Operating a Large Fleet of Mobile Robots using the Plan-Merging Paradigm *

R. Alami, S. Fleury, M. Herrb, F. Ingrand, S. Qutub †
LAAS-CNRS
7, Avenue du Colonel Roche, 31077 Toulouse CEDEX 04
E-mail: {rachid,sara,matthieu,felix,sam}@laas.fr

Abstract

We present and discuss the use of a generic scheme for multi-robot cooperation called the "Plan-Merging Paradigm" for managing a large fleet of autonomous mobile robots.

Each robot, autonomously and incrementally builds and executes its own plans taking into account the multi-robot context obtained by collecting the current plans and goals of the other robots.

We describe the overall system architecture and discuss the properties of our cooperative scheme. We show how the Plan-Merging Paradigm (PMP) can be used in a hierarchical manner and how it "fills the gap" between centralized planning and distributed execution.

We finally illustrate this scheme through an implemented system which allows a fleet of autonomous mobile robots to perform load transfer tasks in a route network environment. The central activity is limited to task allocation and important gains are obtained in system flexibility and robustness to execution contingencies. Simulations (using up to 30 robots) as well as experiments with real robots (3) are presented and discussed.

1 Introduction

In the field of multi-agent cooperation, we distinguish two key issues which are different in nature, and which should call for different resolution schemes. One is the "classical" (although still not solved in the general case) goal/task decomposition and allocation to various agents. The second addresses the problem of the simultaneous operation of several autonomous agents, each one seeking to achieve its own task or goal.

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.

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 based on the information acquired on-line through the perception and the communication means.

While several generic approaches have been proposed in the literature concerning task or goal decomposition and allocation (Contract Nets [Smith, 1980], Partial Global Planning [Durfee and Lesser, 1991], distributed search [Durfee and Montgomery, 1991], negotiation [Jennings, 1995; Asama *et al.*, 1991], motivational behaviors [Parker, 1994; Ephrati *et al.*, 1994]), cooperation for achieving independent goals have been mostly treated using task-specific or application-specific techniques [Le Pape, 1990; Yuta and S.Premvuti, 1992] or more generally "social behaviors" [Shoham and Tennenholtz, 1995], which are specially devised to avoid as much as possible conflicts and to provide pre-defined solutions to various situations.

We believe that the *Plan-merging Paradigm* which we have proposed in earlier papers [Alami *et al.*, 1994; 1995] provides a general framework for the simultaneous operation of several autonomous agents, each one seeking to achieve its own task or goal.

We report here on the design and implementation of a general multi mobile robot system for load transfer based on this cooperative scheme.

In section §2 we present the overall system architecture. In section §3 we discuss the main features of the plan-merging paradigm. We then discuss the requirements imposed by the use of such a paradigm on the decisional and functional components of the robots and how it can be plugged into existing robot systems (section §4). Section §5 provides some results we have recently obtained in simulation as well as on a set of three mobile robots.

*This paper has been published in the 1997 IEEE International Conference on Robotics and Automation, Albuquerque, New Mexico (USA)

†Authors list in alphabetical order.

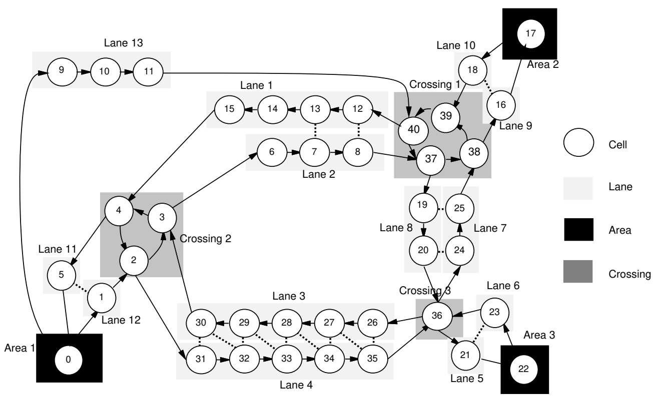

Figure 1: An Environment: Topological Representation.

