Hybrid Systems Modeling of Cooperative Robots

Luiz Chaimowicz[1,2], Mario F. M. Campos[1], and Vijay Kumar[2]

[1]DCC – Universidade Federal de Minas Gerais, Belo Horizonte, MG, Brasil, 31270-010

[2]GRASP Laboratory – University of Pennsylvania, Philadelphia, PA, USA, 19104

{chaimo, kumar}@grasp.cis.upenn.edu, mario@dcc.ufmg.br

Abstract

This paper proposes a methodology that uses hybrid systems to model multiple robots in the execution of cooperative tasks. Basically, each robot is represented by a hybrid automaton and the cooperative task execution is modeled by the composition of several automata. We describe in details our approach to perform the composition of these automata and demonstrate the effectiveness of the proposed methodology modeling a cooperative manipulation task.

1 Introduction

Cooperative robotics has been an active research field in recent years. Fundamentally, 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. An important aspect of cooperative robotics that has not received much attention so far is how to model the execution of tasks by multiple robots. In general, cooperative robotics systems normally require the representation of both continuous and discrete dynamics, together with synchronization, communication, etc. The modeling technique should address all these requirements and also provide ways of obtaining some formal results about the cooperative system.

In previous works [6, 7], we gave the first steps into modeling cooperative robotics using hybrid systems. In this paper, we extend and formalize this methodology, showing how the composition of several hybrid automaton, one for each robot, can be used to model the execution of cooperative tasks.

We have chosen hybrid systems in order to represent cooperative robotics for two main reasons. Firstly, hybrid systems provide a simple, structured, and formal way of reasoning about, representing, and implementing multi-robot cooperation. Also, as will be shown in this paper, the mapping from cooperative robotics to a hybrid automaton is relatively simple. The second main reason is that hybrid systems have a powerful theory in its background, which may allow the development of formal proofs about some aspects of the cooperative task execution. This aspects include stability, reachability of undesirable or goal states, etc. In this paper we focus on the cooperative task modeling while formal verification and other types of analysis are left as future work.

In spite of the large number of works dealing with coordination mechanisms and approaches to execute cooperative tasks, very few works have considered aspects of modeling and formal verification of cooperative robotics. A framework that uses a more formal approach to implement cooperative robotics has been developed in the GRASP Laboratory [2]. It uses the Charon language [4] for the description of multiple agents and their behavioral structure. Charon is a high level modeling language for the specification and simulation of hybrid systems, and can be used for modeling cooperative robotic systems. An example of this framework in the control of multi-robot formations can be found in [9]. Another work that tries to formalize the execution of cooperative robotics is described in [11], in which finite state machines are used to encapsulate behaviors and discrete transitions are controlled by binary sensing predicates. Finite state machines do not model continuous aspects of the system but can help in modeling the discrete part of the cooperation. Petri Nets, that are commonly used for modeling multi-process systems, have also been used to model robotics tasks [12, 13].

An interesting discussion about the specification and formal verification of robotic tasks can be found in [8]. One of the conclusions of that paper is that "all the area of hybrid systems (modeling, programming, formal verification) is a key research domain for the future, the results of which will find particularly relevant applications in robotics". Our hybrid systems modeling of cooperative robotics presented in this paper is a contribution in this direction.

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. Discrete transitions are instantaneous state changes that separate continuous state evolutions [1].

Normally a hybrid system can be represented using a Hybrid Automaton. A hybrid automaton 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. Discrete states (control locations) contain evolution laws and the values of the variables change according to these laws while the system is in a specific discrete state. The discrete transitions of the system are labeled with a set of guards and assignments. A transition is enabled when the logical condition of its guard is satisfied. When a transition occurs, the assignment associated with that transition is executed, possibly modifying the values of the variables. Additionally, each control location has an invariant condition that must hold whenever the system is executing in that location. More formally, a hybrid automaton H can be defined as:

$$H = \langle Q, V, E, f, Inv, Init \rangle,$$

where:

- $Q = \{q_1, q_2, \ldots, q_n\}$ is the set of discrete states of the system, also called *control modes* or *control locations*.
- V is a finite set containing 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, at any moment, the state of the system is given by a pair (q, ν), composed by the discrete state and the valuation of the variables.
- Discrete transitions between pairs of control modes are specified by control switches represented by E (also called *edges*). Each transition has an associated predicate g that is called a guard or jump condition. A discrete transition can only be taken (transition is enabled) if its predicate g is satisfied. Each transition may also have a reset statement r that changes the value of some variable or perform some action during a discrete transition. Finally, control switches can be tagged with a label $l \in \Sigma$. This label can be also called *event* and is important for synchronization issues in the composition of multiple automata. Thus, each discrete transition $e \in E$ can be represented by a tuple $\langle q_i, q_j, g, r, l \rangle$ where $q_i \in Q$ and $q_j \in Q$ are the source and destination states, g is the guard, r is the reset statement and $l \in \Sigma$ is the label.
- The dynamics of the continuous variables are determined by the flows f, generally described as differential equations inside each control mode.
- Invariants (Inv) are predicates related to the control modes. The system can stay in a certain control mode q while its invariant is satisfied and must leave the mode when it becomes invalid. If the invariant becomes false and there are no transitions enabled for the current control mode, the system is considered to be in a deadlock.
- $Init$ is the set of initial states of the system. Each initial state is composed by a pair $(q, \nu(X))$, where $q \in Q$, and $X \subseteq V$.

Other representations of hybrid automata may exist depending on the author and the context in which the definition is applied (for example [1] and [10]).

3 Modeling of Cooperative Robots

The behavior of a robot can be modeled using a hybrid automaton. Basically, it can be in one of several control modes, being controlled by different continuous equations in each mode. Information within each mode (such as the continuous states, sensor readings, etc.) can be described by continuous and discrete variables, and updated according to the equations inside each control mode and reset statements of each discrete transition. Coordination is done using the role assignment mechanism [6], in which the robots dynamically change their behavior during task execution. This can be modeled through the discrete transitions and is controlled by the guards and invariants.

Explicit communication can be represented considering that there are communication channels between agents and using a message passing mechanism. The basic actions are *send(channel, value)* to send a message containing a certain value in a channel and *receive(channel, variable)* to receive a message and put its value in a local variable. Messages are sent and received during discrete transitions. It is possible to use a *self-transition*, *i.e.*, a transition that does not change the discrete state, to receive and send messages. In this way, a robot can stay in one discrete mode and continuously send or receive messages.

The execution of a cooperative task by multiple robots can be modeled using a parallel composition of several automata, one for each robot. We built our approach to formally describe the composition of hybrid automata based on the composition of transition systems[1] [3], extending this concept for the composition of hybrid automata.

Let $H_1 = \langle Q_1, V_1, E_1, f_1, Inv_1, Init_1 \rangle$ and $H_2 = \langle Q_2, V_2, E_2, f_2, Inv_2, Init_2 \rangle$ be two hybrid automata. Some considerations have to be made before defining the composition. Firstly, we assume that the sets of

[1] A transition system can be considered the discrete part of a hybrid automata, *i.e*, the graph (Q, E) of the automata without considering variables, invariants, etc.

variables V_1 and V_2 are disjoint. This is a reasonable assumption since there are no shared variables. The two automata can share information, but this information will be stored in different variables and gathered in different ways. Another consideration is that two transitions from E_1 and E_2 labeled with the same label l are taken at the same time in both automata. This is a way of synchronizing the execution of both automata and can be done, for example, using communication. With these considerations, the parallel composition $H_1 \parallel H_2$ of H_1 and H_2 is the hybrid automaton:

$$H_1 \parallel H_2 = \langle Q_1 \times Q_2, V_1 \cup V_2, E', f_1 \cup f_2, Inv', Init' \rangle.$$

The control modes of the compound automaton are pairs (q_1, q_2), where $q_1 \in Q_1$ and $q_2 \in Q_2$, and the set of variables of $H_1 \parallel H_2$ are the union of the sets V_1 and V_2 from both automata. The discrete transitions (E') are defined based on the transitions of both automata:

1. for $l \in \Sigma_1 \setminus \Sigma_2$, for each transition $\langle q_1, q_1', g_1, r_1, l \rangle$ in E_1 and every $q \in Q_2$, E' contains the transitions $\langle (q_1, q), (q_1', q), g_1, r_1, l \rangle \, \forall q \in Q_2$;
2. for $l \in \Sigma_2 \setminus \Sigma_1$, for each transition $\langle q_2, q_2', g_2, r_2, l \rangle$ in E_2 and every $q \in Q_1$, E' contains the transition $\langle (q, q_2), (q, q_2'), g_2, r_2, l \rangle \, \forall q \in Q_1$;
3. if there is a transition labeled l on both automata ($l \in \Sigma_1 \cap \Sigma_2$), for every transition $\langle q_1, q_1', g_1, r_1, l \rangle$ in E_1 and $\langle q_2, q_2', g_2, r_2, l \rangle$ in E_2, E' contains the transition $\langle (q_1, q_2), (q_1', q_2'), g_1 \wedge g_2, r_1 \cup r_2, l \rangle$.

The flows f' of $H_1 \parallel H_2$ are simply the union of the flows from H_1 and H_2: $f' = f_1 \cup f_2$. Since the variable sets V_1 and V_2 are disjoint, the continuous update in each compound control mode (q_1, q_2) is simply the union of the continuous updates from both q_1 and q_2. In the same way, the invariant of each compound mode (q_1, q_2) is the conjunction (*logical and*) of the invariants of both q_1 and q_2: $Inv'(s_1, s_2) = Inv_1(s_1) \wedge Inv_2(s_2)$. This means that the execution of the automaton $H_1 \parallel H_2$ can stay in control mode (q_1, q_2) while the invariants of both q_1 and q_2 are true. Finally, the initial state set $Init'$ of $H_1 \parallel H_2$ is a composition of $Init_1$ and $Init_2$.

Theoretically, the composition of hybrid automata is not very complex and can be implemented by a computer program. But it is important to mention that the modeling of very large systems may require the use of very complex automata and the composition of several automata may cause an exponential explosion of discrete states. Basically, if we have n robots with m control modes each, the number of control modes in the compound automaton will be m^n, thus, the number of control modes of the compound automaton increases exponentially with the number of robots. To minimize these problems we may try to reduce the total number of states to be analyzed, trying to divide the modes of compound automaton. We will do this in the example of the next section, considering only the control modes in the compound automaton that are reachable from the initial state.

4 Example: Modeling Manipulation

In order to demonstrate the composition of hybrid automata in the modeling of cooperative tasks we have modeled a cooperative manipulation task in which two robots coordinate themselves to transport an object (box). We consider that the transportation is performed in one dimension and each robot i has an estimate of its position (x_i) and the position of the box (x_{bi}). In fact, they need only to know their distance to the box, which can be obtained using some kind of range sensor. We also consider that the robots can exchange messages using explicit communication. The robots use a very simple kinematic model, in which the only input is the linear velocity ($\dot{x} = u$). Figure 1 shows a diagram of this task.

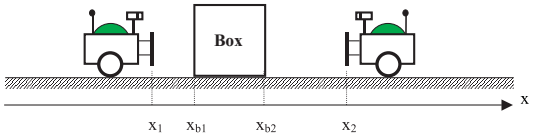


Figure 1: Diagram of two robots transporting a box.

Before formally describing the behavior of the robots using hybrid automata, let us give a brief overview of the execution. The robots can be in one of four discrete states (modes): Approach, Wait, Transport, and Lost. Each robot starts in the Approach mode, where it uses a proportional controller in order to get closer to the box. When the box is close enough, the robot switches to the Wait mode and sends a message to the other robot informing that its approach phase is complete. When both robots have approached the box, they synchronously start the Transport mode, in which they cooperate to move the box. In this mode, one of the robots (robot 1) is the leader and has a plan that sets its velocity. The follower (robot 2) estimates the leader's velocity (v_l) (through sensing or communication, for example) and sets its velocity to the estimated value. If one of the robots loses contact with the box, it sends a message to the other robot and switches to the Lost mode. The other robot will receive this message, send an acknowledgment and both robots will synchronize and switch to the Approach mode, restarting the cycle.

