A Learning-Based Semi-Autonomous Controller for Robotic Exploration of Unknown Disaster Scenes While Searching for Victims

Barzin Doroodgar, Yugang Liu, *Student Member, IEEE*, and Goldie Nejat, *Member, IEEE*

***Abstract*—Semi-autonomous control schemes can address the limitations of both teleoperation and fully autonomous robotic control of rescue robots in disaster environments by allowing a human operator to cooperate and share such tasks with a rescue robot as navigation, exploration, and victim identification. In this paper, we present a unique hierarchical reinforcement learning-based semi-autonomous control architecture for rescue robots operating in cluttered and unknown urban search and rescue (USAR) environments. The aim of the controller is to enable a rescue robot to continuously learn from its own experiences in an environment in order to improve its overall performance in exploration of unknown disaster scenes. A direction-based exploration technique is integrated in the controller to expand the search area of the robot via the classification of regions and the rubble piles within these regions. Both simulations and physical experiments in USAR-like environments verify the robustness of the proposed HRL-based semi-autonomous controller to unknown cluttered scenes with different sizes and varying types of configurations.**

***Index Terms*—Hierarchical reinforcement learning, rescue robots, semi-autonomous control, urban search and rescue.**

I. Introduction

SEARCH and rescue operations in urban disaster scenes are extremely challenging due to the highly cluttered and unstructured nature of these environments. Moreover, in some scenarios, the task of rescuing victims from collapsed structures can be extremely hazardous due to the instability of damaged structures, and/or the presence of dust, toxic chemicals, or radiation. Furthermore, accessing a collapsed structure sometimes requires entering voids, which may be too small or too deep for rescue workers and rescue dogs [1], [2]. To address these challenges, rescue robots are being developed to assist rescue workers in urban search and rescue (USAR) operations.

Current applications of rescue robots require a team of human operators to remotely guide them in a disaster scene [1], [3]. However, teleoperation of robots while also searching for victims in USAR environments can be a very stressful task, leading to cognitive and physical fatigue. Furthermore, human operators can have perceptual difficulties in trying to understand an environment via remote visual feedback. As a result, operators can suffer low levels of alertness, lack of memory and concentration, and become disoriented and lose situational awareness during these time-critical situations [4].

Autonomous controllers provide an alternative to teleoperated control for rescue robots, which eliminates the need of constant human supervision. However, deploying fully autonomous rescue robots in a USAR scene requires addressing a number of challenges. First, rescue personnel do not generally trust an autonomous robot to perform such critical tasks without any human supervision [2]. Second, rescue robots have very demanding hardware and software requirements. Namely, autonomous navigation can be very difficult due to rough terrain conditions; and victim identification can be particularly challenging due to the presence of debris, dust, poor lighting conditions, and extreme heat sources in disaster scenes [2].

To address the challenges and limitations of both teleoperated and fully autonomous control of rescue robots in USAR missions, recent efforts have been made toward developing semi-autonomous controllers that allow rescue robots to share tasks with human operators [5]–[9].

Alternatively, our work uniquely focuses on developing learning-based semi-autonomous controllers for rescue robots in USAR applications [10], [11]. In this paper, we present the overall development of our unique hierarchical reinforcement learning (HRL)-based semi-autonomous control architecture for a rescue robot to both explore cluttered environments and search for victims. The HRL-based controller allows a rescue robot to learn and make decisions regarding which rescue tasks need to be carried out at a specific time and if the robot or the human operator should perform these tasks to achieve optimal results. This decision making ability enables the robot to continuously learn from its surrounding environment, and its own previous experiences in order to improve its task performance in unknown disaster environments. A new direction-based exploration technique which directly takes into account terrain information is incorporated into the HRL controller to expand the search area of the robot via the classification of regions and the rubble piles within them.

Manuscript received October 21, 2013; revised March 7, 2014 and March 17, 2014; accepted March 18, 2014. This work was supported by the Natural Sciences and Engineering Research Council of Canada.

The authors are with the Autonomous Systems and Biomechatronics Laboratory, Department of Mechanical and Industrial Engineering, University of Toronto, Toronto, ON M5S 3G8 Canada (e-mail: barzin.doroodgar@utoronto.ca; yugang.liu@mail.utoronto.ca; nejat@mie.utoronto.ca).

II. Related Work

In this section, semi-autonomous controllers, HRL techniques and robot exploration methods, as applied to mobile robots, are reviewed briefly.

A. *Semi-Autonomous Control*

To date, the semi-autonomous controllers developed for mobile robotic applications have mainly focused on the shared control of navigation and obstacle avoidance tasks in varying environments. Recently, there has been a handful of semi-autonomous controllers specifically developed for control of rescue robots in USAR applications, focusing on the two main tasks of robot navigation and victim identification [5]–[9]. These semi-autonomous controllers can be classified based on a robot's level of autonomy.

Semi-autonomous controllers with fixed robot autonomy levels focus on such low level tasks as collision avoidance and motor control, allowing the human operator to concentrate on high level control and supervisory tasks such as path planning and task specification [12]. Compared to fully teleoperated control of robots, these semi-autonomous controllers have proven to be more effective for robot navigation and obstacle avoidance. However, they lack the required flexibility for more challenging problems such as control of rescue robots in rubble-filled USAR environments. For example, in the case of a robot getting physically stuck in a cluttered disaster scene, the human operator may have to take over the control of low level operations in order to assist the robot. Similarly, high level control may be required from the robot controller when the operator cannot perform these tasks due to task overload, loss of situational awareness, or communication dropout. To address these concerns, semi-autonomous controllers with variable autonomy provide a promising solution.

Semi-autonomous controllers with two different levels of robot autonomy were experimentally compared in [5]. The results suggested that a shared mode, where the robot provides the optimal path for navigation based on the operator's directional inputs, demonstrated the best performance. Using the shared mode, the operator is allowed to set the controller's level of autonomy beforehand. However, he/she is not able to change this level on the fly during a search and rescue operation, which may be needed in unknown environments if either the robot or the operator faces a situation where one needs assistance from the other.

Controllers presented in [6]–[9] solve this problem by providing on the fly adjustments of a robot's level of autonomy by the operator [6] or automatically by the controller [7]–[9] during USAR operations. In [6], an operator is in charge of dividing a scene into regions and prioritizing these regions, while the robot explores each region to find victims. The operator can take over the control of the robot at any time during the operation. In [7], three semi-autonomous control modes were proposed for a rescue robot to distribute tasks between an operator and the robot. The specific mode to implement was determined automatically by the robot based on the manner in which the operator was interacting with the overall system during a USAR operation. In [8], the semi-autonomous controller presented in [5] was extended to include a mode suggestion system which uses information from the robot and the operator to determine when an autonomy mode suggestion should be made to the operator. The operator can then decide either to accept or decline the system's suggestions during USAR operations. In [9], the concept of a sliding scale autonomy system was introduced. The proposed approach could potentially create, on the fly, new levels of robot autonomy between existing preprogrammed autonomy levels by combining human and robot inputs using a small set of variables consisting of force fields, speeds and obstacle avoidance to determine a robot's movements.

The aforementioned research highlights the recent efforts in semi-autonomous control for USAR applications. While promising, the current controllers do not incorporate learning into their control schemes, which is essential to deal with unknown and unpredictable USAR environments. In this paper, we propose the use of a learning-based semi-autonomous controller to enable a robot to learn from its own prior experiences in order to better adapt to unknown and cluttered disaster environments.

B. *Hierarchical Reinforcement Learning for Robot Control*

There have only been a handful of cases where HRL is used in mobile robotic applications, for multiagent cooperation [13], mobile robot navigation [14], and robotic soccer [15].

In this paper, we explore for the first time, the use of the MAXQ HRL method within the context of a semi-autonomous controller for robot exploration and victim identification in USAR environments. In USAR applications, the environments of interest are unknown and cluttered, increasing the complexity of the learning problem. With an MAXQ approach, the overall search and rescue task can be decomposed into a series of smaller more manageable subtasks that can be learned concurrently. MAXQ has fewer constraints on its policies, i.e., mapping of states to possible actions, and thus, is generally known to require less prior knowledge about its environment [16]. This feature is advantageous and makes MAXQ suitable for unknown USAR environments. In addition, MAXQ can support state, temporal, and subtask abstraction.

State abstraction is essential in USAR applications since when a robot is navigating to a specific location only that particular location is significant, the reason why the robot is navigating to that location is irrelevant and should not affect the robot's actions. The need for temporal abstraction exists in USAR scenes due to the fact that actions may take varying amounts of time to execute as a result of the complexity of the scene and the location of a robot within the scene. Subtask abstraction allows subtasks to be learned only once, and the solution can then be shared by other subtasks. For example, local navigation can be shared by a subtask dedicated to globally exploring a USAR environment as well as a victim identification subtask, where the latter requires the robot to move toward objects of interest in order to determine if they could represent potential victims.

C. *Mobile Robot Exploration Strategies*

There has been extensive research into addressing the robotic exploration problem for unknown environments. In particular, to date, frontier-based exploration strategies have been developed for mobile robots to explore unknown environments. These techniques expand a known map of the environment by deploying either a single robot or a team of coordinated robots into target locations at the frontier of the map, i.e., the boundary between explored and unexplored regions of the map. A utility or cost function is normally utilized to determine where to deploy a particular robot.

In [17], the frontier-based exploration technique was introduced for a single robot, which determined the robot's next target location based on the shortest accessible path for the robot. A multirobot exploration strategy was adapted using the cost of reaching a frontier cell and the utility of that cell in [18]. The cost for a particular robot reaching a frontier cell was based on the probability of occupancy of each cell (i.e., the presence of an obstacle) along the path of the robot to the corresponding frontier cell, as well as the robot's distance to that target point. The utility of the cell was dependent on how many robots were moving to that particular cell. In [19], in addition to determining the nearest frontier point to a robot that provides maximum coverage of unknown space, the deployment locations were chosen such that a robot at one of these locations is visible to at least one other robot.

