Semantic Based Knowledge Representation and Adaptive Mission Planning for MCM Missions using AUVs

Georgios Papadimitriou, David Lane
Ocean Systems Laboratory, School of Engineering & Physical Sciences
Heriot-Watt University, EH14 4AS, Edinburgh, UK
Email: gp125@hw.ac.uk

Abstract—**Mine Countermeasures (MCM) are a substantial challenge in the domain of underwater operations especially when using Autonomous Underwater Vehicles (AUVs). The work presented in this paper focuses on the semantic representation of knowledge and its utilization for mission planning and plan adaptation in a simulated MCM environment using the Nessie AUV. With respect to semantic knowledge representation this work builds upon the KnowRob system. Various ontologies model the environment, the AUV components and capabilities as well as the planning domain. For planning and plan adaptation we use the POPF Planning Domain Definition Language (PDDL) planner. Plan adaptation reuses previous plans when calculating a new plan thus saving computational resources. The main contribution of this work is an approach that relies on semantic information to represent knowledge in a MCM context and its usage for planning MCM missions in an energy efficient manner.**

I. Introduction

Underwater Mine Countermeasures (MCM) deal with identifying mined areas to be avoided and locating and neutralizing individual mines [1]. The purpose of MCM operations within the Unmanned Underwater Vehicles (UUVs) context can be summarized into the following: "to field a common set of unmanned, modular MCM systems operated from a variety of platforms or shore sites that can quickly counter the spectrum of threat mines assuring access to naval forces with minimum mine risk" [2]. MCM can be broken down to the following phases: Detect (D), Classify (C), Identify (I) and Neutralize (N) [2]. These phases can be either performed by one or more vehicles in one or more passes. In the case of one vehicle and two phases the approach to MCM could be that in the first pass the vehicle would perform Detection (D) and and Classification (C) of Mine Like Objects (MLOs) using appropriate sensors (e.g. side-looking sonar). In the second pass, the vehicle would perform Identification (I) of mines using electro-optic sensors (e.g. cameras) and neutralization (N) of those deemed to be mines using some neutraliser (e.g. a stationary bomblet that is placed in the area and is remotely detonated later using an acoustic command [2]). The MCM scenario that is investigated in this paper is a variation of the one described above and involves detection, classification, reacquisition and inspection of mines in two passes using one vehicle. The first pass involves the phases of detection and classification of MLOs while the second, the phases of reacquisition and inspection.

This work has been supported by DSTL contract DSTLX-1000064104.

When using Autonomous Underwater Vehicles (AUVs) for MCM operations, it is important that these vehicles demonstrate persistence in their autonomy (i.e. they make fewer requests for assistance from a human supervisor when they get stuck). For doing so, an efficient knowledge representation, reasoning and planning approach is needed. Current systems used across a variety of applications use ontologies to represent causal links between objects or concepts. Typically, these ontologies are not driven by live sensor data operating in the domain. In addition, environments are partially known *a priori*, and should be observed and updated as a basis for decision making in planning. Goal sequences that produce *a priori* mission plans must be capable of adaptation on the fly while maintaining coherence with entry and exit states required by other parts of the plan. Consequently, this requires adaptability in world modelling and planning in both space and time, in the presence of vehicle and sensor constraints and limitations (e.g. endurance, speed, range and accuracy).

In this paper we wish to demonstrate: i) an approach based on ontologies that can be used by AUVs performing MCM to semantically express knowledge about their components, their state and the state of the world in which they are deployed, ii) The usage of semantic knowledge to plan MCM missions, adapt it on the fly in the presence of new information and execute it successfully in a simulation environment using a Planning Domain Definition Language (PDDL) planner.

II. Related Work

Work on ontological representations for robotic applications builds largely on a significant prior body of work on ontologies for structuring and representing knowledge in various other domains. These include the semantic web, life sciences, medicine, biology, aerospace, astronomy and the maritime industry among others. Robotics offer unique challenges to knowledge representation and processing.

Research on utilizing ontological knowledge representations in robotic applications includes systems such as KnowRob [3], [4], a knowledge representation and knowledge processing framework built for household assistant robots. A similar framework is the ORO system [5], which leverages a core robotics ontology to integrate data from diverse sources such as sensors, domain knowledge, and human input and to provide useful, structured knowledge that can help the robot in interactions with humans. An ontology for general situation awareness in military applications robotics is presented in [6]. A particularly relevant work is [7] where the authors describe a planning and control system for AUVs built on an ontology. Planning and mission data, vehicle capabilities, and vehicle status information are all stored in the ontology, which allows for much more reactive and flexible planning and improved situation awareness.

Regarding adaptive mission planning, we find relevant approaches that adopt the reusing previous planning effort philosophy. In *plan repair* approaches if a plan turns out to be invalid a process is triggered in which consistent parts of the problematic plan are reused in order to formulate a new plan. In [8], plan repair is devised in the context of MCM scenarios using AUVs in an effort to repair already computed plans to resolve problems that have invalidated them. In [9] plan repair is compared to replanning from scratch in the context of maintaining plan stability. The results show that plan repair and consequently the reusing previous planning effort approach can be beneficial.

