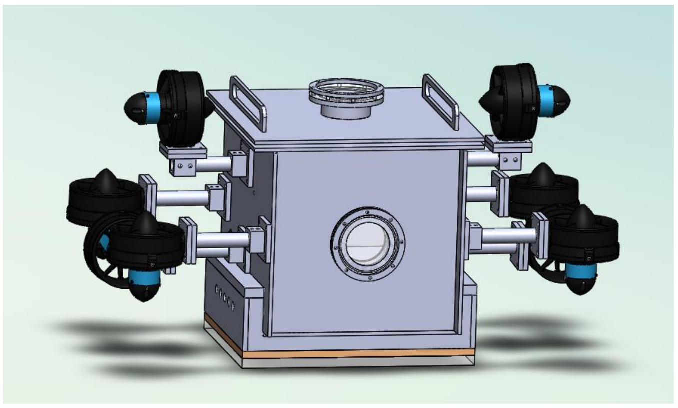
Fig 1. Full model of Cubeception 3.

B. Hardware Overview

Cubeception 3's core is based off of Cubeception 2's core, with the same screw-fastened lid and portholes for cameras to see out of. However, changes in external mounting, a much needed upgrade in thrusters, and the addition of a more maintainable battery enclosure make Cubeception 3 a more robust and maintainable platform than ever before.

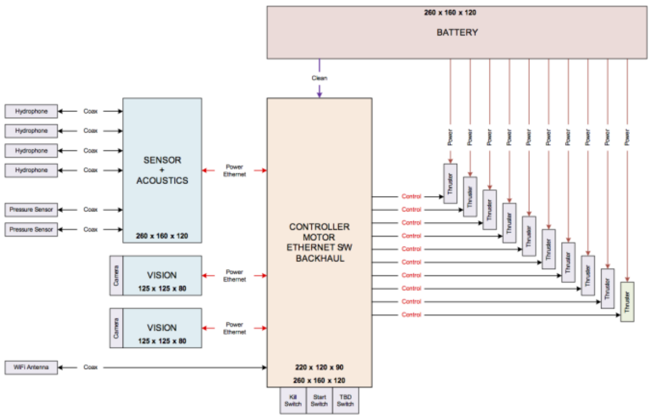

Fig 2. High level overview of Cubeception's hardware.

C. *Electronics Overview*

Cubeception 3's electrical hardware has been completely redesigned for this year. Each of the many custom circuit boards was designed in the EAGLE PCB design suite, then sent out for fabrication to a local PCBA manufacturing business. Many design considerations were made to effectively meet the requirements that were placed on Cubeception 3's design. The final result is an electrical subsystem that efficiently delivers power to and carries signals between the different parts of the robot.

D. *Software Overview*

Cubeception 3's new distributed computation model improves the performance of each individual component and promotes high parallelism in its software functionality. The cluster of Raspberry Pi 2 computers onboard naturally allows many processes to be run at the same time, increasing the overall throughput of the system. Individual computers are given specialized tasks, creating clear divisions between responsibilities. Instead of a "tall" software stack with many processes built upon each other, Cubeception 3's stack is "wide", with many processes sharing information, but able to proceed with the loss of any individual machine.

III. Mechanical Design

A. *Thrusters*

Cubeception 3 features a major upgrade in thrusters over previous designs. One of the major issues in previous designs was the complexity and unreliability of maintaining 24 bilge pumps, which broke often and did not provide much thrust. When deciding on a new system to replace the previous 24 bilge pump system, VideoRay thrusters and Blue Robotics T-100 thrusters were the primary candidates. Both types of thrusters required a significantly greater amount of power, although the VideoRay thrusters would require too much. Therefore, the best option was 8 T-100's with future plans of implementing the VideoRay thrusters as a main source of forward thrust. This system requires only a third of the number of motors from previous designs, and provides twice the amount of thrust, reducing complexity and increasing performance significantly.

B. *External Mounting*

The PVC rectangular prism frame was removed from Cubeception 3 this year, as it did not provide enough benefits to justify its continued use. Although this exposes the core and outer components to potential impacts, it also simplifies cable management for the robot, allows for easier maintenance of the thrusters, and allows the motors to run with less interference against the sensors located on the core. Pylons will extend from the sides of the robot to house these Blue Robotics thrusters, away from the robot and each other.

