Cooperative dynamic pathfinding for multiple autonomous underwater vehicles using D*

Vincent Myers
DRDC Atlantic

Defence Research and Development Canada – Atlantic
Technical Report
DRDC Atlantic TM 2011-044
January 2011

Abstract

A method for finding least-cost paths through an area by collaborating agents is presented. The method relies on extending existing dynamic path planning algorithms, namely D*, to the multi-agent case. The search tree is centrally updated and then used by the agents to cooperatively decide upon which areas to survey, the results of which are then used to update the current best path. Several numerical examples are given where it is shown that, on average, a competitive ratio — the ratio between the cost of the path found by dynamic algorithm and one operating with full knowledge of the area — of nearly one can be achieved while requiring a survey of less than half the total area. It is also shown that the total mission time scales linearly with the number of vehicles, meaning little to no effort wasted on vehicle coordination. In the context of mine countermeasures (MCM) operations with autonomous underwater vehicles (AUVs), the results indicate that a channel through a potentially mined area representing an amount of risk nearly equal to the optimal path can be found in significantly less time than the time required to survey the entire area.

Résumé

Voici une méthode permettant de trouver le parcours le moins coûteux dans une zone au moyen d'agents collaboratifs en étendant à ces derniers les algorithmes actuels de planification dynamique de parcours, à savoir D*. L'arborescence de recherche, mise à jour de façon centralisée, est utilisée par les agents pour décider conjointement des zones à couvrir. Les résultats ainsi produits collectivement servent alors à mettre à jour le meilleur parcours. Plusieurs exemples quantitatifs montrent qu'on atteint un rapport concurrentiel moyen de près de un entre le coût du parcours trouvé par l'algorithme dynamique et celui obtenu lorsque la totalité de la zone a été couverte, mais en limitant la couverture à moins de la moitié de l'ensemble de la zone. Ils montrent également que la durée totale de la mission varie de façon linéaire avec le nombre de véhicules, ce qui signifie qu'il n'y a pratiquement pas de temps perdu à coordonner le mouvement des véhicules. Dans un contexte d'opérations de lutte contre les mines marines (LMM) au moyen véhicules sous-marins sans équipage (VSSE), les résultats indiquent qu'il est possible de trouver une voie traversant une zone potentiellement minée présentant un niveau de risque presqu'égal à celui du parcours optimal dans un temps considérablement plus court que la couverture de toute la zone.

This page intentionally left blank.

Executive summary

Cooperative dynamic pathfinding for multiple autonomous underwater vehicles using D*

Vincent Myers; DRDC Atlantic TM 2011-044; Defence Research and Development Canada – Atlantic; January 2011.

Background: In the context of naval mine countermeasures (MCM) missions, one common task is to survey a channel using sonar such that mines — up to a degree of probability — are detected, classified and neutralized such that the number of mines that remain afterwards represents an acceptable degree of risk to the follow-on traffic. However, the probability of detecting and classifying objects, and the time required to neutralize them, is not the same in all areas, with some areas (*e.g.* a flat seafloor with little clutter) requiring much less effort than others (*e.g.* a rough seafloor with high clutter). Obviously, one would prefer that the channel in question contain areas where minehunting is easy, therefore reducing risk and on-task time (OTT). However, the actual placement of the channel by the Tasking Authority (TA), especially in expeditionary scenarios, is typically placed in a convenient location, one that minimizes transit distance or simply connects two staging areas.

This report presents a methodology where a path between two points is found through an initially unknown area. The method is greedy in that it continually attempts to minimize the cost of the current path. This method would be implemented using autonomous underwater vehicles (AUVs) equipped with side-scan sonar. In addition, several vehicles can collaborate together to more quickly find a suitable path; the vehicles perform surveys to assess the minehunting conditions (in this report, the number of mine-like objects in a area) and update one another's search spaces, which can result in a change in the best path, and corresponding adaptation of each AUV's mission.

Principal results: The technique is tested on simulated data and the fitness of the path found by the proposed on-line algorithm compared to the path that would be found if complete information about the area was known in advance. It is shown that essentially the same path as the optimal one can be found using this method, but requiring one to survey significantly less of the area, which directly translates into less surveying time and less time spent prosecuting contacts.

Military significance: It is hoped that the preliminary results shown in this report leads to a shift in tactics in relation to the location and placement of channels and staging areas during MCM operations when little to no previous information about the minehunting environment is known. The capability for AUVs to perform on-board sensor processing,

adaptive mission autonomy and collaboration is reaching a point where it may be feasible to allow a team of AUVs to find the best channel or box in a significantly larger area that meets the mission objectives while also minimizing the risk, subject to certain constraints such as water depth or distance from shore. This would compel the TA to issue an order to ”find a 200 meter wide channel from A to B with the most easy minehunting conditions,” rather than simply picking one semi-arbitrarily and leaving the resulting conditions to chance.

Future work: The near future could see a version of this algorithm implemented in DRDC's IVER2 vehicles and tested in a real scenario. This would require modules implemented in the Mission Oriented Operating Suite (MOOS) autonomy framework that drives the DRDC vehicles. The present work is a proof of concept of using incremental search algorithms in a cooperative way. As such, many technical improvements are possible, such as

- A decentralized version of the algorithm which could significantly decrease the amount of data that needs to be shared between vehicles.
- A modification of the algorithm that not only greedily chooses the next best area to survey (in the present case, the closest to the present location of the vehicle is selected), but also looks ahead a few time steps to see if perhaps another area has better long-term payoffs.
- Plan surveys that more quickly survey the potential path.
- Choose locations that also take into account the ability of the agents to communicate with each other.

Sommaire

Cooperative dynamic pathfinding for multiple autonomous underwater vehicles using D*

Vincent Myers ; DRDC Atlantic TM 2011-044 ; Recherche et développement pour la défense Canada – Atlantique ; janvier 2011.

Introduction : Le déroulement habituel des missions de lutte contre les mines marines (LMM) est marqué par la couverture d'une voie au moyen du sonar pour y détecter, classifier et neutraliser des mines avec dans un certain degré de probabilité afin que le nombre de mines résiduelles présente un niveau de risque que peut accepter le trafic qui emprunte par la suite la voie dégagée. En revanche, la probabilité de détecter et de classifier des objets et le temps nécessaire pour les neutraliser varient d'une zone à l'autre. En effet, cette durée et le degré d'encombrement du fond marin sont étroitement liés. De toute évidence, il est préférable que la voie à couvrir comporte des zones faciles à déminer pour réduire le degré de risque et le temps alloué à la tâche. Mais, l'autorité de l'attribution des tâches, surtout dans le cadre de scénarios expéditionnaires, choisit habituellement la voie en fonction de son utilité, c'est-à-dire la plus courte possible ou simplement entre deux zones de regroupement et d'attente (ZRA). Le présent compte rendu décrit une méthode qui permet de trouver un parcours entre deux points dans une zone inconnue au départ. Elle repose sur un principe d'économie, en ce sens qu'elle tente à tout moment de réduire au minimum le coût du parcours. Cette méthode est mise en oeuvre au moyen de véhicules sous-marins sans équipage (VSSE) équipés de sonar à balayage latéral. De plus, plusieurs véhicules peuvent collaborer pour trouver plus rapidement un parcours approprié. Ils effectuent des levés pour évaluer la situation en termes de chasse aux mines (dans le présent compte rendu, il s'agit du nombre d'objets de type mine présents dans la zone) et se transmettent entre eux des données à jour sur leurs espaces de couverture respectifs. Le tracé du meilleur parcours peut être ainsi modifié à chaque instant, et la mission de chaque VSSE est adaptée à ces changements.

