Distributed Multi-AUV Coordination in Naval Mine Countermeasure Missions

Sanem Sariel\textsuperscript{1}, Tucker Balch\textsuperscript{2} and Jason Stack\textsuperscript{3}

\textsuperscript{1}Istanbul Technical University, Department of Computer Engineering, Istanbul, 34496, TURKEY
sariel@cs.itu.edu.tr

\textsuperscript{2}Georgia Institute of Technology, College of Computing Department, Atlanta, GA, 30332, USA
tucker.balch@cc.gatech.edu

\textsuperscript{3}Naval Surface Warfare Center, Panama City, FL, 32407 USA
Jason.stack@navy.mil

Summary. In this paper, we propose a general framework, DEMIR-CF, for a multi-robot team to achieve a complex mission including inter-related tasks that require diverse capabilities and/or simultaneous executions. Our framework integrates a distributed task allocation scheme, cooperation mechanisms and precaution routines for multi-robot team execution. We apply our distributed coordination framework, DEMIR-CF, to Naval Mine Countermeasure missions on the Navy’s ALWSE-MC simulator. Marine applications provide additional challenges such as noisy communication, position uncertainty and the likelihood of robot failures. There is a high probability that the initial assignments are subject to change during run time in these kinds of environments. Our framework ensures robust execution and effective completion of missions against several different types of failures. Preliminary results for MCM missions are promising in the sense of mission completion, and AUV paths are close to optimal in the presence of uncertainties. In this work, we evaluate qualitative performance of our framework on a realistic simulator for handling different contingencies that may arise run time.

1 Introduction

In this paper, we present a generic framework, Distributed and Efficient MultiRobot - Cooperation Framework (DEMIR-CF), designed for effective mission achievement of a multi-robot team for inter-related tasks that require diverse capabilities and simultaneous executions. Since real world applications present additional challenges than software platforms, robustness is a key issue of a multi-robot coordination framework. DEMIR-CF with its integrated structure can respond to several real time contingencies while dynamically maintain high solution
quality. In this paper, we present the generic architecture of our framework even suitable for complex mission execution in environments with failure potentialities and limited communication features (such as bandwidth limitations, limited ranges, or unexpected delays). We propose a distributed task allocation scheme integrated with dynamic task switching features to effectively respond to several contingencies while effectively achieving the tasks. Therefore the system is robust to failures and communication limitations. We report the experimental results and several scenarios in the context of Naval Mine Countermeasures mission for multi-AUV coordination.

2 Background and Related Work

Multi robot coordination has been an active and attractive field for robotic researchers during the last decade because of the demand for multiple robot, UAV, UGV or rover missions, especially in military and space applications. Among different approaches, the centralized approach is not robust especially when communication is limited between operator and individual robots, and failures are highly probable. Therefore our focus is on distributed coordination frameworks in this work, and we will review literature on this subject. Parker presents one of the earlier works for distributed multi-robot task allocation, ALLIANCE, with a behaviour based framework [8] for instantaneous task assignment. M+ [1] is a distributed task allocation and achievement scheme for multi-robot cooperation addressing many real time issues including plan merging paradigms. MURDOCH [4] is a framework achieving publisher/subscriber type allocation for instantaneous assignment. Dias et al. proposes a combinatorial auction based task allocation scheme: TraderBots [2]. Lemarie et al. proposes a task allocation scheme for multi-UAV cooperation with balanced workloads of robots [7]. Task allocation capabilities in the face of environmental changes and optimality analysis are investigated separately on earlier systems given above. According to [3], existing market mechanisms are not fully capable of re-planning task distributions, re-decomposing tasks, rescheduling commitments, and re-planning coordination during execution. We would like to fill these gaps by our integrated cooperation framework. Our primary contribution in this work is the presentation of an integrated cooperation framework for multi robot task execution and the extensive design, use and analysis of precaution routines and solution quality maintenance schemes for single-item auctions in real time task execution. Tasks in our mission definitions may require simultaneous executions and precedence relations among each other. With an effective bid evaluation approach for the selected domain, DEMIR-CF generates near optimal solutions [10]. From our point of view, task allocation, execution and contingency handling should be integrated into the cooperation framework without assuming they are achieved separately, if globally efficient solutions are desired. This is the main rationale behind our framework.
3 Distributed and Efficient Multi Robot - Coop Framework

DEMIR-CF is designed for complex missions including inter-related tasks that require diverse agent capabilities and simultaneous execution. The framework combines distributed task allocation scheme and coalition formation procedures, and dynamic task selection mechanism as cooperation components and Plan B precaution schemes some of which are implemented by dynamic coalition maintenance/task switching procedures. These components are integrated into one framework to provide an overall system that finds near optimal solutions for real time task execution.