The majority of these frontier-based exploration approaches have shown to work well in simulated and real laboratory and/or office environments. However, they cannot be directly applied to USAR-like environments as they do not consider the (rough) terrain conditions in USAR scenes. In this paper, we extend the frontier-based exploration approach to robots in USAR environments by incorporating terrain information in determining optimal exploration directions in order to promote maximum search of these cluttered rubble-filled scenes. Namely, we propose a direction-based exploration technique that can be used within our semi-autonomous HRL controller.

III. Proposed Learning-Based Semi-Autonomous Control Architecture

The proposed semi-autonomous control architecture is depicted in Fig. 1. Sensory information is provided by onboard robot sensors which include: 1) a real-time 3-D mapping sensor that provides 2-D and 3-D images of the environment; 2) a thermal camera that provides the heat signature of the surrounding environment; and 3) infrared sensors that provide proximity information of a rescue robot's surroundings. The 2-D and 3-D images provided by the 3-D mapping sensor are used by the *simultaneous localization and mapping (SLAM)* module to identify and match 3-D landmarks in the USAR scene and create a 3-D global map of the environment. The details of the 3-D sensor and the *SLAM* technique are provided in [20].

The *HRI Interface* module provides the operator with the user interface needed for human control (*HC*) of a robot. The operator can use the interface to obtain sensory information from the environment and the robot, as well as the 3-D map in order to monitor or control the robot's motion. The *Deliberation* module provides the robot with the learning and decision making capabilities during semi-autonomous deployment. Since the robot is designed to be semi-autonomous, it is within the *Deliberation* module the robot primarily decides its level of autonomy. Namely, using MAXQ the robot is able to learn and make decisions regarding which tasks should be carried out at a specific time, and whether the robot itself or the human operator should execute them for optimal results. If *HC* is prominent, the decision making within this module is made by the operator via the *HRI Interface* module. The *Robot Actuators* module takes the robot actions provided by the *Deliberation* module and translates them into appropriate motor signals.

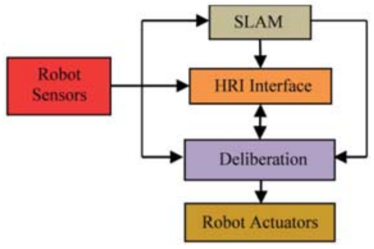

Fig. 1. Semi-autonomous control architecture.

The next section presents the detailed design of the MAXQ HRL technique utilized by the *Deliberation* module for robotic exploration and victim identification in a USAR scene.

IV. Implementation of MAXQ-Based HRL for Semi-Autonomous Control in USAR Environments

The MAXQ technique works by decomposing a given Markov Decision Process (MDP), M, into a finite set of subtasks $\{M_0, M_1, \ldots, M_n\}$, which define the MAXQ hierarchy [16]. Herein, M_0 represents the root subtask which defines the overall problem, and is further decomposed into subtasks M_1–M_n. For every subtask M_i, a policy, π_i, is defined which maps all possible states of M_i to a child task. The child task can be either a primitive action or another subtask under M_i to execute. Subsequently a hierarchical policy, π (a set containing the policies for all subtasks), is defined for the entire task hierarchy. Also, a projected value function is stored for every state and action pair in all subtasks. The projected value function is defined as the expected cumulative reward of executing policy π_i in subtask M_i, as well as all the policies of the subtasks that would be executed under M_i until M_i terminates.

A. *MAXQ Task Hierarchy*

The proposed MAXQ task hierarchy for our semi-autonomous control architecture is shown in Fig. 2. The *Root* task represents the overall USAR task to be accomplished by the robot—finding victims while exploring a cluttered USAR scene. This *Root* task can be further decomposed into four individual subtasks defined as: 1) *Navigate to Unvisited Regions (NUR)*; 2) *Victim Identification (VI)*; 3) *Navigate (NG)*; and 4) *HC*. The following subsections provide a detailed discussion of each of the individual modules that make up the overall task hierarchy.

1) Root Task: The *Root* task defines the overall goal of a rescue robot in a USAR mission. The MAXQ state function of the *Root* task is defined as $S(V, L_R, M_{xyz})$. V denotes the presence of potential victims in the environment. L_R is the

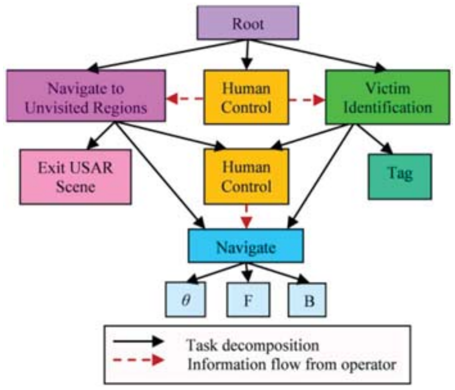

Fig. 2. MAXQ task graph for our semi-autonomous controller.

robot's location with respect to the global coordinate frame (defined to be at the location at which the robot enters the scene), and M_{xyz} represents the 3-D map of the USAR scene the robot is exploring. L_R and M_{xyz} can both be obtained by the *SLAM* module in the control architecture (Fig. 1).

2) NUR Subtask: The purpose of this subtask is to allow the robot to explore unvisited regions within disaster environments. The state definition for this subtask is $S(L_R,M_{xyz})$, where L_R and M_{xyz} have the same definitions as above. *Exit USAR Scene* is a primitive action used by the *NUR* subtask to trigger the end of exploration and guide a rescue robot out of the scene.

In order to efficiently explore an unknown USAR environment, we have developed a direction-based exploration technique to be used by this subtask. This technique is an extension to the frontier-based exploration strategies reported in [17]–[19], focusing on finding and exploring new regions within the environment in order to expand the search area of the robot. Our main addition is the explicit incorporation of the cluttered terrain information of USAR environments. To effectively choose an exploration direction, the 3-D map of the scene is decomposed into a 2-D grid map consisting of an array of equally sized cells, where each cell categorizes the status of a predefined area of the real environment. The depth profile of the terrain of a cell obtained from the front-facing 3-D mapping sensor is stored and used as a measure of traversability. Furthermore, travel distance and coverage are also taken into account in our proposed exploration technique in order to determine an appropriate exploration direction for the robot. To further address the cluttered and uncertain nature of USAR environments, a direction-based strategy, instead of a path to target approach, is utilized; and a learning technique is implemented to aid the robot to locally navigate these rubble-filled scenes. The proposed direction-based exploration technique is detailed in Section V.

3) VI Subtask: The aim of this particular subtask is to identify victims in cluttered USAR scenes. The state of the subtask is defined as $S(L_{V/R}, M_{xyz})$, where $L_{V/R}$ represents the locations of potential victims in the scene with respect to the robot's location as obtained by the 3-D mapping sensor. When a victim is identified in the scene, the primitive action *Tag* is executed in order to tag the global location of the victim within the 3-D map of the scene, M_{xyz}. The *Navigate* subtask

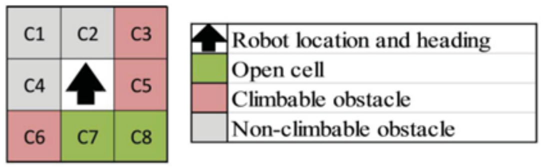

Fig. 3. Representation of the local environment surrounding a robot.

is executed when the robot is required to move closer to a potential victim, in order to enable a positive identification.

For our proof-of-concept approach, a thermal camera is used to identify victims in the scene. A probability-based approach is used, where the probability of a victim being present, $p = f(h, s)$, is based on two features: the existence of human thermal heat signature, h (represented as a binary value), and the size, s, of this heat source. Namely, the probability increases with respect to the presence of such heat regions and the size of these regions in thermal images. For example, if the probability of a victim is high (i.e., $p > 0.7$), the robot automatically tags the victim, if the probability is low (i.e., $p < 0.3$), the robot can dismiss the possibility of a victim being present. A probability between 0.3 and 0.7 represents the robot's uncertainty regarding the presence of a victim and hence, *HC* is requested.

4) NG Subtask: The goal of the *NG* subtask is to perform local navigation and obstacle avoidance. The state definition of the *Navigate* subtask is defined as $S(C_i, D_E, D_{xy}, L_{V/R})$, where C_i, $i = 1$ to 8, represents the grid map information for the eight surrounding cells of the robot, as shown in Fig. 3, D_E corresponds to the desired exploration direction (determined by the direction-based exploration technique in the *NUR* module), and D_{xy} contains the depth profile information of the rubble pile in the surrounding environment. There are two conditions for which *NG* can be performed: local scene exploration or navigate to a potential victim location. When navigating to a potential victim location rather than just exploring a scene, $L_{V/R}$ (relative location of a potential victim to the robot) is used in the *NG* module to reach the target victim location. As soon as the victim location is reached, the *NG* subtask terminates and allows the *VI* subtask to perform the task of identifying victims.

Local exploration is performed by the *NG* subtask by utilizing the information of the eight adjacent cells, C_i, to the robot's current cell location (Fig. 3). Based on the status of the robot's surrounding cells, the optimal primitive actions to move the robot forward (F) or backwards (B), or to rotate the robot by an angle θ within the environment are provided to the robot's low-level controller to appropriately convert into motion commands. Hence, the *NG* subtask can search its local environment by exploring new cells and performing obstacle avoidance. Immediate rewards are given to the *NG* subtask for the choice of primitive actions implemented, in order to encourage desirable actions such as moving to adjacent unvisited areas in a local environment, successfully avoiding dangerous situations such as collisions with obstacles, and moving in the desired exploration direction.

