charybdis

DUKE UNIVERSITY

Gareth Guvanasen, Brian Hilgeford, Andrew Waterman,
John Felkins, Josh Johnston, Tyler Helble, Ethan Eade
Faculty Advisor: Dr. Jason Janet, PhD

In keeping with the Duke Robotics' goal of ever increasing innovation, the team has worked to improve all facets of Charybdis, our autonomous underwater vehicle (AUV). Using the lessons learned over the past few years, Charybdis builds upon its strengths while addressing the many weaknesses in the previous design. This year Charybdis retains its hallmarks of agility, precision navigation, and light weight while gaining greater environmental sensory capability and translational movement control.

With the help of strong sponsorship support from Duke University and private donors, the Duke Robotics Team upgraded a large part of the Charybdis feature set by adding stabilizing fins, a refined acoustics system, improved power conversion chips, and enhanced vision software.

As before, Charybdis is constructed with commercial-grade components and craftsmanship in order to provide a reliable and stable platform. Lithium polymer batteries provide power to the thrusters and the electronics tube, which houses a new Versalogic computer, Diamond Systems DAC, wireless Ethernet card, power management board, relays, and DC/DC converters. Subconn connectors that pass through the tube bulkheads are used to prevent leaks. A powerful software suite includes device drivers that control a full set of sensors, computer vision algorithms, high-level mission control and decision-making software, as well as a scripting language supporting intuitive entry of mission parameters.

Introduction

After the 8th Annual Office of Naval Research/AUVSI International Underwater Robotics Competition in 2005, Duke Robotics decided to address several weaknesses of the vehicle that surfaced during and after the competition. Charybdis was revised in an effort to increase the versatility and stability of our already innovative platform. Charybdis itself was created as a successor to our unique Gamera AUV that competed in the 2001-2003 competitions.

Charybdis was designed around a unique and groundbreaking propulsion system. Three thrusters mounted around a circular chassis enable the vehicle, with appropriate control algorithms, to move in any direction without rotating, yielding full holonomic control. A fourth thruster positioned vertically controls up and down motion. This system allows for unprecedented precision and agility in an Autonomous Underwater Vehicle. Other physical components were designed around this basic plan, with the airtight electronics and battery tubes arranged with the sensors around the central thruster housing. Two translucent acrylic shells encapsulate these components for protection and to remove concepts of “front” and “back” from the drag profile.

The electronics suite was rebuilt with new components to simplify the electronics stack and increase reliability. This task was completed with an eye toward upgradeability and with the intention of not having to unseal the electronics tube unless a new component is being installed. The power system was also upgraded from lead acid to lithium polymer batteries, providing 8 times the power density.

Finally, the software package was completely redesigned with a new, more intuitive waypoint scripting language. The rest of the mission and control logic was rewritten to place the entire body of code within a modular, object-oriented framework.

Propulsion

Three Tecnadyne 250 propeller-based thrusters are mounted around the perimeter of the round hull, each 120 degrees apart. Since each thruster has bi-directional variable power, it is possible to move in any horizontal-plane direction by using the correct combination of thrust. A fourth thruster is mounted vertically in a low-loss channel that runs through the middle of the vehicle and allows vertical control. This gives Charybdis 4 DOF holonomic control, while pitch and roll are taken care of with a positive righting moment created by appropriate weight and flotation distribution.

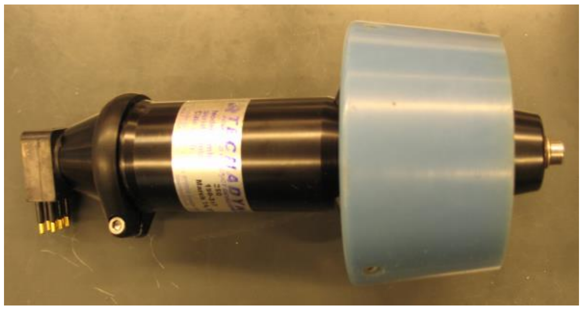
Figure 1: Thruster

This central thruster, which draws water through the machine, is the inspiration for the vehicle's name. In Greek mythology Charybdis was the daughter of Poseidon and Gaia until Zeus turned her into a monster. She lived in a cave in the Straight of Messina and sucked water in and out three times a day, forming a whirlpool that destroyed passing ships. Charybdis, opposite the straight from the six-headed monster Scylla, was one of the obstacles in Homer's Odyssey.