Principaux résultats : Des données simulées ont servi à mettre la méthode en pratique. La pertinence du parcours trouvé par l'algorithme en ligne proposé a été comparée à celle du parcours optimal tracé à l'aide de données exhaustives connues d'avance sur la zone. Les tests montrent que les deux parcours sont pratiquement identiques. En revanche, le premier a nécessité une couverture considérablement moins importante de la zone. On obtient ainsi une économie substantielle de temps consacré aux levés ainsi qu'à la poursuite des contacts.

Portée militaire : Il est à souhaiter que les résultats préliminaires dont fait état le présent compte rendu mèneront à un changement de tactique en termes de choix de l'emplacement

des voies et des ZRA au cours des opérations de LMM lorsque l'environnement où s'effectue la chasse aux mines est peu connu ou totalement inconnu. La capacité actuelle des VSSE de traiter les données des capteurs embarqués, d'adapter leur mission et de collaborer entre eux est telle qu'il est maintenant possible de laisser un groupe de ces véhicules localiser la meilleure voie ou région dans une zone considérablement plus vaste qui répond aux objectifs de la mission tout en minimisant le risque et qui comporte certaines contraintes, comme la profondeur de l'eau et la distance par rapport au littoral. L'autorité d'attribution des tâches pourrait ainsi émettre l'ordre de « trouver une voie de 200 mètres de largeur entre les points A et B qui présente les conditions les plus faciles en termes de chasse aux mines » plutôt que choisir cette voie de façon quasi arbitraire et de s'en remettre à la chance pour que les conditions soient favorables.

Recherches futures : Dans un proche avenir, on pourrait voir une version de cet algorithme appliquée à des véhicules IVER2 de RDDC et testée en situation réelle. Il faudrait pour cela mettre en oeuvre des modules dans le cadre d'autonomie de l'outil (Mission Oriented Operating Suite (MOOS), qui commande les véhicules du RDDC. Le travail présenté ici constitue une validation du concept de l'utilisation d'algorithmes de recherche incrémentiels en mode coopératif. De nombreuses améliorations techniques sont ainsi possibles, notamment :

- une version décentralisée de l'algorithme qui réduirait considérablement le volume de données échangées entre les véhicules ;
- une version modifiée de l'algorithme qui choisit la meilleure alternative à la zone à couvrir en termes d'économies réalisées (dans le cas présent, celle qui se trouve le plus à proximité du véhicule) tout en analysant également quelques pas de temps pour identifier une autre zone plus rentable à long terme ;
- une planification des levés pour couvrir plus rapidement le parcours possible ;
- une sélection des emplacements en tenant également compte de la capacité des agents de communiquer entre eux.

Table of contents

List of figures

1 Introduction

Autonomous Underwater Vehicles (AUVs) equipped with high-resolution imaging sonar have become a common tool for many seabed surveying tasks, particularly those where it is either difficult to send a manned platform such as under-ice operations, or when it is desirable to keep personnel out of danger, such as mine countermeasures (MCM) activities. One common minehunting task is to survey a potentially mined channel or area using sonar so that all mines — up to a degree of probability — are detected, classified and neutralized such that the number of mines that remain afterwards represents an acceptable degree of risk to the follow-on traffic. However, the probability of detecting and classifying objects, and the time required to neutralize them, is not the same in all areas, with some areas (*e.g.* a flat seafloor with little clutter) requiring much less effort than others (*e.g.* a rough seafloor with high clutter). Obviously, one would prefer that the channel in question contains areas where minehunting is easy, therefore reducing risk and on-task time (OTT). However, the actual placement of the channel by the Tasking Authority (TA), especially in expeditionary scenarios, is typically placed in a convenient or semi-optimal way with respect to, for instance, the transit distance between two connected staging areas. This channel, however, may be severely suboptimal with respect to the amount of effort that must be expended to reduce the risk to an acceptable effort.

If one had complete information about the surrounding area into which the channel must be placed, then it would be trivial to choose a path that minimizes the minehunting difficulty. However, in general this information is not available, and an AUV must first survey the entire area in order to find the optimal path from the start to end points. When considering the area of typical sea lanes of communication — on the order of 10-20 km $\times$ 0.5-1 km — versus the sweep rate of a vehicle — 60 m $\times$ 1.5 m/s — this can require a significant amount of time, on the order of several days. One obvious way to speed up mission time is to employ n vehicles to survey the area in question. This currently done by simply dividing the area into $n_a \geq n$ smaller areas, each assigned to a vehicle, perhaps depending on that vehicle's particular capability or environment [1]. While advantageous, this method still requires that the entire area be surveyed in order to find the best channel. An improved tactic would be to direct survey activities in such a way as to find a suitable path around those regions that are particularly poor in terms of minehunting performance as those regions are discovered. In addition, when employing multiple vehicles, these should be allowed to communicate (as much as the available bandwidth permits) and coordinate their actions as much as possible. For instance, if the path that one vehicle is surveying seems particularly promising, then the other vehicles should alter their mission in such as way as to take into account this fact.

In this report, the area will be discretized into a number of cells of resolution $r \ll \ell$, where ℓ is the length of the channel; a graph $G = (V, E)$ will then be created using these cells as vertices V and edges E connecting them to neighbours in the north/south/east/west directions,

such that standard graph search algorithms can be applied. The shortest path problem is finding a path between two vertices that minimizes the total cost between the two vertices with the edges of the graph are assigned a cost, typically the Euclidean distance between the two vertices. Such algorithms are widely used in automatic navigation systems based on GPS-enabled route finding tools as well as web mapping services such as MapQuest® and Google Maps®. Many algorithms exist to solve the shortest path problem for two vertices in G, the most widely known being the A* algorithm [2], a method that uses heuristics to speed up the search for the shortest path.

1.1 Related Work

Generally, in the route finding problem defined above for AUVs, the graph G, specifically the edge costs E, are not known in advance. As the vehicle surveys the area and gathers sonar data, the true costs are discovered. These costs can be based on, for instance, some classification of the bottom in terms of the difficulty of finding targets of interest or, as in this report, some measure of the number of target-like contacts in the area. The problem becomes one of minimizing the total edge costs — a path that minimizes the total minehunting difficulty or one that minimizes the total number of contacts that require investigation and potential neutralization (in addition to the higher probability of missing targets in areas of high-clutter) — while the costs can change from the initial assumptions.

Path planning in a dynamic or unknown environment is a well-studied problem with many proposed solutions, such as D* [3], Focused D* [4], Lifelong planning A* [5] and its derivative, D* Lite [6]. Of course, when G changes one could simply opt to re-run an algorithm such as A* from scratch. However, if changes happen frequently, this becomes prohibitively expensive. These algorithms are extensions of heuristic search with *incremental* search methods, and attempt to re-use as much of the the search tree that has already been computed, and quickly identify those vertices that need to be updated, resulting in faster replanning times. This report will apply incremental search algorithms to the route finding problem to obtain a cooperative behaviour from multiple vehicles, and different strategies for assigning which regions of the area should be surveyed in order to reveal the values of the edges of G, and how the vehicles can cooperate in doing so.

The use of multiple agents in incremental search has been limited to hunter/prey scenarios, where one agent (the hunter) must move to the same location as a second agent (the prey), while the latter moves around the graph, usually while trying to evade the former. An example of this is Moving Target D*Lite [7]). In the present problem formulation, vehicles are permitted to communicate and cooordinate their actions as well as update each other's search trees. Multiple collaborating agents have been studied in exploration-type problems, which is in fact related to the present problem.

The coverage [8] and exploration [9] problems are related to the present path finding problem, and both have been extended to included teams of cooperating robots [10, 11]. The

coverage problem determines the path a robot must take in order to cover the space (*i.e.* visit each node in the search tree), and the back-and-forth pattern employed by AUVs while surveying (see the example in Figure 2 below) is precisely the same as the technique used in most coverage problems (the *boustrophedon* algorithm). However, research has focused on finding efficient ways of decomposing the free space, which contains obstacles and walls, in such a way as to allow an efficient coverage path to be computed. Efficient path planning methods for an AUV in an arbitrary environments, including the effect of underwater currents, were presented in [12]. The present problem is much more related to exploration, and the strategies described here can be seen as using the paths defined by the incremental search methods to compute the value of the different potential candidate cells for exploration.

