Hierarchical Multi-Agent Command and Control System for Autonomous Underwater Vehicles

Tan Yew Teck
Department of Electrical and Computer Engineering
National University of Singapore
Singapore
william@arl.nus.edu.sg

Mandar Chitre
ARL, Tropical Marine Science Institute and
Department of Electrical and Computer Engineering
National University of Singapore
Singapore
mandar@arl.nus.edu.sg

***Abstract*—Inspired by the command structure of a manned submarine, we have developed a Command and Control (C2) system for autonomous underwater vehicles (AUVs) that allocates mission, navigation and vehicle tasks to individual self-contained agents, each with their own responsibilities and behaviors. These agents are distributed over different levels of control hierarchies where they behave deliberately at the supervisory level and reactively at the vehicle and navigational level. The collective interactions among the pool of agents enables the AUV to achieve its mission objectives autonomously.**

The mission supervisory level adopts a backseat driver paradigm where mission-level decisions are made based on the inputs provided by a pool of backseat driver (BD) agents. Each BD agent is responsible for handling different aspects of a mission and provides input in the form of mission points to achieve specific mission sub-tasks. This approach offers several advantages. Firstly, complex mission objectives can be divided into simpler mission sub-tasks and handled by different BD agents. Secondly, the C2 system's capabilities in coping with new mission scenarios can be easily extended through the introduction of new BD agents that generates the required maneuvering patterns. New mission behaviors may emerge from the cooperation and/or competition among the BD agents. These complex behaviors increase the level of mission autonomy.

The C2 system described above is being used in the STARFISH AUVs and has been used to perform single AUV surveying missions as well as multi-AUV cooperative positioning missions.

I. Introduction

Instead of developing complex, expensive monolithic Autonomous Underwater Vehicles (AUVs) for underwater missions, researchers nowadays are moving their attention towards building simpler, low-cost modular AUVs [1], [2], [3]. Modularity in AUV development at software and hardware level provides benefits to the developers and users [1], [4], [5]. Different modules of an AUV can be built separately by different groups of developers in parallel and they can also be exchanged depending on the functionalities needed for a particular mission task.

Every changeable module has its own software modules that implement different algorithms depending on the module's responsibilities in the overall AUV setup. When put together, they form a complete working AUV. However, this plug-and-play capability can only be achieved if the underlying Command and Control (C2) system is capable of adapting to the various AUV configurations for different missions.

Fig. 1. The STARFISH [1] modular AUV with DVL payload module attached.

The availability of low-cost modular AUVs with different payload configurations has driven researchers' desire for autonomy in a team of vehicles. Multi-vehicle missions offer several advantages over single vehicle missions. Multiple vehicles are capable of simultaneously surveying different points of a mission area, thus providing spatio-temporal sampling that single vehicles simply cannot. This is particularly important in the environmental sensing and monitoring missions where the features of interest dynamic evolve at multiple spatial and temporal scales. Additionally, multiple vehicles offer some degree of tolerance to vehicle failure during a mission.

Cooperative missions using a team of modular AUVs poses a challenge to the designers of the underlying C2 system. Not only does the C2 system have to handle its own mission tasks and onboard sensor data, but has to deal with potentially complex interactions among the peer vehicles in the team. One common approach to ease the problem is to adopt the divide-and-conquer model where high level mission tasks are decoupled from the low level vehicle navigational tasks [6]. The division provides a clear view on the overall control architecture and makes the C2 system development and maintenance more manageable. Typically the vehicle navigational control remains unchanged regardless of the complexity of the overall mission objectives. However, the requirements of the high level mission tasks evolve with the nature of the cooperative missions specified for a team of vehicles. The underlying C2 system must be extensible to handle the new

requirements as they are introduced.

In this paper, we focus on the development of a C2 system for the STARFISH [1] AUVs developed at the ARL, National University of Singapore (NUS). The STARFISH AUV is a low-cost modular AUV developed as a research platform and is capable of being extended with various sensor payload modules for sensing and monitoring missions. The project calls for the deployment a team of AUVs to perform tasks cooperatively to achieve the mission objectives. The ARL currently operates two STARFISH vehicles and is in the process of adding more AUVs to the team. Fig. 1 shows one of the AUV with a Doppler Velocity Log (DVL) payload module attached.

