California Institute of Technology Autonomous Underwater Vehicle: Development and Testing of the AUV "Bruce"

Justin Koch (leader), Joel Burdick (mentor), Kushal Agarwal, Srinivasa Bhattaru, Solomon Chang, Christopher Cousté, Erin Evans, David Flicker, Edward Fouad, Bryan He, Juliana Kew, Jake Larson, Gregory Izatt, Tyler Okamoto, Torkom Pailevanian, Jeffrey Picard, Tatiana Roy, Alejandro Sanchez, Xi Xu, Frank Zhou

Abstract

"Bruce" is an entirely student-built littoral class autonomous underwater vehicle (AUV) built by undergraduate members of the California Institute of Technology Robotics Team (CRT) with the intention of entering the annual AUVSI and ONR International RoboSub Competition in 2014 for the first time. Beginning in October 2013, the team designed, manufactured, assembled, and began in-water testing of Bruce by February. This year's AUV features an adjustable frame with 6 thrusters and 4 pressurized hulls modularized for future flexibility. The primary waterproof hull houses the main computer with an Intel core i7 processor, a power distribution system, a custom designed optoisolator board, and electronic speed controllers. The vehicle also includes an entirely new custom software stack with testing modules to allow for development prior to vehicle completion.

Introduction

The Caltech Robotics Team researches new solutions to existing problems in robotics through competing in various robotics competitions. We aim to gain both knowledge and experience of the various disciplines which comprise the field of robotics, including computer programming, electrical and mechanical engineering, and systems engineering. This year the team has designed and built an AUV for the AUVSI and ONR International RoboSub Competition in July at the TRANSDEC facility in San Diego (Figure 1). At this competition, each university's student-built entry will compete by autonomously completing a timed course whose obstacles mirror realistic AUV missions. These missions vary year to year, and include recognizing and manipulating objects, navigating, shooting markers, locating sonar pingers, and swimming through and above various obstacles.

Mechanical System

We implemented an open frame design to facilitate the design and construction processes. We started by constructing a minimum viable product, which consisted of the frame, thrusters, pressure hull, and battery box, so that the Programming Team could begin testing as soon as possible. This allowed us to add the remaining components incrementally as they were completed, such as the cameras and DVL. The vehicle is slightly positively buoyant and maintains an upright orientation both while floating on the surface and while completely submerged. All custom mechanical components for the vehicle were machined by team members in-house through the use of the Caltech Mechanical Design and Prototyping Laboratory.

I. Chassis

We constructed the Chassis from 1 inch 80-20 structural channel. This flexible, modular design allowed us to add components incrementally, and to easily reposition components for thruster optimization and buoyancy adjustment. We machined our own customized T and L brackets to robustly secure the members of the frame together. All of the frame members and brackets were anodized for corrosion resistance. The finished frame design can be seen in Figure 2.

II. Propulsion

To maneuver the vehicle, our design consists of six VideoRay thrusters with two forward thrusters, two vertical thrusters, and two lateral thrusters. This gives us control over five degrees of freedom: forward translation, vertical translation, lateral translation, pitch, and yaw. Since we never require actuation of the roll axis, we simply use passive buoyancy stabilization to prevent undesirable roll rotations.

Since the VideoRay thrusters are designed to mount to a watertight compartment, we needed to design a way to integrate them into our design. Our design concept can be seen in Figure 3. We designed and fabricated a watertight compartment that mounts to each thruster. A waterproof connector screws into the compartment, and a corresponding cable connects the thruster to our pressure hull. The compartment contains an o-ring groove, and a lid screws on with four countersunk screws to make a seal. The lid has two flanges to bolt to the 80/20 channel, allowing the thrusters to be easily placed and repositioned.

III. Pressure Hull

The pressure hull houses the main electrical system of the vehicle. The design is shown in Figure 4. It consists of two end caps and a clear, acrylic tube. One end of the tube is bonded using marine-grade epoxy to a groove inside the fixed end cap. The fixed aluminum end cap contains two flanges that allow for mounting to the 80/20 frame. To the other end of the tube is bonded a circular, aluminum lip with grooves for two concentric o-rings, and 12 radially distributed tapped holes. The removable end cap mounts to this end, creating a double o-ring face seal with the lip.

The removable end cap contains all of the vehicle's bulkhead connectors. It is also secured to three perpendicular plates that support the electronics. This scheme allows the pressure hull to be opened without stretching or tangling any of the internal wires.