The following section reviews the general shortest path problem, and presents the idea behind incremental search. In Section III, numerical examples are shown, where several AUVs are used to cooperatively search for a channel through an area. Some comments on those results, as well as future work, is given in Section IV.

2 Path finding in dynamic environments

In the single-source path finding problem [13] one is given a connected graph $G = V, E$ with vertices V and edge costs E, with the function $c : E \mapsto \mathbb{R}$ that maps edges to real-valued costs. A path $p = \langle v_0, v_1, \ldots, v_k \rangle$ in G is an ordered set of vertices and the total cost of p is:

$$c(p) = \sum_{i=1}^{k} c(v_{i-1}, v_i), \tag{1}$$

is the sum of the costs of the edges connecting the vertices in the path. The minimum cost from the start vertex v_s to the goal vertex v_g is

$$c^*(v_s, v_g) = \begin{cases} \min c(p) \text{ for } p = \langle v_s, \ldots, v_g \rangle & \text{if a path exists from } v_s \text{ to } v_g \\ \infty & \text{otherwise} \end{cases} \tag{2}$$

and p^* is any path where $c(p) = c^*$.

2.1 A* review

A* is a heuristic search algorithm for finding p^*. The algorithm makes use of a function

$$f(v) = g(v) + h(v), \tag{3}$$

to guide the search, which is composed of the function $g(v)$ which maintains the cost of the path from the start vertex v_s to the current vertex v and the heuristic $h(v)$ which is an *estimate* of the cost from a given vertex v to the goal v_g. The algorithm maintains a priority queue Q which initially contains only v_s, as well as a closed list C which is initially empty. The vertices in Q are indexed by their f value. The algorithm repeatedly removes the element v from Q with the smallest f value and adds it to C. It then adds all of the successor vertices of v to Q, indexed by their f value. This process continues until the goal vertex v_g is at the top of the priority queue, after which the optimal path p^* can then be reconstructed by starting in v_g and following the parent of each vertex back to the start vertex. The algorithm is given in pseudo code in Algorithm 1.

A* is an example of a *best-first search* — or *seemingly* best-first — since at every iteration, the vertex with the smallest f value, and so the one which appears to be the best, is chosen for expansion. One of the properties of A* is that if h is an admissible heuristic, meaning that it never overestimates the true distance to the goal node, then A* is guaranteed to find the shortest path. While A* is probably the most widely-known path finding algorithm, it can require a significant amount of computation time. In the present application, as the autonomous vehicles explore the space and discover changes in edge costs for the current path, the current path estimate will not be valid. Every time that an edge cost changes, one needs to re-run A* in order to compute a new optimal path. Since the entire area is

unknown at the beginning, this will result in A* constantly being called, and will result in prohibitively long computation times. To address this, a number of techniques have been developed to find paths in dynamic, changing environments that use incremental search methods to speed up search times. This is discussed next.

Algorithm 1 Rough outline of A*. The algorithm makes use of the POP function which removes the first element of the priority queue — corresponding to the one with the smallest f value — and returns it; and the SUCC(v) function, which returns all the vertices that are successors of v (that is, all v_i for which there exists a directed edge from v to v_i).

```
C ← ∅                                          ▷ The closed set C is initially empty
g(v_s) ← 0
h(v_s) ← heuristic distance to v_g
f(v_s) ← g(v_s) + h(v_s)
v_s → Q                                        ▷ Add the start vertex to priority queue
while Q ≠ ∅ do
    v ← POP(Q)                                 ▷ Take the first element of the priority queue
    if v = v_g then
        return                                 ▷ Found a path to v_g
    end if
    v → C                                      ▷ Insert v into the closed set
    for each v_i ∈ SUCC(v) do
        PARENT(v_i) ← v
        g(v_i) ← g(v) + c(v, v_i)
        h(v_i) ← heuristic distance to v_g
        f(v_i) ← g(v_i) + h(v_i)
        if v_i ∉ C then                        ▷ If v_i is not already on the closed set ...
            v_i → Q                            ▷ ... add it to the priority queue
        end if
    end for
end while
```

2.2 D* review

In order to speed up search times with changing edge costs, incremental search methods that reuse the results from previous searches to avoid repeatedly solving the same problem from scratch have been developed. The algorithms considered here transform the current search tree to the new search tree by identifying which nodes need to be added and deleted. The search is then able to carry on from this search point, reducing the computation time.

The original D* (meaning Dynamic A*) [3] and Focused D* [4] were developed to solve the robot navigation problem in unknown terrain. D*Lite [6] is another technique, derived from Lifelong Planning A* [14], and is the one used in this report. The main search

procedure is given in Algorithm 2 with support functions given in Algorithms 3 and 4.

Unlike A*, the dynamic algorithms start from the goal node v_g and work their way back to the start node v_s. D*Lite maintains two estimates to guide the search, a g-value, similar to the one above for A* (except that it is the distance from the goal, since the search has been reversed) and a *rhs* [1] value which is a one step look-ahead value for g, so:

$$rhs(v) = \begin{cases} 0 & \text{if } v = v_s \\ \min\limits_{v_i \in \text{PRED}(v)} g(v_i) + c(v_i, v) & \text{otherwise} \end{cases} \tag{4}$$

where PRED is a function that returns the predecessor vertices of v. Like A*, D*Lite maintains a priority queue which is kept sorted not by f, but by a two-valued key k defined as (see Algorithm 4):

$$k = [k_1, k_2], \tag{5}$$

where,

$$k_1 = \min g(v), rhs(v) + k_m + h(v), \tag{6}$$

and

$$k_2 = \min g(v), rhs(v). \tag{7}$$

The key also uses the heuristic function $h(v)$ as an approximate distance to the start node v_s. A vertex v is called consistent when $g(v) = rhs(v)$. Otherwise it is called inconsistent, specifically underconsistent if $g(v) < rhs(v)$ and overconsistent if $g(v) > rhs(v)$. The priority queue Q contains only those vertices which are inconsistent. The goal node is first added to Q with $rhs(v_g) = 0$ and $g(v) = \infty$, making it overconsistent, and calls the COMPUTESHORTESTPATH function, shown in Algorithm 3 , which again repeatedly removes the element with the smallest key from Q and expands it until the start node is reached. At this point, the same nodes as those that would have been expanded by A* are expanded. Then, the agent begins to move in the world, and discovers changes in edge costs. At this points, *rhs* values must be recomputed and nodes added or removed from Q as necessary, with updated keys. Then, COMPUTESHORTESTPATH is run again from the goal node but now to the current position of the robot using the nodes already in Q.

A large amount of computation time is spent on keeping Q sorted. To speed this up, D*Lite makes use of a key modifier k_m that keeps track of how much must be added to the first component of the key when a new key needs to be computed (see [6]) and avoids having to constantly re-order Q. The priority queue is this report has been implemented in a binary heap [13] in the MATLAB® environment.