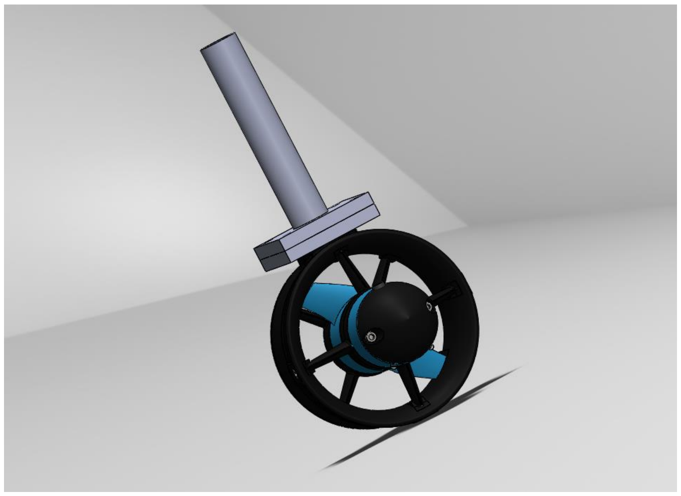

Fig 3. Pylon mount for Blue Robotics T100 Thruster.

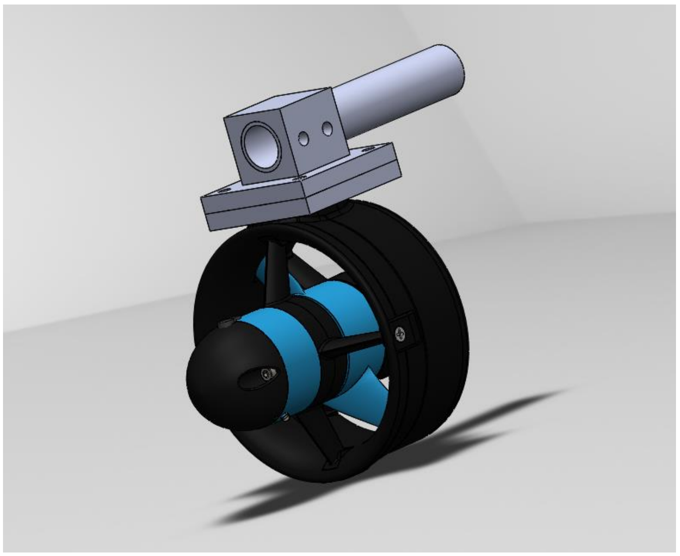

Fig 4. Alternate pylon mount for Blue Robotics T100 Thruster.

C. *Cube Core*

As an improvement over Cubeception 2's core design, Cubeception 3's core features greater ease of access to the batteries while improving the robustness of the seal by replacing threaded holes with latches. With the introduction of a separate battery box outfitted with these latches, the lifespan and reliability of the seal has increased greatly.

The two most appealing options for such an enclosure were two smaller enclosures on the side of the core or a large one integrated into the core. Although the two smaller enclosures offered a more even distribution of weight, the single integrated enclosure was easier to build and required fewer changes from Cubeception 2’s core. More space was also available on the sides of the core for thruster mounting in the single enclosure design. This decision resulted in a trade-off involving slightly increased complexity in software for greatly reduced complexity in hardware.

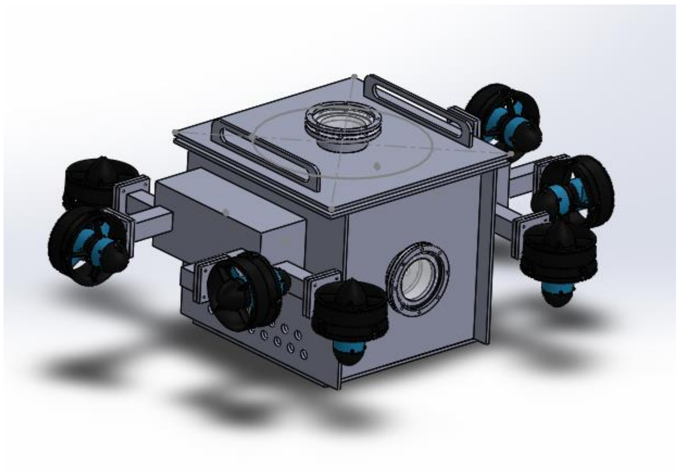
Fig 5. Alternate design with two battery enclosures.

