The 2009 IEEE/RSJ International Conference on
Intelligent Robots and Systems
October 11-15, 2009 St. Louis, USA

Negotiation of Target Points for Teams of Heterogeneous Robots: an Application to Exploration

Claudio Rossi*, Leyre Aldama*, Antonio Barrientos*, Alberto Valero**, Carlos Cruz*
Universidad Politécnica de Madrid
Madrid, Spain
claudio.rossi@upm.es

Abstract— **In this paper, we present an application to Search and Rescue of a task negotiation protocol for teams of heterogeneous robots. Self-organization through autonomous negotiations allow the robots to assign themselves a number of target observation points decided by the operator, who is relieved from deciding the optimal assignment. The operator can then focus on monitoring the mission and deciding next actions. The protocol has been tested on both computer simulations and real robots.**

I. INTRODUCTION

Multi-robot systems (MRS) are a very active field of research. A variety of techniques have been proposed in order to approach the problem of coordination in different kinds of applications [1]. Cooperation applications can be roughly divided in two classes: tight cooperation requires a continuous coordination between the robots, like for instance in box pushing and formation keeping. Loose cooperation requires coordination at the beginning of the mission for planning a division of labour, and at given moments of times, when re-planning may be needed. Exploration and mapping are typical applications.

In this work, we focus on exploration, with an application to Search and Rescue (SAR) with a team of heterogeneous robots. The mission is to explore and map a disaster scene in semi-autonomous mode. An operator at the command and control center receives the information of robots' positions and status, partial maps and other sensory information, and have to analyze such information in order to guide the search process. One of his/her duties is to decide to which points of the scenario to send the robots for further exploration, being it mapping an unexplored part or revising an interesting feature (e.g. a possible victim). Once target locations have been decided and assigned to individual robots, these can navigate autonomously to their destinations.

In order to aid the operator in such process and reduce his/her workload, it would be useful to consider the team as a whole, and assign the objective points to the team, instead of deciding for each individual point which is the robot that shall go. Once the team has received the task, it can decide which is the most suited robot for each of the points, taking into account the robots' preferences and limitations w.r.t, e.g. locomotion, sensors, battery status, distance, and achieving an optimal allocation.

*Robotics and Cybernetics Research Group, UPM
**Intelligent Control Research Group, UPM

In other words, a given task, consisting in a set of target points, has to be partitioned in sub-tasks, and sub-tasks have to be assigned to individual team-members for being executed. Market-based [2] and auction [3] techniques are commonly used in this class of problems. Most of the coordination techniques assume that the task subdivision step is performed at a high-level, either by a command and control station or by a specific team-member, and focus on the sub-task allocation problem.

In recent work [7] we have developed a distributed negotiation algorithm that performs a simultaneous task subdivision and allocation, taking into account robots preferences already in the task partitioning stage, and not only after the partitioning has been performed and robots have to bid for the subtasks.

The protocol relies on an abstract concept of task, and the negotiation algorithm based on such an abstraction allows different applications with minimal changes. Let us clarify that, in the context of loose cooperation, with task we mean the object to be divided, and not the activity to be performed on such object. In the exploration task object of this paper, we are mainly interested in partitioning the set of target points and assigning subsets to the agents. The role of their preferences (for instance w.r.t. their capabilities) is encapsulated in the cost/reward the agents associate to the set of points.

Negotiations have been widely studied in the context of socio-economic studies [4] using, amongst others, Game Theory [5]. An example of recent application is electronic commerce using agents [6]. The main problem with game theory approaches is that the theoretical results obtained refer to very simplified models that are not immediately applicable to complex applications. The protocol we propose is based on Rubinsteins alternate-offers protocol [8]. Since Rubinstein's protocol is based on a uni-dimensional good, a search mechanism for the best (counter)-offer had to be devised for the protocol to be applied in real multi-dimensional tasks.

