Release the Kraken 2.0

San Diego City Robotics – Autonomous Underwater Vehicle 2014

Design synopsis

Robert Pruitt (Faculty Advisor/Electrical Design Lead), **Patricia Shanahan** (Programming Advisor), **Jason Plojo** (Team Leader), **James Haak** (Second Programming Advisor), **Oscar Aguilar** (Mechanical Team), **Nick Cantrell** (Quality Management), **Juan Delgado** (Electrical Design Team) **Ed Eberle** (Webmaster), **Alex Gilbert** (OpenCV Integration Lead), **Richard Harris** (Technical Writer) **Sammy Insixiengmai** (IMU integration), **Luis Jarquin** (Electrical Integration Lead), **Michael Knapp** (Technical Writer), **James May** (Test Engineer), **Eric Mudd** (Lead Technical Writer), **Nils Olsson** (Linux Applications Engineer), **Adan Perez** (Video Production), **Eric Vize** (Conceptual Artist)

The Kraken 2.0 is an AUV (Autonomous Underwater Vehicle) developed by San Diego City Robotics (SDCR) with extreme cost constraints in mind. Thanks to a generous donation from the Inforce Computing® corporation, a new mainboard with a small physical footprint was aptly used as an upgraded computer for the team. The raw processing power of this new device allowed our software to compute the vision processing that plagued teams of past for SDCR. Due to the decreased size of the main computer, it allowed our build team to have expanded options on hull design. Two completely new hull designs were generated with one final build as the outcome. The Kraken's SONAR subsystem also had improved programming and circuit changes. The main strength of the build is its use of commercial off-the-shelf parts as result of a constrained budget.

0 INTRODUCTION

SDCR is a group composed of students, professionals, and faculty. This year's team received an opportunity to radically change the design of SDCR's AUV when the previous primary computer was damaged. This in turn created new obstacles and dilemmas for a fresh project. Due to the high attrition rate of a junior college, the SDCR team membership varies from year to year. This allows for change and innovation to happen with very little difficulty. The current SDCR team was divided into three groups; a build design team, a software development team, and an electronics hardware team. Each team met their own challenge this year and it was all centered on a design change for the main computer. Because of the composition of SDCR, the team was able to identify weaknesses of past builds and decided that a complete redesign was worthwhile. With this decision, the Kraken 2.0 was born!

1 Build Design Team

A Gantt chart was created at the beginning of the Fall Term so that build progress could be tracked and coordinated with the software and electrical teams. Brief weekly "standup meetings" allowed for team business to be discussed at the start of each class.

1.1 Build Process

It was understood by the build team that the 2013 Kraken had some unresolved mechanical problems. The Build Design Team was given the opportunity to make a proposal to SDCR for a new pressure housing in order to request funding from team resources. This subject rapidly became contentious. To resolve these conflicts, a presentation called “Dueling Banjos” was made to the team suggesting a design study should be conducted in Solidworks®.

Once the design study was finished, the participants in the build team could vote with their feet in a process of parallel development. This allowed all parties an opportunity to use their personal resources to deliver a waterproof container to the team for consideration. The concept which worked best for the software and electrical teams would be selected by them for final vehicle integration.

1.1.1 Requirements Management

In order to manage the risk of a mechanical redesign, a requirements management process was established to keep things organized and focused. A requirements tracking matrix was created in Google Sheets and a performance score point-target was identified by doing a statistical analysis of the previous three years of Robosub competition scores. Finally, mission objectives were evaluated based on point value and a qualitative assessment of difficulty according to available team resources. This process guided the Build Design Team’s efforts to prepare proposals for evaluation by the rest of the team.

1.1.2 Design Study

A design study was conducted using Solidworks® CAD software where all participants in the build team could have their ideas visualized for minimal expense or labor investment in manufacturing. This process also allowed the potential paths forward to be evaluated by the team, using quantitative metrics, to make the determination. All of the options considered were developed using a bottom-up design methodology where CAD representations of all components believed to be necessary for the software and electrical team integration were downloaded from suppliers or reverse engineered with calipers. Because the system diagram was virtually identical regardless of the pressure housing selected, the process of reverse engineering or tracking down CAD models was productive to the build team’s activities regardless of what pressure housing was selected.

