A fleet of autonomous and cooperative mobile robots

Rachid Alami

A fleet of autonomous and cooperative mobile robots

Rachid Alami,
LAAS/CNRS
7, Avenue du colonel Roche
31077 Toulouse cedex 4 - France
e-mail: rachid@laas.fr

Abstract

We present and discuss a generic cooperative scheme for multi-robot cooperation. It is based on an incremental and distributed plan-merging process.

We discuss the properties of this cooperative scheme (coherence, detection of dead-lock situations) as well as the class of applications for which it is well suited. We also discuss how this paradigm "fills the gap" between centralized planning and distributed execution.

We show how this scheme can be used in a hierarchical manner, and in contexts where planning is performed in parallel with plan execution.

We finally illustrate it through an implemented system which allows a fleet of more than ten autonomous mobile robots to perform load transfer tasks in a route network environment with a very limited centralized activity and important gains in system flexibility and robustness to execution contingencies.

1 Introduction

In the field of multi-agent cooperation, besides goal/task decomposition and allocation to various agents, another key issue involves the simultaneous operation of several autonomous agents, each one seeking to achieve its own task or goal. This is particularly true for autonomous multi-robot applications and, more generally, when the allocated tasks or goals cannot be directly "executed" but require further refinement.

These two issues are different in nature, and should call for different resolution schemes. While the first issue is more oriented towards the collective search for a solution to a problem and calls for a purely deliberative activity, the second involves a more "compliant" behavior of the agents and integrates a closer interaction between deliberation and action.

While several generic approaches have been proposed in the literature concerning task or goal decomposition and allocation (Contract Nets [16], Partial Global Planning [7], distributed search [8], negotiation [10, 5, 6], motivational behaviors [13, 9]), cooperation for achieving independent goals have been mostly treated using task-specific or application-specific techniques [11, 12, 14]

We argue that there is also a need for generic approaches to the problem of the simultaneous operation of several autonomous agents, each one seeking to achieve its own task or goal. One can of course make the agents respect a set of rules (e.g. traffic rules), or more generally "social behaviors" [15], which are specially devised to avoid as much as possible conflicts and to provide pre-defined solutions to various situations. However, this cannot be a general answer applicable to various domains.

We would like to devise a scheme which guarantees a coherent behavior of the agents in all situations (including the avalanche of situations which may occur after an execution failure) and a reliable detection of situations which call for a new task distribution process.

In the following, we propose a paradigm [3, 4] which, we believe, fulfills such requirements.

2 The *Plan-Merging* Paradigm

Let us assume that we have a set of autonomous robots equipped with a reliable inter-robot communication device which allows to broadcast a message to all robots or to send a message to a given robot.

Let us assume that each robot processes sequentially the goals it receives, taking as initial state the final state of its current plan. Doing so, it incrementally appends new sequences of actions to its current plan.

However, before executing any action, a robot has to ensure that it is valid in the current multi-robot context, i.e. that it is compatible with all the plans currently under execution by the other robots. This will be done by collecting all the other robot plans and by "merging" its own plan with them. This operation is "protected" by a mutual exclusion mechanism and is performed *without modifying* the other robots plans or inserting an action which may render one them invalid in order to allow the other robots to continue execution.

We call this operation, a *Plan-Merging Operation* (PMO) and its result a *Coordination Plan* (i.e. a plan valid in the current multi-robot context). Such a *coordination plan* consists of a sequence of actions and events to be signaled to other robots as well as events which are planned to be signaled by other robots. Such

events correspond to state changes in the multi-robot context and represent temporal constraints (precedence) between actions involved in different individual coordination plans.

Note that such an operation involves only communication and computation and concerns future (near term) robot actions. It can run in parallel with the execution of the current coordination plan.

Everything works as if there was a *global plan* produced and maintained by the set of robots. In fact, no robot elaborates, stores or maintains a complete representation of such a *global plan*. At any moment, the "global plan" is the graph representing the union of all current robot coordination plans. Such a global plan is valid (i.e. it does not contain inconsistent temporal constraint) if it can be represented by a *directed acyclic graph* (*dag*).