In the following, we will first present our definition of task, how agents take into account costs and rewards to evaluate (sub-)tasks and the negotiation protocol. Section II also describes the representation of the exploration problem with our system. The results of preliminary experiments we have performed to tune and validate the algorithm are described next. Section IV describes the test performed with

978-1-4244-3804-4/09/$25.00 ©2009 IEEE

real robots. Finally, Section V reports the experiments we have performed using the simulation environment used in the RoboCup Rescue competition [10].

II. TASKS AND TASK NEGOTIATIONS

We define a task T as an element of a set $\mathbb{T}$, $T \in \mathbb{T}$. An element of $\mathbb{T}$ is described by a set of k parameters $x \in \mathbb{P}^k$ (without loss of generality we can assume they are all of the same type –in most practical cases x will be an array of real numbers: $x \in \mathbb{R}^k$). Then we can write $T = T(x)$, that is, we consider that a task T is the product of a function that maps a set of parameters into a task: $T : \mathbb{P}^k \rightarrow \mathbb{T}$. A task T has to be divided in R subtasks: $T(x) = \{T_1, \ldots, T_R\}$. Each subtask $T_i, i = 1..R$, can in turn be described by a set of parameters x_i:

$$T(x) = \{T_1(x_1), \ldots, T_R(x_R)\}$$

In general a good subdivision is such that there is minimum overlapping between sub-tasks (ideally null), and such that the subtasks cover the original task. That is,

$$T_i \cap T_j = \oslash, \qquad \forall i, j = 1 \ldots R \qquad and \qquad \bigcup_{i=1}^{R} T_i = T$$

where the operators $\cap, \cup : \mathbb{T} \times \mathbb{T} \rightarrow \mathbb{T}$ are to be defined according to the meaning of the task. In this application a task is a set of target points to be visited, and the intersection ad union of two tasks are the normal set operations. Thus, the objective of the task partitioning is having subsets of the target points assigned to the robots, in such a way that robots have not points in common, and all the points are assigned to a robot. Let $g : \mathbb{T} \rightarrow \mathbb{R}$ be a reward function, giving the value of a (sub)task. Then, the function

$$f : \mathbb{P}^k \rightarrow \mathbb{R} = T \circ g$$

associates a reward to a set of parameters describing a task. We associate to a subdivision $T = \{T_1, \ldots, T_R\}$ an index called global coverage G, that takes into account the total coverage of the subtasks and their pair wise overlapping.

$$G = \sum_{r=1}^{R} f(x_r) - \frac{\sum_i \sum_{j \neq i} g(T(x_i) \cap T(x_j))}{2}$$

Then, the problem of task subdivision can be formulated in the following way:

Given a task T and a number R of agents, find the R sets of parameters x_i, $i = 1 \ldots R$, such that G is maximized:

$$max_{x_1 \ldots x_r}(G).$$

Note that G is a global performance index. During the negotiation, each robot uses its own reward function g_i to evaluate a task. Hence, the robots can give a different value to the same task, depending on their characteristics.

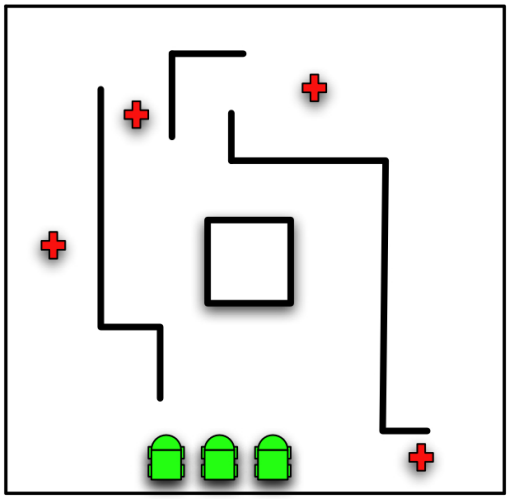

Fig. 1. An example of the scenario: a team of robots is assigned a group of target points to visit (typically along the frontiers, i.e. the limits of the explored area, or locations where interesting features shall be analyzed, such as potential victims.

A. Evaluation of a task

As mentioned earlier, during the negotiation each robot has to evaluate the cost and reward of a given task. To this aim, it takes into account its internal parameters to evaluate the cost of executing the task, the start-up cost (for instance, to reach the execution site), specific constraints (e.g. forbidden zones, turn angles, sensors) and the general reward associated to the task (expressed as function g).