1.1.3 Quality Management

To get the most benefit from the CAD activities, accurate weights were recorded in SI units when they were provided by the manufacturer or supplier; or a digital scale was used to take measurements of the remaining components. By tracking the displacement of the pressure housings under consideration, as well as the weight of the control system electronics, the expected battery mass/energy storage could be predicted to achieve neutral buoyancy while understanding the weight bonus point impact of the decisions. The most attractive benefit of this process however was that by assigning price and supplier properties to the .SLDPRT files, the process of generating and maintaining the Bill of Materials (BoM) was largely automated. This translated in to very accurate accounting for total vehicle cost before investing a single dollar.

1.2 Component Development

In order to minimize risk, very little component development was done by the build team. Two undertakings were selected to maximize performance improvements and cost reductions. These were a new pressure housing and developing a BLDC motor and Electric Ducted Fan (EDF) based 3D printed thruster. The 3D printed thrusters were designed to use an identical mounting hole pattern as other commercially available thrusters used by the team.

1.2.1 Pressure Housing

Frustration with Pressure Housing issues in the past motivated the build team to a consensus that a replacement was essential to avoid disappointing outcomes. Taking on a mechanical design and manufacturing project in Q1 of 2014 was understood to be a threat to the pool testing schedule, however the potential benefits were believed to justify the risk of taking on this project. While this build process was underway, the old pressure housing was left intact; allowing for a graceful course correction if the build team schedule slipped too egregiously.

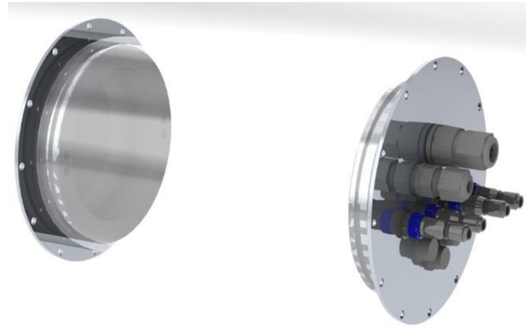

Figure 1 – Pressure Housing Model

1.2.2 Thrusters

This year's build team concluded that the old "bilge pump" motors that were used as dive and lateral thrusters needed to be replaced. Although the design was unique and useful, they became a burden as they aged. They soon took more time and resources to repair them than to design a new version. The decision was made to use the same BLDC and EDF combination selected by the 2012 San Diego Mesa College Robosub team. These components were more than adequate to meet our requirements and had already been demonstrated as an effective solution by another team. To improve on the work done by Mesa College, 3D printed prop guards were designed in Solidworks® that were compatible with existing thruster mounts that were made for commercially produced thrusters.

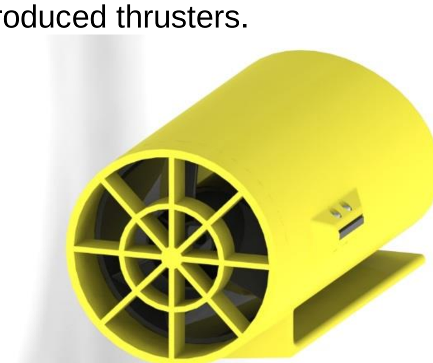

Figure 2 – Thruster Solid Model

1.3 Manufacturing

The manufacturing process faced several challenges which guided the decision making process for the team. Primarily these challenges were driven by severe resource constraints which consisted of finances, scheduling conflicts, and availability of skilled labor. Operating within these constraints guided the ambition and scope of what components were available to the design team for customization. Targeted decisions were made on which undertakings were expected to provide maximum benefit to the vehicle performance; which were then adjusted for perceived risk and resource expense. One strategy used by SDCR to expedite vehicle testing was to adopt Fused Deposition Modeling (FDM) based additive manufacturing as our preferred fabrication technology. This provided a high level of flexibility to changing requirements while the software team worked to simplify complexity and reduce component count. When possible, commercially available solutions were given a significant preference to avoid manufacturing lead-time impacting schedule.

1.3.1 Aluminum End-Caps

The Build Design Team's decision to use an acrylic pressure housing created a challenge for the manufacturing team which could not be avoided. The cast acrylic pipe purchased from McMaster-Carr was sold with a nominal internal diameter which did not facilitate the use of commercially available end-cap solutions. A Finite Element Analysis (FEA) study conducted by the build team supported the belief that the end-caps could be 3D printed in ABS plastic and still sustain the necessary pressure to meet the operating depth requirements of the Robosub competition. This approach (if successful) would have allowed greater flexibility, reduced cost, and enabled more rapid delivery of a waterproof vessel to contain the Kraken's electronics.