The key point here is how to devise a system composed of a set of robots which should, as much as possible, plan independently to achieve their tasks while maintaining such property of the global plan.

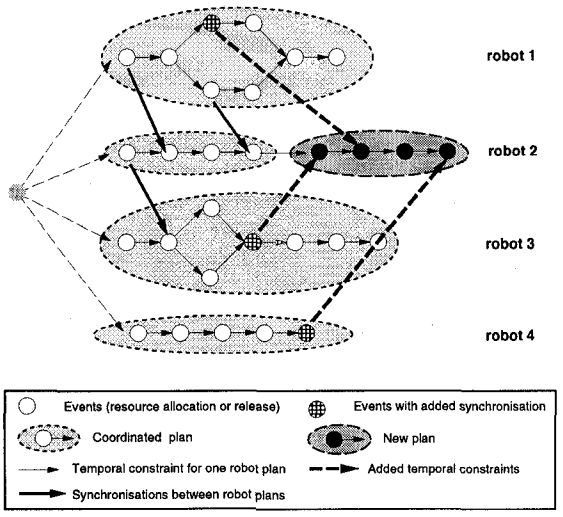

Figure 1: Part of the "global plan" *dag* during the insertion of a new plan by robot 2.

2.1 Situations where PMO is deferred or where deadlock is detected

When a robot tries to perform a PMO, it may fail to produce a plan which satisfies the properties discussed earlier.

This may happen in two situations:

1. the goal can never be achieved. This can be detected if the robot cannot produce a plan even if it was alone in the environment.
2. the robot can generate a plan but this plan cannot be inserted in the global plan. This means that the final state of another robot forbids it to insert its own plan.

   In such situation, the robot can simply abandon the PMO and decide to wait until the robots, that it has identified, have performed a new PMO which may possibly make them change the states preventing it to insert its plan.

Hence, we have introduced two types of events:
1 - *execution events:* i.e. events which occur during plan execution and which allow robots to synchronize their execution.
2 - *planning events:* i.e. events which occur whenever a robot performs a new PMO. These events can also be awaited for.

Note that, even when a robot fails in its PMO, it leaves the global plan in a correct state (it is still a *dag* and its execution can continue).

In order to detect deadlocks, a robot which finds itself in a situation where it has to wait for a *planning event* from a particular robot, must inform it. Then, it becomes possible for a robot to monitor and detect *deadlock situations* by propagating and updating, a graph of robots waiting (directly or by transitivity) for *planning events* from itself. Indeed, a deadlock is detected when a robot finds itself in the list of robots waiting for itself.

When a deadlock occurs, it is necessary to take explicitly into account, in a *unique planning operation*, a conjunction of goals (which have been given separately to several robots).

This simply means that the global mission was too constrained to be solved using the Plan-Merging Paradigm. Here we must recall that we do not claim that the Plan-Merging paradigm can solve or help to solve multi-robot planning problems. The main point here is that the Plan-Merging paradigm is *safe* as it includes the detection of the deadlocks i.e. situations where a a new task decomposition should take place.

Note also that, in the case where only a small number of robots are involved in a deadlock, one can decide to allow the robot, which detected the deadlock, to plan for all the concerned robots . The Plan-Merging paradigm remains then applicable: the inserted plan will then concern several robots at a time.

The paradigm and the protocol presented so far is generic. We believe that it can be used in numerous applications. Several instances of the general paradigm can be derived, based on different planners: action planners in the stream of STRIPS, as well as more specific task planners or motion planners. We present in the sequel an application in the case of a fleet of autonomous mobile robots.

