Remote Communication for Monitoring and Control of Autonomous Underwater Vehicles

J. H. Kim[1] *J. J. Leonard*[2] *J. G. Bellingham*[1] *B. A. Moran*[1] *S. T. Tuohy*[3]

[1] MIT Sea Grant College Program, Underwater Vehicles Laboratory
[2] MIT Department of Ocean Engineering
[3] Draper Laboratories, Simulation Lab
Cambridge, MA 02139, USA

Abstract - This paper describes research on remote monitoring and control of autonomous underwater vehicles (AUVs). This work is part of a larger effort to create autonomous ocean sampling networks (AOSN), an exciting new concept for collecting synoptic oceanographic data. AOSN is based on the operation of small, low cost autonomous underwater vehicles (AUVs) within an array of moorings that provide communication and power-recharging nodes. Critical to the realization of AOSN will be the ability to control these assets and monitor the data collection process remotely. During the time in which our AOSN partners have been developing acoustic communications, our laboratory has performed a variety of remote monitoring and control experiments to test different types of network distributed control software. Experimental results are presented for runs in the Charles River in which a radio modem towed on a float was used to monitor and control the vehicle from a workstation on the internet. In addition, we have used our acoustic navigation system to send simple command messages to an Odyssey vehicle, achieving simultaneous control of multiple AUVs operating in a high current environment in the Haro Strait.

I. INTRODUCTION

The Autonomous Ocean Sampling Network offers a new paradigm in oceanography — one in which the scientist need not be onsite to gather data [11]. This capability is already in place using satellite imagery, however, these measurements are limited to the top layer of the oceans. Trying to understand oceanographic phenomena solely from these measurements is akin to trying to understand atmospheric weather based only on ground measurements. AOSN is directed to provide the essential capability of gathering oceanographic data in all three spatial dimensions.

In the AOSN, a fleet of small, inexpensive Autonomous Underwater Vehicles (AUVs) work together to collect data. These AUVs are controlled from and communicate with a network of moored nodes via acoustic modems. The nodes also provide a dock for recharging batteries and transferring large amounts of data, as well as a radio or satellite link with the shore. By connecting the shore station with a global communications network such as the Internet, the AOSN allows scientists to carry out "real-time oceanography" – receiving data from and redirecting AUVs as the changing situation warrants – without ever having to step aboard a ship. Three links are seen in the final AOSN system: 1) AUV to node (via acoustic modems), 2) node to shore (via radio or satellite), and 3) shore to scientist (via then Internet).

While acoustic communication technology is being developed by collaborating institutions in the AOSN team, we have conducted a variety of experiments using off-the-shelf hardware for communication. The motivation has been to explore the issues in remote monitoring and control of AUVs so that a reliable software base will be in place when acoustic modem technology has matured.

Using a radio modem on a float towed by the vehicle in the Charles River, we have demonstrated data telemetry from and control of an AUV by communicating over the Internet from a workstation in our lab. We have also accomplished this capability using a computer aboard ship in Buzzard's Bay. In addition, we have used the radio modems to "log in" to the the vehicle computer and launch missions in different demonstrations across campus and across the country. Finally, we have used a spare acoustic channel of the Odyssey long-baseline (LBL) navigation system to send simple command messages to an Odyssey vehicle. This enabled us to simultaneously control multiple Odyssey AUVs operating in a high current environment in the Haro Strait off the coast of British Columbia.

II. THE ODYSSEY CLASS AUVS

The MIT Sea Grant Odyssey class AUVs represent a different approach from many AUVs currently in use and development [5, 4]. The Odyssey AUVs are designed to provide a lightweight, flexible, affordable oceanographic survey platform which can be operated by a small crew using a minimum of support equipment. Instruments can be swapped in and out as they are needed, saving both weight and power. To date, our laboratory has built six Odyssey AUVs, which have seen extensive field testing.

