

# On the Development of Cooperative Behavior-Based Mobile Manipulators

Bruno S. Pimentel

Guilherme A. S. Pereira

Mário F. M. Campos

VERLab – Laboratório de Visão Computacional e Robótica,  
Departamento de Ciência da Computação,  
Universidade Federal de Minas Gerais, Belo Horizonte, Brazil  
[brunosp, gpereira, mario]@dcc.ufmg.br

## ABSTRACT

We present an approach to the problem of cooperative object carrying, to be performed by a group of mobile robots controlled with a behavior-based architecture. Inherent characteristics of this method such as simplicity, rapid development, robustness, low memory and processing requirements have allowed us to implement a behavioral control system in rather simple, low-processing-power mobile platforms. Robot coordination may be achieved through either implicit or explicit communication. Although implicit communication can make the system more robust to faulty communication environments, we show that the use of the explicit form can avoid some undesirable system locks. Experimental results where two robots carry a relatively large box in an environment cluttered with both easily and hardly avoidable obstacles show the flexibility and effectiveness of the proposed architecture.

## 1. INTRODUCTION

Mobile-robot control systems have been implemented in various ways, ranging from purely deliberative to purely reactive. Deliberative reasoning has its bases on traditional artificial intelligence, and relies on relatively complete world models to predict the outcome of robot actions, so as to optimize its performance for some given criteria. This, however, makes those systems heavily dependent on symbolic representational world models, which must be continuously updated with incoming sensor data in order to allow robot operation in dynamic domains. Although allowing more reliable and robust construction of suitable solutions, deliberative control is harder to implement efficiently. On the other hand, reactive robotic systems tightly couple perception to action – typically in the context of motor behaviors – to produce real-time robotic response in dynamic, unstructured environments, without the use of abstract representations or time history [2]. Reactive systems were the basis for the development of recent behavior-based architectures, which

rely heavily on sensing without constructing potentially erroneous global world models, thus being better suited to situations where the real world cannot be accurately characterized. Although optimal (or even *viable*) actions may not be produced – specially in more complex tasks – lower computational requirements usually result in real-time robot response for incoming sensor data.

In the context of cooperative robotics, several complex situations involving coordination, communication and interaction between multiple agents may arise. The main goal of a cooperative system is to carry out a task improving its overall performance or even allowing it to be completed at all. *Loosely coupled* cooperation arises in those tasks where the main objective does not require close interaction between robots and the goal can be distributed amongst the team, using a strategy similar to divide-and-conquer. For instance, if the task at hand is to explore an unknown environment, it could be more efficiently accomplished by partitioning the area to be covered among the team members. The failure of one or more robots should not compromise the main goal (but probably increase the total exploration time), provided that there were enough units to finish the job. On the other hand, if a task can only be completed through the interaction of a minimum number of robots, it is called a *tightly coupled* task. For example, a single robot is not able to carry a heavy, large box if it doesn't have enough power or grasping ability [7]. Also, a group of mobile robots acting as cooperative predators that must capture a prey is another example of a tightly coupled task since a single robot alone is unable to confine the prey [8]. Given the team capabilities, these tightly coupled tasks could only be satisfactorily completed by close interaction of a number of robots.

Reactive systems have been extensively implemented in many simple loosely coupled tasks. On the other hand, deliberative control seems to be better suited to tightly coupled tasks, due to their inherent complexity in coordinating robot interaction. However, the simplifying characteristics of the first motivate us to apply the reactive control paradigm on the development of a behavior-based architecture that allows a group of cooperative mobile manipulators to carry a large object, while navigating in an unknown, unstructured environment.

Cooperative transport is a task found in many real-world

situations. Ants of various species, for example, perform it in order to move back to their nest large preys otherwise impossible for a single ant to retrieve. As pointed by Kube and Bonabeau in [11], the coordination of ants collective transport seems to occur through the item being transported, since a movement caused by one ant modifies the stimuli perceived by its teammates, which consequently produces changes in their position and orientation. Like the ants, our robots sense the movement of each other through the object being carried. Basic behavioral building blocks were designed and combined to produce actions of targeted navigation, obstacle avoidance, and object grasping control.