3 A fleet of autonomous mobile robots

We have applied the Plan-Merging Paradigm in the framework of the MARTHA project[1] which deals

[1]MARTHA: European ESPRIT Project No 6668. "Multiple Autonomous Robots for Transport and Handling Applications"

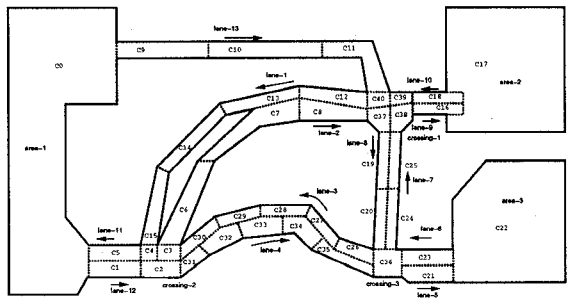

Figure 2: An Environment Model.

with the control of a large fleet of autonomous mobile robots for the transportation of containers in harbors, airports and railway environments.

In such context, the dynamics of the environment, the impossibility to correctly estimate the duration of actions (the robots may be slowed down due to obstacle avoidance, and delays in load and un-load operations, etc..) prevent a central system from elaborating efficient and reliable detailed robot plans.

The use of the Plan-Merging paradigm allowed us to deal with several types of conflicts in a general and systematic way, and to limit the role of the central system to the assignment of tasks and routes to the robots (without specifying any trajectory or any synchronization between robots) taking only into account global traffic constraints.

3.1 Overall Architecture

A Martha system is composed of a Central Station (CS) and a set of autonomous mobile robots able to communicate with each other and with the Central Station.

The CS as well as the robots make use of the same description of the environment for several purposes dealing with mission specification, robot navigation or multi-robot conflict resolution.

Environment Model An environment is a topological graph of *areas*, *routes* and *crossings*. The areas contains docking/undocking *stations*. The routes are composed of *lanes*; crossing and lanes are then composed of *cells* which have a nominal (but not exclusive) direction. Cells, areas and stations have a geometrical description (polygonal regions). An example of such an environment is given in Figure 2.

Besides, one can have a geometrical description of known obstacles as well as complementary data for localization or docking purposes.

The robots heavily rely on this model to plan their routes and trajectories. Nevertheless, the real environment may also contain unknown obstacles/objects which have to be dealt with on-line by the robots (detection and avoidance if possible).

Martha's Robots Missions Although one of the goal of the Martha project is to alleviate the burden on a Central Station (CS), one remains present. However, its role is mainly to plan the transshipment operations (which robot loads/unloads which container)[2] and the routes the robot *should* use. The CS uses the topological model to plan these routes. The CS does not intervene in the robot plans coordination (such as in crossing or area), nor does it plan the precise trajectory which are executed by the robots. As a consequence, the communication bandwidth required between the robots and the CS is very low. Moreover, the computational power devoted by the CS to control the robot is far less important than the one used in a completely centralized application.

The robots receive their missions from the Central Station. From then on, each robot is on its own to perform the mission. It has to refine the mission, to plan its routes and then its trajectories, to coordinate the resulting plans and trajectories with other robots and to execute all these actions, monitoring critical situations (such as unknown obstacles) and reporting unrecoverable action failure to the CS (mostly those requiring an operator assistance).

3.2 A Plan-Merging Protocol for Multi-Robot Navigation

For the case of a number of mobile robots in a route network environment, we have devised a specific *Plan-Merging Protocol* (PMO) based on spatial resource allocation (see [4]). It is an instance of the general protocol described above, but in this context, *Plan-Merging Operation* is done for a limited list of required spatial resources: a set of cells which will be traversed during the plan to merge. The robot broadcasts the set or required cells, receives back the set of coordination plans from other robots which have already planned to use some of the mentioned cells, and then tries to perform a plan insertion which ensures that the union of the considered plans is a directed acyclic graph.

One of the most interesting attributes of this protocol is that it allows several PMOs to be performed simultaneously if they involve disjunctive resource sets. This is particularly useful when there are several local conflicts at the same time.