For effective navigation in highly cluttered environments, a rescue robot will need to be able to climb over climbable rubble piles. D_{xy} represents a 2-D array that contains the depth

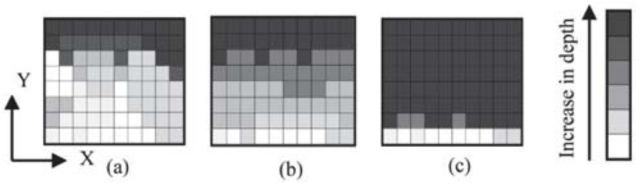

Fig. 4. 2-D D_{xy} array for classification of rubble pile profiles showing (a) non-climbable rubble pile with a relatively large slope and height, (b) climbable rubble pile with a gradual slope, and (c) drop (defined in the non-climbable category) with an abrupt increase in the depth values. x-axis corresponds to the width and y-axis corresponds to the height of the 3-D sensor's field of view.

profile information of a rubble pile that appears in the robot's path at any given time. This information is obtained from the 3-D map generated from the sensory information provided by the 3-D mapping sensor and is conveniently represented in D_{xy}. Fig. 4 shows examples of the depth profile information of non-climbable and climbable rubble piles in a robot's forward view as represented by D_{xy}. The slope and smoothness of a rubble pile is categorized by fitting a plane to its 3-D sensory data using a least squares method. The combination of these two parameters determines if the rubble pile is climbable or non-climbable. The non-climbable obstacle category includes terrain too rough for the robot to traverse and sudden drops in the height of the rubble pile i.e., edge of a cliff, which can be detected by an abrupt increase in the depth values in the D_{xy} array (Fig. 4(c)).

5) HC Subtasks: The purpose of the *HC* subtasks is to pass over the control of the robot to the human operator in case the robot is unable to perform any of its tasks autonomously. There are two levels in the task hierarchy in which *HC* can be used (Fig. 2). The objective of the 2nd level *HC* subtask is to have a human operator share with the robot the tasks pertaining to the *VI* and/or *NUR* subtasks.

With respect to the *VI* subtask, the operator can assist in identifying victims in the scene and tagging their locations. The operator utilizes the corresponding 2-D and 3-D information of the robot's surrounding cells in the grid map, as obtained in real-time from the 3-D mapping sensor, to tag the location of a victim. With respect to the *NUR* subtask, the operator can assist in choosing preferred search directions to explore.

The 3rd level *HC* subtask allows for operator control of the local navigation subtask of the robot. If the robot is unable to navigate autonomously due to noisy or insufficient sensory data, or it has gotten physically stuck due to the highly cluttered nature of the environment, this *HC* subtask allows the operator to take over the low level navigation and obstacle avoidance tasks of the robot. While the control is passed over to the operator, the robot's sensory system still tracks the motion of the robot and updates the 3-D map of the scene.

For both levels, when *HC* is executed, the appropriate subtask continues to learn by observing the implemented actions of the operator via the information path and the outcomes that result from these actions (Fig. 2).

B. MAXQ-Based Robot Learning and Decision Making

In general, the action-value function (Q-value) for subtask M_i, under the policy π can be defined as [16]:

$$Q^{\pi}(i,s,a) = V^{\pi}(a,s) + C^{\pi}(i,s,a) \tag{1}$$

where $Q^{\pi}(i,s,a)$ represents the expected cumulative reward for subtask M_i of performing subtask or action a in state s and then following the hierarchical policy π until subtask M_i terminates. $V^{\pi}(a,s)$ and $C^{\pi}(i,s,a)$ are the projected value and completion functions of executing a in s, and are defined as:

$$V^{\pi}(a,s) = \begin{cases} Q^{\pi}(i,s,\pi_i(s)), & \text{if } M_i \text{ is composite} \\ \sum_{s'} P(s',|s,i)\, R(s',|s,i), & \text{if } M_i \text{ is primitive} \end{cases} \tag{2}$$

$$C^{\pi}(i,s,a) = \sum_{s',N} P_i^{\pi}(s',N\,|s,a)\,\gamma^N Q^{\pi}(i,s',\pi(s')) \tag{3}$$

where N is the number of transition steps from s to the next state s', and γ denotes a discount factor. P and R represent the corresponding probability transition function and expected reward function, respectively. For more details on the general MAXQ algorithm, the reader is referred to [16].

For the USAR problem at hand, the action-value function for the *Root* task can be defined with respect to each corresponding subtask in the hierarchy as follows:

$$\begin{aligned} Q(Root,s,NUR) &= V(NUR,s) + C(Root,s,NUR) \\ Q(Root,s,VI) &= V(VI,s) + C(Root,s,VI) \\ Q(Root,s,HC) &= V(HC,s) + C(Root,s,HC) \end{aligned} \tag{4}$$

where $V(NUR, s)$, $V(VI, s)$, and $V(HC, s)$ denote the projected value functions of executing the corresponding subtasks/primitive action in state s. $C(Root, s, NUR)$, $C(Root, s, VI)$, and $C(Root, s, HC)$ are the completion functions, which represent the discounted cumulative reward of executing the corresponding subtasks/primitive actions.

The value functions and completion functions for the *Root* task can further be defined as:

$$\begin{aligned} V(NUR,s) &= Q(NUR,s,\pi_{NUR}(s)) \\ V(HC,s) &= P(s'|s,HC)\,R(s'|s,HC) \\ V(VI,s) &= Q(VI,s,\pi_{VI}(s)) \end{aligned} \tag{5}$$

$$\begin{aligned} C(Root,s,NUR) &= \sum_{s'\in S_{Root},N} \{P_{Root}(s',N|s,NUR)\,\gamma^N \\ &\qquad Q(Root,s',\pi_{Root}(s'))\} \\ C(Root,s,HC) &= \sum_{s'\in S_{Root},N} \{P_{Root}(s',N|s,HC)\,\gamma^N \\ &\qquad Q(Root,s',\pi_{Root}(s'))\} \\ C(Root,s,VI) &= \sum_{s'\in S_{Root},N} \{P_{Root}(s',N|s,VI)\,\gamma^N \\ &\qquad Q(Root,s',\pi_{Root}(s'))\} \end{aligned} \tag{6}$$

where $\pi_{Root} \in \{NUR, HC, VI\}$, $\pi_{NUR} \in \{EUS, NAV, HC\}$ and $\pi_{VI} \in \{Tag, NAV, HC\}$ are the policies for the corresponding subtasks, where *EUS* and *NAV* denote the *Exit USAR Scene* primitive action and *Navigation* subtask; and S_{Root} is the state space for the *Root* task.

The value function V and completion function C for each primitive action are updated immediately after receiving the reward R, while the value functions for the *Root* task or subtasks are updated recursively through the hierarchy until they terminate. The MAXQ HRL-based learning algorithm updates the value functions in order for them to converge to the unique value functions of the recursively optimal policy.

To determine the probability transitions between subtasks and actions, an epsilon-greedy policy [21] is used during the learning process. Namely, an action is chosen randomly, with probability ε, to promote exploration of all available actions, and the optimal action, i.e., action with the highest Q-value, is chosen with probability 1-ε. The value of ε is reduced as the system is trained. The advantage of this policy is that it provides sufficient exploration and exploitation for the learning algorithm, and thus, ensures convergence to optimal solutions.

Positive rewards are given to encourage transitions from the robot's current state to desirable states. For example, if the robot exits the USAR scene after having explored the entire scene, a positive reward of +10 is given to the *NUR* subtask. For the *VI* subtask, a positive reward of +10 is given for correctly tagging a new victim found in the scene. For the *NG* subtask, the reward is based on three main factors: local exploration, obstacle avoidance, and the global exploration direction. Within this subtask, positive rewards are given to encourage desirable actions such as moving to adjacent unvisited areas in the local environment, successfully avoiding dangerous situations such as collisions with obstacles, and moving in the desired exploration direction. For example, moving into an unvisited region in the desired global exploration direction is given a positive reward of +15 as it directly relates to the main objective of having the robot explore an unknown environment, while avoiding an obstacle is given a positive reward of +10. In addition, a large positive reward of +100 is also given to the *Root* task when the overall USAR task is completed to successfully reward the overall goal being achieved.

Negative rewards are given when a transition is made from the robot's current state to an undesirable state. For example, a negative reward of -10 would be given to *NUR* if *Exit USAR Scene* is executed instead of *NG* when there is still known regions to explore. In the *VI* subtask, a negative reward of -10 would be given if *Tag* is executed when no new victims were identified in the scene. In *NG,* negative rewards would be given for executing commands that lead to collisions (-20) or navigating into already visited areas (-1) of a USAR scene. Colliding with an obstacle may damage the robot itself; hence there is a higher cost associated with its reward than when navigating into an already visited area which only affects time efficiency. With respect to *HC*, similar positive and negative rewards are given based on the subtask that the operator is involved in, allowing the robot to learn from the operator's actions. The transition awards are summarized in Table I.

The reward values in Table I are chosen based on the following considerations: 1) the rewards should be selected to encourage transitions from the robot's current state to desirable states and to avoid transitions to undesirable states and 2) the potential benefits and costs of states should be considered in the magnitudes of the rewards. In general, the magnitudes of the negative and positive rewards for the subtasks mainly influence the speed of convergence to the optimal policies and do not affect the overall trend toward convergence.