The Odyssey software suite is a constellation of applications which are used to prepare, test, run, and analyze missions [7]. These include: mission configuration software, vehicle control code, data parsing routines, mission data analysis/visualization, simulation models, and data structures. The high-level control software is a behavior based layered control architecture, which allows relatively complex behaviors (e.g. a vertical yo-yo concurrent with a horizontal survey) to be easily produced [3, 7]. The vehicle software has been proven reliable in extensive field operations with Odyssey II AUVs over the past several years, including operations in under-ice, deep-ocean, and high-current environments.

The Odyssey software architecture has been designed to simplify the task of supervisory control [6]. Monitoring the execution of a vehicle mission is based on selecting and reporting desired values from the vehicle sensor data structure [7]. Control of the vehicle is effected by downloading individual parameters such as behavior arguments, or by changing the vehicle mission completely by downloading a new mission file. Each of these capabilities has been demonstrated in different experiments over the past year.

III. ACOUSTIC COMMUNICATIONS

Acoustic communication in the ocean is limited by five processes not typically encountered in other communications channels. Sound spreads and is absorbed by various oceanic processes as it travels. Natural, biologic, and man-made noises are always present and can mask sounds of too low intensity. Turbulence and other variations in temperature, density, and water composition cause fluctuations in the local sound speed. The presence of a surface and bottom and a gradual change in the sound speed profile with depth gives rise to multipath. Finally, sound travels at a relatively slow 1500 m/s, which results in long round-trip signal travel times for for a typical AOSN deployment size of 10x10 km.

The absorption of sound is frequency dependent, with higher frequencies being absorped more quickly. This process often imposes an upper bound on the usable frequency of acoustic communications. Ambient noise tends to increase as the frequency drops below 10 kHz, and this along with the lower throughput of lower frequencies often imposes a lower bound on the usable frequency. Changes in the local sound speed leads to fluctuations in signal amplitude and phase, and may require the use of lower-throughput incoherent demodulation [10]. Multipath causes variations in arrival times from a single signal, requiring complex processing to extract the original signal. The long travel times involved make retransmission an impractical means of error correction, and redundancy and error correction coding must be added to the original signal [9].

In the last decade, advances in digital signal processing, acoustic channel modeling, and communications theory have led to improved performance of underwater acoustic modems. On the scale of the AOSN, transmission rates of 1 to 10 kb/s can be expected [10]. While these speeds are equivalent to a slow dialup modem, there is no real parallel other than the peak performance rate. The maximum transmission rate will change as the two modems change position in the water and relative to each other. Transient phenomena such as noise can cause reliability problems unavoidable even with error correction. The time delays involved make it impossible for the vehicle to instantly respond to commands or data requests. In addition, in a multivehicle environment, several modems may be competing for the same frequency space, and a procedure for allocating acoustic bandwidth will have to be developed.

IV. INTERNET

The Internet was designed to provide reliable high speed and wide bandwidth communications, and compared to underwater acoustic communications imposes few restrictions on communications. Network programming standards such as Sockets allow virtually unchanged source code to work on nearly all types of UNIX computer systems [13]. A potential difficulty arises from the Internet using the Transmission Control Protocol (the TCP in TCP/IP) - a point-to-point protocol - as its transport level protocol.

Point-to-point protocols were developed with only communications between two machines in mind, and are wasteful of network resources if many machines request the same information. One machine establishes a connection with the other, and the two transfer data. If a third machine wants the same data, it must open another connection and the first machine must retransmit it. As more machines ask the first machine for the data, the network bandwidth becomes saturated transmitting the same thing over and over. This is normally not a problem if only a few connections or small amounts of data are required, but otherwise the level of network traffic can quickly exceed the available bandwidth. For example, in the Haro Strait experiment, the link to the Internet was nothing more than a modem and telephone line. Had many researchers around the world wanted to view the raw data as it came in, the bandwidth would have quickly been saturated. The problem is even worse if several machines must all share data between each other, as the case may be if distributed computing resources are used to analyze the incoming data. The required network bandwidth then goes from scaling linearly to polynomially.