Plan-Merging for cell occupation:

In most situations, robot navigation and the associated Plan-merging procedure are performed by trying to maintain each cell of the environment occupied by at most one robot. This allows the robots to plan their trajectories independently, to compute the set of cells they will cross and to perform Plan-Merging at cell allocation level.

[2]The transshipment operations planning problem, which remains under the responsibility of the CS is more or less a temporal allocation problem and is not presented in this paper.

In order not to constrain unnecessarily the other robots, the allocation strategy makes a robot allocate one cell ahead when it moves along lanes, while it allocates all the cells necessary to traverse and leave crossings.

When reasoning about cells is not sufficient

While, most of the time, the robots may restrict their cooperation to cells allocation, there are situations where this is not enough. This happens when they have to cross large (non-structured) areas or when an unexpected obstacle, encountered in a lane or in a crossing, forces a set of robots to maneuver simultaneously in a set of cells. In such situations, a more detailed cooperation (using the same protocol but a different planner: the motion planner) takes place allowing robots to coordinate their actions at trajectory level.

Thus, we have a hierarchy of PMOs:

- first, at the cell level, based on resource (cells) allocation
- then, depending on the context, at trajectory level: motion planning in a set of common cells determined by the first level

This hierarchy authorizes a "light" cooperation, when possible, and a more detailed one, when necessary.

3.3 Examples

We shall now illustrate the plan-merging paradigm and its capabilities with some sequences from our experimentation with simulated robots.

Example 1: Coordination at cell level

Step 1: Figure 3 shows the involved cells of the environment.
The robot destinations are the followings:

- Robots 0 and 1 on the right go to cell $C8$ above the crossing using cell $C4$.
- Robots 2 and 3 at the bottom right traverse the crossing to reach the left cell $C0$ using cells $C5, C4$ and $C2$.
- Robot 6 goes from left to the right cell $C7$ using cells $C3$ and $C5$.
- Robot 4 goes from up to the lower cell $C10$ using cells $C2$ and $C3$.

The PMOs have occurred in the following order: robot-0 then robot-2 and then robot-6 in parallel with robot-1 (because robot-6 and robot-1 have disjunctive lists of resources) and finally robot-4.
Step 2: The following synchronizations have been planned: robot-2 on robot-0 (which frees $C4$), robot-6 on robot-2 (which frees $C5$) and robot-1 on robot-2 (which frees $C4$), robot-4 on robot-2 (which frees c2) and robot-6 (which frees $C3$)
Step 3: One should note that at this stage, robot-3 PMO fails and is deferred because robot-2 has not yet planned an action to free the cell $C0$.

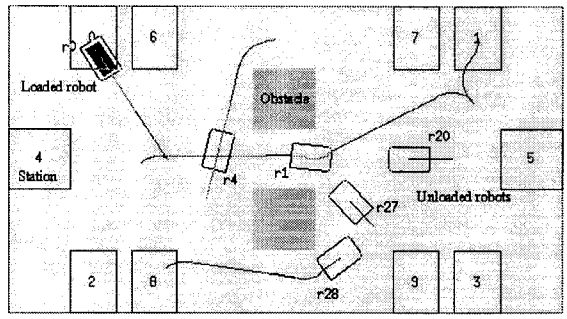

Figure 4: Plan-merging at the trajectory level.

Example 2: Trajectory level

The second example illustrates PMO at trajectories level in a large open area with two obstacles in the middle, and 10 docking/undocking stations. In such an environment, there are no cell allocations (the robots are all in the same cell), all synchronizations are made at trajectory level.

Figure 4 shows a situation where all the robots have planned and coordinated a complete trajectory. The trajectories displayed on the figure are the one which have been sent by the robots for execution display.
• The robot destinations are: $r0$ goes to station 9, $r4$ to station 5, $r1$ to station 1, $r28$ to station 7, $r27$ to station 0, $r20$ to station 4.
• PMOs were done in the following order: $r1$, $r4$, $r27$, $r20$, $r28$ and $r0$.
• One can see the synchronizations established by the robots: $r4$ on $\{r1\}$, $r27$ on $\{r1, r4\}$, $r20$ on $\{r1, r27, r4\}$, $r28$ on $\{r27, r20, r1\}$, $r0$ on $\{r1, r4, r28, r27\}$.