III. Semantic Knowledge Representation and Processing

In a semantic approach for representing knowledge, logic based representations are the natural choice for several reasons. First, because logic based representations are ideal for representing relational knowledge. Second, because logic based systems can be very expressive, depending on the kind of logic used. Third, because logic based representations are governed by well defined syntax and semantics and offer a variety of well researched inference approaches. In our approach semantic knowledge is expressed through OWL DL (Web Ontology Language Description Logics) ontologies. OWL DL provides much of the expressiveness of first-order logic while providing very desirable features such as soundness, completeness and decidability upon applying reasoning. Our knowledge framework builds upon KnowRob [3], [4]. KnowRob is implemented mostly in SWI Prolog [10] and is available through ROS (Robot Operating System) [11]. Using Prolog, KnowRob loads ontologies into the system and stores them as triples of the form `rdf(Subject, Predicate, Object)`. OWL files are parsed using the Thea OWL parser, which itself uses SWI-Prologs Semantic Web library for parsing the RDF/XML syntax into RDF triples. Thea then builds a representation of the OWL ontology, meaning the OWL ontology syntax is implemented as Prolog terms. This internal representation allows knowledge to be accessed through Prolog predicates. Prolog predicates are not only used for accessing knowledge in the knowledge base (KB) but are also used for interfacing the KB with external data and for implementing KnowRob reasoning modules which can be extended based on application needs. This extensibility is a major advantage of KnowRob, meaning it can be adapted to the requirements of many robotic projects. Another benefit of KnowRob is that not all knowledge needs to be immediately available in the knowledge base: a special class of Prolog predicate called `computable` can be defined. With `computables` we can dynamically query or calculate a result using information from sensors. For instance, we can query the perception system for the attributes of some object or calculate the relationship between two objects using their positions (e.g. two detected mines are close to each other). In this way, the world plays the role of an additional, virtual knowledge base.

Following the KnowRob architecture we developed ontologies that model the AUV components and capabilities, an ontology that models the environment as well as an ontology that is used by the planner to generate mission plans. The advantages of using multiple ontologies for various aspects of an MCM task is that the system is easily extensible and modular, allowing for alternative ontologies that offer different conceptualizations of the same domain to be plugged into the system.

A. World Ontology

The world ontology models the environment in which the AUV platform will operate performing an MCM mission. It consists of six main classes: *Area*, *DetectedFeature*, *MineLikeObject*, *Mine*, *Point* and *Vector* as well as various subclasses and instantiations that are illustrated in figure 1.

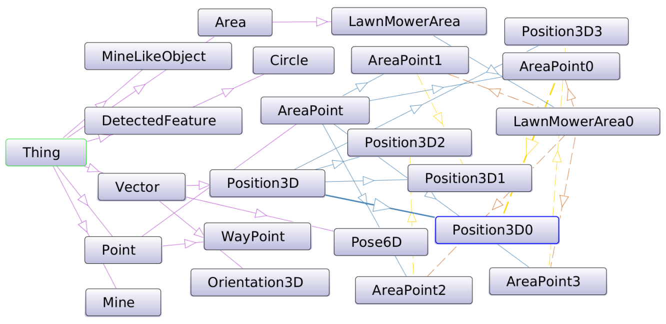


Fig. 1. MCM world ontology. Purple connections denote a subclass relationship and blue connections denote instantiations. Brown and yellow connections are denote properties that connect objects with each other.

The *Area* class and its subclass models the search area that the AUV will visit while in search mode for mines; in our case a *LawnMowerArea*. A *LawnMowerArea* instance (*LawnMowerArea0*) comprises four *AreaPoint* instances (brown connections). Every *AreaPoint* instance is associated with a *Position3D* instance denoted in yellow arrows. The *Waypoint* class is used for waypoints generated by the system for navigation and are associated with *Position3D*, *Orientation3D* and *Pose6D* instances (the combination of *Position3D* and

Orientation3D instances). The *DetectedFeature* class is used for features that are detected during the search for mines; in our case circular detections that will be used for asserting *MineLikeObject* instances into the ontology. The *Mine* class is used for instantiations that are created based on *MineLikeObject* instances that are deemed to be mines after the robot has reacquired their position. The instantiations shown in figure 1 represent the initial information that is available to the robot for starting its mission. In practice, *AreaPoint* instances based on which the system will generate a lawnmower trajectory (*Waypoint* instances) to perform a survey for mines.

B. Components and Capabilities Ontology

The components and capabilities ontology models the hardware and the software components of the AUV as well as its capabilities. The capabilities are determined based on the available components. Modelling such information in an ontology offers the advantage of performing platform capability matching with the mission that is at hand and determining if the platform is suitable for performing it.

Additionally, in the presence of a dynamic internal platform environment, components such as sensors can go offline due to some malfunction. In this situation the platform must have the ability to reassess its status and its capabilities. Based on our modelling, such reassessment is possible with the status of malfunctioning components changing accordingly and the capabilities being updated internally. Consequently the robot can determine whether it can progress with its mission by querying the ontology for relevant information. Due to space limitations, figure 2 shows a salient fraction of the components and capabilities ontology with their interdependencies. Brown connections from the *Nessie* AUV instance to the *MCM0* and *Navigation0* capability instances denote that Nessie has the capability of performing MCM and Navigation. The yellow connection from the *MCM0* capability instance to the *Navigation0* capability instance denotes that for an MCM capability to be available, the vehicle must first have a navigation capability. Finally, brown connections from the *Navigation0* capability instance to the *Thruster0-Thruster5, NavModule0, Gyro0, Compass0, and DVL0* component instances denote that the navigation capability is dependant on the existence of thrusters, a navigation module a gyroscope, a compass and a Doppler Velocity Log (DVL) sensor.

C. Planning Ontology

Typically a PDDL planner utilizes information encoded in a PDDL `domain` and a PDDL `problem` file in .pddl format in order to generate mission plans (see section IV). In a system that addresses the MCM problem, information exchange and sharing is critical for successful mission execution. Ergo, the knowledge representation structure used by the planner and the one used by the rest of the system should be integrated through a common basis to maximize efficiency. As a first step towards that direction we encoded the MCM PDDL `domain` along with its semantics into an ontology. Therefore, the system can query for information that was previously only accessible

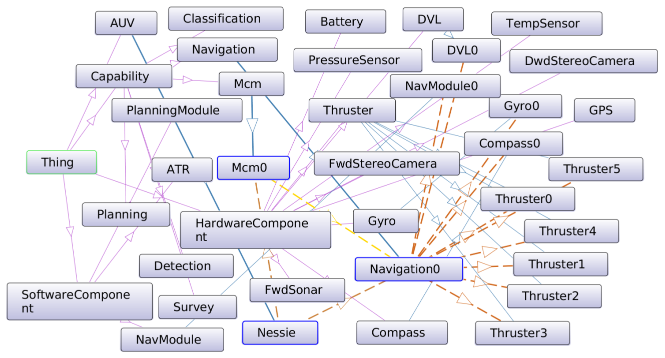


Fig. 2. Fraction of the components and capabilities ontology. Purple connections denote a subclass relationship and blue connections denote instantiations. Again, brown and yellow connections are properties that connect objects with each other.