Thus, given the complete task T, evaluation function g_i of subtask S_i for agent i takes the form:

$$g_i(S_i) = g(S_i) + dim(S_i) - \sum_{j \neq i} g(S_i \cap S_j) - C$$

where $dim()$ is a dimension measure. In this case, $dim(S_i)$ is the number of points of S_i normalized w.r.t. the total number of points, and function $g(S_i)$ the total distance to be travelled to visit all the points is S_i normalized w.r.t. the length of the minimum tour that connects all the points of T. The last term defines the penalty for overlapping sub-tasks (points in common with other robots). Finally, element C accounts for the constraints. All terms have a weight factor used for tuning the mission-specific behavior not expressed in the equation for simplicity. More factors can be included in function g_i for other mission-specific costs or rewards.

B. Task Negotiations

A given task T can be executed by a team of R robots, after a suitable subdivision of the task has been performed, and an assignment of the subtasks to the robots have been established. Our aim is to perform these two actions simultaneously and in a distributed way. In our system, the number of sub-tasks is determined by the number of robots willing to participate in the negotiation. In the following, we will describe the negotiation protocol for the case $R = 2$.

We assume that robots aim at maximizing their reward, the only limitations being their available resources (endurance, computation power, battery consumption etc.). Thus, in a negotiation, each agent will try to maximize its reward by (i) trying to get the biggest possible subtask and (ii) minimizing overlapping with other agents task. Each agent starts proposing the biggest possible share for itself, and reduces it until the counterpart finds it acceptable. In this way, a good near-optimal solution can be achieved, and global index G is optimized in a distributed way, without even being computed explicitly.

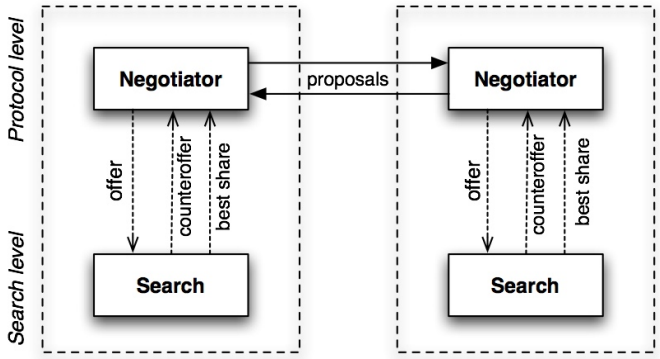

Fig. 2. Architecture of the negotiation module.

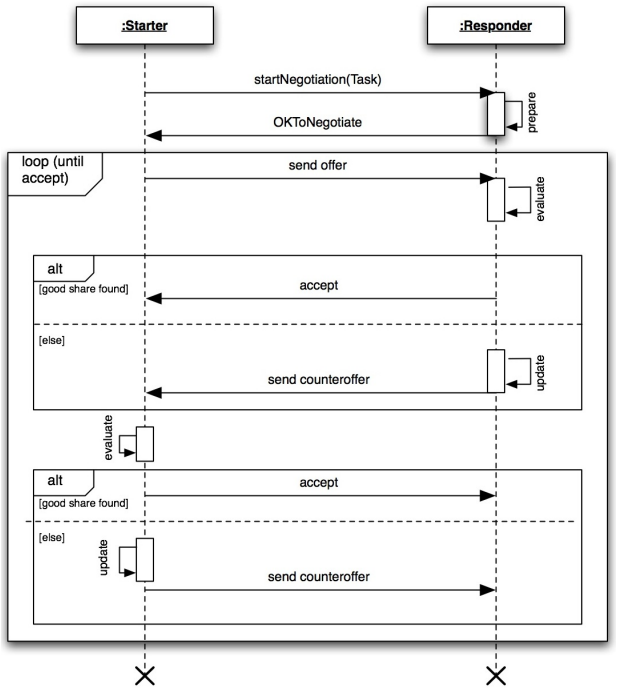

Fig. 3. Negotiation protocol. The first counteroffer is the maximum possible for the robots