3.4 Implementation and Results

We have developed a complete robot control system which includes all the features described. Its architecture is based on a generic control architecture for autonomous mobile robots developed at LAAS [2]. It is instantiated in this case by adding an intermediate layer for performing Plan-Merging operations.

Emulation Testbed

For testing and demonstration purposes, the robot control system has been linked to a robot simulator.

Experiments have been run successfully on a dozen of workstations (each workstation running a complete robot simulator) communicating through Ethernet. A 3-d graphic server has been built in order to visualize the motions and the load operations performed by all the robots in a route network (Figure 3) or in-door (Figure 4) environments. The simulated robots where

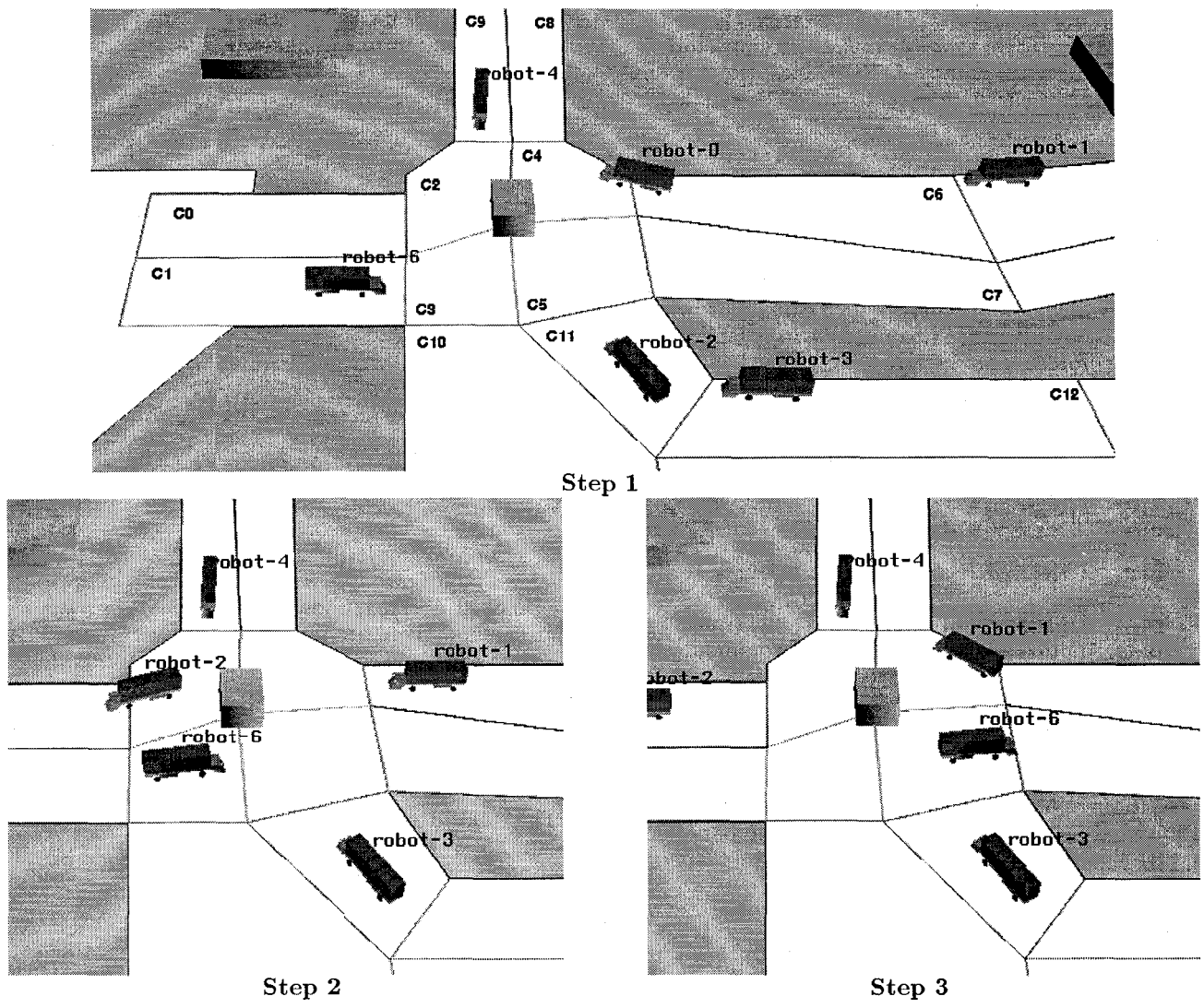

Figure 3: Plan-merging at the cell level.

able to achieve navigation missions, performing hundreds of PMOs and solving local conflicts. Motion planning and PMOs were sufficiently efficient to allow most often the robots to elaborate and merge their plans without stopping unless necessary.

Running a fleet of ten robots during thirty minutes on a large outdoor environment slightly more complex than the one presented on Figure 2 generates about 4024 messages between the robots which can be classified into 416 PMO requests, resulting into 3744 responses. 128 cooperation plans are exchanged resulting into 48 synchronizations between executable plans. 32 wait for planning are generated. 15 K-bytes of compressed data are exchanged over the robot network.