The execution of each robot can be formally modeled using a hybrid automaton. The automaton $H_1 = \langle Q_1, V_1, E_1, f_1, Inv_1, Init_1 \rangle$ for robot 1 is

depicted in Figure 2. The variables of H_1 includes the position of the robot and the box, and three boolean conditions that indicate if the other robot has approached, lost the box, or sent an acknowledgment message. Some self-transitions are used to set these conditions according to the messages received from the other robot. One example is the transition $\langle A_1, A_1, \text{MsgAvailable}(\text{C}_{21}), \text{Recv}(\text{C}_{21}, \text{DockOk}_2), \phi \rangle$, that sets the variable DockOk_2 when a message from robot 2 is available in channel C_{21}. In Figure 2, it is possible to see the guards, actions, and labels of each transition: the guards are shown in normal font, the actions in bold and the labels in bold/italic. Note that some of the transitions may not have all of these components. Flows and invariants are set according to the behavior explained in the previous paragraph, and the execution starts in the Approach mode with the boolean conditions set to false.

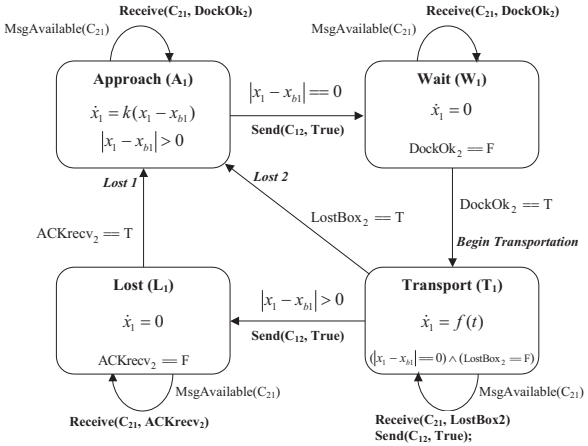


Figure 2: Hybrid automaton for robot 1.

The automaton $H_2 = \langle Q_2, V_2, E_2, f_2, Inv_2, Init_2 \rangle$ for robot 2 is very similar to H_1 and is shown in Figure 3. It is basically a copy of H_1 with the variables and communication channels renamed (from 1 to 2 and 2 to 1). The basic difference is in the dynamic equation of the transport mode. As mentioned, robot 2 is the follower and sets its velocity to be equal to the leader's velocity: $\dot{x}_2 = \hat{v}_l$.

The composition of H_1 and H_2 represents the cooperative execution of the task by the two robots. The hybrid automaton $H_1 \parallel H_2$ is built following the rules explained in the previous section. Figure 4 shows part of this automaton. In fact, $H_1 \parallel H_2$ is composed by 16 control modes, the product of the modes of the two automata ($Q_1 \times Q_2$), but, to simplify the figure, we only show the 8 control modes that are reachable from the initial state of the system.

The composition of control modes, invariants and flows is relatively simple, as explained in the previous section. The major challenge is to define the discrete transitions of the new automaton. According to the

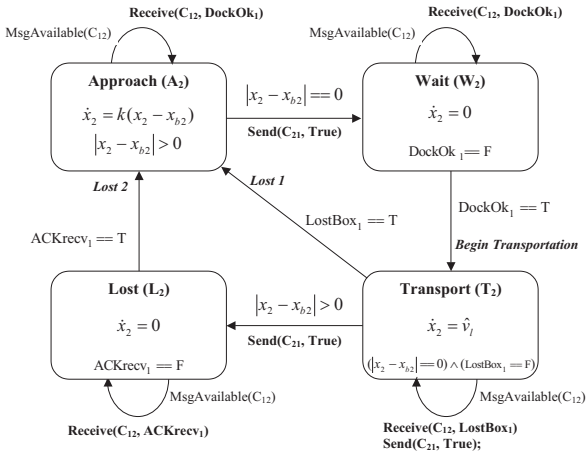


Figure 3: Hybrid automaton for robot 2.

