DISTRIBUTED EMBEDDED MODULAR ARCHITECTURE (DEMA) FOR AUTONOMOUS UNDERWATER VEHICLE (AUV) CONTROL SYSTEMS

Steve Taylor
Cathexis Innovations Inc.
St. John's, Newfoundland (www.engr.mun.ca)
steve@cathexis.ca

Mark Simms
Cathexis Innovations Inc.
St. John's, Newfoundland (www.engr.mun.ca)
mark.simms@cathexis.ca

Siu O'Young
Center for Instrumentation, Control and Automation
St. John's, Newfoundland (inca.engr.mun.ca)
oyoung@engr.mun.ca

Neil Bose
Canada Research Chair in Offshore and Underwater Vehicles Design
St. John's, Newfoundland (www.engr.mun.ca)
nbose@engr.mun.ca

Abstract

Developing a robust control system for harsh environments presents numerous challenges. Reliability, modularity and resilience to operational conditions are key features of such a control system. For an Autonomous Underwater Vehicle (AUV) size, power efficiency, redundancy and cost become vital attributes as well. A novel architecture called Distributed Embedded Modular Architecture (DEMA) was developed. DEMA successfully addresses these requirements through physically and functionally modularizing the control system. The result is a hierarchical and inherently redundant controller. Implementation of a DEMA system realized the goals of reduced power consumption, redundancy, compactness, modularity and increased cost effectiveness. Ongoing research and development will capitalize on the distributed intelligence of DEMA and further quantify the impact on reliability and robustness.

This paper deals with the development of a distributed hardware architecture to address issues encountered with the previous PC-based control system in an AUV. The focus of this paper is on the design and implementation of this system, and a discussion of the preliminary results from system testing. To this end, the paper includes: a background on AUVs, a discussion of the design philosophy, an overview of the system architecture and a discussion of its performance as well as future developments.

Background

Autonomous Underwater Vehicles, or AUVs, are a technology that is quickly maturing and finding more use in oil and gas operations. As offshore exploration moves further and further into deeper waters, the limitations of remotely operated vehicle (ROV) technology create a need for reliable and versatile AUV platforms [1]. More precisely, the depth to which an ROV can dive is limited by the length of the tether, which provides power and control connections. This cable length is restricted by the tensile strength of the cable itself and the signal attenuation that results due to extreme cable length. The self-contained and un-tethered nature of an AUV eliminates these particular restrictions.

AUV Requirements

The development of an AUV has inherent challenges, imposed by the environment in which the vehicle is operating as well as the nature of the missions that the vessel is intended to carry out. Typical AUV missions occur at significant ocean depth, with power being supplied from the onboard power system and navigation under control of the preprogrammed mission computer. This necessitates the following system capabilities:

Controller:

- Low power consumption
- Cost-effective
- Reliability
- Redundancy
- Small size

Vessel:

- Positional awareness
- Real world interface (sensors)
- Trajectory control
- Communication
- Navigation
- Data acquisition

Historical Systems

Typically, AUVs have been developed around a PC-based system. A PC would be interfaced to a motion control unit to allow fin control, and addition of a data acquisition card would allow for sensor feedback. These peripherals were then tied together via the PC to provide the framework upon which to develop an autonomous controller. This approach has some significant drawbacks; most prominently among these are the roadblocks that this system introduces to future development. Typical roadblocks are power consumption inefficiencies, lack of scalability and size restrictions.

Vessels are typically dimensionally large, needing a relatively large support crew, which is in contradiction to the primary goal of AUVs - operational independence.
Unfortunately, because reliability is such an issue, directly related to the high cost associated with vessel loss, organizations get trapped in a cycle of "reliable-obsolescence"; wherein a technology remains in use because it is known to work, motivated by the cost that would be incurred if a new system were to fail. As well, there may be a tendency for groups to "stick with what they know", further leading to developmental stagnation and failure to adopt more powerful technologies

In the particular case of the C-SCOUT, the PC was too delicate for the harshness of the environment. The size limitations required the PC be stripped down and mounted on a chassis. This wear and tear eventually took it's toll on the PC and it ceased to function. As well, the data acquisition card was large and not intended for such applications. It too lacked the robustness for the application and began to malfunction. As shown in Figure 1, the PC-based controller is developed around a single point of failure, meaning if any one of the components in the controller fails, the complete system becomes immobilized. If this were to result during a mission, the vessel could be lost. Introducing redundant systems to account for this is not practical due to the cost, power and size limitations that would be introduced.

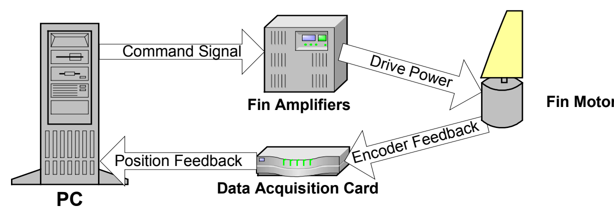

Figure 1: C-SCOUT PC-based system

These limitations ultimately acted as the catalyst for development of the distributed control system by the authors.

