A Multi-Agent Framework for Autonomous Underwater Vehicles for Mine Countermeasures with MOOS-IvP

Liam Paull[1], Sajad Saeedi[1], Mae Seto[2], Howard Li[1]

[1] University of New Brunswick
[2] DRDC Mine and Harbour Defense Group

Abstract. In this research, a framework for building Autonomous Underwater Vehicle control algorithms that is based on the MOOS-IvP middleware is presented. The Sidescan Sonar Sensor (SSS) is commonly used to generate sonar images in which mine-like objects can be identified. Here, a base station community is implemented that maintains a map of coverage confidence of the SSS, and provides the user with 2D and 3D simulations and the ability to implement advanced control schemes. The development can happen in three stages: 1) A minimalist configuration where only the necessary applications are used to develop and test outer loop control, 2) A configuration that includes simulated hardware and 3) The configuration with actual hardware, which should extend smoothly and easily from stage 2. The benefits are ease of use, faster development, and reduced hardware testing and cost.

Keywords: Multi-Agent Systems, Autonomous Underwater Vehicles, MOOS-IvP.

1 Introduction

Robotics systems and missions are becoming more complex. As a result, it is important for agents and sub-systems to work in parallel and share information to achieve difficult tasks more efficiently. One such task is underwater mine countermeasures (MCM). The US Navy has referred underwater mine removal as the most problematic mission facing unmanned undersea vehicles and the Navy at large [9]. Using multiple vehicles increases efficiency and quality of results through data redundancy and increased robustness, and also allows for missions that would be infeasible with only one vehicle through the use of heterogeneous vehicle configurations [17].

The Mine and Harbor Defense group at DRDC Atlantic conducts research and development on concepts which increase the autonomous decision-making capabilities of unmanned systems. One of these techniques is to use collaborative systems for countering undersea threats. The use and coordination of multiple vehicles tends to increase the complexity and the number of operators required. The use of a Multi-Agent System (MAS) can mitigate this to an extent. In particular, DRDC Atlantic and the Canadian Navy is experimenting with the concepts in the use of multiple AUVs for underwater exploring/mapping and mine-hunting surveys. Such AUVs have on-board Sidescan Sonar Sensors (SSS) that generate sonar imagery of the seabed.

MASs on the ground or in the air have been shown to have remarkable capabilities (for example [1, 14]). However, these architectures either do not apply or require significant modification for use in an undersea environment because of the extremely limited communication capabilities.

In [17], a multiagent architecture is developed for AUVs that uses a blackboard-based broadcast communication system called DELPHIS. Here, the system has three main components: the mission controller, the AUV database, and the mission model. The potentially limiting aspect of this framework is that distinct plans are generated for each AUV in the 'collective' and, as such, it will be difficult for them to achieve global goals efficiently.

In [6], a framework is developed for cooperative navigation amongst AUVs. In particular, optical sensors are used for docking. The description of hardware is useful, however, there is a very limited range of missions that are achievable in this case.

In [15], an auction-based system is developed for multirobot naval MCM called DEMiR-CF. As is common in many multirobot undersea MCM schemes, one agent is responsible for performing a fast area coverage, and one agent is responsible for investigating further each mine-like object (MLO) detected. The 'cover' and 'identify' tasks are optimally allocated to available resources.

MOOS-IvP is a middleware that was developed specifically for underwater robots by Paul Newman at Oxford University, and a team led by Michael Benjamin at Massachusetts Institute of Technology [7]. Eickstedt, et al. have used MOOS-IvP to support sensor driven missions where the robotics platform must

adapt its mission plan based on real-time measurements[2]. The platform is also capable of joint control with other sensing agents. Jiang et al. have developed an auction-based multiple AUV system with MOOS-IvP through the development of bidder and auctioneer applications. However, this auction system is heavily dependent on reliable communication between the auctioneer and the bidder [4].

In general, most previous underwater MAS suffer from overly simplistic and inefficient mission specifications, heavy reliance on reliable communication of unrealistic amounts of data, and system configurations that are not intuitive for users who are not expertly trained in the system.

In this research, a multiple AUV framework is developed with MOOS-IvP that is extensible, reliable, scalable, and easy to use. A base station community is implemented that will allow researchers to: 1) Observe the status of their mission through the developed 3D visualizer as well as the MOOS application *pMarineViewer*, 2) Monitor the coverage confidence over the environment, and 3) Implement long term path planning strategies more easily to achieve missions more efficiently. Only the pose of the vehicles and the waypoints generated at the base station need to be transmitted. The inner loop control, sensing, actuation, and navigation is performed on-board the AUVs, which are each implemented as separate communities.