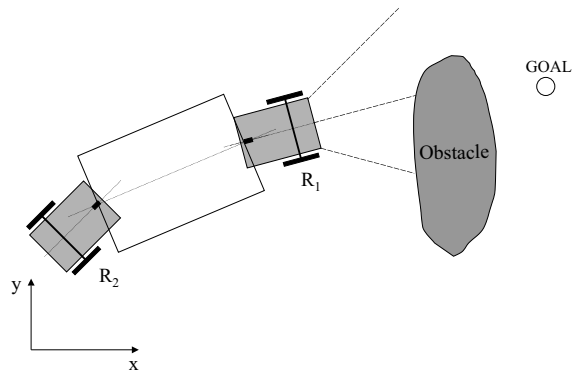
The remaining of the paper is structured as follows: Section 2 discusses some of the applications of the behavior-based paradigm and previous approaches to the cooperative object-carrying task. Section 3 describes the problem tackled in this work. Methodology and system architecture are both presented in Section 4, while Section 5 validates the proposed approach through real-world experiments. Finally, the main contributions are summarized and discussed in Section 6.

## 2. RELATED WORK

Behavior-based cooperative systems have been successfully implemented to perform a wide range of loosely coupled tasks [16, 3]. Formalized behavior-based software architectures have also been developed to allow heterogeneous multi-robot cooperation [13]. Yet, Sugar et al. [18] observed that it is not clear whether the behavior-based methodology could be applied to controlling grasp forces in order to hold and transport objects [7], which is an example of a tightly coupled task. Although Mataric et al. [12] and Kube and Bonabeau [11] have implemented cooperative behavior-based robotic systems in box-pushing tasks, Parker [14] has pointed that carrying an object is considerably more difficult than pushing it, since the carrying task requires the robots to maintain appropriate grasp on the object, while navigating to a given destination in a coordinated fashion. These aspects, together with the need to design cooperative systems capable to operate reliably in simple low-cost devices, provide the main motivation to our work.

Cooperative mobile manipulation has been thoroughly discussed in the literature. The centralized control of tightly coupled cooperative robotic systems has been proven rather difficult [4], motivating the development of distributed approaches [10, 17]. However, depending on the robots kinematic constraints and on the configuration observed between them and the object, complex situations — such as the unstable truck-trailer backing problem [1, 9] — may arise. Sugar and Kumar [20] present a decentralized cooperative system where two robots maintain a tight formation in order to carry a large box. Trajectory information is exchanged via a wireless Ethernet and positioning errors in the formation are compensated by a compliant arm [19]. This architecture is further improved through the implementation of a dynamic leader-follower architecture, which provides additional flexibility to the robotic team [7].

The key aspect of our approach, and the main contribution of this paper, is then the design and implementation of a behavior-based architecture that accomplishes a cooperative



**Figure 1: Two nonholonomic robots carrying a box in an unstructured environment.**

object-carrying task by a group of mobile robots.

## 3. PROBLEM DESCRIPTION

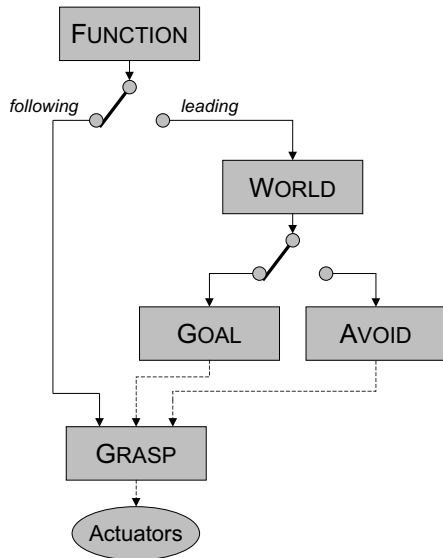
Our problem consists in providing two non-holonomic robots with the basic behaviors needed to cooperatively carry a rigid object from one point to another in an unknown, unstructured environment, clustered with static and dynamic obstacles (Figure 1). Non-holonomicity imposes kinematic restrictions to the group when previously unknown obstacles (common to dynamic, unstructured environments) need to be avoided. Carrying rigid objects simplifies the problem since object deformation does not need to be considered and force information can be directly transmitted from one robot to the other.

In order to maintain appropriate grasp of the object, the robots must use some force-sensitive mechanism such as the compliant arm presented in [19]. The robots can localize themselves in the environment through built-in proprioceptive and exteroceptive sensors. Exteroceptive sensors such as vision, infrared or sonar rangefinders are also used to detect close obstacles.