We adopted a multi-agent approach where mission, navigation and vehicle control tasks are allocated to individual software agents that are arranged in a hierarchical order according to their corresponding control responsibilities. Our previous work [7] adopted a similar approach and high level mission tasks were handled by only two agents at the Supervisory level. This has suffered from the "fat" agent problem where one agent is overloaded with various tasks and resulted in reliability and maintainability issues. In this work, we alleviate the issue by adapting the Backseat Driver (BD) paradigm introduced in [8], [9] to handle new mission requirements where the decisions are made through the exchange of mission level information rather than leveraging on user-defined objective functions.

In what follows, the overall C2 system's architecture is presented in section II, the backseat driver design paradigm is presented in section III. Simulation and field experiment results are discussed in section IV and followed by the conclusion and future work in section V.

II. Architectural Overview

The C2 system is based on a hybrid-hierarchical model as shown in Fig. 2. It adopts a deliberative-reactive architecture that consists of a set of interacting agents organized in three different levels within the control hierarchy: Supervisory level, Mission level and Vehicle level. Each of the control levels assumes different C2 responsibilities and defines the responsive requirements for an agent located within it. The Supervisory level is in charge of making high-level mission decisions, monitoring the vehicle status and communicating with the operator/mothership. The agents within this level maintain internal states and deliberate upon the state information for decision making. The Vehicle level is responsible for performing low-level vehicle control and interacts reactively with Sentuator (vehicle sensors and actuators) components to generate the desired maneuvering behaviors. The agents in the Mission level act as the arbitrators among the agents in the Supervisory and Vehicle level by translating the mission goals into collision-free path waypoints for the vehicle to follow.

This C2 system design offers many benefits. The hybrid architecture allows high level mission control to behave deliberatively while decoupling the low level reactive vehicle control. It also defines the real-time requirements of the agents residing within each control level. The breaking down of the C2 tasks into individual agents presents an explicit view of the clearly defined control responsibilities at different levels of the control hierarchy. The architecture provides an important guideline for agent developers and ensures the resultant C2 system's integrity.

Each agent has its private data and implements its own algorithms depending on the assigned responsibilities. All the agents are self-contained and have a uniform software interface to facilitate inter-agent communication via a message-passing mechanism. The vehicle's C2 tasks are achieved through the interaction and cooperation among the involved agents. The agent-based design provides flexibility in terms of software implementation. Rather than modify existing software components, new agents can be built and loaded when necessary.

A. C2 agents

The responsibilities of the C2 agents within the architecture shown in Fig. 2 are briefly described below:

1) **Captain:** Starts, coordinates, oversees and controls the execution of missions. Listens to the safety notifications from the Safety Officer and aborts the mission if any abnormality is observed. Executes Operator's commands delivered by the Signaling Officer and broadcasts mission planning requests to the Agent Services and assigns the mission point generator.
2) **Signaling Officer:** Acts as the AUV's external communication node. Updates the Operator with the latest mission and AUV status periodically. Decodes the mission commands received and passes them to the Captain.
3) **Safety Officer:** Detects any abnormality reported by the Health Monitor and monitors vehicle navigational status (maximum pitch, roll, depth and minimum altitude) and system resources. Ensures that the vehicle navigates only within the geofenced area defined by the Operator.
4) **Backseat Drivers and Scientists:** Generates mission points to achieve mission objectives. Interrupts the Captain with new mission points when needed. The Scientist is a type of the Backseat Driver that controls and interacts with optional payload modules attached to the AUV. More details are discussed in section III.
5) **Executive Officer:** Receives mission tasks in the form of mission points and passes them to the Navigator for waypoint planning. Once waypoints are received, sends them to the Pilot for execution. Notifies the Captain if local obstacle avoidance to reach the next mission point fails.
6) **Navigator:** Plans a collision-free path from one mission point to the next. Re-plans the waypoints if obstacles are detected in the path. Notifies the Executive Officer if a collision free path to the next mission point is not found. Marks the newly detected obstacles in the mission Chart Room. Navigators with different waypoint planning algorithms can be instantiated according to the mission's requirement.

