A Hybrid Dynamic Task Allocation Approach for a Heterogeneous Multi-Robot System

Yan Meng and Kashyap Shah

Department of Electrical and Computer Engineering
Stevens Institute of Technology, Hoboken, NJ 07030, USA

Abstract. In this paper, we propose a communication-efficient hybrid task scheduling algorithm for a heterogeneous multi-robot system under dynamic unknown environment, where each robot makes its own decision through communicating with others as well as checking a global task status queue. The proposed hybrid algorithm takes advantage of centralized approaches to improve the overall system efficiency and distributed approaches to reduce the communication overhead, which automatically leads to a reasonable reduction of power consumption. This algorithm avoids unnecessary communication by broadcasting global information which is in everybody's interest and meanwhile limits specific information which is in interest of some specific robots only. In addition, each robot would dynamically allocate the task to robots which are capable and most available. This feature makes the system robust against communication failures and robot failures. Simulation results demonstrate the efficiency and robustness of the proposed approach.

1 Introduction

With growing need of building reliable real-time applications coupled with advancement of high-speed networks and high-performance computers, in the past decade heterogeneous multi-robot systems have been increasingly used for many real-time applications, like urban search and rescue, surveillance, hazardous materials detection, and reconnaissance, in which the correctness of the systems depend not only on the results of a computation, but also on the time which these results are produced [1]. To achieve real-time performance of such a complex system, an efficient task allocation and coordination among the team members is required. Vali Veloso stated in [2] that team performance can be drastically increased if the team coordinates well and the information is being shared by all teammates in a multi-robot environment.

Dynamic task allocation for multi-robot systems under dynamic environment is a challenging problem, which aims to efficiently finish all of unknown tasks as fast as possible while keep the cost as low as possible. Although some algorithms have been proposed to tackle this problem, such as auction-based algorithm like MURDOCH [3][4], behavior-based algorithm like ALLIANCE [5][6], and instantaneous greedy scheduler based approaches, all of these available methods have a great deal of broadcast communication overhead to share information with all of team members.

Meng Y. and Shah K. (2007).
A Hybrid Dynamic Task Allocation Approach for a Heterogeneous Multi-Robot System.
In *Proceedings of the 3rd International Workshop on Multi-Agent Robotic Systems*, pages 3-13

Some of available algorithms are only good for a homogeneous robot team with one global task like mapping or exploration of an area, and some of them don't take consideration of system robustness in the case of communication failure or robot malfunctions.

In this paper, we aim at investigating a task scheduling algorithm for a heterogeneous multi-robot system under dynamic unknown environment. As we know, a centralized approach consists of making all decisions in one place, where all the tasks to be performed are collected by a central scheduler. This centralized scheduler decomposes tasks into programs of actions, order actions when necessary and assigns them to robots with respect to their capabilities, work loads, and locations. The centralized approach is efficient with small number of agents, but its performance would be degraded significantly in a large-scale team. Furthermore, centralized approaches are not appropriate for coordinating the action of multiple robots in a dynamic unknown environment where unforeseeable events may occur.

On the other hand, in a decentralized approach, each robot makes its own decisions for a particular set of tasks. No central unit is needed. Some initial decomposition of the global scheduling may be imposed and robots can negotiate with others to make the best of coordination and solve conflicts dynamically. Furthermore, to improve the system robustness, error handling and system recovery are critical issues. According to Dias and Zink [7], in a multi-robot environment, system failure can occur in three different ways: (1) communication failure; (2) partial robot malfunctioning; and (3) robot death. The scheduling algorithm should take these situations into consideration.

Based on the above observation, we propose a hybrid task scheduling algorithm, where each robot makes its own decision through communicating with others as well as checking a global status queue to improve the coordination efficiency. This algorithm avoids unnecessary communication by broadcasting global information which is in everybody's interest and meanwhile limits specific information which is in interest of some specific robots only. Therefore, the proposed algorithm takes advantage of centralized approaches to improve the overall efficiency and distributed approaches to reduce the communication overhead. To improve the system robustness under dynamic environment, instead of making each robot to adapt to some unexpected tasks which may be beyond its own capability due to changed environment, the robot would send help signals to those who can handle the tasks. In addition, by tracking the communication signal that it has sent and expected to receive, each robot would dynamically allocate the tasks to robots which are capable and most available. This feature makes the system robust against communication failures and robot failures.