Since the problem is inherent in the protocol that forms the basis for the Internet, there is no simple way around it. One could use a point-to-multipoint protocol such as the User Datagram Protocol (UDP), but then it couldn't propagate over the Internet. Fortunately, this is not a new problem, and the Multicast Backbone, or MBone, was created to address it. The MBone uses special routers available on a portion of the Internet to "tunnel" UDP packets over the Internet's data lines. Data which is multicast is sent over the entire MBone, but only once. All machines interested in the data can listen in on the single transmission rather than the originating machine having to send a separate copy to each machine.

A potentially useful multicast standard is Distributed Interactive Simulation (DIS), which allows multiple wargame simulations to interact with each other in the same virtual environment [8]. While initially DIS appeared attractive, we decided the extraneous data fields in a DIS packet (e.g. projectiles fired) were contrary to our attempts to minimize the amount of data transmitted, and thus decided against using DIS to encode data. Still, there is a large amount of DIS source code freely available, and it may be a useful standard to investigate in the future for AOSN.

We initially decided a key criteria for any Internet communications software developed should be cross-platform capability. Unfortunately, at the onset of this research, it was difficult to produce and maintain network code that would work work with Unix, Macintosh, and PC compatible computers. The recent growth of the World Wide Web (WWW) and introduction of Java has changed this. The Virtual Reality Modeling Language (VRML), a 3-D extension to the WWW, shows promise as a means of displaying AOSN data via a web browser on any platform, using the WWW as the medium for network data transmission. The wide acceptance of Java as a web browser extension on nearly all computer platforms means the same network source code could run on all these systems without modification. However, since the WWW is based on TCP/IP, both VRML and Java suffer from the same bandwidth problems mentioned previously. Future work will have to consider the benefits the cross-platform compatibility of the WWW offers against the improved network performance of the MBone and multicast routines.

V. EXPERIMENTAL RESULTS

We have conducted many experiments operating one of the MIT Sea Grant Odyssey II AUVs remotely. Since an acoustic modem pair were unavailable, we used a pair of radio modems to simulate the acoustic link. We ran a tether from a serial port aboard the vehicle to a towed float carrying watertight plastic box. The first radio modem, with an omnidirectional antenna mounted on the outside of the box, was sealed inside this box. This modem was powered by a sealed lead-acid battery, also inside the box. The second radio modem with a directional antenna hooked up to a Unix workstation and acted as the receiving node and shore station. For our Charles River tests, the workstation was also connected to the Internet. During the Buzzard's Bay experiments, the workstation was aboard a ship and hence not attached to the Internet.

We modified the vehicle software to transmit a subset of the data logged, including heading, depth, altitude, and dead-reckoned position, at regular intervals (1 sec in our initial tests) to the serial port for transmission via modem. On the workstation, a short program called "comm" checked its serial port for incoming data from the modem. The routine tested the data packet for errors, then, if it was error-free, multicast it over a free MBone IP address. On the receiving (user-end) machine, a Matlab script called an external C program which checked the MBone address for incoming data. If it received any data, it was again checked for errors then displayed on an xy plot and a depth/altitude plot. The script also accepted user input to abort the mission or to reconfigure the current setpoint. Any user commands to the vehicle went through the reverse procedure, being multicast, received by the "comm" program, and sent over the modem back to the vehicle. Because all the data transmission between the "comm" program and the user-end machine are multicast, it does not matter where the two machines are on the Internet, or even how many user-end machines there are. All user-end machines receive and display data, and all can issue commands to the vehicle. For research purposes, the logistical and security problems this caused were not deemed important, but this area will need work in the future.

The original intent was to use Matlab as a cross-platform display and control interface, with a different C routine for each computer platform containing platform-specific network code. However, Matlab scripts appear to run as a single process, which causes enormous performance difficulties. The graphical display needed nearly half a second for each update, during which it was impossible to type or send commands to the vehicle. The net result was a control panel that felt unresponsive or seemed to be "locked up" most of the time. For this reason we have decided any future display and control interface must be platform specific, or have support for threads (multiple processes, e.g. Java).

