Journal of Computer and System Sciences 81 (2015) 553–567

Contents lists available at ScienceDirect

Journal of Computer and System Sciences

www.elsevier.com/locate/jcss

Dynamic heterogeneous team formation for robotic urban search and rescue

Tyler Gunn, John Anderson *

Department of Computer Science, E2-445 EITC, University of Manitoba, Winnipeg, MB R3T 2N2, Canada

ARTICLE INFO

Article history:
Received 15 November 2013
Received in revised form 30 May 2014
Accepted 30 July 2014
Available online 18 November 2014

Keywords:
Multi-robot systems
Team formation
Team management
Heterogeneity
Roles
USAR

ABSTRACT

Much work on coalition formation and maintenance exists from the standpoint of abstract agents. This has not yet translated well to robot teams, however: most multi-robot research has focused on pre-formed teams, with little attention to team formation and maintenance. This means solutions fail in challenging environments where equipment is easily lost, such as urban search and rescue. This paper describes a framework for coordinating a changing collection of heterogeneous robots in complex and dynamic environments such as disaster zones. The framework allows a team to reshape to compensate for lost or failed robots, including adding newly-encountered robots or additions from other teams, and also allows new teams to be formed dynamically. The framework also includes provisions for task discovery and assignment, under the conditions of changing team membership. We evaluate this framework through an implementation where robots perform exploration in order to locate victims in a simulated disaster environment.

1. Introduction

Although much research has been performed on teams and coalition formation in multi-agent systems, most works tend to focus on abstract agents performing high level tasks in domains lacking a physical grounding (e.g. package delivery in abstract space [1,2]). While important in principle, these do not take into account many of the physical challenges of being grounded in the real world (e.g. communication distance and reliability [3]). Further, work involving teams of heterogeneous robots is usually restricted to relatively controlled laboratory environments and relies on fixed team structures determined in advance (e.g. [4,5]).

Robots operating in any real-world environment have many challenges to contend with, such as noisy and inaccurate sensor data. Localization is imperfect, and algorithms to intelligently interpret visual data are computationally expensive and inaccurate. Operation in hazardous environments, such as those presented by the exploration of other planets and disaster zones must additionally deal with the fact that robots can be damaged or destroyed. In domains such as these, communication between robots is short range, unreliable, and sporadic in nature: in disaster areas, for example, infrastructure can be heavily damaged, and the debris itself can interfere with wireless communication.

A good example of a highly challenging domain in which robots can be of value is in the aftermath of a natural or man-made disaster. This is commonly known as *Urban Search and Rescue* (USAR), and involves exploring damaged structures to locate and assist human casualties. Operation in a USAR environment presents significant mobility and sensory

* Corresponding author.
E-mail address: andersj@cs.umanitoba.ca (J. Anderson).

http://dx.doi.org/10.1016/j.jcss.2014.11.009
0022-0000/

difficulties [6]. Debris and uneven terrain can make navigation difficult and can cause a robot to become stuck. Structural changes to the environment as a result of the disaster can render existing floor-plans and maps useless.

The challenges present in a USAR environment make it likely that robots may become lost or separated from their team. Further, robots can become physically damaged or destroyed, impairing the team's effectiveness. It is also possible for different teams of robots operating in geographically separated areas to encounter one another as the mission progresses, providing an opportunity for teams to exchange members or combine resources. New robots can also be expected to arrive sporadically since not all equipment arrives at once or is sent in at the same time.

This paper describes a framework for coordinating a changing collection of heterogeneous robots operating in complex and dynamic environments such as USAR. The framework includes facilities for team formation and management (Section 4.2) as well as facilities for task discovery and assignment (Section 4.3).

2. Related work

Until recently, there has not been a large focus on how to form and maintain teams of robotic agents. Most previous works assume teams were formed in advance and will not change during the course of operation (e.g. [7–12]).

Extensive prior work (e.g. [1,2,13]) has resulted in techniques to enable self-interested agents to form mutually beneficial partnerships in groups of two or more. Although these concepts are generally applicable to robotic domains, these works are demonstrated in domains which are too abstract to show direct applicability for robots in challenging conditions. There is no consideration for issues surrounding the perception of agents in the environment, localization, or the impact of limited range unreliable communication.

George et al. [14] studied a method to form sub-teams in a larger overall team of unmanned aerial vehicles (UAVs) operating in a region. Their approach is more realistic, as it assumes robots are heterogeneous and must cooperate to achieve a common goal. Communication is assumed to be limited range, and operation takes place in a more realistic domain, but there is little or no consideration of the ability to form new teams as opposed to sub-groups.

Cheng and Dasgupta [15] developed a technique to form teams among robots exploring an area. Although their work assumes a more real-world domain, it aims to form teams for the explicit purpose of maximizing the overall explored area, where our work attempts to maintain teams for carrying out a broader set of overall tasks, the nature of which can change over the course of the mission.

Kiener and von Stryk [4] present a framework for the cooperative completion of tasks by teams of heterogeneous robots. Their framework achieves this by modeling the individual tasks of the overall mission, and storing the degree to which each of the robots can perform these tasks. The capabilities of (only) a single humanoid and single wheeled robot are determined in advance, along with weights identifying the suitability of each to all possible tasks. This information allows a central controller to allocate tasks to each robot. While the tasks involved are significant in that they involve fine motion control and interaction, this is still very primitive in terms of task allocation. The broadly different robot skills and task demands result in a predefined set of tasks with only one logical way to map these tasks to the robots in their system. Our framework instead assumes that there may be potentially a large number of potential mappings. Their approach also requires constant communication with a central controller, where our framework performs task allocation in a distributed manner.

Howard et al. [5] developed a system to automatically deploy a sensor network using heterogeneous robots. Resourceful *leader* robots guide network deployment and provide guidance to sensor nodes to keep them in formation. In contrast to our work, the teams, formation, and deployment positions of the sensor nodes are all pre-computed in advance and rely on reliable communication to a central processing unit. Changes in team structure due to loss of robots are accounted for by looking up a new pre-computed deployment pattern and adjusting the formation. No attempt is made to recover lost robots. Task allocation also relies on fixed mapping betweens tasks and robots.

Dorigo et al. [16] developed the *swarmanoids* architecture as a means of encouraging research into swarm robotics in real-world domains. In their work, three heterogeneous robot types cooperate to complete the mission of location a book on a shelf and retrieving it. The capabilities of the robot types, however, preclude them from being used in any other combination, and leaves little opportunity to adapt to changes in available robot types.

Auction-based approaches (e.g. [17,18]) have been developed to perform distributed task allocation amongst teams of heterogeneous robots. Similar to our work, these approaches assume robots can fail at any time and that communication may be unreliable. Auction-based approaches assume bidders will bid only on tasks they are capable of carrying out, where our work makes use of roles to guide the task assignment process to robots which are potentially capable of carrying out a task. Further, auction-based approaches typically assume all robots have the necessary capabilities to assign tasks, where our work assumes task allocation is itself a task which is delegated to a more capable robot.

Gage et al. [19] developed an approach to multi-robot task allocation that uses an emotion-based approach to assigning tasks. Similar to our approach, their work assumes unreliable communication and that robots can fail at any time. In their work, robots continually announce tasks requiring assignment. Those robots that hear the tasks calculate a *shame* value corresponding to their suitability to carry out the task. Similar to a threshold in a market-based system, the shame value determines whether the robot responds with an offer to carry out the task. In contrast to a standard market-based approach, not responding increases the shame value, changing the response threshold. This results in the best suited robots responding first, and the less-suited robots responding later. Although their approach attempts to reduce communication overhead, it places the burden of task allocation on all robots, even the most primitive. This is unrealistic for dangerous environments,

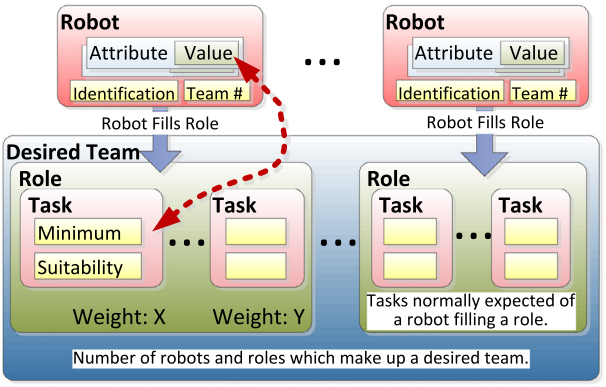


Fig. 1. Relationship between robots, roles and desired teams.

since expendability in robotics will mean that there will be likely many simpler robots that can be sent to areas of greater risk.