The paper is organized as follows. Section II introduces background and related work in the field of task allocation algorithms for multi-robot systems. Section III describes the problem statement. Section IV proposes a real-time dynamic task allocation algorithm for heterogeneous multi-robot systems. Extensive simulation results are discussed in Section V. The paper is concluded by Section VI.

2 Related Work

Increasing amounts of research have been conducted in the area of dynamic task scheduling for multi-robot systems. One of the easiest approaches to work in dynamic task assignment is trial and error method, where all robots will try the same task one by one until the perfect match is found. This method is very inefficient and time consuming. James McLurkin [8] proposed three different methods of task assignment in robot swarms: random-choice algorithm, which is extremely communication extensive and inefficient, card-dealer's algorithm, which assigns tasks to individual robots sequentially, using minimal communications but a great deal of time, and tree-recolor algorithm, which is a compromise between extreme-communication and card-dealer's, balancing communications use and running time.

Ashely and Ramprasad [9] proposed a behavior-based planning algorithm for multi-robot systems, where each robot predicts the behavior of its companion and proceeds for further steps. The main idea of this method is that the robot should not try to adapt to the situation but instead should directly transfer that task to an associated robot who can handle that situation. Brumitt and Stentz [10] proposed a dynamic mission planning for multi-robot systems in a dynamic environment, where the planning system dynamically reassign robots to goals in order to continually minimize the time to complete the mission. Trade-offs between robot's traveling cost and running cost of mission planner has to be balanced. Smith and Davis [11] proposed a contract net protocol, where the collection of nodes (robot) can be represented as a contract net. There are many auction based methods available for handling dynamic task allocation and MURDOCH [3][4] is a popular one among them, which uses contract net protocol as its communication protocol. Generally, distributed systems rely on fitness based actions and negotiation protocols. MURDOCH uses publish/subscribe messaging for distributed control of multi-robot systems.

A task-assignment architecture was proposed in [12] for cooperative transport by multiple mobile robots in an unknown static environment, which consists of two real-time planners: a priority-based task-assignment planner and motion planners based on short-time estimate. This method is also compared with Stillwell's algorithm in [13], where homogenous robots are ant like objects who try to move a big piece of food from one place to their nest. Each of them tries to contribute in the most efficient way. The scheduler proposed in [14] is one example of greedy decentralized schedulers. Generally, these kinds of co-operative search approaches are efficient and robust in applications like military scouting and automatic trash collection. A novel emotion-based recruitment approach was proposed in [15] for a multi-robot task allocation problem. Affective recruitment is tolerant of unreliable communication channels, and can find better solutions than simple greedy schedulers.

3 Problem Statement

A simplified proof-of-concept task environment, as shown in Fig. 1, is divided into 3 different sub-areas: high-pressure sub-area, intensive-light sub-area, and smoking sub-area. Different types of robots are defined based on their capabilities. For exam-

ple, *P type robot*, which is only capable of working in high-pressure subarea, but not the other two subareas. Here *P* stands for pressure. Similar definitions are applied to *L type robot* and *S type robot*. Some robots may have capabilities suitable for multiple sub-areas, such as *PS type* or *LS type*. In order to implement the assigned missions in a more efficient manner, task preemption is not allowed among the robots in the proposed task allocation approach. It is assumed that robots are autonomous, are able to localize themselves within the environment, can avoid obstacles and plan path to a destination.

Consider that we have *N* heterogeneous robots and *M* different tasks randomly distributed in different sub-areas. Here task is a conceptual terminology, which can be defined as various physical jobs, such as trash can collection, de-mining, transportation, construction, or assembling. The robots are expected to move to the position where the tasks are located and process the task. The environment can be as simple as Fig, 1(a) 1 or as complex as Fig. 1(b), which is unknown to robots. However, it is assumed that each robot has on-board sensors capable of detecting different subareas. Initially, if some predefined tasks have been stored in a robot, it will move to those tasks. If no predefined task exists, robots would randomly move around to search for the tasks in the environment. The requirement of these tasks may be changed due to dynamic environment. The objective of this project is to develop an efficient task scheduling algorithm among heterogeneous robots under dynamic environment so that all of the tasks would be completed as soon as possible meanwhile cost (i.e. power consumption) can be reduced as low as possible.