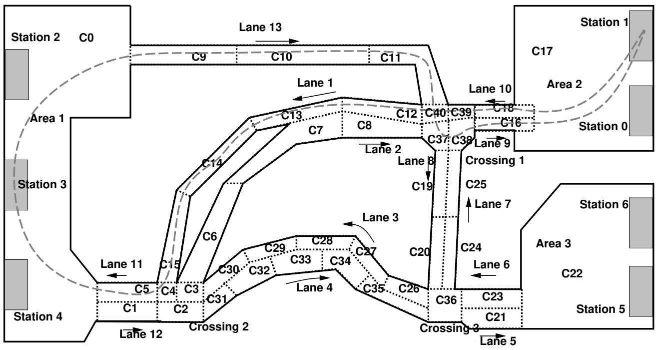

Figure 2: Geometrical Representation.

2 System Description

The complete 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 and the robots use of the same description of the environment for mission specification, robot navigation or multi robot conflict resolution.

Environment Model. This model has been designed to enable the implementation of the multi robot cooperation scheme presented in this paper, and as such is very much linked to this approach. The model is a topological and geometrical representation of the environment.

An environment is a topological graph (Figures 1) 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) (Figure 2).

Besides, one can have a geometrical description of known obstacles as well as complementary data for localization or docking purposes. The real environment may also contain unknown obstacles/objects which have to be taken into account on-line by the robots (detection and avoidance if possible).

```
; Starting from station 1.
(mission (.
   (action 1 (goto (station 3))
             (using (lane 10) (lane 1) (lane 11)))
   (action 2 (dock))
   (action 3 (pick-up (container 5)))
   (action 4 (undock))
   (action 5 (goto (station 1))
             (using (lane 13) (lane 9)))
   (action 6 (dock))
   (action 7 (putdown))
   (action 8 (undock)) .))
```

Figure 3: Example of a robot mission.

Robots Missions. Although one of the goal of the project is to alleviate the burden on a Central Station (CS), one remains present. However, its role is *only* to plan the transshipment operations (which robot loads/unloads which container)[1] and the routes the robot *should* use (See Figure 3 and 2 for a mission example). 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.

Robot Capabilities. The robots receive their missions from the Central Station. Each Robot Control System (RCS, on-board the robot) is then 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 (based on the PMP paradigm) 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 PMP Ideas and Principles

Planning and plan coordination can be classified along different strategies or choices.

Global versus local. When one plans actions for a fleet of robots, one can consider the whole fleet or limit the scope of planning to the sets of robots with conflicting actions. However, this global versus local tradeoff is only possible when dealing with a properly sized environment. If the number of critical exclusive resources is more or less equal to the number of robots, conflict resolution may, by propagation, involve the whole fleet.

[1]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 (see [T.Vidal *et al.*, 1996]).

On the other hand, if the environment is properly sized, conflicts remain local, and the solutions are negotiated locally without disturbing the unconcerned robots.

Complete versus incremental. Similarly, one can limit the scope of the planning and plan coordination in time. When a mission is sent to a robot, it can plan (or try to plan) and coordinate the whole mission. But considering the execution hazards, it seems to be inefficient (not to say a waste of time and resource) to plan too far ahead. The plan coordination should be done continuously, to guarantee a fluid navigation, and slightly ahead, to avoid to over constrain the others robot plans and to break the coordinated plans too often.

Centralized versus distributed. This last aspect of the planning and plan coordination problem is where should it take place: on a centralized computer or on board the robots? This does not change the computing complexity of the treatment itself. However, in a centralized approach, all the data (which are mostly local) need to be sent to the central station, and therefore require a more reliable communication link with a higher bandwidth between the robot and the central station. Moreover, the proposed protocol can be implemented with a local communication between the robots.

Our approach can be classified as *local*, *incremental* and *distributed.* Missions are sent to each robot which plans and coordinates on-board *(distributed processing)*, up to a small resources horizon *(incremental)*, using a distributed coordination approach involving only the robots which are concerned by the required resources *(local)*.

This is performed by an operation called *Plan-Merging Operation* (PMO), which consists in "merging" the desired plan into the set of all collected ones. This operation is performed under a mutual exclusion mechanism and does not affect the other robots actions.

The result of a successful PMO is called a *Coordinated Plan*[2]. Such a plan specifies the sequence of actions and the necessary synchronization between these actions and the other robots coordinated plans (Figure 4).