This framework allows for rapid development of more complex missions. In addition, the need for expert knowledge of hardware programming or middleware is removed as all necessary mission level variables can be specified through the mission files.

The remainder of the paper will be structured as follows: Secion 2 will provide some background on AUV dynamics, the SSS, and MOOS-IvP, Sec. 3 will describe the MOOS-IvP multi-agent framework, and Secion 4 will discuss some conclusions and future work.

2 Background of Research

Unmanned underwater systems have received less attention than their air or ground counterparts. This section will provide some background on AUV dynamics and kinematics, the SSS, and the MOOS-IvP middleware.

2.1 AUV Simulation

The analysis of the motion of an AUV body requires six degrees of freedom since six coordinates are required to define the pose: $\{x, y, z, \phi, \theta, \psi\}$ where x, y, and z are used to describe position and ϕ, θ, and ψ describe the orientation. The quantities and parameters are defined according to the DTNSRDC Revised Standard Submarine Equations of Motion [3].

It is convenient to define two coordinate frames: the body frame is centered at the vehicle's center of geometry and moves with the vehicle, and the inertial or fixed frame which is assumed to coincide with the earth. The linear and rotational velocities corresponding to the six degrees of freedom are $\{u, v, w, p, q, r\}$ and are defined with respect to the body frame.

Kinematics: The kinematics is described with respect to these two frames. Consider:

$$\begin{aligned} \eta &= [\eta_1^T, \eta_2^T]^T, \\ \eta_1 &= [x, y, z]^T, \\ \eta_2 &= [\phi, \theta, \psi]^T, \end{aligned} \tag{1}$$

are defined with respect to the earth fixed frame, and:

$$\begin{aligned} \nu &= [\nu_1^T, \nu_2^T]^T, \\ \nu_1 &= [u, v, w]^T, \\ \nu_2 &= [p, q, r]^T, \end{aligned} \tag{2}$$

are velocities that are defined with respect to body frame.

The vehicle's position is updated by the relation:

$$\dot{\eta_1} = J_1(\eta_2)\nu_1, \tag{3}$$

where,

$$J_1(\eta_2) = \begin{bmatrix} cos(\phi)cos(\theta) & -sin(\psi)cos(\phi) + cos(\psi)sin(\theta)sin(\phi) & sin(\psi)sin(\phi) + cos(\psi)cos(\phi)sin(\theta) \\ sin(\phi)cos(\theta) & cos(\psi)cos(\phi) + sin(\psi)sin(\theta)sin(\phi) & -cos(\psi)sin(\phi) + sin(\psi)sin(\phi)cos(\theta) \\ -sin(\theta) & cos(\theta)sin(\phi) & cos(\theta)cos(\phi) \end{bmatrix} \tag{4}$$

The angles are updated using the equation:

$$\dot{\eta_2} = J_2(\eta_2)\nu_2 \tag{5}$$

where,

$$J_2(\eta_2) = \begin{bmatrix} 1 & sin(\phi)tan(\theta) & cos(\phi)tan(\theta) \\ 0 & cos(\phi) & -sin(\phi) \\ 0 & sin(\phi)/cos(\theta) & cos(\phi)/cos(\theta) \end{bmatrix} \tag{6}$$

Dynamics: The rigid-body dynamics assuming that the origin of the body frame coincides with the principle axes of inertia are given in (8) [18].

$$\begin{aligned} X &= m[\dot{u} - vr + wq - x_G(q^2 + r^2) + y_G(pq - \dot{r}) + z_G(pr + \dot{q})] \\ Y &= m[\dot{v} - wp + ur - y_G(r^2 + p^2) + z_G(qr - \dot{p}) + x_G(qp + \dot{r})] \\ Z &= m[\dot{w} - uq + vp - z_G(p^2 + q^2) + x_G(rp - \dot{q}) + y_G(rq + \dot{p})] \\ K &= I_x\dot{p} + (I_z - I_y)qr + m[y_G(\dot{w} - uq + vp) - z_G(\dot{v} - wp + ur)] \\ M &= I_y\dot{q} + (I_x - I_z)rp + m[z_G(\dot{u} - vr + wq) - x_G(\dot{w} + uq + vp)] \\ N &= I_z\dot{r} + (I_y - I_x)pq + m[x_G(\dot{v} - wp + ur) - y_G(\dot{u} - vr + wq)] \end{aligned} \tag{7, 8}$$

where $\{X, Y, Z\}$ are external forces due to gravity, buoyancy, hydrodynamics (drag), inputs (thrust, elevator, and rudder), as well as control fin configuration [16]. $\{K, M, N\}$ are the moments of the external forces about the origin. $\{I_x, I_y, I_z\}$ are the moments of inertia about the x, y, and z axes respectively, and $\{x_G, y_G, z_G\}$ is the center of gravity of the body.