A. *Charles River Experiments*

For the Charles River experiments in October and November of 1995, we placed the Unix workstation and the second modem with directional antenna at the MIT sailing pavilion. An ethernet drop at the dock provided the connection to the Internet. The user interface package usually ran on this workstation, but data was always multicast over the MBone (deliberately limited to only MIT's network), so anyone on MIT's network could have observed the incoming data at any time.

An initial series of trials only receiving telemetry from the vehicle without attempts at control yielded fair reception. The modem's range seemed to extend approximately 300 meters, but there were periods of 5 to 20 seconds during which we received no telemetry. Analysis failed to yield a pattern, but our primary suspect was the underwater cable between the vehicle and float, which may have had a faulty splice.

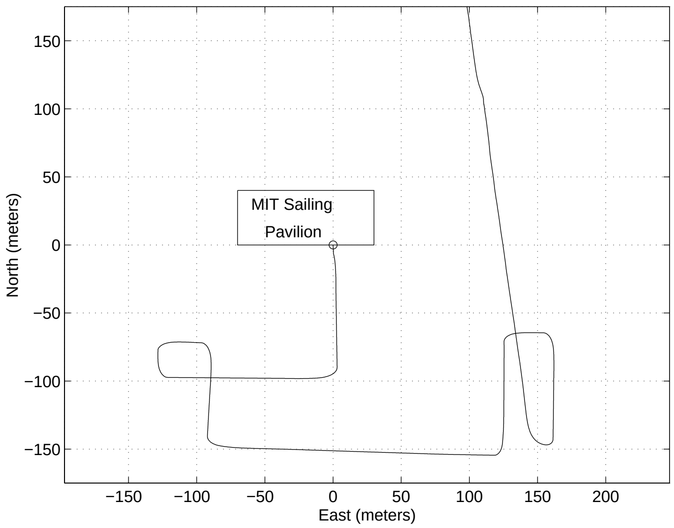

Figure 1. Dead-reckoned vehicle trajectory for early Charles River experiment.

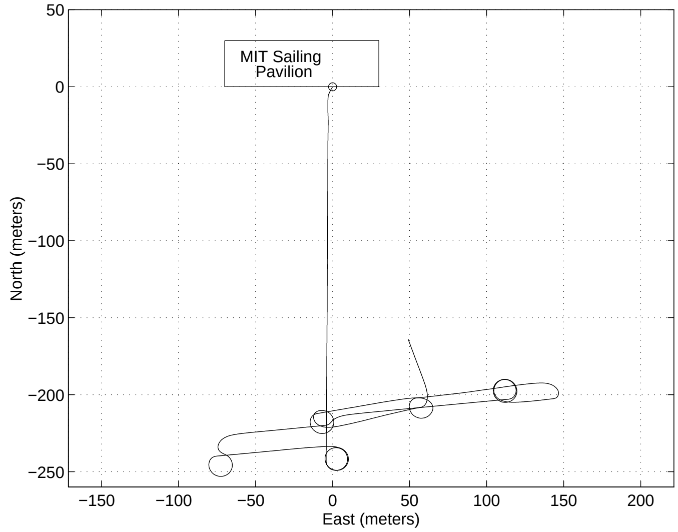

Figure 2. Dead-reckoned vehicle trajectory for Internet controlled Charles River experiment.

We decided to try commanding the vehicle heading despite the dropouts. After some initial successful runs, Figure 1 shows a run that ended in near disaster (the north shore of the river runs approximately East-West about 5 meters north of the origin). While commanding the vehicle to turn, we lost communications with the vehicle as it was headed towards us and the stone wall on the near shore. According to the dead-reckoned position, the vehicle ran straight into the wall at full speed. Fortunately, we were able to get a partial command through to the vehicle (error checking of commands received by the vehicle had not been implemented at this time), which commanded the vehicle to zero depth. Once the vehicle was at the surface, propeller efficiency dropped and the vehicle lost speed and thus ended up about 15 meters from the wall. The vehicle currently has no water velocity sensor so the dead-reckoning track, which assumes a constant vehicle velocity, shows the vehicle going through the wall.