The overall objective of the robot team \( r_j \in R, 0 \leq j < ||R|| \) equipped with our framework is to achieve a mission \( M \) consisting of interrelated tasks \( T_i \) \( (0 \leq i < ||M||) \), by incremental assignment of all \( T_i \) to \( r_j \in R \) while optimizing the specified objective function. Coalitions \( (C_i) \) \([11]\) are formed to meet requirements of simultaneous executions of tasks synchronously by a homogeneous group of robots. Sizes of coalitions vary according to the required minimum number of robots \( (reqno_i) \) to execute the tasks. An example of such a task may be pushing a heavy object requiring more than one robot. The capabilities \( (cap_j) \) of robots \( r_j \) in a coalition should be a superset of the required capability set for the task \( T_i \) \( (reqcap_i) \). Tasks are preemptive: the activity of task execution can be split during runtime if another advantageous situation arises or environmental conditions impel.

**Definition 1 (candidate task and suitable robot).** \( T_i \) is a candidate task for robot \( R_j \) if the \( reqcap_i \) is a superset of \( cap_j \) and precedence constraints of the task are satisfied; \( R_j \) is a suitable robot for task \( T_i \).

**Definition 2 (executable task).** \( T_i \) is an executable task, if at least \( reqno_i \) number of robots can be assigned for execution.

![Fig. 1. DEMIR-CF modules](image-url)
For the robots to be prepared for the contingencies, models of the system tasks and robots are kept in each robot’s world knowledge. Each robot keeps track of the statuses of them in corresponding FSMs. Task states are: free, auctioned, being executed, achieved, uncertain (interpreted as state “free”) and invalid. Robot states are: idle, executing, failed and auctioneer. The state transitions of FSMs are activated by either own motivations or incoming information from other robots. Model Update module is responsible for updating models. All modules in the framework and information flow among them are given in Fig. 1. Model Update, System Consistency and Dynamic Task Selection/Switching modules perform Plan B precaution routines. Allocation scheme ensures distributed task allocation. Coalition module implements synchronized task execution and coalition maintenance schemes. A sample flow of the operations in the framework is summarized:

1. Mission task definitions are given to the robots (time-extended representation of tasks with precedence constrains to achieve overall mission).
2. Each robot selects the most suitable candidate task to execute by global cost consideration among mission tasks (dynamic task selection/switching).
3. Corresponding robots offer auctions for the selected tasks. In auctions, inconsistencies and conflicts are resolved.
4. Coalitions are formed for the announced tasks making sure that each robot is in the most suitable coalition from global solution quality point of view.

During task execution, simultaneously, dynamic task selecting/switching mechanism ensures to switch between tasks, if it is profitable, and then corresponding auction and coalition formation procedures are applied. Real time situations in which task switching is necessary are given in the next section.

3.1 Real Time Issues and Requirements

Since the world is beyond the control of the robots and change continuously in real world applications, the difficulty of multi robot task execution problem goes beyond the task allocation problem. In particular, multi-robot systems deal with difficulties arising from noisy sensor information, unexpected outcomes of actions, environmental limitations (especially in communication) and presence of failures of hardware. All these factors may affect the overall solution. We list evolving circumstances that may change the solution as:

1. Own failure detection: Robots detect their own failure.
2. Failure detection of another robot: Robots detect another robot’s failure.
3. Change in the estimated task execution cost/time: Environmental dynamics, uncertain knowledge, or hardware problems may cause delays on task execution or early achievements of tasks. Uncertain sensor and/or localization information may also result in incorrect estimations.
4. Change in the task definitions: Task dependencies, priorities, or the overall objective (goal) may change. Some tasks may become invalid during runtime.
5. New online tasks: New tasks may be given by human operators or new tasks may be discovered by robots themselves.
6. New robots: New robots may be released, or some failed robots may be repaired or may recover from trap like threats.

7. Intervention and manual changes on assignments by external agents.

   Some of these situations may arise after either internal or external events. Given these contingencies, even solution of an approach capable of finding optimal solutions may become sub-optimal under uncertainties of real world applications. Verification of the solution optimality is a difficult issue for real world applications. Therefore in the last decade researchers proposed effective approaches, opportunistic methods without giving boundaries on the overall solution quality except [6]. However, their work assumes perfect communication and contingencies are not considered in these boundaries. For now, these boundaries are given under the assumption that the information available to the robot is complete.