2.2 Sidescan Sonar Sensor

As discussed in [13], most path planning research has assumed that the robot is equipped with a forward-looking sensor with a two-dimensional footprint. However, in this case, the area covered by the sensor is assumed to be effectively one dimensional perpendicular to the direction of motion. As a result, the sensor only covers an area of the map while the AUV is in motion. In addition, data obtained from the SSS is only considered meaningful while the AUV is in rectilinear motion. Fig. 1 shows an example of the area covered by the sensor for a given path.

One of the most important considerations of the SSS is that the area directly underneath the AUV is not covered. It should be noted that the entire area covered by the sensor will not be covered with the same sensor performance. The sensor performance is a function of many factors including environmental factor such as sea depth or seabed conditions. The sensor performance is defined by the $\mathcal{P}(y)$ curve which defines the confidence that a target is correctly classified as a function of lateral distance from the AUV. Three sample $\mathcal{P}(y)$ curves are shown in Fig. 2, one each for sand, cobble and clay seabed types. The curves are computed using the method described in [8].

2.3 MOOS-IvP

MOOS is a middleware used by the US navy to implement multi-agent systems. The *IvPHelm* application was later built so that higher level behaviors can be defined.

MOOS-IvP utilizes three main philosophical frameworks: the backseat driver, publish-subscribe, and behavior-based control.

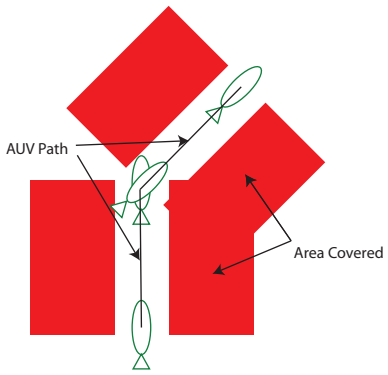

Fig. 1. An example of the AUV path and corresponding area covered by its side-looking sonar.

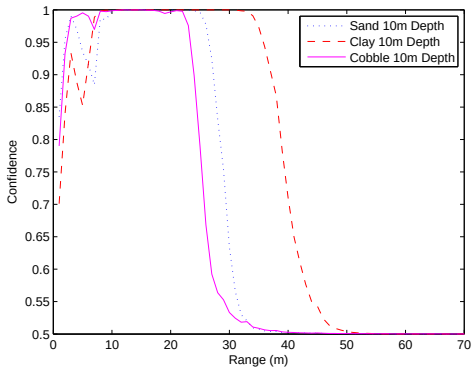

Fig. 2. $\mathcal{P}(y)$ curves for three different seabed conditions

Backseat Driver: In the backseat driver paradigm, the vehicle control and autonomy are decoupled. Essentially, MOOS does not worry about how the vehicle's navigation and control system function, as long as it receives accurate information from the sensors, and the vehicle can act upon the autonomy decisions that are sent back, as shown in Fig. 3 [7].

Publish-Subscribe: The architecture of the MOOS middleware is shown in Fig. 4 [7]. All communication goes through the *MOOSDB*. MOOS applications publish events, which are variable-value pairs and applications also subscribe to different variables. Most of the MOOS applications will normally be sensors, actuators or the visualizers etc., but one is special: the *pHelmIvP* application. The *pHelmIvP* is responsible for reconciling all the desired behaviors.

Behavior-Based Control: Different desired behaviors are defined in the mission behavior file. Behaviors can be selected from amongst predefined behaviors, or can be defined from scratch. On each iteration, the *pHelmIvP* generates an objective function from each behavior, and generates a 'decision' consisting of variable-value pairs that define the action that will optimize the multiple objectives. Fig 5 illustrates the process [7].

In addition, attributes of the behaviors can be assigned dynamically using mode variables and hierarchical mode declarations.

3 MAS Framework with MOOS-IvP

In the proposed MAS framework, there are separate communities for each AUV and one for the base station. Several applications run on each community, some of which are included in a MOOS-IvP distribution and some of which were developed by the authors.