Distributed Embedded Modular Architecture - DEMA

The authors took a novel approach in developing the control system for the C-SCOUT AUV. Rather than base the control architecture on prior systems, a top-down approach was taken to break the system down into sub-components. In addition, on-going research into emerging technologies by the authors presented them with an enhanced awareness of new technology that was ideally suited to the project.

System Requirements

The system requirements for the vehicle were a subset of the specifications required for complete autonomous operation. The immediate objective was to develop a reliable test platform, which would serve as a proof-of-concept for the DEMA controller as well as a foundation upon which further research and development could be built. Based on this the following system requirements were distilled: Low power consumption, reliable, compact, and cost effective.

Design Philosophy

Further to clearly defining the system specifications, the authors developed a system architecture in the context of the following design principles:

- Modularity
- Inherent redundancy
- Scalability
- Seamless integration with traditional systems and technology
- Robustness
- Ease of testing, simulation and debugging

While this presented a challenge, the top-down approach provided insight that allowed for a successful solution to the problem. By dividing all of the vessels functions into sub modules, it became evident that most of the required functionality required little individual processing power. The tasks that were previously completed by the computer were extreme overkill and a waste of resources. The authors realized these tasks could be accomplished by dedicated "smart" controllers. This gave rise to the distributed architecture notion, allowing for each aspect of the system to be developed independently.

System Architecture

From this concept of a distributed architecture arose a complete solution that was in accordance with all the aspects of the design philosophy. Figure 2 is a physical block diagram illustrating the system level implementation of a DEMA system [2].

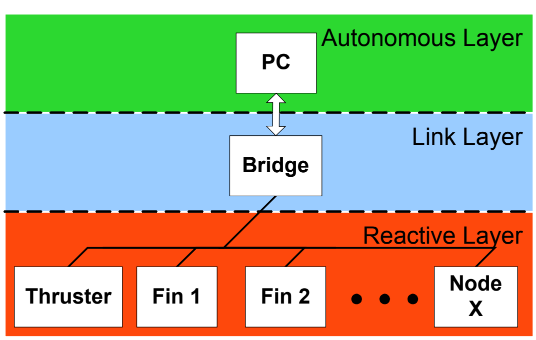

Figure 2: High level DEMA architecture

The top layer, or Autonomous layer, is responsible for performing all of the high-level "intelligent" tasks, such as image recognition, and other computationally intensive tasks that require a lot of computing power. The Autonomous layer connects to the Link layer via a standard interface such as Ethernet, USB or, in the instance of the C-SCOUT, RS-232. This permits development of complex algorithms in a common and powerful programming environment, and communication over a standard interface.

The Link layer, containing the bridge is responsible for interpreting the commands received from the Autonomous layer. Predefined data interfaces facilitate this process, as both layers know the format of the command sequence. Upon interpreting the Autonomous layer data, the Bridge issues the corresponding command set points to the Reactive layer via the field-bus backbone.

In the Reactive layer, each node has a unique address allowing direct bi-directional communication between the Bridge and a specific node. Again, there are standard data interfaces between the Link and Reactive layers, abstracting the underlying complexity.

Essentially, systems running in the link layer do not need to know all of the details of how to communicate with the various modules in the reactive layer. This reduces the complexity involved in communicating between the link and reactive layers.

From a controls perspective, the link between the Autonomous layer and the Reactive layer is open loop. The Reactive layer receives a series of commands or set points that it responds to. It is within each node on the Reactive layer however, that close-loop control is implemented. The Autonomous layer has the capability to obtain feedback from each node, however this is primarily used as a system management tool, not as a control mechanism.

Using the autonomous layer to directly control devices in the reactive layer would result in the same architecture that was deemed unsuitable in the original control system (see Historical Systems). That is, if the autonomous layer malfunctioned, then the entire control system ceased to function.

Implementation

To achieve the performance objectives, the system had to incorporate four fin controllers for maneuvering, a thruster for main propulsion, and a communication link to allow wireless remote control. For the initial version of the hardware, a human operator mimicked the autonomous controller of the vehicle. This reduced development time and allowed for early integration and testing of the DEMA components. It is important to note that DEMA does not encapsulate the autonomous capability. While the architecture lends itself well to intelligent robotic systems, DEMA delivers robust reliable hardware performance regardless of the high-level software that controls it. In essence, DEMA abstracts the complexity of the lower-level link and reactive layers to provide a standard data interface that can be used by virtually any controller.