   We designed DEMIR-CF as being capable of dealing with the situations presented above. The framework can effectively respond to these events and solution quality is maintained simultaneously with real time task execution.

3.2 Task Representation

Tasks are represented by a data structure containing information regarding the task execution requirements and the task status. Tasks are represented as septuples $<$id, type, reqcap, dep_list, reqno, rel_info, prec_info$>$. Some task ids are generated initially before mission execution (system generated). These ids are common for all robots. However online task ids may be different for each robot. Robots agree on task types (type) and corresponding execution methods before mission execution. Requirements (reqcap) define special sensors and capabilities required to execute the task. Dependencies (dep_list) are represented with hard and soft dependent task ids. We define two types of dependencies for representing precedence relations. Hard dependency implies sequential execution while soft dependency allows parallel execution [9]. Minimum number of robots to execute the task (reqno) is determined. Related information (rel_info) represents information regarding the type. Latest location, target location are such examples. Precaution information (prec_info) is used for contingency handling. These are: task state, estimated task achievement time and current execution cost.

   The mission is defined as an acyclic complete graph (not necessarily a connected graph) of inter-related tasks connected by arcs representing dependencies. An example graph representation for the object construction mission can be found in [9]. Task definitions can be changed during execution. In particular, rel_info, prec_info and reqno are subject to change during execution.

3.3 Distributed Task Allocation Scheme

Task allocation and initial assignments may be carried out by using operations research methods. However, our research addresses issues of real time execution
when managing the overall team by a central authority is not possible due to several real world limitations. Auction based task allocation approach is suitable to provide a scalable and efficient way of distributing tasks. We implement Contract Net Protocol (CNP) [12] to select task executers. Although CNP presents the formalism on relationships between managers and contractors, it does not present details for the following questions: When should task announcements be made? How should bid values be defined to get globally effective (or close to optimal) solutions? Which subset (or all) of the already allocated tasks should be re-auctioned to maintain solution quality? When should reallocations be implemented and who decides on them? Most auction based task allocation schemes offer solutions for allocating one/subset of tasks of the overall mission. However there is usually little information about when task announcements and reassignments are made. In our framework, any robot becomes an auctioneer when it intents to execute a task. Robots select one of the best suitable candidate tasks among mission tasks by the dynamic task selection scheme details of which are explained in the next section. Basically, auction announcements are ways to illustrate intentions to execute tasks for which $reqno = 1$ or to select members of coalitions to execute tasks for which $reqno > 1$. Therefore, if more than one robot intends to execute the same task, more suitable one(s) is selected in the auction by considering cost values. Single items are auctioned and allocated in auctions. Auction negotiation implemented in the framework consists of standard steps to clear an auction. Robots can get the necessary task details from the auction offers and then check the validity of the auction. If the auction is invalid, related precaution routines are activated. Otherwise, the candidate robot sends its cost value as a bid. The other candidate robots behave simultaneously as well. If the auctioneer cannot get the required number ($reqno$) of bids (also counting in own bid) from the other robots until the predefined deadline, it cancels the auction. Otherwise it ranks all bids and assigns the best suitable robot with the lowest cost value to the executable task (if $reqno = 1$), or suitable coalition members (if $reqno > 1$). The framework allows multiple auctions and winners for different tasks at the same time.

Considerations regarding coalition formation

Since robots may be suitable to participate in different coalitions, common values of coalitions should be distributed among coalition members to improve solution quality [11]. We avoid using complicated approaches to form coalitions. Instead we are working on simple but effective approaches to determine coalition values embedded in the bid evaluation. We are currently analysing the design of simple approaches to evaluate coalitions and provide ways to put each robot in its most suitable coalition among different alternatives from global perspective.

3.4 Dynamic Task Selection Scheme and Online Scheduling

Dynamic task selection mechanism is used by robots to switch between tasks whenever their world knowledge is updated or there are online tasks. Therefore issues related to both online scheduling and scheduling under uncertainty addressed.
Dynamic task selection is implemented by forming a priority queue of unachieved candidate tasks. Priority queue is sorted by costs for executing these tasks. If costs are the same, the priorities are considered in the given order: Robot’s current task (if any), tasks already being executed, tasks awarded in auctions, and free tasks.