TABLE I
TRANSITION REWARDS FOR MAXQ FOR USAR PROBLEM

Subtask	Robot state transition	Reward
Root	The overall USAR task is completed.	+100
Navigate to Unvisited Regions	Exit the USAR scene after successfully exploring the entire scene.	+10
Navigate to Unvisited Regions	Exit USAR scene when there is still known regions to explore.	-10
Victim Identification	Correctly tagging a victim found in the scene.	+10
Victim Identification	Tag is executed when no new victim is identified in the scene.	-10
Navigate	Moving into an unvisited region in the desired global exploration direction.	+15
Navigate	Avoiding an obstacle.	+10
Navigate	Obstacle collision.	-20
Navigate	Navigating into already visited area.	-1

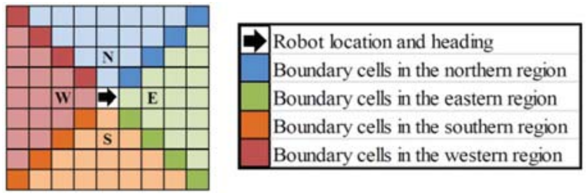
N
W
E
S
Robot location and heading
Boundary cells in the northern region
Boundary cells in the eastern region
Boundary cells in the southern region
Boundary cells in the western region

Fig. 5. 2-D grid map of the four regions surrounding the robot.

V. DIRECTION-BASED EXPLORATION TECHNIQUE

To address the cluttered and unstructured nature of USAR environments, a direction-based exploration strategy is proposed for the *NUR* module in order for a rescue robot to find and explore new regions within the environment. Mainly, the exploration technique defines a search direction for a rescue robot to follow in order to explore unknown regions. The exploration method determines one of four universal search directions, North, South, East, or West, for the robot to explore in order to navigate into unvisited regions in the scene. The chosen direction for exploration, D_E, is then incorporated into the rewarding system of the *Navigate* module which focuses on implementing the primitive actions necessary to locally navigate the robot in this defined search direction through the rubble. The search direction is updated as necessary in order for the robot to search new regions. In general, our proposed direction-based technique is more robust to map uncertainty than direct path planning techniques that require accurate scene representations in order for a robot to reach target locations and it does not require the robot to move in wide open spaces, which are not always available in USAR scenes.

To effectively choose an exploration direction, the exploration technique uses the information in the grid map of the environment to locate potential cells within the map that have yet to be explored as well as cells which represent the frontier of the known map of the scene, in order to expand the robot's knowledge about the environment. The 2-D grid map is divided into four regions, where each region represents one of the search directions, as shown in Fig. 5.

The diagonal cells with respect to the robot's current location in the known map are used to determine the boundaries between the four search regions and are defined with respect to the robot's current location in the known map. Each region encompasses only one set of boundary cells defined to be in the counter-clockwise direction from the region.

Prior to evaluating an exploration direction, the cells in each region are categorized into two types: explored and unexplored cells. This classification is based on whether the robot has visited a particular cell or its adjacent cells. If the robot has already explored a cell, then the terrain or the presence of victims in that cell will be known and can be categorized accordingly. Based on the sensing information of the robot, the explored/unexplored cells can be further identified as follows.

i) Obstacle: Obstacles can be defined to be, for example, rubble piles, which are categorized as known or unknown obstacles. Known obstacles are further classified into climbable (i.e., robot can navigate over them) and non-climbable obstacles. The climbable obstacles category is further divided into visited and unvisited climbable obstacles. Unknown obstacles are those detected in the robot's surrounding cells, however, available sensory information is not enough to classify them further.

ii) Open visited: An open and traversable area that has been visited previously by the robot.

iii) Open unvisited: This cell has not been visited by the robot but has been detected as an obstacle-free area. Herein, a distinction is made between visited and unvisited open space for USAR environments since exploring an unvisited open cell can potentially lead to the exploration of unknown regions of the scene beyond this cell, which may not be detectable until the robot physically enters this particular cell due to the cluttered nature of the environment.

iv) Victim: The cell contains a human victim.

v) Unknown: The cell information is unknown, since the cell has not been explored and no sensory information is available.

The cells categorized as open visited, known non-climbable, visited climbable obstacles or victim cells are defined to be explored cells, whereas unexplored cells include the unknown, unknown obstacles, open unvisited, and unvisited climbable obstacle cells. Unexplored cells can add to the robot's knowledge of the scene and lead to exploring new regions. On the other hand, revisiting already explored cells may not necessarily provide any new information about the environment. Therefore, unexplored cells are considered as cells of interest in robot exploration and assist in determining exploration direction. To determine the robot's exploration direction in the scene, the following utility function is defined:

$$u_j = \sum_{x=1}^{n} (\omega_{xj}\lambda_{xj}\delta_{xj}) \tag{7}$$

where u_j is the utility function for each of the four individual regions, and j represents the identity of the region, i.e., North, East, South, and West; x corresponds to the identity of a *cell(x)* in region j, n is the total number of cells of interest in region j, and ω_{xj}, λ_{xj}, and δ_{xj} represent three non-zero positive coefficients for *cell(x)* in region j and are initially given the value 1. The exploration utility function weighs the benefits of exploring a scene based on terrain, the number of cells of interest in a region of the scene, and the travel distance to cells of interest using the values of the three coefficients.

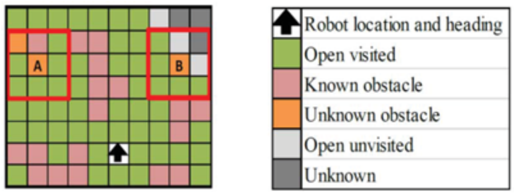

Fig. 6. Scenario illustrating contribution of the exploration coefficient λ_{xj}.

1) Terrain Coefficient: ω_{xj}, is the coefficient that is given to *cell(x)* in region j based on the type of terrain of that cell, in particular:

$$\omega_{xj} = \begin{cases} w_l l_1, & \text{if cell x is an open unvisited space} \\ w_l l_2, & \text{if cell x is an unvisited climbable obstacle} \\ w_l l_3, & \text{if cell x is an unknown obstacle} \\ 1.0, & \text{elsewhere.} \end{cases} \tag{8}$$

where $l_1 > l_2 > l_3 > 1$.

Herein, w_l is a positive weighting applied to l_1, l_2, and l_3. The weighting can be used to set a higher priority to this particular coefficient. Open unvisited cells are given the largest l value since they are obstacle-free regions and hence, allow a robot to easily navigate the cell. Unvisited climbable obstacles have the second largest value. Unknown obstacles have the lowest value due to the uncertainly regarding the true nature of the obstacles.

2) Neighboring Cells Coefficient: λ_{xj} is the coefficient given to *cell(x)* in region j based on the information in its eight neighboring cells, namely:

$$\lambda_{xj} = \begin{cases} w_p \sum_{k=1}^{8} v_x^k, & \text{when } \sum_{k=1}^{8} v_x^k \neq 0 \\ 1.0, & \text{when } \sum_{k=1}^{8} v_x^k = 0 \end{cases} \text{, where} \tag{9}$$

$$v_x^k = \begin{cases} p_1, & \text{if cell k is unknown.} \\ p_2, & \text{if cell k is an open unvisited space.} \\ p_3, & \text{if cell k is an unvisited climbable obstacle.} \\ p_4, & \text{if cell k is an unknown obstacle.} \\ 0.0, & \text{elsewhere.} \end{cases}$$

and, where $p_1 > p_2 > p_3 > p_4 > 1$.

v_x^k is the exploration value of the kth neighboring cell of *cell(x)*, where $k = 1$ to 8. w_p is a positive weighting applied to $\sum_{k=1}^{8} v_x^k$. λ_{xj} is designed to provide a higher value to cells that are adjacent to unknown cells or other cells of interest, since exploring those cells may immediately lead to the exploration of other surrounding cells. For example, consider the case in the partial 2-D grid map shown in Fig. 6, where the two cells A and B have the same type of terrain (unknown obstacle), therefore $\omega_A = \omega_B$. However, cell A has one unknown obstacle in its eight neighboring cells ($\lambda_A = p_4$), whereas, cell B has two open unvisited cells and one unknown cell, giving it the larger exploration value of $\lambda_B = p_1 + 2p_2$, when $w_p = 1$.

3) Travel Coefficient: δ_{xj} is the coefficient associated with moving to *cell(x)* in region j from the robot's current cell location, and is a function of d_x which is defined as the *distance* of *cell(x)* to the robot's current cell. As the distance to *cell(x)* increases, the value of traveling to *cell(x)* decreases. δ_{xj} favors unknown cells closer to the robot's current cell for exploration to allow for more efficient search of the environment:

$$\delta_{xj} = \begin{cases} w_q q_1 \text{ , if } \mathrm{d}_x \leq \; d_1 \\ w_q q_2 \text{ , if } d_1 < \; \mathrm{d}_x \leq d_2 \\ w_q q_3 \text{ , if } d_2 < \; \mathrm{d}_x \leq d_3 \\ w_q q_4 \text{ , if } d_3 < \; \mathrm{d}_x \leq d_4 \\ 1.0, \text{ elsewhere.} \end{cases} \tag{10}$$

where $q_1 > q_2 > q_3 > q_4 > 1$.
w_q is a positive weighting applied to $q_m(m = 1 - 4)$. $d_n(n = 1 - 4)$ represents predefined distance thresholds that can be set by an operator. These thresholds can be increased as the size of the known map increases to reflect more realistic distances.

Once the utility functions for all four regions are evaluated, the direction corresponding to the region with the largest utility value is chosen as the exploration direction. The desired exploration direction is passed to the *NG* subtask in order to locally move the robot in this direction. A robot operator is able to select the weightings for the three coefficients based on the rescue scenario at hand. In particular, he/she may decide to increase the weighting for one or more of the coefficients in order to have the robot explore in a desired manner or to maximize energy efficiency of the robot. The influence of these coefficients on the number of exploration steps is discussed in Sections VI-B and VII-A.

VI. Simulation Examples

Simulations were conducted at two stages: a training stage of the MAXQ hierarchy, and a simulation evaluation stage to verify the performance of the MAXQ HRL-based semi-autonomous controller and direction-based exploration technique in exploring unknown cluttered environments.