Ma et al. [20] developed an approach to performing frontier-based exploration using an auction-based task allocation methodology. Similar to our work, robots respond to task allocation requests with bids describing their suitability to carry out the task. Their approach, however, uses a multi-round auction approach which would be more susceptible to communication failures of the type we assume in our work. Further, our work assumes task allocation is generally centralized to a single robot, where Ma et al. assume it is performed in a completely distributed manner.

3. Preliminaries

Before discussing team management in our framework, it is necessary to have some understanding of the way our framework represents knowledge of tasks, roles, and teams in order to facilitate its primary goals of team maintenance and task management. This section describes these concepts, as well as how our framework deals with sharing knowledge among team members. Our framework assumes the task types, roles and desired team composition are determined in advance of operation by a human expert (future work could involve using learning techniques to learn or adjust these over the course of a mission).

In our framework, descriptions of possible tasks are created in advance, and describe the units of work required to complete the mission. As illustrated in Fig. 1, a task has both a minimum requirements and suitability expression defined in terms of the attributes of a robot (see Section 4.1 for how these are evaluated). The minimum requirements determine the set of capabilities a robot must possess in order to carry out the task. For robots that meet these minimum requirements, the suitability expression defines the degree to which a robot is suited to carry out that task. The minimum requirements expression for a mapping task may, for example, describe that a task requires a robot with the capability to map an area. The suitability expression of this task could assign greater suitability to a robot possessing a laser range-finder versus a sonar. Describing tasks in this manner forms the basis on which robots can reason about the best available team member to carry out a specific task. This knowledge also serves to indicate when the current team structure is less than adequate, and describes the needs of desirable new team members. The minimum requirements and suitability expressions are heuristic in nature, and determined by a human expert in advance of the mission.

To facilitate efficient task allocation and assign a general responsibility of duties, we define roles in terms of the tasks a robot filling the role is normally expected to be able to perform. A role can be thought of as a characterization of the type of work a robot filling the role would normally encounter. Thus, the tasks expected of a role determine the capabilities required of a robot filling that role. Since task requirements are formulated such that a robot's suitability to carry out a task can be calculated, it follows that a robot's suitability to fill a role is the aggregate of its suitability to complete each task normally expected of the role. Using roles provides a short-cut when assigning tasks to other robots: a task can be assigned to a team member occupying a role that is normally expected to carry out that task, eliminating the need for further reasoning.

The concept of a *desired team* is central to team maintenance in our framework. The desired team identifies the required roles and the quantity of each in order to make an effective team. This description then forms the goal that the framework's team maintenance operations aims to achieve. The desired team composition is highly domain and equipment-dependent and is determined by a human in advance of operation. Future work could support the adaptation of this description over time.

Within a team, a *team coordinator* is a special-purpose role responsible for directing the overall operation of the team. This role designates the responsibility of assigning tasks to a single robot and provides a single coordination point where the results of tasks are collected. The team coordinator attempts to ensure that each teammate has a small backlog of tasks to complete, helping robots remain productive in times of communication outages.

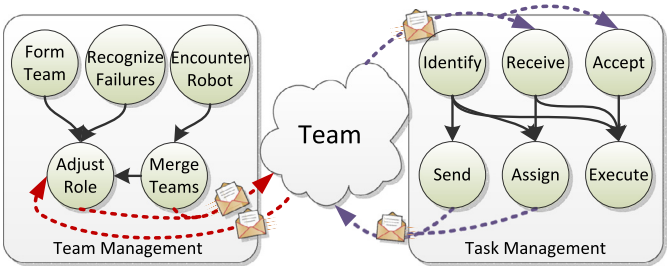


Fig. 2. Framework operation for a single robot and its interaction with the team.

4. Methodology

The primary objectives of our framework are *team maintenance* and *task management*. Fig. 2 shows how these operations are accomplished from the standpoint of an individual robot operating as a part of a team. The dashed lines indicate the communication flow between a robot and other members of the team as a result of the team and task management operations. Team maintenance operations, described in Section 4.2, focus on the formation and maintenance of teams, and their reaction to changes in team structure. Task management operations, described in Section 4.3, focus on the identification and assignment of tasks to the most suited members of the team. A deficiency in team structure can result in a situation where tasks are mapped to robots whose abilities would not provide a high likelihood of carrying these tasks out successfully. An exploration task in a disaster zone, for example, may be more likely to succeed if carried out by a tracked robot which is better able to navigate through debris-filled areas, versus a wheeled robot which may become stuck or blocked. Team maintenance operations attempt to correct deficiencies in team structure so that a higher level of suitability can be achieved when assigning tasks.

Thus, using our framework, teams are a fluid aggregation of robots, where robots switch roles within the team and change teams as necessary to make the best use of their abilities. As robots change roles and teams, the overriding goal is to form stable teams that meet the definition of a desired team as closely as possible. Our framework does not differentiate teams of one from larger groups, and so single robots can form larger teams through encounters. A single robot that fills the team coordinator role well will likely retain that role, and one that does not will cede it to a more appropriate teammate.

4.1. Task and role suitability

As described in Section 3, the team maintenance and task management operations rely upon tasks defined in terms of a minimum requirements and suitability expression. The minimum requirements expression for a task is a simple boolean expression defining the attributes and corresponding values required for a robot to carry out a task. A victim verification task, for example, could have minimum requirements $HasMap = true \wedge (HasCamera = true \vee HasBreathSensor = true)$.

Similar to the minimum requirements expressions, suitability expressions are composed of *and* and *or* clauses, except the individual clauses are assigned a weight. For each condition in the suitability expression that is satisfied (evaluates to true), a value equivalent to the weight is generated. Conditions evaluating to false generate a value of 0. The evaluated weights for conditions combined with the *and* logical operator are added together. For conditions combined with the *or* logical operator, the result is the maximum weight of the evaluated conditions. The net effect is that conditions combined with the *and* operator increase the suitability for every condition that is met, while the *or* operator acts to increase suitability based on the most valuable condition met. The suitability expression $HasMap[30] = true \wedge (HasCamera[10] = true \vee HasBreathSensor[30] = true)$, for example, favours a robot with a breath sensor over one with a camera, and assigns increased suitability to a robot with a map.

$$S = \sum_{i=0}^{n} S(T_i) W_i \tag{1}$$

Calculating a robot's suitability to fill a role involves summing up its suitability to carry out the tasks normally expected of that role. As shown in Eq. (1), for all the n tasks T normally expected of a robot filling the role R, the suitability of a robot to fill that role is calculated as the sum of the weighted suitability for the robot to carry out each task expected of the role. $S(T_i)$ is the suitability of the robot to carry out task T_i, and W_i is a weighting factor describing the relative important of the task to the role R.

These operations involving tasks, roles and suitability expressions form the basis for the team maintenance operations described in the following section.

4.2. Team maintenance

Team maintenance aims to ensure robots fill roles on the team which result in a close approximation to the definition of a desired team. Adjusting the roles robots fill on the team can occur when a robot loss or failure is recognized, or when

a new robot is encountered and a team merge and redistribution occurs. Team maintenance is also responsible for ensuring the team coordinator role is filled, should that role become open due to a change in team structure.

