Improving the Coordination Efficiency of Multiple AUV Operations Using Prediction of Intent

Chris C. Sotzing, David M. Lane
Ocean Systems Laboratory, Heriot-Watt University
Edinburgh, Scotland, UK, EH14 4AS

Abstract

This research addresses the problem of coordinating multiple autonomous underwater vehicle (AUV) operations. A mission executive has been created that uses multi-agent technology to control and coordinate multiple AUVs in communication deficient environments. By incorporating a simple broadcast communication system in conjunction with real time vehicle prediction this architecture can handle the limitations inherent in underwater operations and intelligently control multiple vehicles. In this research efficiency is evaluated and then compared to the current state of the art in multiple AUV control.

Keywords: Autonomous Underwater Vehicles, Distributed Control, Multi-Robot Systems, Robot Coordination, MCM, Agent Prediction, Hybrid Architectures

Introduction

The goal of this research is to merge cooperative robotics, Autonomous Underwater Vehicle (AUV) and multi-agent technologies together to create a robust multi-AUV control architecture. The potential benefits of multi-vehicle operations include force-multiplier effects, redundancy avoidance and the utilisation of heterogeneous vehicle configurations to reduce risk to expensive assets.

Despite these benefits, current architectures for such operations are often simple extensions of single vehicle designs that do not allow for large amounts of communication, let alone cooperation. Ideally, a multi-agent architecture should allow for a significant increase in mission efficiency without requiring a proportional increase in communication. This study utilises the architecture created in the Ocean Systems Laboratory (Figure 1) and creates an intelligent mission executive (the DELPHIS system) which is compared to the current state of the art in multiple-AUV control.

In architectures such as these missions are planned in the deliberative layer using an
adaptive mission planning module. These plans are then passed to the executive layer where they are completed by the mission executive. When faced with problems or new information the mission executive traditionally requires the mission planner to perform a mission re-plan. This process is time consuming and not practical in most real world situations.

Aimed to work in conjunction with the mission planner described in [1], this work presents an intelligent mission executive that can coordinate vehicles in communication poor environments while keeping the need for a mission re-plan to a minimum. This executive repair functionality aims to keep most conflicts local (solvable by the mission executive) and not global (only solvable with a mission re-plan).

**Background and Related Work**

**Multi-Agent Systems**

A multi-agent system is a computational system where two or more software modules or components, known as agents, work together to solve a problem. An agent is considered to be a *locus of problem solving activity* that tries to fulfill a set of goals in a complex, dynamic environment [2]. Depending on the type of environment, an agent can take many different forms. Robots can be considered agents that can sense and actuate in the physical world. The ability of such systems to exchange information, decisions and plans at distance and on heterogeneous platforms make them a powerful tool.

**Cooperative Robotics**

Research into cooperative robotics dates back to the late 1980’s when the challenges of multiple robot systems began to be investigated. This was the fusion of two different research areas, singular robotics and distributed problem solving, which until that point had been independent. The work can be grouped into two general areas: non-mobile robots (for example, production line robotic assembly), and the less constrained, arguably more challenging area of mobile robots.

Most of the initial research in the area of multiple Autonomous Mobile Robots (AMR), concentrated on the behaviour-based paradigm [3]. The biological inspiration for this paradigm has led many researchers into investigating techniques that can mimic the social characteristics of insects and animals. However much of this innovative research has to date focused on robots working in highly constrained environments [4].

Many of the real-world applications being undertaken and proposed for AMRs are highly task and plan oriented. Examples of applications include exploration/mapping [5], area patrol [6] and formation control [7]. Existing architectures draw strongly on conventional computer structures and symbolic algorithms (for example, script-based task execution, and expert knowledge systems); it is not clear that biologically inspired behaviour-based paradigms will be useful in the foreseeable future.

The challenge is that as the role of AMRs continues to expand, and as the level of autonomy increases, how can autonomous decision making be extended across multiple vehicles?

**AUV Control Architectures**

The current state of the art in AUV control consists of three different types of architectures: deliberative, reactive and hybrid [8]. Each of these control paradigms are widely used in the mobile robotics world though the descriptions here will be focused specifically on their application in AUVs.
Deliberative architectures are based around planning. This planning process utilises a world model that consists of a set of goals. A plan is generated based on these goals and then executed by a control system. One of the downsides of deliberative architectures is that all actions have to be planned by the high level planner modules. This becomes a problem when a plan has to be modified during execution.