A mathematical model was developed that determines the thrust contribution from each thruster to move in a particular direction. This model ensures that the overall net thrust vector passes through Charybdis' center of mass (which is roughly the center of drag), thereby preventing a rotational moment. The Tecnadyne thrusters don't have a linear force response to voltage, nor are they symmetrically powerful, having 12 lbs thrust in the forward direction and only 6 in reverse. Thus, a lookup table was added that converts the desired force contribution from each thruster into the required voltage, taking into account these nonlinearities. The computer applies this voltage through a DAC card and amplifiers, which are on a separate circuit from the electronics to protect the sensitive devices from high current.

The central thruster is simpler. It gives control in the vertical direction, or Z-axis. Since this thruster is located at the center of mass, there is never a concern about a rotational moment. A low-loss channel runs through the vehicle with the thruster mounted in the middle. This channel also serves a mechanical purpose by taking weight off of the shells and sensors during storage and display of the vehicle. The Z-axis thruster is oriented with the more powerful side pointed upward to counteract the natural slight buoyancy of Charybdis.

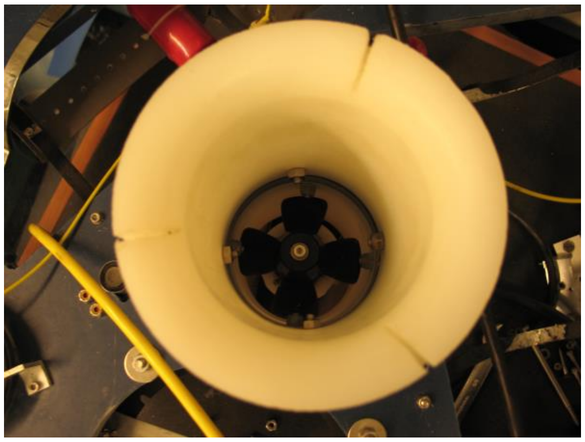
Figure 2: Z-Intake

In the 2004 competition Charybdis's center thruster control chip was damaged by water in a pierced cable. To prevent future leakage, all thruster end caps were upgraded with bulkhead connectors to guarantee full water blockage. Micro connectors were used on the thrusters and the electronics tube end cap to reduce the space consumed by connectors. A backup thruster was also acquired as a spare, as our platform depends on all four thrusters being operational for total control.

Connectors and Wet/Dry Interfaces

Consistent with the Charybdis design team's goals of industry standard construction and components, all electrical connections and wet/dry interfaces use Subconn Low Profile Series underwater connectors. These 4 and 7 pin cables can be wet mated and will not short or leak. Components were either purchased with Subconn cabling, as in the case of the thrusters, or were sent to Open Ocean Engineering to have Subconns potted onto the open-ended whips, like the DVL. The aluminum end caps of the electrical, battery, and camera housings all have Subconn bulkhead connectors to ensure that water doesn't enter these dry spaces. The 2005 vehicle utilizes many Micro Low Profile Series connectors to reduce the space used by bulkhead connectors for easier and faster cable connection/disconnection.

Chassis and Shells

Ultra-High Molecular Weight (UHMW) polyethylene was selected to serve as the chassis. This decision was made based on the Duke Robotics Team's success using this material on Gamera. Work began by cutting a circle 28" in diameter with Abrasive Water Cutting Techniques, and then holes were added for the electronics and battery tubes, DVL, pressure sensor, camera housing, altimeter, and central thruster. Some later adjustments were made with the mill in the Duke Undergraduate Machine Shop.