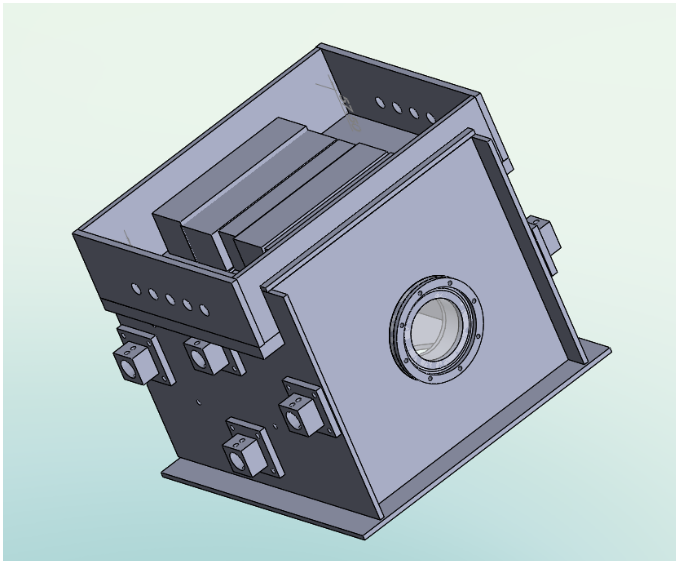
Fig 6. Cube core with integrated battery box on top.

IV. Electronics Design

A. Motor Driver Design

To control each of the many outputs that must be driven on the vehicle, a motor driver shield board is used. The main IC in use to create the pulse-width modulation (PWM) outputs is a MAX6696ATE. The board attaches to a Raspberry Pi and communicates with it over SPI. Software can command any of the 20 channels to produce a specific PWM signal. These signals are either available as singled-ended drive, as in the Raspberry Pi, or opto-isolated drive for use with the motors. This helps eliminate potential ground loop problems that can occur in large systems.

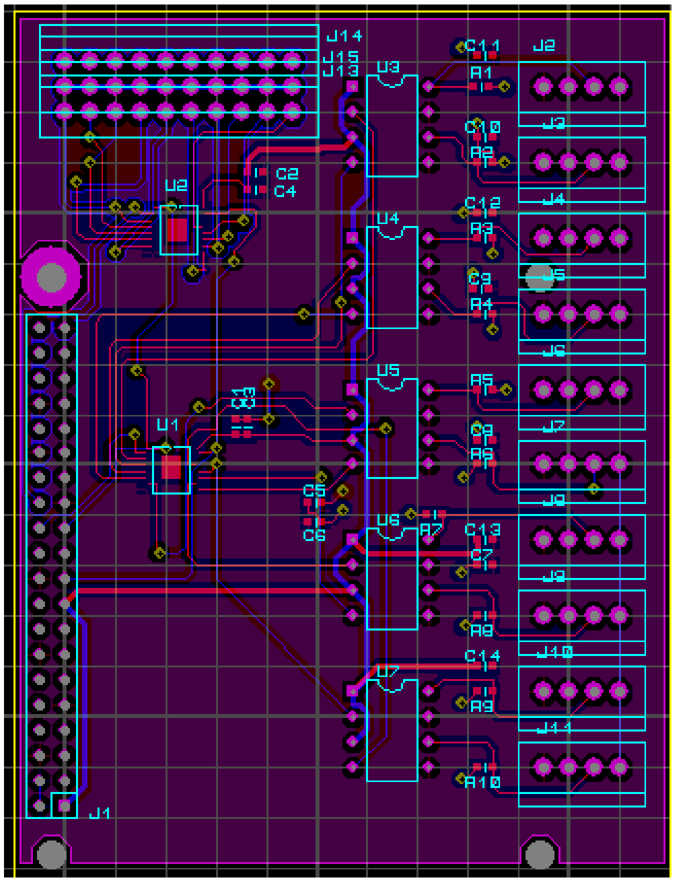

Fig 7. Layout of motor driver shield.