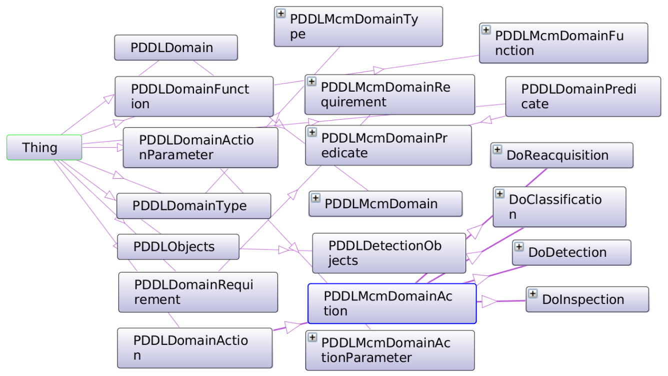


Fig. 3. Class decomposition of the planning ontology. Purple connections denote a subclass relationship. The *DoDetection*, *DoClassification*, *DoReacquisition* and *DoInspection* classes represent actions that are available within an MCM domain.

by the planner. The system also uses the `domain` information encoded into the ontology to generate the `domain` file dynamically by querying for relevant information. For instance, in order to construct a PDDL action the system queries for the action name, its parameters, its precondition for it to be applicable as well as its postcondition/effect after it has executed. Figure 3 shows the class decomposition of the planning ontology depicting the modelling of the PDDL `domain` for our MCM mission while figure 4 shows the reconstructed PDDL `domain` from the planning ontology.

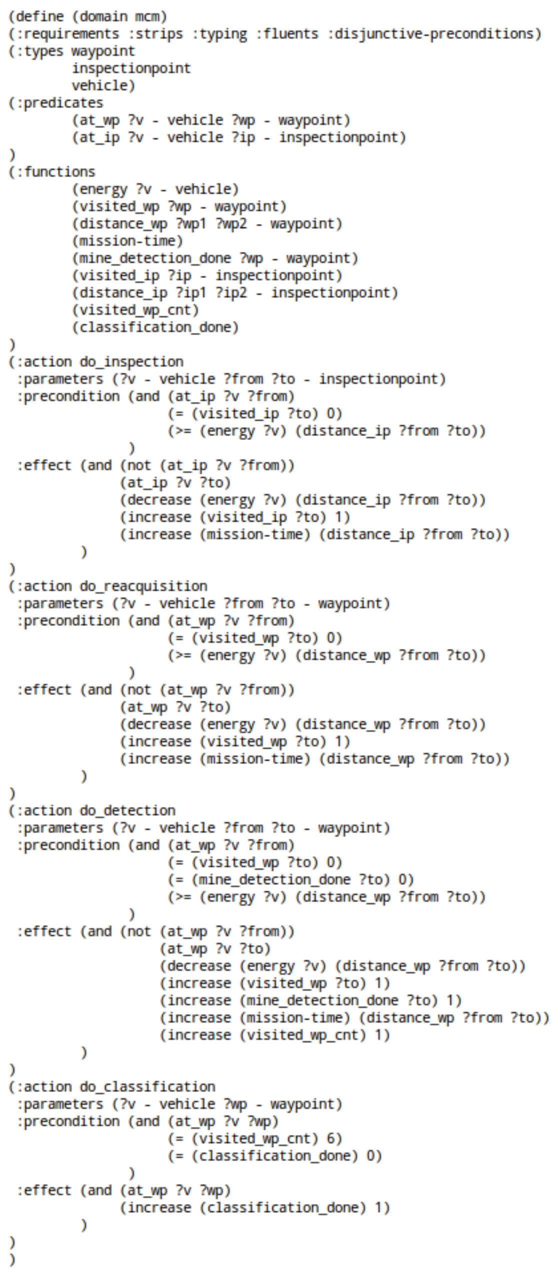
IV. Mission Planning

(Mission) Planning is the process which, given some initial state, a desired goal state and some actions, generates an action sequence (plan) that transitions a system from the initial to the goal state. Mission planning can be a very resource demanding and time consuming process that is strongly affected by the size of the problem's state space. With respect to the MCM domain, causes of state space explosion are: the dynamic nature of the environment, partial observability and the inherent uncertainty of the sensors. Consequently, plan adaptation approaches are essential for robust mission planning which is a key requirement towards building persistently autonomous vehicles. Our work is based on the planning framework within the PANDORA project [12] which uses the POPF PDDL planner [13]. Our approach uses deterministic planning and plan adaptation reusing previous planning effort to build new plans in the event of new information being available to the system. Some information about how planning is performed was given in section III-C. In the following subsections we present a more in depth analysis of how a plan is generated, how it is adapted and how the vehicle visits and inspects mines in an energy efficient manner.

A. Generating Mission Plans

To enable the planner generate mission plans we need to provide a `domain` and a `problem` to solve within this `domain`. In sort the `domain` file contains type, predicate, function and action definitions that the planner needs in order to generate a plan given some `problem` with some objects and some initial and goal state.

