High-Level Commands in Human-Robot Interaction for Search and Rescue

Alain Caltieri and Francesco Amigoni

Artificial Intelligence and Robotics Laboratory, Politecnico di Milano, Milano, Italy
alaincaltieri@gmail.com, francesco.amigoni@polimi.it

Abstract. Successful search and rescue operations require an appropriate interaction between human users and mobile robots operating on the field. In the literature, use of waypoints for driving the robots has been identified as the main approach to trade-off between fully autonomous robotic systems, which can exclude human users from the control cycle, and completely tele-operated robotic systems, which can excessively burden human users. In this paper, we propose an intermediate level between full autonomy and waypoint guidance. Specifically, human users can issue *high-level commands* to the robots, like "explore along a direction" and "explore in this area", which do not explicitly specify the target locations, but introduce a bias over the autonomous target selection performed by the robots. Experimental results show that high-level commands are effective, provided that notification messages coming from the robots are filtered.

Keywords: human-robot interaction, search and rescue, multirobot systems.

1 Introduction

In many search and rescue settings, the interaction between human users and mobile robots operating in harsh environments is fundamental, given the time constraints and the importance of rescuing victims. Although some fully-autonomous multirobot systems for search and rescue have been developed in the last years, they are not expected to be widely employed in real scenarios, mainly because human users tend to be excluded from the control cycle [1, 2]. As a consequence, autonomous and human components should be balanced. In most of the current multirobot systems for search and rescue, this trade-off is obtained by allowing human users to directly issue waypoint commands (by which the user specifies a target location, often close to robot current position, and the robot autonomously plans a path to reach it) or lower-level tele-operation commands.

In this paper, we propose to introduce an intermediate level between full autonomy and waypoint guidance. Operating at this level, human users can issue *high-level commands* to the robots, like "explore along a direction" and "explore in this area", which do not explicitly specify the target locations, but introduce a bias over the autonomous target selection performed by the robots. High-level commands give the robots more autonomy than waypoint guidance and require an exploration technique that supports them. Experimental results show that using high-level commands increases both the area explored and the number of victims found, but increases also user's *workload*, namely the cognitive effort for controlling the multirobot system, decreasing user's *situation awareness*, namely the ability to figure out the global status of the system. We show that, introducing an appropriate filtering of notification messages coming from the robots, workload is reduced, keeping only the positive effects of high-level commands on performance of multirobot systems for search and rescue.

We consider a search and rescue setting in which a number of robots coordinate in order to explore an initially unknown environment and find victims. The robots communicate with a base station, which features a Graphical User Interface (GUI) through which a human user can interact with the robots (issuing commands and receiving notifications). Our human-robot interface is embedded in the *PoAReT* (*Politecnico di Milano Autonomous Robotic Rescue Team*) system that has been developed for the Virtual Robot Competition of the RoboCup Rescue Simulation League. Full information about the system, including a link to the source code is available at `http://home.deib.polimi.it/amigoni/research/PoAReT.html`. The PoAReT system operates in environments simulated in USARSim [3].

2 Human-Robot Interaction in Search and Rescue

The role of the human component in a robotic system can vary depending on the extension of human influence and on the level of autonomy of the system. In the system presented in this paper, the human user interacts both with the whole robotic system and with the single robots (see [4] for a complete taxonomy of forms of human-robot interaction). Specifically, the human user can act as *supervisor* and as *operator* [5]. In the first role, the human user is focused on high-abstraction-level information and tasks. In the role of operator, the human user must also comply with lower-level tasks, ranging from direct tele-operation of robots to their manual coordination. Supervision requires some autonomy of the system in executing complex actions [6]. The most common solution is to implement a dynamic level of autonomy, modifiable by the human operator (*adjustable autonomy*), by the robotic system (*adaptive autonomy*), or by both (*mixed initiative*). Results in [7] show that a mixed initiative approach yields a better global performance in some urban search and rescue tasks.

In the following of this section, we survey some of the most significant systems for interaction between human users and autonomous multirobot systems specifically developed for search and rescue and employed in realistic settings (e.g., competitions).