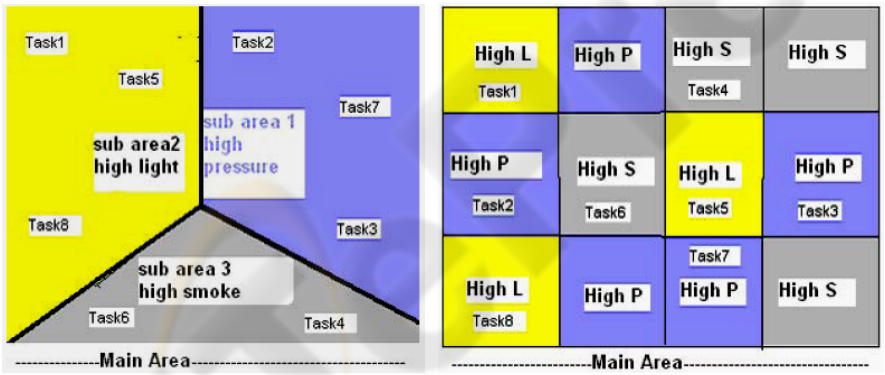


Fig. 1. Possible task environments.

4 A Hybrid Approach

To tackle this scheduling problem, a hybrid centralized and decentralized method is proposed, where each robot makes its own decision, communicates with others to share task information, as well as to check a global task status queue to improve the coordination efficiency. The architecture of the approach is shown in Fig. 2.

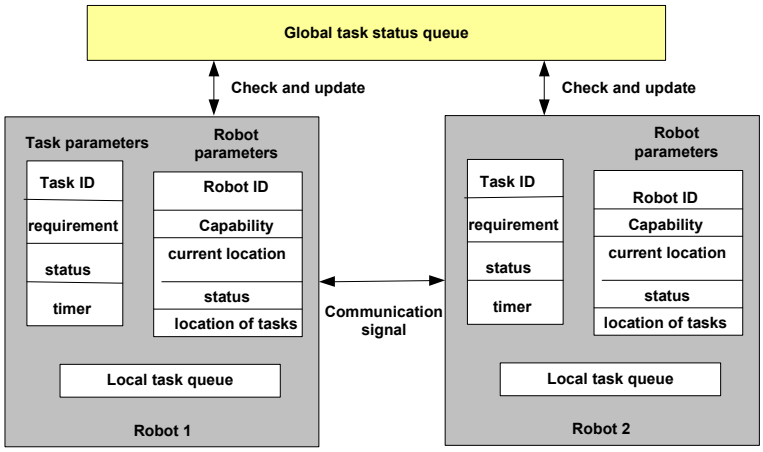


Fig. 2. The architecture of the hybrid approach.

Each robot has a local database to keep all the information it required to make decisions. This database structure, as shown in Fig.2, includes three major parts: robot parameters, task parameters, and local task queues. Robot parameters consist of robot ID, capability vectors of all robots, current locations of all robots relative to a reference coordinate system, status of all robots, locations of all tasks. Task parameters consist of task ID, task requirement, task status, and task timer.

Robot ID is a unique identification number for each robot. Capability is robot's ability to perform a task, which is a combination of different sub-areas represented by a capability vector. Robot status consists of free (its local task queue is empty), busy (some tasks are in its local task queue), or failed. Task ID is represented by its physical location in a reference coordinate. Task requirements depend on the sub-area where the task located. Task status shows whether the task is in progress, completed, in-trouble, or time-out. Task timer is used to track how long the task has been processed. To prevent the system to be hanged by one task forever, if the processing time is greater than a predefined threshold, time-out status would be labeled on the task.

Local task queue keeps a list of tasks a robot will perform sequentially. These tasks may be some predefined tasks before the system starts, or tasks detected or reassigned on the fly. Once the robot finishes its first task in its task queue, it would remove the finished task and go to next one until the last task in the queue. Once this queue becomes empty, robot will start moving randomly to search for a new task. Since it is difficult to predict all tasks in advance, some predefined tasks may not be appropriate for a robot anymore under dynamic environment. If a robot finds out that it is difficult for it to process a task in its local queue during execution, it would send help request to those robots whose capabilities match the task requirement. If help responses are received, the robot will assign the task to the responded robot, and delete it from its local queue. Meanwhile the responded robot would add that task in its local queue.