When applied to the multi-robot navigation in an environment like the one described above, the Plan-Merging is done for a limited list of spatial resources: the set of cells which will be traversed during the plan to merge. This allows the robots to plan their trajectories independently and to apply various allocation strategies depending on the execution context.

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.

While, most of the time, the robots may restrict their cooperation to cells occupation plans merging, there are situations where this is not enough. This happens when they have to cross large (non-structured) areas or when an unexpected obstacle forces a set of robots to maneuver simultaneously in a set of cells. In such situations, a more detailed cooperation (using the same paradigm but a different planner: the motion planner) takes place allowing robots to coordinate their actions at trajectory level (Figure 5).

[2]To limit the size of the paper, we do not present the PMO here. A detailed discussion on the properties of the Plan-merging paradigm as well as on its ability to cope with execution failures can be found in [Alami *et al.*, 1994; 1997; Alami, 1996].

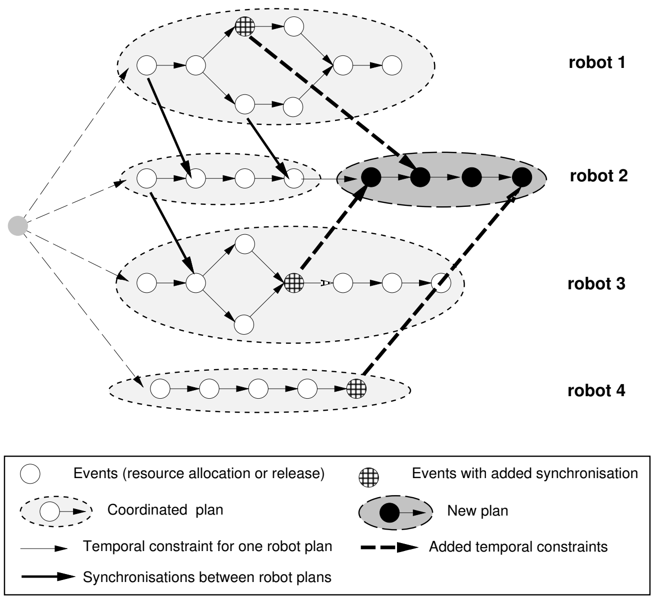

Figure 4: Robot 2 has merged a new plan.

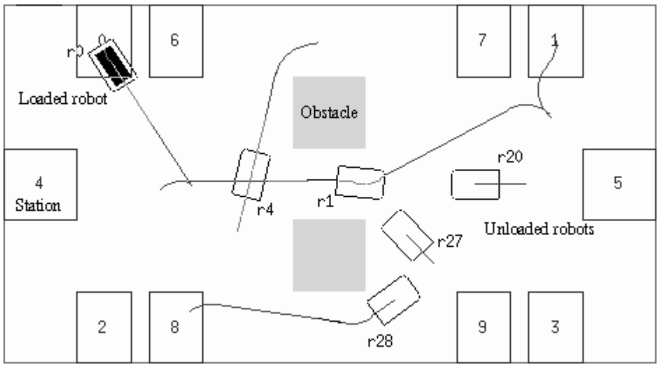

Figure 5: **Plan-merging at trajectory level.** *This example illustrates PMO at trajectories level in a large open area with two obstacles in the middle, and 10 docking/undocking stations. In such an environment, there are no cell allocations (the robots are all in the same cell), all synchronizations are made at trajectory level.*

4 Adding PMP to Existing Robotics Systems

One of the main goal of our approach was to design a paradigm which can be used in a variety of planning/plan merging situations. We already used it at two different levels in our multi robots navigation application (at cells allocation level and at trajectories coordination level). In this section we shall define and identify the requirements which allow to use the PMP paradigm.

4.1 Planning Requirements

If one considers a plan as a sequence (or a partially ordered graph) of actions, the use of PMO will impose the following requirements on the plan representation.

Plan interaction: temporal dependencies. The various plans produced by the different agents must interact in such a way, that all the possible conflicts can be solved by synchronizing part of their plan. In other words, for any two valid plans, there exists a sequence, which at most add synchronization points between actions, which make the two plans feasible.