4.2.1. Recognizing failures

The most obvious sources of failure in a USAR domain are hardware damage and becoming physically lost. Our framework helps robots detect failures by tracking the last time a robot has heard from the other robots on its team. Teammates that a robot has not heard from in a specified period of time are considered by that robot to no longer be a member of the current team. Each robot on a team maintains its own model of the current team structure, based on the communications it has received from other members of the team.

A team recognizes the failure of robots and responds by adjusting the roles of the remaining robots in a decentralized manner. Each robot is responsible for determining if any of its teammates have failed, and adjusting the role it fills on the team in response (a *Role Check*). Robots gain knowledge of their current team composition through inspection of wireless traffic. In our framework, all wireless messages sent by a robot include the sender's team affiliation and role, which robots in range overhear. This knowledge provides a robot with the necessary information to maintain its model of the current team structure, which it uses to perform a role check, and potentially change roles in an attempt to ensure the team matches the definition of a desired team a closely as possible. Further, role changes ensure that the team coordinator role is filled by the best suited robot. The recognition of failures is implicit in that a robot adjusts the role it fills on their team in response to deficiencies it detects in its current view of the team complement.

During a role check, a robot calculates a weighted suitability to fill each role on its team, guided by the shared desired team definition. Roles which are currently under-filled according to the desired team definition are given a higher weighting. This encourages robots to fill roles to which they are less suited in the absence of a more suitable team member.

$$W = \begin{cases} \frac{D_{min}-N}{D_{min}} I_r & \text{if } N < D_{min} \\ \frac{D_{max}-N}{2D_{max}} I_r & \text{if } N \geq D_{min} \wedge N < D_{max} \\ I_r & \text{if } N \geq D_{max} \text{ and robot is better suited.} \end{cases} \tag{2}$$

Given a role R, the role check operation first calculates the suitability S of the robot to fill the role, and adds a weighting W to encourage under-filled roles to be filled. As shown in Eq. (2), the calculation of W makes use of the desired team definition and the robot's own knowledge of other robots on its team. The weighting uses knowledge of I_r, a weighting of importance for the role on the team, D_{min}, the desired minimum number of robots filling the role R, D_{max}, the desired maximum number of robots filling the role R, and N, the number of other known robots filling R on the team. If $N < D_{min}$, the role does not meet the minimum requirements, so the calculation of W provides a stronger incentive for the robot to fill the role, depending how close the role is to meeting its minimum. If $N \geq D_{min} \wedge N < D_{max}$, the role has met its minimum but is still under its maximum, a lesser weighting is given to the role. Finally, if $N \geq D_{max}$, the role is currently filled. In this case, if another robot on the team fills the role R, and has a suitability less than S, W will encourage the more suitable robot to assume R in place of the less suitable robot. Once the better suited robot switches to R, the less suited robot filling R will switch to a different role when it performs its next role check.

A helpful reviewer pointed out that there are combinations of D_{max}, D_{min}, and N for Eq. (2) that can produce counterproductive effects by biasing the system toward cases where the minimum number of robots are already present. This is indeed a weakness. However, it is important to note that W is a smaller secondary factor added after calculating S, minimizing the overall impact if such a problematic case was encountered. In the context of our particular implementation (Section 5), the values chosen along with the particular robot types and role definitions allowed the intent of the equation to be preserved and unintended role changes precluded. We are, however, investigating alternative formulae for this weighting. Eq. (3) in Section 7 shows a correction that preserves the original intent of Eq. (2).

If the role check identifies that a role change is necessary, the robot implements it and informs its teammates of the change, which can trigger subsequent role checks to occur for other members of the team. To help prevent two robots from simultaneously determining their role and implementing the same role change, a random interval is added between role checks. Each robot is responsible for changing roles on its own, and can do so without the explicit authorization of any other robot. It is, however, also possible for a robot to be instructed to change roles by another robot in the case where a team merge occurs.

Since we assume communication is unreliable, messages communicating the current team structure are not guaranteed to reach every member. This results in discrepancies between individual views and the actual team structure, the size of which is proportional to communication accuracy. As a result, a robot can potentially initiate a role change that is sub-optimal due to having an incomplete view of its team structure. These deviations are expected in the context of the team of a whole, and are compensated for in part by defining a desired team in terms of a minimum and maximum number of robots filling each role. Restricting the role checks to a periodic basis also helps to prevent the team from continually restructuring itself as its view of the team structure changes. Such oscillations are common in distributed settings, and a similar approach has been shown to be useful for choosing opponents to block in robotic soccer, for example [21].

4.2.2. Encountering robots

Where two robots on different teams encounter each other, the robots act as *representatives* for their teams (they may also be individuals and thus represent the entirety of two teams) and calculate a potential merge or redistribution, begin-

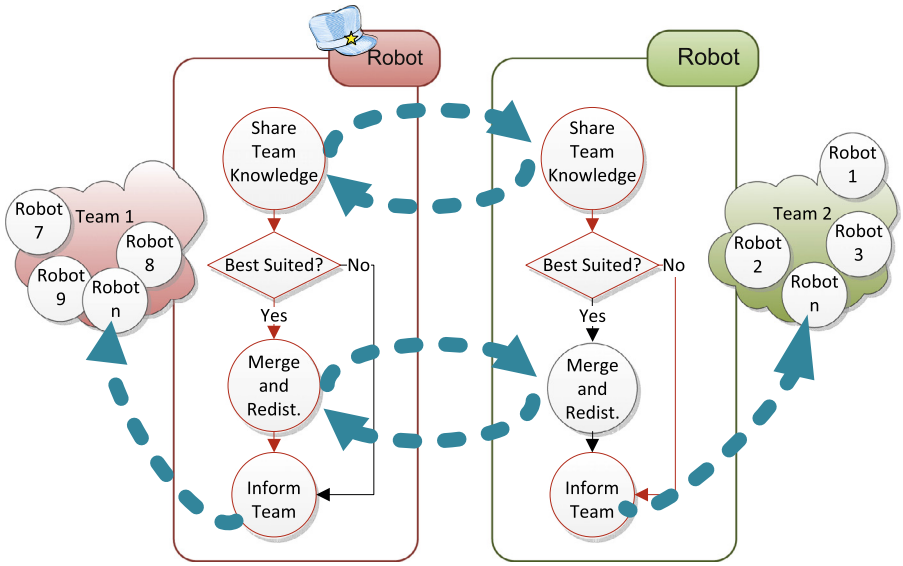


Fig. 3. Encountering robots act as representatives for their team and perform a merge and redistribution.

ning by exchanging their individual perspectives of team members and roles. It is important to note that neither of the representative robots needs to be the team coordinator. The representatives are given temporary responsibility to perform team coordination activities on behalf of the overall team. The process used in our framework allows individuals to form a larger team, gathering more individuals in future encounters, and also allows the re-balancing of existing teams that could better fit the definition of a desired team by exchanging members. Fig. 3 illustrates the major steps of the encounter process. Where more than two robots are within encounter range, our framework assumes a merge and redistribution will first occur between the first two robots which successfully begin to negotiate a merge and redistribution. The robot(s) unable to engage in a merge and redistribution will attempt to initiate one while they remain in range of robots with which they have not yet performed a recent merge and redistribution.

Our approach selects the most appropriate of the two representative robots to perform the computational work involved in this process. The determination of which robot will perform the merge and redistribution is determined by having each robot compute its suitability to fill the team coordinator role. While neither may wind up as a team coordinator, using this role as a basis for comparison allows the most capable robot of the two to be chosen. Where neither robot meets the minimum requirements to fill the team coordinator role the process is abandoned (since better robots may be encountering one another shortly).