Reactive architectures, also known as behavioural architectures, were the catalyst that helped start the field of collaborative robotics. These architectures are event based systems where certain situations result in certain behaviours. The real world acts as a model to which the robot reacts [8]. Though reactive architectures solve the quick reaction time problem present in deliberative architectures, they lack their global planning ability and consequently aren’t as able to solve complex problems without large populations of robots.

Hybrid architectures combine the benefits of deliberative and reactive architectures into one. Normally these systems are divided into three distinct layers: the deliberative high level layer for planning, the executive layer for vehicle instruction and sensor processing, and the functional reactive layer for data input and motor control [9]. Because hybrid architectures implement characteristics from deliberative and reactive architectures, they are currently the most common choice for AUV control [10].

**Multi-AUV Coordination**

Current AUV missions are getting more complex and it is becoming common to require multiple vehicles to work together to solve these types of tasks. Currently there are two main approaches towards getting AUVs to cooperate: the adaptation of single vehicle architectures, and the application of current cooperative robotics architectures. Currently, the state of the art in AUV mission control is script-based mission execution systems. Missions are written in the form of scripts which are executed sequentially by the vehicle. In these systems task order is predefined by the user, though event based script-switching is possible [11]. The benefit of such systems is that because all vehicle actions are predefined, behaviour is very predictable. Conversely, though predictable, scripted plans don’t allow for dynamic task allocation and consequently often do not optimally execute the mission given unforeseen events.

The most common way to coordinate multiple AUVs is what is known as a stoplight system. In these systems current single-AUV script execution systems are modified to include stoplight points where vehicles can synchronise with each other. The benefit of such systems is that current AUV control architectures can be utilised with few modifications. However, autonomy is sacrificed because the user has to plan the static sequence of the mission in advance including all synchronisation points.

Another approach to multi-AUV cooperation is the modification of existing cooperative robotics approaches to the underwater domain. Sariel et al. [12] proposes a bidding system used to coordinate simulated AUVs in a mine-countermeasures operation. In this system any vehicle can act as auctioneer to generate bids on a task, and assuming good communication, allows for dynamic goal selection and a high level of cooperation. However, the dependency on consistent communication illustrates a major challenge when applying existing multi-robot techniques to the underwater domain: how can a multi-AUV architecture maintain a high level of cooperation and simultaneously minimise the need for an similarly high level of communication?
System Design

The DELPHÍS system [13] is an intelligent mission execution system that allows for efficient multi-AUV coordination in environments where communication cannot be guaranteed. The system acts as the mission executive in a standard hybrid three-layer architecture (Figure 1). The sub-architecture for the DELPHÍS system is shown in Figure 2.

![Figure 2: System schematic of the DELPHÍS system.](image)

The Abstract Layer Interface (ALI) developed in the Ocean Systems Laboratory uses a defined set of messages that are translated to be understood by the host architecture. In using the ALI this system can be used to control any vehicle simply by applying its generic constructs as the mission executive of the host-vehicle architecture. Within the DELPHÍS system there are three major modules: the mission model, AUV database and mission controller.

**Mission Model**

The mission model module contains the mission file and controls interaction with it, including status updates, goal retrieval, etc. A custom mission representation system was created that embraces the limitations of underwater operations in addition to allowing for intelligent multi-agent, multi-vehicle coordination. The system, known as BIIMAPS (Blackboard Integrated Implicit Multi-Agent Planning Strategy), is described in more detail below.

Mission plans are currently represented in a number of different ways. The most common options are if-then-else scripts and hierarchical task networks. If-then-else scripts are the current state of the art however they tend to be rather inflexible and unable to cope with unforeseen events. Hierarchical task networks show more promise for flexibility due to the fact that the goals aren’t necessarily constrained by any order. The BIIMAPS system is based on these plan representations and adds the functionality of blackboard systems to allow for more intelligent, robust behaviour.

In the BIIMAPS system (Figure 3), mission plans are represented as a tree of tasks, or goals, which are the major building blocks of the system plan. Goals (represented by large circles in Figure 3) can either be atomic (leaf goal), or divided into sub-goals; each goal can be in one of three states: complete, ready or locked. In addition, goals can be marked current when they are currently being executed by an agent.