Merging/coordinating module. For any plan to be merged in a set of already coordinated plans, we need a function which will produce the resulting plan by adding synchronization actions in the plan to merge associated to events generation in the other coordinated plans (Figure 4).

Plan range or horizon: can be limited. The plan produced by the various agents must be such that one can merge and combine a sub part of it. This is not really a requirement, but merely a "good practice" wish. Indeed, if an agent commit a rather large and long part of its future plan, it will certainly over-constrain other agents plans.

4.2 Functional Requirements

From a functional point of view, there are a number of requirements which are needed to implement a PMO.

Perfect Communications. A perfect communication media is required to implement the PMO. On top of this communication, a mutual exclusion mechanism (for example using tokens) must be available to guarantee that the plan merging is appropriately done. Note that the mutual exclusion can be total or limited to a number of resources (which is more efficient). The communication channel is also used to exchange the plans and the synchronization events.

Execution Monitoring. While executing its plan, the robot must be able to recognize the synchronization events which it has been asked to inform its occurrence to another agent.

4.3 PMO module

Providing the aforementioned requirements, we believe that the PMO module we have developed can be extracted and used on a variety of systems to solve a number of conflicts for loosely coupled robot missions.

This PMO module would be called whenever a plan has to be merged with others collected plans, and would either return a valid result (i.e. a set of synchronization actions and events to be generated) or the reason for the failure (i.e. the plan cannot be totally merged in the collected plan, or the overall produced plans are globally invalid).

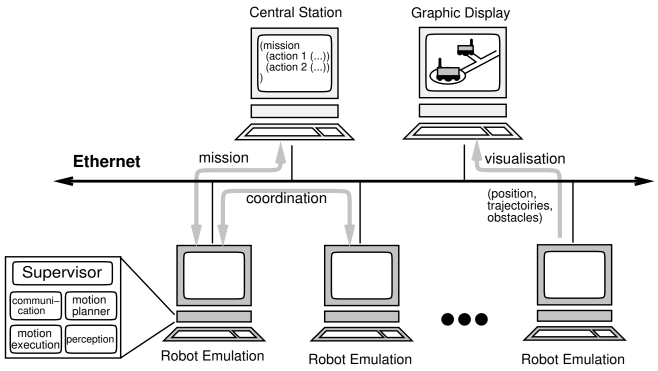

Figure 6: n Unix workstations implementing n robots

One should note that the plan merging operation itself is not very expensive, and does not hold the mutual exclusion too long. [3] However, the plans collecting phase (which we also perform under mutual exclusion) can be rather long if some robots take some time to respond. Nevertheless, if the mutual exclusion token is limited to the resources used by the plan to merge (instead of being global over the whole fleet), then the complete PMO is usually done in few seconds.

5 Experimental Results

We have developed a complete robot control system based on a generic control architecture for autonomous mobile robots developed at LAAS [Fleury *et al.*, 1994; Ingrand *et al.*, 1996]. It is instantiated in this case by adding an intermediate layer for performing Plan-Merging operations.

All software components were developed and run under Unix *and* the real-time operating system VxWorks. The demonstration on a large number of emulated robots under Unix was necessary to validate our approach, while using our three mobile robots running VxWorks demonstrates the effectiveness and the ability to run the whole system on board real robots.

5.1 Emulation Testbed

To validate the RCS architecture and the PMO paradigm, we developed an emulation testbed (see Figure 6) which includes an emulation of the Central Station, a display tool to visualize the evolution of the robots and the n emulated robots.

The display tool (see Figures 7 and 5 (2D version)) is basically a display server on which each robot is connected and updates its position at a high rate. This server has a model of the environment and shows in real

[3] The insertion of a sequence of s plan steps into a coordination plan composed of n steps is proportional to $s * n$.