**Coalition Maintenance and Dynamic Task Switching Schemes**

In the framework, instead of using complicated re-allocation procedures, we propose incremental selection and task switching schemes for behaving myopically while thinking globally using bid evaluation heuristics. Provided with an effective bid evaluation heuristic, dynamic task selection mechanism of the framework ensures effective task selection and switching whenever it is profitable. Each robot, independent from executing a task or not, can offer another auction or select to execute a task already being executed by another robot with a worse cost value than that it will cost for itself. If task switching occurs with a coalition member, the corresponding coalition member is released from the coalition becoming a suitable robot for other tasks. When robots participate in coalitions, they are only allowed to select other tasks, when they are released from these coalitions.

**3.5 Bid/Cost Evaluation**

The impact of bid (cost) evaluation on the solution quality is inevitable for auction based systems, and research in this area desires more investigation. According to the taxonomy given in [5], multi robot task allocation problems are divided into two types of classes based on the mission description: instantaneous vs. time-extended. Most of the multi-robot architectures offer solutions for instantaneous assignments. We do not have an assumption on the mission description, and we offer an incremental allocation method of tasks with effective bidding strategies for both types of classes. Therefore global solution quality is maintained for the mission tasks from time-extended view of the problem by means of the bid considerations. However, the approach is capable itself to offer solutions for instantaneous changes on the task description. Therefore we classify our framework capable of addressing both types of problem classes. Unless effective bid evaluation strategies are designed, it is not possible to observe globally optimal solutions for NP-Hard problems, and additional adjustments are required to change allocations with an additional cost of communication as in combinatorial auctions. In our earlier work, we have shown that by effective bid evaluation approach, globally near optimal solutions can be observed for auction based approach. In that work, we analyse performance of three different heuristic cost functions combined with our framework for Multi-robot Travelling Salesman Problem [10]. Incremental assignment approach eliminates redundant considerations for environments in which the best solution is highly probable to change, and effective bidding strategies ensure solutions to be close to optimal with a time-extended view of the problem. Although we have shown that our approach can find near-optimal solutions for multi robot exploration problem, we still need further investigation on bidding strategies for different domains.
3.6 Models for Contingency Handling and Plan B Precautions

There are certain additional duties for the robots to keep system consistency:

- Robots broadcast known achieved tasks in predefined time periods to prevent redundant executions. This feature provides a bucket-brigade type of information sharing handling communication range limitations.
- Robots broadcast new discovered online tasks which are unachieved yet.
- Task execution messages containing the updated cost value and estimated task achievement time information are sent in predefined time periods as clues for the executer robot is still alive and the task is under execution.
- Task achievement message is sent when the task is achieved.
- Cancellation message is sent if task execution is cancelled.
- Task invalidation message is sent when invalidities are detected.

Table 1. Precautions for Contingencies and Conflicts

<table>
<thead>
<tr>
<th>Contingency or Conflict by consistencies</th>
<th>Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Any message from an unrecognized system robot</td>
<td>Robot model is created with the corresponding status derived from the message</td>
</tr>
<tr>
<td>Any message related to an unrecognized system task</td>
<td>Task is added to the task list with the corresponding status in its model</td>
</tr>
<tr>
<td>An already achieved task is announced as a new task/being executed/cancelled/auctioned</td>
<td>Warning message is sent to the sender</td>
</tr>
<tr>
<td>A task being executed/auctioned is announced as being executed/auctioned</td>
<td>While the robot with the minimum cost continues to the operation, the other robot cancels its operation</td>
</tr>
<tr>
<td>Cancellation message is received for a task already being executed by own</td>
<td>Nothing is done for the task, robot model is updated accordingly (setting “idle”)</td>
</tr>
<tr>
<td>If cancellation is message is received for a task being executed by the sender robot</td>
<td>Task and robot models are updated accordingly (task status is set as free) (robot status is set as “idle”)</td>
</tr>
</tbody>
</table>

Plan B Precaution routines embedded in the framework enable the system to dynamically respond to various failure modes and recover from them. Current implementation use explicit communication to detect conflicts and contingencies. However failures in communication can also be handled by precaution routines. (If robots can observe each other implicitly, model updates can be implemented in a similar manner.) Related to the contingent situations, appropriate precaution routines are activated to either correct the models, or initiate a recovery. Recovery operations may include warning other robots about the problem or changing the model accordingly. These inconsistencies usually arise when robots are not informed about tasks that are achieved, under execution or under auction. Precaution routines are given in Table 1. Most of the contingencies are detected by model updates (Table 2) and model checking (Table 3). One standard way of detection of robot failures is sending heart-beat signals. However in our framework, incoming messages from other robots are taken as clues for running properly. More complicated prediction models may be used for more accurate failure prediction. Some misleading beliefs such as setting status of a robot running properly as “failed” may cause parallel executions. This is a desired feature for the mission completion point of view. Designed precautions resolve these kinds of inconsistencies if
communication resources permit in later steps. In the design of precautions, it is assumed that robots are trusted and benevolent.