In the alternate-offers protocol proposed by Rubinstein, each part of the bilateral negotiation, in turn, proposes a subdivision of a uni-dimensional good of size 1. The responder can agree with the subdivision, or disagree with it, and in this case it has to propose a counteroffer. The protocol assumes that each part has a target (desired) reward, and a negotiation cost called *discount factor* that makes the target reward decrease at each step, imposing a time pressure to the reaching of an agreement. Such protocol has interesting theoretical properties. By applying discount factors as negotiation cost, it guarantees a termination and can forecast the final agreement, which will be a perfect equilibrium in the sense of Game Theory.

However, the theoretical results are not immediately applicable to the multi-dimensional case. In the multidimensional case, given a proposal $x \in \mathbb{P}^k$ on the whole task T_0, an agent shall search the space $T_0 \backslash T(x)$ to decide if the share it would get is acceptable and to generate a counteroffer, since many different configurations are possible.

Thus, we divide the negotiation in two levels: the protocol level and the proposals evaluation and generation level (see Fig. 2). The protocol level is governed by parameters such as impatience to reach an agreement (time pressure as discount factor) and desired target reward. Moreover, at each new offer received, it estimates the other agent's discount factor, which is private information, in order to estimate the maximum possible share it can expect, according to Rubinstein's theory, and updates its desired share accordingly. The protocol is depicted in Fig. 3.

The proposal generation level searches the space for a good share given a proposal from the other part, taking into account its own resources, parameters and limitations. This level is also responsible for updating counteroffers. In fact, in a multi-dimensional space there are may ways they can be updated. In this level a search procedure is implemented that looks for the best combination of parameters $x \in \mathbb{P}^k$ that maximizes its objective function g_i. In the current implementation, this search is performed by a heuristic algorithm that calculates the minimum path to visit all the points that are not claimed by the other negotiators. The update function aims at reducing the offer in such a way to reduce the overlapping (points in common).

In case $R > 2$, negotiation rounds can be performed. Each robot, in turn, makes its proposal. If everybody agrees the negotiation ends, otherwise offers are reduced in the following round.

When an agreement has been reached, the result is a subdivision of the original task and at the same time an assignment of the sub-tasks. Note that one agent does not need to know information about the private characteristics of the team-mates. The only information it needs is their offers, in form of an array of parameters $x \in \mathbb{P}^k$.

III. SIMULATIONS RESULTS

In order to assess the effectiveness of the proposed protocol, we have performed some simulations with a varying number of randomly positioned target points and different initial position of the robots. We have compared the total distance that the robots will travel in case of negotiated allocation and the optimum total travel distance. The optimum has been computed with a simple enumerative algorithm that

guarantees to find an optimal allocation. Only Euclidean distance has been considered for the comparison. All the robots used the same discount factor $\delta = 0.988$. The percentage of additional path that the robots have to travel w.r.t the optimal solution is reported in Table I. All entries are the average over 10 runs [1].

Roughly, if we consider the total distance travelled, the solution agreed is worst of by a factor between 30% and 60%. However, if we consider the longest path travelled by each robot (that relates to mission time), the difference is reduced to $5\% - 10\%$. This happens because, depending on the positions of the robots and of the points, often it is cheaper to let fewer robots visit many points, for example if the points are close to each other. In this case, the total distance travelled is lower, but the individual paths of the robots are longer (consider the extreme case of one robot visiting all the locations). Clearly, such kid of solutions are not desirable since imply a lower degree of parallelism, to the extreme case that some of the robots are not employed at all. The longest path relates to mission time, which sometimes can be a crucial factor, as in the case of the robocup competition. The plot of Figure 4 shows the average difference between the optimum solution and the result of the negotiations, for a fixed number of robots.

TABLE I

RESULT OF THE NEGOTIATION W.R.T OPTIMAL SOLUTION: TOTAL DISTANCE. PERCENTAGE OF ADDITIONAL PATH TRAVELLED.