A significant set-back in the team's build-schedule was encountered when both of the 3D Printers expected to be available for manufacturing the end-caps simultaneously became unavailable. After several weeks of waiting for these resources to return to service, the decision was made to use more traditional manufacturing techniques. Once the labor expense of lathe operation had been committed, the decision to use a higher strength material (6061 Aluminum) appeared to offer a greater maximum operating depth for similar labor and material cost.

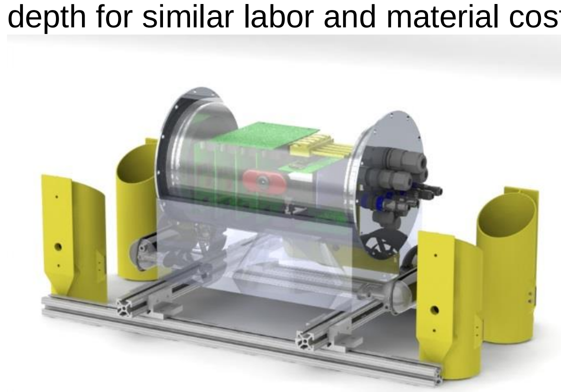

Figure 3 – Solid Model of Chassis

2 Electronics Hardware Team

The Electronics Hardware Team was divided into three subgroups: (1) an electrical design team that handled specific circuitry and power for the different accessories for the sub; (2) a computing team that handled the motherboard and its accessories; and (3) the sensors team.

2.1 Electrical Design

During the process of creating the “requirements tracking matrix”, electronics components were assigned electrical properties such as voltage, current, and power requirement in watts. This was a valuable asset during the process of designing the electrical and mechanical systems as it allowed for the batteries to be appropriately matched in capacity, voltage, and max discharge power. Components powered from a $12V_{DC}$ or $5V_{DC}$ input were preferred because of the local availability of supplies such as fuses, relays, and DCDC converters in these voltages.

2.2 Computers

Last year’s Kraken design used a micro-ATX motherboard with an Intel processor. When the vehicle took on water, the motherboard and CPU were both lost. Although this equipment damage was unfortunate, the tragedy became an opportunity to select a new mainboard. This freedom allowed for the vehicle’s displacement to be greatly reduced by selecting a smaller footprint component. Ultimately the board selected was an IFC6410 Pico-ITX Single-Board Computer (SBC) from the Inforce Computing® Corporation.

2.2.1 Main Computer – Inforce Computing ® - IFC6410

This Snapdragon™ based board was generously donated to SDCR. It features an ARM® architecture Qualcomm® Snapdragon™ S4 Pro APQ8064 quad core processor clocked at 1.7GHz, as well as an Adreno 320 GPU. The smaller Pico-ITX form factor allowed for a reduction in pressure housing deplacement. It contains 2 GB of onboard DDR3 RAM and a 4 GB eMMC (embedded MultiMedia Card) drive for storage. It also includes a MicroSD card connector that team uses for the storing of the Kraken’s software logs.

Figure 4 – IFC6410 DragonBoard

2.2.2 Contingency Computer - Raspberry Pi

During the portion of the schedule where the IFC6410 SBC was being configured for Linux, a “Raspberry Pi” (Broadcom® BCM2835 SBC) was explored as a contingency plan. Despite this board only having a single core 700 MHz processor, there was a belief that OpenCL based GPU programming could overcome these processing limitations for vision code. Although these efforts to use the OpenCV OpenCL library were successfully compiled in Microsoft Visual Studio®(for an x64 architecture PC), attempts to compile the same code for the Raspberry Pi’s (ARMv6 architecture) VideoCore® IV mobile graphics core were unsuccessful.

2.2.3 Arduino MEGA 2560

The standard Arduino Mega used in 2011 was upgraded to an Arduino Mega 2560 multi-controller. The 2560 doubles

the Kraken's program space as well as the memory available to NavBox. By increasing the processing capacity, the end result allowed for a more robust program to run the Kraken.
The Arduino Mega reads the current heading and depth information provided by the IMU and pressure sensor. It also receives data from the Kraken’s sonar subsystem via Twin Wire Interface (I_2C). From that data, the Arduino (via NavBox) then determines the appropriate speed, heading, and depth of the Kraken; then sends signals to the motor controllers.