1) The MCM Domain: The MCM `domain` naturally contains all the definitions relevant to MCM scenarios (see figure 4). Domain `types` are the types of the objects/parameters that are used in domain `predicates`, `functions` and `actions`; in our case `waypoint`, `inspectionpoint` and `vehicle`. Domain `predicates` as well as domain `functions` are used in domain `actions`. Predicate `at_wp ?v - vehicle...` for instance denotes that some vehicle `?v` is at waypoint `?wp` while `at_ip ?v - vehicle...` denotes that some vehicle `?v` is at inspection point `?ip`. Vehicle `?v`, waypoint `?wp` and inspection point `?ip` are variables/parameters. Function `energy ?v - vehicle` for instance represents the remaining energy of the vehicle (executing actions consumes energy). Domain `actions` are consisted of `parameters`, a `precondition` that needs to hold in order for the `action` to be selected as part of a plan (and consequently be executed) and an `effect` or `postcondition` which is the outcome of executing the `action`. Action `do_detection` for instance is an action that is used in plans where the goal is to detect mines. The `parameters` of the action are a vehicle (`?v`) and two waypoints (`?from` and `?to`). As a `precondition` the vehicle `?v` needs to be at waypoint `?from`, waypoint `?to` must not have been visited, the mine detection task must not have been performed up to waypoint `?to` and the energy of the vehicle `?v` must be enough to cover the distance between waypoints `?from` and `?to`. The `effect` of executing the action is that the vehicle `?v` is no longer at waypoint `?from` but it is at waypoint `?to`, the vehicle's energy has decreased relatively to the distance travelled between waypoints `?from` and `?to` and waypoint `?to` is now visited. Furthermore, the mine detection task has now been performed up to waypoint `?to` and the mission time has increased relatively to the distance travelled. Action `do_reacquisition` is used in plans where the goal is to reacquire mines, action `do_inspection` is intended for plans where the goal is to inspect mines while action `do_classification` for plans where the goal is to classify object detections. The analysis of these actions is not presented here since it is very similar to the `do_detection` action.

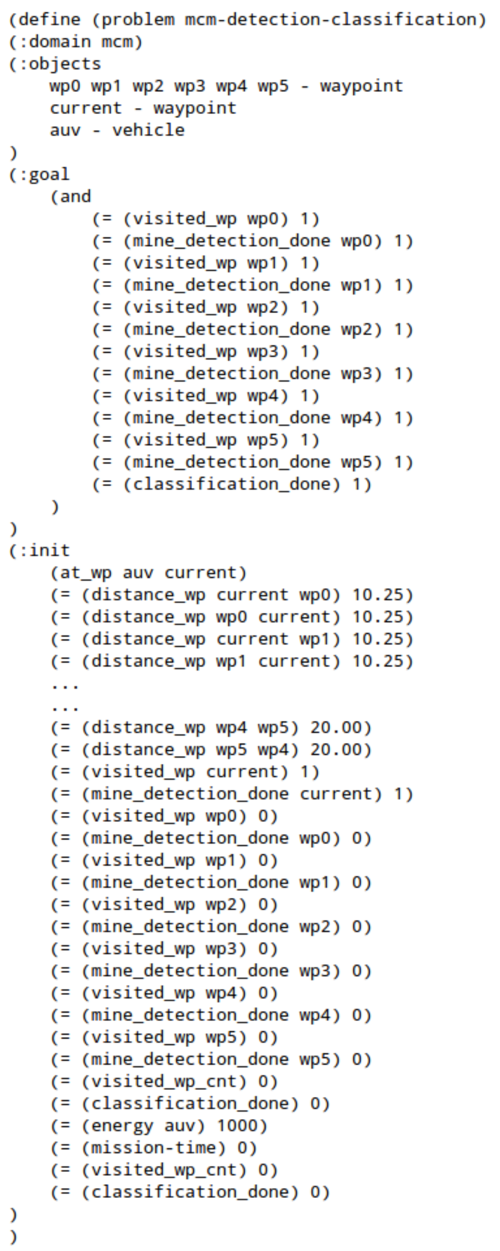


```
(define (domain mcm)
(:requirements :strips :typing :fluents :disjunctive-preconditions)
(:types waypoint
        inspectionpoint
        vehicle)
(:predicates
        (at_wp ?v - vehicle ?wp - waypoint)
        (at_ip ?v - vehicle ?ip - inspectionpoint)
)
(:functions
        (energy ?v - vehicle)
        (visited_wp ?wp - waypoint)
        (distance_wp ?wp1 ?wp2 - waypoint)
        (mission-time)
        (mine_detection_done ?wp - waypoint)
        (visited_ip ?ip - inspectionpoint)
        (distance_ip ?ip1 ?ip2 - inspectionpoint)
        (visited_wp_cnt)
        (classification_done)
)
(:action do_inspection
 :parameters (?v - vehicle ?from ?to - inspectionpoint)
 :precondition (and (at_ip ?v ?from)
                    (= (visited_ip ?to) 0)
                    (>= (energy ?v) (distance_ip ?from ?to))
               )
 :effect (and (not (at_ip ?v ?from))
              (at_ip ?v ?to)
              (decrease (energy ?v) (distance_ip ?from ?to))
              (increase (visited_ip ?to) 1)
              (increase (mission-time) (distance_ip ?from ?to))
         )
)
(:action do_reacquisition
 :parameters (?v - vehicle ?from ?to - waypoint)
 :precondition (and (at_wp ?v ?from)
                    (= (visited_wp ?to) 0)
                    (>= (energy ?v) (distance_wp ?from ?to))
               )
 :effect (and (not (at_wp ?v ?from))
              (at_wp ?v ?to)
              (decrease (energy ?v) (distance_wp ?from ?to))
              (increase (visited_wp ?to) 1)
              (increase (mission-time) (distance_wp ?from ?to))
         )
)
(:action do_detection
 :parameters (?v - vehicle ?from ?to - waypoint)
 :precondition (and (at_wp ?v ?from)
                    (= (visited_wp ?to) 0)
                    (= (mine_detection_done ?to) 0)
                    (>= (energy ?v) (distance_wp ?from ?to))
               )
 :effect (and (not (at_wp ?v ?from))
                   (at_wp ?v ?to)
                   (decrease (energy ?v) (distance_wp ?from ?to))
                   (increase (visited_wp ?to) 1)
                   (increase (mine_detection_done ?to) 1)
                   (increase (mission-time) (distance_wp ?from ?to))
                   (increase (visited_wp_cnt) 1)
         )
)
(:action do_classification
 :parameters (?v - vehicle ?wp - waypoint)
 :precondition (and (at_wp ?v ?wp)
                    (= (visited_wp_cnt) 6)
                    (= (classification_done) 0)
               )
 :effect (and (at_wp ?v ?wp)
              (increase (classification_done) 1)
         )
)
)
```

Fig. 4. Reconstructed MCM PDDL domain from the planning ontology.