To validate the pressure hull and check for leaks, the removable end cap contains a Schrader valve and pressure gauge. We pressurize the hull with a bicycle pump the day before each underwater test, allow it to sit overnight, and verify that the pressure has not dropped when we arrive the next day. This was particularly valuable on the earlier iterations of the pressure hull, as it allowed us to diagnose problems in the hull and find sources of leaking.

The pressure hull is mounted vertically in the center of the vehicle. This allows the buoyant pressure hull to have the most positive stabilization effects on the vehicle. In addition, it allows the hull to be easily opened by pulling the removable end cap vertically out of the vehicle.

IV. Battery Box

The mechanical design of the battery box is almost identical to that of the pressure hull. The design is shown in Figure 5. It too contains a clear acrylic tube, fixed and removable end caps, and an aluminum lip for a double o-ring face seal. The battery box houses two 6S 10 Ah LiPo battery packs, which are connected to the pressure hull by a single 4-pin cable. The battery box is mounted to the very bottom of the vehicle due to the buoyancy stabilization effect of its large weight.

V. Camera Boxes

We constructed two camera boxes to house a forward and downwards facing Point Grey Bumblebee stereoscopic camera. The design is shown in Figure 6. The camera boxes implement the exact same design as the pressure hull and battery box. A single plate extends inwards from the removable end cap to support the long, rectangular camera.

The cameras bolt to the 80/20 frame using two custom-machined brackets. These brackets give the camera box 180 degrees of rotational freedom, allowing us to tweak the orientations of the cameras before locking them in place.

VI. DVL Housing

The SonTek Argonaut Doppler Velocity Log consists of the head, where the emission and receiving take place, as well as several electronics boards to process the raw data. All are mounted inside a customized waterproof housing, whose design is shown in Figure 7. The head consists of two concentric, radial o-rings, so we designed an aluminum insert to match it. The insert, which is epoxied to the end of a clear acrylic tube, has a tapered inner-diameter that allows the radial o-rings to make a seal as the head is screwed on. The other end of the acrylic tube is capped by a solid aluminum insert, containing an 8-pin bulkhead connector and a Schrader valve with a pressure gauge to validate against leaking.

Since the electromagnetic noise of the thruster speed controllers inside the pressure hull was so large, it skewed the magnetometer readings of our Attitude and Heading Reference System (AHRS). Therefore, we decided to mount the AHRS inside the DVL housing to shield it from magnetic interference. The 8-pin cable that interfaces with the DVL housing is able to transmit the signals from both the DVL and the AHRS. The DVL mounts to the frame via two brackets that conform to the curvature of the acrylic tube and hold it in place. It faces directly downwards to detect velocity relative to the bottom of the pool.