Global task status queue is a queue in which robot keeps the information about all tasks being processed or completed. The main purpose of this global task status queue is to prevent any unnecessary redundancy among robots to process the same task.

This global queue will be updated by all robots whenever task status has been changed.

Initially, all robots move around randomly in the environment searching for tasks. When a robot detects a task, it checks the requirements of the task first. If the requirements of the task match with its own capability, then the robot would perform the task and update the global task status queue. When a robot completes a task, it would update the global task status queue about its current task state. The states of a task include beginning, completion, in-trouble, and time-out. This global status of tasks is stored in the global task status queue for all robots. Whenever a robot needs help, it only broadcasts helps to those capable robots instead of everyone. Here a trade-off between memory capacity of robots and communication overhead among robots has to be made. The system stops when the global task status queue is filled up with all tasks with status of completion.

If multiple robots respond to the help requesting signal, the helper needs to make decision which robot to pick. On the other hand, if a robot was selected by more than one helper, it also need to make decision which task to select (if more than one responded robots) or which task to put into its local task queue first (if only one robot can do these tasks). A fitness function is required for this decision making. Here, a auction-based method is applied, which is defined as follows:

$$F(c_i, s_i, d_i) = k\frac{f(c_i \mid t_j) f(n_i)}{d_i}, i = 1,2,..., N, j = 1,2,..., M. \quad (1)$$

Where c_i, d_i and n_i represent the capability, distance from the current task location, and number of tasks in local queue of robot *i*, respectively. t_j represents the task types, and *k* is a scale factor. $f(c_i \mid t_j)$ is a matching function of the capability of robot *i* related to the type of task *j*, which is defined as follows:

$$f(c_i \mid t_j) = \begin{cases} 1, & \text{if the robot capability matches the task type} \\ 0, & \text{otherwise} \end{cases}.$$

And $f(n_i)$ is defined as follows:

$$f(n_i) = \begin{cases} 1, & \text{if } n = 0 \\ 1/n, & \text{otherwise} \end{cases}$$

In other words, if all of the responding robots are busy, the numbers of tasks in their local task queues are compared. The smaller the task number in queue, the higher the possibility of the corresponding robot would be selected as the helper.

Since robots need share task information with others, a specific communicate protocol is required for this application. Basically, four types of signal frames are defined in the communication protocol, (1) help requesting signal frame; (2) help responding signal frame; (3) help accepting signal frame; and (4) global task updating signal frame. The detailed frame definitions are shown in Fig. 3. When a help seeking robot receives help responding signal, it would send help accepting signal back to the selected robot. If responder robot is busy at that time, it would add that task in its local task queue and continue working on its current task. Global task updating signal is used to update the global task queue.