[1] meaning *right-hand-side*

Algorithm 2 Summary of (unoptimized) D* Lite for a mobile agent (from [6])

$v_\ell = v_s$ ▷ v_ℓ is used to compute the key modifier when edge costs change below
$k_m = 0$
$rhs(v_g) \leftarrow 0$
$g(v_g) \leftarrow \infty$
$v_g \rightarrow Q$
COMPUTESHORTESTPATH
while $v_s \neq v_g$ **do**
  $v_s = \min_{v_i \in \text{SUCC}(v_s)} c(v_s, v_i) + g(v_i)$
  move to v_s
  if any $c(u,w)$ have changed **then**
    $k_m \leftarrow k_m + h(v_\ell)$
    UPDATE(u)
  end if
  COMPUTESHORTESTPATH
end while

2.3 Cooperative D*Lite

So far, we have described methods from the literature on path planning in static (A*) and dynamic (D*) environments. With few modifications, D*Lite can be used directly to solve the single vehicle path finding problem. D*Lite was conceived for a robot that is moving and discovers changes in edge costs, usually in the form of obstacles, and must now re-plan its route from its current location. The channel-finding problem described here is slightly different, in that an AUV is able to move directly (at some transit speed) to any part of the space, that is, any vertex in G. It should be noted that, unlike other path-finding problems with mobile robots, the AUV's movement is not affected by the terrain costs of the area; those terrain costs represent some fitness criterion for the traffic that will follow afterwards.

At each invocation of COMPUTESHORTESTPATH, D*Lite computes the path from the current location of the robot to the goal vertex v_g. In the MCM problem defined above, v_s never changes. The objective is always to find a path from v_s to v_g and, while the robot may move around the space, these vertices never change. One consequence of this is that the key modifier k_m is not required.

In D*Lite, when a path is found, the robot at position v_s begins to move to the vertex v_i successor of v_s that minimizes:

$$v_{\text{next}} = \min_{v_i \in \text{SUCC}(v_s)} c(v_s, v_i) + g(v_i). \tag{8}$$

For the AUV, it can directly proceed to any vertex in the graph at a constant transit speed. In this case, it shall move to the nearest vertex v_i that is on the current computed path p that

Algorithm 3 Function to compute the shortest path for D* Lite. **Notes**: The TOPKEY function returns the key of the top element in the priority queue, and UPDATEKEY(v) sets the key of vertex v using the COMPUTEKEY function. The PRED(v) returns the set of vertices which are predecessors of v. In this report the PRED and SUCC functions are equivalent. However, this may not always be so in the general case.

```
function COMPUTESHORTESTPATH
    k = TOPKEY(Q)
    while k <COMPUTEKEY(v_s) or rhs(v_s) ≠ g(v_s) do
        v ← POP(Q)
        if k <COMPUTEKEY(v) then
            UPDATEKEY(v)
            v → Q
        else
            if g(v) > rhs(v) then                    ▷ Vertex v is overconsistent
                g(v) ← rhs(v)                        ▷ Make v consistent
                for v_i ∈ PRED(v) do
                    UPDATE(v_i)
                end for
            else                                     ▷ Vertex v is underconsistent
                g(v) = ∞                             ▷ Set the g value to ∞ and update the queue.
                for v_i ∈ PRED(v) ∪ v do
                    UPDATE(v_i)
                end for
            end if
        end if
    end while
end function
```

Algorithm 4 UPDATE and COMPUTEKEY functions used in D*Lite.

function UPDATE(v)
  if $u \neq v_g$ **then**
    $rhs(g) \leftarrow \min_{v_i \in \text{SUCC}(v)} c(u, v_i) + g(v_i)$
  end if
  if $v \in Q$ **then**
    $Q \leftarrow \{Q - v\}$ ▷ Remove v from the priority queue, since only inconsistent vertices are permitted in Q
  end if
  if $g(v) \neq rhs(v)$ **then** ▷ v is now inconsistent and must be added to Q
    UPDATEKEY(u)
    $u \rightarrow Q$
  end if
end function
function COMPUTEKEY(v)
  $k_1 = \min g(v), rhs(v) + k_m + h(v)$ ▷ First element of the key
  $k_2 = \min g(v), rhs(v)$ ▷ Second element of the key
  return $[k_1, k_2]$
end function

has yet not been explored, that is:

$$v_{next} = \min_{v_i \in p} d(v_i, v_s) \text{ and } c(v_j, v_i) = \gamma, \tag{9}$$

where $d(u,v)$ is a distance function and γ is the *assumed* cost of getting from v_j to v_i, before any information is gained about that edge. If no costs on p are equal to γ, then a path has been found and the process terminates. Once the AUV arrives at the position corresponding to v_{next} it must then start a slower speed survey in order to discover the actual costs of those edges. The new costs are updated, with corresponding updates to the priority queue, and a new path is computed.

One way to extend the the single vehicle procedure to n vehicles is to first initially position each vehicle at n points along the first path that is computed before any information has been gathered — the straightest line path from v_s to v_g. Then, as vehicles survey the area, information about new edge costs is broadcast to the other vehicles, which then possibly changes the current path. The vehicles must each choose a new vertex to explore, which is basically Eqn (9), except that

- each vehicle must maintain an *exclusion zone* around the vertex a given vehicle is intending to survey. That way, vehicles do not spend time surveying the same area, and in practice, manages the waterspace to avoid collisions. Priority is given arbitrarily from vehicle 1 down to vehicle n.

- if, for a given vehicle, several vertices satisfy (9) then ties are broken in favour of the vertex that maximizes the distance from the other vehicles. This encourages a group behaviour where vehicles do not cluster together and spread out as much as possible around the search space.

The discrete time version of this algorithm is shown in Algorithm 5.

Algorithm 5 The multiple vehicle version of D*Lite. The function EXCLUSIONZONE(v) returns the vertices found on the defined exclusion zone around v – in this report these are the vertices that will be surveyed in addition to the vertex v. The function TIEBREAK(v) decides between several potential vertices in favour of the one which maximizes the distance between itself and the other vehicles' positions

$rhs(v_g) \leftarrow 0$
$g(v_g) \leftarrow \infty$
$v_g \rightarrow Q$
COMPUTESHORTESTPATH ▷ Returns the current best path p
$V_\gamma \leftarrow \{v_i \in p \cap c(v_j, v_i) = \gamma\}$ ▷ V_γ contains nodes on p that have unknown edge costs
while $V_\gamma \neq \emptyset$ **do** $X \leftarrow \emptyset$ ▷ X contains the vertices in the exclusion zone
  for $z = 1$ to n **do** ▷ Sequentially loop through the n vehicles
    $v_{next} = \min_{v_i \in V_\gamma} d(v_i, v_z)$ ▷ v_z is the current location of vehicle z
    if more than one $v_i = d(v_{next}, v_z)$ **then**
      $v_{next} =$ TIEBREAK ▷ Tie-break rule.
    end if
    $X \leftarrow X \cup$ EXCLUSIONZONE(v_{next}) ▷ Add v_{next} to the exclusion zone
    move to v_{next} and perform survey
  end for
  if any $c(u, w)$ have changed **then**
    UPDATE(u)
    COMPUTESHORTESTPATH
  end if
end while

function TIEBREAK ▷ Tie-breaking function
  $J \leftarrow \left\{ v_j, d(v_j, v_z) = \min_{v_i \in V_\gamma} d(v_i, v_z) \right\}$ ▷ Set of vertices with the same min. distance
  $W \leftarrow \{v_w, w = [1, \ldots, z-1]\}$ ▷ Locations of the other (lower priority) vehicles
  $v_{next} = \max_{v_j \in J}$ MEAN$(d(v_j, v_w \in W))$ ▷ Maximize the mean dist. from the other AUVs
  return v_{next}
end function

3 Numerical Results

In this section the method developed above is demonstrated with some numerical simulations. Data is generated to estimate its performance in finding a route through an area such that this route minimizes the total amount of MCM effort required to reach a given degree of clearance/risk, the cost function defined by Eqn (1). In practice, this could be defined in any number of ways: By using doctrinal bottom type definitions[2], one could instruct the vehicle to minimize the risk by staying as much as possible in areas in which it is easier to carry out MCM. In this study, the approach taken will be to minimize the total number of detected mine-like objects found on the path. Practically, in addition to the increased difficulty of finding targets in areas of high clutter, each one of these objects would need to be identified and possible neutralized, and so it makes sense to try to choose a route that contains the smallest possible amount of objects. Therefore, each vertex v that make up the area is assigned an integer number of objects $M(v) = [1, \ldots, m_{\max}]$. With the minimum number of objects equal to 1, it is possible to use the Manhattan distance as an admissible heuristic h for the informed search algorithms.