composition rules, there are three situations: (i) when there is a transition labeled l in E_1 but not in E_2, (ii) when there is a transition labeled l in E_2 but not in E_1, and (iii) when there is a transition labeled l in both E_1 and E_2. For situation (i), a transition $q_i \rightarrow q_j$ from mode q_i to mode q_j in E_1 generates a transition $(q_i, q) \rightarrow (q_j, q)$ in the compound automaton for each mode $q \in Q_2$. To exemplify, let us consider the transition $A_1 \rightarrow W_1 \in E_1$. In our modeling, we label only the transitions that are part of both automata. Transitions without labels are unique to H_1 or H_2. In the compound automaton, this transition should appear four times connecting the modes: AA $\rightarrow$ WA, AW $\rightarrow$ WW, AT $\rightarrow$ WT, AL $\rightarrow$ WL [2]. Since we are showing only the reachable modes, this transition appears twice in the automaton of Figure 4. The same thing happens for situation (ii) when there is a transition in H_2 that is not present in H_1. For example, the transition $T_2 \rightarrow L_2 \in E_2$ turns into the transitions AT $\rightarrow$ AL, WT $\rightarrow$ WL, TT $\rightarrow$ TL, LT $\rightarrow$ LL in the automaton $H_1 \parallel H_2$.

The other possible situation (iii) is when there is a transition labeled l in both H_1 and H_2. In this case, as explained in the previous section, a single transition is generated in the compound automaton. For example, the transitions $\text{W}_1 \rightarrow \text{T}_1 \in E_1$ and $\text{W}_2 \rightarrow \text{T}_2 \in E_2$ have the same label *Begin Transportation*. Their composition results in the transition $\langle \text{WW}, \text{TT}, \text{DockOk}_1 == \text{T} \wedge \text{DockOk}_2 == \text{T}, \phi, \textit{Begin Transportation} \rangle$ in the compound automaton. As mentioned, these labeled transitions are used for synchronization. In the modeling, we consider that transitions that have the same label in both automata are taken at the same time. This is the case, for example, of the *Begin Transportation*

[2] To simplify the notation, we use only two letters for the names of the compound modes. The first letter is the mode of Q_1 and the second the mode of Q_2. For example the mode AW in $Q_1 \times Q_2$ is in fact the mode composed by A_1 and W_2.