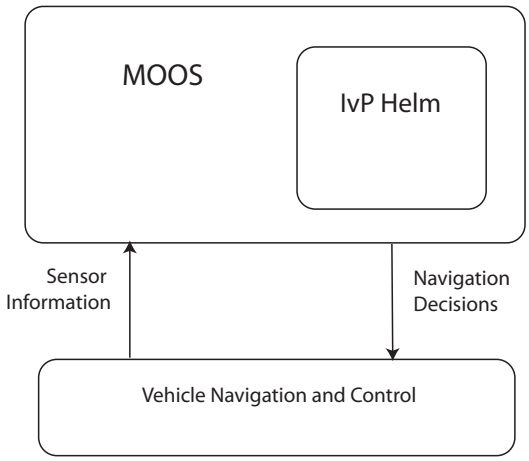

Fig. 3. Backseat driver paradigm

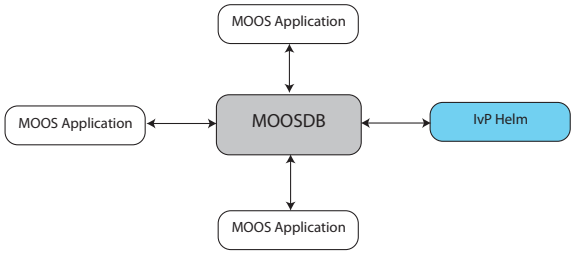

Fig. 4. Publish/subscribe paradigm

3.1 System Overview

In the proposed framework, there are three possible configurations: 1) A minimalist configuration that accelerates the development of advanced control and planning strategies, 2) A simulation configuration substitutes variables for actual sensor and actuator data at the lowest level [12], and 3) The actual hardware implementation.

Configuration 1 - Minimalist: A diagram showing the minimalist configuration with no sensor or actuator processes is shown in Fig. 6.

In this setup, there are three communities, two of which represent actual AUVs and eventually will run on-board the vehicles, and one that is a base station and intended to run on a ship or shore based computer.

In each AUV community there is a *MOOSDB*, a *pLogger*, a *pNodeReporter*, a *pHelmIvP*, a *pMarinePID*, and a *pMOOSBridge*.

- *MOOSDB*: No community can exist without a *MOOSDB*. It is the center of the architecture and controls publishing and subscribing of variables as shown in Fig. 4.
- *pLogger*: Writes MOOS variables to a log file [7].
- *pNodeReporter*: Maps pose into a report that is consumed by the *pMarineViewer* 2D map [7].
- *pHelmIvP*: Responsible for taking a set of behaviors such as waypoint following (*BHV_Waypoint*) or avoid collisions (*BHV_AvoidCollision*) with associated priorities, and resolving them into desired values for the inner loop controller [7].
- *pMarinePID*: A simple inner loop PID controller that tracks the references generated by pHelmIvP by generating the appropriate actuator values (thrust, rudder, and elevator) [7].
- *pMOOSBridge*: Responsible for the communication of variables with other communities. The *pMOOSBridge* can also rename local variables and send them to specific communities. The major advantage of this configuration is that the other applications in the AUV do not need to worry about what community they are in, they can all be identical. The pMOOSBridge appends the 'AUV_N_' prefix onto the variables so that they can be handled appropriately by the base station [10].

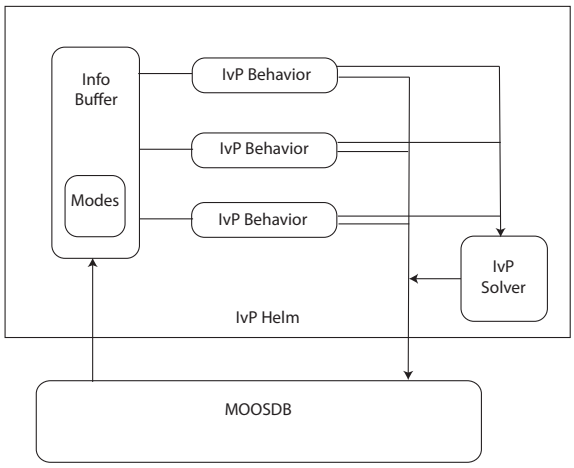

Fig. 5. Behavior-based control

The *BaseStation* community does the long term planning through an interface with MATLAB® called *iMatlab* [11], maintains the confidence map through the ConfidenceMap application, has 2D map visualization through *pMarineViewer* ([7] sec.10), has 3D visualization, and does the kinematics and dynamics simulation through *uMVS* (although this could also happen in the AUV communities), as well as requiring its own *MOOSDB* and *pMOOSBridge* applications. To summarize, the long range planner (Sec. 3.4), *ConfidenceMap* (Sec. 3.2), and *Visualizer3D* (Sec. 3.3) are applications that were developed. The other applications were appropriately configured and their purposes will be discussed here:

- *MOOSDB*: Same as above, handles all published and subscribed data.
- *pMOOSBridge*: Does the inverse of the *pMOOSBridge* applications on the AUVs. Specifically, it maps community specific variables generated on the base station, such as poses or waypoints, and sends them to the appropriate community with generic names.
- *pMarineViewer*: A 2D mapping application that is also capable of displaying multiple vehicles at one time [7]. A screenshot of two AUVs is shown in Fig. 7.
- *iMatlab*: An application that allows a matlab script to enter a MOOS community and therefore publish and subscribe to variables. It is based on using 'mex' to compile some MOOS code that MATLAB® can call directly [11].
- *uMVS*: A multiple vehicle kinematics and dynamics simulator that is also capable of simulating acoustic communications [12].

Configuration 2 - Hardware Simulation: In this next configuration, the hardware sensing and actuation modules are added to the AUV communities as shown in Fig. 8. By setting the *simulator* variable to true in the mission file, then the sensing instruments subscribe to simulation variables (SIM_*) generated by the simulator, and publish actual values. Similarly, the actuator applications subscribe to simulated desired actuator variables and re-publish the same values as actual actuator values. For more details, refer to [12]. The *pNAV* application is responsible for taking the asynchronous sensor readings and producing accurate estimates of vehicle pose using Extended Kalman Filtering.

Configuration 3 - Hardware: The simulation mode is set to false. The *uMVS* block is removed from the base station community, and acoustic communications are implemented. Sensor and actuator applications are configured to communicate with the actual sensors via serial communication. The goal is that the testing in configurations 1 and 2 should allow for easy transition to configuration 3. The communication amongst communities is achieved through acoustics.

3.2 ConfidenceMap

The mission for which this framework was designed is seabed area coverage. As mentioned, SSS performance is crucial to being able to identify MLOs. The SSS performance is dependent on seabed type, depth and other