No. of robots	Number of points				
	2	*3*	*4*	*5*	*6*
2	28%	56%	19%	38%	50%
3	35%	34%	15%	29%	55%
4	27%	55%	23%	61%	64%
5	28%	46%	19%	55%	64%
6	28%	47%	21%	65%	60%

TABLE II

RESULT OF THE NEGOTIATION W.R.T OPTIMAL SOLUTION: LONGEST PATH OF THE ROBOTS. PERCENTAGE OF ADDITIONAL PATH TRAVELLED.

No. of robots	Number of points				
	2	*3*	*4*	*5*	*6*
2	5%	12%	12%	6%	9%
3	5%	8%	8%	13%	11%
4	9%	9%	9%	10%	8%
5	10%	12%	12%	7%	8%
6	10%	8%	9%	14%	9%

IV. EXPERIMENT WITH REAL ROBOTS

We have deployed the negotiation agents on a fleet of mobile robots (WiFiBots 4G, equipped with electronic compass and GPS for self-localisation) running suitable navigation software developed in the framework of the project.

The scenario was composed of three robots an five target points, located on an area of 15×15 meters. Figure 6 shows

[1]Here, we are comparing only the quality of the final solution. As far as computational efforts is concerned, it is worth noting that finding the optimum assignment took up to two orders of magnitude longer than negotiation time.

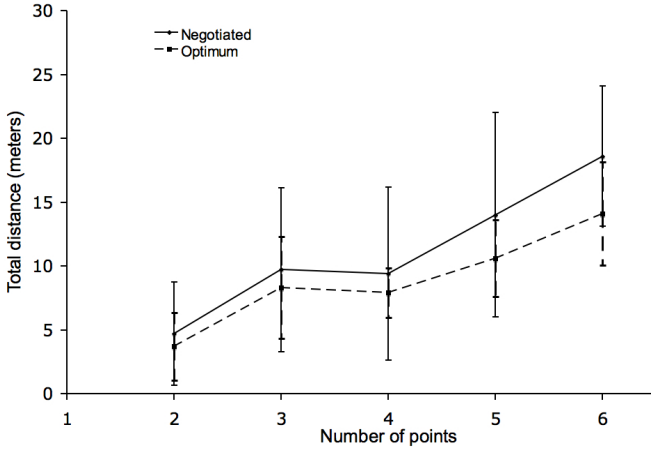

Fig. 4. Detail of total distance travelled for three robots. Average over ten runs.

two different allocations, depending of different parameters settings (discount factor of the robots). In this experiment the optimum solution as far as total travelled path is of 15.58 meters, while the total travelled distance of the negotiated allocation was of 17.87 (left) and 18.01 meters (right), approximately 15% longer than the optimum. The mission time, accounted for as the longest path of the individual robots, was of 28.16 seconds for the optimum solution (corresponding to the time it takes robot $R2$ to complete its task, for a travelled path of 9.29 meters), while it was of 26.81 seconds for the negotiated solution (corresponding to the time it takes robot $R1$ to complete its task, for a travelled path of 8.85 meters), approximately 4% slower.

This experiment also shows how the sub-task allocation depends on the discount factors of the robots. In fact, different discount factors lead to different expected values, as described earlier. Thus, agents can accept solutions that are worst for them, if they are more impatient. In the example shown in Figure 6 it is clear that the two rightmost robots (robot $R2$ and robot $R3$) agree to leave point $P4$ to the teammate according to their discount factor: the one with the smaller discount factor (i.e. the most impatient) gives up the negotiation earlier, getting a smaller share of the task. Figure 5 shows the detail of the last negotiation step. The target rewards are computed according to Rubinstein's theory as $M_1 = 0.4, M_2 = 0.4, M_3 = 0.2$. The share $S_2 = \{P3, P4\}$ robot $R2$ ask to robot $R3$ is rejected by this, since the share it would get, $S_3^* = \{P5\}$ has a value $g_3 = 0.13$ which is smaller than its target reward $M_3 = 0.2$. However, $R3$ then updates its previous offer releasing point $P4$ (note how the value of the new proposal $g^{(t=15)}$ is δ_3 smaller then the previous $g^{(t=14)}$). Such proposal is accepted by $R2$ since the value it gets is higher the its target reward.

