2009 IEEE International Conference on Robotics and Automation
Kobe International Conference Center
Kobe, Japan, May 12-17, 2009

Simultaneous Task Subdivision and Allocation for Teams of Heterogeneous Robots

Claudio Rossi, Leyre Aldama, Antonio Barrientos
Robotics and Cybernetics Research Group
Universidad Politécnica de Madrid
Madrid, Spain
claudio.rossi@upm.es

Abstract— **In this paper, we present a negotiation protocol for simultaneous task subdivision and allocation for heterogeneous multi-robot systems. An abstraction of the concept of task is presented that allows to apply the protocol on a variety of tasks. The negotiation adopts Rubinstein's alternate offers protocol, where offers are evaluated and generated using a heuristic search step. The protocol has been tested on computer simulations.**

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. Behaviour-based [2] and schemas [3] are examples of techniques suitable for the first class of coordination problems, while market-based [4] and auction [5] techniques are commonly used in the second class of problems.

In this work, we focus in loose cooperation. In this class of problems, a given task has to be partitioned in sub-tasks, and sub-tasks have to be assigned to individual team-members for being executed. 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. This approach, although applied with success in many applications, has two principal drawbacks: first, it is not really distributed, since the task partitioning is done in one place, and second, the partitioning algorithm is usually considered outside the coordination protocol. Often, the details of how the original task is partitioned are not given at all. Moreover, the robots preferences and limitations are considered only in the assignment stage, when the robots decide whether to accept (or opt for) a task or not.

Such features do not suit our need of a fully distributed approach that should consider robots capabilities already at the task partitioning stage. This feature is especially important when the robots that compose the team are of different kinds with respect to mobility and equipment, and the tasks they can perform thus vary significantly. The result of a task partitioning that does not take this into consideration may be unfeasible. The negotiation protocol we have developed performs a simultaneous task subdivision and allocation, taking into account robots preferences.

Our multi-robot system is not designed with a particular task in mind. Rather, it serves as a platform for operating in different mission scenarios. Our aim is to develop a cooperation system that is capable of working with different kinds of tasks (cf. Fig. 1). For this purpose, an abstract task concept should be defined. A 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. For example, if the task is surveying a given area, we are mainly interested in partitioning the area and assigning sub-areas to the agents. The activity the agents will have to perform and their preferences (for instance w.r.t. their capabilities) have an important role in the negotiation. This role is encapsulated in the cost/reward the agents associate to the task.

Negotiations have been widely studied in the context of socio-economic studies [6] using, amongst others, Game Theory [7]. An example of recent application is electronic commerce using agents [8]. 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 [9]. Since such 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. To the best of the authors knowledge, the only similar approach has been proposed in [10] [11], where a co-evolutionary genetic algorithm is used to negotiate the payoff matrix of a coordination game (using a trusted third party), and then an optimal agreement between the parts is searched for reasoning on the matrix.

In the following, we will first present our definition of task and how agents take into account costs and rewards to evaluate (sub-)tasks. Section III briefly describes Rubinstein's alternate offers paradigm and the negotiation protocol we

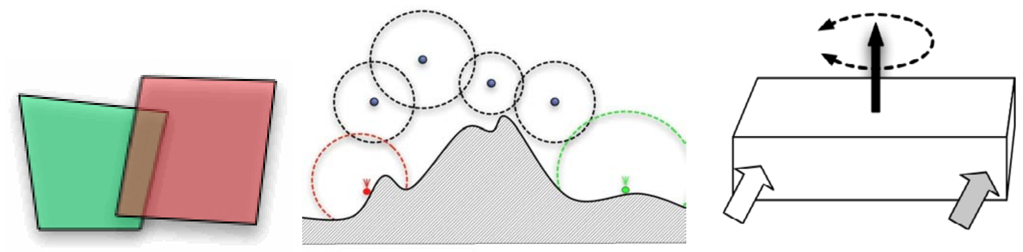


Fig. 1. Examples of instantiation of tasks: areas, communication ranges and torques in box pushing.

have developed based on this, and its extension to the case of more than two negotiators. Finally, Section IV describes the tests performed on different instantiations of tasks and analyzes the results.