![Figure 3: BIIMAPS mission modelling system diagram. Large circles represent goals; white circles contain operations. Dependency is represented by a dotted arrow.](image)

All leaf goals have operations associated with them (represented by white circles in the diagram). An operation is the behaviour associated with the certain goal.
For instance if a goal is to navigate to a certain point, the operation would be the waypoint coordinates. In addition non-leaf goals can also have operations associated with them. If a goal’s sub-goals were a set of waypoints the root goal’s operation could be to operate a camera so that it recorded throughout the execution of all sub-goals. This situation is represented in Figure 3.

To determine when a goal has been completed, each goal has a condition. Conditions can be simply the completion of a goal’s operation or they can be externally dependent, such as a message received from another agent. The conditions for non-leaf goals can be a logical combination of their sub-goals, as shown in the diagram.

Goals can also have dependencies and constraints applied to them to determine when they are available for execution. Dependencies are when a goal relies upon another (represented by the dotted arrow in Figure 3). A good example of this would be if goal $x$ is to navigate to a point, goal $y$ might be to clear the area of mines. Goal $x$ would then depend upon goal $y$. A constraint is similar, but instead of referring to another goal, it refers to the state of the world.

The BIIMAPS system was designed as a mission representation system with blackboard functionality. In this case BIIMAPS itself can be considered the blackboard and the agents updating the mission act as the knowledge sources. Because of this the system can be rolled back to a previous state should any previous update that other updates depend upon become untrue.

**AUV Database**

The AUV database is a module that contains information about all of the AUVs in the mission, a system similar to that being used in [12]. This information includes static data such as vehicle type and dimensions as well as variable data like battery life, average speed, sensors, and sensor status. Variable data is updated periodically to ensure that each AUV has the most recent status of its peers.

The goal of the AUV database is to allow for intelligent goal selection as well as behaviour prediction in the case of a loss of communications. When communications are lost, each vehicle still has to make decisions about the mission, despite not being able to contact the others. By consulting the AUV data-base, predictions can be made using the most recent status of each vehicle.

**Mission Controller**

The mission controller is the main decision making unit in the architecture. Using data from the mission model and the AUV database it coordinates all execution level decision making within the vehicle. It is made up of a number of sub-modules that fall into three main categories: communication, goal selection and agent prediction.

![RAUVER autonomous underwater vehicle (Ocean Systems laboratory, Heriot-Watt University).](image_url)
message system [14]. In this way control commands can be easily passed to the autopilot in addition to allowing for simple interfacing with any OceanSHELL capable vehicle, including the hover capable AUV, RAUVER (Figure 4).

This architecture utilises an acoustic broadcast system whereby vehicles periodically send out status information to all other vehicles in range. This information includes AUV data with a unique ID number, AUV type (hover capable, torpedo, etc.), current position, average speed, battery life as well as mission information, consisting of current goal, a list of previously achieved goals and newly discovered targets.

Broadcast communication has been chosen over vehicle to vehicle (unicast) transmissions to allow for greater ability to handle intermittent communications, which is common in systems operating in the underwater environment.

System Functionality

In DELPHÍS system each AUV is considered an agent and functions as such. Initially, all AUVs start out with an identical copy of the mission plan. Before beginning execution of the mission, the status beacon is started so that vehicles acting in the system can register with each other, subsequently populating each other’s AUV data-bases. This allows for external AUV information (including position, intention, etc.) to be factored in to initial AUV goal selection.

Goal selection works in conjunction with the mission model to select the best possible goal for the AUV to achieve. Utilising the BIIMAPS system, the mission model is able to return a list of only the goals in the mission that are available for execution based on the conditions, dependencies and status of each goal. This list is then passed to the goal selection algorithm where it is pruned down to a single goal representing the best available task for the given AUV. If no goal is available the AUV will wait in a holding pattern until one becomes available. Once the goal is executed, its status is updated in the mission model and the loop repeats until the mission is complete. The algorithm is represented in pseudocode below.

```
(1) while(there are available goals)
(2)   get list of available goals
(3)   select goal
(4)   remove goals currently in progress
(5)   remove goals that require payloads AUV lacks
(6)   remove all but the highest priority goals
(7)   choose the closest goal
(8) |
(9)   execute goal
(10) update goal status in mission model
(11) |
```

When AUV broadcasts are received, the information pertaining to the sending AUV is used to update the receiving AUV database. In addition, the current goal and list of completed goals from the sending AUV are passed into the receiving mission model, synchronising it. In this way all mission models contain the same data across all vehicles.