One major difficulty to control the mobile manipulators in an unknown environment is the obstacle detection and avoidance procedure. Depending on the configuration, the group may not be able to fully avoid the obstacle, without dropping the box. In such situations, the robots must move backwards with some correction orientation to then retry the avoidance procedure. This drawback is mainly due to the sensors' limited range and the ensemble kinematics. If the leading robot ( $R_1$  in Figure 1) detects an obstacle and simply moves backwards without first communicating its intention to the other robot ( $R_2$  in Figure 1), the control problem degenerates to the truck-trailer backing problem, which is a known open-loop unstable system [1, 9]. The following section describes our hierarchical behavior-based architecture designed to accomplish the object-carrying task, and still account for the truck-trailer backing problem.

## 4. THE BEHAVIOR-BASED APPROACH

Behavioral architectures comprise software systems and specifications that provide languages and tools for the construction of behavior-based systems [2]. Our approach shares some of the benefits obtained by the purely reactive behavior-



**Figure 2: System behavioral architecture.** Solid lines represent activation signals, while dashed lines represent control signals.

based *subsumption* architecture [5], such as simplicity, incremental system design and robustness.

We propose a dynamic hierarchical architecture, in which a leader robot guides the ensemble through the environment while a follower robot simply tries to keep up, controlling its relative position to the leader through the maintenance of appropriate grasp on the object. Using such approach, robot  $R_1$  in Figure 1 is initially set as the team leader, being able to pass the leadership every time the group needs to move backwards. The truck-trailer backing problem is avoided, since the leading robot is always moving forward. Also, it is assumed that only  $R_1$  knows the target position. Thus, in order to complete the task, the same robot must start and finish as the team leader.

Given the task requirements, we define the basic behaviors for each robot to perform the object-carrying task:

- **FUNCTION:** Negotiates robot main state through switching between two different behavior patterns – *leading* or *following*. Switching occurs on both robots as a function of the *observed effects* of all active behaviors.
- **WORLD:** Observes the world around the robot in order to trigger the AVOID behavior once an obstacle is sufficiently close; always activates the GOAL behavior when the path is clear.
- **GOAL:** Drives the group towards the target position. To both robots this behavior produces motion patterns that bring them closer to task completion. When the path is completely clear, or robot  $R_1$  is able to successfully avoid the close obstacle without dropping the object, the trajectory followed is a simple curve towards the target position. However, when robot  $R_2$  must assume the leadership — due to the impossibility

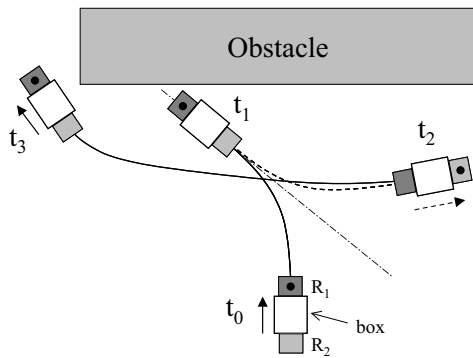
of  $R_1$  to avoid the close obstacle without dropping the object — its trajectory serves more like a controlled maneuver that eventually clears  $R_1$ 's path from the previously unavoidable obstacle (see Figure 3). In  $R_2$ , this behavior is also responsible for resigning the leadership after completing the maneuver.

- **AVOID:** Allows the robot to make an attempt on avoiding close obstacles.
- **GRASP:** Controls robot position to maintain an appropriate force applied on the box.

Our approach combines the functionality of these basic behavioral building blocks into the distributed architecture shown in Figure 2. It is important to notice that robot motion is controlled as a function of its interaction with the object. In this way, the leading robot drives in a determined direction while the other robot tries to maintain the forces and torques applied to the object constant. Initially, the object is in equilibrium since the summation of the forces applied to it equals zero. When  $R_1$  begins moving, the resultant vector points towards the direction of movement. At the same time,  $R_2$ 's controller commands it to move in a way to keep the force summation null.