This experience taught us that any command sent to an autonomous vehicle should always be assumed to be the last command it will ever receive, and a routine command right now may be a disastrous command a short time later. In response, we implemented a circling routine. A fixed amount of time (60 seconds in these runs) after receiving a new command, the vehicle loiters in a tight circle to await a new command. In the future, the vehicle could default to some other preprogrammed behavior. In the presence of navigation equipment capable of pinpointing the vehicle's position, there should be a preprogrammed boundary which the vehicle is not to cross.

With the circling routine in place, we operated the vehicle successfully from a remote location. One person monitored the vehicle from the dock, while another commanded the vehicle over the Internet from the Sea Grant laboratory several blocks from the river. Figure 2 shows one such run and the circling behavior. Since all data and commands were multicast, both persons could observe the vehicle's position and issue commands, as could anyone else at MIT had they had the software. By simply changing one software variable, the vehicle data could have been sent throughout the world.

B. Buzzard's Bay Experiments

During the Buzzard's Bay experiment in March-April 1996, we tested telemetry and control of the vehicle in salt water at an actual experiment site. Operations were based from the R/V Diane G, with a Unix workstation acting as a node and shore station as with the Charles River experiments. The acoustic modems were still not integrated yet so we continued to use the radio modems. After an initial trial ended with no telemetry whatsoever, we ordered a waterproof cable to replace our handmade cable. When this failed to correct the problem, we tried placing the float mounted antenna inside the watertight box, as it was the only remaining hardware still exposed to water. We received strong telemetry afterwards, and even the dropout problem we experienced on the Charles River disappeared.

Improvements to the software included full cyclic redundancy check (CRC) error checking, and the ability to request and modify any variable in the vehicle computer's data stack. Although the R/V Diane G had no connection to the Internet, the software was still configured to multicast, and had there been an Internet connection or another computer listening on the ship's local area network, it would have been able to observe the experiment

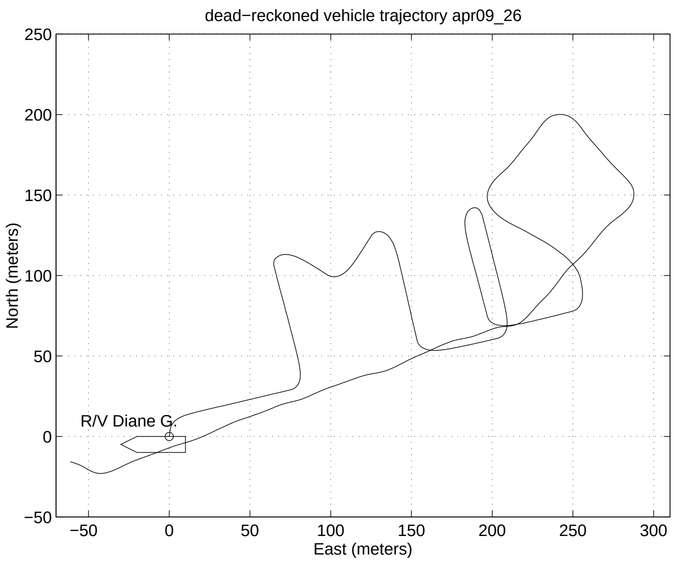

Figure 3. Dead-reckoned vehicle trajectory for modem controlled Buzzard's Bay experiment.

with us. Figure 3 shows the vehicle steered from the R/V Diane G to spell most of the initials of the sponsoring institute. An operator error and rapidly approaching Nor'easter forced us to abort before completing the initials of MIT.

VI. CONCLUSION

The field of acoustic communications is steadily improving data transmission rates. Throughput in the near future for AOSN should be expected to be up to about 10 kb/s.