II. TASKS

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 $x \in \mathbb{P}_1^{k_1} \times \ldots \times \mathbb{P}_h^{k_h}, \sum_i k_i = k, \mathbb{P}_i, i = 1 \ldots h$, being parameters types. 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,

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

where the operators $\cap, \cup : \mathbb{T} \times \mathbb{T} \rightarrow \mathbb{T}$ are to be defined according to the meaning of the task. Note that there can be exceptions, depending on the application (for example, in a communication relay application, the overlapping between the range of action of two robots must not be null). 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.

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:

$$\begin{aligned} g_i(S_i) = \quad & g(S_i) + dim(S_i) - dist(pos_i, site_i) - C \\ & - \textstyle\sum_{j \neq i} dim(S_i \cap S_j) - dim(S_i \setminus T) \end{aligned}$$

where $dim()$ is a dimension measure (e.g. area, length, number of targets) and $dist()$ is a distance function from the initial position of the robot to the task execution starting point. The last two terms are important as they define the penalty for overlapping sub-tasks and for the part of S_i outside T. Finally, element C accounts for the constraints. All terms can 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.

III. 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. Let us first discuss the case $R = 2$. We assume that robots aim at maximizing their

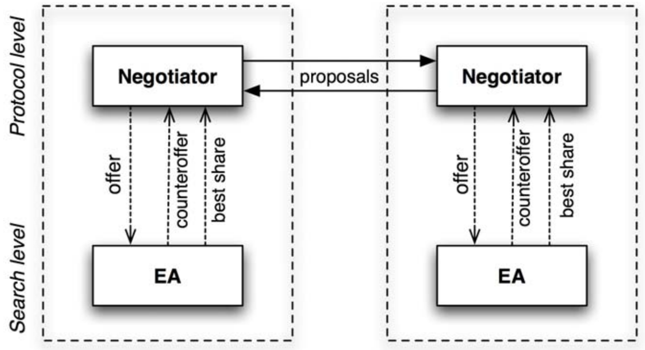


Fig. 2. Architecture of the negotiation module. The search level is currently implemented with an Evolutionary Algorithm.

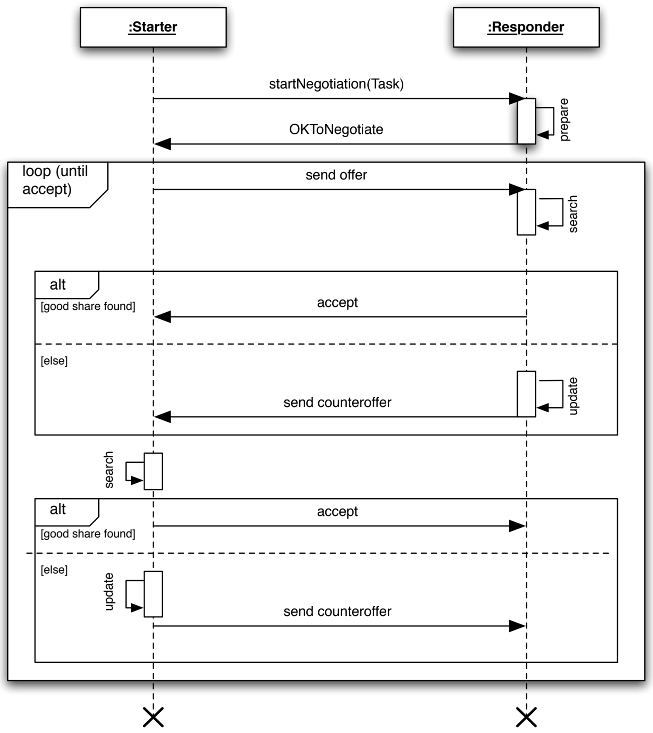


Fig. 3. Negotiation protocol. The first counteroffer is the maximum possible for robot i. The counteroffer update tries to produce a value that is close to a factor of δ_i smaller than the previous.

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 (see next section).

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, 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. Let the discount factors be $0 < \delta_1 \leq 1$ and $0 < \delta_2 \leq 1$, and let $y_{t+1} = y_t \cdot \delta, t = 1 \ldots n$ be the update rule of target share y. Rubinstein's theory guarantees that one perfect equilibrium point exists such that the share y the initiator agent will get is:

$$y = \frac{1-\delta_1}{1-\delta_1\delta_2}. \tag{1}$$

If the discount factors were known to both parts, each could know *without negotiating at all* which will be its share. However, these are not immediately applicable to the multi-dimensional case. In the uni-dimensional case, when an agent makes a proposal p of what it would like to get, it is immediate that the other would get $1-p$. 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 it 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 an Evolutionary Algorithm, specifically a (μ, λ)-Evolutionary Strategy [12], where $\mu = 1$, $\lambda = 6$ and a candidate solution is an array $x \in \mathbb{R}^k$ encoding a set task of parameters. The update function aims at reducing the offer in such a way to reduce the overlapping. In order to comply with Rubinstein's theory hypothesis, the new offer should have a dimension $dim(T^t) = \delta \cdot dim(T^{t-1})$.

