

THE USE OF HYBRID SYSTEMS IN THE CONTROL OF COOPERATIVE ROBOTS

LUIZ CHAIMOWICZ, MARIO F. M. CAMPOS

*VERLab – DCC – Universidade Federal de Minas Gerais,
Belo Horizonte, MG, Brasil, 31270-010
chaimo@dcc.ufmg.br, mario@dcc.ufmg.br*

VIJAY KUMAR

*GRASP Laboratory – University of Pennsylvania,
Philadelphia, PA, USA, 19104
kumar@cis.upenn.edu*

Abstract— This paper presents a methodology for modeling the coordination of multi-robot teams in the execution of cooperative tasks. The main idea is to use hybrid systems theory in order to model the cooperation. Basically, during the execution of the task each robot can be in one of several discrete states (called roles) being controlled by different dynamic equations in each role. The coordination of the team is obtained by the synchronization of discrete state transitions (role assignments) and communication among the robots. The methodology is demonstrated in a cooperative manipulation task, in which two heterogeneous robots cooperate in order to transport a large object between two different locations in an environment containing obstacles.

Key Words— Robotics; Cooperation; Hybrid Systems.

1 Introduction

Cooperative Robotics has been an active research field in the last few years. Basically, it consists in using a group of robots working cooperatively to execute various types of tasks, trying to increase the robustness and efficiency of task execution. The use of multi-robot teams brings several advantages over single robot approaches. Firstly, depending on the type of the task, multiple robots can execute the task more efficiently by dividing the workload among the team. For other tasks, called *tightly coupled tasks*, the execution cannot be performed by a single robot working alone and the use of multi-robot teams working in coordination is required. Also, groups of simpler and less expensive robots working cooperatively can be used instead of an expensive specialized robot. Robustness is also increased in certain tasks by having robots with redundant capabilities and dynamically reconfiguring the team in case of robot failures.

To execute cooperative tasks, the robots must be coordinated. The coordination of multi-robot teams in dynamic environments is a challenging task. Basically, the actions performed by each team member during each phase of the cooperation must be specified considering several aspects such as robot properties, task requirements, and characteristics of the environment. Also, the coordination mechanism should provide flexibility and adaptability, allowing the robots to complete cooperative tasks more efficiently and robustly. All these factors make the development of adequate coordination mechanisms one of the key issues in cooperative robotics.

In this paper we present a methodology that uses hybrid systems in order to model and control multi-robot teams in the execution of cooperative tasks. Hybrid Systems are dynamical systems composed by discrete and continuous states. In our methodology, the robots can be in one of several discrete states, being controlled by different continuous dynamics within each state. We call each discrete state a *role* in the task. The coordination of the team is obtained by synchronizing the discrete state transitions (role assignments) and exchanging information among the robots. The methodology is demonstrated in a cooperative manipulation task, in which two heterogeneous robots cooperate to carry an object between different locations.

Several researchers have been working in the development of mechanisms for cooperative robotics. Surveys of the field can be found in (Parker, 2000) and (Cao et al., 1997). An interesting approach for multi-robot coordination is the Alliance architecture (Parker, 1998), a behavior-based software architecture for heterogeneous multi-robot cooperation. It has a fault tolerance mechanism that allows the robots to detect failures in their teammates and adapt their behaviors to complete the task. Among the cooperative tasks, cooperative manipulation is one of the most used test beds for cooperative robotics. Several approaches have been proposed for this problem ranging from behavior based fault tolerant approaches (Mataric et al., 1995) to planner based architectures (Noreils, 1993). The use of complex dynamic models and force sensors to describe and sense the interaction among the robots and the object has also been proposed (Kosuge

et al., 1999; Khatib et al., 1995). More recently, different cooperative approaches to the problem of material transportation by multiple robots have been studied. For example, the coordination of multiple robots manipulating objects using ropes (Donald et al., 2000), the cooperation among a human and a group of robots (Hirata and Kotsuge, 2000) and the use of communication for task allocation in a box-pushing task (Gerkey and Mataric, 2001).

Hybrid systems theory has been used by some researchers in the control of cooperative robots. In (Fierro et al., 2001) for example, robots switch between different controllers in order to keep a formation while moving. Cooperative robotics has also been used as a test bed for hybrid system tools, like CHARON (Alur et al., 2000). In this paper, we do not focus on aspects of hybrid systems theory like reachability or stability. Instead, we try to show that hybrid systems are adequate for modeling cooperative tasks and demonstrate this in a cooperative manipulation.