While it would appear most useful to defer to the actual team coordinators for the two teams to perform these negotiations, this is problematic in the types of domains this framework is intended for: team coordinators' knowledge of current team structure is also imperfect, it increases reliance on two specific agents and therefore vulnerability, there are additional levels of indirection involved in contacting team coordinators, and the distance may be much greater between them, leading to less successful communication and greater likelihood of the process failing. Moving the team coordinators also prevents them (and possibly others) from doing useful work at the same time.

The encountering robots share their respective knowledge about their own team compositions. The representative robot chosen to consider the merge or redistribution combines this new knowledge along with its knowledge of its own team. This combined knowledge is used to form new teams by iteratively finding the best suited robot to fill a role on the new teams and assigning that robot to its best suited role and team (guided by the shared desired team description). Unlike the role check operation, the merge and redistribution attempts to place robots into the best suited roles on the resulting teams. The result will be either a single combined team, two teams with robots swapped between teams to ensure both teams match the definition of a desired team, or no changes (the latter happening if the teams cannot be improved).

After this operation, the other robot is informed of the role and team changes that its teammates must implement as a result. Both encountering robots are then responsible for informing their own teammates of the changes that must be implemented. Inaccurate knowledge of the current team compositions and lost role change instructions can cause the resulting teams to deviate from the desired team definition, and future role checks provide an opportunity to compensate for this. Similarly, cases where multiple team merge and redistributions occur at the same time have the potential to cause conflicting role change instructions to be distributed to a team, which can cause a deviation from the desired team definition until future role checks occur.

Where both encountering robots are operating alone, the encounter provides an opportunity for a new team to form out of the individuals. The new team can continue to merge in individuals it encounters, further strengthening the team's capabilities. In this way, it is possible for teams to build up starting with a single robot.

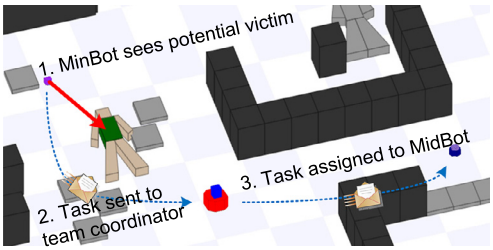


Fig. 4. Example of task discovery and assignment.

4.3. Task management

While forming and adapting teams as events unfold in a dangerous environment is important, the ability to do useful application-driven work in conjunction with this is equally vital. Tasks must be received or discovered by members of a team, and this work must be carried out by team members or left for others. This goes hand in hand with evolving team membership, since the appropriate agent to carry out a task will change as a team does.

In our framework, every robot on a team plays a part in the task allocation process. As robots carry out work, they are also responsible for identifying new tasks in the environment to the degree their sensor equipment allows them to do so.

Fig. 4 illustrates the task discovery and assignment process, using elements from our example implementation (Section 5). Robots are expected to discover tasks as they operate. In this example, a robot has found a potential victim and creates a task to confirm. The robot lacks the necessary capabilities to carry out the task, and sends it to the team coordinator for assignment. The team coordinator performs task assignment, ensuring the task will be carried out by a robot with the necessary capabilities.

Robots use a task list to track tasks they are currently involved with. This includes tasks discovered during operation, tasks undergoing assignment, and tasks the robots will carry out itself.

4.3.1. Task lists

We assume tasks are generally carried out to completion by a single robot without interruption. In our example implementation, we made the provision that higher priority tasks could interrupt the execution of lower priority tasks to ensure that more important victim verification tasks could preempt less important exploration tasks. Our implementation assumes of the tasks identified for the robot to carry out on its own on its task list, a robot will attempt to carry out the highest priority, oldest task first. We chose this task preemption and priority scheme for our example implementation due to a desire to ensure high priority victim identification tasks received prompt attention. We assume other task prioritization schemes could be more suitable for other domains.

We assume the robot filling the team coordinator role will attempt to assign tasks as soon as possible, helping to ensure tasks are distributed amongst the members of the team. Each robot maintains a small backlog of each type of task on its task list (we chose a fixed limit of 5 for our implementation, but this of course can vary between robots and the estimated difficulty of tasks). This means, for example, a robot could maintain a backlog of 5 exploration tasks and 5 victim verification tasks. The backlog helps ensure robots have work they can carry out during periods where communication conditions prevent the acquisition of further tasks from the team coordinator. The fixed size of the backlog helps ensure tasks are more evenly spread amongst the members of a team. Assigning tasks as soon as possible helps ensure a failure of the team coordinator does not result in the loss of a large number of unassigned tasks.

When a robot discovers a task, it adds it to its task list. If based on the minimum requirements expression (Section 4.1) the robot is able to carry out the tasks on its own, and its backlog for that task type has not been exceeded, the robot will retain the task and carry it out by itself when time permits. If the robot lacks the necessary minimum requirements to carry out the task, or its backlog for that task type has been exceeded, the robot attempts to send the task to its team coordinator for assignment.

A robot may be unable to send a task to its team coordinator for assignment during periods of poor communication. Further, it is possible the robot filling the role of team coordinator has changed, and the robot did not yet discover the change. If this occurs, and the robot has the necessary capabilities to assign tasks on its own, it will attempt to do so. If the robot lacks the necessary capability to assign the task, it will retain it until a point in time when it can successfully send the task to the team coordinator (i.e. when communication conditions improve, team knowledge is updated, or the robot joins another team). Should the robot become unable to send its stored tasks back to the team coordinator (due to becoming lost, or destroyed), it is expected that other members of the team will re-discover these tasks as operation progresses.

4.3.2. Task assignment

Rather than assuming task assignment is completely distributed, we assume the robot on a team filling the role of team coordinator is generally responsible for assigning tasks to members of the team. If the robot filling the team coordinator role changes, the responsibility of assigning tasks shifts to the new team coordinator. We assume a team will be made

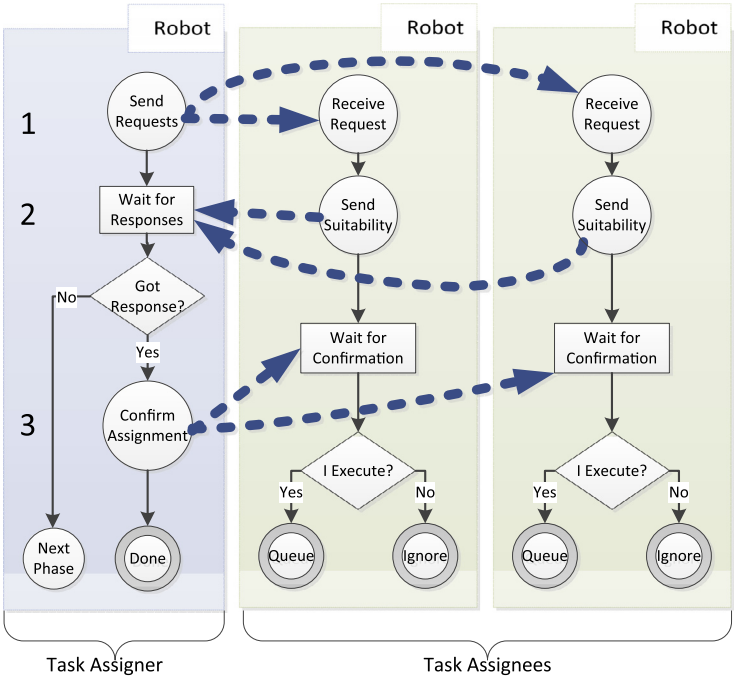


Fig. 5. The task assignment process.

up of many very simple robots which lack the necessary capabilities to perform effective task allocation, and as such it is advantageous to rely on a more capable team coordinator for this operation. Assigning tasks via the team coordinator also helps reduce duplication of effort, as the team coordinator has a more complete picture of the work required in the domain and is able to filter out duplicate tasks reported by different team members. It also provides a central point to collect the results of carrying out tasks.

The team coordinator will periodically attempt to assign tasks from its task list to other members of the team. For each task requiring assignment, the robot will first attempt to use a *role-based assignment* process. If no suitable robot is found using this process, an attempt is made to use a less restrictive *exhaustive assignment* process.