When an agreement has been reached, the result is a subdivision of the original task and at the same time an

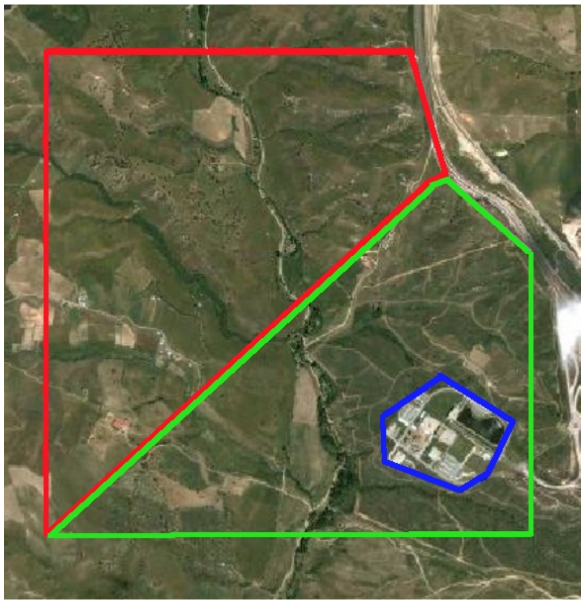


Fig. 4. Negotiation with forbidden area (buildings contained in the blue area). Red agent, an aerial vehicle, refuses all proposals including the no-fly zone.

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 $x \in \mathbb{P}^k$.

A. Learning

Each time a new offer is received, both the search in the parameters' space for their best share and the generation of the counteroffer are not started from scratch. Rather, they take as starting point the last better result. In this way, we can say that the search algorithm learns the opponent's preferences and constraints, and produces offers that implicitly depend on them. The no-fly zone and box pushing applications described below are an example. In both cases, one agent is constrained to avoid certain parameters' combinations (in one case defining a forbidden area, in the other the maximum push force). Although the other agent is not aware of such constraints, as they are private information (may be result of a partial failure) it will learn to generate offers that avoid the forbidden configurations.

Moreover, explicit learning is performed by estimating the opponent's discount factor. The only restriction imposed by Rubinstein's protocol is that is that such factor must be constant.

B. Extending to more than two negotiators

When there are more than two agents, the proposed negotiation protocol can be extended in several ways. It is important to point out that the difference between a two-party and a three-party game reflects a basic qualitative difference between the types of processes that take place within the negotiation. The involvement of any more than three parties, can be seen as an extension of a three-party process [13]. Hence, let us focus on the case $R = 3$.

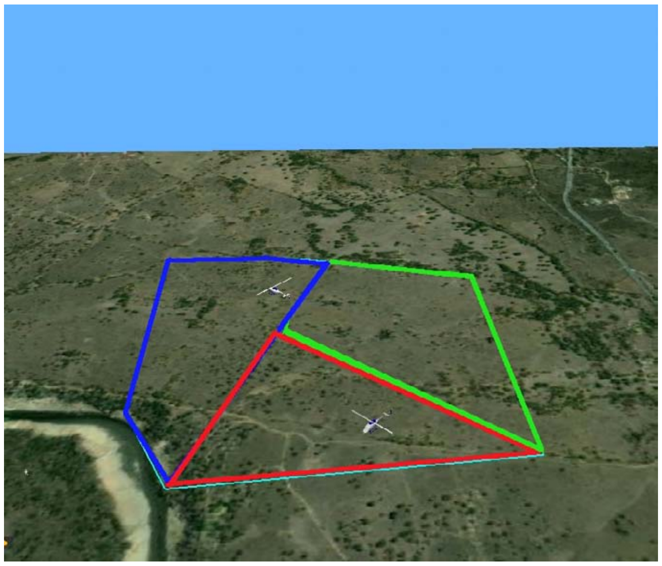


Fig. 5. Negotiation with three agents using rounds.