2) The MCM problem(s): Based on the description of the MCM scenario given in section I, the MCM `problem` can be broken down to additional problems: one `problem` for the phases of detecting objects and classifying them as MLOs (two-phase first pass), one for the phase of reacquiring them and one for the phase of inspecting them (two-phase second pass). A fraction of the formulation of the `problem` for the first pass is shown in figure 5.

First pass: The problem `objects` shown in figure 5 are



```
(define (problem mcm-detection-classification)
(:domain mcm)
(:objects
    wp0 wp1 wp2 wp3 wp4 wp5 - waypoint
    current - waypoint
    auv - vehicle
)
(:goal
    (and
        (= (visited_wp wp0) 1)
        (= (mine_detection_done wp0) 1)
        (= (visited_wp wp1) 1)
        (= (mine_detection_done wp1) 1)
        (= (visited_wp wp2) 1)
        (= (mine_detection_done wp2) 1)
        (= (visited_wp wp3) 1)
        (= (mine_detection_done wp3) 1)
        (= (visited_wp wp4) 1)
        (= (mine_detection_done wp4) 1)
        (= (visited_wp wp5) 1)
        (= (mine_detection_done wp5) 1)
        (= (classification_done) 1)
    )
)
(:init
    (at_wp auv current)
    (= (distance_wp current wp0) 10.25)
    (= (distance_wp wp0 current) 10.25)
    (= (distance_wp current wp1) 10.25)
    (= (distance_wp wp1 current) 10.25)
    ...
    ...
    (= (distance_wp wp4 wp5) 20.00)
    (= (distance_wp wp5 wp4) 20.00)
    (= (visited_wp current) 1)
    (= (mine_detection_done current) 1)
    (= (visited_wp wp0) 0)
    (= (mine_detection_done wp0) 0)
    (= (visited_wp wp1) 0)
    (= (mine_detection_done wp1) 0)
    (= (visited_wp wp2) 0)
    (= (mine_detection_done wp2) 0)
    (= (visited_wp wp3) 0)
    (= (mine_detection_done wp3) 0)
    (= (visited_wp wp4) 0)
    (= (mine_detection_done wp4) 0)
    (= (visited_wp wp5) 0)
    (= (mine_detection_done wp5) 0)
    (= (visited_wp_cnt) 0)
    (= (classification_done) 0)
    (= (energy auv) 1000)
    (= (mission-time) 0)
    (= (visited_wp_cnt) 0)
    (= (classification_done) 0)
)
)
```

Fig. 5. Fraction of the planning problem for the first pass (detection & classification).

the actual objects that will be used for solving the problem for the first pass; in our case seven objects (`wp0-wp5`, `current`) of type `waypoint` and an object (`auv`) of type `vehicle`. `Current` is the waypoint that the `auv` is currently in and `wp0-wp5` are the intermediate waypoints of the lawnmower area. The `goal` is to generate a plan in which all the lawnmower waypoints will be visited (practically a lawnmower survey trajectory will be followed) and detection and classification tasks will be performed. The initial condition `init` is that the `auv` is at its current position, the distances of waypoint pairs (expressed in meters) are known, the current waypoint is visited and all the other waypoints are unvisited. Moreover, mine detection has not occurred inside the lawnmower area apart up to the current waypoint, the initial energy of the vehicle is known, the mission time is zero (the mission has not started) and classification has not been performed. The detection as well as the classification into MLOs are described in section V.

Second pass - reacquisition: Similarly we formulate the `problem` for the reacquisition phase of the second pass. This time though, waypoints represent positions of the MLOs. The `goal` is to generate a plan in which mines will be reacquired (practically all MLOs will be visited for reacquisition). The initial condition `init` is that the `auv` is at its current position, the distances of waypoint pairs (expressed in meters) are known and the current waypoint is visited. Finally, all the other waypoints are unvisited and detection and classification have occurred, the energy of the vehicle is known (the remainder after executing the survey following a lawnmower trajectory) and the mission time continues from the previous part of the mission.

Regarding energy efficiency, for the second pass we use a metric within the problem definition `:metric maximize (energy auv)` so that the planner will attempt to generate a plan maximizing the remaining energy of the vehicle by visiting MLOs in such a way that the smallest possible distance is travelled. In practice this formulates a Travelling Salesman Problem (TSP). The TSP is addressed by the planner with build in heuristics, hence the solution produced is taking into consideration energy efficiency but is suboptimal.

Second pass - inspection: The `goal` is to generate a plan to inspect every reacquired mine. We generate six inspection points (`ip0-ip5`) for every reacquired mine in a circular sequence around the mine and then perform inspection. The initial condition `init` is that the `auv` is at its current position, the distances of inspection point pairs (expressed in meters) are known and all the inspection points are unvisited. The energy of the vehicle is known (the remainder after visiting the MLO and reacquiring the mine) and the mission time continues from the previous part of the mission.

Based on the formulation of the MCM domain and the problems presented above the planner produces a sequence of actions (plans) that are shown in Table I. Currently, the PDDL `problem` files are not fully reconstructed from an ontology but most of the information residing in them is fetched (e.g. which waypoints are unvisited) or calculated (e.g. pair distances are calculated based on waypoint positions) from the knowledge framework described in section III. Consequently, the knowledge representation framework is tightly connected to the planning framework and guides decision making by providing critical information to it.