The type of vehicle modeled is based upon the specifications of a small man-portable AUV such as the DRDC's IVER2 [15] vehicle shown in Fig 1. The two parameters that affect the total mission time are used: The transit speed — the speed at which a vehicle will travel from one vertex to another — is set to $v_{tran} = 2.5$ m/s; and the survey speed — the speed at which the vehicle will perform a search for objects in an area — is set to $v_{sss} = 1.5$ m/s[3]. The sensor that is modeled is a side-looking high-frequency imaging sonar, such as the MarineSonic 900/1800 kHz sidescan sonar [16]. This sensor is assumed to have a maximum (one-sided) range of 30 m, so that its coverage rate can be calculated as $1.5 \text{ m/s} \times 60 \text{ m} = 90 \text{ m}^2/\text{s}$.

The data is generated by randomly drawing uniformly distributed integer numbers in the range $[m_{\min} \ldots m_{\max}]$. A representative channel finding task is defined by delineating an area of length $\ell = 10$ km and width $w = 3$ km, gridded at a resolution $r = 200$ m and an additional one-cell border around the outer perimeter, resulting in a graph with $|V| = 884$ vertices. Each vertex is connected to its east, west, north and southern neighbours. When a vehicle is tasked with surveying vertex v_i, it first transits there from its current location and plans a characteristic sidescan sonar survey that, once completed, reveal the number of mine-like objects in its 8-connected neighbourhood (*i.e.* the vertex v_i itself, plus its nord-south-east-west connected neighbour vertices as well as the diagonal vertices), as shown in Figure 2.

Three metrics will be used to assess the performance of the proposed technique: the total

[2]The are the definitions of minehunting difficulty as defined by military doctrine, such as *e.g.* type A for easy, B for more difficult, and type C for most difficult.

[3]AUVs typically have different transit and survey speeds, as the sensors such as synthetic aperture sonar (SAS), require slow speed in order to create a good image

mission time T, the percent of the area surveyed P_S and the competitive ratio ρj, defined below.

The total mission time T is defined as the time when the last of the n vehicles finishes its survey before a path is declared. This includes time spent surveying and time spent transiting from one part of the area to another. Note that, unlike most maze or terrain navigation problems found in the path finding literature, a underwater vehicle can transit from a vertex v_i to v_j at a fixed speed, and is not affected by terrain modifications, much like an aerial vehicle. As a baseline reference for the area defined above, to survey the entire area a single vehicle surveying would require a total of 109.14 hours (all time spent surveying). If one had n_{AUV} vehicles available and subdivided the area into v disjointed sub-regions is $109.14/n_{\text{AUV}}$[4]. In the case of $n_{\text{AUV}} = 2$, this means 54.57 hours of mission time is required to survey the entire area, after which the best path would be computed using A*.

The percent of the area surveyed, P_S, indicates the fraction $\times 100$ of the area that was required to be surveyed before a path was found. Again, in the baseline scenario, 100% of the area would be surveyd before finding the optimal path. Finally, the competitive ratio ρ compares the cost of the path p found by the on-line algorithm with the cost of the path p^* that would be found if the agents knew *in advance* the number of mine-like objects in the area. In this latter case, the optimal path p^* would be found by running the off-line A* algorithm on the known map. Then

$$\rho = \frac{c(p^\dagger)}{c(p^*)}, \tag{10}$$

where $p^\dagger$ is the optimal path determined by the on-line algorithm, to differentiate it from p^* which is the optimal path found with full knowledge of the space. A competitive ratio of 1 means that the cost of the best path and the cost of the one found using the proposed algorithm are the same. We will also compare the cost of the straight-line or shortest-distance path from v_s to v_g using

$$\rho_d = \frac{c(p^d)}{c(p^*)}, \tag{11}$$

where, similarly, a competitive ratio ρ_x of 1 indicates no improvement over arbitrarily picking the path to be p^d, the one that corresponds to the shortest distance from the start node to the goal node.

3.1 Example 1 - No defined path

The first example consists of simply generating uniformly distributed random integers $M(v_i) = \mathcal{U}(m_{\min}, m_{\max})$ from $m_{\min} = 1$ and $m_{\max} = 10$ for each vertex in $v_i \in V$. The

[4]This is the total time the AUV spends *on task*, and does not include things like vehicle recovery, battery charging times, repair, etc... that would increase to total mission time.

CF/DRDC deployment of IVER2 AUV
in Halifax harbour, fall 2010

Figure 1: *The IVER2 Autonomous Underwater Vehicle, manufactured by OceanServer Technology Inc. of Fall River, MA, USA, being deployed from a RHIB by DRDC and CF personnel in Halifax harbour. 2010.*

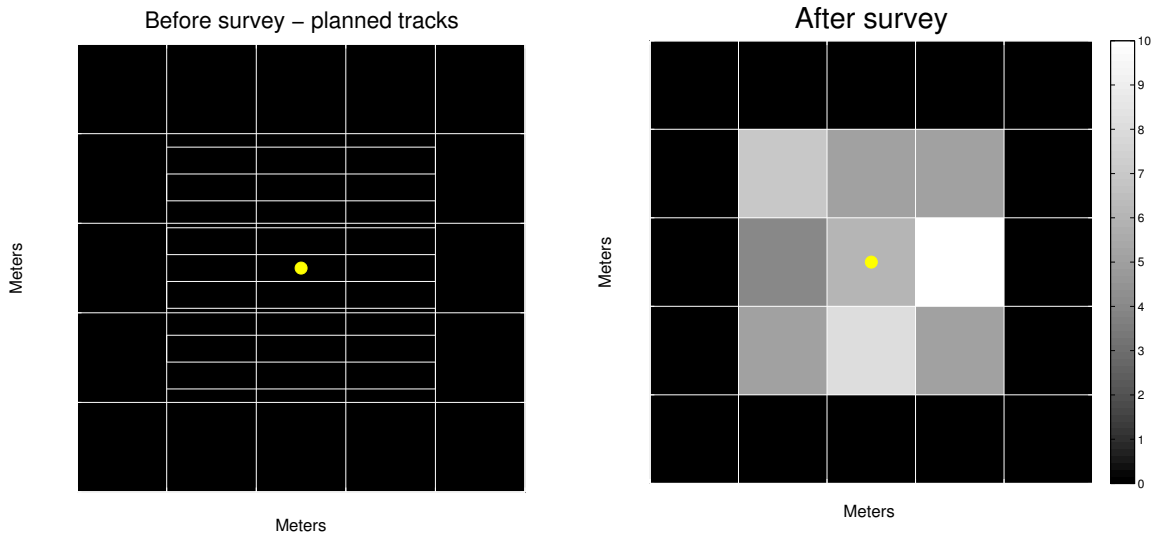

Figure 2: *Example track plan to survey the area surrounding the center vertex (marked with a yellow circle) of this 1 km* $\times$ *1 km area (each vertex represents a square area of resolution* 200×200 *meters. On the left, the standard lawnmower / ladder search, with survey legs spaced at 60 meters apart to ensure complete coverage. After the survey, the number of mine-like objects is reported for the vertices as well as its 8-connected neighbours, as shown on the right hand side of the figure.*