The simplest extension is to do negotiation rounds where each agent, in turn, makes its proposal. At the end of the round, if all agents are satisfied the negotiation is closed, otherwise another round starts. In this case, each agent considers the union of the two proposals it receives when negotiating, as if the other two agents would form a coalition against it and act as a single agent. In this case, the estimation of the expected maximum share y (which determines the termination of the negotiation, Eq. 1) is less precise, since the combination of the two offers is not guaranteed to this shrink by a constant factor, as requested by Rubinstein's theory.

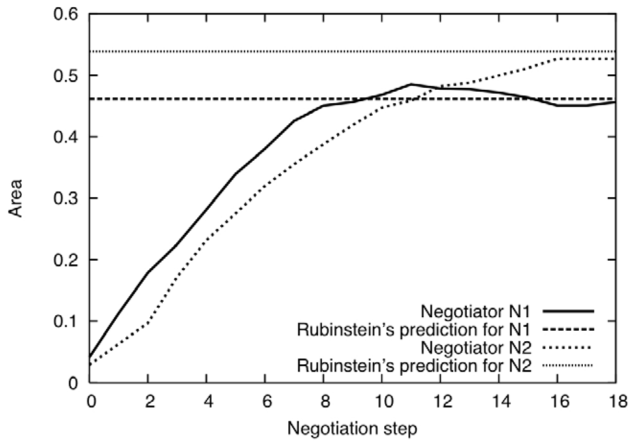
IV. SIMULATIONS RESULTS

In order to assess the effectiveness of the proposed protocol, we have performed several simulations with different examples of task instantiations using the tool Webots [14].

Figure 4 shows how a hard constraint, a forbidden area, makes the agents agree on a subdivision that exclude such area from the part obtained by the constrained vehicle. The subdivision of an area among three agents is depicted in Figure 5.

Figure 6 shows an example the shares of area partitioning between two agents. The left plot shows how the shares the agents obtain are actually close to values predicted by Rubinstein's theory. The Plot on the right shows how the overlapping task is reduced during the negotiation, and how global coverage G is very close to the optimum value (in this, case the total area T). Also, note how global coverage is maintained throughout the negotiation. In other words, the whole area is covered by the two robots at all times.

In the communications relay experiments (Fig. 7), $dim(T)$ is the distance between the two points to connect, and G is the length of the shortest path between agents obtained, which is longer due to the presence of an obstacle (buildings). In this case, the vehicles negotiate positions.



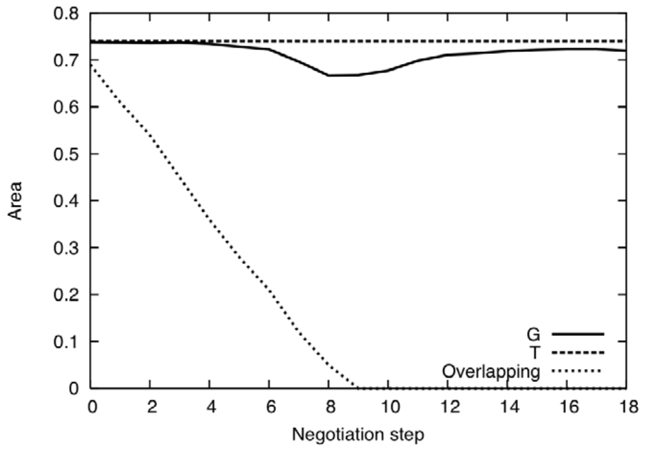


Fig. 6. Negotiation on how to partition an area. Best share and Rubinstein predictions (left). Total area T and global coverage G (right). $\delta_1 = 0.945, \delta_2 = 0.955$, both agents start claiming 1.

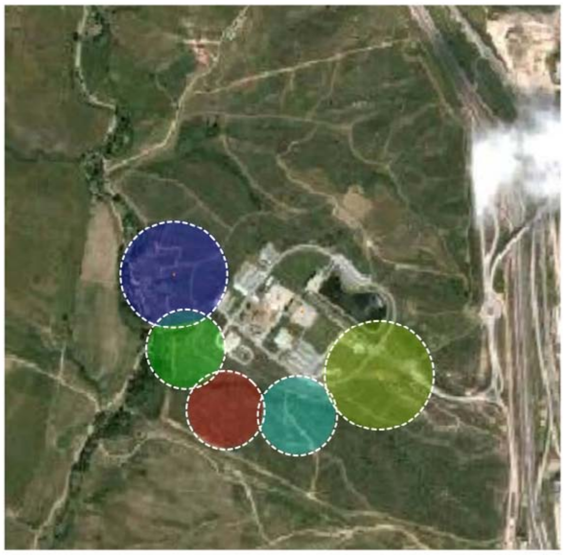


Fig. 7. Communication relays. Three ground vehicles have to position themselves in order to guarantee communications between two fixed stations (outermost circles). In total there are five agents negotiating. Circles represent signal range.