The translucent shells were fabricated by free-form techniques out of acrylic. They have holes to allow the sensors and central thruster to protrude unimpeded. The purpose of these shells is to protect the internal devices from damage, as well as to move the center of drag to the middle of the vehicle, to coincide with the center of mass. Without the shells, it would be difficult to move in a straight line because the direction-dependent drag profiles, from protrusions such as sensors and the electronics tube, would cause Charybdis to spin and complicate control algorithms.

Static Stabilizer Fins

The static stabilizer fins, one set located on each side of the rear, eliminate the high-velocity pitch control problems of previous years. They now allow Charybdis to operate with at least 150% of its previous velocity threshold, making the robot much more competitive in the event that time becomes a deciding factor. They have an angle of attack less than 12 degrees to avoid the development of significant drag forces and are located as far towards the back as possible without adding extensions. This is to provide the maximum correcting moment without compromising the radial symmetry that gives the robot full holonomic capabilities.

A new chassis has been developed to include dynamic stabilizer fins that are actively controlled by the computer. They are to be deployed only when critical velocities are achieved so that they do not interfere with holonomic sensor sweeping. This technology is still being perfected, but will make its debut in a subsequent competition.

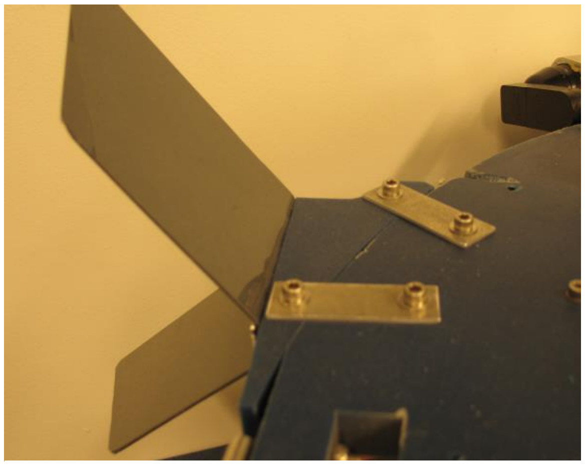

Figure 3: Static Stabilizer Fin

Electronics and Battery Housings

Two dry tubes are used to house the electronics and batteries. Each is constructed of clear acrylic and capped with anodized aluminum bulkheads equipped with double O-ring seals. The battery housing is 12" long with 4.5" external diameter, while the electronics tube has greater volume at 10.5" length and 8" external diameter. Both of these sizes were chosen to maximize usable space and fit between the center thruster mount and outer shells of the robot.

Buoyancy

The two dry compartments, containing the batteries and electronics, have positive net buoyancy. These elements sit opposed to the DVL, which, due to its weight, has negative net buoyancy. To balance Charybdis, flotation inserts for the DVL area were created from shape-able foam and weights were added near the two tubes. Charybdis is slightly positively buoyant as a safety measure, causing the vehicle to surface in the event of a power loss.

Pressure Sensor

Charybdis uses an MSP-600 pressure sensor manufactured by MSI. The output of this sensor is in the form of a 0-5V level, which is converted by the computer into a depth measurement. This data is used to maneuver and remain submerged. If the robot deviates too far from the desired depth, the central thruster is used to make corrections.

Doppler Velocity Logger

This year Duke Robotics completed its purchase of an RD Instruments Workhorse Navigator 1200 kHz DVL at a generous discount. This sensor is the main tool in

Charybdis' dead-reckoning navigation system. By measuring the Doppler shift from four SONAR beams, the DVL is able to determine vehicle speed and, after integrating over time, distance traveled. It communicates this information, plus several other useful data including compass heading, pitch, and roll, to the main computer using the RS-232 protocol over a Subconn connection.