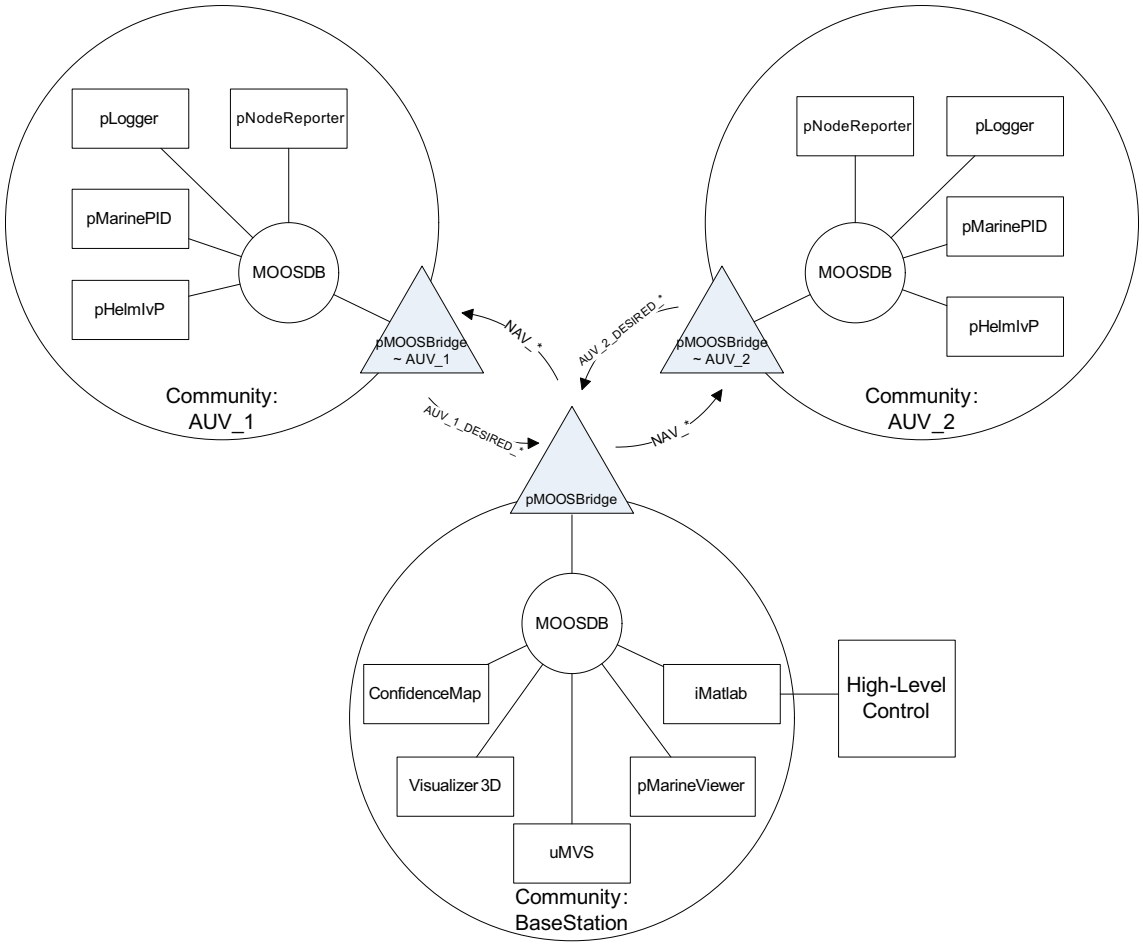

Fig. 6. Configuration 1: 2 AUV communities and one base station community with minimal applications.

factors. The mission is considered as completed when the average coverage confidence over the environment reaches a specified threshold.

The *CoverageMap* application on the base station registers for the pose variables of all AUVs and any environmental factors that they report. A map of the confidence over the entire area of interest is maintained and used by the long range planner. The *CoverageMap* application is responsible for signaling the completion of the mission. A confidence plot that was generated by MATLAB® is shown in Fig. 9.

In the figure, the coordinates of the environment are $((0,0),(0,100),(100,100),(100,40))$. Anything outside of the environment is assigned a confidence of -1. The confidences over the entire environment are initialized to 0.5 and then updated as the AUVs move about. In this case, there is one AUV that began at $(0,0)$ and then traveled parallel to the y axis and made a right turn.

This task is more challenging than may seem due to the intermittency of the variable updates. The application maintains current and previous poses and updates the map accordingly. If the change in yaw is above a threshold, the update is not made because the SSS data is not valid in the case that the AUV is turning. Otherwise, a line is drawn between the previous and current (x,y) locations and is stepped along as perpendicular locations are updated.

3.3 3D Visualizer

A major challenge of AUV development is that it is difficult and costly to observe the vehicles in the environment. To mitigate this problem, an OpenGL based 3D visualizer originally designed to work with the MIRO middleware has been adapted and ported to MOOS-IvP [5]. Seabed data must be provided from previous bathymetric surveys. The 3D visualizer subscribes to the poses of all AUVs and displays them as

Fig. 7. A screenshot of the *pMarineViewer* application with two AUVs shown.

shown in Fig. 10. One AUV is always at the center of the screen, while others will appear if they are in view (in the bottom left of the figure).

3.4 Decentralized Outer Loop Control on the Base Station

Although the long-term planning is occurring at a central location, it need not suffer from the scalability issues inherent in a centralized control paradigm. The planner waits for requests for new waypoints, and then generates them based on the information available at that time. Paths are represented as tracks, where a track is defined by (x, y), an angle θ, and a length l: $t = \{x, y, \theta, l\}$. Tracks are generated on request to minimize some objective function, for example:

$$R(t) = w_B \cdot B(t) - w_J \cdot J(t) - w_D \cdot D(t), \tag{9}$$

where R is the measurement profit, B is the Expected Entropy Reduction (EER) that would result from following track t based on the information that is already known about the environment and the current confidence map, J is the energy required for the AUV to get from its current location to the start of the next track t, and D is the length of track t. w_B, w_J and w_D are the associated weights [13].

Given that the planning is happening on request and without considering the poses of all AUVs the computation time is a linear function of the number of AUVs.

4 Conclusion

A framework for painless development of advanced control algorithms for Autonomous Underwater Vehicles with Sidescan Sonar Sensors has been presented. In the proposed approach, there are three levels of development: one with minimal overhead, one with hardware simulation, and then the final stage with actual hardware and acoustic communication. A base station community is used to simulate the vehicles in 2D and 3D, as well as maintain coverage confidences over the search space and generate paths to achieve the global mission goals. The implementation of this system will allow for reduced development time, impact on hardware, and cost.

Acknowledgment

This research is supported by Natural Sciences and Engineering Research Council of Canada (NSERC) and Canada Foundation for Innovation.