**Table 2. Model Updates related to the messages**

<table>
<thead>
<tr>
<th>Message Type</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Any type</td>
<td>Last communication with the robot and for the task is registered</td>
</tr>
<tr>
<td>&quot;achieved&quot; - valid</td>
<td>The robot status is set as “idle”. Task model is set as “achieved”. If the task is in consideration (in schedule or in execution) consideration is cancelled</td>
</tr>
<tr>
<td>&quot;execution&quot; - valid</td>
<td>If there are other tasks of which statuses are set as “being executed by this robot”, statuses are changed as “uncertain”</td>
</tr>
</tbody>
</table>

**Table 3. Model Checking for Tasks and System Robots**

<table>
<thead>
<tr>
<th>Status</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time difference from latest communication with a robot is more than the threshold</td>
<td>Robot status is set as “failed”</td>
</tr>
<tr>
<td>Task in execution is not achieved although the estimated deadline is reached</td>
<td>Related task statuses to that robot is set as “uncertain”</td>
</tr>
<tr>
<td>Task status is “auctioned” longer than pre-defined time period</td>
<td>Task status is set as “uncertain”. (If there isn’t a failure, robots would either get award message or execution message from at least one of the robots).</td>
</tr>
</tbody>
</table>

4 A Case study: Naval Mine Countermeasures Mission

Naval mine countermeasures (MCM) are actions taken to counter the effectiveness of underwater mines. MCM operations include finding and seizing mine stockpiles before they are deployed, sweeping desired operational areas, identifying mined areas to be avoided, and locating and neutralizing individual mines [13]. Our research is focused on the subset of MCM that involves locating and mapping all individual mines in an operational area. In general, recognizing proud mines on the seafloor is not overly difficult; the difficulty arises with the abundance of non-mine objects on the seafloor that possess mine-like characteristics (e.g., geologic outcroppings, coral, manmade debris, etc.). This ample supply of false alarms has necessitated the following strategy typically employed by the Navy: detect and classify the mine-like objects (MLOs) with high-coverage rate sensors (e.g., side-looking sonar), employ advanced signal processing techniques for maximal false alarm reduction, then revisit the remaining MLOs with identification-quality assets (e.g., electro-optic sensors) to confirm them as mines or dismiss them as false alarms. It is this strategy which the research proposed herein attempts to implement in a distributed, near optimal fashion. Achieving this mission with an AUV team requires effective task allocation mechanisms and several precautions.

The reference mission in this research is to detect, classify, and identify underwater mines in a given operational area simulated in a PC-based software, ALWSE-MC[14], analysis package designed to simulate multiple autonomous vehicles performing missions in the littoral regions including mine reconnaissance, mapping, surveillance, and clearance. This mission employs two types of vehicles:
unmanned underwater vehicles (UUVs) which are free swimming AUVs and possess large-footprint sensors (e.g., UUV cap list has side-scan sonar representation) for detection and classification (D/C) of mines and seafloor crawlers equipped with short-range, identification-quality sensors (e.g., crawler cap list has camera representation). The crawlers have the ability to stop at an object and take a picture of it with a camera. Therefore two types of tasks are defined for vehicles: waypoint visiting (w) and MLO identification (tmlo). In the task representation, reqcapw contains side-scan sonar and reqcaptmlo contains cameras besides the standard capabilities of AUVs common in both types of vehicles.

In particular, in this case study, tasks do not have interdependencies. The coverage mission \( (M_c) \) contains predefined number of waypoints \( (w_i \, 0 \leq i < ||M_c||) \) to be visited by AUVs. One way of task representation is to directly assign tasks for each waypoint. However this representation has a drawback of high communication requirements (i.e. bandwidth) for effective completion of the mission. Instead we represent tasks as interest points of regions/search areas \( (W_j = \cup w_i, \forall w_i \text{ is unvisited, and } W_j \text{ s may overlap}) \). These regions (and corresponding centers) are defined by robots during runtime. The advantage of this representation is that the only information necessary to negotiate over tasks contains the centers of these regions (interest points). Regions/interest points determined by different UUVs may vary. However, the uncertainty related to the tasks is within an acceptable degree compared to the requirements of complete knowledge sharing. Before defining regions, relative distance values, \( \text{rel_dist}(r_j,w_i) \), are determined for each unvisited waypoint \( w_i \) as in Eq. 1. Then unvisited waypoints are sorted according to their \( \text{rel_dist}(r_j,w_i) \) values for each robot to define regions of unvisited waypoints, number of which is close to the proportion of all unvisited waypoints over the number of UUVs believed to be running properly. \( \text{dist} \) function returns the Euclidean distance between two points. For bid evaluations, we selected two different heuristic functions proved to provide close to optimal results for Multi-robot TSP problem[7]. These cost functions, explained in the next section, provide time extended consideration of tasks for instantaneous assignment with a tractable and efficient way.