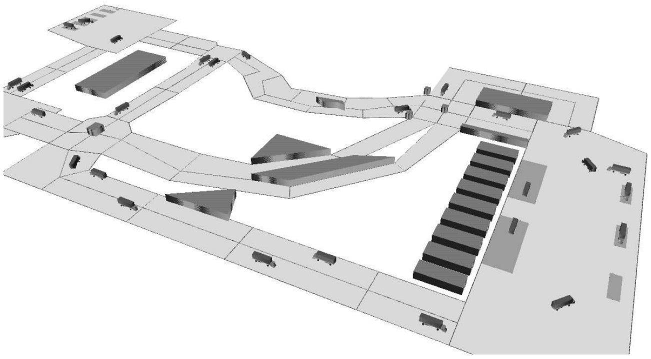

Figure 7: Simulation of a fleet of 30 robots

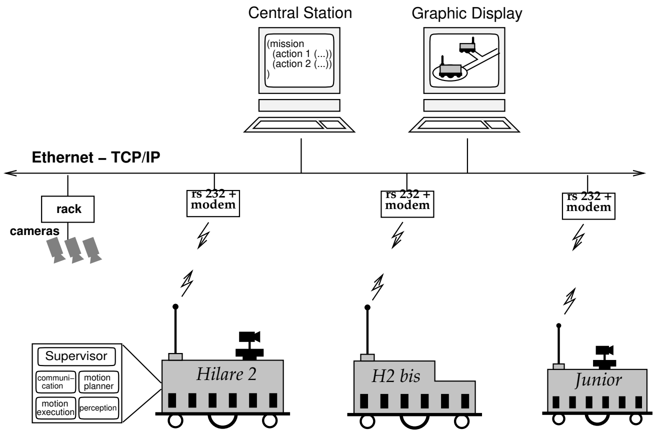

Figure 8: The Experimental Testbed

Figure 9: The Three Hilare Robots in Mission

time the evolution of the fleet. Runs can be recorded and replayed at a later time for a finer analysis.

Other tools record the various communications taking place between the robots and allows us to make statistical analysis on the number/size of messages exchanged and to evaluate the minimum required communication bandwidth.

To thoroughly test our approach, we have build approximatively fifteen different environments with a great variety of features: indoor versus outdoor; small (10-100 meters) versus large environment (100-500 meters); small (1-2 meters) robots versus large (17 meters) container carriers; large open areas versus complex graphs of areas/lanes/crossing; environments with and without unknown obstacles, and so on.

Overall, the emulation testbed was very useful as it allowed us to:
- make hundreds of hours of near real experimentation,
- run a much larger fleet of robots than the one we currently have...
- test and debug the RCS, the robot functional modules and the PM paradigm,
- test and tune the environment description choices (size of areas, number of cells in crossing, and so on),
- validate the size of a fleet for a given environment.

5.2 Some numerical results

Running a fleet of 30 robots during one hour on a large outdoor environment (Figure 7) generated 59971 messages between the robots which can be classified into 47644 messages induced by the distributed mutual exclusion protocol [Naimi *et al.*, 1996], 515 PMO request, resulting into 11383 responses. 462 cooperation plans were exchanged resulting into 152 synchronizations between executable plans. 75 wait for planning are generated. 300 K-bytes of data are exchanged over the robot network ($<$ 100 bytes per second).

Another example involving ten robots for thirty minutes, in a more constrained indoor environment resulted in 10168 messages exchanged, including 928 PMO requests (note the large number of PMO due to the more constrained environment) which led to 8352 responses. Due to the small number of cells, and large areas, 220 PMO request conflicts arose. 936 cooperations plan were exchanged, 176 cell synchronization and 104 trajectories synchronization were done. 219 plan dependencies were managed and led to 439 plan update messages. 100 K-bytes of data were exchanged over the robot network ($<$ 60 bytes per second).

5.3 Real Robots Testbed

Extensive experiments have also been performed using three laboratory robots (Figure 9).

The Hilare robots are equipped with two driving wheels, four free wheels, a VME rack supporting CPU boards of the Motorola 680x0 family, running under the VxWorks real-time system. The sensors used on each robot in this experiment are: an odometer and a gyroscope maintaining its position, a laser range finder used for absolute localization and obstacle modeling. In addition, one of the robots is equipped with a belt of sonars for obstacle detection. The problem of cooperation between sonars on several robots was not approached in this experiment. All robots use radio modems to communicate with the CS and with the other robots. A set of external cameras attached to the ceiling of the room completes the set of sensors. They are used to provide an absolute localization of the robots.

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 2.

In 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,
- executes 250 coordination operations which yield to 70 synchronizations at the trajectory level and 20 at the resource level.
- the decisional level produces 1500 requests to the functional 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.