V. AN APPLICATION TO SEARCH AND RESCUE

As an application of the target points negotiation, we implemented the negotiation capability as a module of the open RDK architecture [9] in the multirobot system used in the

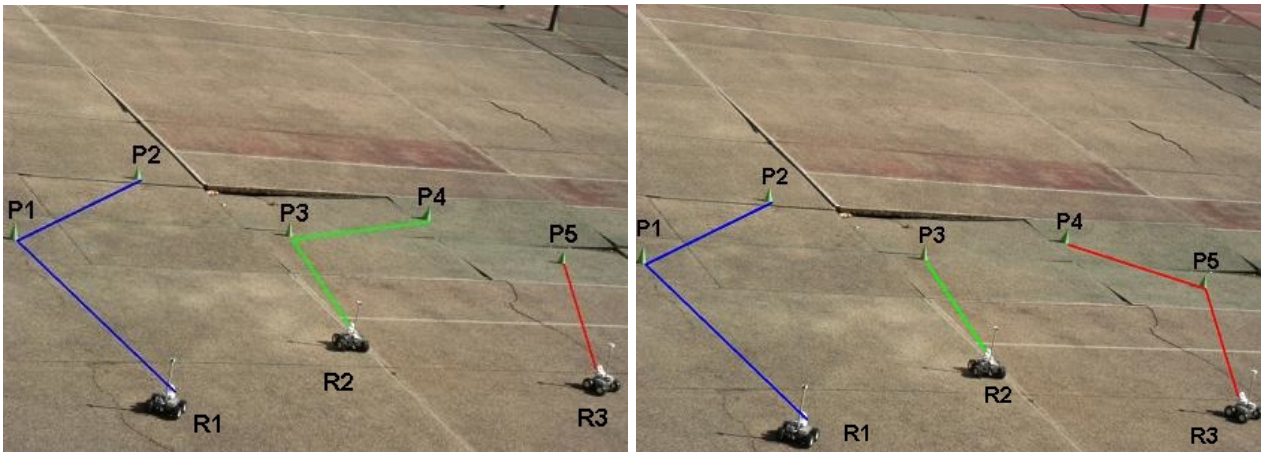

Fig. 6. Snapshots of the tests with real robots. Different solutions are agreed according to the impatience of the negotiators. In this example $\delta_1 = 0.887, \delta_2 = 0.877, \delta_3 = 0.867$ (left) and $\delta_1 = 0.887, \delta_2 = 0.867, \delta_3 = 0.877$ (right). Green cones represent target locations, and superimposed lines represent the final allocation.

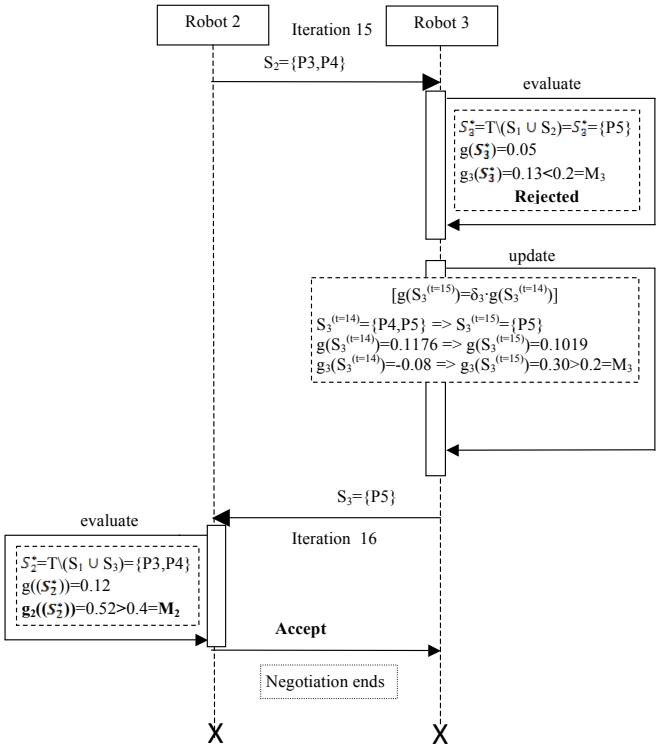

Fig. 5. Detail of the last negotiation steps between robots $r2$ and $R3$ for assignment of Fig. 6 (left). Both robots $R2$ and $R3$ want to visit point $P4$.

context of the robocup rescue [10] competition ("simulation" league).