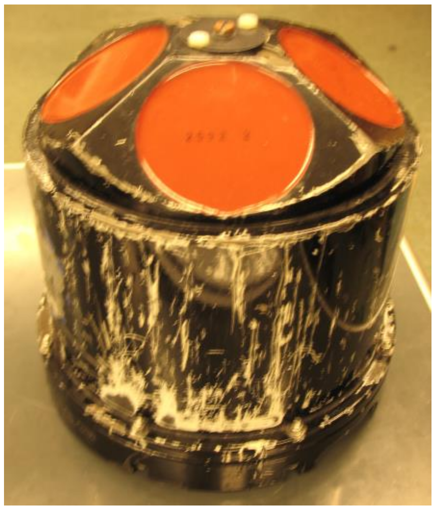

Figure 4: Doppler Velocity Logger

Passive Acoustics

Charybdis' acoustics system uses an array of four hydrophones arranged in a square about the widest part of the chassis. The DAC samples the hydrophones to determine the time each received the ping. An overspecified system of equations that relates the geometry of the hydrophone array to the relative times of receipt is then optimized to find the bearing angle. In concept, the distance from the vehicle to the pinger could also be computed from the pairwise time differences, but the samples are not yet of sufficient quality to make this observation practical. A Fourier transform could be performed on the data to isolate on a ping of known frequency. This improvement has been considered for a future version.

Figure 5: Hydrophone

Voltage Meters and Temperature Sensors

Available for optional use in Charybdis 2005 are temperature and voltage sensors. The voltage sensors are on a plastic mount that attaches above or to the back of the DC/DC board. Two voltage meters indicate the voltage of each 30-volt battery power channel. These can be used to visually monitor battery voltage without using a multimeter. The meters require an input of 9V DC provided via a standard 9-volt battery.

The temperature sensing circuits are two boards with temperature chips mounted to them. The chips require 5V DC power to operate. Each chip produces a 0-5V output to be read by the DAC. The temperature sensors are placed near temperature-sensitive components, ideally the CPU and the 5V DC/DC converter.

Leak Detection System

Available for optional use in Charybdis 2005 is a leak detection system. The system must be installed into the E-Tube by drawing two small wires .5cm apart with conductive ink. The amplification/threshold circuit can be connected to an LED or to the DAC. The system emits a positive 5V signal when water has crossed the lines.

Electronics Stack

The electronics stack is structurally supported by an adjustable skeleton of four threaded rods with high density nylon separators and is housed within the acrylic electronics tube. The lower section contains a

PC104 stack containing a computer, I/O connections, and a DAC, the upper section contains relays and external connectors, and the DC/DC amplifier board rests along the side.

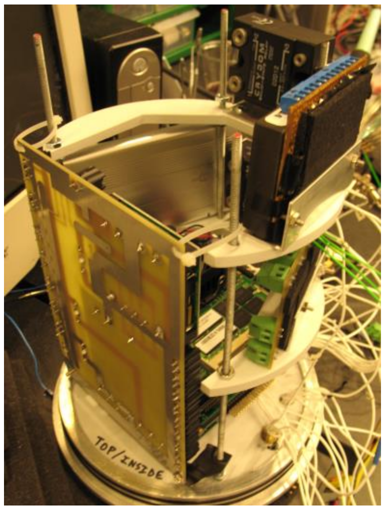

Figure 6: Electronics Stack

PC104 Stack

Computer

This year Duke Robotics is using a new Versalogic EPM-CPU-10 with a Pentium III/Celeron processor as the main computer. Rated at 566 MHz, the processor is throttled to 350 MHz to minimize waste heat. This board also has 256 MB of low power RAM. An IDE controller is connected to 20GB 2.5” laptop hard drive, which is used to store system software and logs.

The Versalogic computer has a breakout ribbon cable that precludes the need for a separate I/O board. This gives us the capability to connect the DVL and Altimeter via RS-232 Serial and the two cameras via USB. There is also an Ethernet port that is used by our floating wireless buoy.

Data Acquisition Card

The DAC is a Diamond Systems MM-32-AT with 4 analog output pins and 32 analog inputs. The inputs are used to monitor the power level of the batteries and get information from some of the sensors, such as the pressure sensor. The analog outputs are used to control the four thrusters. There are also several digital output pins, which are used to control relays like those that trigger the marker droppers.