In the Steel system [8] every function is carried out by a separate module and modules are linked together by the Machinetta framework. The GUI is map-centric (namely, the map built by the robots and representing the environment is the central element of the GUI) and allows the operator to interact with many robots simultaneously. The Steel system allows three different levels of interaction (and autonomy). With full autonomy, the human operator can select some interest areas to be explored with higher priority by the whole multirobot system. Autonomous coordination and exploration modules then accomplish the task. With manual specification of the path to be followed by the robots, the operator can set a number of waypoints. Finally, with tele-operation, the operator directly controls the motion of the robots.

The Hector project [9] aims at developing a fully-autonomous robotic system in which the human user acts as a supervisor. The human user interacts with the system by mean of some high-level policies. For example, a policy could specify what actions should be performed when an exception is raised during a navigation task. The system considers such policies, together with the information about the surrounding environment and the goals, and autonomously decides the action to execute. The human supervisor can cancel or override the decision of the system.

The system in [10] allows a single operator to control a team of robots, interacting with them at three abstraction levels: full autonomy with coordination between robots, waypoint selection, and tele-operation.

Finally, Team Michigan [1] won the MAGIC 2010 competition with a system composed of 14 robots and 2 human operators, in charge of sensor data cleaning and integration and of the allocation of tasks to robots, respectively. Team Michigan features a rich feedback interface that presents information filtered and ordered by priority on a 3D map of the explored environment. The task operator can interact with the system by adjusting the level of autonomy, from full autonomy (the robots autonomously decide what to do and coordinate without human intervention), to semi-autonomy (the human operator assigns to a robot a target point and the robot automatically plans and follows a path to it), to tele-operation.

As it emerges from the discussion above, in most of the current systems, there is no intermediate level between full autonomy and waypoint guidance. In this paper, we propose an approach that contributes to fill this gap.

3 The PoAReT Multirobot System

In this section, we overview the PoAReT system architecture (as reported in [11], to which the reader is referred for full description), highlighting some of its features in order to set the context for illustrating the Human-Robot Interaction (HRI) system proposed in this paper.

The PoAReT system is a controller for simulated robots in USARSim. In particular, it controls mobile platforms (usually Pioneer All Terrain P3AT), each one equipped with laser range scanners, sonars, and a camera. Laser range scanners are used to build a two-dimensional geometrical map of the environment that is represented with two sets of line segments. The first set contains the line segments that represent (the edges of) perceived obstacles. The second set contains the line segments that represent the *frontiers*, namely the boundaries between the known and the unknown portions of the environment. The free space already explored is a polygon (generally with holes) whose edges are either obstacles or frontiers.

When operating autonomously, the main cycle of activities of the PoAReT system is: (a) building a map of the environment composed of line segments, (b) selecting the most convenient frontiers to reach, and (c) coordinating the allocation of robots to the frontiers. At the same time, the system detects victims on the basis of the images returned by the onboard cameras and interacts with the human operator via the GUI.

The architecture of the system is organized in two different types of processes, one related to the base station and one related to the mobile robots. The *base station* embeds the HRI system, which is described in the next section. The base station process can spawn new robots in the USARSim environment: for each robot, a new independent process is created and started. The processes of the base station and of the robots communicate only through Wireless Simulation Server (WSS) [12]. A distance vector routing protocol [13] is implemented to deliver messages.

The *robot* process is structured in different modules, each one related to a high-level functionality: motion control, path planning, SLAM, exploration, coordination, and victim detection. The main functionalities of the above modules are described in the following. The motion control module is straightforward, given the locomotion model of P3AT. The path planning module is invoked to reach a location with a path that lies entirely in the known space (e.g., the location can be on a frontier between known and unknown space). The algorithm of the path planning module is a variant of the Rapidly-exploring Random Tree (RRT) algorithm [14].

The Simultaneous Localization And Mapping (SLAM) problem is tackled using a feature-based method. The SLAM module associates the line segments of a laser scan (points of a scan are approximated with line segments by using the split-and-merge algorithm [15]) to the the linear features in the map, with respect to a distance measure, such as that in [16]. Then, the module executes an Iterative Closest Line (ICL) algorithm (like [16]) to align the scan and the map. All the line segments of a scan are added to the map; periodically a test is carried out to determine whether there is enough evidence to support the hypothesis of two previously associated line segments being in fact the same; if so, they are merged.