This paper is organized as follows: next section explains some basic concepts about hybrid systems and hybrid automata. Section 3 presents our modeling of cooperative robotics using hybrid systems. Section 4 describes the cooperative manipulation, the test bed task used in this paper. The results of the experiments in real platforms are presented in Section 5 and Section 6 brings the conclusion and possibilities for future work.

2 Hybrid Systems

A hybrid system is a dynamical system composed by discrete and continuous states. The execution of a hybrid system can be defined by a sequence of steps. In each step, the system state evolves continuously according to a dynamical law until a discrete transition occurs. These transitions are instantaneous state changes that separate continuous state evolutions (Alur et al., 1995). Hybrid systems have been used as mathematical models for important and diverse applications such as air traffic management, robotics, biological and medical applications, embedded systems, real-time communication networks, etc. In general, the majority of the systems that are composed by a digital program controlling an analog plant can be modeled using hybrid systems.

A hybrid automaton can be used to describe hybrid systems. It is a finite automaton augmented with a finite number of variables that can change continuously, as specified by differential equations or discretely, according to specific assignments. A hybrid automaton H can be defined as:

$$H = \{Q, V, E, f, Inv, G, Init, R\}.$$

$Q = \{q_1, q_2, \dots, q_n\}$ is the set of discrete states,

also called *control modes* or *control locations*. The set V represents the variables of the system and can be composed by discrete (V_d) and continuous (V_c) variables: $V = V_d \cup V_c$. Each variable $x \in V$ has a value that is given by a function $\nu(x)$. This is called *valuation* (ν) of the variables. Thus, the state of the system is given by a pair (q, ν) , composed by the discrete state and the valuation of the variables. The dynamics of the continuous variables are determined by the flows f , generally described as differential equations inside each control mode. Discrete transitions between pairs of control modes (q_i, q_j) are specified by the control switches E (also called *edges*). Invariants (Inv) and guards (G) are predicates related to the control modes and control switches respectively. The system can stay in a certain control mode while its invariant is satisfied, and can take a control switch when its guard (jump condition) is satisfied. The initial states of the system are given by $Init$, and each control switch can also have a reset statement R associated, to change the value of some variable during a discrete transition.

A classical example of a hybrid automaton is the control of a thermostat. As shown in Figure 1 (Henzinger, 1996), a hybrid automaton can be pictorially represented by a directed graph, where discrete modes are represented by vertices and discrete transitions by edges. In Figure 1, the system starts in the *Off* mode with the temperature x set to 20 degrees. In this mode, the temperature falls according to the differential equation $\dot{x} = -0.1x$. Due to the jump condition $x < 19$, the heater may go to the *On* state as soon as the temperature falls below 19 degrees. And according to the invariant condition $x \geq 18$, at the latest the heater will move to the *On* state when the temperature falls to 18 degrees. In the *On* control mode, the temperature increases according to the equation $\dot{x} = 5 - 0.1x$ and the control jumps to the *Off* mode when the temperature is between 21 and 22 degrees. It is important to note that the hybrid automaton allows nondeterminism in the execution of the system. A discrete transition can be taken in any moment since its guard is valid. To force determinism in the execution, guards and invariants must be synchronized in such a way that a transition will become enabled in the exact moment that an invariant is violated. In the thermostat example, this would be possible setting the guards to $x \leq 18$ and $x \geq 22$.

3 Modeling Multi-Robot Cooperation

In our methodology, we call each discrete state a *role* in the task. We consider that a role is a function that one or more robots performs during the execution of a cooperative task. Each robot will be performing a role while certain internal and external conditions are satisfied, and will as-

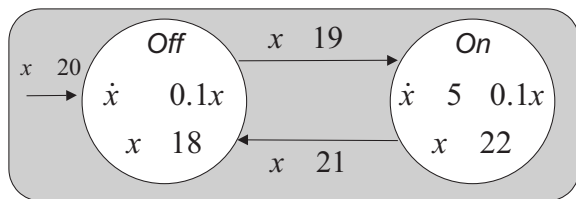


Figure 1. Hybrid automaton for a thermostat.

sume another role otherwise. A role depends on the internal robot state and on information about the environment and other robots, and defines the set of controllers that will be controlling the robot in that moment. Dynamically assuming and exchanging roles, the robots are able to perform the task more efficiently, adapting to unexpected events in the environment and improving their individual performance in benefit of the team. More details about the dynamic role assignment mechanism can be found in (Chaimowicz et al., 2002).