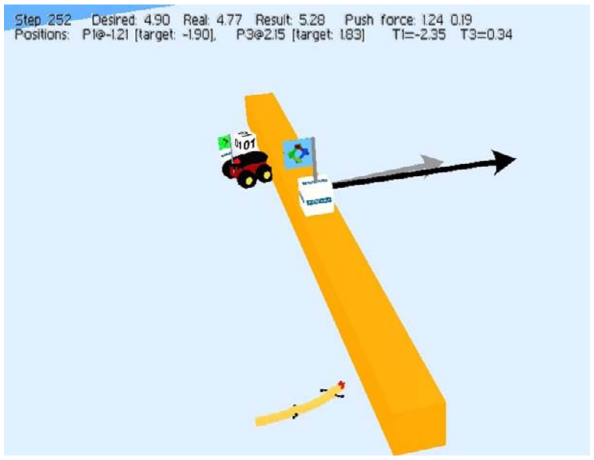

Step 252 Desired: 4.90 Real: 4.77 Result: 5.28 Push force: 1.24 0.19
Positions: P1@-1.21 [target: -1.90], P3@2.15 [target: 1.83] T1=-2.35 T3=0.34

Fig. 8. Snapshot of box pushing experiment. Two robots of different kind (wheeled and legged) have to negotiate push position and force in order to make the box move in a given direction. The arrows depict target direction (black) and obtained heading (gray).

Table I shows that negotiations only take tho order of seconds to conclude and that the global performance index G is optimized. Each figure is the average of 10 runs. Note that not always the optimum value for G is reached, but this is due to time pressure: agents are forced to reach an agreement soon. In applications where more precision is needed, this can be reached by imposing less time pressure.

TABLE I

SUMMARY OF EXPERIMENTS.

Experiment	*Agents*	*Steps*	*Time (sec.)*	*dim(T)*	*G*
No fly zone	2	28	1.82	0.67	0.66
Area Survey-3	3	58	2.72	0.74	0.73
Area Survey-4	4	65	3.53	0.74	0.72
Comm relay	5	9	0.36	0.90	1.02

To conclude the section, let us present a case of tight cooperation where our approach can be used, through periodic renegotiation. Figure 8 shows the negotiation of pushing force and direction between two agents, a wheeled robot and a legged robot [15]. In this case, the task to be negotiated is how to push the box in the desired direction. The parameters of the task are position and force, i.e. a torque to apply for a fixed period of time Δt, in order to rotate the box towards the target direction. In this case, periodic renegotiation is needed because the effect of performing the subtasks modifies the environment (difference between target and actual direction) and hence the torques to apply must be recalculated. Figure 9 shows the target direction, the real heading of the box and the torques agreed at each renegotiation. The legged robot (Agent 2) can push with a force that is lower than the one achieved with the wheeled rover, therefore it prefers to stay in an more extreme position and push less stronger. Again, the rover does not have any knowledge of that, it simply agrees to comply with such preferences during the negotiation process.