Initially, the FUNCTION behavior in robot  $R_1$  sets its main state as *leading*, while in robot  $R_2$  as *following*. If there are no close obstacles, GOAL will be active, and a simple proportional controller drives  $R_1$  towards the target position. The control signals generated by GOAL are combined with those produced by GRASP in order to limit  $R_1$  angular and linear velocities in a way to ensure an appropriate grasp on the box. Meanwhile, robot  $R_2$  — in the *following* main state — has only its GRASP behavior active, which generates motor inputs to control the forces applied to the object by adjusting the robot's position and orientation relative to the object. However, if  $R_1$ 's WORLD behavior detects the possibility of obstacle collision, it activates the AVOID behavior, and  $R_1$  attempts to generate a trajectory that leads the group away from the obstacle. Again, the control signals generated by AVOID are combined with those of GRASP in order to avoid dropping the object. Yet, if  $R_1$  fails to avoid the obstacle and the group must maneuver moving backwards, the FUNCTION behavior switches the main robot state on both  $R_1$  and  $R_2$ . FUNCTION detects this need of switching main states every time the leading robot moves backwards, which is sensed by the other through the object. This action is actually interpreted by FUNCTION as transferring the leadership from one robot to the other.

In fact, one could view this form of leadership exchanging as a form of *implicit communication*, since it occurs as a side-effect of the robots' actions [14]. Assuming that a leading robot always moves forward, whenever one of them detects a backward movement, it most certainly mean that the other has resigned the leadership. Conversely, *explicit communication* can also be used to actually exchange control information via wireless, infrared or even wired channels. In such situations, information is exchanged between the FUNCTION behaviors on both robots. Although it has already been show that the implicit form can produce efficient results with simpler actions [3], in Section 5 we further



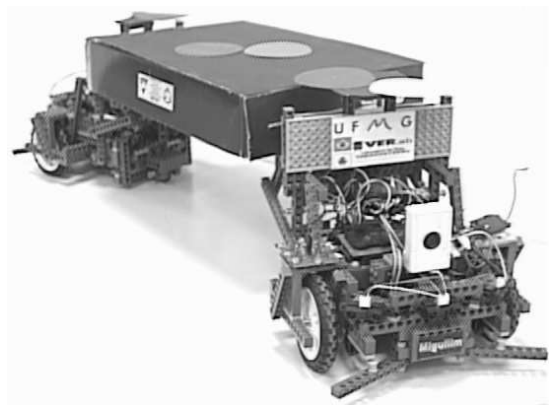
**Figure 3:** Trajectory of robot  $R_2$  when it is driving the group (dashed line) and when it is not (solid line). In  $t_1$ ,  $R_1$  abandons the leadership, and in  $t_2$ ,  $R_2$  returns it. Observe that an heuristically determined mirrored trajectory traced by  $R_2$  maneuvers the group away from the obstacle.

evaluate the impact of both implicit and explicit communication to the system overall performance. Eventual communication losses are handled through timeout mechanisms embedded in FUNCTION, which always select robot  $R_1$  as the team leader whenever both robots happen to be in the *following* main state. One should notice that the situation of two leaders could never occur, since leadership cannot be taken, only abandoned. Also, when a robot assumes the leadership, no advice needs to be sent to the other, since the GRASP behavior on both always ensures that the object will not be dropped.

When  $R_2$  receives the leadership it may not know which direction to take, since the target position may be unknown to it. However, by having observed  $R_1$ 's path prior to leadership exchanging,  $R_2$  may choose a "reasonable" trajectory. The choice of this path is performed by the always active FUNCTION behavior. A simple heuristic which has experimentally proven to be rather efficient is for  $R_2$  to select a mirrored trajectory of  $R_1$  (Figure 3), which significantly improves the group's obstacle avoidance process. A good approximation to  $R_1$ 's trajectory is  $R_2$ 's own trajectory, since both robots are tightly coupled through the box. This trajectory can be easily estimated by local odometry. If it is not possible, due to some robot restriction (e.g. lack of memory or processing), to store  $R_1$ 's trajectory,  $R_2$  can simply move backwards maintaining its current absolute orientation. Although this may not lead to the optimal action, the ensemble would eventually avoid the obstacle.  $R_2$ 's job as group leader is to execute that trajectory for some amount of time after which the leadership is resigned through a backward movement similar to the one previously used by  $R_1$ .