The exploration module evaluates new frontiers to explore, in order to discover the largest possible amount of the environment, and calls the coordination module to find an allocation of robots to the frontiers. In our system, we use Multi-Criteria Decision-Making (MCDM) [17], a framework to compose different criteria and obtain the utility $u(f, r)$ of a frontier f for a robot r. Criteria we consider include distance between f and the current position of r, expected information gain at f, and probability of communicating with the base station once in f. One of the main features of MCDM is that adding further criteria is particularly easy. Please refer to [17] for further details and full mathematical treatment of MCDM.

The coordination module is responsible of allocating tasks to the robots. The mechanism we use is market-based and sets up auctions in which tasks (i.e., frontiers to reach) are auctioned to robots [18] that bid according to the evaluation returned by their exploration modules. Market-based mechanisms provide a well-known mean to bypass problems like unreliable wireless connections or robot malfunctions.

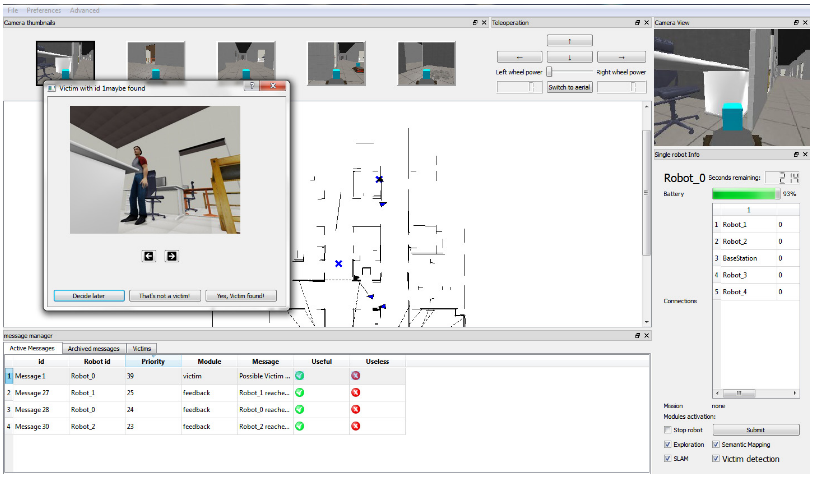
Finally, the victim detection module is responsible for searching victims inside the environment. It analyzes images coming from a robot camera (using a skin detector in HSV (Hue, Saturation, Value) color space) and classifies them according to the presence or absence of victims (using a version of the Viola-Jones algorithm [19]). In the positive case, the victim detection module signals the human operator.

4 The PoAReT Human-Robot Interaction System

The PoAReT Human-Robot Interaction (HRI) system allows a single human operator to effectively control a relatively large group of robots. It displays data to the operator and accepts commands from the operator to control the spawned robots. It reduces the workload of the operator and increases her/his situation awareness through a mixed-initiative approach. On the one hand, the PoAReT HRI system allows a single operator to control the system by issuing high-level commands to robots. On the other hand, the PoAReT HRI system filters out notifications arriving from the robots, based on the operator's preferences, past behavior, and situation parameters.

4.1 The PoAReT GUI

The GUI of the PoAReT system (Fig. 1) is map-centric, modular, and with fully-reconfigurable windows to allow the operator to better use the space on the screen and possibly discard unwanted components. The main components of the interface are:



Robot_0 Seconds remaining: 214
Battery 93%

	1	
1	Robot_1	0
2	Robot_2	0
3	BaseStation	0
4	Robot_3	0
5	Robot_4	0

Connections

Mission none
Modules activation:
☐ Stop robot
☑ Exploration ☑ Semantic Mapping
☑ SLAM ☑ Victim detection

	id	Robot id	Priority	Module	Message	Useful	Useless
1	Message 1	Robot_0	39	victim	Possible Victim ...	✓	✗
2	Message 27	Robot_1	25	feedback	Robot_1 reache...	✓	✗
3	Message 28	Robot_0	24	feedback	Robot_0 reache...	✓	✗
4	Message 30	Robot_2	23	feedback	Robot_2 reache...	✓	✗

Fig. 1. GUI of the PoAReT system

- The map (Fig. 1, left center). The local line segment maps sent by the robots are merged and the resulting map is displayed to the operator. The map is zoomable, scrollable, and interactive: by clicking on the map the operator can issue commands to the robots and select them. Commands are: operate fully autonomously, high-level commands (see Section 4.2), waypoint guidance, and tele-operation.
- Message manager (Fig. 1, left bottom). Beyond maps, every robot sends to the base station information regarding its internal status (e.g., battery level and wireless connectivity) and events like faults or victims detected. All these data are classified by the HRI system (see Section 4.3) that presents the relevant information to the operator as messages. A pop-up window signals particularly important notification, like a victim detection (as in Fig. 1).
- Camera manager (Fig. 1, left top). Small boxes containing the compressed (reduced framerate and resolution) video streams captured by the cameras of the robots are always shown to the operator.
- Selected robot's camera view (Fig. 1, right top). It shows the video stream of the selected robot, with full framerate and resolution.