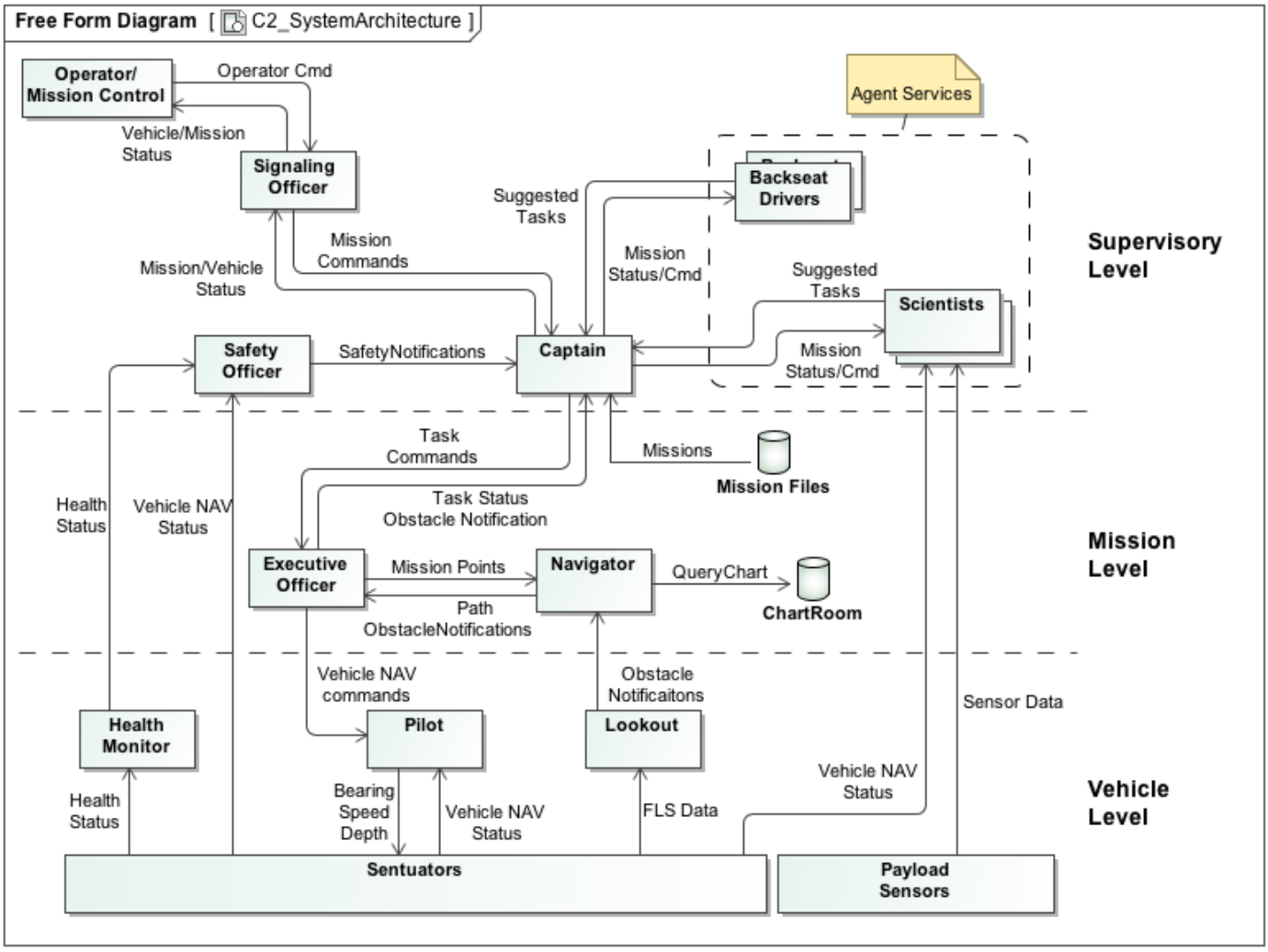

Fig. 2. Overview of the Command and Control System.