## 5. EXPERIMENTS

Our behavior-based architecture has been implemented in two small, low-processing autonomous robots built from commercially available assembling blocks (Lego<sup>TM</sup>), equipped with common off the shelf sensors, and low cost, simple, in-house-built imaging and communication systems. Figure 4 shows the VERLab robots, *Manuelzão* and *Miguilim*, named after two famous countryside characters immortal-



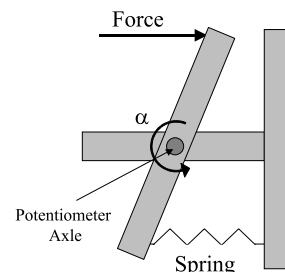
**Figure 4:** Miguilim and Manuelzão in a box carrying task.

ized by Guimarães Rosa, a famous Brazilian writer. It is important to notice that both robots are mechanically identical, which naturally imposes that one of them moves in a non-preferential direction when carrying the box. This kind of behavior can also be observed in some species of ants [11].

### 5.1 Hardware Description

The robots' control system runs on the Handy Board<sup>TM</sup>. This board is based on a 52-pin Motorola MC68HC11 processor with 32K static RAM, four PWM outputs and 7 analog and 9 digital sensor inputs. Several types of sensors are installed on the robots, both for localization and interaction with the environment:

- Proximity Sensors – off the shelf Sharp GP2D15 infrared emitter/receiver sensors that enable obstacle detection in a distance range of 10 to 60cm;
- Contact Sensors – micro-switches mounted on Lego<sup>TM</sup> blocks, used to detect obstacles that were not detected by the proximity sensors;
- Shaft-encoders – optical incremental shaft-encoders, with 16 counts per turn, used for dead reckoning;
- Force Sensors – two force sensitive devices used when the robot is carrying objects. These sensors were assembled using springs and angular potentiometers, as shown in Figure 5.



**Figure 5:** The robots' grasping device.

In order to validate the experiments ground-truth information is needed. The system used here is composed by a color CCD camera and real-time vision software, developed at VERLab [6]. The vision software receives as input a sequence of images containing the robots, which are identified with specific color marks.

Explicit communication between the robots is accomplished by a serial cable connecting the Handy Board serial ports in each robot. A cable is used in order to assume error free communication. This wired serial link can be used in tasks where the robots maintain approximately constant poses relative to each other, within a close distance, as is the case in this work, since the cable does not interfere with the ensemble dynamics. More details about the platforms can be found in [15].

## 5.2 Results

We have tested the proposed system in two different environment configurations. Depending on obstacle size, position and orientation relative to the robot's trajectory, the resultant group actions may differ. In both situations, the robots were programed to transport a box from the origin of the coordinate system to a goal position. Furthermore, obstacles were placed in order to create typical avoidance situations. Robot localization is performed by dead reckoning using shaft encoders. Since these sensors have poor resolution, significant errors are observed. Thus, in order to allow the robots to complete the task, not only small distances were considered, but also the goal was defined as a 15 cm radius circle rather than a point.

Figure 6 shows an example situation in which robot  $R_1$  is able to control its trajectory to successfully deviate from close obstacles without dropping the box. In this case, the goal is located at  $(-2.5\text{ m}, 1.5\text{ m})$ , which forces the ensemble to traverse a somewhat narrow passage between two large obstacles. The control inputs generated by  $R_1$ 's GOAL behavior and limited by GRASP produce a trajectory that successfully guides the group away from the obstacles and towards the target location.

The second environment configuration considers the goal at position  $(2.0\text{ m}, -1.5\text{ m})$ , as shown in Figure 7. The obstacle location forces the robots to exchange leadership in order to reach the target. When robot  $R_1$  fails to avoid the obstacle and a collision is detected by its contact sensors, it begins to move backwards and the FUNCTION behavior on both robots switch their main state. Depending on the configuration, this procedure may repeat itself several times until  $R_1$ 's path is cleared.

On both situations described before, the robots do not communicate explicitly. Despite the good results obtained, in the situation depicted in Figure 7 two distinct problems were observed: *locks* and *failures*. *Locks* occur when none of the robots sense the backward movement, therefore leaving the ensemble with no appointed leader. In this case,  $R_1$ 's FUNCTION behavior triggers the *leading* main state after some time and this robot drives the group to the goal. The completion time, however, is compromised. On the other hand, *failures* occur mainly due to  $R_2$ 's poor estimation of  $R_1$ 's trajectory, which makes the group maneuver in a wrong di-