- Selected robot's info (Fig. 1, right center). Information includes connectivity, battery status, and internal modules activation (like victim detection module or autonomous exploration module).
- Tele-operation (Fig. 1, center top). This widget allows the operator to directly control the movement of the selected robot.

Despite its modularity, the GUI has a centralized logic and all information is always coherently presented to the operator. Internally, the information flows are managed by a single module, called Base Station Core, which delivers the information to the proper component (internal modules or robots).

4.2 High-Level Commands

As discussed, workload and situation awareness are two main factors that strongly affect the performance of a human operator when controlling a multirobot system. A trade-off between workload and situation awareness by balancing autonomy and direct control of robots has been found in literature mainly using waypoint guidance (Section 2).

We introduce a new kind of High-Level Commands (HLCs) that combine the positive effects of robot's autonomy with the need to preserve the human operator in the control loop. HLCs allow an operator to indicating a preference about the exploration policies of a robot, specifically a preferred direction to explore (a vector $\boldsymbol{\alpha} = (\alpha_x, \alpha_y)$ with origin in the robot's current position) and a specific area of interest (a point $p = (p_x, p_y)$ around which the robot should explore). In the PoAReT GUI, the human operator can issue a HLC through the map component: by drawing an arrow $\boldsymbol{\alpha}$ and by double-clicking a point p. These commands are then integrated in the exploration strategy of the robot that explores autonomously (without human intervention) but following human directives.

In detail, the preferred direction $\boldsymbol{\alpha}$ impacts on the MCDM-based exploration strategy by introducing a new criterion $\mathcal{D}_{\boldsymbol{\alpha}}(f, r)$ that evaluates a frontier f with respect to a robot r and that returns the smaller (better) values the more the vector $r - f_c$, from the current position of robot r to the centroid f_c of frontier f, lies along the direction of $\boldsymbol{\alpha}$. Namely, $\mathcal{D}_{\boldsymbol{\alpha}}(f, r) = \theta(\boldsymbol{\alpha}, r - f_c)$, where $\theta(\cdot, \cdot)$ returns the angle between the directions of two vectors. The areas of interest impact on MCDM in a similar way, by introducing a new criterion $\mathcal{A}_p(f)$ that returns the smaller values the more f lies close to p. Namely, $\mathcal{A}_p(f) = d(p, f_c)$, where $d(\cdot, \cdot)$ returns the Euclidean distance between two points. Criteria $\mathcal{D}_{\boldsymbol{\alpha}}(f, r)$ and $\mathcal{A}_p(f)$ are merged by MCDM with the other criteria discussed in Section 3 in order to evaluate the utility $u(f, r)$ of frontier f for robot r.

With HLCs the human operator can control robots at a higher level of abstraction than waypoint guidance, without losing control over the single robots (as it could happen in case of full autonomy). The higher level of abstraction is relative to the fact that the human operator does not explicitly tell the robots where to go, but gives them some bias about what locations they should (autonomously) choose. Since our HLCs are issued to single robots, they differ from the approach in [8], in which the operator selects interesting areas for the *whole* multirobot system that autonomously coordinates the motion of the robots. Moreover, the "explore along a direction $\boldsymbol{\alpha}$" command is not considered in [8].

4.3 Message Filtering

The human operator, to effectively control a multirobot system, needs not only to issue commands to the robots, but also to receive appropriate feedback information. The robots, during their missions, can encounter difficulties or detect interesting situations, in which cases they send a message to the human operator. When the number of robots is large, missions are critical, and time is scarce, the number of notifications can easily overwhelm the attention capabilities of human operators.

We then introduce a message filtering system in order to display to the operator only important information and limit the workload due to message handling. Messages that robots can send to the base station and that can be displayed to the operator, include: victim detection messages coming from the victim detection modules, feedback messages coming from the motion control modules that signal that an assigned motion task (e.g., reach a waypoint) has been completed, and fault messages that can come from motion modules (e.g., a collision) or from path planning modules (e.g., a frontier cannot be reached).