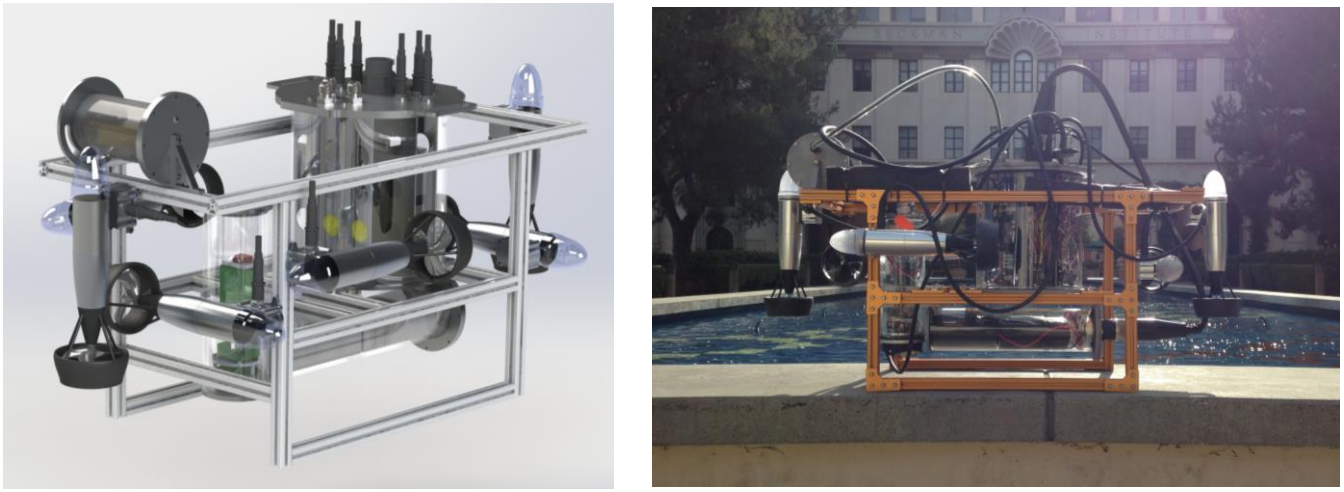

Figure 1: CAD Model and photograph of the Completed Vehicle

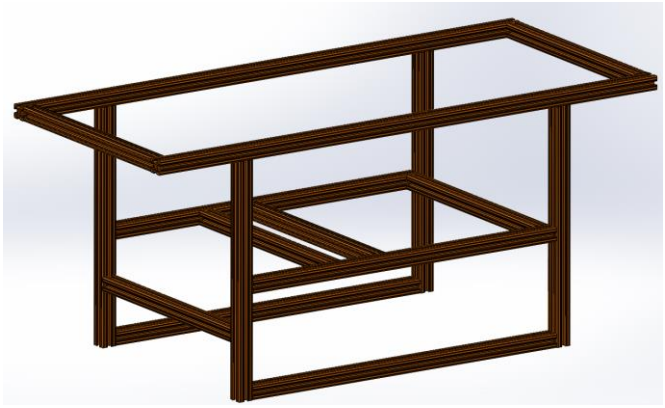

Figure 2: Chassis Design

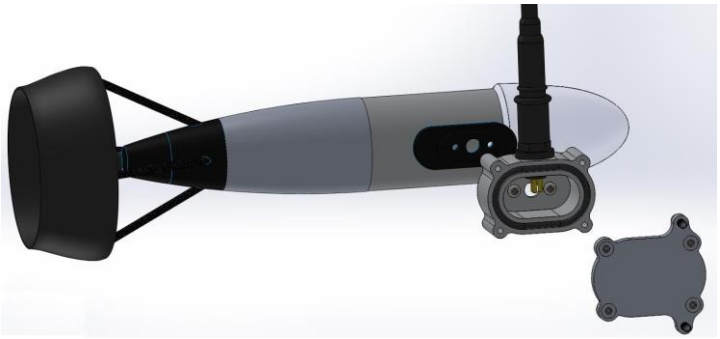

Figure 3: Custom Mounting Scheme for VideoRay Thrusters

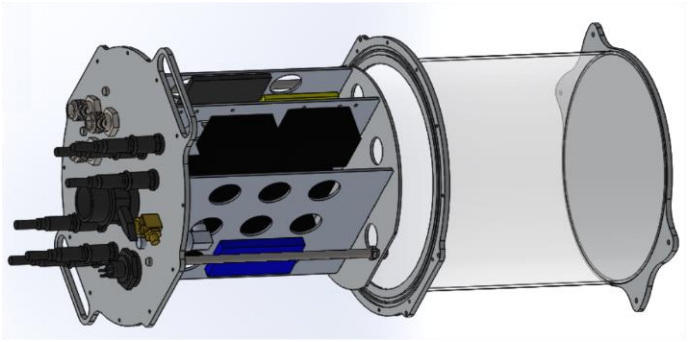

Figure 4: Pressure Hull Design

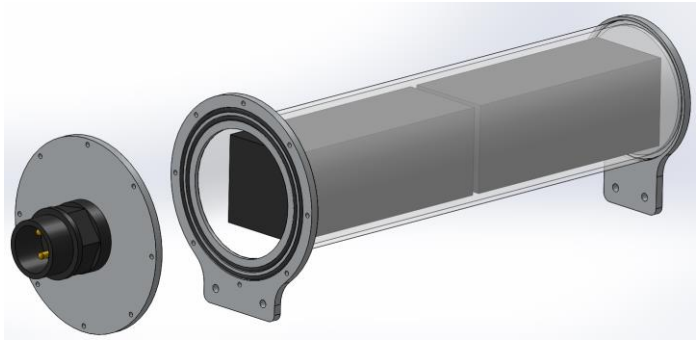

Figure 5: Battery Box Design

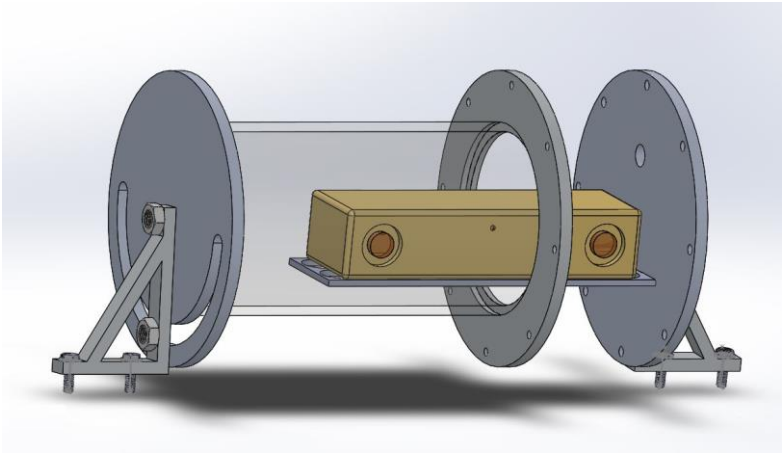

Figure 6: Camera Box Design

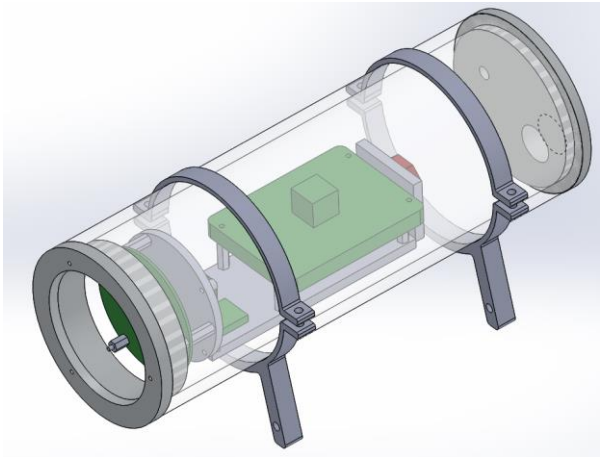

Figure 7: DVL Housing Design

Electrical System

The vehicle's electrical system is the actuation and sensing component of Bruce. It provides the computing hardware and information from the sensors for the software system and control over the thrusters for the mechanical system. The electrical system is designed to be simple and modular. We chose off-the-shelf components as much as possible to reduce development time and risk compared to developing everything ourselves from scratch. Therefore, we could focus our development time and effort on the components that are truly unique to Bruce.

I. Power system