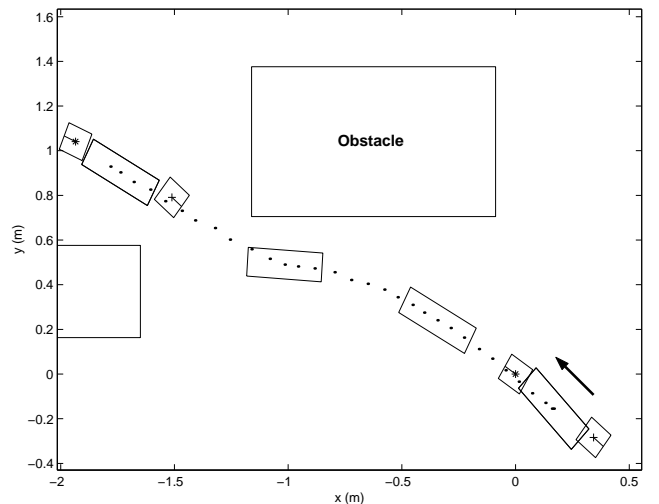


Figure 6: The robots successfully navigating in an unknown, unstructured environment, avoiding static obstacles, without dropping the box.

Table 1: Tests results

| Communication | Successes (%) | Locks (%) | Failures (%) |
|---------------|---------------|-----------|--------------|
| Implicit      | 80            | 12        | 8            |
| Explicit      | 100           | 0         | 0            |

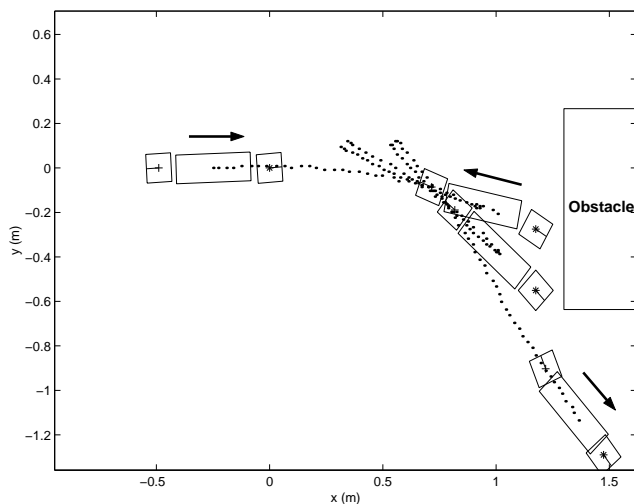
rection. It is important to notice that, despite those unsuccessful situations, the box is not dropped.

In order to overcome such drawbacks, explicit communication was used. In this case, the robots can communicate through a reliable, bidirectional point-to-point channel, so that  $R_1$  can inform  $R_2$  the path to be followed and also request the group leadership when it is able to resume its task. Table 1 shows the results for both situations. It can be seen that, with explicit communication, the robots successfully complete the task in all trials.

## 6. CONCLUSION

We have presented a behavior-based solution to the problem of cooperative object-carrying by two non-holonomic mobile robots. This method has inherent characteristics such as simplicity, rapid development, robustness, low memory and processing requirements, that allow the design to be done incrementally by building basic individual behavior blocks and then integrating their functionality into the architecture. Furthermore, these characteristics motivated the implementation of the system in two very simple robots.

Results from real-world experiments showed that the robots were able to traverse an unknown, unstructured environment, cluttered with obstacles, without dropping the box being carried. Trials involving easily and hardly avoidable obstacles were both carried out thoroughly, showing the flexibility of the behavior-based architecture. It is also important to notice that our behavioral implementation may be unable to deal with some situations, for example, when the AVOID behavior drives the robot in a direction from which



**Figure 7: The robots successfully navigating in an unknown, unstructured environment, avoiding static obstacles, without dropping the box.**

the target position may never be reached. Although our experiments did not show such situation, it could easily take place in unknown, unstructured environments. This problem is directly related to the reactive control paradigm which do not consider world models.

We have also shown that both implicit and explicit communication can be used for our cooperative box carrying task. Although implicit communication can make the system more robust to faulty communication environments, our experimental results showed that the explicit form avoids some undesirable situations. Also, in other scenarios where more complex data must be transmitted, implicit communication may be harder or even impossible to implement.

Future work includes investigating the possibility of using the behavior-based method to perform the carrying task using more than two robots. The initial grasping procedure could also be studied. Still, hybrid reactive/deliberative behaviors might be included in order to improve system performance and avoid undesirable situations. Finally, it would be interesting to evaluate the applicability of the behavior-based method for other tightly coupled tasks.