For each message M coming at time t from module m of robot r, the priority of M is calculated according to three factors:

- The inverse of the current operator workload, $W^{-1}(t) = 100 - \frac{100}{1+100\cdot\exp^{-n(t)}}$, where $n(t)$ is the number of messages currently displayed in the message manager to the operator (see Fig. 1); the larger $n(t)$, the smaller $W^{-1}(t)$.
- The current reliability of module m of robot r, $R_{m,r}(t) = \frac{100}{1+\exp^{-z_{m,r}(t)/4}}$, where $z_{m,r}(t)$ is the difference between the number of messages coming from m that have been archived positively and the number of those that have been archived negatively by the operator, from the beginning of the mission (initially, $z_{m,r}(t = 0) = 0$ for all modules m and robots r).
- The relevance I_M of the message M, which is related to the specific mission. For example, we considered relevance $I_M = 100$ for messages M about victim detection, $I_M = 80$ for messages M about faults, $I_M = 50$ for feedback messages, and $I_M = 10$ for other messages. These values can be dynamically set by the operator during the mission.

The priority of M is then calculated as $p(M) = \frac{W^{-1}(t)+R_{m,r}(t)+I_M}{3}$. Obviously, $0 \leq p(M) \leq 100$. Messages are displayed in the message manager ordered by their priority, after elimination of messages with priority less than 10 (this threshold can be set by the operator), which are assumed to be not relevant.

The rationale of our approach is that, when the workload is high, only very important messages are displayed to the operator. Moreover, the reliability of the sending module m depends on the previous interaction of the operator with the system and accounts for the fact that some modules might be sending unreliable information (e.g., in the case of the victim detection module, due to malfunctioning or to adverse environmental conditions, like scarce light). Finally, the nature of the message (e.g., detection of a victim or successful completion of motion) influences the priority in an obvious way, namely the higher the importance, the higher the priority.

Messages can be removed from the message manager windows in two ways. Firstly, messages are automatically archived after a given time interval (25 seconds in our experiments, but the value can be set by the operator). In this case, the reliability balance $z_{m,r}$ of the module that sent the message is left untouched. Secondly, messages can be removed by the operator, explicitly marking them as positive or negative. In this case, the corresponding reliability balance $z_{m,r}$ is updated accordingly ($+1$ and -1 for positive and negative markings, respectively).

5 Experimental Activity

5.1 Experimental Setup

We implemented our PoAReT HRI system in C++ and we experimentally tested three different system configurations with an increasing number of functionalities. In the *standard* configuration the robots are tele-operated or controlled by waypoint commands. This configuration represents a basic level of functionalities available in many of the works discussed in Section 2 and is used as a baseline for comparison. *High-level commands* (HLC) configuration adds autonomous exploration, driven by high-level commands issued by the operator (Section 4.2). Still no message filtering capabilities is used. Finally, *full functionalities* (FF) configuration includes both improvements proposed in this paper: high-level commands and message filtering (Section 4.3).

We tested our approach in three indoor environments simulated in USARSim (shown in Fig. 2). The environments are created in order to be fully mapped in 5 minutes with almost perfect coordination among a given number of robots: the small environment has 8 victims for runs with 2 robots, the bigger one has 10 victims for runs with 5 robots, and the huge one has 10 victims for runs with 8 robots. Victims are randomly placed and can be detected automatically by the robots or by the operator looking at camera views of the GUI (Section 4.1).

Experimental tests are carried out by 14 volunteers, 7 are expert users of the system (with at least 1 full-day training) and 7 are non-expert users (a similar number of testers

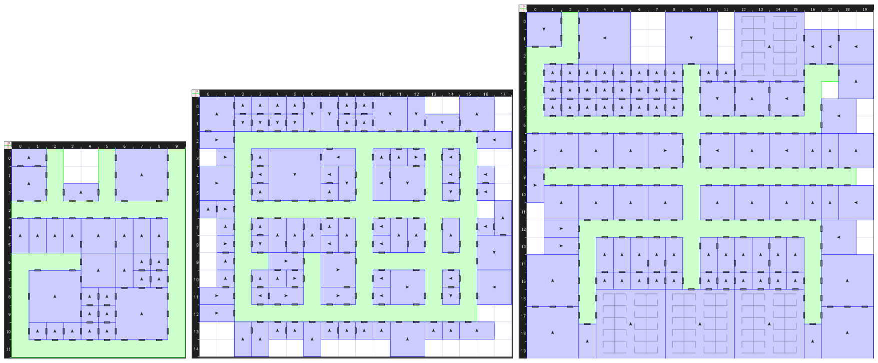


Fig. 2. Test environments (sizes are 27 m × 33 m, 51 m × 42 m, and 57 m × 57 m; corridors are in green, rooms in blue, inaccessible areas in white, and obstacles in black)