Power Management Board

Also on the PC104 stack is a power management board that controls power distribution to both the electronics stack and thrusters. Power is input from the batteries and sent to each thruster. Electronics power is sent to the DC/DC converter inputs. Finally, a voltage divider sends 1/11 of the input voltage on to the DAC to allow monitoring of battery strength. This division allows the DAC, with its maximum input of 10V, to measure the level of the 33V-maximum battery output.

Relays

Digital outputs from the DAC are used to trigger Crydom relays to control robot functions. These relays are located in the upper electronics stack and respond to +5V. Two of these relays utilize power from the thruster supply to activate the marker droppers.

DC/DC Converter

The DC/DC Converter is mounted alongside the main electronics structure and generates the voltages used by onboard electronics. It creates a floating ground and the three available power supplies: +5V, +12V, and -12V. Even if the input battery voltage fluctuates between 18V and 36V, the outputs will remain stable. The board contains Astrodyne and Astec converters; the Astec chip supplies +5V and the Astrodyne chip supplies low wattage -12V and +12V channels. The custom board was redesigned in 2005 to feature lower power chips that output less heat. This design change was in response to the power board in 2004 constantly overheating during prolonged use, causing drops in the power supply.

Reed Switch

To satisfy the safety requirement of an emergency kill switch, Duke Robotics wanted to use something new and innovative. We turned to a reed switch, which contains two contacts in a glass tube that is filled with an inert gas. The switch is closed in the presence of a magnetic field and open in its absence. On Charybdis, the switch controls a relay that sits between the batteries and the rest of the vehicle. A red permanent magnet connected to a bright yellow cord provides the magnetic field, and can be removed easily by a diver, thus disconnecting the batteries from all electronics and thrusters. The reed switch is more consistent and much easier to operate than a traditional throw switch because the magnet simply needs to be removed and is easily grasped. Because of the sealed nature of the switch, it is ideal for underwater use.

Batteries

This year Duke Robotics upgraded to Lithium Polymer batteries, which provide eight times the power density of lead-acid ones. Specifically, four 14.8V 8000mAh ThunderPower batteries are employed. Two packs are connected in series to provide 29.6V for the thrusters, and two packs are in series to supply 29.6V to the electronics tube. This configuration provides 6-8 hours of computing time an d 1-2 hours of thruster use. Batteries are recharged in about 4 hours with an Orbit Microlader charger.

Care needs to be taken that the batteries do not drain below 10% of their charge (about 24V). The voltage levels of each system are measured through the DAC and monitored by the computer, which can shut down to prevent damage. This data also allows the computer to update the thruster algorithms to compensate for lower available voltage. LED battery monitors are also available for optional use that can display the real-time

Figure 7: Lithium Polymer Battery

voltages of each battery through the transparent tube.

Operating System

The computer boots from the hard drive into a slimmed-down version of Slackware 9.1 operating the Linux 2.6.6 kernel. The PWC (Philips Webcam) 9.0 kernel module drivers are used to control the two web cameras. The DAC is operated with the Diamond Systems Universal Driver 5.8 interface library. The Linksys Prism 2.5 wireless PCMCIA card is supported by the linuxwlan driver, allowing ad-hoc network communication with the computer.

High Level Software Control

The control system is written entirely in C++. With the exception of the DAC library, libjpeg for image IO, libpthread for POSIX threading, and the C++ runtime library, all the code is written from scratch. GCC version 3.2 or ICC version 8.0 are used to compile a program that runs as a single multi-threaded process.

When the mission preparation begins, a configuration file is loaded that contains all of the modifiable settings for the system, such as calibration values, propulsion control variables, log options, and flags that enable or disable each of the main devices. Each component of the system reads information that it requires for initialization from this configuration file and attempts initialization accordingly.

Device-Software Interfaces

Devices that are treated as self-managing components include the propulsion, Doppler velocity logger (DVL), altimeter, acoustics, pressure sensor, digital acquisition card, and cameras. All of these components except the pressure sensor and the DAC create their own parallel execution thread during initialization. The threads that handle the DVL, altimeter, acoustics, and cameras each perform input and output in a continuous loop, delivering data and updating shared information without blocking the execution of the rest of the system. For instance, the DVL and altimeter components are each constructed with knowledge of a RobotState object. As each component receives data (in the case of the DVL and altimeter, over the RS232 serial ports) it updates the appropriate variables in the RobotState object. Concurrent access to shared data is protected with synchronization variables. Once all components are initialized, the system enters a loop of event generation and handling.