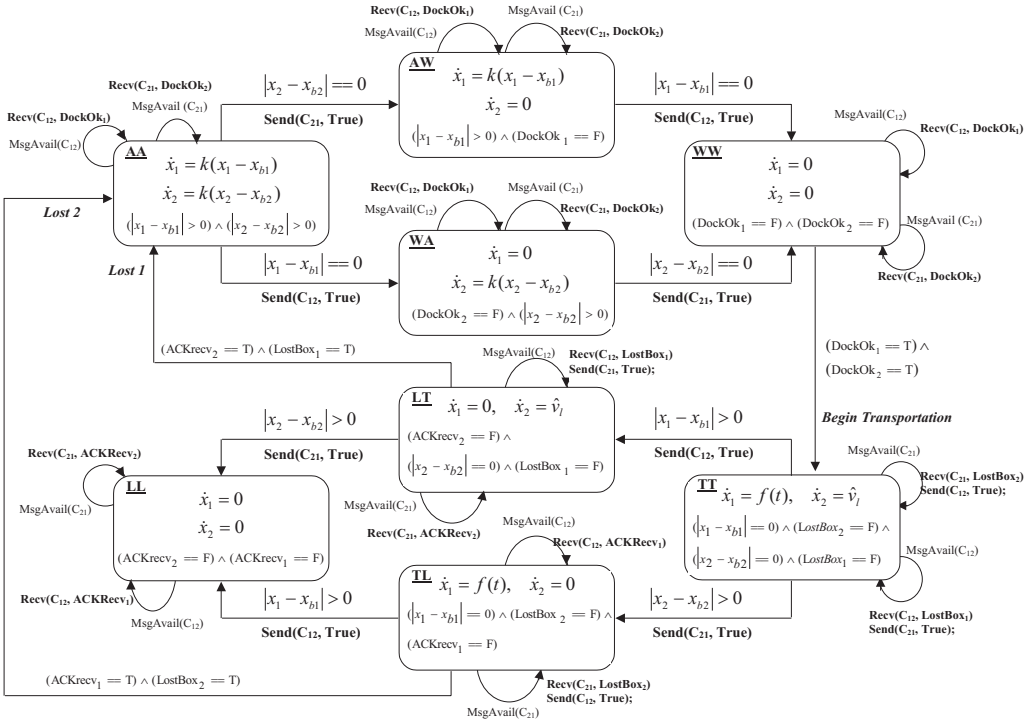


Figure 4: Compound automaton.

transition that synchronize the start of the transportation phase. Another example is the *Lost 1* transition, in which both robots go to the Approach mode when robot 2 receives the information that robot 1 have lost the box and robot 1 receives the acknowledgment.

In real implementations, synchronization is obtained using explicit communication. When a robot receives a certain message or acknowledgment, it sets some of its variables and switches to another discrete state. In the model presented in this example, we consider that a command to send a message is automatically followed by one to receive the message, before any continuous update of the automaton. If this condition is not followed, some undesirable situations such as deadlocks may happen. An example of deadlock is the mode LL of the compound automaton. The mode LL is a sink, since there are no transitions leaving from it. In normal situations, the system does not go to this mode from modes LT or TL (the transitions *lost 1* and *lost 2* to the AA mode are taken instead), but if there is some delay between the sending of a message and the receiving of an acknowledgment and both robots lose contact with the box in this interval this state can be reached. In fact, these deadlock situations can be avoided in the modeling by introducing some extra modes with the objective of waiting for specific messages and synchronizing the automata. A more detailed discussion regarding deadlock situations in this modeling can be found in [5].

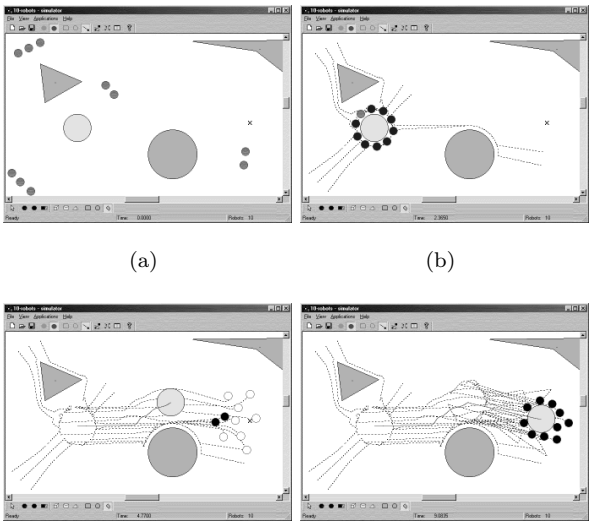
5 Simulations

To show the effectiveness of the hybrid systems modeling we performed some simulations using MuRoS [5], a multi-robot simulator that we have developed for the execution of cooperative tasks. In these simulations we used a group of robots for the transportation of an object in an environment containing obstacles. The hybrid automaton for each robot is similar the ones shown in the previous section (Figures 2 and 3). The main difference is in the definition of the flows, because instead of using a leader-follower controller the robots use potential field controllers.

Initially, the robots are in the Approach mode and are attracted by the object. At the same time, they are repelled by each other, being able to distribute themselves along the object and prepare for the transportation. When one robot senses that it is close enough to the object, it goes to the Wait mode, and broadcasts a message communicating that it is ready. When all robots are ready they change to the Transport mode, and there is a controller switch so that each robot becomes attracted by the goal. If for some reason one robot losses contact with the object, it will change to the Lost mode, broadcast a message and stop moving. If all robots lose contact, they regroup and start docking again.