In this system, the human-robot interface consists of a base station that allows the operator to monitor the progress of the mission, know the status of the robots, give directives to them and even tele-operate them (see Fig. 8). Such interface allow choosing between four operation modes: the tele-operation and safe-teleoperation allow to operate only one robot. The safe tele-operation consists in using robots navigation sensors to override commands in case on danger (e.g. proximity to a wall causes the robot to stop despite ordered to advance). On the opposite side, in the autonomous mode, the robots explore the area choosing autonomously the locations to explore. In the semi-autonomous mode, the operator specifies a number of interesting locations that the robots shall visit, and robots navigate autonomously to them. In the latter operation mode, the operator has to specify which robot goes to which points. In the case the number of robots composing the team grows, it become increasingly difficult for the operator to control them, even in the case of semi-autonomous mode. For this reason, we propose to exploit our robot-coordination mechanism and give the operator the vision of the team as a whole: the operator decides the locations to be visited (the *task*), and let the *team* organize itself in order to accomplish with the task, thus reducing the operator's overload. Figure 9 shows two stages of a test. The square in the center are the robots, and the crosses are the points the operator would like to be visited to enlarge the explored area and to check a possible victim (bottom-right location, see also Fig. 7).

Fig. 7. Example of the USARSim simulated world.

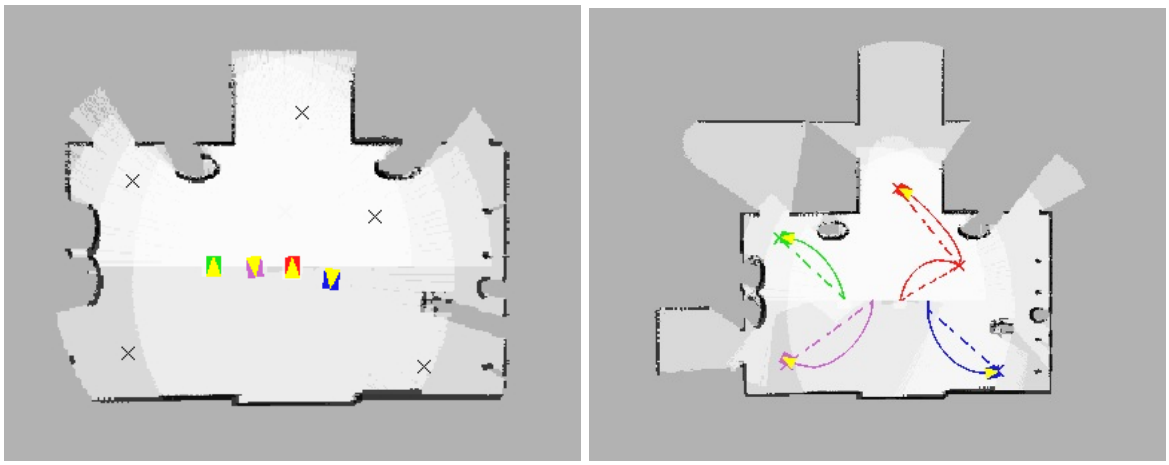

Fig. 9. Allocation of target points in robot search and rescue: initial (left) and final (right) configuration. The operator indicates the locations to be visited by the team (crosses of the left image) and the robots autonomously decide a subdivision. The image on the right shows the robots final positions and path travelled. Note how the map is updated as consequence of the exploration.