7) **Health Monitor:** All the sensor and actuator (Sentuators) drivers in the vehicle implement a health reporting mechanism. The Health Monitor collects the information and analyzes the severity when Sentuators are found unhealthy. It notifies the Safety Officer if the severity is high.
8) **Pilot:** Translates the waypoints into primitive vehicle control (bearing, speed, depth and altitude control setpoints). This is done according to the mode of waypoint execution. Two modes are currently implemented: Waypoint Following and Path Following.
9) **Lookout:** Processes and analyzes the sensor data provided by the Forward Looking Sonar for obstacle detection. Informs the Navigator with the range and bearing of newly detected obstacles.
10) **Mission Files and Chart Room:** Storage of mission files and mission area bathymetry.

III. Backseat Driver Design Paradigm

To allow for different mission behaviors and cater to various payload modules with potentially different mission requirements, the Supervisory level adopts a backseat driver paradigm where mission decisions are made based on the input provided by a pool of BD agents. This pool is termed as Agent Services (AS). Each BD agent in the AS implements different algorithms and monitors various sensor data to generate inputs in the form of mission points, which when accepted for execution, achieve a specific mission task. Depending on the requirements of the mission, the BD agents can be tasked to generate mission point-sets in some pre-planned pattern, or to generate single mission points iteratively adapting to sensor data as the mission progresses.

The Scientist agents are special BD agents that interact exclusively with the payload modules. This enables the payload developers to implement algorithms that make use of the payload sensor data to adaptively generate mission points for sensing and tracking missions. A Scientist agent can be

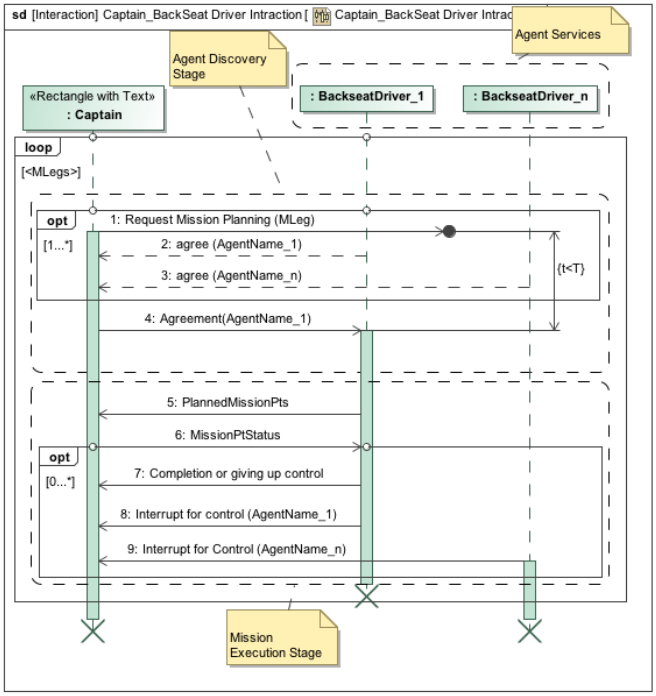

Fig. 3. BackseatDriver Design paradigm.

introduced with each newly developed payload module. Depending on the vehicle's final payload setup, different Scientist agents can be loaded to generate mission points to achieve the overall mission objectives. However, this does not prevent the introduction of payload module like the DVL that provide only navigational data for the onboard positioning system, and do not need a corresponding Scientist agent.

The BD design paradigm relies strongly on the interactions between the Captain agent and the pool of BD agents. Essentially, the interactions happen in two different stages during a mission: Agent Discovery stage and Mission Execution stage. The Agent Discovery stage takes place when the mission starts. A mission consists of at least one or more mission legs (MLegs). Each MLeg encodes the MLeg_type, position and parameters that define speed, waypoint radius, maximum depth and minimum altitude. When the start command is received from the operator, the Captain agent broadcasts a request for mission planning with the corresponding MLeg. BD agents that are programmed to handle the particular MLeg_type will respond.