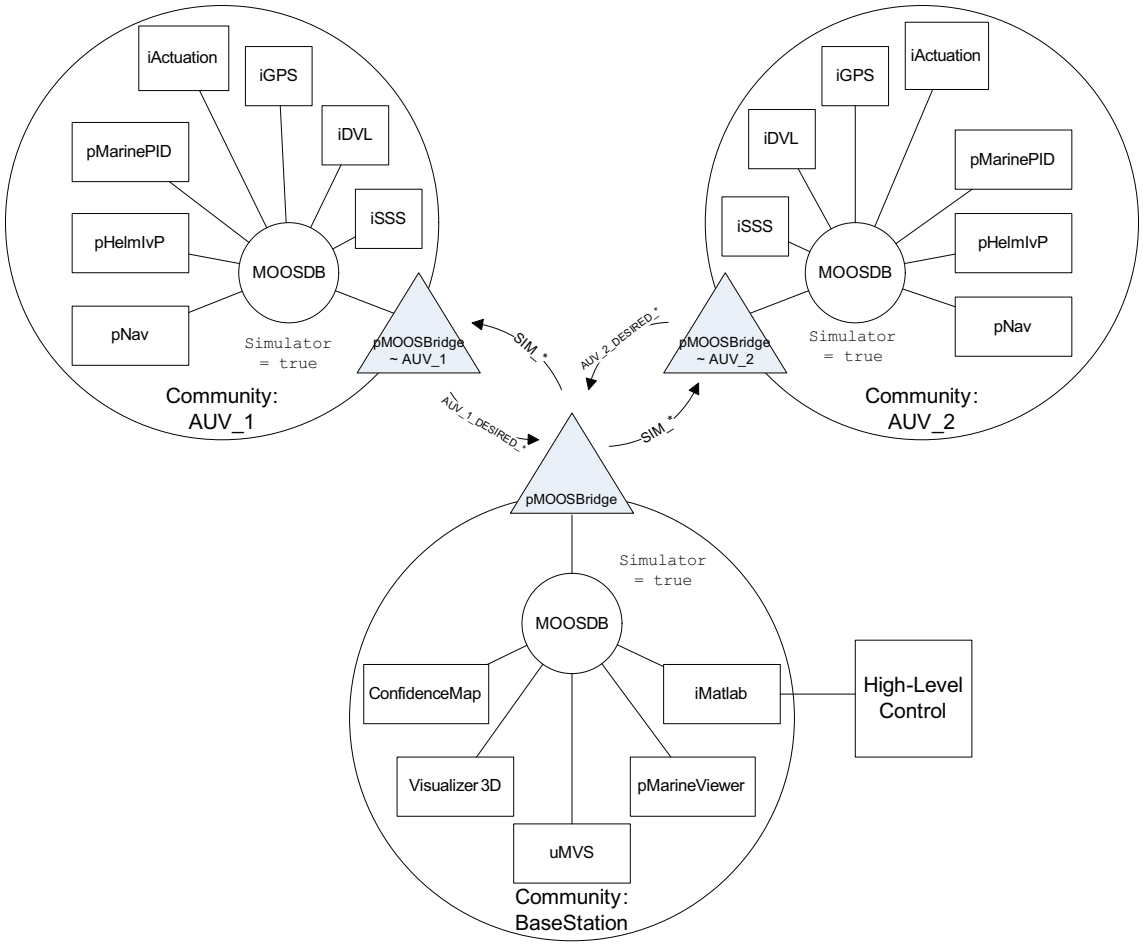

Fig. 8. Configuration 2: Hardware simulation includes all sensor and actuator applications.

References

[1] Broten, G., Mackay, D., Monckton, S., Collier, J.: The robotics experience. IEEE Robotics Automation Magazine 16(1), 46 –54 (March 2009)

[2] Eickstedt, D., Benjamin, M., Curcio, J.: Behavior based adaptive control for autonomous oceanographic sampling. In: Robotics and Automation, 2007 IEEE International Conference on. pp. 4245 –4250 (2007)

[3] Feldman, J.: Dtnsrdc revised standard submarine equations of motion. Tech. rep., David W. Taylor Naval Ship Research and Development Center (June 1979)

[4] Jiang, D., Pang, Y., Qin, Z.: Coordination of multiple auvs based on moos-ivp. In: Control and Automation (ICCA), 2010 8th IEEE International Conference on. pp. 370 –375 (2010)

[5] Li, H., Popa, A., Thibault, C., Trentini, M., Seto, M.: A software framework for multi-agent control of multiple autonomous underwater vehicles for underwater mine counter-measures. In: Autonomous and Intelligent Systems (AIS), 2010 International Conference on. pp. 1 –6 (2010)

[6] Mathew Dunbabin, Peter Corke, L.V., Rus, D.: Experiments with cooperative control of underwater robots. International Journal of Robotics Research 28, 814–833 (2009)