6 Conclusion

We have applied the Plan-Merging Paradigm to the control of a large fleet of autonomous mobile robots which perform load transfer tasks.

In such a 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.

The system has been completely implemented. The extensive tests in simulation (running up to 30 robots) and with real robots (3) convinced us of the validity of the concept, its efficiency and its ability to be put onboard the robots.

Works is still ongoing in this project, in particular, we are studying extensions of the PMP to minimize the waiting time, in particular in trajectory coordination. Such extensions could be implemented by:
• taking the time explicitly into account, so that when plans are being coordinated, the robot which is doing the PMO does not always insert its action *after* the other robots conflicting actions, but possibly *before*.
• or relying on a bounded response time layer which could then be used to locally perform a first arrive first cross trajectory.

Other improvements of the paradigm are being considered and currently implemented (convoy mode, better robustness, dynamic add/remove of robots in the protocol) which will improve the overall performance of the system.

Besides the investigation of other classes of applications and the work on a more formal description of the proposed approach, our future work will also concentrate on developing new cooperation schemes by embedding a multi robot planning activity.

References

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

[Alami *et al.*, 1995] R. Alami, F. Robert, F. F. Ingrand, and S. Suzuki. Multi-robot cooperation through incremental plan-merging. In *IEEE ICRA*, 1995.

[Alami *et al.*, 1997] R. Alami, S. Fleury, M. Herrb, F. Ingrand, and F. Robert. Multi Robot Cooperation in the Martha Project. *IEEE Robotics and Automation Magazine, Robotics and Automation in Europe : Projects funded by the Commission of the European Union*, ?(?), 1997. To be published in 1997, also available as LAAS/CNRS TN 96-392.

[Alami, 1996] R. Alami. A multi-robot cooperation scheme based on incremental plan-merging. In *R. Hirzinger and G. Giralt (Eds), Robotics Research : The Seventh International Symposium, Springer Verlag*, 1996.

[Asama *et al.*, 1991] H. Asama, K. Ozaki, et al. Negotiation between multiple mobile robots and an environment manager. In *ICAR*, pages 533–538, June 1991.

[Durfee and Lesser, 1991] E.H. Durfee and V. Lesser. Partial global planning: A coordination framework for distributed hypothesis formation. In *IEEE Transactions on Systems, Man and Cybernetics, Vol 21 (5)*, October 1991.

[Durfee and Montgomery, 1991] E.H. Durfee and 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.

[Ephrati *et al.*, 1994] E. Ephrati, M. Perry, and J.S. Rosenschein. Plan execution motivation in multi-agent system. In *AIPS'94, Chicago*, June 1994.

[Fleury *et al.*, 1994] S. Fleury, M. Herrb, and R. Chatila. Design of a modular architecture for autonomous robot. In *IEEE ICRA*, 1994.

[Ingrand *et al.*, 1996] F. F. Ingrand, R. Chatila, and R. Alami F. Robert. Prs: A high level supervision and control language for autonomous mobile robots. In *IEEE ICRA*, 1996.

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

[Le Pape, 1990] C. Le Pape. A combination of centralized and distributed methods for multi-agent planning and scheduling. In *IEEE ICRA*, pages 488–493, May 1990.

[Naimi *et al.*, 1996] M. Naimi, Trehel M, and A. Arnold. A log(n) distributed mutual exclusion algorithm based on path reversal. *Journal of Parallel and Distributed Computing*, 34, 1996.

[Parker, 1994] L.E. Parker. Heterogeneous multi-robot cooperation. In *MIT Technical Report AITR-1465*, February 1994.

[Shoham and Tennenholtz, 1995] Y. Shoham and M. Tennenholtz. On social laws for artificial societies: Off-line design. In *Artificial Intelligence, Vol 73, pp 231-252*, 1995.

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

[T.Vidal *et al.*, 1996] T.Vidal, M. Ghallab, and R. Alami. Incremental mission allocation to a large team of robots. In *IEEE ICRA*, 1996.

[Yuta and S.Premvuti, 1992] S. Yuta and S.Premvuti. Coordination autonomous and centralized decision making to achieve cooperative behaviors between multiple mobile robots. In *IEEE IROS*, pages 1566–1574, July 1992.