The Captain agent waits for a time period, T, and collects all the BD agents' responses. If no response is received for the requested MLeg_type, the mission is aborted since there is no BD agent that is capable of translating the mission leg command into mission points. If only one response is received, the corresponding DB agent will be assigned for mission point generation. However, in the case where more than one response is received, the Captain agent identifies one of the BD agents as the mission point generator according to a specific preference. This preference can be determined by simple priority look-up table (adopted in this paper) or more sophisticated approaches like the market-based approach [10] and automated planning approach like T-REX [11]. Once selected, the assigned BD agent is notified with an agreement and contracted as the mission point generator.

During the Mission Execution stage, the assigned BD agent provides the generated mission points to the Captain agent and monitors the mission status. This process is repeated until the completion of the mission leg or the BD agent gives up the control (due to failure in achieving the mission leg's objective). In the later case, the mission is aborted for safety reasons.

Depending on the mission requirements and the BD agents' configuration, the current executing mission point can be aborted and replaced with new mission points by the same BD agent or by another BD agent to pursue tasks of interest with higher priority. This approach allows the C2 system to adapt the mission during execution. A mission that requires the vehicle to survey an area and track a feature of interest, if and when it is detected, benefits from the C2 system's adaptive capability. Detailed interactions between the Captain agent and the BD agents during a sample mission are shown in Fig. 3. The chosen mission scenario presented in section IV showcases the usefulness of the adaptive capability.

IV. Simulation and Field Experiments

The C2 system described above has been implemented using a JAVA agent framework (JAF) that was developed in-house. JAF provides an environment for agent communication, interaction and management. The same C2 system can also be implemented using other popular component-based software frameworks such as ROS [12] or MOOS [13]. In this section, we show the results from various simulated missions as well as field experiments.

A. Simulation: Cooperative Positioning

This simulation demonstrates the C2 system's capability in performing a cooperative positioning mission. We refer the reader to [14] and [15] for detailed algorithms. The key idea behind cooperative positioning is to have an AUV with high-quality positioning information (positioning vehicle) transmit its position information to other AUVs (survey vehicles) within its communication range. Range information from the time-of-flight of the communication packet is derived at each survey AUV and fused with data obtained from the onboard navigational sensors such as compass and DVL to reduce the positioning error during underwater navigation.

The cooperative algorithm was implemented within a BD agent (BD_Coop) in the positioning vehicle while the lawn mowing survey path was generated by another BD agent (BD_Lawnmower) in the survey vehicle. During the mission, the BD_Coop generated the mission points adaptively depending on the supported survey vehicle's current position, such that when range and position information were exchanged between the AUVs, the position errors of the survey vehicle was minimized. Fig. 4 shows the resultant path of the positioning

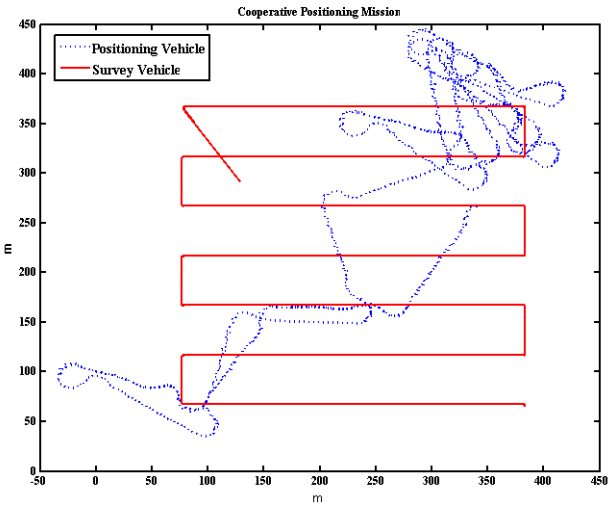

Fig. 4. Simulation results show the resultant path generated by the positioning AUV (blue dotted line) to minimize the position errors of the Survey AUV (red solid line).

vehicle (dotted blue line) to support a survey mission (red solid line).

B. Field Experiment: Surveying Mission

During the field experiment carried out at Tuas, Singapore in August 2012, the C2 system was asked to carry out a surveying mission at the outlet of the power generator's cooling tower. The mission reused the BD_Lawnmower agent mentioned in section IV-A to generate the mission points in a lawn-mowing pattern within an area specified by the operator. The mission points were sent to the Mission level for waypoints planning and then to Vehicle level for execution.