has been used also in [1,7,10,20]). Testers are aged between 18 and 27 and have expertise at least in using personal computers. Each tester performs 9 runs (corresponding to three system configurations in the three environments) of 5 minutes each. To limit biases caused by learning, which can be relevant for non-expert users, runs are randomly ordered for each tester.

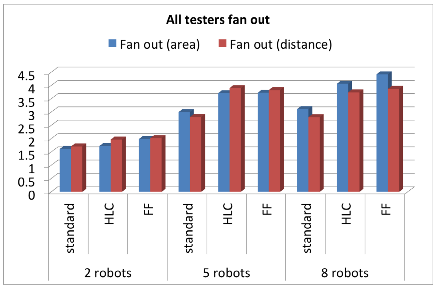
For performance evaluation we use some well-known metrics. *Fan out* is defined as number of robots properly controlled by a single operator. We calculate it by measuring the amount a of area covered (mapped) in a run by all robots. As a benchmark we considered the area $\bar{a}$ mapped by a single robot manually operated by an average expert operator. Fan out is then defined as follows: *fan out* $= \frac{a}{\bar{a}}$. A similar measure is used for fan out relative to distance travelled. Since the main goal of a search and rescue mission is to find as many victims as possible, the percentage of *victims found* is also considered (as in [7]). Finally, we measure the workload (e.g., see [21]) considering the number of cognitive events the operator must face. A cognitive event can be a command issued to a robot, a message received, or an interaction with the GUI.

5.2 Experimental Results

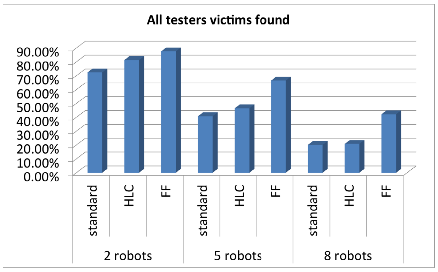
Let start from fan out results (Fig. 3(a)). Fan out relative to area explored and distance travelled shows a relevant grow when HLCs are introduced. These results are statistically significant according to an ANOVA analysis with a threshold for significance p-value < 0.05 [22]: for example, the difference between fan out of standard and HLC configurations with 5 robots has p-value $= 0.019$ for area mapped and p-value $= 0.0037$ for distance travelled. With 8 robots, the p-values are 0.011 and 0.040, respectively. These performance improvements are more evident with 5 and 8 robots because controlling more robots is more challenging. Experts testers perform slightly better than non-expert ones in terms of both area mapped and distance travelled (data not shown here due to space constraints). The difference is more relevant when the number of robots grows: performance of non-experts users in controlling 8 robots is similar to that in controlling 5 robots.

Percentage of victims found (Fig. 3(b)) slightly grows when introducing HLCs but the increase is not statistically significant (for example, p-value $= 0.37$ for the difference between percentage of victims found of standard and HLC configurations with 5 robots and p-value $= 0.83$ with 8 robots). A more evident improvement is obtained by message filtering, leading to statistically significant better performances in FF configuration (for example, p-value $= 0.0010$ for the difference between percentage of victims found of standard and FF configurations with 5 robots and p-value < 0.0001 with 8 robots). This is probably due to a better awareness of the status of every single robot. Expert testers perform slightly better than non-expert ones in terms of victims found, especially when controlling a large number of robots (data not shown here due to space constraints).