4.3.3. Role-based task assignment

Given that our framework defines a role in terms of the tasks normally expected of a robot filling that role, role-based task assignment limits the task assignment process to those robots filling roles normally expected to be able to carry out a task. This speeds the task assignment process by eliminating the need to match task requirements to the attributes of each team member when determining which team members can potentially carry out the task. Another advantage of this approach is that the task assignment messages can be specifically addressed to potential assignees, rather than using general broadcasts which all robots in range must process.

Role-base task assignment occurs in three phases (illustrated in Fig. 5): *sending task assignment requests*, *waiting for responses*, and *sending confirmations*.

During the first phase (Fig. 5, Step 1), the task assigner uses its knowledge of the roles robots fill on its team to determine which robots fill a role normally expected to be able to complete the task. The assigner offers the task to all potential assignees simultaneously. Each assignee responds back based on its current workload and capabilities.

As illustrated in Fig. 5, Step 2, the assignee first determines its suitability to carry out the task by evaluating the minimum requirements and suitability expression of the task against its own capabilities. Where the assignee meets the minimum requirements of the task, and its current backlog for tasks of the same type permits, it responds to the assigner indicating its acceptance of the task, along with its calculated suitability to carry out the task. Where the assignee does not meet the minimum requirements, or its workload does not permit, it responds with a rejection message.

The task assigner waits for a fixed period of time before evaluating the task acceptance messages it has received (Fig. 5, Step 3). The task acceptance messages are evaluated to determine which assignee reports the highest suitability to carry out the task (if any). We use a mission-specific cost determination to break ties between equally suited robots. In our example implementation the cost is proportional to the distance from each assignee to the task location, ensuing that when multiple assignees are equally suitable to carry out a task, the task will be assigned to the one physically closest to the task location.

Once the assignee is chosen, a confirmation is sent to all task assignees indicating which one was chosen to carry out the task. Upon receipt of the confirmation, the chosen assignee marks the task as accepted and sends back an acknowledgment to confirm its acceptance of the task. At this point the chosen assignee is free to carry out the task as workload permits.

The other assignees remove the task from their task lists upon receipt of the confirmation message. Where communication conditions are poor, it is possible an assignee may not receive a confirmation. A timeout is used to ensure the assignee only waits a short time for a response; where it does not receive a response in time it removes the task from its task list.

To confirm successful assignment of the task, the task assigner waits for the assignee's acknowledgment message for a fixed period of time. The task assignment is considered successful and the task is removed from the assigners task list, if an acknowledgment is received. If an acknowledgment is not received, task allocation moves on to the next phase.

Poor communication conditions may prevent the assignee's acknowledgment message from being received by the task assigner. This could cause the task to be assigned to multiple robots. This can lead to a duplication of effort due to multiple robots carrying out the same task. Some tasks, however, are not as negatively impacted by multiple assignment. Two robots assigned the same exploration task in our example implementation would be expected to traverse different paths to reach the task location. This results in an opportunity for each robot to discover new work along the way. In other domains, duplication of effort may be undesirable, and an implementation would need to explicitly take measures to minimize the impact of duplicate work.

4.3.4. Exhaustive task assignment

Due to communication failures, or the task assigner having inconsistent knowledge of its current team composition, it is possible for the role-based task assignment process (Section 4.3.3) to complete without finding a suitable robot to assign a task to. As shown in Fig. 5, in such a scenario the task assigner will move on to the next phase of task allocation, *exhaustive task assignment*, where all robots in range are considered for assignment.

Exhaustive task assignment is not ideal, as it places extra load on the team to process the assignment requests and track responses. It also necessitates a greater reliance on communication between robots.

Exhaustive task assignment uses the same three phases as role-based assignment (see Fig. 5). The distinction is that in exhaustive task assignment, a task assignment request is broadcast to any robot in range that can hear it – no specific addressing occurs. This helps broaden the pool of potential recipients, helping to mitigate against particularly poor communication conditions. Robots receiving the task assignment request on the assigner's team send back responses as in role-based assignment.

If after role-based and exhaustive assignment a task is still unassigned, it is re-queued on the assigner's task list so that an attempt can be made to assign it later.

4.4. Coping with inconsistencies

Since our framework assumes communication is unreliable, it is expected that the knowledge any one robot has of its team will be inconsistent with the actual team composition. When communication conditions are favorable, it can be expected that each member of a team will have a reasonably good picture of its current team composition. Conversely, during periods of poor communication conditions, it can be expected that members of a team will have not receive all messages sent by its teammates, resulting in its own knowledge of the current team composition differing from reality. Section 4.4.1 discusses the impact of inconsistent team knowledge on the team management process. Section 4.4.2 discusses the impact of inconsistent team knowledge on the task assignment process.

4.4.1. Team management inconsistencies

Due to inconsistent team knowledge, a robot could potentially implement a role change that results in the overall team composition deviating away from, rather than closer to, the definition of a desired team. As other robots learn about the role change, there is an opportunity for them to change roles to compensate. Further, as the original robot gains more accurate knowledge of its team, it can change roles again if necessary.

Inaccurate team knowledge can have a similar impact to the team redistribution resulting from encounters (Section 4.2.2). The reformed teams would be determined based on the inaccurate knowledge, causing some robots to be instructed to change to a sub-optimal role. Further, role and team change instructions could fail to be implemented by team members as a result of communication failures, resulting in a deviation from the redistributed team from what the encountering agents intended. As team members perform periodic role check operations, the team has an opportunity to adjust its structure to account for the changes which failed to be implemented.

Our framework assumes the role and team change instructions are only used by the robots implementing the changes, and not by other teammates to attempt to learn about the new team composition. This helps ensure only the successfully implemented changes are recognized by the team.

It is also possible for robots to be instructed to change teams when it does not make sense to do so in the context of the other changes which ultimately succeed. The result will be one or both teams operating in a degraded state. Future encounters between the same teams or other teams can provide an opportunity to compensate.

Another potential source of inconsistency is the case where two robots on a team simultaneously encounter other robots (either on the same team or different teams) and negotiate team redistributions at the same time. Such scenarios are considered to be rare (they did not occur often in our experiments), and would likely result in the teams involved deviating from the definition of a desired team. As knowledge of the implemented spreads, team members will change roles in an

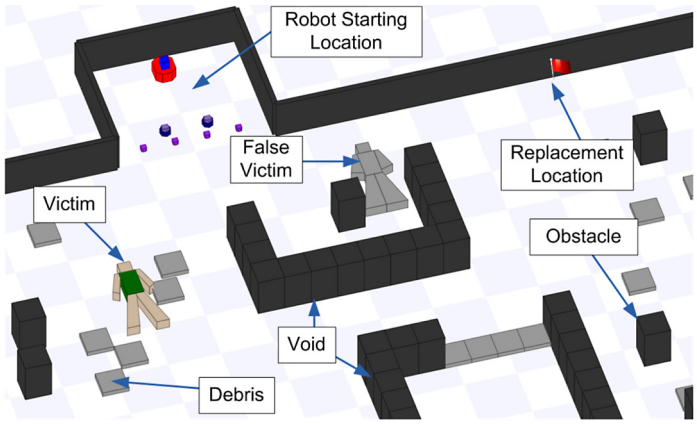


Fig. 6. Elements present in simulated USAR environment.

attempt to compensate. Future encounters with other teams would provide an opportunity to make up for deficiencies in team structure.

4.4.2. Task allocation inconsistencies

Inconsistent team knowledge also impacts the assignment of tasks within a team. Due to inconsistent team knowledge, it is possible a task assigner may not assign the task to the best suited robot on the team. The minimum requirements expression of a task ensures only robots that have a chance of carrying out a task will attempt to do so. As the team operates, knowledge of team structure and team composition will change. Thus, at some points in time task assignment operations will assign tasks to the best suited robots, while at others they will not. Our framework assumes assigning tasks to less suited robots is better than performing no task assignments at all.