2.3 Sensors

The Kraken achieves a “high level of autonomy” through accurate sensor data. All the Kraken’s sensors were selected with both quality and ease of integration in mind. As a result we chose an Invensense® MPU-6050™ based Attitude and Heading Reference System (AHRS) from 3D Robotics LLC as a low-drift heading reference and for tracking vehicle attitude. For ease of integration, a differential analog pressure transducer from OMEGA Engineering Inc. was selected to measure vehicle depth.

2.3.1 IMU (Inertial Measurement Unit)

The process of maintaining an accurate position in a given space is a prerequisite for much of the sub’s activities during the course of the competition. This year the team decided to use the ArduIMU v3 given to us from 3D Robotics® that is based on the MPU-6050™. It utilizes MotionFusion™, a software tool that uses a three-axis gyroscope, a three-axis accelerometer and a digital compass. It provides the instantaneous and reference heading to our Navigation Box control code.

2.3.2 Camera

Before problems were discovered with the Raspberry Pi GPU, the USB Cameras were selected based on work being done with that SBC. The Raspberry Pi suffers from a known issue with its USB 2.0 ports which narrowed the available choices for USB webcams beyond the usual restrictions surrounding compatibility with Video4Linux2 drivers (V4L2). One of the few USB webcams known to be compatible with both the Raspberry Pi’s USB 2.0 power supply, as well as the V4L2 drivers was the Logitech C210. This Webcam has a published Video Resolution of 640 x 480, a reported current draw of 170mA and a secure supply chain via Amazon Prime™. All of these attributes made it a preferred choice for ease of development.

2.3.3 Sonar

In conjunction with Professor Pruitt, SDCR was able develop a viable sonar option. The Hydrophone Array consists of 4 custom fabricated piezoelectric elements that are potted and mounted into an anodized aluminum adjustable scan angle frame.

The design challenges we faced included signal level variation with location, unwanted frequencies, and the mathematical complexity of converting arrival times to pinger heading angles. One problem was the variation in received signal level depending on the Kraken’s location in the pool. This was solved by using a high gain operational amplifier circuit which outputs a constant voltage square wave.

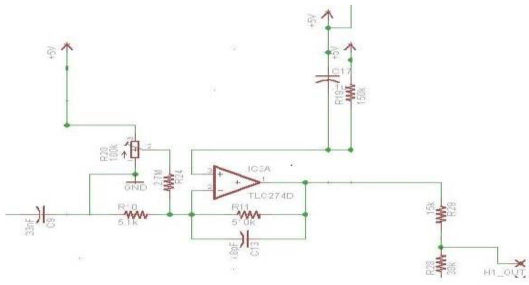

Figure 5 – Gain Amp

There was a need to discriminate against frequencies outside of the desired frequency range. The fixed-level square wave signals are sent to a programmable 4th order (MAX7490) switched capacitor filter, leaving only a sine wave of constant level. The center frequency of each MAX7490 chip is controlled by a Direct Digital Synthesizer (DDS) generated clock, which allows on-the-fly frequency selection in the 22KHz to 30KHz range. Because distortion was observed at the filter output, a voltage divider was then added between the amplifier and the filter, resulting in a cleaner output from the filter circuit. The final analog processing step was to delay one side of each signal pair using an all-pass phase shifter circuit. The sonar microcontroller, an ATMega16, counts the number of crystal clock pulses between the two arriving data streams and converts the difference into distance, and then pinger angle.

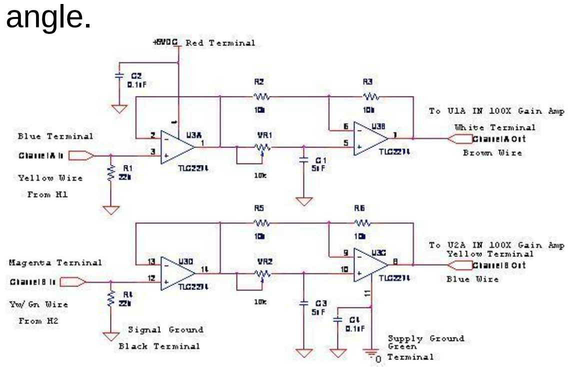

Figure 6 - Phase shifter

A counting timer in the code is set to only store a measurement if there has been silence for more than 1 second to keep the sub from chasing echoes coming from the wrong direction. This microcontroller is on a custom PCB that was designed by Robert Pruitt, Keith Dwyer, and Frank Alsaro.