\[
\text{dist}(r_j,w_i) = \infty \quad \text{if } r_j \text{ is believed to be failed}
\]
\[
\text{rel_dist}(r_j,w_i) = \text{dist}(r_j,w_i) - \min_{w_i \neq w_j \text{ unvisited}} (\text{dist}(r_j,w_i))
\]  

(1)

4.1 Exploration for Detection and Classification of MLO Locations

To begin the mission, the UUVs survey the operational area following waypoints determined \textit{a priori}; however, corresponding regions containing waypoints may be reassigned by negotiations among UUVs autonomously. After determining regions/interest points, each UUV offers an auction for the best suitable region for itself and offers its selected interest point information as an auction. After negotiations on several auctions, each UUV is assigned to a region. If there is no communication at all, more than one UUV may select the same region (or overlapping re-
regions). UUVs evaluate bids(costs) for these regions. Cost value $c_{ji}$ calculated for AUV $r_j$ and interest point (of a region) $ti$ is given in Eq.2.

$$W_j = \cup w_i \text{ where } \arg\min_{r_j, \text{all } r_i \text{ not killed}}(dist(r_j, w_i)) = r_j,$$

$$w_{j1}, w_{j2} \in W_j \text{ and } dist(w_{j1}, w_{j2}) = \max(dist(w_j, w_j)), \quad w_j, w_j \in W_j$$

$$c_{ji} = \alpha \cdot dist(r_j, t_i) + (1 - \alpha) \cdot (\max(dist(t_i, W_j)) - \max(dist(t_i, W_{j1}), dist(t_i, W_{j2})))$$

This heuristic function considers boundary targets, $w_{j1}$ and $w_{j2}$ in $W_j$ which are the targets having the maximum $dist(w_{j1}, w_{j2})$ value. The basic idea of this function is that if an area is formed ($W_j$) containing all targets closer to $r_j$, these targets determine the diameter of the area and both of them should be visited. This heuristic method forwards robots to these farthest targets within their area to some degree. By introducing a constant ($\alpha$) this degree can be adjusted and it is taken as 2/3. This heuristic function produces close to optimal results for multi-robot TSP [10]. If there are more than one pair of boundary targets, the pair of which has a member with the smallest distance to the AUV is selected. After regions are determined, UUVs form their schedules by the waypoints in their selected region descending ordered by their cost values (Eq.(2)). Allocations of the regions may also change during run time to maintain solution quality. Whenever UUVs detect failures or recoveries from failures they change their region definitions accordingly and offer new auctions for the most suitable regions for themselves. In the beginning of the simulations, no AUV is informed about each other. Whenever they are informed about each other, they update their knowledge. Therefore their schedule lists are subject to change during first steps. In the beginning, all waypoints are in their schedules. If they are not informed about each other they will have one task (region) of visiting all known unvisited waypoints.

As UUVs detect the MLOs on their way, these estimated target positions are broadcasted to all AUVs within communication range (ie. tasks are generated for crawlers online). Then MLO information propagates (in bucket-brigade fashion) to all other UAVs in the group that can possibly be reached.

### 4.2 Identification of MLOs

When crawlers are informed about MLO locations, they update their world knowledge and dynamically select the best MLO target to visit and offer auctions. Therefore they can switch among tasks when new online tasks appear, if it is profitable. It is also possible that a crawler may inadvertently discover a mine without being informed of its position by a UUV. In this case, the crawler identifies the target, adds it to its task list as an achieved task, and broadcasts achievement information for maintaining system consistency. Crawlers determine their bid values by Eq. (3). This cost function provides a greedy look ahead for visiting MLO targets rather than only considering the distances between target and the AUV. In this function an additional penalty is applied to the cost, if there is another profitable alternative way of visiting tasks.
\[ c_p = \begin{cases} \text{dist}(r, t) + \text{dist}(r, t_f) - \text{dist}(r, t_i) & \text{if dist}(t, t_f) > \text{dist}(r, t_i) \\ \text{dist}(r, t) & \text{otherwise} \end{cases} \]  

(3)

In the identification task, when crawlers are within an area close to a MLO location, they begin keeping time. Whenever time limit is reached, they set the task status as “achieved” and broadcast this information. If there is detection during this time period, MLO location is considered as an actual mine and task achievement is directly applied, otherwise it is determined as a false alarm after deadline. In either case, the task is achieved.