During periods of particularly poor communication conditions, in addition to team knowledge being inconsistent between members of a team, the overall task assignments will have a tendency to fail. Poor communication will result in fewer task assignment requests reaching the intended assignee robots. This means the pool of available task assignees will be smaller than would normally be expected. Where none of the potential assignees meet the minimum requirements to carry out a task, a task will tend to remain unassigned. Further, where few robots meet the minimum requirements, there is a lower chance the available robots are well suited to carry out the task, resulting in less than ideal task assignments. The task backlog each robot maintains helps ensure that the robot has a backlog of work to be performed during these periods of poor communication conditions.

Where communication conditions are extremely poor, the result will be few discovered tasks sent to the team coordinator for assignment, and few tasks assigned to the team by the team coordinator. In such scenarios, robots must fall back to their backlog of tasks to ensure useful work is carried out.

When a robot becomes lost or separated from its team, it is possible for the robot to build up a backlog of discovered tasks which it lacks the capabilities to carry out on its own. Further, the robot may lack the necessary capabilities to assign these tasks on its own. In such a scenario the robot will retain the tasks it encounters, as its capabilities permit. As the robot continues operation, it can either re-encounter its current team, or potentially join another team in operation in the environment. At this point, the backlog of tasks would be sent to the team coordinator for assignment. Where the robot joins a different team, the unassigned tasks provide another means of transferring knowledge from one team to the other.

It is also possible for a robot to be destroyed, taking a backlog of tasks with it. Our methodology assumes these tasks become lost, and would be re-discovered by other robots as the mission progresses. In a domain where it is critical to ensure no tasks are lost, the team coordinator could retain knowledge of all tasks and reassign uncompleted tasks assigned to lost team members.

The team coordinator attempts to detect and remove duplicate tasks from the pool of tasks it maintains. It is still possible, however for poor communication or an overlap in team operational areas to cause similar tasks to be identified and carried out. This did not appear problematic in our example implementation, as the paths robots took to reach a task location would typically be different, providing an opportunity to discover other useful work on the way. In a domain where duplication of work is undesirable further effort would be required to minimize or eliminate rework.

5. Experimental evaluation

We evaluated the effectiveness of our approach using a simulated USAR domain created using the Stage multi-robot simulator [22]. Heterogeneous robots explore damaged structures in order to build a map of the environment and locate casualties. We assume robots can become lost or separated from their team and new robots will be released into the environment as time goes on. Our implementation was coded directly against the Stage API, allowing for repeatable simulation

runs, executed much faster than real-time. A modular design approach ensures the implementation can form the basis of future experiments. We also made modifications to the Stage simulator to provide simulated, unreliable communication between robots.

Using simulation to study our approach is appropriate as the primary focus of our work is to support teamwork and coordination between robots, rather than complete accuracy in USAR in particular. The approach of using a simulated USAR environment for multi-robot research is well established (e.g. [23] used simulated USAR environments).

Fig. 6 shows the main elements of our simulated USAR environments. Teams of robots begin operation along the periphery of the environment, and are composed of a mix of three different robot types arranged in a formation facilitating the mutual identification of teammates. Room-like voids are present in the environment, with access to some being blocked by low-lying debris. Obstacles impassable to all robots are randomly distributed in the environment, as are areas of low-lying debris passable by some robots.

Finally, victims and debris configurations resembling victims (false victims) are distributed in the environment. Since detection of victims in a real-world USAR environment is a challenging problem unto itself, the two victim types provide a simplified means of introducing uncertainty into the simulated victim confirmation process. Some robot types are able to distinguish between the victims and false victims, where others must rely on more capable robots to make the distinction.

Our implementation assumes three types of robots are available; *MinBots*, *MidBots* and *MaxBots*. The MinBots are small (10 cm diameter), expendable robots with a wheeled physiology, restricting them to open areas. They are equipped with sonar sensors for navigation and mapping, and sensors capable of detecting the potential presence of victims in the environment. The MinBots do not posses the memory or processing capabilities to coordinate a team. The MaxBots are larger (40 cm diameter), complex robots with the computational capabilities required to coordinate a team and plan an effective exploration of the environment. They are equipped with a tracked drive, allowing them to drive over areas of light debris, giving them access to areas the other robot types cannot access. The MidBots are 20 cm in diameter and have computational capabilities that fall in the middle; they are able to coordinate a team, but not as effectively as the MaxBots. The MidBots posses high fidelity victim identification sensors, and are able to confirm potential victim readings reported by the MidBots.

The tasks in our environment are focused on exploration of frontiers identified by more powerful robots and verification of potential victims identified by less powerful robots. These tasks are grouped into roles focused primarily on exploration, and others focused primarily on victim verification.

The robot filling the team coordinator role is responsible for directing the general exploration. The team coordinator generates exploration tasks using a frontier-based exploration paradigm, similar to [24]. Using the team coordinator's consolidated map of the environment, a frontier identification module generates tasks representing the unexplored areas in the environment. The MaxBots make use of a path-planning algorithm to help guide task assignees to the frontiers, where the MidBot is assumed to lack the necessary capabilities. This means the MaxBots will generally do a better job at guiding the exploration than the MidBots, as it can provide directions to assignees which will help avoid them getting stuck in local minimums, for example.

Fig. 5 illustrates the concept of victim identification using the different robot types. A MinBot possessing a less capable victim identification sensor sees a potential victim. Since it cannot distinguish between victims and false victims, it creates a victim verification task, which is sent to the team coordinator for assignment. The team coordinator assigns the task to a nearby MidBot possessing the necessary sensory capabilities to confirm the victim's presence.

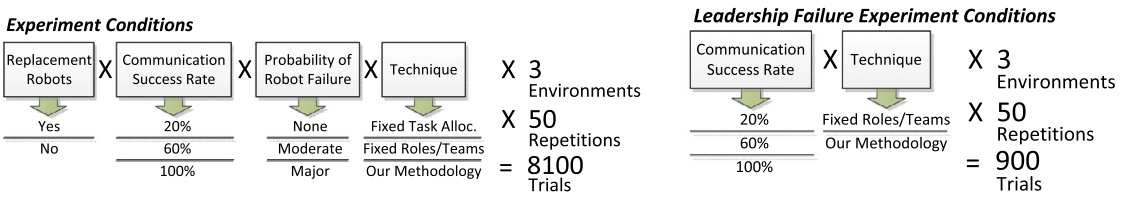
The task types are grouped into roles focused primarily on exploration, and others focused primarily on victim verification.

5.1. Experiment

Environments have two teams which start with one MaxBot, two MidBots, and four MinBot. Teams begin operation in opposite corners of the environment, out of communication range from one another. The environments are 60 m×60 m in size, and include 50 randomly positioned rooms which are 5–12 m wide, and 5–12 m long. Access to 60% of the rooms is blocked by debris, restricting access to the MaxBots. The remainder of the environment is filled with randomly placed debris and obstacle elements, 60% of which is passable by the MaxBots, until 13% of the environment is filled with debris, obstacles or rooms. 20 victims are distributed in the environments. An additional 10 debris configurations resembling victims are included, allowing for the potential of misidentification through MinBot sensor errors.

To evaluate our methodology, we performed an experiment to study the efficacy of our methodology when coping with communication and robot failures. As shown in Fig. 7a, two of the independent variables in our experiment control communication success rate, and the probability of robot failure. Our methodology also allows a team to adapt to accommodate the arrival of replacement robots. Another independent variable determines whether replacement robots are available or not. Replacements (10 MinBots, 2 MidBots, and 1 MaxBot) begin operation from the edge of the environment at the 5 minute mark.

We compared our methodology against two base cases. In the first, robots are not permitted to change roles and cannot switch teams. This means team structure is fixed: teams cannot gain team members, and team members cannot voluntarily leave the team. Further, robots are not able to change their roles or team membership to adapt to changes to the team as a result to failures. This provides a means of evaluating our framework's performance from the standpoint of adaptive team



(a) Experiment conditions.