Another example also involving ten robots for the same duration, in a more constrained indoor environment resulted in 10168 messages exchanged (note the increase of the number of messages), including 928 PMO requests which led to 8352 responses. Due to the small number of cells, and large areas, 220 PMO request conflicts arose. 936 cooperation plans were exchanged, 176 cell synchronizations and 104 trajectories synchronizations were done. 219 plan dependencies were managed and led to 439 plan update messages. 100 K-bytes of compressed data were exchanged over the robot network.

Real Robots Testbed

Extensive experiments have also been performed using three laboratory robots (Figure 5).

Our experiment room (which is about 10×7 meters large) has been structured into two areas including six docking stations and two lanes, according to the environment model presented in Section 3.1. In

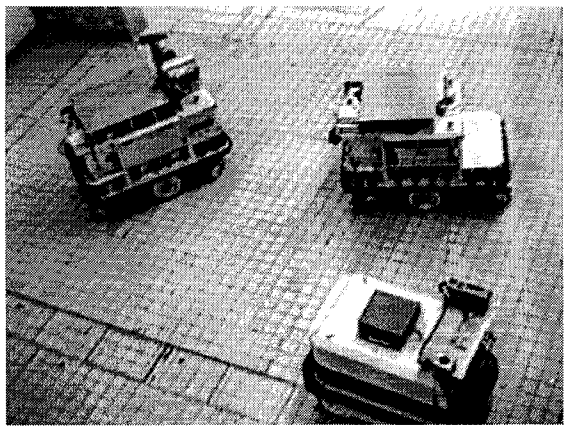

Figure 5: The Three Hilare Robots in Mission

this environment, we have conducted runs where the robots keep going for more than two hours. In a typical run, during one hour, one robot: covers a cumulated distance of 300 meters, exchanges 900 messages with the other robots, and executes 250 coordination operations which yield to 70 synchronizations at the trajectory level and 20 at the resource level.

The high number of coordinations observed here is a consequence of the small size of the environment compared to the size of our robots. But it fully demonstrates the capabilities of our decisional level.

4 Conclusion

We have argued that, in the field of multi-robot (and more generally multi-agent) cooperation, it is useful to distinguish between two main issues: **C1** goal/task decomposition and allocation, and **C2** cooperation while seeking to achieve loosely coupled goals. We have then proposed a "generic" approach called *Plan-Merging Paradigm* which deals with **C2** issues and clearly establishes a link with **C1** issues.