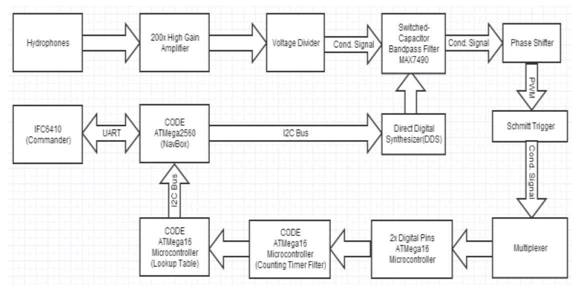

Figure 7 – Sonar Flow Diagram

Several problems arose in the mathematical calculations necessary to convert the timer counts to information that was useful to the commander module. The original microcontroller clock was chosen to be 4 MHz for stability, but the time it took the code to measure the clock counts was too close to the actual time difference being measured for small degree values to the far left. By quadrupling the clock frequency to 16 MHz, usable horizontal and vertical resolution was obtained.

In the original plan, a fixed lookup table at the median pinger frequency of 26 kHz was used to speed up calculations. This year the electronics team reviewed how the fixed frequency lookup table performed. They found out that it produced a margin of error of 6°. The group decided that was too much and changed the way the lookup table was produced. They allowed the microprocessor to do the mathematical calculations for the specific pinger frequency and placed generic arcsine values in the lookup table. A spare bus on the microcontroller accepts the daily pinger value from the commander module. This then produced a margin of error of only 0.5°. The sonar is much more accurate now.

The arcsine is calculated and converted to signed pinger angles via the lookup table and then the data is sent physically via Twin Wire Interface (I_2C) to the Arduino. NavBox then instructs the motors on how to steer.

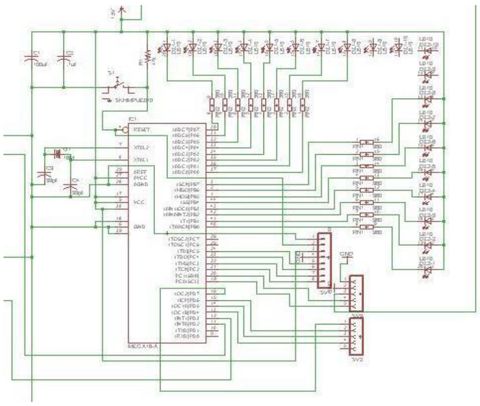

Figure 7 - Microprocessor/LED Circuit

3 Software Development Team

The goal of the computing system was to be small, require little power and eventually allow the Kraken the capability to attempt every competition task. Based on the aforementioned requirements, the new system consisted of an IFC6410 computer, an ATMega2560, and two cameras. In order to maximize system performance, all software is built upon a Linux operating system and written in a mixture of C (which is used for the microcontroller), C++ (used for the main application), and Assembly (used for the Custom Sonar Board). This year's software is a new, more polished iteration based upon the success of previous systems. The new structure features multi-threaded vision processing, control systems, and a sophisticated testing and debugging scheme.

The new computing platform posed a unique and interesting challenge to coders this year. Since the computer uses Qualcomm's newest Snapdragon™ processor, Google's Android OS was installed by Inforce as the default control for the board.

Using the Android Software Development Kit (SDK) would have required refactoring code or use of the Android Native Development Kit (NDK). Although some team members continued to work on this front, committing to this path was perceived to be too risky and beyond the scope of what the software team could accomplish in the time allotted.

A consensus was made to install an alternate distribution of Linux. Due to the very young nature of the board, support and availability of distributions was few and far between. There were many methods attempted by the team.

The first attempt was specific to a distribution of Arch Linux. Further attempts were made with Ubuntu, and a distribution known as "PragmaTux". Finally, an attempt was made with Red Hat's Fedora. All of these attempts were unsuccessful. The trial and error process took about two months, but a final solution was found through the manufacturer.

As time progressed the Inforce website was able to increase support and availability of alternate operating systems for control of its hardware. A modified version of the Ubuntu made specifically for the IFC6410 board was found. This was downloaded and installed to the board without any further problems.

The application coded by the software development team consists of 4 major parts: (1) the Dashboard, a graphical user interface (GUI); (2) the Commander, the overseer of the Krakens subsystems;(3) the Navigation Box (NavBox), movement control code for the Kraken that gets sent to the Arduino; and (4) the TRANSDEC Simulator (SimBox).