Transmitting data over the Internet currently involves tradeoffs between the cross-platform compatibility of the WWW and Java, and the network bandwidth efficiency of multicasting over the MBone. The control interface should run independently of the data display so that poor performance of the display does not interfere with control of the vehicle.

On the topic of remotely controlling AUVs, several issues have already become clear. Any command sent to the vehicle should be assumed to be the last command it will ever receive. The on-board vehicle intelligence must be capable of preserving the vehicle and recovering from any situation that command may put the vehicle in. In particular, the operational area of the vehicle should be predefined, and the vehicle programmed not to leave the area. Some type of encryption or security feature will be required to prevent unauthorized persons on the Internet from seizing control of a vehicle.

VII. ACKNOWLEDGMENTS

The authors would like to thank all the members of the MIT Sea Grant Underwater Vehicles Laboratory for their assistance, especially Mr. Robert J. Grieve and Mr. Joseph A. Ziehler for their invaluable help during the Buzzard's Bay and Haro Strait experiments.

Support for this work was provided by MIT Sea Grant under grant numbers NA90AA-D-SG-424 and NA46-RG-0434 and the Office of Naval Research under grant numbers N00014-91-J-1014 and N00014-92-J-1287.

REFERENCES

[1] A. B. Baggeroer. Acoustic telemetry — an overview. *IEEE J. Ocean Engineering*, OE-9(4):229–235, October 1984.

[2] J. G. Bellingham and T. R. Consi. State configured layered control. In *Proceedings of the IARP 1st Workshop on Mobile Robots for Subsea Environments*, pages 75–80, Monterey, CA, USA, October 1990.

[3] J. G. Bellingham, T. R. Consi, R. Beaton, and W. Hall. Keeping layered control simple. In *Proceedings AUV '90*, 1990.

[4] J. G. Bellingham, C. A. Goudey, T. R. Consi, J. W. Bales, D. K. Atwood, J. J. Leonard, and C. Chryssostomidis. A second generation survey AUV. In *IEEE Conference on Autonomous Underwater Vehicles*, Cambridge, MA, 1994.

[5] J. G. Bellingham, C. A. Goudey, T. R. Consi, and C. Chryssostomidis. A small, long-range autonomous underwater vehicle for deep ocean exploration. In *Proc. International Society of Off-shore and Polar Engineers*, 1992.

[6] J. G. Bellingham and D. Humphrey. Using layered control for supervisory control of underwater vehicles. In *Proceedings ROV '90*, 1990.

[7] J. G. Bellingham and J. J. Leonard. Task configuration with layered control. In *Proceedings of the IARP 2nd Workshop on Mobile Robots for Subsea Environments*, pages 193–302, Monterey, CA, USA, May 1994.

[8] D. P. Brutzman. *A Virtual World for an Autonomous Underwater Vehicle*. PhD thesis, Naval Postgraduate School, Monterey, CA, December 1994.

[9] D. P. Brutzman. Internet Protocol over seawater (IP/SW): Towards interoperable underwater networks. In *Proc. Int. Symp. on Unmanned Untethered Submersible Technology*, pages 444–457, 1995.

[10] J. Catipovic. Performance limitations in underwater acoustic telemetry. *IEEE J. Ocean Engineering*, 15(3):205–216, July 1990.

[11] T. B. Curtin, J. G. Bellingham, J. Catipovic, and D. Webb. Autonomous ocean sampling networks. *Oceanography*, 6(3):86–94, 1993.

[12] J. J. Leonard, S. T. Tuohy, J. G. Bellingham, B. A. Moran, J. H. Kim, H. Schmidt, N. M. Patrikalakis, and C. Chryssostomidis. Virtual environments for AUV development and ocean exploration. In *Proc. Int. Symp. on Unmanned Untethered Submersible Technology*, pages 436–443, September 1995.

[13] Richard W. Stevens. *Unix Network Programming.* Prentice Hall, 1990.