For both sets of simulations, USAR-like scenes consisting each of a 20 by 20 cell environment (approximately 336 m^2) with individual cells having an area of 0.84 m^2 were used. The robot used in the simulations occupied only one cell at a time. The robot was provided with the following sensory inputs: i) 3-D depth information pertaining to the cell in front of the robot for classification of cells as open space, non-climbable obstacle, or climbable obstacle based on the D_{xy} profile; ii) distance sensing on the sides and back of the robot, indicating the presence of any obstacles in the robot's neighboring cells; and iii) thermal information indicating the presence of a victim.

A. Training of MAXQ Hierarchy

Training of the MAXQ hierarchy was performed to verify the learning of robot actions and convergence of the *Q*-values. While the boundary of the scenes remained the same, the layout of each scene was automatically generated via random sensory information in order to promote learning as the robot explored the unknown environment. Overall, a total of 1800 trials were performed, with each trial having its own unique scene layout.

The exploration policy parameter, ε, and learning rate were initially set to 1 and decreased by 0.01 every 100 episodes to allow for sufficient exploration. Herein, one episode represents a single step taken by the robot, i.e., a primitive action for the *NG* subtask. The cumulative rewards for the overall USAR

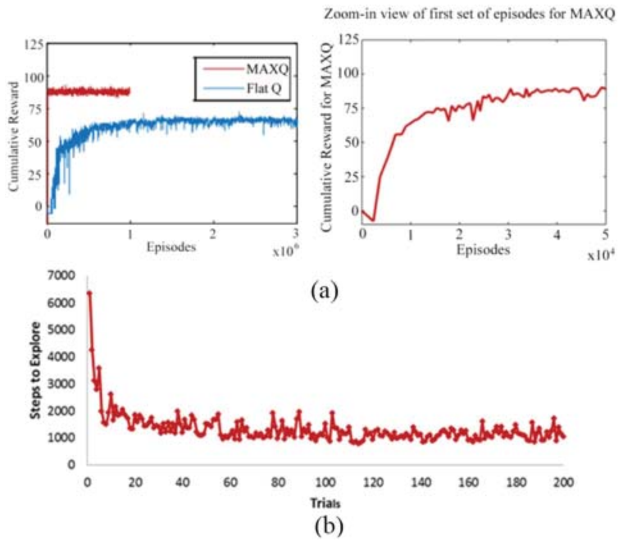

Fig. 7. (a) Convergence rate for both MAXQ and flat *Q*-learning for proposed USAR problem, and (b) exploration steps required to fully explore unknown scenes in the first 200 trials of training using the MAXQ approach.

task are shown in Fig. 7(a). The robot's efficiency in exploring the unknown scenes was improved significantly by using learning. Fig. 7(b) presents the decrease in the total number of exploration steps for the first 200 trials of learning.

We compared the convergence rate of our MAXQ approach versus a traditional flat *Q*-learning approach. The same learning rate, initial *Q*-values, state parameters, and primitive actions were used for both implementations. Fig. 7(a) also shows the cumulative rewards for the flat *Q*-learning technique. The overall results show that our MAXQ method converges at a faster rate. For flat *Q*-learning, there were over 1 million *Q*-values stored. With the aid of state and subtask abstraction, the MAXQ approach significantly reduces the amount of *Q*-values to be stored to only 4288 *Q*-values, making it considerably more efficient for this decision making problem.

B. Performance Evaluation

Once training was completed, an additional 432 trials in different cluttered USAR-like scenes were also conducted to observe the performance of the proposed MAXQ HRL-based semi-autonomous controller and the direction-based exploration algorithm in exploring and navigating unknown cluttered environments. The simulated scenes were designed to have a high density of non-climbable obstacles, creating tight corners for the robot to navigate around in order to find potential victims. In addition, the scene layouts included many isolated regions separated by non-climbable obstacles in which the robot could only enter through a single open cell or by traversing over climbable obstacles. The layout of each scene consisted of a unique combination of cell categories, providing different levels of difficulty for robot exploration. In each scene layout, 7–8 victims were placed strategically in corners and hard to reach regions of the scene. The performance of the MAXQ controller was evaluated based on: 1) the ability to explore the entire scene using the exploration technique; 2) collisions (if any) with non-climbable obstacles; and 3) the number of victims found.

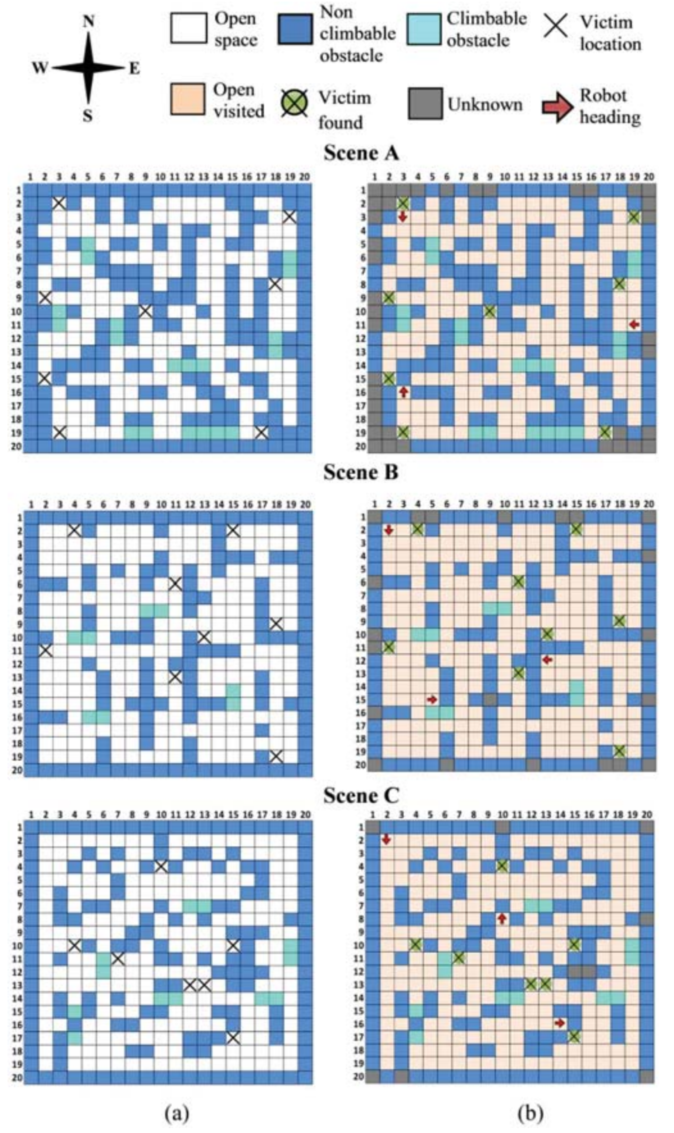

Fig. 8. Simulated USAR-like scene. (a) Actual scenes used and (b) 2-D grid map developed of the scenes during simulations with robot starting location and heading.

Herein, we will discuss simulations conducted on three of these USAR-like scenes consisting of different complexities due to space limitations. The three scenes, denoted as Scenes A, B, and C, are shown in Fig. 8(a). Scene A was designed with the largest amount of climbable and non-climbable obstacles with respect to the other two scenes. Scene B consisted of more isolated regions connected by single cell openings. Scene C had the most difficult layout for exploration due to the sizeable rubble boundaries created by large numbers of adjacent non-climbable cells.

We also investigated the overall effect of the exploration coefficients on exploration time by conducting simulations in each scene for four different cases. In the first three cases, a higher weighting was given to one of the three exploration coefficients, (i.e., $w_i = 20$, for $i = l, p$, or q), and in the fourth case, weightings on all coefficients were set to 1. Herein, exploration time is defined as the number of steps (i.e., the primitive navigation actions F, B, or θ) taken to explore the

TABLE II
EXPLORATION COEFFICIENTS USED IN THE SIMULATIONS

Conditions		Simulation			
		1	2	3	4
ω_x	Cell x: open unvisited space	200	10	10	10
	Cell x: unvisited climbable obstacle	100	5	5	5
	Cell x: unknown obstacle	40	2	2	2
$w_p v_x^k$ *	Cell k: unknown	3	60	3	3
	Cell k: open unvisited space	2	40	2	2
	Cell k: unvisited climbable obstacle	1.5	30	1.5	1.5
	Cell k: unknown obstacle	1	20	1	1
δ_x	$d_x \leq 3.0$	10	10	200	10
	$3.0 < d_x \leq 6.0$	5	5	100	5
	$6.0 < d_x \leq 9.0$	2	2	40	2
	$9.0 < d_x \leq 12.0$	1.5	1.5	30	1.5

* λ_{xj} is obtained from the sum of $w_p v_x^k$ as described by (9).

entire unknown scene. Each trial was repeated six times and the average number of steps was determined. The parameters utilized in these simulations are presented in Table II, and were selected following the requirements outlined in (8)–(10). The exact values of the exploration coefficients shown in Table II will mainly influence the magnitudes of the utility functions for the four regions (i.e., North, South, East, and West). The exploration directions themselves are not affected as long as the requirements for (8)–(10) are met, since they are determined based on the relative values of these utility functions. The upper and lower limits on d_x were selected to be proportional to the size of the map.

To further test the robustness of the navigation and exploration modules of the MAXQ HRL-based semi-autonomous controller to different starting locations in a scene with varying surrounding cell configurations, simulations were performed using three different robot starting positions in each of the scenes, as shown in Fig. 8.

The simulation results are presented in Figs. 8(b) and 9. Also shown in Fig. 8(b) are the different robot starting positions and headings used in each scene layout. The robot was able to explore each scene from all three starting positions, providing the opportunity to find all victims using the proposed MAXQ approach. Furthermore, it was able to classify the different types of cells including open cells, climbable and non-climbable obstacles. The robot was also able to navigate the scenes without any collisions. The only cells in the scenes that were still unknown at the end of exploration were those fully obstructed by non-climbable cells or victim cells as the robot was not allowed to enter these cells.