Information in the broadcast relating to the mission status is continually built upon throughout runtime. As the vehicles run the mission their list of completed goals (and discovered targets) continually grows. Due to this, every message received contains the entire mission history of the sending vehicle. This is extremely important for two reasons: first it allows for simple reconciliation of mission data in the likely event of dropped communications, and second, it allows for new AUVs to easily enter the mission at any point during mission execution.

Prediction

The ability to handle and synchronise data in intermittent communications is not enough for a multi-AUV coordination system, since in many cases vehicles may be out of contact for extended periods of
time. In these scenarios it is important to be able to predict the actions of other AUVs to enable more intelligent goal selection.

This architecture handles this eventuality with an AUV-Monitor module that keeps track of the AUV database and makes predictions for vehicles when the time since last communication rises above a certain threshold. Utilising the most recent data from the AUV database (last known position, average speed, current goal, etc.) the AUV-Monitor can estimate the current position of the vehicle. In addition it can update the mission model should the predicted position indicate that the vehicle has achieved its goal (in this case the updated status is recorded in the mission model with a predicted flag). Additionally, the AUV-Monitor can then calculate the most likely next goal by passing the AUV information into its own goal selection algorithm. When communication returns, the mission model is re-synchronised and the predicted values replaced with actual data.

As multiple vehicles are working together and constantly updating a common plan, in certain cases the current goal for a certain AUV may become less suitable than another during execution. Rather than force complete execution on any goal attempted, this architecture has a Mission-Monitor module that keeps track of the mission model and constantly checks to see if the current goal is the best possible choice. Should it be determined that this is not the case, the current goal execution is stopped and goal selection is run again. In this way the system ensures that each vehicle is always performing the best possible action.

In the cases of intermittent communications, it sometimes occurs that two AUVs are found to be executing the same goal. The Mission-Monitor is able to detect this and calculate which AUV is most suited for execution. Because this module is running on both vehicles and each one has information about the other in their AUV database, they will come to the same decision.

Results

In order to evaluate the system’s ability to increase efficiency, this architecture has been tested against the current state of the art in multiple-AUV operations (script-based stop-light systems) in executing a mine countermeasures (MCM) mission.

![Image](image.png)

Figure 5: Three RAUVER AUVs coordinating an MCM mission in simulation using the augmented reality visualisation tool found in [15].

In this scenario, efficiency of multiple AUV operations is defined as a combination of the following:

- Mission speed
- Goal redundancy
- Missed goals
- Mine discovery

Experiments were conducted in simulation to determine the efficiency of the system in deteriorating communication environments. A high-fidelity simulator developed in the Ocean Systems Laboratory can replace and very accurately model the RAUVER AUV. This in combination with an augmented reality visualisation tool [15] provides a realistic environment for the software-in-the-loop evaluation of vehicle execution code. It also provides a useful test-bed for the development and testing of vehicle
modules before embedding them on the vehicle. An image of the DELPHÍS system coordinating a simulated MCM mission is shown in Figure 5.

**Efficiency**

To evaluate efficiency this research compared four different multi-AUV control architectures: a stoplight system and three versions of the DELPHÍS system (optimised, un-optimised and prediction failure). The un-optimised system was simply the DELPHÍS system with all optimisation modules disabled. The prediction failure system was the DELPHÍS system with the prediction module predicting actions happening faster than in reality.

The MCM mission was executed by each system for 2-4 vehicles in varying degrees of communication success ranging from 100-10%. Results can be seen in Figure 6, Figure 7 and Figure 8.

As can been seen here the efficiency of the four coordination systems shows clear trends as the communication rate drops. The optimised DELPHÍS system is able to maintain a high level of efficiency throughout, with only a minor drop towards the 10-20% communication success rate. The prediction failure DELPHÍS system was only marginally less efficient reporting data that was just under that of the optimised system. This is logical as incorrect predictions would result in some decrease in efficiency. However, in many cases the other mission optimisation techniques would make up for any major errors.