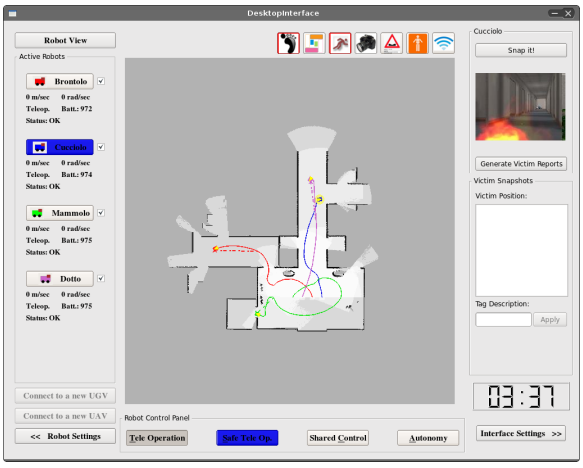

Fig. 8. The interface used in robot rescue. The user has different views of the explored area and monitors mission and robots' status information.

VI. CONCLUSION

We have presented a negotiation mechanism capable of partitioning a given task among a group of robots. The algorithm is distributed and takes into account the robots' preferences and limitations. In this work we have applied the task negotiation to semi-autonomous exploration.

The exploration and mapping task is a classic application on multi-robot systems, since the advantage of using multiple agents working simultaneously is evident. In semi-autonomous exploration, such as search and rescue, an operator monitor the evolution of the mission and take decisions on where to send the robot to explore. However, controlling multiple robots is not a trivial task, even in the case their number is not high, and for three or more robots it becomes quite difficult. Thus, the idea is to provide the team with self-organizing capability, and allow the operator to consider the team as a whole, reducing his/her workload.

In these experiments, only euclidean distances were considered in the cost function. However, the system is intended to work considering the estimation of real distances by the robots and/or resources needed to reach them, which can be easily done by simply changing the Euclidean distance function with any other more sophisticated function. Although the negotiated solution is not optimum, it does achieve a good task partitioning, if we consider mission time and degree of parallelism in the execution of the task.

Future work will be aimed at testing the system in the robocup competition, and at incorporating actual robot status information such as battery level for the evaluation of a task in the experiments with real robots.

ACKNOWLEDGMENTS

This work is partially funded by project FRACTAL (fleet of autonomous aerial and ground robots - DPI2006-03444) of the Ministerio de Educacíon y Ciencia (MEC) of Spain.

REFERENCES

[1] L.E. Parker, Current Research in Multi-Robot Systems, J. Art. Life and Robotics, Vol. 7, 2003.
[2] M. Bernardine Dias and A. Stentz, TraderBots: A Market-Based Approach for Resource, Role, and Task Allocation in Multirobot Coordination, Technical report, CMU-RI, 2003.
[3] B. P. Gerkey and M. J. Mataric, Sold!: Auction Methods for Multirobot Coordination,IEEE Trans. on Robotics and Automation, 18:5, 2002.
[4] K. Chatterjee and L. Samuelson. Bargaining with two-sided incomplete information: An infinite horizon model with alternating offers. Review of Economic Studies, 54:175-192, 1987.
[5] M.J. Osborne and A. Rubinstein. A course in game theory, MIT Press, 1994.
[6] T. Sugasaka, K. Tanaka, R. Masuoka, A. Sato, H Kitajima, F Maruyama, An Agent-Based System for Electronic Commerce Using Recipes, in Proc. of the Seventh International Conference on Parallel and Distributed Systems, 2000.
[7] C. Rossi, L. Aldama, A. Barrientos, Simultaneous Task Subdivision and Allocation for Teams of Heterogeneous Robots , in Proc. of Intl. Conference on Robotics and Automation (ICRA), 2009.
[8] Ariel Rubinstein, Perfect Equilibrium in a Bargaining Model, Econometrica, Vol. 50, No. 1, 1982.
[9] D. Calisi, A. Censi, L. Iocchi, and D. Nardi, OpenRDK: a modular framework for robotic software development. In Proc. of Int. Conf. on Intelligent Robots and Systems (IROS). Nice, France. September 2008. pp. 1872–1877.
[10] http://www.robocuprescue.org/