Using a hybrid automaton and representing roles as control modes^a, we are able to better describe the robots during a cooperative task. The internal states and sensory information can be specified by continuous variables, and updated according to the dynamic equations within each mode. The role assignment is represented by the discrete transitions. The invariants and guards define when each robot will assume a new role. Cooperative execution can be represented by a parallel composition of several automata, one for each robot. Using communication the robots are able to synchronize their automata and execute the cooperative task.

In fact, communication is very important in the execution of cooperative tasks, mainly for tightly coupled tasks. Basically, there are two methods for changing information among robots: implicit and explicit communication. Implicit communication occurs as a side-effect of robots' actions, or through the way they change the environment. It does not require any special mechanism: the robots simply use their sensors in order to observe the actions of their teammates gathering the desired information from this observation. On the other hand, in explicit communication robots deliberately send messages to the others, normally using some wireless communication mechanism.

In the experiments presented here, a explicit communication mechanism was used for exchanging information. In a companion paper we show the use of implicit communication in the coordination of multiple robots (Pereira et al., 2002). Here, the robots exchange messages using a wireless ethernet link. There are two types of messages: syn-

chronous and asynchronous. Synchronous messages are sent and received continuously in a constant rate while in asynchronous communication, an interruption is generated when a message is received. Synchronous messages are important in situations where the robots must receive constant updates about the state of the others. On the other hand, asynchronous communication is used when, for example, one robot needs to inform the others about unexpected events or discrete state changes such as the presence of obstacles, robot failures, etc.

4 Cooperative Manipulation

In this paper, we demonstrate the hybrid systems modeling and coordination in a cooperative manipulation task. In this task a team of robots cooperate to carry a large object in environments that may have both static and dynamic obstacles. It is an example of a tightly coupled task because it cannot be performed by a single robot working alone and requires a tight coordination to grasp and transport the object without dropping it.

In the experiment presented here, two heterogeneous robots were used in the implementation of the cooperative manipulation. Figure 2 shows the robots carrying a box. The robot on the left is a TRC Labmate platform, equipped with an actively controlled compliant arm (Sugar and Kumar, 1998). The platform is non-holonomic, and the only on-board sensors are encoders located at the arm and at the two actuated wheels. All the programming is done using Simulink and Real Time Workshop and compiled for DOS. The other robot is a XR4000, developed by Nomadic Technologies. It has a holonomic driving system offering three degrees of freedom (x, y, θ) and is equipped with several types of sensors such as two rings of 24 ultrasound and infrared sensors, a stereo vision system and several encoders. It is also equipped with a fork-lift arm that has one prismatic joint along a vertical axis. It uses the Linux system and the programming is done using C. Both robots are equipped with wireless Ethernet boards and exchange messages using a connectionless protocol.

The robots use a leader-follower approach for the cooperative manipulation: one robot is identified as a leader and the other is designated as a follower. The assigned leader has a planner and broadcasts its estimated position and velocity to the follower using synchronous messages. The follower has its own trajectory controller that acts in order to cooperate with the leader. The planner and the trajectory controllers send set points to the low level controllers that are responsible for the actuators. All robots have a coordination module that controls the cooperative execution of the task. This module receives information from

^aSince we are using control modes to represent roles, we use the terms roles, modes, control modes and discrete states with the same meaning throughout this text.



Figure 2. Two robots carrying a box

the sensors and exchanges asynchronous messages with the other robot. It is responsible for the role assignment and for other decisions that directly affect the planners and trajectory controllers.

The role assignment is used to exchange the leadership responsibilities among the robots: At any moment, the robot performing the leading role can become a follower, and the follower can take over the leadership of the team. This allows the robots to react and adapt easily to unexpected events such as obstacle detection, sensor failures, etc. It is also important to divide the leadership among the robots in such a way that, in each phase of the cooperation, the robot that is best suited in terms of sensor power, manipulation capabilities, etc., will be leading the group.