(b) Leadership failure experiment conditions.

Fig. 7. Experiment conditions: coping with communication and robot failure.

management. Because tasks are still allocated in the base case using our task allocation methodology, it shows how much performance improvement is due to improved team management.

The second base case uses fixed roles and team membership, and includes the restriction that there is a 1:1 mapping between tasks and robot types. This means each task type can only be carried out by one type of robot. This essentially provides a worst-case comparison for multi-robot exploration in this domain: we can compare our approach and the base case above to a situation where all robot interaction is completely inflexible.

We used a factorial experiment design (Fig. 7a) resulting in 8100 experimental trials. We used 3 different environments to help eliminate potential bias due to features of any one environment. The environments were generated using a tool, ensuring the same environment coverage, number of victims, and equal distance between team start locations. We performed 50 repetitions of each experimental condition, which were run for 30 minutes of simulated time each.

To evaluate the efficacy of our methodology, we recorded two values at fixed times throughout each trial: the percentage of the environment covered, and the percentage of victims successfully identified.

5.2. Failures in team leadership

We performed a second experiment with a smaller scope to evaluate our framework's ability to cope with failure of a team's leadership structure (Fig. 7b). We introduced a failure in the leadership structure of a team at fixed times (10 and 15 minutes) to allow observation of the performance of a single team as it adapts to these failures. This second experiment was limited in scope, and did not consider the availability of replacement equipment. Further, our methodology was compared only against the first base case scenario, where roles are fixed and team membership is fixed.

6. Discussion

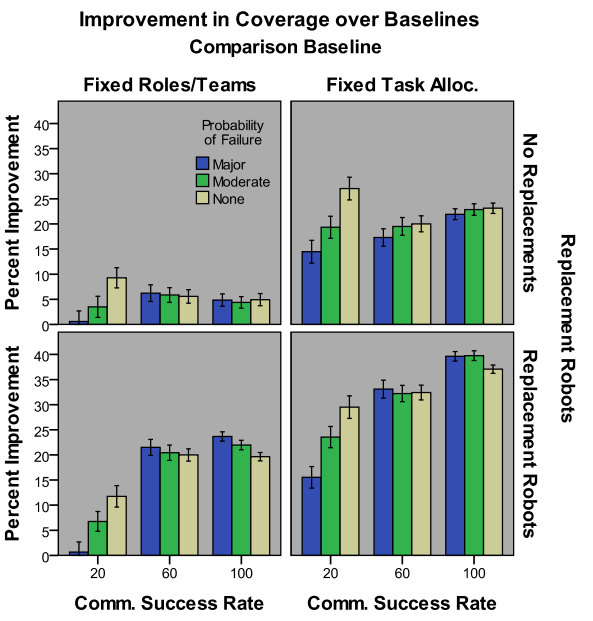
Figs. 8a and 8b show the improvements realized using our framework in the percent of the environment covered and victims identified, respectively, over the baseline cases. Although our methodology shows an improvement over the base cases at the 20% communication success rate, it is important to note that the performance of all methodologies was actually quite poor. Too few messages were successfully delivered, resulting in a general failure to allocate tasks, and a tendency for teams to break apart over time.

With a communication success rate of 60%, our methodology was able to compensate for the coordination issues associated with robots becoming separated from their team, by allowing them to join another team, or form a new team in response. The smaller number of victim identification tasks compared to frontier exploration tasks, coupled with the scarcity of robots with the capability to identify victims resulted in a higher performance improvement in terms of the number of victims identified (Fig. 8b) compared to the percent of the environment covered (Fig. 8a).

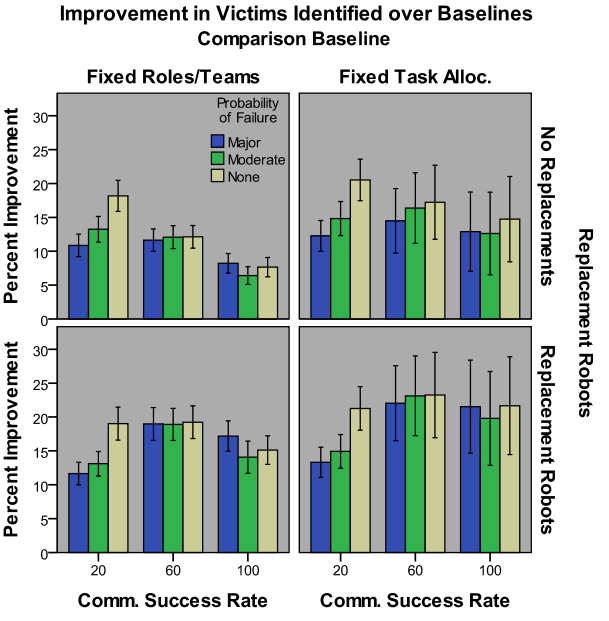
Figs. 8a and 8b also show the improvements realized where replacement robots are available. These results show that the framework is able to allow teams to take advantage of the replacement equipment which becomes available after the start of the mission.

The leadership failure experiment clearly demonstrated our methodology's ability to enable a team to continue operation, despite the failure of a robot filling the critical team coordinator role. Where roles and team membership are fixed, the failure of the team coordinator resulted in the team ceasing to make further progress. Using our methodology, the team adjusted to the failure of the team coordinator, and was able to continue making progress despite the loss of the better suited robot.

Fig. 9 illustrates the impact of leadership failures on a team's progress. It shows the percent of the environment covered over the duration of the trials where the communication success rate is 60%. The vertical lines indicate where failures of robots in the team coordinator role were introduced. In the baseline where roles are fixed and teams are fixed, the team ceases to make progress after the first failure. Using our framework, the team is able to adjust itself to compensate for the loss of the robots occupying the team coordinator role, and to continue making useful progress.



(a) Improvement in coverage.



(b) Improvement in victims identified.

Fig. 8. Experiment Results: coping with communication and robot failure.

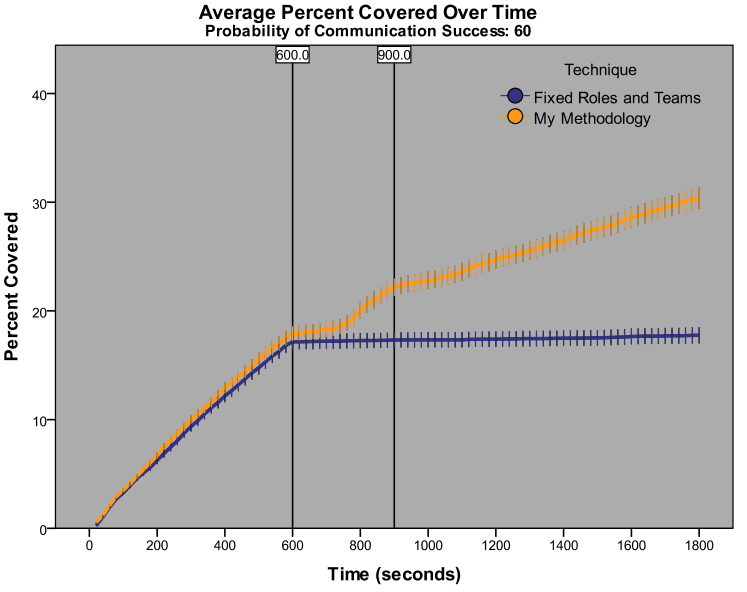


Fig. 9. Leadership failure experiment coverage over time.

7. Future work

Although our methodology showed significant utility, our example implementation revealed a number of areas where improvements could be made, or other interested research could be performed. An implementation in a physical environment using real robots would be valuable, as it would increase the difficulty of operation considerably.