Bruce is powered off of two 6S 10 Ah LiPo battery packs that provide plenty of energy for the system. Each battery supplies a totally separated and isolated power system. One runs the thrusters and the other runs the computing and sensing systems. We chose to separate the power systems because it keeps the computing side power very clean and it gives us more modularity over supply voltages and energy requirements.

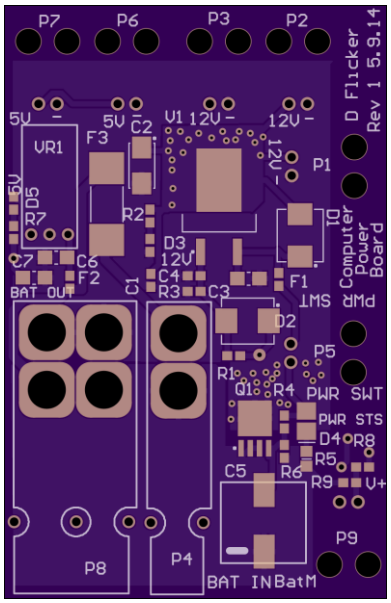

Figure 8: Computer Side Power PCB

To better manage the system power, we designed a custom computer side power PCB that provides regulated +12V and +5V rails and a divided battery voltage signal for input into a 0 - 5V ADC. In addition, there is a computer-side kill switch to completely disconnect power to the electronics and inrush current limiting to prevent arcing on the battery contacts and stress on the components. The main computer is powered using an m4 DC/DC ATX PSU.

The motor-side kill switch is a reed-switch mounted on the inside of the top end-cap that is the control input to a relay that completely disconnects power to the motor side.

II. Computing system

The computing system is exclusively made of off-the-shelf components, which significantly reduced our development time and allowed the software team to get an early start on their tasks. The main computer is an off-the-shelf microATX motherboard with Intel core i7 processor, which gives us both full expandability with a variety of PCI and PCI-Express slots and plenty of computing horsepower.