[7] Michael Benjamin, Paul Newman, H.S., Leonard, J.: An overview of moos-ivp and a brief users guide to the ivp helm autonomy software. http://dspace.mit.edu/bitstream/handle/1721.1/45569/MIT-CSAIL-TR-2009-028.pdf (June 2009)

[8] Myers, V., Pinto, M.: Bounding the performance of sidescan sonar automatic target recognition algorithms using information theory. IET Radar Sonar Navig. 1(4), 266–273 (2007)

[9] Navy, U.: The navy unmanned undersea vehicle (uuv) master plan (tech rep. a847115). Tech. rep., U.S. Navy (2004)

[10] Newman, P.: Bridging communities with pmoosbridge. http://www.robots.ox.ac.uk/ pnewman/MOOSDocumentation/ Essentials/Bridging/latex/MOOSBridge.pdf (June 2009)

[11] Newman, P.: Moos meets matlab - imatlab. http://www.robots.ox.ac.uk/ pnewman/MOOSDocumentation/ tools/iMatlab/latex/iMatlab.pdf (March 17 2009)

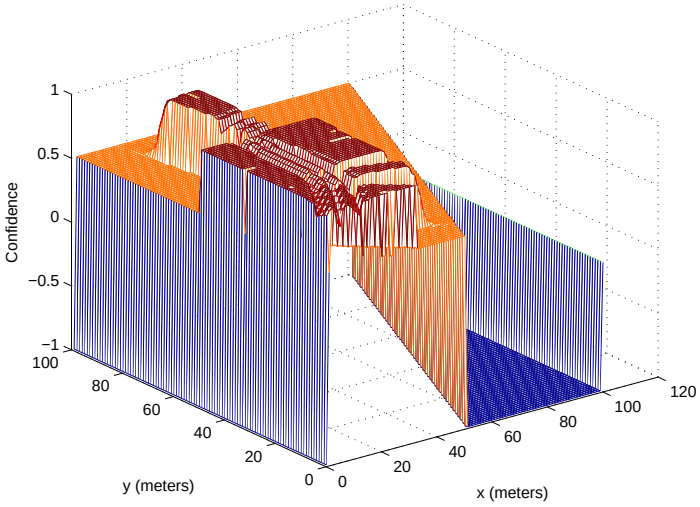

Fig. 9. A map of confidence over a test environment.

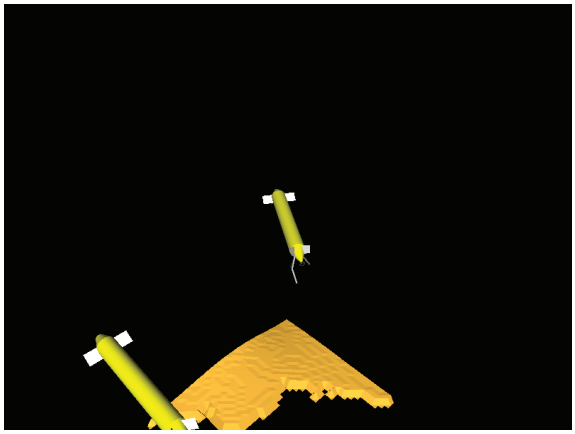

Fig. 10. A screenshot of the developed 3D Visualizer displaying 2 AUVs and bahymetric data.

[12] Newman, P.: Using the marine multivehicle simulator: umvs. http://www.robots.ox.ac.uk/ pnewman/MOOSDocumentation/ tools/Simulation/Marine/latex/MarineMultiVehicleSimulator.pdf (March 2009)

[13] Paull, L., Saeedi, S., Li, H., Myers, V.: An information gain based adaptive path planning method for an autonomous underwater vehicle using sidescan sonar. In: Automation Science and Engineering (CASE), 2010 IEEE Conference on. pp. 835 –840 (2010)

[14] Pavone, M., Savla, K., Frazzoli, E.: Sharing the load. Robotics Automation Magazine, IEEE 16(2), 52 –61 (June 2009)

[15] Sariel, S., Balch, T., Erdogan, N.: Naval mine countermeasure missions. Robotics Automation Magazine, IEEE 15(1), 45 –52 (2008)

[16] Seto, M., Li, H.: On-board auv autonomy through adaptive fins control. In: Automation Science and Engineering (CASE), 2010 IEEE Conference on. pp. 933 –939 (2010)

[17] Sotzing, C.C., Lane, D.M.: Improving the coordination efficiency of limited-communication multi-autonomus underwater vehicle operations using a multiagent architecture. Journal of Field Robotics 4, 412–429 (2010)

[18] T.I.Fossen: Guidance and Control of Ocean Vehicles. Wiley (1994)