Our methodology assumes agents can suffer failures and responds to them. It would be useful to incorporate a failure model of the robots and their components. This information could help drive the task assignment process (e.g. [25] found exploration performance could be improved by anticipating failures based on a robot's reliability), ensuring critical tasks are carried out by more reliable agents, for example. Robots could also monitor their performance, and that of others, in order to adjust reliability knowledge based on actual experiences.

In terms of team structure and membership, it would be useful to investigate the use of techniques to enable teams to learn the ideal team structure based on its experiences. It would also be interesting to study the impact of relaxing the restriction that a robot is a member of only one team, providing more opportunity for collaboration between teams and the sharing of rarer capabilities.

It would also be useful to investigate viewing team recruitment as an allocatable task. Currently, team members are obtained by encountering them or balancing membership with another team. However, there may be situations where actively searching for a given type of robot may be more beneficial than allocating tasks in the interim to less-suited robots. The balance between effective task allocation and active management of team membership is a subtle one and deserves greater exploration.

$$W = \begin{cases} \frac{D_{min}-N}{D_{min}} I_r & \text{if } N < D_{min} \\ \frac{D_{max}-N-D_{min}}{D_{max}-D_{min}} \frac{1}{2} I_r & \text{if } N \geq D_{min} \wedge N < D_{max} \\ I_r & \text{if } N \geq D_{max} \text{ and robot is better suited.} \end{cases} \tag{3}$$

Finally, it was noted that the design of Eq. (2) led to the potential to bias role switches away from rather than toward underfilled roles. Our implementation allowed this to work as intended, and thus the results presented above are a reasonable statement of the abilities of this framework. However, this error could negatively affect performance in implementations that differ in equation parameters or in the roles and types of robots employed. Eq. (3) shows a correction that preserves the original intent of the formula we used. Making this correction, however, also led to much thought about the elements that could be used to bias role switches, and we intend to study the impact of alternative formulae in our sample implementation and in domains with more and less heterogeneous robots.

8. Conclusion

Our methodology shows strong benefits when helping a team cope with team members getting lost due to unreliable communication and the difficult nature of the environment. Lost team members are able to form their own team or join another team they encounter. Where a robot has especially important skills, our methodology helps ensure a lost robot is able to continue providing useful work towards the overall mission. Where critical members of the team suffer a failure, our methodology shows a significant improvement over the baseline case, allowing a less suited robot to take on the team coordinator role. Where replacement robots are available, our methodology shows a clear benefit in allowing the replacements to form new teams, and be integrated into existing teams.

This research has demonstrated the utility of defining the composition of a team in terms of roles describing the types of work normally expected of its members, and using this as a means of reasoning about the changes which can be implemented to the structure of these teams in response to the loss or failure of team members, or the discovery of new potential team members.

It is our hope that the success of this research will encourage future research into the issues involved with the effective coordination of robots operating in difficult and challenging domains. Robots operating in these domains must cope with difficult conditions, and cannot make assumptions about team structure or composition, making research into team formation and maintenance in these conditions even more important.

References

[1] P.S. Dutta, S. Sen, Forming stable partnerships, Cogn. Syst. Res. 4 (3) (2003) 211–221.
[2] M. van de Vijsel, J. Anderson, Increasing realism in coalition formation, in: Proc. of CIRAS-2005, 2005.
[3] L. Vig, J. Adams, Issues in multi-robot coalition formation, in: Multi-Robot Systems, vol. 3, Springer, 2005, pp. 15–26.
[4] J. Kiener, O. von Stryk, Cooperation of heterogeneous, autonomous robots: a case study of humanoid and wheeled robots, in: Proc. of IROS-2007, 2007, pp. 959–964.
[5] A. Howard, L. Parker, G. Sukhatme, Experiments with a large heterogeneous mobile robot team: exploration, mapping, deployment and detection, Int. J. Robot. Res. 25 (5–6) (2006) 431–447.
[6] R. Murphy, J. Casper, J. Hyams, M. Micire, B. Minten, Mobility and sensing demands in USAR, in: Proc. of IECON-2000, vol. 1, 2000, pp. 138–142.
[7] I. Rekleitis, V. Lee-Shue, A.P. New, H. Choset, Limited communication, multi-robot team based coverage, in: Proc. of ICRA-2004, vol. 4, 2004, pp. 3462–3468.
[8] J. Bruce, M. Bowling, B. Browning, M. Veloso, Multi-robot team response to a multi-robot opponent team, in: Proc. of ICRA-2003, vol. 2, 2003, pp. 2281–2286.
[9] N. Lau, L. Lopes, G. Corrente, N. Filipe, Multi-robot team coordination through roles, positionings and coordinated procedures, in: Proc. of IROS-2009, 2009, pp. 5841–5848.
[10] N. Boonpinon, A. Sudsang, Constrained coverage for heterogeneous multi-robot team, in: Proc. of ROBIO-2007, 2007, pp. 799–804.
[11] L. Giannetti, P. Valigi, Collaboration among members of a team: a heuristic strategy for multi-robot exploration, in: Proc. of MED-2006, 2006, pp. 1–6.
[12] T. Li, C.-Y. Chen, Y.-C. Yeh, C.-C. Yang, H.-K. Huang, H.-L. Hung, C.-H. Chu, S.-H. Hsu, D.-Y. Huang, B.-R. Tsai, M.-C. Gau, R.-J. Jang, An autonomous surveillance and security robot team, in: Proc. of ARSO-2007, 2007, pp. 1–6.
[13] S. Liemhetcharat, M. Veloso, Modeling and learning synergy for team formation with heterogeneous agents, in: Proc. of AAMAS-2012, 2012, pp. 365–374.
[14] J. George, P. Sujit, J. Sousa, F. Pereira, Coalition formation with communication ranges and moving targets, in: Proc. of ACC-2010, 2010, pp. 1605–1610.
[15] K. Cheng, P. Dasgupta, Multi-agent coalition formation for distributed area coverage: analysis and evaluation, in: Proc. of IEEE WI-IAT-10, vol. 3, 2010, pp. 334–337.

[16] M. Dorigo, D. Floreano, L. Gambardella, F. Mondada, et al., Swarmanoid: a novel concept for the study of heterogeneous robotic swarms, Tech. rep., IRIDIA, Université Libre de Bruxelles, 2011.
[17] B. Gerkey, M. Mataric, Sold!: auction methods for multirobot coordination, IEEE Trans. Robot. Autom. 18 (5) (2002) 758–768.
[18] S. Sariel-Talay, T. Balch, N. Erdogan, A generic framework for distributed multirobot cooperation, J. Intell. Robot. Syst. 63 (2011) 323–358.
[19] A. Gage, R. Murphy, K. Valavanis, M. Long, Affective task allocation for distributed multi-robot teams, Center for Robot-Assisted Search and Rescue, 2004.
[20] X. Ma, F. Meng, Y. Li, W. Chen, Y. Xi, Multi-agent-based auctions for multi-robot exploration, in: Proc. of WCICA-2005, vol. 2, 2006, pp. 9262–9266.
[21] J. Anderson, J. Baltes, An agent-based approach to introductory robotics using robotic soccer, Int. J. Robot. Autom. 21 (2) (2006) 141–152.
[22] R. Vaughan, Massively multi-robot simulation in stage, Swarm Intell. 2 (2) (2008) 189–208.
[23] M. Eghbali, M. Sharbafi, Multi agent routing to multi targets via ant colony, in: Proc of ICCAE-2010, vol. 1, 2010, pp. 587–591.
[24] B. Yamauchi, Frontier-based exploration using multiple robots, in: Proc. of AGENTS, ACM, New York, NY, USA, 1998, pp. 47–53.
[25] S. Stancliff, J. Dolan, A. Trebi-Ollennu, Planning to fail – reliability needs to be considered a priori in multirobot task allocation, in: Proc. of IEEE Intl. Conf. on Systems, Man and Cybernetics, SMC, 2009, pp. 2362–2367.