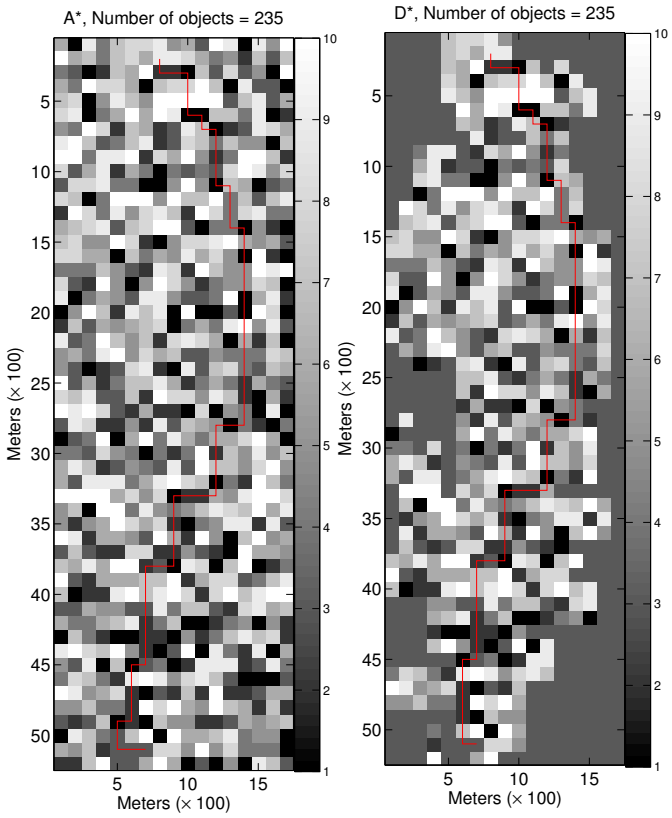

Figure 3: *Representative terrain from Example 1 — there is no defined path through the area, however in this case* $p^{\dagger} = p^{*}$. *Much of the area has been surveyed, with* $P_S = 67.19\%$.

cost of all edges leading into v_i are equal to:

$$c(v_j, v_i) = M(v_i), \forall v_j \in \text{PRED}(v_i). \tag{12}$$

This is the *true* cost of the edge and is only known after the vertex has been surveyed by one of the vehicles. Before any surveying has taken place, the *assumed* cost is equal to γ, which in this case has been set to $\gamma = 3$. Twenty random areas were generated and the technique applied using $n_{\text{AUV}} = 2$ vehicles. The mean and standard deviations of the performance metrics for these samples are reported in Table 1. The time taken for the proposed cooperative pathfinding method is on average 39.61 hours (versus 54.57 hours for the full survey), with an average competitive ratio very near unity. So for 73% of the time required to perform the full survey, one is able to find a path through the area that is essentially the same. An example of the area and path and resulting survey are given in Figure 3.

These results depend heavily on the choice of the assumed edge costs γ. A choice of γ that is too high will result in the agents taking any path that quickly gets them to the destination and making them unwilling to explore area in search of a better path. Conversely, a choice of γ that is too low will result in the agents being very much willing to spend time surveying unexplored areas in search of a better path. Here, the choice of $\gamma = 3$ reflects the value chosen in the next set of experiments, where a clear path does exists and must be found, for easier comparative purposes. Intuitively, γ must represent some value that signifies how much one is willing to tolerate a suboptimal path, or some prior knowledge about the best or average possible value of cost of the path. For the present example using uniformly distributed numbers everywhere, while the value of γ will definitely affect the amount of the area that is surveyed (and the corresponding mission time) it will, statistically, not have an effect on the cost of the resulting path $c(p^{\dagger})$ since on average this will be the same as there is no clear path through this area. The next set of examples present a clear path through the area that can be found by the proposed strategy. The mean value of $\rho_d = 1.279$ means that the direct path from start to goal had roughly 27.9% more targets on it, so that it was definitely possible to find a better path even in this setup.

3.2 Example 2 - Clear path with non-overlapping distributions

The second set of experiments is derived from the first one, except that now there will be a clear path from v_s to v_g where the density of objects follows a distribution that has a lower mean. The maximum number of objects that can be present in any cell on the path is lower than the minimum number of objects that be in any cell *not* on the path, *i.e.* the distributions do not overlap. The data is generated by first creating a random path from the start vertex to the goal vertex using a random walk in all possible directions of variable step lengths.

	mean μ	std deviation σ
Mission Time (T in hours)	39.61	7.35
Time transiting (v_1, v_2)	5.13 , 5.22	1.35, 1.36
Time surveying (v_1, v_2)	34.2, 32.98	5.90, 5.83
Percent Area Surveyed (P_S)	61.555	10.64
Competitive Ratio (ρ)	1.001	0.002
Direct Path Ratio (ρ_d)	1.279	0.138

Table 1: *Performance of the first experiment for $n_{\mathrm{AUV}} = 2$ cooperating survey vehicles for 20 random samples.*

The vertices along this path are randomly assigned a number of targets that is drawn from $\mathcal{U}(m^p_{\min}, m^p_{\max})$ and those not on the random path are drawn from $\mathcal{U}(m^q_{\min}, m^q_{\max})$, and where $m^p_{\max} < m^q_{\min}$. Here, the edges on the path are drawn from the interval $[1,\ldots,3]$ and those not on the path are drawn from $[4,\ldots,10]$. Note that the random walk method for generating the path also generates dead-ends and false leads. An example using two vehicles is shown in Figure 4.

Table 2 shows the results using 1, 2 and 4 vehicles. Now, there is a clear path to find that has significantly less objects and thus a significantly lower total cost. The competitive ratio is nearly unity meaning that this optimal path was indeed found the vast majority of the time; however, the time required to find the path is less than half that of surveying the entire area (21.70 versus 54.57 hours in the case of two vehicles). It is also worth noting that even with a single vehicle performing an adaptive survey, it is possible to reduce significantly on the time required to find an optimal or near-optimal path (42.03 hours versus 109.14 hours to survey the entire area). The direct line competitive ratio ρ_d is now roughly 1.71, meaning a significant improvement of 71% in the cost of the path has been obtained. The value of $m^p_{\max} = 3$ provides an easy method for choosing $\gamma = 3$, since any value above it cannot be on the path, as the distributions are non-overlapping. It is also noted that the mission time and area surveyed appears to scale linearly with the number of vehicles, meaning that the vehicles do not waste an excessive amount of time de-conflicting and transiting, and thus are used efficiently.

3.3 Example 3 - Clear path with overlapping distributions

The final example consists of the same general setup that was used in example 2, meaning that there is a clearly defined but random path from the start vertex to the goal vertex. However, instead of having two non-overlapping distributions for the vertices on and off the path, now those two distributions will overlap somewhat, but the mean number of

	$n_{\text{AUV}} = 1$		$n_{\text{AUV}} = 2$		$n_{\text{AUV}} = 4$	
	μ	σ	μ	σ	μ	σ
Mission Time (T in hours)	42.03	10.93	21.70	3.40	12.51	2.34
Time transiting (avg all v)	3.49	1.20	1.93	0.41	1.49	0.37
Time surveying (avg all v)	38.54	9.74	19.34	3.03	10.21	1.98
Percent Area Surveyed (P_S)	35.31	8.93	35.44	5.52	37.42	6.89
Competitive Ratio (ρ)	1.0076	0.010	1.0072	0.011	1.008	0.012
Direct Path Ratio (ρ_d)	1.707	0.214	1.707	0.214	1.707	0.214

Table 2: *Performance of the first experiment for varying number of vehicles for 20 random samples.*