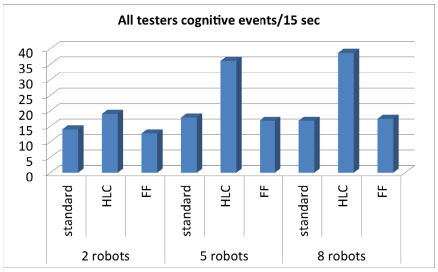
Workload grows, as expected, with the number of robots to be controlled (Fig. 3(c)). An evident increase comes with HLCs: robots explore the environment by themselves, sending feedbacks to the operator and asking for help when facing some difficulty. Feedback messages, together with a more intense action of the robots (as highlighted by the results on fan out), cause an increase of the workload of the operator. Message filtering



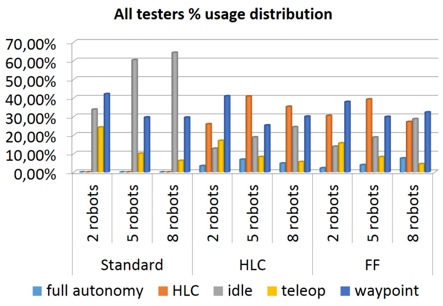
(a) Fan out



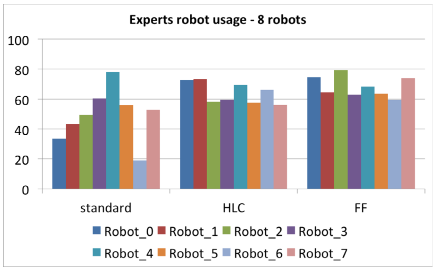
(b) Percentage of victims found



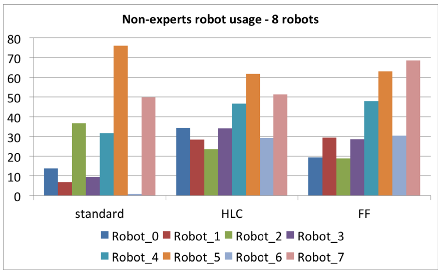
(c) Workload



(d) Robots' state distribution



(e) Travelled distance (in m), experts



(f) Travelled distance (in m), non-experts

Fig. 3. Experimental results (averages are over runs and testers)

effectively limits this drawback of autonomous exploration, reducing the workload to a level comparable to a standard configuration (p-value < 0.0001 for the difference between workload of HLC and FF configurations with both 5 and 8 robots; p-value $= 0.72$ and p-value $= 0.77$ for the difference between workload of standard and FF configurations with 5 and 8 robots, respectively). This is a relevant result because workload is not effectively reduced for large numbers of robots in some other systems, for example in the Steel system [8]. We noticed that expert testers could work with a consistently higher workload with respect to non-expert testers. This can explain the performance gap between the two groups of testers.

We now show some results that provide insights on the impact of our approach on the control of robots during a mission. Fig. 3(d) shows the percentage distribution of mission time (5 minutes) over the possible states of a robot: operating fully autonomously, or with high-level commands, waypoint guidance, or tele-operation, or being idle. At any time, a robot can be in only one of these states. As expected, a drastic reduction of idle time is observed thanks to HLCs (p-value < 0.001 for the difference between idle time of standard and HLC configurations with both 5 and 8 robots). This is a relevant result because idle time is relevant for some other systems, like the Team Michigan system [1]. Even more important is the fact that expert users strongly prefer to issue HLCs instead of letting robots explore fully autonomously. We guess this is due to the better level of control experienced by users with HLCs.

During the runs we noticed that expert users were able to use almost all robots, while non-experts used just a subset of robots, presumably in order to feel more confident with the system. This behaviour is related to the differences in workload we already noticed, and it is more evident when the number of robots grows. In Figs. 3(e) and 3(f) we show the distance travelled (in meters) by single robots for the two groups of testers with 8 robots. In general, robots are used more homogeneously by expert testers. HLCs and message filtering further improve the homogeneity in controlling the robots.

6 Conclusion

In this paper, we have presented the PoAReT human-robot interaction system that allows operators to issue high-level commands to the robots. Experiments show that our approach increases the performance of the system (in terms of fan out and victims found), without worsening the cognitive effort of the operators (in terms of workload) thanks to a mechanism that filters out messages coming from the robots. The PoAReT HRI system allows to effectively control a large number of robots exploiting and driving the potentialities of autonomous exploration. This is one of the reasons that led the PoAReT system to win the 2012 edition of the Virtual Robot competition within the RoboCup Rescue Simulation League, effectively controlling up to a dozen of robots.

Future work will mainly focus on moving from a realistic simulation (like that provided by USARSim) to a robotic system operating in the real world. This will allow to further assess the significance of our approch.