Figure 5 shows some snapshots of the simulator during the manipulation of a round object by ten holonomic robots in an environment with three obstacles. The goal position is marked with an x. Snapshot

(a) shows the robots in the Approach mode, starting to move in the direction of the object (light gray circle). Snapshot (b) shows nine robots in the Wait mode while the last one is still finishing the approach phase. In (c) eight of the robots have lost contact with the object because of a collision with an obstacle. It is important to note that the robots lose contact when the object is outside their sensor range, so two robots (showed in black) are still in contact. As these two robots move towards the goal, they will also lose contact with the object. When this happens, they regroup, grab the object again and resume the transport finishing the task (snapshot (d)).



(a) (b)

(c) (d)

Figure 5: Snapshots of the manipulation task

6 Conclusion

In this paper, we modeled multi-robot cooperation under a hybrid systems framework, using hybrid automata to specify the behavior of each robot and the parallel composition of automata to model the cooperative task as a whole. This has allowed us to better formalize the execution of cooperative tasks by multiple robots providing a framework that can be used for developing formal proofs about the cooperation.

Future work is direct towards the use of the modeling presented in this paper in order to perform formal verification and analysis on cooperative task execution. As mentioned, hybrid systems have a powerful theory in its background and can be used to study stability and reachability of mixed continuous and discrete systems. Thus, we can use this theory in cooperative robotics to detect deadlocks, test reachability of undesired or target states and study stability of the cooperative system. For this, it may be necessary to abstract our representation trying to obtain simpler models (for example a linear automata) from our general hybrid automata, in order to use the tools already available for the analysis of these classes of hybrid systems.

References

[1] R. Alur, C. Coucoubetis, T. A. Henzinger, P.-H. Ho, X. Nicollin, A. Olivero, J. Sifakis, and S. Yovine. The algorithmic analysis of hybrid systems. *Theoretical Computer Science*, 138:3–34, 1995.

[2] R. Alur, A. Das, J. Esposito, R. Fierro, G. Grudic, Y. Hur, V. Kumar, I. Lee, J. Ostrowski, G. Pappas, B. Southall, J. Spletzer, and C. Taylor. A framework and architecture for multirobot coordination. In D. Rus and S. Singh, editors, *Experimental Robotics VII, LNCIS 271*. Springer Verlag, 2001.

[3] R. Alur and D. Dill. A theory of timed automata. *Theoretical Computer Science*, 126:183–235, 1994.

[4] R. Alur, R. Grosu, Y. Hur, V. Kumar, and I. Lee. Modular specification of hybrid systems in charon. In *Proceedings of the 3rd International Workshop on Hybrid Systems: Computation and Control*, 2000.

[5] L. Chaimowicz. *Dynamic Coordination of Cooperative Robots: A Hybrid Systems Approach*. PhD thesis, Univ. Federal de Minas Gerais - Brazil, June 2002. http://www.cis.upenn.edu/~chaimo/thesis.pdf.

[6] L. Chaimowicz, M. Campos, and V. Kumar. Dynamic role assignment for cooperative robots. In *Proceedings of the 2002 IEEE International Conference on Robotics and Automation*, pages 292–298, 2002.

[7] L. Chaimowicz, T. Sugar, V. Kumar, and 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, 2001.

[8] B. Espiau, K. Kapellos, M. Jourdan, and D. Simon. On the validation of robotics control systems. part i: High level specification and formal verification. Technical Report 2719, INRIA - Rhône-Alpes, 1995.

[9] R. Fierro, A. Das, V. Kumar, and J. Ostrowski. Hybrid control of formations of robots. In *Proceedings of the 2001 IEEE International Conference on Robotics and Automation*, pages 3672–3677, 2001.

[10] T. Henzinger. The theory of hybrid automata. In *Proceedings of the 11th Annual Symposium on Logic in Computer Science*, pages 278–292, 1996.

[11] R. C. Kube and H. Zhang. Task modeling in collective robotics. *Autonomous Robots*, 4:53–72, 1997.

[12] D. Milutinovic and P. Lima. Petri net models of robotic tasks. In *Proceedings of the 2002 IEEE International Conference on Robotics and Automation*, pages 4059–4064, 2002.

[13] F. Wang and G. Saridis. Task translation in integration specification in intelligent machines. *IEEE Transactions on Robotics and Automation*, 9(3):257–271, 1993.