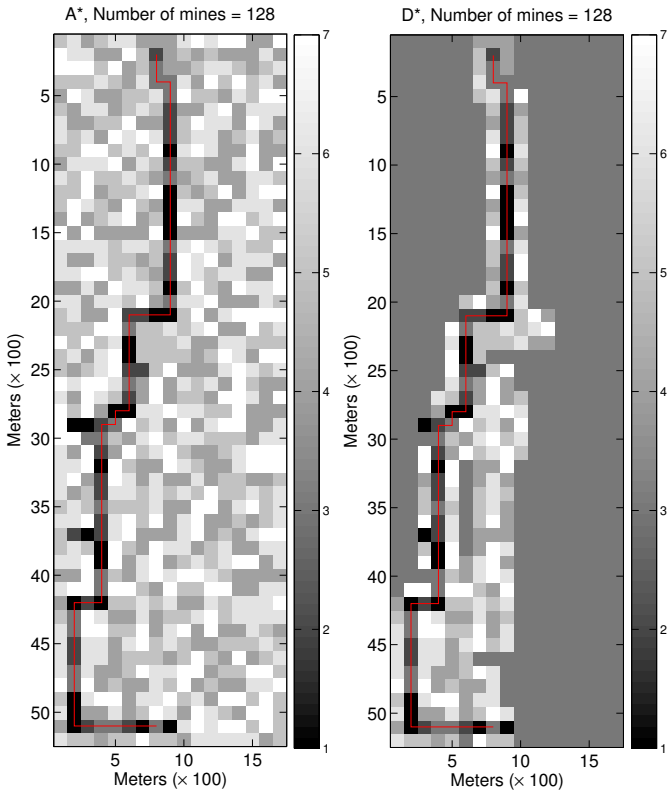

Figure 4: *Terrain from Experiment 2— there is a clear path which was found ($\rho = 1$) and $P_S = 33.03\%$.*

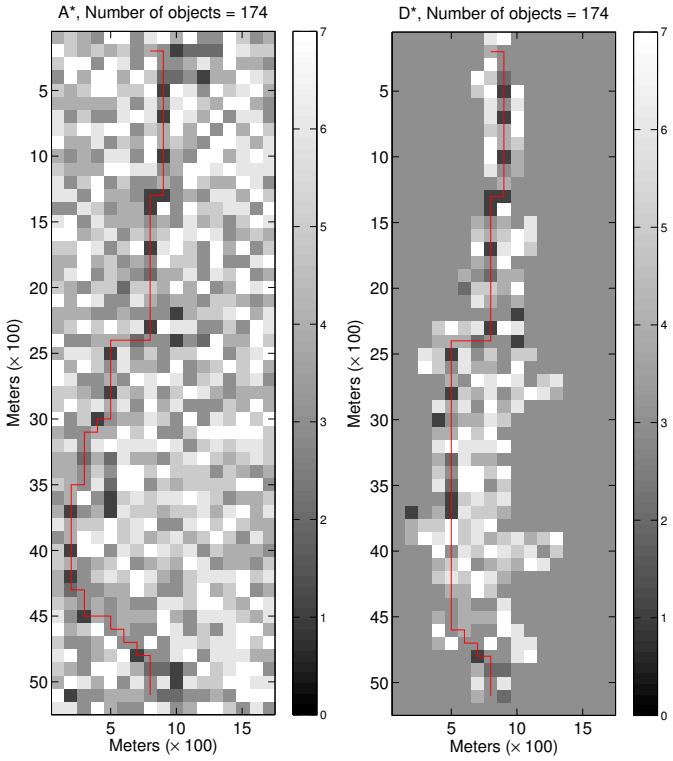

Figure 5: *Terrain from Experiment 3— the path is more difficult to find, however* $\rho = 1$ *and* $P_S = 32.58\%$.

objects for the vertices that are on the path will be lower than the ones off the path, and is accomplished by allowing $m^p_{\max} \geq m^q_{\min}$, specifically $m^p_{\max} = 5$ and $m^q_{\min} = 3$. As shown in Figure 5, the path is no longer as clear and becomes lost in the background. In the previous example, since the path was connected it could be considered easier to follow the path from start to finish by, for instance, considering any vertex v with $M(v) > m^p_{\max}$ to be effectively infinity. This is no longer the case in this example.

Results are shown in Table 3. In spite of the more challenging circumstances, the search strategy is still able to achieve a competitive ratio of near unity, with P_S less than half of the total search area. The value of ρ_d shows an improvement of over 32% in the total number of mine-like objects versus the straight line path. Again, mission time scales linearly with the number of vehicles, as was the case in Example 2. The value of $\gamma = 3$ was also used in these simulations.

	$n_{\text{AUV}} = 1$		$n_{\text{AUV}} = 2$		$n_{\text{AUV}} = 4$	
	μ	σ	μ	σ	μ	σ
Mission Time (T in hours)	48.974	16.549	24.53	7.17	14.06	3.74
Time transiting (avg all v)	4.529	2.107	2.57	1.18	1.80	0.61
Time surveying (avg all v)	44.44	14.48	21.49	5.81	11.41	2.93
Percent Area Surveyed (P_S)	40.72	13.27	39.39	10.71	41.83	10.64
Competitive Ratio (ρ)	1.001	0.005	1.005	0.018	1.001	0.003
Direct Path Ratio (ρ_d)	1.327	0.132	1.327	0.132	1.327	0.132

Table 3: *Performance of the third experiment for varying number of vehicles for 20 random samples.*

4 Discussion and Future Work

This report has extended single agent pathfinding algorithms for dynamic environments to the case with multiple collaborating agents. This was applied to a mine countermeasures (MCM) task of finding a channel through a larger area that contains the minimum total number of mine-like objects. It was shown that it is indeed possible to find paths with the same or nearly the same total cost as that which could be found by surveying the entire area, without having to do so. In some cases, such as the one where a clearly defined path exists, the path can be found by surveying only roughly 40% of the entire survey area. This result applied equally to the single vehicle and multiple vehicle case.

The methodology presented here provides some initial results as a proof of concept, however many improvements could be imagined, especially when considering a practical implementation. Notably, communications requirements, while not especially onerous in the case of bandwidth — a position and number of targets found in a small area to be transmitted occasionally is generally considered to be within the bandwidth capacity of today's underwater acoustic modems — could be however problematic in terms of transmit ranges and scheduling. In the multi-vehicle version of D* (Algorithm 5) one could choose the next vertex could be further optimized to keep in mind acoustic Tx/Rx constraints. In fact the present form of this algorithm is greedy and does not look ahead more than one step. This could lead to situations where, for instance, a vehicle must transit a large distance from an one region of the area of the current path whose cost has suddenly become very high. One possibility could be to include a look-ahead function for the vehicles, where the group can plan two or more time steps in the future as to where they will likely be, as was done by Chitre [17] for a beacon vehicle that positions other vehicles. Such a scheme would require some stability in the estimated path, so as to be able to plan ahead, and some study on how quickly the best estimated path is changing from one time step to the next would be needed.

Communications requirements can also be potentially reduced by adopting a decentralized method versus the proposed centralized method. If a vehicle performs the dynamic path planning itself and only relies on occasional updates from the other vehicles, then the amount of data that needs to be transmitted could be minimized. For instance, a technique where two vehicles each try to compute a lowest cost path to the other's current position would require only the occasional exchange of positioning information. However, in this case, one would expect that the decrease of situational awareness and information would result in a decrease in performance. Quantifying this decrease is essential.

In short, the (cooperative) D* method presented here only allows one to more quickly be able to find a path between two vertices given the information available. While this is a fundamental piece of the overall system, future research should focus on the strategies that the agents or vehicles use to explore that potential path, coordinate their search, and satisfy any constraints placed upon their formation.

The results shown here suggest a new way of thinking about how MCM missions are planned. When the Tasking Authority (TA) needs a sea lane from one area to another, they will likely place a suitable channel (probably in a straight line), from the one convenient location at the edge of one area to the next[5]. The result could be a channel that is unnecessarily risky or in which it is particularly difficult and time consuming to hunt for mines. However, if the TA prefers to spend effort searching for a safer channel, then it would require a complete search of a large area to then be able to pick a low risk channel through the area. What has been shown here is that it is possible to find a channel through the area that is optimal or nearly optimal, while only requiring a survey of a fraction of the area.

[5]Notwithstanding other factors that come into play when making such a decision, such as *e.g.* channel depth, or likelihood of anti-AUV barriers or fire from the shore.