For low-level thruster control and sensing, we have selected the Arduino Mega as the onboard microcontroller development system. The Arduino development libraries allow us to write firmware in a fraction of the time it would have taken to develop everything from scratch. This speed is key because, as a rookie team, we are still changing our sensor suite on a monthly basis. The Arduino connects to sensors that have only a low-level interface such as analog output, SPI, I2C, or TTL serial. It has on-board peripherals for these interfaces and already-developed Arduino libraries for communicating with them. The Arduino connects to the main computer over a UART to USB bridge. The Arduino and main computer communicate using a custom serial protocol that allows us to define different message types quickly so we can get new sensors integrated as fast as possible.

Finally, we created a WiFi buoy that connects to the vehicle using an underwater Ethernet cable. This allows us to connect to and control the vehicle from land while the vehicle is underwater. This system is invaluable for quickly diagnosing and troubleshooting issues while testing.

III. Sensing

A. VectorNav PN-100 Attitude and Heading Reference System (AHRS)
   Bruce depends on having accurate and drift-free attitude information to accurately determine the absolute roll, pitch and yaw axes. We selected this AHRS system because it has a dynamic heading accuracy of less than two degrees and provides quaternion output with an onboard extended Kalman filter algorithm.

B. MPL115A1 Barometric pressure sensor
   The barometric pressure sensor measures the internal pressure of the main hull, which allows us to determine whether the hull is leaking air after pressurization.

C. Omega PX-319 external pressure transducer
   This sensor measures the pressure outside the vehicle, allowing us to determine our depth underwater. The PX-319 provides a 0V - 5V output signal that directly connects to the Arduino Mega's ADC. The signal is extremely linear and clean requiring only a small amount of oversampling to determine an accurate depth.

D. SonTek Argonaut Doppler Velocity Log (DVL)
The DVL provides accurate surge and sway velocities at 1 Hz. The DVL is a key element in the vehicle's control system, as the accurate velocity data can be integrated once to provide a position estimate. Since GPS does not function underwater, and there is no acoustic baseline, the DVL is the only reasonable way to achieve an accurate position estimate.

E. Point Grey Bumblebee stereoscopic cameras
These cameras provide the critical vision information to the main computer. They are connected to the main hull using an air tube, as the high-speed FireWire800 signal needs a carefully shielded and impedance matched cable. One is mounted facing forward, and the other is mounted downward.

IV. Actuation

The Arduino Mega connects six PWM outputs (one for each thruster) to our custom designed optoisolator board. The optoisolator board transfers the PWM signal from a computer side voltage into a motor side voltage. Then, PWM signal is sent to each of six Hobbywing EZRUN-150A-PRO electronic speed controllers (ESC). These ESCs power the brushless DC motors inside each thruster.

Software

All the higher level functionality of the vehicle is achieved through a custom developed software system. The software stack is built upon an Ubuntu Linux operating system that includes a custom shared memory, message communication, vision, and control system. All software was implemented with C/C++ with a Make build system.

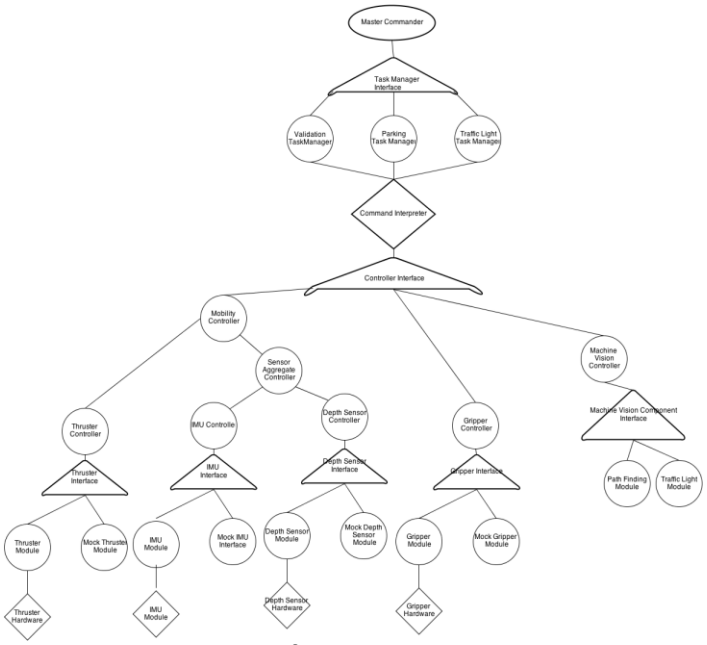

Figure 9. Software stack

I. Design Paradigm

As shown in Figure 9, the software system involved heavy use of interfaces and mock modules. This design paradigm of dependency injection facilitated testing procedures and greatly accelerated development of the software system. Developers were able to easily test their new code with the mock modules instead of consuming testing time with the actual robot.