B. Plan Adaptation

As mentioned earlier, in this paper plan adaptation is based on reusing previous planning effort and it is triggered when new information, relevant to the current plan executing, becomes available to the system. More specifically, plan adaptation is performed during the second pass of the MCM scenario

TABLE I
ACTION SEQUENCES (PLANS) FOR DIFFERENT PHASES OF THE MCM SCENARIO.

First pass - detection & classification	Second pass - reacquisition	Second pass - inspection
1. (do_detection auv current wp0)	1. (do_reacquisition auv current mlo_wp6)	1. (do_inspection auv current ip0)
2. (do_detection auv wp0 wp1)	2. (do_reacquisition auv mlo_wp6 mlo_wp2)	2. (do_inspection auv ip0 ip1)
3. (do_detection auv wp1 wp2)	3. (do_reacquisition auv mlo_wp2 mlo_wp5)	3. (do_inspection auv ip1 ip2)
4. (do_detection auv wp2 wp3)	4. (do_reacquisition auv mlo_wp5 mlo_wp0)	4. (do_inspection auv ip2 ip3)
5. (do_detection auv wp3 wp4)	5. (do_reacquisition auv mlo_wp0 mlo_wp1)	5. (do_inspection auv ip3 ip4)
6. (do_detection auv wp4 wp5)	6. (do_reacquisition auv mlo_wp1 mlo_wp3)	6. (do_inspection auv ip4 ip5)
7. (do_classification auv wp5)	7. (do_reacquisition auv mlo_wp3 mlo_wp4)	

after the vehicle has performed reacquisition. The vehicle starts off the second pass with the positions of the MLOs (asserted into the world ontology) being known after detection and classification completion. As the vehicle executes its plan (see second column of Table I) and reacquires a mine, it needs to inspect it. The new information available to the system in this case is the position of the reacquired mine. Here we make the assumption that there is uncertainty arising from the vehicle sensors. Therefore, the positions of the MLOs are inaccurate compared to the positions of the reacquired mines due to their detections being performed from a much longer distance than reacquisition. In this new situation, the vehicle has to both inspect the reacquired mine and continue reacquiring mines for inspection. Instead of replanning from scratch by considering a new problem which is the merge of the two different phases of the second pass we keep the remaining unexecuted plan from the reacquisition phase and append it to the end of the plan for inspection that is shown in the third column of Table I. In this way the planner does not have to consider how to devise a plan for continuing reacquiring mines since it has already done so once. As a result, we save the planner from unnecessary computational load, speeding up the planning process (see Table II).

V. MINE DETECTION AND CLASSIFICATION

Mine detection and classification are vital phases of an MCM mission since the success of subsequent phases (e.g. reacquisition) is greatly affected by their quality. Approaches for detecting and classifying mines are beyond the scope of this paper, rather we focus on approaches for knowledge representation and planning in an MCM context. The assumption made is that for a real world application, an efficient Automatic Target Recognition (ATR) module is available to the system.

Nevertheless, for the purpose of presenting our approach, we used a circle detector for detecting circular features based on the Hough transform provided by the OpenCV [14] library. During the first pass, as circles are detected, they are asserted into the world ontology (instances of the *Circle* class) along with their positions. This constitutes the basis for the classification that follows.

Due to multiple detections of the same object, the detected circles are clustered using the affinity propagation clustering method [15]. Affinity propagation is a method that considers measures of similarity between pairs of data points and is very efficient with uneven cluster sizes. Using affinity propagation, cluster centroids are classified as MLOs. Figure 6 shows the

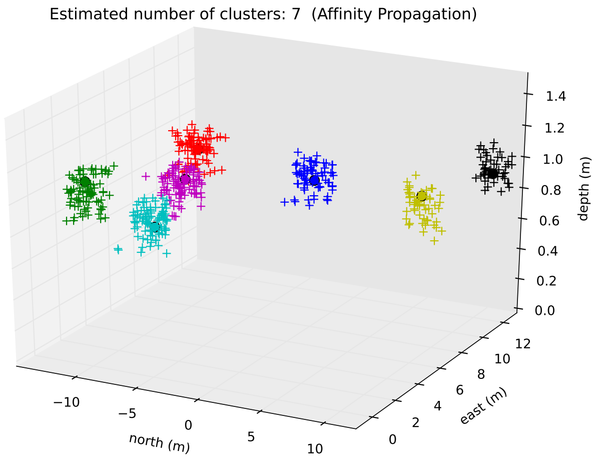


Fig. 6. Clustering of detections. Crosses represent detected circular objects while coloured circles represent cluster centroids.

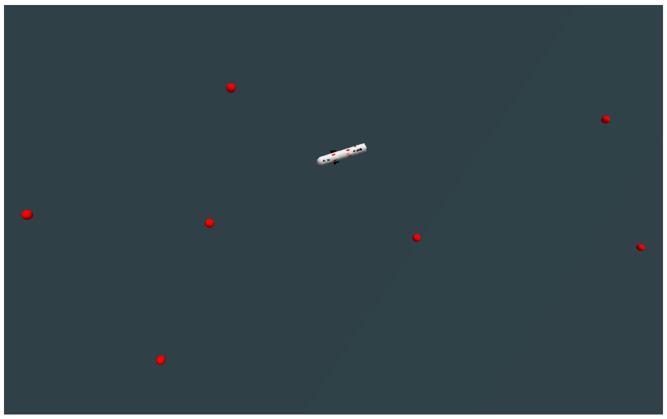
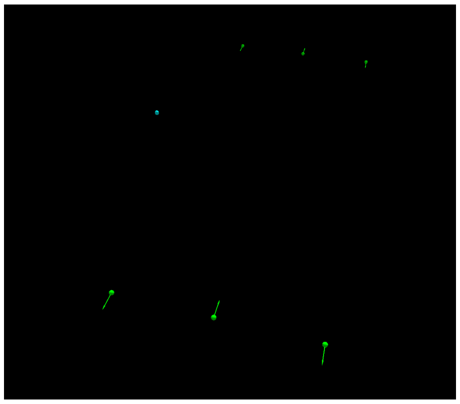


Fig. 7. Experimental setting illustrating the Nessie V AUV and seven mines (red spherical objects).