A conceptual flowchart summarizing operations of UUVs, crawlers and the general operations implemented by both types of AUVs are given in Fig. 2.

---Plan B Precautions---

Resolve inconsistencies

---Dynamic Task Selecting/Switching---

System Model is updated:
Mission Execution Begins
failures, recovers, own inconsistencies, new tasks

---Distributed Task Allocation---

Select the most suitable MLO location

Define Regions
Select the most suitable region

Offer auction for the selected region

Visit waypoints in the assigned region

Award another UUV

Broadcast:
known unachieved MLO locations
visited waypoints
execution message for the next waypoint in the schedule

\{ UUV Operations \}

\{ General Operations \}

\{ Crawler Operations \}

Fig. 2. Conceptual Flowchart related to the AUV Operations

5 Experimental Results

Our framework’s mission completion performance and precaution routines are evaluated in ALWSE-MC. Three sample scenarios in the simulation are given to illustrate performance of our framework for Naval MCM missions. UUVs are equipped with sensors capable of detecting mines within 30 feet from skin of tar-
get. However they are not able to correctly identify them. Crawlers are equipped with cameras which can both detect and identify mines within 20 feet. None of the AUVs have certain search patterns, all waypoint assignments are implemented by auction based approach. UUVs have internal navigation errors therefore their estimated location values are different from actual locations in most cases. Two AUVs can communicate each other whenever the receiver AUV is in the sender AUV’s transmitter range, within its transmitter beam width, and sender AUV is within transmitter AUV’s receiver beam width. Beam patterns for transmitting point forward relative to the AUVs’ body and receive beam width is 170 degree in each side, source level 180 dB and frequency is 9.6 kHz.

All UUVs and crawlers begin execution from a deployment area. There is no a priori information about mine locations. Waypoints are known but are not assigned. UUVs begin negotiations and divide the overall mission area into three (known number of UUV vehicles) regions. Since they are in the same location, they can communicate their location information. Therefore initially defined regions are the same for all UUVs. Fig. 3 illustrates a successful mission scenario in which there are no AUV failures. Allocations of waypoints after negotiations can be seen in Fig. 3(b). Waypoint assignments do not change during run time. However crawlers sometimes switch among tasks if they are not informed about tasks that are being executed. And sometimes parallel executions occur. Whenever they are in communication range they can resolve this problem effectively by means of the precaution routines. As in Fig. 3(a), crawlers can also detect mines without being informed. Routes of the crawlers may seem somewhat random. However it should be noted that tasks appear online during run time because MLO locations are discovered online and communication ranges are limited.

![Fig. 3. Scenario 1. UUVs cover the area by visiting waypoints (a). Crawlers visit MLO locations as they are informed. (Deployment area is circled.) Each AUV is assigned to a region after auction based allocation of interest points (b).](image)
Initially all UUVs begin execution. UUV 3 fails, other UUVs take responsibility of all unvisited waypoints.

Search area assignments after failure for UUVs 1 and 2. Because of an uncertainty, one waypoint is left uncovered.

UUV 2 completes its search area coverage task, and adds the waypoint missing in (b) to its schedule after detecting, it is not visited.

Fig. 4. Scenario 2. UUV-3 fails during execution. Other UUVs handle its tasks.

In the second scenario, one of the AUVs fails on the same setting of scenario 1 (Fig. 4). Initial regions for all UUVs change after UUV 3 fails (Fig. 4(b)). Other UUVs change region definitions and, after negotiations, they share the full area as indicated in the figure. Visited waypoints are not in their region coverage. Because of the uncertainties, some waypoints are left uncovered in schedules. However this uncertainty related problem is resolved by UUV 2 and the mission is completed.

In the third scenario (Fig. 5), UUV 3 fails and other UUVs detect the failure and they negotiate over the remaining unvisited waypoints and new schedules are determined as in Fig. 5(b). While these UUVs executing their tasks, another UUV (4) is released from the deployment area. Detecting a new UUV arrival, other UUVs change their region definitions accordingly (Fig. 5(d)) and offer auctions for these areas. UUV 4 initially is not informed about the visited waypoints and it defines its regions wit this knowledge. After negotiations, regions are assigned and UUVs form their schedules. While UUV 4 is executing its tasks by incoming information, change its region definitions.
(a) UUV 3 fails, other UUVs take responsibility of the waypoints initially assigned to UUV 3

(b) Search area assignments after failure for UUVs 1 and 2.