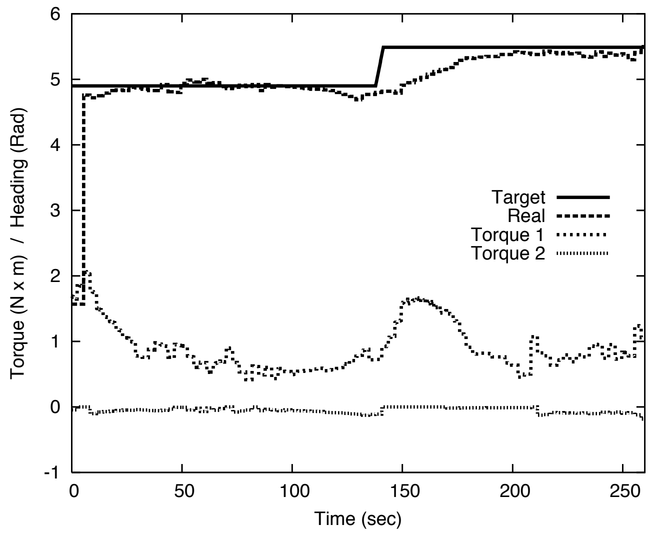


Fig. 9. Box pushing experiment. This Figure shows the target and resulting heading of the box (top curves). The torques agreed in order to reach the target heading are shown in the two bottom curves. Renegotiation is performed each $\Delta t = 3$ seconds. At time $t = 140$ seconds the target direction is changed (a new task is assigned), and the two agents adapt to the change of task.

V. CONCLUSION AND FUTURE WORK

The main contributions of this paper are two. First, we propose a formal definition of the concept of task, general enough that allows expressing different problems. The negotiation algorithm implemented using such formulation then applies to a vast variety of multi-robot tasks. Second, a negotiation protocol that takes advantage on theoretical results that guarantee some important properties such as termination and prediction of the outcome.

Experiments conducted in computer simulations show the effectiveness of the proposed approach, and the adherence of the numerical results with the theoretical results coming from game theory. The focus of this work was on loose cooperation. Nevertheless, the simulations on box pushing showed that our approach can also be adapted for some cases of tight cooperation problems, through periodic renegotiation of the tasks, provided that during the time between negotiations the agents can work almost independently. This means that the approach wouldn't work, for instance, on the cooperative transportation of a body rigidly connected to the robots.

We are currently working on two fronts. On the practical side, we are deploying the negotiation algorithms on our fleet of aerial and ground autonomous vehicles for testing with real robots, and performing tests with different kind of tasks. Although the results presented refer to simulations, the first tests on real robots confirm the effectiveness of the negotiation protocol we propose, which just required the adjustment of the parameters of the cost function for the real robots characteristics.

On the algorithmic side, we are studying how to incorporate the estimation of the opponents discount factors for deciding the termination of the negotiations, in the case $R > 2$. Moreover, for this case, other extensions are possible, such as bilateral negotiations and coalitions. In both cases the negotiation would be reduced to a series of bilateral negotiations. We are currently exploring these options.

ACKNOWLEDGMENTS

The first author is ”Ramon y Cajal” research fellow. This work is 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] L.E. Parker,L-ALLIANCE: Task-Oriented Multi-Robot Learning in Behaviour-Based Systems, Advanced Robotics, Special Issue on Selected Papers from IROS'96, 1997.
[3] R. C. Arkin, Cooperation without Communication: Multiagent Schema-Based Robot Navigation, Journal of Robotic Systems, 1992.
[4] M. Bernardine Dias and Anthony Stentz, TraderBots: A Market-Based Approach for Resource, Role, and Task Allocation in Multirobot Coordination, Technical report, CMU-RI, 2003.
[5] Brian P. Gerkey and Maja J. Mataric, Sold!: Auction Methods for Multirobot Coordination,IEEE Trans. on Robotics and Automation, Vol. 18, No. 5, 2002.
[6] 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.
[7] Martin J. Osborne and Ariel Rubinstein. A course in game theory, MIT Press, 1994.
[8] 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.
[9] Ariel Rubinstein, Perfect Equilibrium in a Bargaining Model, Econometrica, Vol. 50, No. 1, 1982.
[10] Soo, V-W and Wu, S-H., Negotiation Without Knowing Other Agents Payoffs in the Trusted Third- Party Mediated-Game, Second workshop on game theoretic and decision theoretic agents, 2000.
[11] J-H. Chen, K-M. Chao, N. Godwin, C. Reeves, P. Smith, An automated negotiation mechanism based on co-evolution and game theory, in Proc. of the 2002 ACM symposium on Applied, 2002.
[12] A.E. Eiben, J.E. Smith, Introduction to Evolutionary Computing. Springer, 2003.
[13] Theodore Caplow, Two Against One: Coalitions in Triads. Englewood Cliffs, S.J.: Prentice- Hall, 1968.
[14] O. Michel, Webots: Professional Mobile Robot Simulation. *J. Adv. Robotics Systems*,1(1):39-42, 2007.
[15] A. Ijspeert, A. Crespi, D. Ryczko, and J.M. Cabelguen. From swimming to walking with a salamander robot driven by a spinal cord model. *Science*, 315(5817):1416-1420, 2007.