## 7. REFERENCES

- [1] C. Altafini and A. Speranzon. Backward line tracking control of a radio-controlled truck and trailer. In *Proceedings of the 2001 IEEE International Conference on Robotics and Automation*, pages 169–174, Seoul, Korea, May, 21–26 2001.
- [2] R. C. Arkin. *Behavior Based Robotics*. The Massachusetts Institute of Technology, 1999.
- [3] T. Balch and R. C. Arkin. Communication in reactive multiagent robotic systems. *Autonomous Robots*, 1(1):1–25, 1994.
- [4] J. Borenstein. Control and kinematic design for multi-degree-of-freedom mobile robots with compliant linkage. *IEEE Transactions on Robotics and Automation*, 11(1):21–35, February 1995.
- [5] R. A. Brooks. A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation*, 2(1):14–23, March 1986.
- [6] M. Campos, M. Anício, M. Carvalho, R. Dias, A. Hartmann, D. Nagem, V. Oliveira, E. Oliveira, G. Pereira, A. Ribeiro, F. Sanches, and M. Silveira. MIneiROSOT – the development of a centralized control set of soccer-playing micro-robots. In *Proceedings of 1998 FIRA Robot World Cup*, pages 57–62, Paris, July 1998.
- [7] L. Chaimowicz, T. Sugar, V. Kumar, and M. F. M. Campos. An architecture for tightly-coupled multi-robot cooperation. In *Proceedings of the 2001 IEEE International Conference on Robotics and Automation*, pages 2292–2297, Seoul, Korea, May 2001.
- [8] T. Haynes and S. Sen. Evolving behavioral strategies in predators and prey. In *Workshop on Adaptation and Learning in Multiagent Systems*, pages 32–37, Montreal, Canada, 1995.
- [9] D. F. Hougen, J. Fischer, M. Gini, and J. Slagle. Fast connectionist learning for trailer backing using a real robot. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1917–1922, April 1996.
- [10] K. Kosuge and T. Oosumi. Decentralized control of multiple robots handling an object. In *Proceedings of the IEEE/RJS International Conference on Intelligent Robots and Systems*, pages 318–323, 1996.
- [11] C. R. Kube and E. Bonabeau. Cooperative transport by ants and robots. *Robotics and Autonomous Systems*, 30:85–101, 2000.
- [12] M. Mataric, M. Nilsson, and K. Simsarian. Cooperative multi-robot box-pushing. In *Proceedings of IEEE International Conference on Intelligent Robots and Systems*, pages 556–561, Pittsburgh, PA, 1995.
- [13] L. E. Parker. Alliance: An architecture for fault tolerant multi-robot cooperation. *IEEE Transactions on Robotics and Automation*, 14(2):220–240, April 1998.
- [14] L. E. Parker. Current state of the art in distributed autonomous mobile robotics. In *Proceedings of the International Symposium on Distributed Autonomous Robotic Systems*, pages 3–12, Knoxville, TN, October 2000.
- [15] G. A. S. Pereira, B. S. Pimentel, and M. F. M. Campos. A simple test bed for cooperative robotics. In *Anais do V Simpósio Brasileiro de Automação Inteligente*, Canela, Brasil, 2001, also available at <http://www.verlab.dcc.ufmg.br/coop>.

- [16] P. Pirjanian and M. Mataric. Multi-robot target acquisition using multiple objective behavior coordination. In *Proceedings of the 2000 IEEE International Conference on Robotics and Automation*, San Francisco, CA, April 2000.
- [17] D. Rus, B. Donald, and J. Jennings. Moving furniture with teams of autonomous robots. In *Proceedings of the IEEE/RJS International Conference on Intelligent Robots and Systems*, pages 235–242, Pittsburgh, PA, August 1995.
- [18] T. Sugar, J. Desai, V. Kumar, and J. Ostrowski. Coordination of multiple mobile manipulators. In *Proceedings of the 2001 IEEE International Conference on Robotics and Automation*, pages 3022–3027, Seoul, Korea, May 2001.
- [19] T. Sugar and V. Kumar. Design and control of a compliant parallel manipulator for a mobile platform. In *Proceedings of the 1998 ASME Design Engineering Technical Conferences*, 1998.
- [20] T. Sugar and V. Kumar. Multiple cooperating mobile manipulators. In *Proceedings of the 1999 IEEE International Conference on Robotics and Automation*, Detroit, MI, May 1999.