Shapiro–Wilk tests of normality confirmed that the simulation data were normally distributed. An analysis of variance (ANOVA) was performed on the exploration steps for the four different cases to determine that a statistical significant difference between the steps does exist: $F(3) = 6.50, p < 0.01$. Additional paired one-tail t-tests were performed, and the results confirmed that by placing a higher weighting on the travel coefficient, the average number of steps were reduced compared to placing a higher weighting on the terrain

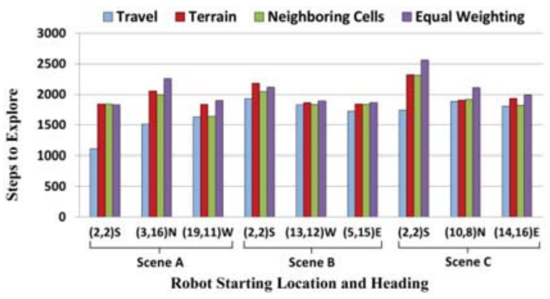

Fig. 9. Number of exploration steps for the robot in Scenes A, B, and C with respect to the different cases when higher weightings were given to one of the three exploration coefficients, and when equal weightings were given to all coefficients. Placing higher weightings on the travel coefficient resulted in lower exploration steps.

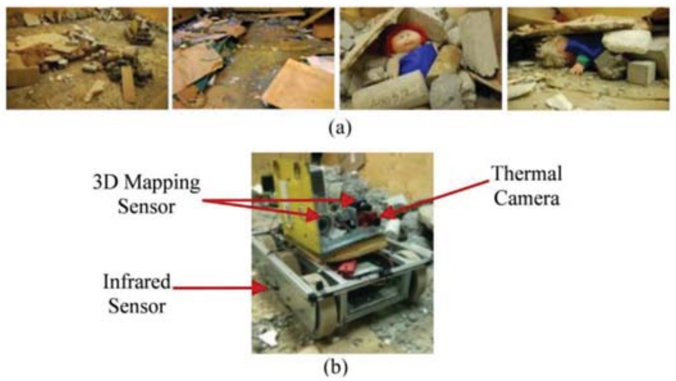

Fig. 10. (a) USAR-like environment used in the experiments and (b) rescue robot used in the experiments.

coefficient ($t(53) = 3.67$) and the neighboring cells coefficient ($t(53) = 3.02$), and having equal weighting on all coefficients ($t(53) = 4.41$). This occurred because when placing a higher weighting on the travel coefficient, the robot would explore more cells in its surroundings before moving on to other regions. In the cases where the weightings were the same or, either the terrain or the neighboring cells coefficients had higher weights, the robot revisited a number of already visited cells, therefore increasing the number of exploration steps.

ANOVA was also performed across the USAR Scenes. From the analysis, the robot took an average of 1786 ($\sigma = 293$), 1911 ($\sigma = 135$), and 2026 ($\sigma = 249$) exploration steps to explore Scenes A, B, and C, respectively. We conducted an additional ANOVA test to determine whether the level of difficulty of the scene affected the number of exploration steps. With a 95% confidence level, there was no statistical significance between the average numbers of exploration steps for these three scene layouts, emphasizing the robustness of our proposed technique to different scene layouts. Furthermore, ANOVA results with a 95% confidence level also showed that there was no statistical significance between the robot's starting location within a scene and the average number of exploration steps.

VII. Experiments

Experiments were conducted in a cluttered 12 m^2 USAR-like environment to test the performance of our MAXQ HRL-based semi-autonomous controller in a physical environment. The USAR-like environment (Fig. 10(a)), was designed to mimic a disaster scene, containing different types of objects and materials such as concrete, wood, metal, plastic, cardboard, ceramic, brick, plaster, paper, rubber-like polymers, and rocks. Six victims, represented by dolls and mannequins, were also placed in the environment. The majority of the victims were partially obstructed by rubble in which case only a portion of their bodies were visible such as their limbs or head. Rubble was strategically placed in the environment such that the robot would have to explore around obstacles and corners to search for victims. Furthermore, within the scene, inclines were made to allow a robot to climb over rubble piles and navigate an elevated upper level of the scene.

Two sets of experiments were carried out to evaluate the MAXQ HRL-based semi-autonomous controller. The first set of experiments focused on evaluating the performance of the MAXQ hierarchy, while the second set of experiments compared the performance of the overall semi-autonomous controller to full teleoperation by an operator. In both experiments, a rugged mobile robot was utilized (Fig. 10(b)).

The robot is equipped with a real-time 3-D mapping sensor, five infrared sensors, and a thermal camera (Fig. 10(b)). The real-time 3-D mapping sensor utilizes an active structured light technique [20]. In the experiments, the sensing range of the sensor was set to 0.914 m, i.e., the length of one cell, in order to only capture the depth information pertaining to the cells in front of the robot for rubble pile profile classification and updating of the 2-D grid map. In general, when the size of the environment is unknown, the size of a single cell can be defined based on the working range of the sensors. The five infrared sensors distributed along the perimeter of the robot were used to detect the presence of objects in adjacent cells for both obstacle avoidance and updating of the 2-D grid map. A thermal camera was placed at the front of the robot to identify victims. Heat pads were placed on the dolls and mannequins to produce heat signatures to mimic human bodies. The thermal camera was used to sense the heat signatures.

A. *Experiment #1: Testing of MAXQ Hierarchy*

For the first set of experiments, three different scene layouts were used in the environment to test the performance of the MAXQ hierarchy for the different subtasks (Figs. 11(a)–(c)). Scene layout 1 contains both open spaces and non-climbable rubble piles, whereas Scene layouts 2 and 3 also contain climbable rubble providing the robot with an opportunity to navigate an elevated upper level of the scene. In layout 2, the victim in cell (3, 4) is located on the lower level portion of the scene, i.e., hidden under the climbable rubble.

Four different experiments were conducted for each scene, with the first three experiments having a higher weighting of 20 for one of the exploration coefficients, and the last experiment having equal weightings of 1 for all the coefficients. In the experiments, the same coefficient values as in Table II were used, with the exception that the distance limits d_n were adjusted for the smaller physical scene to represent more realistic traveling distances. The new values were $d_1 = 1.5$, $d_2 = 3.0$, $d_3 = 6.0$, and $d_4 = 12.0$. Similar to the simulations, the

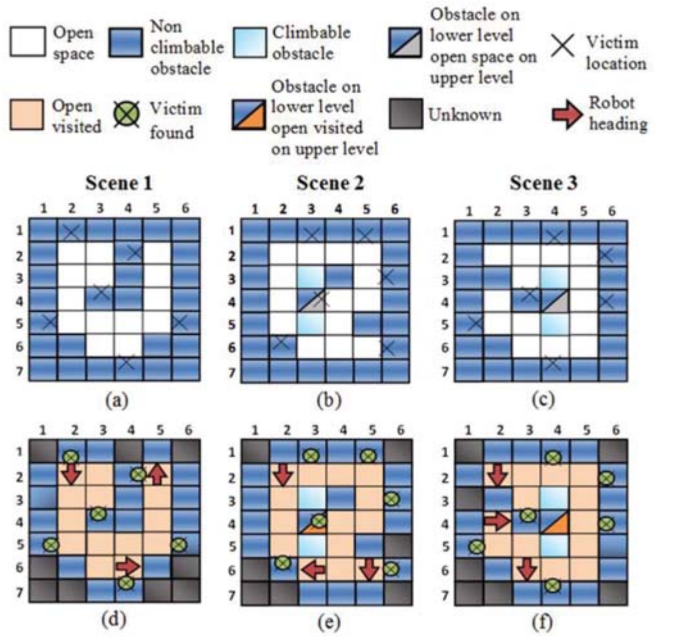

Fig. 11. 2-D representation of experimental scenes (a)–(c) depicting the original scene layouts 1, 2, 3, and (d)–(f) determined by the robot during the experiments.

experiments for each scene were also performed with the robot starting in three different starting positions. For each starting position, the experiment was conducted three times.

1) Experimental Results: The experimental results for each scene layout are presented in Figs. 11(d)–(f) and 12. The results verify that the rubble pile classification was able to correctly categorize the depth profile information in the D_{xy} arrays as climbable obstacle, non-climbable obstacle, or an open space. The unknown cells represent cells in the scene that could not be identified since 3-D sensory information could not be obtained from those regions due to obstruction of the regions by walls or rubble piles. With an accurate categorization of the different rubble piles in the scene in each experiment, the robot was able to create a correct representation of the scene. Fig. 12 summarizes the average number of exploration steps for the three scene layouts and the different robot starting locations and headings within each scene.

As can be seen in Fig. 12, robot exploration, on average, took 78 ($\sigma = 17$), 84 ($\sigma = 14$), and 90 ($\sigma = 6$) steps in scene layouts 1–3, respectively. The weightings on the different exploration coefficients did not have a significant effect on the exploration performance. Using ANOVA, it was found that with a 95% confidence interval, there was no statistical significance on the average number of exploration steps between these three scene layouts, further validating the robustness of our technique to different scenes. However, some of the robot's different starting locations and headings in the scenes did affect the number of exploration steps. The relative higher number of exploration steps for the same scenes were a result of the robot revisiting already visited cells. Due to the small size of the scenes and a limited number of regions to explore, there were fewer exploration options for the robot when compared to a larger more complex scene (as presented in the simulations). From these experiments, it can be concluded that when the size of the scene is small, the effects of the weightings are similar.

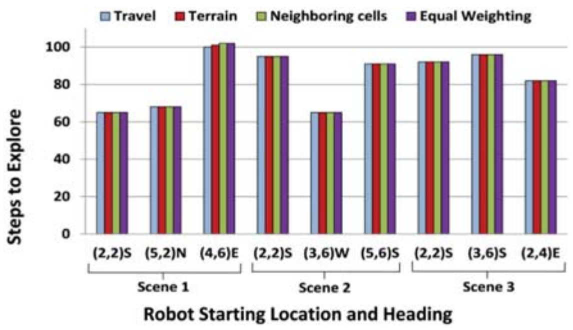

Fig. 12. Number of exploration steps for the robot in Scenes 1, 2, and 3 with respect to the different cases when higher weightings were given to one of the three exploration coefficients, and when equal weightings were given to all coefficients.