B. Power Distribution Board Design

Efficiently supplying each subsystem with power is done by using a custom power distribution board. The design features several DC converters to take the 11.1V input from the batteries and create 5V, 3.3V, and 1.8V outputs. As the battery voltage is subject to dropping largely when multiple motors are in use, creating large fluctuations, the board splits power between “dirty” and “clean” sides. Dirty power may fluctuate greatly, but clean power is regulated to produce a steady voltage, ideal for sensitive electronics.

C. FPGA Shield Board Design

To address the Raspberry Pi’s lack of high throughput I/O, a custom FPGA shield board was designed. An FPGA was found to be the most suitable system because of its high configurability, making the shield a modular attachment to any Raspberry Pi. The card is capable of both input and output, so it is critical to both the sensor and sonar subsystems. The onboard Xilinx Artix-7 FPGA is programmed in VHDL using Xilinx tools and has been found to be more than powerful enough for each task to which it is assigned.

D. Sensor Board Design

Cubeception 3 features a custom circuit board for polling sensor data from its IMU and pressure sensors. The board has a total of nine STMicroelectronics LIS3DSH accelerometers, nine STMicroelectronics LIS3MDL magnetometers, and four InvenSense MPU3300 gyros, exploiting the advantages of multiple-sensor systems to great effect. Additionally, there are analog to digital converters for the Measurement Specialities UltraStable 300 pressure transducers on the vehicle. A Raspberry Pi is incapable of the I/O throughput necessary to read all the sensors in reasonable time, so the FPGA shield card is incorporated with the sensor board. The FPGA is able to read each sensor while the Raspberry Pi processes the data for other subsystems to consume, resulting in a fast and effective sensing system.

E. Sonar Board Design

To produce sonar pings and listen for their return, Cubeception 3 employs 4 Aquarian Audio H1C hydrophones and a specialized circuit card. The PCB turns a digital ping signal into analog output to drive the hydrophones. Alternately, the card can listen to the analog input from the hydrophone and convert it to a digital signal for processing. Both pathways are bandpass filtered to eliminate undesirable noise introduced by the mechanical systems. Switching between modes has been made extremely quick to allow each hydrophone to be a sender and a receiver, increasing the amount of usable data.

V. Software Design

A. Sensor Data

Cubeception 3's enhanced IMU hardware package has been complemented with upgraded computation software. The signal from each sensor on the IMU is first filtered and averaged with other similar sensors. An unscented Kalman filter is then used to fuse the 9-Axis IMU data into an accurate estimation of the robot's orientation and velocity. Cubeception 3 uses quaternions to represent its orientation, preventing gimbal lock and providing more robust estimation overall.

For depth estimation, Cubeception 3 uses its two pressure sensors together to find the depth at a static point in the middle of the vehicle, regardless of orientation. Without such compensation for the location of the sensor, the vehicle would tend to change depth during pitching or rolling maneuvers, so this system is crucial to consistent control.

B. Sensor Calibration

Sensor calibration is handled by an interactive program that collects all the necessary data and outputs a set of coefficients that are used to adjust for small differences between runs. Gyro and accelerometer calibration results in a set of offsets representing the bias in the sensor's output. Magnetometer calibration is more complicated, producing a correction matrix to handle stray magnetic fields and ferromagnetic materials in the vicinity of the vehicle.

C. Networking

The increased complexity brought on by Cubeception 3's distributed computation model required a robust networking system to facilitate communication between the different subsystems. To fulfill this requirement, a distributed shared memory system (DSM) was created. A DSM server process is spawned on all participating computers, managing the underlying UDP multicast and Boost shared memory backend. Client programs can then connect to this server and request data from similar servers running on other machines. Clients may also offer information through the server for other machines to access.

The DSM server was designed to gracefully respond to unexpected errors. A server instance can detect and notify its clients if a remote server fails to send an update within a certain timeframe. If the remote server is brought back up, the stream of data will automatically be restored, allowing clients to proceed normally.