Varying water flow directions were observed in the mission area due to the interaction of the south-east tidal current and

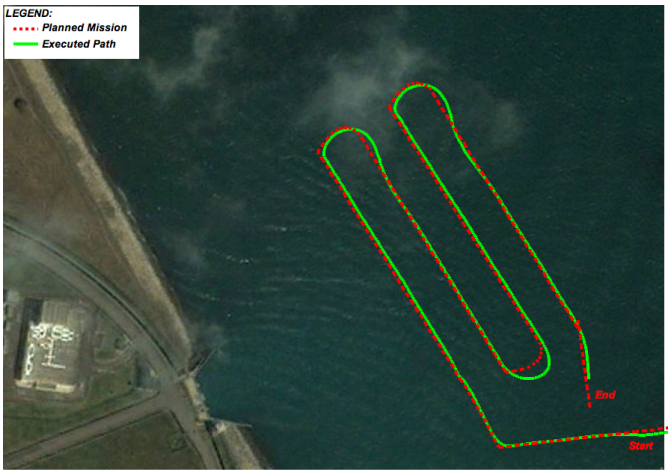

Fig. 5. Planned (red dotted line) and executed (green solid line) survey paths by the STARFISH AUV during the Tuas, Singapore field experiment in August 2012.

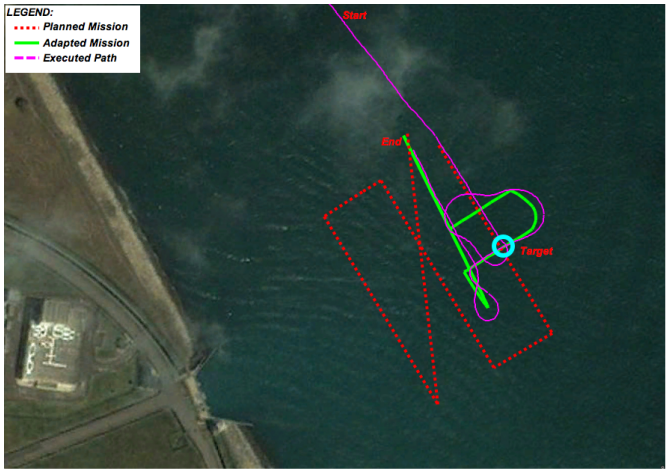

Fig. 6. AUV path during the adaptive mission at Tuas, Singapore field experiment in August 2012. Red dotted line shows the initial planned survey path. When a simulated target is detected, the mission was interrupted and re-planned (green solid line) to re-visit the target before proceeding to the end point.

the outlet's water flow. Although this presents a challenge for the low level vehicle navigational control, the achieved path has little overshot using the path-following algorithm implemented in the Pilot agent. The decoupling of high-level mission point planning and low-level vehicle control in separate agents made the tasks more manageable. The resultant path from the surveying mission is shown in Fig. 5.

C. Field Experiment: Mission Adaptation

Fig. 6 shows another survey mission carried out by the STARFISH AUV in the same area. The mission goal was to simulate a target of interest being detected during the execution of a preplanned lawn-mowing survey mission (red dotted line), and to verify the capability of the C2 system in re-adapting the current mission to re-visit the target with a second fly-by path.

A Scientist agent (BD_SideScanner) was developed and introduced into the AS to simulate continuous monitoring of the sensor data returned by the Side Scan Sonar payload module throughout the surveying mission. Whenever a target of interest is detected, it broadcasts a request to the BD agents within the AS for new mission points that will navigate the AUV to re-visit the target position. Once received, the BD_Lawnmower replies with a set of mission points which results in a second fly-by over the target position. The execution of the current surveying mission is interrupted and replaced with the newly planned mission points. The AUV then navigates through the target for the second time before completing the mission at the end point.

Even though the target detection was simulated, the results of the mission are sufficient to demonstrate the underlying C2 system's capability in handling missions with adaptive behaviors.