Fig. 13. USAR-like environment for Experiment #2.

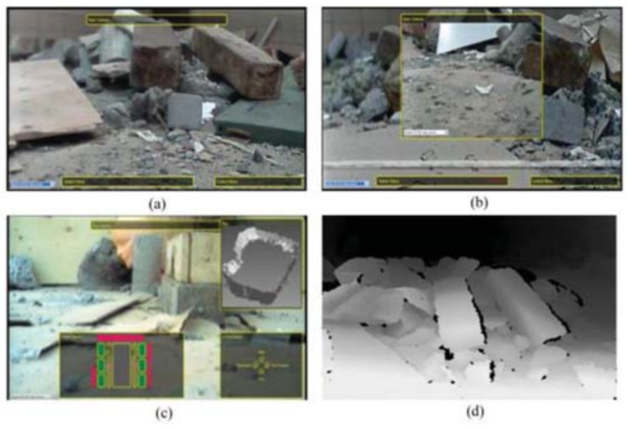

Fig. 14. Information provided to the operator through the HRI Interface. (a) 2-D image of robot's front view, (b) 2-D image of robot's rear view, (c) 3-D map, robot status (green indicates the wheels of the robot are moving), infrared visual feedback on the robot's perimeter (if objects are very close red rectangles appear), and control menu display, and (d) real-time 3-D information.

B. Experiment #2: Semi-Autonomous versus Teleoperated Control

We conducted a set of experiments to evaluate the performance of the MAXQ HRL-based semi-autonomous controller versus robot teleoperation. A new scene layout was created for these experiments (Fig. 13). The scene contained open space, partially obstructed victims, non-climbable obstacles as well as climbable rubble to provide the robot with the opportunity to explore the main lower level and also an elevated level of the scene.

Ten participants ranging in age from 19 to 33 years partook in the experiments. None of these participants had any previous experience in remote robot navigation and exploration,

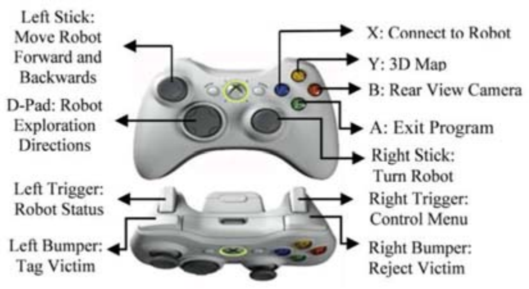
Left Stick: Move Robot Forward and Backwards
D-Pad: Robot Exploration Directions
Left Trigger: Robot Status
Left Bumper: Tag Victim
X: Connect to Robot
Y: 3D Map
B: Rear View Camera
A: Exit Program
Right Stick: Turn Robot
Right Trigger: Control Menu
Right Bumper: Reject Victim

Fig. 15. Xbox 360 wireless controller used as the operator's input device.

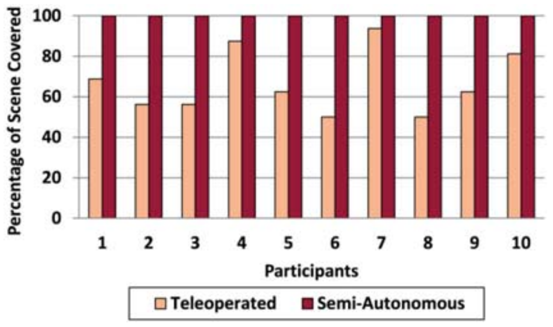

Fig. 16. Percentage of scene explored for both the teleoperation and the semi-autonomous modes.

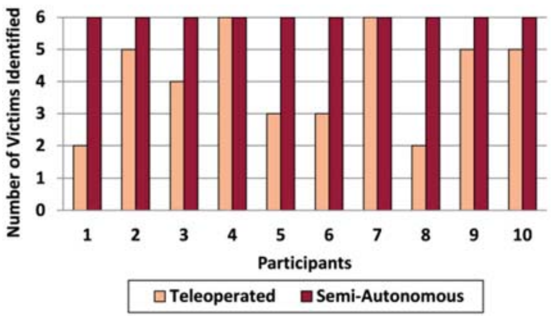

Fig. 17. Number of victims identified for both the teleoperation and the semi-autonomous modes.

which is similar to other teleoperation studies conducted including in USAR-like scenes [22]. Two different trial sets were conducted: 1) the operators were asked to control the robot while having full teleoperated control over the robot in order to navigate and explore the scene while also finding as many victims as possible and 2) the robot was in semi-autonomous mode, in which case the MAXQ HRL-based semi-autonomous controller was implemented, and the operator and the robot shared tasks as needed. The following performance parameters were measured in both trials: 1) number of victims found; 2) percentage of the scene explored; 3) number of robot collisions; and 4) total search time. For both trials, the difficulty level of the scene remained the same; however, noticeable objects in the rubble piles and the victims were distributed in different locations in order to minimize human expert knowledge of the scene.

1) HRI Interface and Control Pad: The interface presented in Fig. 14 was used by the operators during the trials. The interface is based on the following sources of information: 1) 2-D video provided by two wireless cameras (one at the front and one at the back of the robot) (Figs. 14(a) and (b)); 2) an updated 3-D map of the environment which is obtained by the *SLAM* module, (Fig. 14(c)); 3) robot status, which provides feedback as to whether the robot has received control commands and when they have been implemented (Fig. 14(c)); 4) control menu, which allows for connecting the controller to the interface, displaying the map information or exiting the interface (Fig. 14(c)); 5) infrared sensor feedback displayed visually to show the operator the proximity of obstacles surrounding the robot (Fig. 14(c)); and 6) real-time 3-D information of the robot's frontal view obtained from the 3-D mapping sensor (Fig. 14(d)). During the experiments, the 2-D front camera view and the real-time 3-D information were continuously shown on side-by-side monitors to the operator.

Operator control of the robot was achieved using an Xbox 360 wireless gamepad (Fig. 15). The gamepad includes two mini joysticks and a set of buttons whose functions are detailed in Fig. 15. Prior to the experiments, each participant was given a tutorial on how to use the control pad and interface, and then had a chance to control the robot outside of the scene for 15 minutes in order to familiarize him/herself with the robot.

2) Experimental Results: Figs. 16–18 and Table III present the performance comparison results of the teleoperated control trials versus the semi-autonomous control trials. Fig. 16 presents the percentage of the overall rubble scene that was explored by the robot in all 10 trials. In general, during teleoperation none of the operators were able to traverse the entire scene. Participants 4 and 7 explored the scene the most, with 88% and 94% scene coverage, allowing them to find all six victims. In semi-autonomous mode, the robot was able to explore the total available area of the overall USAR-like scene for all 10 trials utilizing the MAXQ approach. On average, approximately four victims were identified in the teleoperated mode versus all 6 in the semi-autonomous mode (Fig. 17). False victim identification also occurred with participants 6 and 7 during the teleoperation trials. Participant 6 identified a red cushion in the scene (shown in Fig. 13) to be a victim. Participant 7 identified the red cushion and also an orange plastic casing to be a victim. On the other hand, Participant 8 failed to identify a victim that was clearly displayed on the interface via the front view camera of the robot.

Fig. 18 presents the total number of collisions that occurred for both experiments. Herein, a collision is defined as the robot bumping into a non-climbable obstacle or a victim, or falling off the edge of the upper level, i.e., a drop, as classified by the rubble pile categorization technique. As can be seen in the figure, the number of collisions was lower in the semi-autonomous operation for all of the trials. Collisions that were detected in this mode occurred when control was passed to the operator to assist the robot in physically getting out of tight cells surrounded by non-climbable rubble. Operators would use brute force to try to navigate the robot over non-climbable rubble and through narrow passages. During teleoperation, these types of collisions occurred at a higher frequency; in addition, five out of the 10 participants actually collided with a victim in the scene. Only four participants in teleoperation

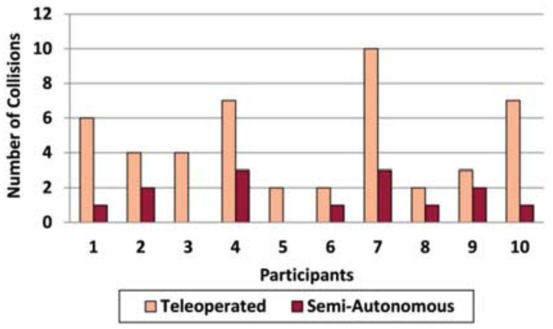

Fig. 18. Number of collisions for both the teleoperation and the semi-autonomous modes.

TABLE III
SUMMARY OF TRIAL TIMES

Participant	1	2	3	4	5	6	7	8	9	10
Teleoperated Trial Times (s)	169	283	179	595	186	292	987	377	402	483
Semi-Autonomous Trial Times (s)	366	369	376	367	364	381	370	355	373	365

mode actually navigated the robot to the upper level of the scene via an inclined climbable ramp for further exploration. Two of these participants ended up driving the robot off of steep cliffs on the sides of the upper level instead of navigating the robot down a gradual decline which was accessible to them.

Table III presents the total time for each trial, which is defined to be the time it took to explore and navigate the scene, identify victims and exit the scene. The average completion time for each trial was determined to be 395 s (σ = 250) and 369 s (σ = 7) for the teleoperated mode and semi-autonomous mode, respectively. Participants 4 and 7 who explored the largest area of the scene and found all victims during teleoperation had the longest trial times at 595 and 987 s. These trial times are significantly higher than the total exploration times that these participants had in semi-autonomous mode to achieve the same tasks.