There are basically three roles in this task: **Dock** where the robots coordinate themselves to get the object, and **Transport** that in fact is composed by the subroles **Lead** and **Follow**. The control modes of the robots' automata are shown in Figure 3.

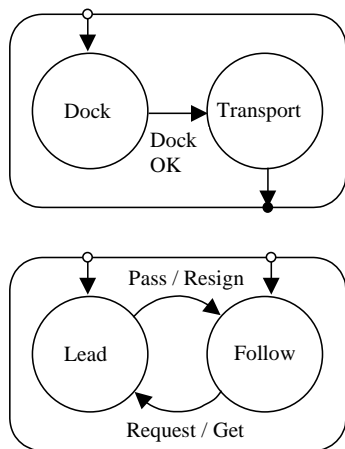


Figure 3. Control modes of the robots in the execution of a cooperative manipulation task

As mentioned, different continuous equations will be used by each robot in each control mode. When the robots are in the Dock mode, they will apply a proportional controller based on the dis-

tance to the object to move in order to grasp the object. In the Transport mode, the robots will have different behaviors when leading or following. In the Lead mode, they will be controlled by planners that send set points to the actuators. In the following mode, the controllers are more complex. The robots should perform a precise trajectory in order to follow and cooperate with the leader. These controllers are described in the next paragraphs.

Since the Labmate is non-holonomic, the inputs for the low level controllers are the linear and angular velocities ($u_1 = v$, $u_2 = \omega$). Thus, the state equations become:

$$\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u}) \rightarrow \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos \theta & 0 \\ \sin \theta & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}.$$

In the Follow mode it uses information sent by the leader together with feedback information from the compliant arm to compute its input. This controller can be thought as if the follower is the rear wheel of a bicycle and the leader is the front wheel. The information of the compliant arm is used to compensate possible odometry errors. Considering that the subscript l refer to information sent by the leader, D is the distance between the robots and \mathbf{x}_a are the state of the compliant arm, the control laws are:

$$u_1 = \hat{v}_l \cos(\hat{\theta}_l - \theta) + f(\mathbf{x}_a), \quad (1)$$

$$u_2 = (\hat{v}_l / D) \sin(\hat{\theta}_l - \theta) + g(\mathbf{x}_a). \quad (2)$$

The XR4000 is holonomic having three degrees of freedom and consequently three inputs ($\dot{x} = u_1$, $\dot{y} = u_2$, $\dot{\theta} = u_3$). When it is performing the Follow role, it tries to maintain a certain distance and orientation relative to the leader. The leader's pose ($\hat{x}_l, \hat{y}_l, \hat{\theta}_l$) and information about the leader's compliant arm ($\hat{\mathbf{x}}_a$) are received through explicit communication^b. For each degree of freedom, we use a proportional controller that tries to keep the robot on the desired position and orientation together with a function that is used to compensate odometry errors based on the position of the leader's compliant arm.

$$u_1 = k_1((\hat{x}_l + D \cos \theta) - x) + f(\hat{\mathbf{x}}_a), \quad (3)$$

$$u_2 = k_2((\hat{y}_l + D \sin \theta) - y) + g(\hat{\mathbf{x}}_a), \quad (4)$$

$$u_3 = k_3(\hat{\theta}_l - \theta) + h(\hat{\mathbf{x}}_a). \quad (5)$$

It is important to mention that all these controllers are specific for these robots and this task, but the general framework, including the role assignment mechanism and hybrid systems modeling, can be used in other cooperative tasks with other types of robots.

^bWe use the superscript ($\hat{\cdot}$) to emphasize that these values are estimates. They may not reflect exactly the actual values due to possible delays and failures in the explicit communication.

5 Results

As mentioned, in the experiment presented in this paper the XR4000 and the Labmate cooperate to carry a box using the leader-follower approach explained before. Basically, there are two ways of changing the leader: leadership request and leadership resignation. In the leadership request, the follower sends a message requesting the leadership. This normally happens when it is not able to follow the leader's plan and/or knows a better way to lead the task in that moment. For example, if one of the followers detects an obstacle, it can request the leadership, avoid the obstacle, and then return the leadership to the previous leader. In the resignation process, the leader relinquishes the leadership to another robot. This can happen when the robot senses that it is unable to continue leading or when it finishes its leading turn in a task that has multiple leaders. In this paper we demonstrate the leadership request process. Other experiments together with a more detailed description of the leader-follower architecture can be found in (Chaimowicz et al., 2001).