Propulsion Control

The propulsion component receives requests for waypoints and other navigation, and processes these requests by continuously calculating the desired force and torque to exert on the vehicle as a function of its spatial state at the time. The component then converts these forces to desired thrusts to be delivered by each of the four thrusters. Each thrust is converted into a voltage using a cubic regression model of the control voltage/thrust relationship of the thrusters (which is nonlinear). All of the variables that affect the translation of waypoint to thrust and thrust to voltage are exposed in the configuration file for easy tweaking. The propulsion component accesses a RobotState object from which it determines the current position of the vehicle at any time, as well as the access to the DAC component used to change the thruster outputs. Because the propulsion component is completely self-contained, the entire propulsion system of the robot could be changed without any of the rest of the code changing. The interface with the new propulsion system would be identical.

Mission Control Logic and Waypoint Entry

The control system is event based, meaning it changes the behavior of the vehicle by responding to certain pre-programmed events. Most of these events are generated by the system itself, and at any time the current behavior of the robot determines how it responds to events. Important examples that behaviors handle are enter, think, leave, and frame events. The enter event is generated when a behavior object is activated for the first time. The think event is generated periodically (typically ten times a second) so that the active behavior can perform its processing. The leave behavior is generated when the current behavior has finished its processing and a new behavior is about to be activated. The frame event is generated by the camera components when new frames are received from the cameras.

Consider the Waypoint behavior as an example of how a behavior processes events. When the Waypoint behavior receives the enter event, it requests a new target position and heading via the propulsion component. Once the request is made, the propulsion system concurrently tries to achieve the target. The Waypoint behavior responds to the think event by checking whether the vehicle has come within a specified tolerance of its target, or whether too much time has elapsed. When either of these conditions is met, it notifies the system that it has completed, and the leave event is generated.

Any behavior can queue other behaviors for later execution or push behaviors ahead of itself on the stack, allowing complex behavioral control at runtime. Furthermore, the frame event is processed in a separate thread of execution, so that longer-running image processing routines do not interfere with the handling of think events.

To further assist rapid modification of behavior and testing, the control system supports a custom scripting language created specifically for AUV control. The scripting language resembles C++ or Java, with the exception that it is loosely typed. It supports real numbers, strings, three-dimensional vectors and lists as primitive types and allows arbitrary compound user type declarations (similar to classes in C++ or Java, but without polymorphism). User types that have certain methods such as onThink(...) and onFrame(...) can be used as behaviors. At runtime, script files are read, parsed, and translated to virtual machine code. When an event is handled by the system, the system passes it to the active behavior. If the active behavior has an associated scripting object, a virtual machine runs the object's appropriate event handler method. Because the scripts are interpreted, there is no need to recompile to test a change, and arbitrary behavior can be expressed in the scripting language (as opposed to simply expressing data to be used by compiled code). Any real processing is done by native C++ functions with an interface exposed to the scripts, so that there is no performance loss when processing images. The scripting engine is also employed to process the configuration file, so that arbitrary expressions such as "dvl.offset_from_center = [0,0.5,0];" can be entered as settings.

Vision and Cameras

Charybdis has two possible vision systems that can be used. The first is based on digital USB webcams; the second consists of analog underwater cameras combined with a digital frame grabber.

Charybdis' digital system uses two Logitech QuickCams for vision-based navigation. The QuickCams are installed in a waterproof tube and connect to the Electronics tube through a Subconn cable. The cameras are interfaced with the Versalogic computer using standard USB cables. The cameras provide 640 X 480 video feed, with optional 1.3 Mega pixel still images and zoom features. The computer is able to process about 10 frames/sec, allowing the AUV to detect interesting features even while moving at full speed through the water. One camera is downward facing in order to detect markings along the bottom of the pool. The other camera faces forward in order to detect the LED lights of the docking station. The cameras gather and store images on the computer while the software analyzes the images for shapes that signify LED sources and shapes.