During the semi-autonomous trials, tasks were shared by the robot and operator. The robot handed over control to the operator in two particular scenarios: 1) when it was physically stuck in a cell due to unsteady terrain conditions or large rubble piles on either side of the robot obstructing the robot's movement and 2) when it could not identify a victim with high probability due to the size of the heat region being detected. On average these types of scenarios occurred five times during each of the trials. For the first case, the system detected that the robot was not successfully executing the navigation commands, hence, the navigation subtask was handed over to the operator. Control was handed back to the robot once the executed commands were performed correctly by the operator. For victim identification the operator was asked to either tag or reject the victim in the viewpoint of the robot using the control pad.

Once the experiments were completed, the participants also completed a questionnaire reflecting on their experiences with the robot. The questionnaire consisted of questions related to their stress level during both trials, their ability to utilize the available sensory information, and how semi-autonomous control affected their overall experience. With respect to stress levels, eight participants stated that they felt stressed during robot teleoperation, while only two participants felt stressed during semi-autonomous control. Eight out of the 10 participants felt that they had a more difficult time monitoring all of the sensory information that was made available to them in the teleoperation mode for the entire duration of the search and rescue task, and hence, seven of them also felt disoriented and confused about where the robot was in the scene and in which direction they should explore during the teleoperation mode. With the robot in semi-autonomous mode, 8 of the 10 participants mentioned that they had a better overall experience compared to the teleoperated mode. In particular, these participants found that the robot's autonomous capabilities enhanced their understanding of the scene and their own decision making abilities, since they did not have to complete all rescue tasks simultaneously on their own, increasing their situational awareness of the overall scene. The experimental and questionnaire results confirm that a MAXQ HRL-based semi-autonomous control architecture for rescue robots has the potential of improving the performance of the search and rescue mission in obstacle avoidance, exploration, and victim identification.

VIII. CONCLUSION

In this paper, a novel MAXQ HRL-based semi-autonomous control architecture was presented for robotic exploration of unknown and cluttered USAR environments. The controller's objective is to provide a rescue robot with the ability to learn from its previous experiences in order to improve its performance in navigating and exploring a disaster scene to find as many victims as possible. This allows a human operator to benefit from the robot's capability to continuously learn from its surroundings and adapt to more complex environments. A new direction-based exploration technique is integrated into the controller to expand the robot's search area via the classification of regions and the rubble piles within these regions. The simulations and experiments conducted verify the robustness of the learning-based MAXQ controller in exploring entire USAR-like scenes and recreating an accurate representation of the scenes. Furthermore, they validated the use of terrain information in determining exploration direction in cluttered environments. The comparison experiments showed improved performance of the proposed semi-autonomous controller over traditional teleoperation. Future work will consist of conducting more experiments in larger-scale physical USAR-like scenes and with more participants, and further optimizing the control architecture modules. Comparison studies will also be conducted with respect to the non-learning-based semi-autonomous controllers proposed in the literature.

ACKNOWLEDGMENT

The authors would like to thank M. Salama, J. Qi, and E. Lin for their assistance in this paper.

References

[1] J. Casper and R. R. Murphy, "Human-robot interactions during the robot-assisted urban search and rescue response at the world trade center," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 33, no. 3, pp. 367–385, Jun. 2003.

[2] R. R. Murphy, "Activities of the rescue robots at the World Trade Center from 11–21 September 2001," in *Proc. IEEE Robot. Autom. Mag.*, 2004, pp. 50–61.

[3] J. Burke and R. R. Murphy, "Human-robot interaction in USAR technical search: Two heads are better than one," in *Proc. IEEE Int. Workshop ROMAN*, Kurashiki, Japan, 2004, pp. 307–312.

[4] J. Burke, R. R. Murphy, M. Coovert, and D. Riddle, "Moonlight in Miami: A field study of human-robot interaction in the context of an urban search and rescue disaster response training exercise," *Human-Comput. Interact.*, vol. 19, no. 1, pp. 85–116, Jun. 2004.

[5] D. J. Bruemmer *et al.*, "I call shotgun! An evaluation of mixed-initiative control for novice users of a search and rescue robot," in *Proc. IEEE Int. Conf. Syst., Man, Cybern.*, The Netherlands, 2004, pp. 2847–2852.

[6] R. R. Murphy and J. J. Sprouse, "Strategies for searching an area with semi-autonomous mobile robots," in *Proc. Conf. Robotics Challenging Environ.*, Albuquerque, NM, USA, 1996, pp. 15–21.

[7] A. Finzi and A. Orlandini, "Human-robot interaction through mixed-initiative planning for rescue and search rovers," in *Advances in Artificial Intelligence*, S. Bandini and S. Manzoni, Eds. Berlin, Germany: Springer, 2005, pp. 483–494.

[8] M. Baker and H. A. Yanco, "Autonomy mode suggestions for improving human-robot interaction," in *Proc. IEEE Int. Conf. Syst., Man, Cybern.*, Hague, The Netherlands, 2004, pp. 2948–2953.

[9] H. A. Yanco *et al.*, "Improving human-robot interaction for remote robot operation," in *Proc. AAAI*, 2005.

[10] B. Doroodgar, M. Ficocelli, B. Mobedi, and G. Nejat, "The search for survivors: Cooperative human-robot interaction in search and rescue environments using semi-autonomous robots," in *Proc. IEEE ICRA*, Anchorage, AK, USA, 2010, pp. 2858–2863.

[11] B. Doroodgar and G. Nejat, "A hierarchical reinforcement learning based control architecture for semi-autonomous rescue robots in cluttered environments," in *Proc. IEEE CASE*, Toronto, ON, Canada, 2010, pp. 948–953.

[12] J. W. Crandall and M. A. Goodrich, "Characterizing the efficiency of human robot interaction: A case study of shared-control teleportation," in *Proc. IEEE/RSJ Int. Conf. IROS*, Lausanne, Switzerland, 2002, pp. 1290–1295.

[13] R. Makar, S. Mahadevan, and M. Ghavamzadeh, "Hierarchical multi-agent reinforcement learning," in *Proc. Int. Conf. Autonomous Agents*, Montreal, QC, Canada, 2001, pp. 246–253.

[14] L. A. Jeni, Z. Istenes, P. Szemes, and H. Hashimoto, "Robot navigation framework based on reinforcement learning for intelligent space," in *Proc. Conf. Human Syst. Interact.*, Krakow, Poland, 2008, pp. 761–766.

[15] P. Stone and R. S. Sutton, "Scaling reinforcement learning toward RoboCup soccer," in *Proc. Int. Conf. Mach. Learn.*, Williamstown, MA, USA, 2001, pp. 537–544.

[16] T. G. Dietterich, "Hierarchical reinforcement learning with MAXQ value function decomposition," *J. Artif. Intell. Res.*, vol. 13, no. 1, pp. 227–303, 2000.

[17] B. Yamauchi, "A frontier-based approach for autonomous exploration," in *Proc. IEEE Int. Symp. CIRA*, Monterey, CA, USA, 1997, pp. 146–151.

[18] W. Burgard *et al.*, "Coordinated multi-robot exploration," *IEEE Trans. Robot.,* vol. 21, no. 3, pp. 376–386, Jun. 2005.

[19] A. Howard, M. Mataric, and S. Sukhatme, "An incremental deployment algorithm for mobile robot teams," in *Proc. Int. Conf. IEEE/RSJ*, Lausanne, Switzerland, 2002, pp. 2849–2854.

[20] Z. Zhang, G. Nejat, H. Guo, and P. Huang, "A novel 3-D sensory system for robot-assisted mapping of cluttered urban search and rescue environments," *Intell. Serv. Robot.*, vol. 4, no. 2, pp. 119–134, 2011.

[21] S. Singh, T. Jaakkola, M. L. Littman, and C. Szepesvari, "Convergence results for single-step on-policy reinforcement-learning algorithms," *Mach. Learn.*, vol. 38, no. 3, pp. 287–308, 2000.

[22] K. S. Jones, B. R. Johnson, and E. A. Schmidlin, "Teleoperation through apertures: Passability versus driverability," *J. Cogn. Eng. Decis. Making*, vol. 5, no. 1, pp. 10–28, 2011.

Barzin Doroodgar received the B.A.Sc. and M.A.Sc. degrees in mechanical engineering from the University of Toronto, Toronto, ON, Canada, in 2009 and 2011, respectively.

Currently, he is an Engineer/Designer Associate with Honeywell, Mississauga, ON, Canada.

Yugang Liu (S'13) received the B.E. and M.A.Sc. degrees from the University of Science and Technology, Beijing, China, and Beijing University of Posts and Telecommunications, Beijing, China, in 1999 and 2002, respectively. He is currently pursuing the Ph.D. degree with the Autonomous Systems and Biomechatronics Laboratory at the University of Toronto, Toronto, ON, Canada.

His current research interests include learning-based semi-autonomous control of multirobot teams in urban search and rescue environments.

Goldie Nejat (S'03–M'06) received the B.A.Sc. and Ph.D. degrees in mechanical engineering from the University of Toronto (UofT), Toronto, ON, Canada, in 2001 and 2005, respectively.

Currently, she is an Associate Professor and the Director of the Autonomous Systems and Biomechatronics Laboratory, Department of Mechanical and Industrial Engineering, UofT. She is also the Director of the Institute for Robotics and Mechatronics at UofT, and an Adjunct Scientist at the Toronto Rehabilitation Institute, Toronto, ON, Canada. Her current research interests include sensing, human-robot interaction, semi-autonomous and autonomous control, and intelligence of assistive/service robots for search and rescue, exploration, healthcare, and surveillance applications.

Dr. Nejat is a member of the IEEE Robotics and Automation Society and the American Society of Mechanical Engineers.