Before beginning the transportation, the two robots must coordinate themselves to get the box. This is called the Dock mode of the cooperation. The XR4000 uses its infrared sensor to approach the box and deploy its arm at the correct position. The Labmate uses feedback information from the compliant arm to move and grasp the box. In this experiment, docking is in one dimension (the robots and the box are aligned), and the Labmate waits until the XR4000 finishes before starting its own docking. The synchronization between the robots is obtained exchanging control (asynchronous) messages, but they do not exchange data (synchronous) messages during the docking phase.

After finishing the docking phase, the robots start the box transportation. Figure 4 shows the trajectories executed by the robots transporting the box with data acquired from odometry. The numbers inside the graphs indicate the initial positions and the points where the leadership has been changed. Each robot is indicated by a letter (X-XR4000, L-Labmate). The Labmate begins leading by going backwards in a curvilinear trajectory (from 0L to 1L). The XR4000 begins following (0X) and requests the leadership when its infrared sensor detects an obstacle in its way (1X). After moving to avoid the obstacle, the XR4000 (2X) returns the leadership to the Labmate (2L) that leads until the end of the task. The leadership exchange here is very important because the leader is not aware of the obstacle in the path of the follower. The XR4000 therefore requests the leadership, avoids the obstacle and returns it to the Labmate.

As mentioned, communication is very impor-

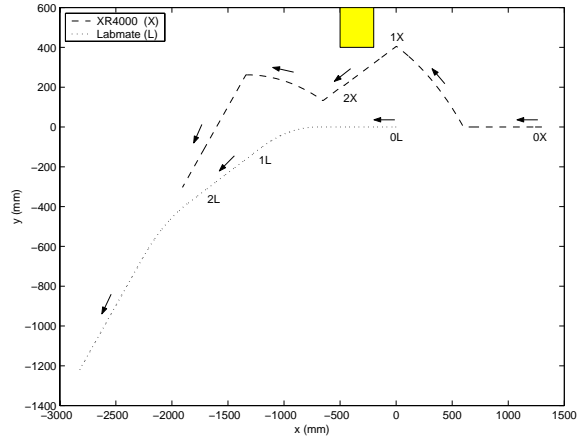


Figure 4. The XR4000 requests the leadership (1X) and returns it (2X) to the Labmate

tant for both coordination and control. Figure 5 shows the hybrid automata of the two robots together with messages exchanged by the robots in each role. The circles represent the discrete states of the robots and the rectangles indicate the continuous dynamics (controllers and planners) for each state. The arrows between the robots are the control and data messages exchanged and the links among the states in a robot are discrete transitions.

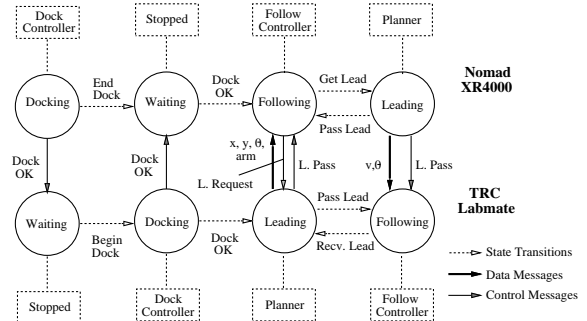


Figure 5. The two automata and the messages exchanged between the robots

6 Conclusion

In this paper we presented a new methodology for modeling the coordination of multiple robots in the execution of cooperative tasks. Each robot performs a set of roles that define its actions during the cooperation. A hybrid systems framework was used to model this roles and the dynamic role assignment, trying to provide a better and more formal way to represent the cooperative system. The methodology was tested in a cooperative manipulation task, where two heterogeneous robots had to exchange roles in order to carry a large object avoiding obstacles. The results showed that the dynamic role assignment allows the multi-robot team to successfully complete the task and

that the hybrid systems approach is a suitable way of modeling this kind of cooperation.

Our future work is directed towards improving the hybrid systems modeling of the cooperation, mainly in the formalization of the parallel composition of multiple automata. This will provide a more complete approach to formally describe the execution of cooperative tasks by multi-robot teams. We also intend to test this methodology in other cooperative tasks, both in real and simulated environments in order to study important aspects such as applicability and scalability.