References

1. Olson, E., Strom, J., Morton, R., Richardson, A., Ranganathan, P., Goeddel, R., Bulic, M., Crossman, J., Marinier, B.: Progress towards multi-robot reconnaissance and the MAGIC 2010 competition. Journal of Field Robotics 29(5), 762–792 (2012)
2. Petersen, K., von Stryk, O.: Towards a general communication concept for human supervision of autonomous robot teams. In: Proc. ACHI, pp. 228–235 (2011)
3. Carpin, S., Lewis, M., Wang, J., Balakirsky, S., Scrapper, C.: USARSim: A robot simulator for research and education. In: Proc. ICRA, pp. 1400–1405 (2007)
4. Yanco, H.: Classifying human-robot interaction: An updated taxonomy. In: Proc. IEEE SMC, pp. 2841–2846 (2004)
5. Scholtz, J.: Theory and evaluation of human robot interactions. In: Proc. HICSS (2003)
6. Kaber, D., Endsley, M.: The effects of level of automation and adaptive automation on human performance, situation awareness and workload in a dynamic control task. Theoretical Issues in Ergonomics Science 5(2), 113–153 (2004)

7. Wang, J., Lewis, M.: Human control for cooperating robot teams. In: Proc. HRI, pp. 9–16 (2007)
8. Velagapudi, P., Kleiner, A., Brooks, N., Scerri, P., Lewis, M., Sycara, K.: RoboCup Rescue Simulation League 2010 Team STEEL (USA). Technical report, Carnegie Mellon University, Pittsburgh (2010)
9. Graber, T., Kohlbrecher, S., Meyer, J., Petersen, K., von Stryk, O., Klingauf, U.: RoboCup Rescue Robot League 2012 Team Hector Darmstadt (Germany). Technical report, Technische Universität Darmstadt (2012)
10. Nevatia, Y., Stoyanov, T., Rathnam, R., Pfingsthorn, M., Markov, S., Ambrus, R., Birk, A.: Augmented autonomy: Improving human-robot team performance in urban search and rescue. In: Proc. IROS, pp. 2103–2108 (2008)
11. Amigoni, F., Caltieri, A., Cipolleschi, R., Conconi, G., Giusto, M., Luperto, M., Mazuran, M.: PoAReT Team Description Paper. In: RoboCup 2012 CD (2012)
12. WSS Wireless Simulation Server, `http://sourceforge.net/apps/mediawiki/usarsim/index.php?title=Wireless_Simulation_Server`
13. Comer, D.: Internetworking with TCP/IP, vol. 1. Addison-Wesley (2006)
14. LaValle, A., Kuffner, J.: Rapidly-exploring random trees: Progress and prospects. In: Donald, B., Lynch, K., Rus, D. (eds.) Algorithmic and Computational Robotics: New Directions, pp. 293–308. CRC Press (2001)
15. Nguyen, V., Gächter, S., Martinelli, A., Tomatis, N., Siegwart, R.: A comparison of line extraction algorithms using 2d range data for indoor mobile robotics. Autonomous Robots 23(2), 97–111 (2007)
16. Li, Q., Griffiths, J.: Iterative closest geometric objects registration. Computers & Mathematics with Applications 40(10-11), 1171–1188 (2000)
17. Basilico, N., Amigoni, F.: Exploration strategies based on multi-criteria decision making for searching environments in rescue operations. Autonomous Robots 31(4), 401–417 (2011)
18. Zlot, R., Stentz, A., Dias, M.B., Thayer, S.: Multi-robot exploration controlled by a market economy. In: Proc. ICRA, pp. 3016–3023 (2002)
19. Viola, P., Jones, J.: Robust real-time face detection. International Journal of Computer Vision 57(2), 137–154 (2004)
20. Kane, B., Velagapudi, P., Scerri, P.: Asking for help through adaptable autonomy in robotic search and rescue. In: Bai, Q., Fukuta, N. (eds.) Advances in Practical Multi-Agent Systems. SCI, vol. 325, pp. 339–357. Springer, Heidelberg (2010)
21. Prewett, M., Johnson, R., Saboe, K., Elliott, L., Coovert, M.: Managing workload in human-robot interaction: A review of empirical studies. Computers in Human Behavior 26(5), 840–856 (2010)
22. Pestman, W.: Mathematical Statistics: An Introduction. de Gruyter (1998)