The un-optimised DELPHÍS system did not fare as well as communication rates dropped. It showed a clear decline in all three group sizes. This decline is due to the amount of coordination conflicts that occurred while communications were down. More specifically, it was the lack of ability to fix these conflicts when communications returned that affected efficiency.

The stoplight system also showed lower efficiency than the optimised DELPHÍS system, however this was mostly due not to coordination errors (since stoplight systems are by definition scripted a priori) but to the time it took for the mission to be accomplished. The stoplight system was often slower than the other three systems and this had a major effect on efficiency. A
good example of this can be seen in the 2 vehicle data where the stoplight system recorded significantly lower efficiency than the other systems. This was because in the 2 vehicle scenario, the MCM mission took significantly longer when controlled by the stoplight system than it did when controlled by the 3 DELPHÍS systems.

Stoplight efficiency data wasn’t only affected by time however. There were some coordination errors as well mostly, regarding the newly discovered mines which were not initially scripted in the mission but added at runtime. These targets were chosen as they came available however in low communications and without the mission optimisation techniques of the DELPHÍS system any conflicts couldn’t be worked out. This was particularly common in the experiments with 3 and 4 vehicles.

In some cases efficiency of the runs was returned as above 100%. In this research mission time was compared to the expected data which is the time data returned by the DELPHÍS system at 100% communications. In the cases where efficiency was above 100% this simply meant that the mission was accomplished faster than the DELPHÍS system. This is evident in the 4 vehicle data (Figure 8) where the stoplight system starts off more efficient than the others.

*Behaviour*

A major advantage of the DELPHÍS system is that it does not require the mission to be started with all the vehicles at once. As mentioned before, AUVs can be added during run time and this has been tested and proven to work with four vehicles (and the data indicates it working with many more). This capability has proven to be extremely useful especially when a mission is taking longer than expected.

Interesting emergent behaviour has been discovered in the case of two vehicles attempting the same goal (this situation has to be manually created, as it only occurs when communication is down and prediction has failed). In this case sometimes AUVs will alternate as to which vehicle should execute the goal, until an agreement is made. This seems to be due to the time between status broadcasts and only happens when vehicles are nearly equidistant from the goal. When the time between broadcasts is decreased this behaviour disappears.

Prediction is an extremely important ability for a truly robust multi-AUV architecture. These results have shown that the efficiency of multi-AUV coordination systems with prediction is higher than those systems that lack it. In addition, even with inaccurate prediction, the DELPHÍS system was able to maintain a higher level of efficiency than the current state of the art.

The DELPHÍS systems ability to handle and communicate newly discovered targets has been shown to work in most conditions except complete communication loss. When a target is discovered it is added to the local mission model with an ID unique to the AUV that found it. This is done so that if while out of communication range two AUVs discover the same target, they aren’t given the same ID number. Due to this, even in badly degraded communications vehicles can accurately transmit new targets to other vehicles without conflict. When no communication is possible, targets are simply held by the AUV and transmitted when communication returns.

*Conclusions*

This system for coordinating multiple AUVs has shown promising results in increasing mission efficiency as compared to the current state of the art. Through the use of dynamic goal selection, distributed
mission plans and agent prediction the DELPHIS system can successfully coordinate multiple heterogeneous vehicles in most ranges of communication. Though focused on AUVs this work can easily be extended to any UxV platform.

**Future Work**

In the future the initial aim is to apply these results to real vehicles and validate these findings. Having already successfully controlled a single vehicle (RAUVER) mission it is expected that the results presented here will be reflected on multiple AUVs in the field.

In addition, another important aspect of this research that will be studied further is the interaction between this system and the mission planner [1]. Because it was the DELPHIS system being tested, this study focused on its ability to coordinate vehicles and simplified the problem by not requiring mission re-plan. In the future this interaction between architecture levels will be investigated.

**References**


Acknowledgements

The work reported in this paper was funded by the Systems Engineering for Autonomous Systems (SEAS) Defence Technology Centre established by the UK Ministry of Defence. The authors thank the members of the Ocean Systems Laboratory at Heriot-Watt University and SeeByte Ltd. for their generous support.