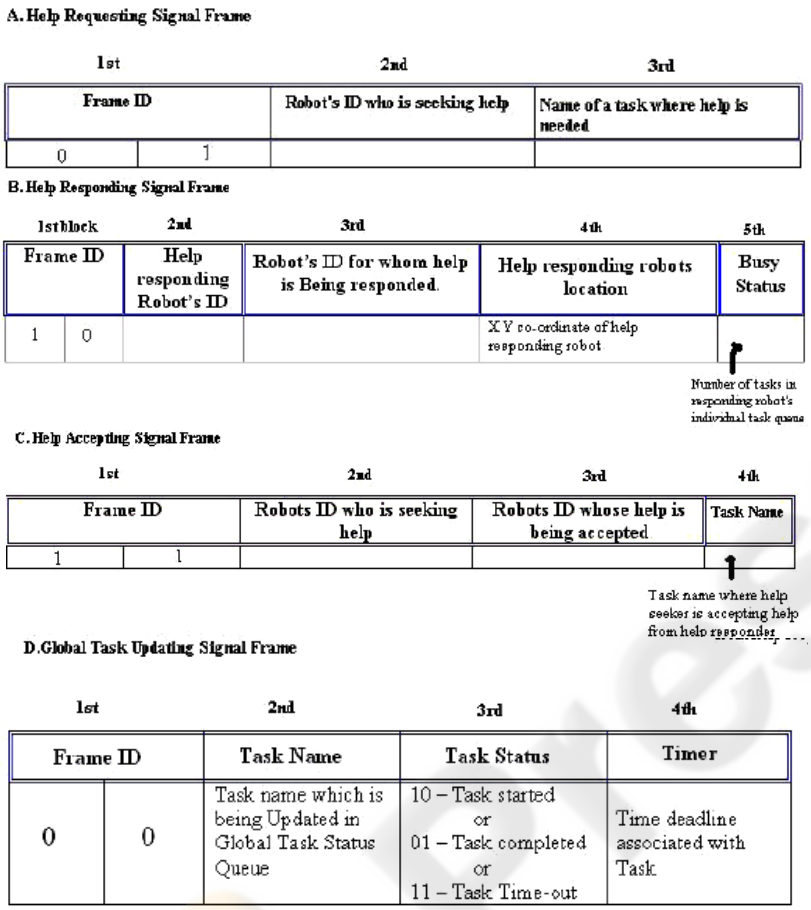


Fig. 3. Communication protocols among robots.

5 Simulation Results

To evaluate the proposed algorithm, a simulator is developed using C/C++ language under Windows environment, and a snapshot of the simulation screen is shown in Fig. 4. Six robots are employed in the simulation including 2 P-type (represented as Rp1 and Rp2), 2 L-type (RL1 and RL2), and 2 S-Type (Rs1 and Rs2). Eight tasks are generated, which are represented by the location coordinate within a reference frame. T1(X1,Y1), T2(X2,Y2) and T3(X3,Y3) are in high-pressure sub-area, T4(X4,Y4), T5(X5,Y5) and T6(X6,Y6) in smoking sub-area, and T7(X7,Y7) and T8(X8,Y8) in intensive-light sub-area. The local task queue is located on top-left, and the global task status queue is listed on top-right. The bottom-left part indicates the communication signals sent or received by robots. Various geometric shapes in the task environment represent static obstacles.

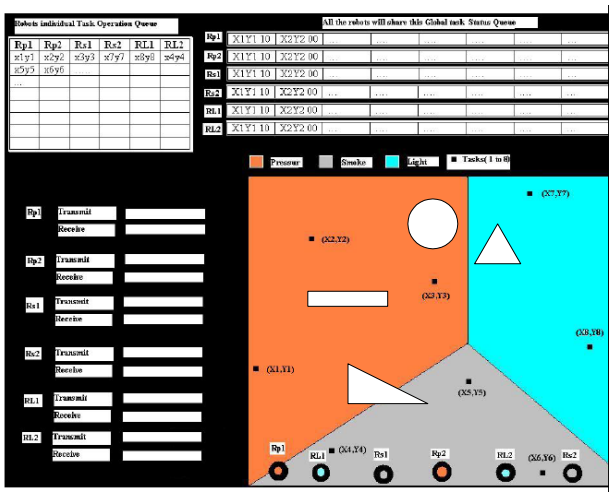
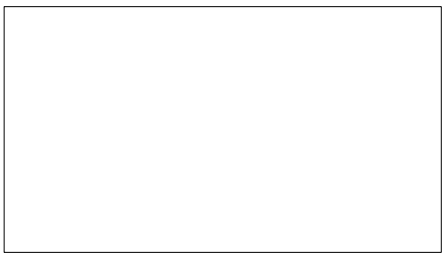


Fig. 4. A snapshot of the simulation screen.

Most task allocation problems among robots have applied group-wide broadcast communication to share the information and negotiate with team members. This kind of broadcast communication usually requires more communication overhead, power, time and cost, especially for a heterogeneous team where only some specific members can conduct specific types of tasks, not all of them. In our algorithm, instead of broadcast to everyone, the help information is only broadcasted to those which have the capability for the task. The communication cost comparison results are shown in Fig. 5, where the communication cost of our approach (i.e. 23 times) has been significantly reduced compared with the group-wide broadcasting method (i.e., 35 times). Communication overhead is directly proportional to task processing time and power consumption. In other words, our approach would be more power efficient and spend less time to finish the tasks than the group-wide broadcasting method. For this simple example with 6 robots and 8 tasks, the time required to finish all tasks are shown in Fig. 5(c).