References

[1] Percival, A. M. and Stoddard, M. (2010), Mine Countermeasure Planning Aid (MCMPA): A New Integrated Decision Support System for Planning and Evaluating Mine Countermeasure Operations, In *OCEANS 2010 Conference Proceedings*.

[2] Hart, P. E., Nilsson, N., and Raphael, B. (1968), A Formal Basis for the Heuristic Determination of Minimum Cost Paths, *IEEE Transactions on Systems Science and Cybernetics*, 2, 100–107.

[3] Stentz, A. (1994), Optimal and Efficient Path Planning for Partially-Known Environments, In *Proceedings of the International Conference on Robotics and Automation*, pp. 3310–3317.

[4] Stentz, A. (1995), The Focussed D* Algorithm for Real-Time Replanning, In *Proceedings of the International Joint Conference on Artificial Intelligence*, pp. 1652–1659.

[5] Koenig, S., Likhachev, M., and Furcy, D. (2004), Lifelong Planning A*, *Artificial Intelligence Journal*, 155, 93–146.

[6] Koenig, S. and Likhachev, M. (2005), Fast Replanning for Navigation in Unknown Terrain, *IEEE Transactions on Robotics*, 21(3), 354–363.

[7] Sun, X., Yeoh, W., and Koenig, S. (2010), Moving Target D^* Lite, In *Proc. of 9th International Conf. on Autonomous Agents and Multiagent Systems (AAMAS)*.

[8] Choset, H. and Pignon, P. (1997), Coverage path planning: The boustrophedon decomposition, In *Proceedings of the Int'l Conference on Field and Service Robotics*.

[9] Yamauchi, B. (1997), A Frontier-Based Approach for Autonomous Exploration, In *IEEE conference on Computational Intelligence in Robotics and Automation*.

[10] Rekleitis, I., New, A. P., Rankin, E. S., and Choset, H. (2008), Efficient Boustrophedon Multi-Robot Coverage: an algorithmic approach, *Annals of Mathematics and Artificial Intelligence*, 52, 109–142.

[11] Yamauchi, B. (1998), Frontier-Based Exploration Using Multiple Robots, In *Autonomous Agents 98*, pp. 48–53.

[12] Pêtrès, C., Pailhas, Y., Patrón, P., Petillot, Y., Evans, J., and Lane, D. (2007), Path Planning for Autonomous Underwater Vehicles, *IEEE Transactions on Robotics*, 23(2), 331–341.

[13] Cormen, T. H., Leiserson, C. E., and Rivest, R. L. (1990), Introduction to Algorithms, Ch. 25, pp. 514–549, MIT Press.

[14] Koenig, S. and Likhachev, M. (2001), Incremental A*, In *Neural Information Processing Systems (NIPS) Conference Proceedings*.

[15] Ocean Server Technologies Ltd. (2011). http://www.ocean-server.com/.

[16] Marine Sonic Technologies Ltd. (2011). http://www.marinesonic.us/.

[17] Chitre, M. (2010), Path Planning for Cooperation Underwater Range-Only Navigation using a Single Beacon, In *International Conference on Autonomous and Intelligent Systems*.

This page intentionally left blank.

DOCUMENT CONTROL DATA

(Security markings for the title, abstract and indexing annotation must be entered when the document is Classified or Designated.)

1. ORIGINATOR (The name and address of the organization preparing the document. Organizations for whom the document was prepared, e.g. Centre sponsoring a contractor's report, or tasking agency, are entered in section 8.)

Defence Research and Development Canada – Atlantic
PO Box 1012, Dartmouth NS B2Y 3Z7, Canada

2a. SECURITY MARKING (Overall security marking of the document, including supplemental markings if applicable.)

UNCLASSIFIED

2b. CONTROLLED GOODS

(NON-CONTROLLED GOODS)
DMC A
REVIEW: GCEC APRIL 2011

3. TITLE (The complete document title as indicated on the title page. Its classification should be indicated by the appropriate abbreviation (S, C or U) in parentheses after the title.)

Cooperative dynamic pathfinding for multiple autonomous underwater vehicles using D*

4. AUTHORS (Last name, followed by initials – ranks, titles, etc. not to be used.)

Myers, V.

5. DATE OF PUBLICATION (Month and year of publication of document.)	6a. NO. OF PAGES (Total containing information. Include Annexes, Appendices, etc.)	6b. NO. OF REFS (Total cited in document.)
January 2011	40	17

7. DESCRIPTIVE NOTES (The category of the document, e.g. technical report, technical note or memorandum. If appropriate, enter the type of report, e.g. interim, progress, summary, annual or final. Give the inclusive dates when a specific reporting period is covered.)

Technical Report

8. SPONSORING ACTIVITY (The name of the department project office or laboratory sponsoring the research and development – include address.)

Defence Research and Development Canada – Atlantic
PO Box 1012, Dartmouth NS B2Y 3Z7, Canada

9a. PROJECT OR GRANT NO. (If appropriate, the applicable research and development project or grant number under which the document was written. Please specify whether project or grant.)	9b. CONTRACT NO. (If appropriate, the applicable number under which the document was written.)
11cf	

10a. ORIGINATOR'S DOCUMENT NUMBER (The official document number by which the document is identified by the originating activity. This number must be unique to this document.)	10b. OTHER DOCUMENT NO(s). (Any other numbers which may be assigned this document either by the originator or by the sponsor.)
DRDC Atlantic TM 2011-044	

11. DOCUMENT AVAILABILITY (Any limitations on further dissemination of the document, other than those imposed by security classification.)

(X) Unlimited distribution
( ) Defence departments and defence contractors; further distribution only as approved
( ) Defence departments and Canadian defence contractors; further distribution only as approved
( ) Government departments and agencies; further distribution only as approved
( ) Defence departments; further distribution only as approved
( ) Other (please specify):

12. DOCUMENT ANNOUNCEMENT (Any limitation to the bibliographic announcement of this document. This will normally correspond to the Document Availability (11). However, where further distribution (beyond the audience specified in (11)) is possible, a wider announcement audience may be selected.)

Unlimited

13. ABSTRACT (A brief and factual summary of the document. It may also appear elsewhere in the body of the document itself. It is highly desirable that the abstract of classified documents be unclassified. Each paragraph of the abstract shall begin with an indication of the security classification of the information in the paragraph (unless the document itself is unclassified) represented as (S), (C), or (U). It is not necessary to include here abstracts in both official languages unless the text is bilingual.)

A method for finding least-cost paths through an area by collaborating agents is presented. The method relies on extending existing dynamic path planning algorithms, namely D*, to the multi-agent case. The search tree is centrally updated and then used by the agents to cooperatively decide upon which areas to survey, the results of which are then used to update the current best path. Several numerical examples are given where it is shown that, on average, a competitive ratio — the ratio between the cost of the path found by dynamic algorithm and one operating with full knowledge of the area — of nearly one can be achieved while requiring a survey of less than half the total area. It is also shown that the total mission time scales linearly with the number of vehicles, meaning little to no effort wasted on vehicle coordination. In the context of mine countermeasures (MCM) operations with autonomous underwater vehicles (AUVs), the results indicate that a channel through a potentially mined area representing an amount of risk nearly equal to the optimal path can be found in significantly less time than the time required to survey the entire area.

14. KEYWORDS, DESCRIPTORS or IDENTIFIERS (Technically meaningful terms or short phrases that characterize a document and could be helpful in cataloguing the document. They should be selected so that no security classification is required. Identifiers, such as equipment model designation, trade name, military project code name, geographic location may also be included. If possible keywords should be selected from a published thesaurus. e.g. Thesaurus of Engineering and Scientific Terms (TEST) and that thesaurus identified. If it is not possible to select indexing terms which are Unclassified, the classification of each should be indicated as with the title.)

Autonomy
AUV
Automatic Target Recognition
MCM