3.1 Dashboard

Dashboard is a GUI that has been developed solely for testing, debugging, and maintenance. It is the proverbial “manual override” that allows the team to troubleshoot found problems and is completely unnecessary for the Kraken to run in competition mode, because it is run on an external computer. It enables real-time data viewing and remote control of the sub while submerged, via the “router buoy” tethered to the AUV. A server running on the Commander transmits data such as, heading, depth, and thruster duty-cycle, to a land-based client machine running Dashboard. Upon receiving the information, Dashboard displays and logs the data, with the ability to replay the session at a later time.

3.2 Commander

Commander consists of 3 major subsystems: (1) Mission control, which determines what task/mission is current; (2) Communication (CommLink), which interfaces with the ATMega2560 board and (3) Vision, which is used to process the vision input from the cameras.

3.2.1 Mission Control

Mission Control is the name given to the subcomponent of Commander that runs on the IFC6410 and is the mission/task logic software. Decisions on which task to attempt is based on time out slots. It makes use of the information generated by the vision code and sonar to make decisions about specific target headings. Once the Mission Control has designated what mission/task to run, and how to run it, the Mission Control relays the mission specifications to the Navigation Box. From that, NavBox will turn on certain motors to move the Kraken to appropriate positions.

Previous versions of Mission Control relied heavily on several “if”, “then”, “else” statements. A more efficient version was written through the use of the idea of “State Machines”.

3.2.2 Communication (CommLink)

The Communication interface or “CommLink” is a second subsystem of the Commander software. This leg of Commander uses the MaiaXmlRpcServer library and allows any accessory attached to Arduino to communicate with the rest of the system. Data can both be received and sent to the Arduino via CommLink and a USB port on the IFC6410. The Kraken’s sonar subsystem uses CommLink for communication to and from the hydrophones.

3.2.3 Vision

The “Vision” processing system is based on OpenCV; making extensive use of existing libraries, but also includes custom code written by the software team. The “Vision” module algorithm begins by preprocessing image frames using histogram equalization in YCrCb space, a Gaussian blur filter and basic HSV thresholding. The resulting binary image is fed to a canny edge detector producing the necessary contours allowing for contour approximation, convexity checks, structural analysis and shape detectors. Hue, Saturation, Value (HSV) space was used in lieu of BGR values because BGR is too discrete for use in the Kraken’s mission logic.

3.3 Navigation Box (NavBox)

Navigation Box is the motor control code that runs directly on the ATMega2560. When given a target depth and heading by Commander, it reads in the data from the IMU and pressure sensor. NavBox then makes the necessary adjustment to the Kraken’s six motors via a Pulse Width Modulation (PWM) signal to reach the desired depth and heading. Commander also relays speed constraints to NavBox for backwards and forward movement.

3.4 TRANSDEC Simulator (SimBox)

One resource which recently became available to SDCR was an Oculus Rift SDK. Using VisualSFM, the 2011 Robosub rules, and aerial photographs; a Solidworks® model of the TRANSDEC was generated. MeshLab was used to convert the Solidworks® generated .STL file to Unity 3D/Oculus Rift SDK compatible .OBJ file formats. Unity 3D allows for the generation of a navigable cross-platform simulator where lighting conditions, physics, and textures can be controlled to accurately represent the expected operating environment. The Oculus Rift SDK allows an immersive experience where the developers can view the environment from the perspective of the vehicle to enhance their understanding of the problems faced during autonomous decision making.

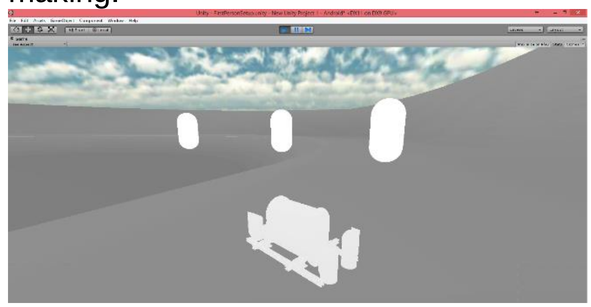

Figure 8 – SimBox Screen Capture

4 Conclusion

This year SDCR set out to design and build a robust research platform on which the software team could develop and refine algorithms in both pool testing and simulation. Setting realistic objectives on what could be accomplished as well as making effective use of our limited resources were priorities. This paper covered the processes used to engineer the "Kraken 2.0" as well as overcome the obstacles we faced along the way.

5 Acknowledgements

All of this was made possible thanks to the generous support received by SDCR from our sponsors San Diego City College, and the San Diego City College Associated Students Government.