V. Conclusion and Future Work

In this paper, we described a hierarchical multi-agent C2 system and its implementation on the STARFISH AUVs in detail. The system architecture mimics the control structure of a submarine, where mission and vehicle tasks are clearly divided and handled by individual agents organized at different levels in a control hierarchy. The multi-agent design allows new mission behaviors and capabilities to be introduced into the C2 system with minimum modification to the overall architecture. Although this architecture was developed with the marine vehicles in mind and implemented using an in-house agent platform, the architecture may also be used in land or aerial vehicles, and implemented using other popular component-based software frameworks. In fact, the integration of this C2 system into Autonomous Surface Vehicles (ASV) such as MIT Scout Kayaks is underway in preparation for an upcoming cooperative positioning and sensing field experiment in Singapore.

Future extensions include the integration of automated planning tools in the Captain agent to increase the level of autonomy in its interaction with the AS, as compared to the simple priority-based look-up table for contracting the BD agents.

Acknowledgment

The authors express their gratitude to members of the STARFISH project team for their support and involvement in field experiments. The authors also like to thank the SMART CENSAM Group for their help during the sea trials at Tuas, Singapore.

References

[1] T. Koay, Y. Tan, Y. Eng, R. Gao, M. Chitre, J. Chew, N. Chandhavarkar, R. Khan, T. Taher, and J. Koh, "Starfish – a small team of autonomous robotic fish," *Indian Journal of Geo-Marine Sciences*, vol. 20, no. 2, pp. 157–167, April 2011.

[2] Iver2 auv. [Online]. Available: http://www.iver-auv.com/

[3] Remus auv. [Online]. Available: http://www.km.kongsberg.com/

[4] M. Chitre, "Dsaav - a distributed software architecture for autonomous vehicles," in *OCEANS 2008*, sept. 2008, pp. 1 –10.

[5] M. Sangekar, M. Chitre, and T. Koay, "Hardware architecture for a modular autonomous underwater vehicle starfish," in *OCEANS 2008*, sept. 2008, pp. 1 –8.

[6] H. Yavuz and A. Bradshaw, "A new conceptual approach to the design of hybrid control architecture for autonomous mobile robots," *Journal of Intelligent and Robotic Systems*, vol. 34, pp. 1–26, 2002.

[7] T. Y. Teck, M. Chitre, and P. Vadakkepat, "Hierarchical agent-based command and control system for autonomous underwater vehicles." Autonomous and Intelligent Systems (AIS), 2010 International Conference on, jun. 2010, pp. 1 –6.

[8] M. Benjamin, H. Schmidt, P. Newman, and J. Leonard, "Nested autonomy for unmanned marine vehicles with moos-ivp," *Journal of Field Robotics*, vol. 27, no. 6, pp. 834–875, Nov 2010.

[9] D. Eickstedt and S. Sideleau, "The backseat control architecture for autonomous robotic vehicles: A case study with the iver2 auv," in *OCEANS 2009, MTS/IEEE Biloxi - Marine Technology for Our Future: Global and Local Challenges*, 26-29 2009, pp. 1 –8.

[10] G. Hollinger and S. Singh, "Multi-robot coordination with periodic connectivity," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, may 2010, pp. 4457 –4462.

[11] C. McGann, F. Py, K. Rajan, H. Thomas, R. Henthorn, and R. McEwen, "A deliberative architecture for auv control," in *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, may 2008, pp. 1049 –1054.

[12] M. Quigley, K. Conley, B. P. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "Ros: an open-source robot operating system," in *ICRA Workshop on Open Source Software*, 2009.

[13] P. M. Newman, "Moos—a mission oriented operating suite (tech. rep. oe2003-07)." Department of Ocean Engineering, MIT, Cambridge, MA:, Tech. Rep., 2003.

[14] M. Chitre, "Path planning for cooperative underwater range-only navigation using a single beacon," in *Autonomous and Intelligent Systems (AIS), 2010 International Conference on*, 2010, pp. 1 –6.

[15] Y. T. Tan and M. Chitre, "Single beacon cooperative path planning using cross-entropy method," in *IEEE/MTS OCEANS, KONA, Hawaii*, September 2011.