Processes running on the same machine can also use the DSM server as a method of interprocess communication. Utilizing this functionality, logging programs can passively attach and observe the state of other processes on the robot. Clients are given the ability to segment their data, so other processes can only to listen to relevant information, allowing for quick additions of logging data without needing to modify other remote processes.

The DSM server and client software are written in C++, but for convenience, a Python wrapper was also implemented. Development for Cubeception 3 can thus leverage the quick turnaround of code written in Python and the speed of code written in C++.

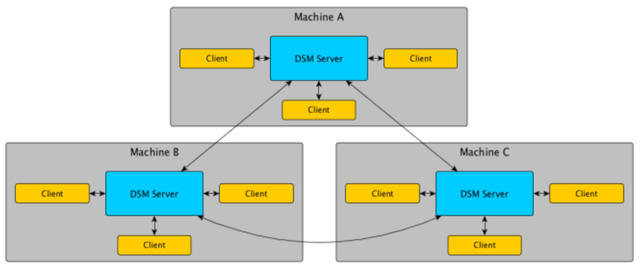

Fig 8. Cubeception 3 network layout.

D. Raspberry Pi Imaging

Cubeception 3's distributed computing created a new challenge for maintainability. Ideally, replacing an individual board in the system does not disturb the other boards. Additionally, replacements should be fast to set up and not require specialized instructions. To meet these needs, a solution was devised using both hardware and software methods. Each board is initially loaded with the same disk image on a microSD card. Specialized GPIO inputs are given to the Raspberry Pi at startup. A setup daemon then runs on the Raspberry Pi recognizing these inputs and initializing the appropriate services and setting a static IP address. Setting the IP address of the board based on its role allows minimal changes on the other boards to incorporate a replacement. SSH is used if necessary to connect to a machine for specific adjustments.

E. Logging

A focus on logging was placed on this year's software design. In the past, only processed data was logged, limiting the potential uses of the data. Cubeception 3's logging backend this year heavily leverages the distributed shared memory system to log all data, raw and processed, to local files and over the network. Raw data can be fed back through the processing pipeline to test new algorithms and find potential issues. This processed data can be plotted in real time for visualization and error identification.

F. Navigation

Using the continuously updated data from the inertial and pressure sensors onboard, Cubeception 3 maintains precise control over all six degrees of freedom. The robot compensates for buoyancy, allowing for nearly identical control regardless of position or orientation. PID controllers are then utilized to meet desired values for linear velocity and angular velocity. Commands to set orientation, depth, and relative position based on predictive motion profiling are also available.

A waypoint system based on dead reckoning was developed to ease navigation to known locations. A signal can be sent to move to an absolute position, where the origin is the starting point of a run. To move to a waypoint, the robot creates a motion profile defining a smooth curve for its velocity over a period of time. In this way, the motion is smooth, minimizing jerk. Additionally, different time periods can be specified, allowing for quick motions, which is good for tasks such as bumping a buoy, making gradual motions, and scanning the floor while keeping field elements undisturbed.

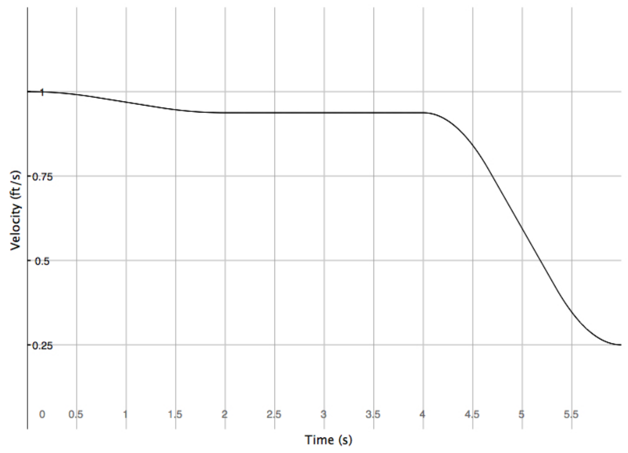

Fig 9. Sample velocity over time curves for a motion profile. This demonstrates smooth acceleration and deceleration.