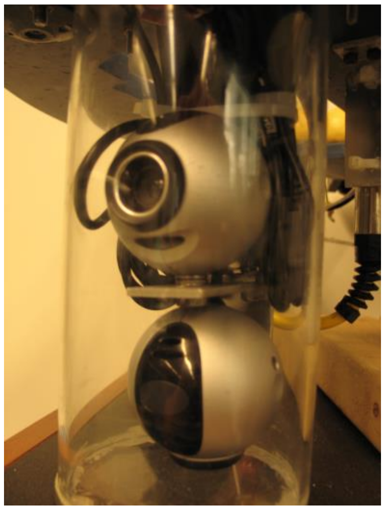

Figure 8: Cameras

For analog vision, Charybdis has a Parvus FG104 frame grabber combined with two underwater cameras. The frame grabber has 4 composite analog video inputs and can capture at a resolution of 768x576 at 30fps. The card utilizes the PC/104 bus. The frame grabber is used in combination with analog video cameras to provide vision for the AUV. The cameras are mounted in the same orientations as the webcams, except without the need for air tubes. Vision is processed in the same manner as for the webcams.

While seeking the horizontal light positioned near the floor targets, the event handler for the frame event first checks the amount of cyan in the image. If this is below a threshold, then the system considers the light absent from the image. If the threshold is met, the handler finds the center of mass of cyan in the image and considers this the center of the light. Next, the vehicle modifies its target waypoint to correspond with the direction of the light.

To find the pipe break, each frame's gradient field is computed, and this field is thresholded by Euclidean magnitude to identify edge pixels. Then connected components (corresponding to non-edge regions) are identified with a queue-based fast flood fill algorithm. Very small regions are discarded. The star-shaped boundary of each such region is computed with the center of mass used as the center of the star. This boundary is simplified by joining segments that are nearly collinear to specified tolerances. The result is a coarsened shape boundary for each region, represented by a simple polygon.

To detect whether these shapes correspond to the shape of the pipe break, each boundary's four longest segments are determined, and these edges' intersections are taken as corners. If the estimated corners have angles within specified bounds, the shape is considered a quadrilateral, and its position is marked as a possible pipe break. The vehicle finds as many “breaks” as it can, then drops a marker over the best defined break. All of the processing occurs in real time (under 100 ms for 10 fps camera feed) on the single CPU.

Conclusion

Charybdis is Duke’s second generation of an AUV; Charybdis is in its own third revision. Undoubtedly there are flaws in the design or improvements that can be made, but this vehicle represents the total combined knowledge that the Duke Robotics Team has gained from Charybdis 2005, Gamera, and background research. After improving the vision, power, acoustics, and mechanical systems, Charybdis is far less prone to failure and much more capable. Most systems, whether electrical, mechanical, or software, contain new innovations and improvements, and all components have been carefully selected or constructed to conform to the highest standards of craftsmanship and upgradeability. The synthesis of these components is a robot that the Duke Robotics Team believes will be exceptionally competitive in San Diego at the ONR/AUVSI Underwater Competition, and will certainly inspire new adaptations of its cutting-edge design.

References:

Duke University Robotics Team http://**robotics**.pratt.**duke**.edu/archives/2005-2006/
ONR/AUVSI Underwater Competition http://www.**auvsi**.com/competitions/water.cfm
Subconn Connectors http://www.**subconn**.com, http://www.**deepocean**.com
Thrusters http://www.**tecnadyne**.com
DVL http://www.**rdinstruments**.com/
Pressure Sensor http://www.**msi**usa.com
Computer http://www.**versalogic**.com
DAC http://www.**diamondsystems**.com
Batteries http://www.**thunderpower**-batteries.com//
Hydrophones http://www.**reson.**com
Slackware 9.1 http://www.**slackware**.com
libjpeg for image IO http://www.iig.org
libpthread for POSIX threading http://pauillac.inria.fr/~xleroy/linuxthreads/