II. Interprocess Communication

Our software system utilized two distinct methods of interprocess communication. Each of the individual interprocess communication libraries focused on transferring a different data size. This not only allows each library to optimize itself for a certain size, but also allowed the potential use of a fallback interprocess communication should one library experience a problem.

A. Shared Memory System

The shared memory system provides an interface for processes to store large amounts of data within the standard POSIX shared memory. This interface allows multiple processes to access the same underlying memory and communicate otherwise inoperable amounts of data. For our specific application, the images captured by the cameras were all transferred via this shared memory system. Our shared memory system provided controlled access to shared memory, eliminating race conditions and data integrity issues.

B. Message Communication

The message communication library is a custom built library designed to pass encoded messages over TCP ports from process to process. This method is the primary vector for which processes communicate the state of the vehicle and control signals. The library utilizes the encoding functionality of Google Protocol Buffers for serializing and deserializing data messages. This system was built to quickly transfer small amounts of data to allow the vehicle to quickly respond to sensor input.

III. Vision Module

The vision module is represented within our software stack as a single process consisting of multiple threads; each of which was responsible for identifying a unique field element. This centralization within a single process allowed all the vision logic to be self-contained, which facilitated signaling which field elements the vehicle was interested in. The division of threads allows faster running machine vision modules to avoid bottlenecking. Each of the vision threads contained a hand tuned object recognition system utilizing the OpenCV framework.

IV. Mobility Control Module

The mobility controller is responsible for all the actuation of the vehicle; for this year, this is limited to the six thrusters. The controller receives the current state of the vehicle and the desired position and attitude. Using various hand-tuned PID controllers, it computes the resulting PWM values to send to the thrusters. This centralization of the actuation logic of the vehicle allows for quick control logic changes and easy debugging.

V. Task Managers

Each task the vehicle needs to perform (i.e. validation gate, path following) has a corresponding task manager that is responsible for the logical sequence of commands to complete the task. This division of the task logic into each task manager allowed for module testing of each task.

VI. Command Dispatcher

The command dispatcher is the centralized module that receives commands from each task manager and dispatcher the command to the appropriate submodule (A "Go Straight" command must go to the mobility control module). This abstraction for the task managers allowed task managers contain high-level logic without concern over which module to send the commands to.

VII. Master Commander

The master commander module is responsible for ordering all the tasks to be perform for the competition. It is responsible for ordering the various tasks and conducted the resource handoff between each of the tasks. This parent module of all the task logic allowed for easy preservation and cleanup of resources from task to task.

Acknowledgements

The CRT would like to thank Caltech for all its support during the past academic year, particularly within the Engineering and Applied Science division. We would like to specially thank our faculty advisor and mentor, Professor Joel Burdick, who has given us direction through our many hours of development and testing. Additional thanks goes out to Maria Koeper, Perry Radford, Jannah Maresh, Matt Burkhardt, John Van Deusen, Professor John Dabiri, Brian Penserini, Heather Morgan, Betsy Mitchell, John Leichty, Glen George, Tom Mannion, and all the gym and pool staff that have helped us through numerous hours of testing. Finally, we express our immense gratitude to all of our gracious sponsors for the 2013-2014 academic year!

Ocean Level: Autodesk, Cooper Interconnect, L-3 Ocean Systems, Thales Avionics, and VideoRay

Sea Level: Impresa Aerospace, Teledyne Impulse, VectorNav, and Western Digital

Lake Level: AA Portable Power Corp., Danco Anodizing, Tom and Patricia Ganz, Marvin Engineering, Satellite Consulting, John and Mary Schroder, and Tom Strickland.

Pond Level: Joel Burdick, Jim Conte, Brock Jones, Peter Payne, Eugene Vajintsky, George and Anne Watts