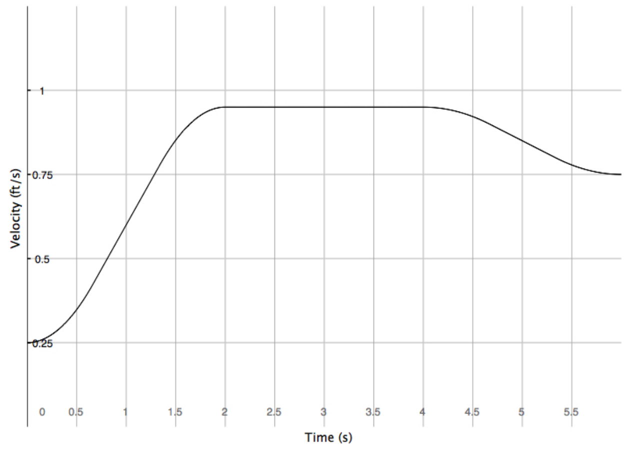

Fig 10. Another velocity over time curve for a motion profile.

To integrate the new physical motor layout on Cubeception 3, a new set of kinematic algorithms defining the robot's motion were derived. The focus of this work was to enhance the predictability of the system without environmental sensor intervention. Forces and torques that act on the robot can be estimated using knowledge of Cubeception 3's geometry and motor outputs. This precise physical model allows Cubeception 3 to balance its thruster outputs to accurately predict motions for dead reckoning.

G. Computer Vision

Cubeception 3's vision software utilizes two 5 megapixel Raspberry Pi cameras connected to separate Raspberry Pi 2s running OpenCV. Cubeception 3's modular design allows for a distinct separation of tasks. Since processing the images in real time can be resource intensive, by having the computer vision systems separate, the rest of the robot can operate smoothly and work asynchronously with the vision system. Cubeception 3 can also continue to function without the vision system in case of an emergency.

Due to the water's translucent nature, Cubeception 3 only uses computer vision as a short range method to precisely maneuver and attempt tasks. It is not used for overall navigation.

A forward-facing camera allows for detection of objects directly in front of Cubeception 3. The computer vision algorithm on this camera begins by averaging all the pixels to determine the hue of the background water in the image. This dynamic color-determining algorithm allows for objects in the frame to be isolated and categorized. They are put through a shape recognition algorithm to determine whether they are buoys, gates, or not of interest. From there, a contour-detection algorithm and trigonometric calculations yield how far and in which direction the objects are so Cubeception 3 can precisely maneuver in an appropriate manner.

The second, downwards-facing camera has a separate navigational algorithm. Cubeception 3 isolates the distinct orange color of the tape lining the bottom of the pool. It also uses its depth sensors to estimate the expected difference between the color of the tape compared to the floor of the pool. Cubeception 3 does this because the lighting varies at different depths and a static tape recognition routine may not properly detect the tape in certain situations.

H. Sonar

Utilizing the sonar board and FPGA shield board, Cubeception 3 is capable of performing both active and passive sonar by calculating times from the emitter to an object to each of the 3 hydrophones or simply receiving signals from a pinger and calculating time differences.

From there, the received times are used to draw ellipses for possible object locations. Target headings and distances can be determined from the intersections of these ellipses, and this data can be used to supplement navigation between mission elements or find pingers.

VI. EXPERIMENTAL RESULTS

A. Sonar Simulations

After experimenting with various two and three linear hydrophone array setups in a Matlab simulation, the sonar subteam found that arranging an emitter and three consecutive hydrophones in a linear fashion allowed for reasonably accurate resolution of target headings and distances. The obtained data suggested that sonar is best used as both a supplement to navigation and a way to move close enough to mission elements that Cubeception 3's onboard cameras could detect colors and features more accurately.

Initially, the sonar subteam created simulations where the simulation would calculate a grid of times for objects placed in the field of view, comparing each of those times to program input to see if an object could possibly be in that location.