We believe that it can be applied to a large variety of contexts and with different planners (from action planners to task or motion planners), and at different granularities.

Such a multi-robot cooperation scheme "fills the gap" between very high level planning (be it centralized or distributed) and distributed execution by a set of autonomous robots in a dynamic environment.

Indeed, it appears to be particularly well suited to the control of a large number of robots navigating in a route network. The application that we have implemented clearly exhibits its main features. It allowed us to make a large number of autonomous robots behave coherently and efficiently without creating a huge activity at the central system.

References

[1] L. Aguilar, R. Alami, S. Fleury, M. Herrb, F. Ingrand, F. Robert *Ten Autonomous Mobile Robots (and even more) in a Route Network Like Environment* In, *IROS'95*, Pittsburgh (USA).

[2] R. Alami, R. Chatila, and B. Espiau. Designing an Intelligent Control Architecture for Autonomous Robots. in *ICAR'93*, Tokyo (Japan).

[3] R. Alami, F. Robert, F. Ingrand, and S. Suzuki. A paradigm for plan-merging and its use for multi-robot cooperation. In *IEEE Int. Conf. on Systems, Man, and Cybernetics*, San Antonio, Texas (USA), October 1994.

[4] R. Alami, F. Robert, F. Ingrand, S. Suzuki. Multi-robot Cooperation through Incremental Plan-Merging In *IEEE Int. Conf. on Robotics and Automation*, Nagoya (Japan), May 1995.

[5] H. Asama, K. Ozaki, et al. Negotiation between multiple mobile robots and an environment manager. In *ICAR'91*, Pisa (Italy), June 1991.

[6] R. I. Brafman, Y. Shoham. Knowledge considerations in robotics and distribution of robotics tasks. *IJCAI'95*, Montréal (Canada), Aug. 1995.

[7] E.H. Durfee, V. Lesser Partial Global Planning: A Coordination Framework for Distributed Hypothesis Formation In *IEEE Transactions on Systems, Man and Cybernetics*, Vol 21 (5), Oct. 1991.

[8] E.H. Durfee, T. A. Montgomery Coordination as Distributed Search in a Hierarchical Behavior Space In *IEEE Transactions on Systems, Man and Cybernetics*, Vol 21 (6), December 1991.

[9] E. Ephrati, M. Perry, J.S. Rosenschein. Plan execution motivation in multi-agent systems. *AIPS'94*, Chicago, June 1994.

[10] N.R. Jennings. Controlling cooperative problem solving in industrial multi-agent systems using joint intentions *Artificial Intelligence*, Vol 73 (1995) 195-240.

[11] C. Le Pape. A combination of centralized and distributed methods for multi-agent planning and scheduling. *IEEE Int. Conf. on Robotics and Automation*, Cincinnati (USA), May 1990.

[12] K. Ozaki, H. Asama, et al. Synchronized motion by multiple mobile robots using communication. *IROS'93*, Yokohama (Japan), July 1993.

[13] L.E. Parker. Heterogeneous multi-robot cooperation. *MIT Technical Report AITR-1465*, Feb. 994.

[14] S. Yuta, S. Premvuti. Coordination autonomous and centralized decision making to achieve cooperative behaviors between multiple mobile robots. *IROS'92*, Raleigh (USA), July 1992.

[15] Y. Shoham, M. Tennenholtz. On social laws for artificial societies: Off-Line design. *Artificial Intelligence, 73 (1995) 231-252*

[16] R.G. Smith. The Contract Net Protocol: High-level communication and control in a distributed problem solver. *IEEE Transactions on Computers*, Vol C-29 (12), December 1980.

Acknowledgments: This work is the result of an intensive collaboration between: F. Robert, F. Ingrand, S. Fleury, M. Herrb, S. Qutub, L. Aguilar. I would like also to thank M. Khatib, H. Bullata, S. Suzuki, J. Perret, T. Siméon, B. Dacre-Wright, M. Devy.