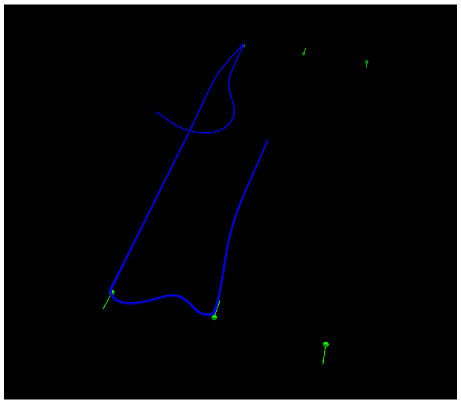
results of clustering and consequently classification of MLOs after completing the first pass.

VI. EXPERIMENTS

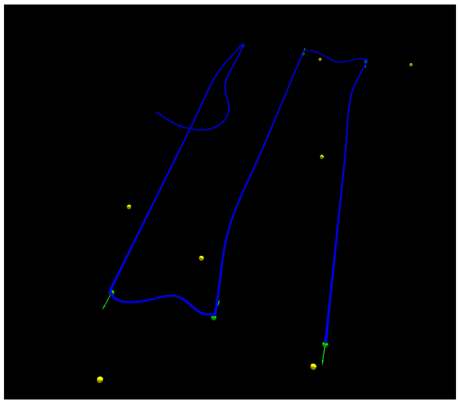
Two experiments are carried out in simulation. In both experiments we simulate a simple MCM setting with seven spherical objects representing underwater mines. The vehicle used is the simulated version of the Nessie V AUV [16]. Figure 7 shows the experimental setting. The two experiments are described below.



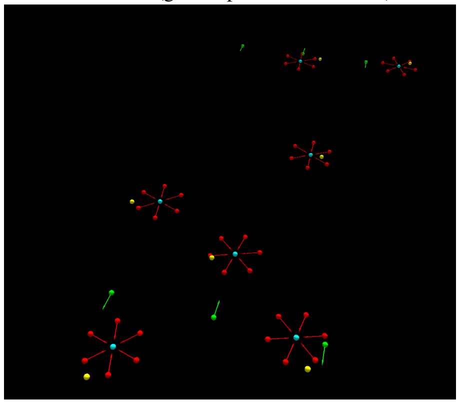
(a) Lawnmower intermediate waypoint generation with orientations (green spheres and arrows).



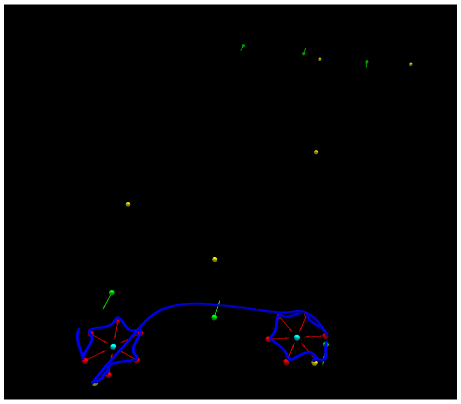
(b) Lawnmower trajectory execution.



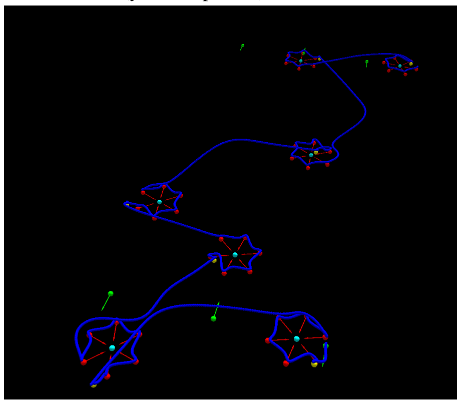
(c) Lawnmower trajectory completion and MLO classification (yellow spheres).



(d) Inspection points with orientations (red spheres and arrows) around reacquired mines (teal spheres).



(e) Reacquisition and inspection execution.



(f) Reacquisition and inspection completion.

Fig. 8. MCM mission stages.

Experiment 1: Following the requirements of the MCM scenario as it is described in section I, the vehicle has to perform a two pass-four phase MCM mission. For the first pass the AUV is given a square area modelled inside the world ontology. Based on the given area the system has to generate a lawnmower trajectory and assert it into the knowledge base along with semantic information (e.g. initially, all the intermediate waypoints of the trajectory are marked as unvisited). The planner has to devise a plan for traversing the intermediate trajectory waypoints and perform detection of MLOs and classification based on the information residing in the knowledge base. Finally the system is tested for reacquiring the mines and inspecting them successfully in an energy efficient manner by devising plans and adapting them appropriately. In this first experiment we apply the "reusing previous planning effort approach".

Experiment 2: In the second experiment we execute the same mission as in the first experiment but this time the planner re-plans without taking into consideration previous planning effort. Consequently, we test the outcome of both approaches in terms of time spent planning.