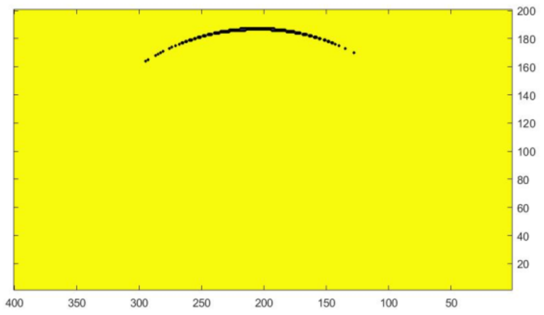

Fig 11. Two hydrophone array comparing received times to a time table generated based on expected times. The black curve represents possible locations.

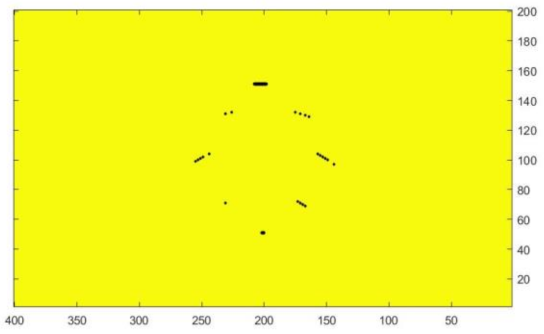

Fig 12. Three hydrophone array detecting eight objects placed in an approximately circular fashion using time tables.

However, the sonar subteam quickly found that using ellipse triangulation with a linear three hydrophone array was more effective, as it was possible to pinpoint the heading and distance of the object. Used in conjunction with previous data and navigation data, this would allow for detection of objects in 3D space.

B. *Computer Vision Testing*

The computer vision subteam's first attempt at an algorithm was to detect circular edges. This failed since it was difficult to pick up edges in the water based on only light values. The algorithm was refined to use hue to distinguish objects, since the background water was a relatively uniform color compared to buoys and other objects. Many test images and videos from last year were tested to make the software perform well under various lighting conditions.

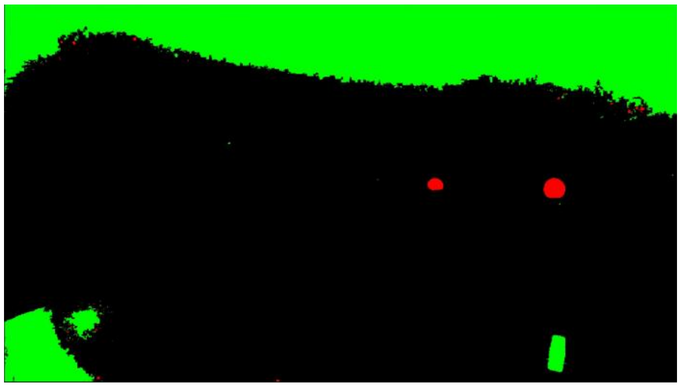

Fig 13. Computer vision on buoys.

C. *Multisensor Calculations*

Many of the same type of sensor on a single platform has several distinct benefits. From a logistical standpoint, having several sensors increases the redundancy of the system, as a single sensor failing does not bring down the entire processing pipeline. Additionally, repairs are not immediately necessary, improving the overall uptime. Considering the quality of data produced, several sensors used together can produce a better signal with some processing. Simple empirical testing saw improvements of up to 30% in angular random walk (ARW) for a gyro. With this in mind, a cost-effective solution to enhancing Cubeception 3's sensor performance was found.

Sensor	Gyro 1	Gyro 2	Average Data
Drift (degrees)	0.241	0.273	0.197

Fig 14. Angular drift of stationary InvenSense MPU-9150 gyros over 15 minute period.

D. *Component Testing*

Before Cubeception 3 is assembled, it goes through a leak test, all of the electronics are tested outside of the vehicle to ensure compatibility, and the software is unit tested to reduce runtime errors. Integrated testing for any potential problems will continue until the RoboSub competition begins.

VII. Acknowledgements

San Diego Robotics 101 would like to thank all of its sponsors for making Cubeception 3 a reality. Solidworks provided invaluable software which made prototyping, simulating, and animating models a very streamlined process. Matlab also provided useful computational software which was utilized heavily by the sonar development team for simulations. Vinatech Engineering, Inc. and Apollo Manufacturing Services were also integral in the creation of Cubeception 3's electronics and core.