References

- Alur, R., Coucoubetis, C., Henzinger, T. A., Ho, P.-H., Nicollin, X., Olivero, A., Sifakis, J. and Yovine, S. (1995). The algorithmic analysis of hybrid systems, *Theoretical Computer Science* **138**: 3–34.
- Alur, R., Grosu, R., Hur, Y., Kumar, V. and Lee, I. (2000). Modular specification of hybrid systems in charon, *Proceedings of the 3rd International Workshop on Hybrid Systems: Computation and Control*.
- Cao, Y. U., Fukunaga, A. S. and Kahng, A. B. (1997). Cooperative mobile robotics: Antecedents and directions, *Autonomous Robots* **4**: 1–23.
- Chaimowicz, L., Campos, M. and Kumar, V. (2002). Dynamic role assignment for cooperative robots, *Proceedings of the 2002 IEEE International Conference on Robotics and Automation*, pp. 292–298.
- Chaimowicz, L., Sugar, T., Kumar, V. and Campos, M. (2001). An architecture for tightly coupled multi-robot cooperation, *Proceedings of the 2001 IEEE International Conference on Robotics and Automation*, pp. 2292–2297.
- Donald, B. R., Garipey, L. and Rus, D. (2000). Distributed manipulation of multiple objects using ropes, *Proceedings of the 2000 IEEE International Conference on Robotics and Automation*, pp. 450–456.
- Fierro, R., Das, A., Kumar, V. and Ostrowski, J. (2001). Hybrid control of formations of robots, *Proceedings of the 2001 IEEE International Conference on Robotics and Automation*, pp. 3672–3677.
- Gerkey, B. and Mataric, M. (2001). Principled communication for dynamic multi-robot task allocation, in D. Rus and S. Singh (eds), *Experimental Robotics VII, LNCIS 271*, Springer Verlag, pp. 353–362.
- Henzinger, T. (1996). The theory of hybrid automata, *Proceedings of the 11th Annual Symposium on Logic in Computer Science*, pp. 278–292.
- Hirata, Y. and Kosuge, K. (2000). Distributed robot helpers handling a single object in cooperation with a human, *Proceedings of the 2000 IEEE International Conference on Robotics and Automation*, pp. 458–463.
- Khatib, O., Yokoi, K., Chang, K., Ruspini, D., Holmberg, R., Casal, A. and Baader, A. (1995). Force strategies for cooperative tasks in multiple mobile manipulation systems, *Proceedings of the 7th International Symposium on Robotics Research*, pp. 333–342.
- Kosuge, K., Hirata, Y., Asama, H., Kaetsu, H. and Kawabata, K. (1999). Motion control of multiple autonomous mobile robots handling a large object in coordination, *Proceedings of the 1999 IEEE International Conference on Robotics and Automation*, pp. 2666–2673.
- Mataric, M. J., Nilsson, M. and Simsarian, K. T. (1995). Cooperative multi-robot box-pushing, *Proceedings of the 1995 IEEE/RJS International Conference on Intelligent Robots and Systems*, Vol. 3, pp. 556–561.
- Noreils, F. R. (1993). Toward a robot architecture integrating cooperation between mobile robots: Application to indoor environment, *The International Journal of Robotics Research* **12**(1): 79–98.
- Parker, L. E. (1998). Alliance: An architecture for fault tolerant multi-robot cooperation, *IEEE Transactions on Robotics and Automation* **14**(2): 220–240.
- Parker, L. E. (2000). Current state of the art in distributed robot systems, in L. E. Parker, G. Bekey and J. Barhen (eds), *Distributed Autonomous Robotic Systems 4*, Springer Verlag, pp. 3–12.
- Pereira, G., Pimentel, B., Chaimowicz, L. and Campos, M. (2002). Coordination of multiple mobile robots in an object carrying task using implicit communication, *Proceedings of the 2002 IEEE International Conference on Robotics and Automation*, pp. 281–286.
- Sugar, T. and Kumar, V. (1998). Design and control of a compliant parallel manipulator for a mobile platform, *Proceedings of the 1998 ASME Design Engineering Technical Conferences and Computers in Engineering Conference*.