A simple auction-based method using broadcasting is applied for comparison, where robots randomly search for tasks and broadcast the task information to all team members. If one robot needs help and more than one response received, the robot which is closest to the task will be selected. Four cases of different task distributions are designed in the simulation, where 8 task locations are re-distributed in different cases. The time required to finish all tasks under different task configuration cases are recorded and shown in Fig. 6. The proposed algorithm obviously outperforms the random searching one. The proposed method selects the helper robot not only depends on its current distance to the task, but also its current status. If there is a long list of tasks in its local task queue of a robot, even if it is closest one to the task, it may end up selecting other robot with a much shorter list of tasks in local queue.



6 Conclusions and Future Work

In this paper, a hybrid task scheduling approach has been proposed, which significantly reduces communication overhead while improving the overall system performance through dynamic task allocation. This algorithm avoids unnecessary communication by broadcasting global information which is in everybody's interest and meanwhile limits specific information which is in interest of some specific robots only. Each robot would dynamically allocate a task which is difficult for itself to other capable and most available robots, and keeps tracking the help requests, which makes the system robust against communication failures and robot failures. Simulation results show robot communication overhead can be significantly reduced, which automatically leads to reduction of power consumption and time consumption. In our future work, more dynamic situations will be considered, such as malicious agents, dynamically adding to or removing agents from the current team, global update failures. Furthermore, the method will be implemented to a real-world multi-robot system, where robot dynamics, kinematics, robot-robot interaction and sensors would have to be considered.

References

1. W. A. Halang, R. Gumzej, M. Colnaric, and M. Druzovec, "Measuring the performance of real-time systems", Journal of Real-Time Systems, 2000, 18(1):59-68.
2. Vail and Veloso, "Dynamic multi-robot coordination," *Multi- Robot Systems: From Swarms to Intelligent Automata, vol*. II, pp. 87–100, 2003.
3. B. Gerkey and M. J. Mataric, "Murdoch: Publish/subscribe task allocation for heterogeneous agents," *Proceedings of Autonomous Agents,* Barcelona, Spain, June 3-7, pp. 203–204, 2000.
4. D. Rus, S. Singh, S.-V. B. Heidelberg, B. P. Gerkey., and M. J. Matarik, "Principal communication for multi robot task allocation," *Experimental Robotics VII,* LNCIS, pp. 353–362, 2001.
5. L. E. Parker, "Alliance: An architecture for fault tolerant multirobot cooperation," *IEEE Trans. Robot. Automat, vol. 14,* p. 220240, Apr 1997.
6. L. E. Parker, "Task-oriented multi-robot learning in behavior-based systems," *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems,* 1996.
7. M. B. Dias, M. Zinck, R. Zlot, and A. Stentz, "Robust multirobot coordination in dynamic environments," *Proceeding on DIAS,*.2004.
8. J. McLurkin and D. Yamins, "Dynamic task assignment in robot swarms,"*IEEE Proceedings of Robotics: Science and Systems, June 2005*. MIT Computer Science and Artificial Intelligence Lab.
9. A. W. Stroupe, R. Ravichandran, and T. Balch, "Value-based action selection for exploration and dynamic target observation with robot teams," *Proceedings of the IEEE Conference on Robotics and Automation,* 2004.
10. B. Brumitt and A. Stentz, "Dynamic mission planning for multiple mobile robots," *Proceedings of the IEEE International Conference on Robotics and Automation,* 1996.
11. R. Davis and R. G. Smith, "Negotiation as a metaphor for distributedo," *Conf. Artificial Intelligence Laboratory, Massachusetts Institute, of Technology, Cambridge, MA* 02139, ETATS-UNIS, vol. 20, no. 1, p. 63109, 1993.

12. N. Miyata, J. Ota, T. Arai, and H. Asama, “Cooperative transport by multiple mobile robots in unknown static environments associated with real-time task assignment,” *IEEE transactions on robotics and automation,* vol. 18, no. 5, 2002.
13. D. J. Stilwell and J. S. Bay, “Toward the development of a material transport system using swarms of ant-like robots,” *Proc. IEEE Int. Conf. Robotics and Automation*, pp. 766-771, 1993.
14. C. Hsieh and R. Murray, “Experimental implementation of an algorithm for cooperative searching of target sites,” *Proceedings of American Control Conference,* 2005.
15. A. Gage, D. R. Murphy, D. K. Valavanis, and M. Long, “Affective task allocation for distributed multi-robot teams,” Center for Robot-Assisted Search and Rescue (CRASAR), TR2006-26.