The concept of abstraction between layers is one of the foundations of the DEMA concept. Essentially, each layer only needs to know a minimum amount of well-define information (i.e. the interface) to communicate with other layers. This allows modules in each layer to change without requiring updating all of the other modules. This is analogous to the phone system. The phone company can install new types of equipment in their central office without requiring everyone to use new phones as well.

The layered architecture is clearly illustrated in Figure 3. As indicated, for the purpose of initial testing, the high-level control was performed by a human operator. A laptop was connected to the bridge via a wireless link, which in turn interpreted the input and relayed all of the commands to the corresponding nodes.

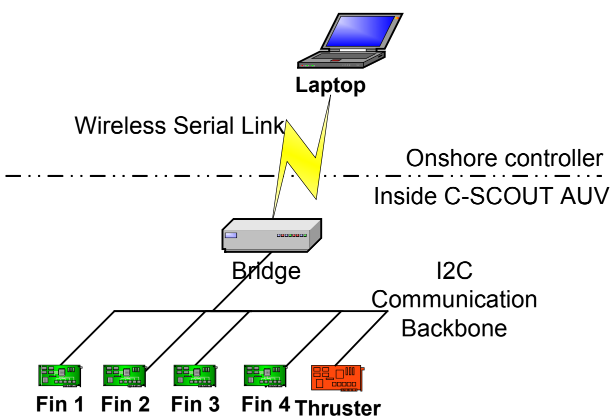

Figure 3: System layout of the initial C-SCOUT DEMA controller

For this instance, a serial RS-232 communication link was used between the Autonomous and Link layers. For the Reactive layer backbone, an I2C bus was implemented. Initially a field bus such as Controller Area Network (CAN) was envisioned, however I2C provided the bandwidth, reliability and ease of setup required for ongoing development. The Bridge and Fin actuators were custom built by the authors.

System Performance

The DEMA architecture has numerous advantages that were highlighted during implementation and testing of the initial platform. From a development perspective, the ability to compartmentalize the function of a subsystem accelerated development and greatly enhanced the debugging and troubleshooting process. As well, the modular nature of the architecture allows the system to be scaled very easily. This is further enhanced by the standardized data and physical interfaces, which impart a “plug and play” facet to the system.

Significant benefits were achieved in the specific areas of power consumption, size and cost. Claims as to the reliability of the system cannot be made, as testing is only preliminary. Further research is ongoing to further analyze system performance and relate this to mission performance.

Table 1 shows the benefits realized from implementing a DEMA system as compared with the performance of the initial PC-based controller.

Criteria	PC-based	DEMA	Ratio (DEMA/PC)
Power Consumption	200 W	21.25 W	10.6 %
Cost	$8000	$1600	20%
Size	1 p.u.	0.25 p.u.	25%

Table 1: Comparison of DEMA Controller Performance

Conclusion

The DEMA controller performed very well, and in fact exceeded the expectations of the authors. The initial lab test results were confirmed by the fully implemented controller, which realized significant reduction in power consumption, size and cost.

Some initial testing has been performed with respect to introducing enhanced reactive intelligence into the Reactive layer nodes. The results were very promising and the controller clearly demonstrated the ability of the DEMA system to recover from error states and simulated hardware malfunctions.

For example, during operations the autonomous layer was deliberately disabled. The reactive layer continued to function, and entered a basic recovery mode. In the previous system disabling the “autonomous layer” (the PC) would have caused total system failure.

Much work remains however to fully develop the technology. Further formalization of the DEMA node interface is needed, as well as incorporation of a higher bandwidth data backbone. This will have a number of positive impacts. First, it will allow any individual to build a DEMA compliant node and have it recognized by the system. This will further enhance the scalability and flexibility of the architecture. It will also encourage acceptance of DEMA as a standard, since researchers can build DEMA compliant nodes of their own for specific purposes, while utilizing nodes developed by other groups or companies.

As well, the distributed nature of the system lends itself to hybrid simulation and testing. In essence, a new DEMA node can be modeled on a PC and plugged into the Reactive layer to mimic its incorporation into the real system. Thus the impact of a new node can be modeled, simulated and analyzed before the hardware is built. To expand on this, any number of DEMA nodes can be "virtualized" allowing for rigorous testing and simulation.

Further development of the C-SCOUT controller requires the addition of hardware to the current baseline version of the AUV. Specifically, an inertial navigation unit, more fins, and an acoustic communication link. This work is ongoing, the next major objective being to achieve autonomous controller integration.

Acknowledgements:

The authors would like to thank the National Sciences and Engineering Research Council of Canada (NSERC) for financial support from their Strategic Projects Program.

References:

[1] Randall, R.E., Bo, G., Albaugh, E.K. "In select applications, auvs work faster, cheaper than tethered vehicles", Tech Rep, Texas A&M University and Mustang Engineering, January 2002.
[2] "Cathexis Innovations Inc. Website", http://www.cathexis.ca/