(c) Another UUV (4) is released from the deployment area

(d) Schedules are changed accordingly after negotiations. However UUV 4 is not informed about visited waypoints and considers all waypoints of the mission. It adds all waypoints that it believes to be visited to its region definition

(e) After being informed about visited waypoints, UUV 4 only visits unvisited waypoints in its schedule.

Fig. 5. Scenario 3. UUV-3 fails during execution. UUV 4 is released from the deployment area. Schedules are changed accordingly.

On the same settings, experiments are conducted to evaluate message loss rate effects on mission completion success. Table 4 illustrates these results. When message loss rate is different from 0, as expected, performance is degraded but linearly. It should be noted that even for rate 0.75, the overall mission by final identification of mines is completed. Number of waypoint visits increase for high message loss rates. When message loss rate is 1 there is no communication among AUVs and they cannot correctly reason about region portions. Therefore each UUV searches the full area completely. Crawlers detect and identify 12.8% of
mines by their local detection in a small area (MLO target information can not be communicated in this case). Since identification is not complete, overall mission is not completed. This table illustrates our framework’s success against message losses. As a final remark, auction generation and clearing in an environment with communication delays desires special attention. Especially auction deadlines should be determined by considering communication delays which may vary during run. Plan B precautions could resolve these kinds of problems. Precautions for delayed messages on invalid situations prevent the system from getting into stuck into further inconsistencies and deadlocks.

Table 4. Performance Results for Different Message Loss Rates

<table>
<thead>
<tr>
<th>Message Loss Rate:</th>
<th>0</th>
<th>0.25</th>
<th>0.5</th>
<th>0.75</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Coverage Mission (Mc)</strong> Completion (%)</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td><strong>Identification Mission (Mi)</strong> Completion (%)</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>12.8, σ = 4.14</td>
</tr>
<tr>
<td>M\text{c} Completion Time</td>
<td>µ</td>
<td>2852.8</td>
<td>3227.6</td>
<td>4205.2</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td>σ</td>
<td>35.3</td>
<td>205.3</td>
<td>836.9</td>
<td>N/A</td>
</tr>
<tr>
<td>M\text{i} Completion Time</td>
<td>µ</td>
<td>3349.4</td>
<td>3683.2</td>
<td>4909.0</td>
<td>5141.2</td>
</tr>
<tr>
<td></td>
<td>σ</td>
<td>60.5</td>
<td>167.1</td>
<td>430.1</td>
<td>938.1</td>
</tr>
<tr>
<td>Waypoint First Visit Time</td>
<td>µ</td>
<td>1380.1</td>
<td>1390.0</td>
<td>1922.0</td>
<td>2256.6</td>
</tr>
<tr>
<td></td>
<td>σ</td>
<td>6.1</td>
<td>16.3</td>
<td>92.8</td>
<td>334.5</td>
</tr>
<tr>
<td>Waypoint # of visits</td>
<td>µ</td>
<td>1.0</td>
<td>1.0</td>
<td>1.01</td>
<td>1.09</td>
</tr>
<tr>
<td></td>
<td>σ</td>
<td>0.0</td>
<td>0.0</td>
<td>0.01</td>
<td>0.04</td>
</tr>
</tbody>
</table>

6 Conclusions

In this work, we present our generic cooperation framework, DEMIR-CF and evaluations in the context of a Naval Mine Countermeasure mission in a realistic simulator, ALWSE-MC. DEMIR-CF is a distributed framework combining an auction based allocation method and several precaution routines to handle failures and limitations of actual systems, and maintain high solution quality with available resources for multi-robot teams. Precaution routines can respond several failures some of which are illustrated in the scenarios shown in this paper. Evaluations also reveal high performance of DEMIR-CF on online task and situation handling. Since the framework is a single item auction method it can be used for the environments in which communication range is limited or communication delays are present. In general, the framework is designed for more complex missions of interdependent tasks. Near future work consists of complex missions with more
limitations for AUVs and task execution, and specifying design issues and rules for the different domains. It should be noted that the selected application domain, objectives and limitations are similar to Search and Rescue (SR) domain. Therefore we believe research in this work can also be useful for different kinds of domains such as SR.

References