Results: The results of the first and second simulated experiments are illustrated in figure 8 and Table II respectively. The first pass and second pass phases where successfully executed by the vehicle as shown in figure 8. Figure 8a illustrates the lawnmower intermediate waypoint generation with the orientation at each waypoint. The orientation does not refer to the waypoint directly but rather to the orientation the vehicle should have when visiting each waypoint. The intermediate waypoints are generated based on the lawnmower area that is encoded in the world ontology (see section III-A). Following the lawnmower intermediate waypoint generation the vehicle devises a plan for following a lawnmower trajectory while searching for mines (see sections III-C, IV-A, V) and executes it successfully as illustrated in figures 8b, figure 8c. Based on the detections that are asserted into the knowledge base the systems classifies MLOs as shown in figure 8c (see also section V). This constitutes the completion of the first pass. The robot then continues with the second pass which is reacquisition and inspection of mines in an energy efficient manner as illustrated in figures 8e, 8f (see also sections IV-A, IV-B).

Moreover, judging from the results presented in table II we can see how beneficial the "reusing previous planning effort" approach is in terms of time spent planning and therefore in saving computational resources.

TABLE II
COMPARISON BETWEEN PLAN ADAPTATION BY REUSING PREVIOUS PLANNING EFFORT AND REPLANNING FROM SCRATCH.

Setting	**Time (s)**	
	Replanning from scratch	**Plan adaptation reusing previous planning effort**
7 Mines	95.024	0.607
6 Mines	22.799	0.213
5 Mines	7.535	0.13
4 Mines	3.693	0.108

VII. CONCLUSIONS

The results demonstrate how semantic based knowledge representation and planning can be used to plan and execute MCM missions successfully in an energy efficient manner even in dynamic environments. Expressing knowledge about the world and the vehicle as well as their respective states using ontologies provides a structured representation that is easy to understand and extend. The experiments have shown how a planner can benefit from an ontological representation in generating mission plans. Furthermore, we demonstrated how semantic information can be used for adapting a mission plan on the fly when new knowledge becomes available to the system. Finally, during adaptation we show how a previously generated plan can be used to reduce the computational load in calculating an adapted one.

Our approach yields promising results. As future work we would like to test how well the approach fits in additional applications to MCM. In particular, we believe that the true power of semantic knowledge representation and reasoning can be demonstrated through more complex scenarios such as the ones offered by underwater IRM (Inspection Repair Maintenance). Such scenarios offer a plethora of possible failure modes (e.g. failure to execute a valve turning task in an underwater oil manifold) that can be potentially dealt with by using semantic knowledge representation and reasoning that guide decision making in overcoming or even avoiding failures.

ACKNOWLEDGMENT

The authors would like to thank the Ocean Systems Lab staff for their input and support and in particular Dr. Tom Larkworthy for his advice on the integration.

REFERENCES

[1] S. Sariel, T. Balch, and J. R. Stack, “Distributed multi-auv coordination in naval mine countermeasure missions,” Georgia Institute of Technology, GVU, Tech. Rep. GIT-GVU-06-04, 2006, http://www2.itu.edu.tr/s̃ariel/publications.php.

[2] U. Navy, “The navy unmanned undersea vehicle (uuv) master plan,” *US Navy, November*, vol. 9, 2004.

[3] M. Tenorth, “Knowledge processing for autonomous robots,” Ph.D. dissertation, Technische Universität München, 2011.

[4] M. Tenorth and M. Beetz, “KnowRob – A Knowledge Processing Infrastructure for Cognition-enabled Robots. Part 1: The KnowRob System,” *International Journal of Robotics Research (IJRR)*, 2013, accepted for publication.

[5] S. Lemaignan, R. Ros, L. Mösenlechner, R. Alami, and M. Beetz, “ORO, a knowledge management platform for cognitive architectures in robotics,” in *International Conference on Intelligent RObots and Systems*, 2010.

[6] C. J. Matheus, M. M. Kokar, and K. Baclawski, “A core ontology for situation awareness,” in *Proceedings of the Sixth International Conference of Information Fusion, 2003*, vol. 1, 2003, pp. 545–552. [Online]. Available: http://ftp.isif.org/fusion/proceedings/fusion03CD/special/s36.pdf

[7] E. Miguelanez, P. Patron, K. E. Brown, Y. R. Petillot, and D. M. Lane, “Semantic knowledge-based framework to improve the situation awareness of autonomous underwater vehicles,” *IEEE Trans. on Knowl. and Data Eng.*, vol. 23, no. 5, pp. 759–773, May 2011. [Online]. Available: http://dx.doi.org/10.1109/TKDE.2010.46

[8] P. Patron, “Semantic-based adaptive mission planning for unmanned underwater vehicles,” Ph.D. dissertation, Heriot-Watt University, Edinburgh, 2010.

[9] M. Fox, A. Gerevini, D. Long, and I. Serina, “Plan stability: Replanning versus plan repair,” in *Proc. ICAPS*, 2006, pp. 212–221.

[10] J. Wielemaker, T. Schrijvers, M. Triska, and T. Lager, “SWI-Prolog,” *Theory and Practice of Logic Programming*, vol. 12, no. 1-2, pp. 67–96, 2012. [Online]. Available: http://dx.doi.org/10.1017/S1471068411000494

[11] “Robot Operating System (ROS),” www.ros.org, last accessed: 2014-02-20.

[12] M. Cashmore, M. Fox, T. Larkworthy, D. Long, and D. Magazzeni, “Planning inspection tasks for auvs,” in *Proceedings of OCEANS’13 MTS/IEEE, 2013*.

[13] A. J. Coles, A. I. Coles, M. Fox, and D. Long, “Forward-chaining partial-order planning,” in *Proceedings of the Twentieth International Conference on Automated Planning and Scheduling (ICAPS-10)*, May 2010.

[14] “Open Source Computer Vision Library (OpenCV),” http://opencv.org/, last accessed: 2014-02-20.

[15] B. J. Frey and D. Dueck, “Clustering by passing messages between data points,” *science*, vol. 315, no. 5814, pp. 972–976, 2007.

[16] R. Baxter, J. Cartwright, J. Clay, O. Clert, B. Davis, J. Lopez, F. Maurelli, Y. Petillot, P. Patrón, and N. Valeyrie, “Nessie v autonomous underwater vehicle,” last accessed: 2014-02-20.