Multi-Modal Control of Constrained Nonlinear Systems

T. John Koo[1,2], George J. Pappas[2], and Shankar Sastry[1]

[1] Department of EECS, University of California at Berkeley, Berkeley, CA 94720
koo,sastry@eecs.berkeley.edu
[2] Department of EE, University of Pennsylvania, Philadelphia, PA 19104
jkoo,pappasg@grasp.cis.upenn.edu

Abstract. This paper is primarily focused on formal approach of hybrid system design for multi-modal control. In multi-modal control paradigm, a set of controllers of satisfactory performance have already been designed and must be used. Each controller may be designed for a different set of outputs in order to meet the given performance objectives and system constraints. When such a collection of control modes is available, an important problem is to be able to accomplish a variety of high level tasks by appropriately switching between the low-level control modes. In this paper, we propose a framework for determining the sequence of control modes that will satisfy reachability tasks. Our framework exploits the structure of output tracking controllers in order to extract a finite graph where the mode switching problem can be efficiently solved, and then implement it using the continuous controllers. Special emphasis is put on scalability and interoperability to cover the complete design flow from control law derivation, and mode switching synthesis to architecture conception. Throughout, a helicopter based unmanned aerial vehicle is used as a design example to demonstrate the effectiveness of the proposed design paradigm for solving the multi-modal control problem.

1 Introduction

Based on operational, financial and environmental considerations, large scale systems ranging from automated highway systems[24], air traffic management systems[22], unmanned aerial vehicle networks[12], communication networks and power distribution networks have been advocated to have higher levels of automation. Recent advances of embedded system technologies in sensing, computation and communication have enabled the rapid realization of sophisticated, high-performance embedded controllers. The control of large scale systems is extremely challenging since by nature the systems are distributed and highly dynamic, the environments in which the systems reside in are usually rapidly evolving, and multi-objective design specifications intensify the complexity of system design. One natural way to manage the complexity of system design is by compositional methods. Of particular interest is multi-modal control paradigm in which control systems are designed by hierarchically nesting of composition of

modes of operation such that each mode of operation is designed to cope with a designated scenario with respect to a design specification while the organization of modes of operation depends on the ordering of these specifications.

A multi-modal control system can be modeled as a hierarchical nesting of parallel and serial composition of discrete and continuous components. Furthermore, a model of computation (MOC) [15] governs the behaviors and interactions of components at each level of the hierarchy. Hybrid systems[4, 1, 18] are considered as formalisms used to describe a complex system as combinations of MOCs. This naturally leads to the generalization of the design problem for the control of large scale systems as a problem of multi-modal control synthesis/design in the modeling framework of hybrid systems. The high-profile and safety-critical nature of such applications has fostered a large and growing body of work on formal methods for hybrid systems: mathematical logics, computational models and methods, and automated reasoning tools supporting the formal specification and verification of performance requirements for hybrid systems, and the design and synthesis of control programs for hybrid systems that are provably correct with respect to formal specifications.

The multi-objective nature of complex control systems requires the use of a suite of controllers instead of using a single controller. In this multi-modal control paradigm, a set of controllers of satisfactory performance have already been designed and must be used. Each controller may be designed for a different set of outputs in order to meet the given performance objectives and system constraints. When such a collection of *control modes* is available, an important problem is to be able to accomplish a variety of high level tasks by appropriately switching between the low-level control modes.

Multi-modal control has been studied especially in the context of stability and safety; see [6] for stability results of switching between stable linear time-invariant controllers, [8] for *safe* switching conditions for systems with pointwise-in-time constraints on state and control, and [19, 3] for controller designs and switching conditions for satisfying multiple objectives such as safety property and optimal performance.

In [11], we have proposed a framework for the synthesis of mode switching for reachability specifications. As opposed to [6], we are interested in reachability specifications rather than preserving stability of a global equilibrium point. The stability of each control mode is assumed to be preserved with respect to the state variables of interest and taken care of by the control design. Switching condition between modes defined in [11] is a generalization of the the result presented in [8] which requires the controllers to be designed in a total order with respect to safety specifications. The outcome of applying our algorithm results in a partial ordering of the given control modes, and not only can captures the outcome generated as shown in [8] but also is more expressive since it captures more possible switching combinations.

In this paper, a *control mode* is defined as the operation of the system under a controller that is *guaranteed* to track a certain class of output trajectories while simultaneously avoiding violation of specified (state or input) constraints. Given

a set of control modes, the mode switching problem attempts to find a finite *sequence* of the control modes as well as *switching conditions* in order to satisfy the given reachability tasks. Hence, the mode switching problem can be posed as:

Problem 1. Given a control system and a finite set of control modes for the system, determine whether there exists a finite sequence of modes that will steer the system from an initial control mode to a desired final control mode. If such a sequence exists, then determine the switching conditions.

In order to reduce the complexity of the mode switching problem, we start by assuming that output tracking control laws have been designed for each control mode. Feedback greatly simplifies the continuous models in each discrete location since the complexity of the continuous behavior is now reduced to the complexity of the trajectories we design. Therefore, many reachability computations that are required in our approach can be greatly simplified by properly *designing* the desired trajectories. Even though feedback control simplifies the continuous complexity, the problem of having nested reachability computations is still present. In order to avoid such expensive computations, as shown in [11], we place a *consistency* condition in our mode switching logic which is reminiscent of the notion of *bisimulation* [20]. We propose an algorithm which given an initial set of control modes, constructs a *control mode graph* which refines the initial control modes but is consistent. Construction of the mode graph can be done off-line or every time a new control mode is designed, allowing the mode switching problem to be efficiently solved on-line, in real time. A helicopter based unmanned aerial vehicle (UAV) is used as a design example to demonstrate the effectiveness of the proposed design paradigm for solving the multi-modal control problem.

2 Formulation of Multi-modal Control Problem

In this section, we introduce a concept of control mode, and precisely define Problem 1 as a *mode switching problem*. First, consider a nonlinear system modeled by differential equations of the form

$$\begin{aligned} \dot{x}(t) &= f(x(t)) + g(x(t))u(t) \\ x(t_0) &= x_0, \quad t \geq t_0 \end{aligned} \tag{2.1}$$

where $x \in \mathbb{R}^n$, $u \in \mathbb{R}^p$, $f(x) : \mathbb{R}^n \to \mathbb{R}^n$ and $g(x) : \mathbb{R}^n \to \mathbb{R}^n \times \mathbb{R}^p$. The system is assumed to be as smooth as needed. Each control mode corresponds to a output tracking controller applying to the nonlinear system (2.1). We now define a concept of control mode.

Definition 1 (Control Modes). *A control mode, labeled by q_i where $i \in \{1, \ldots, N\}$, is the operation of the nonlinear system (2.1) under a closed-loop feedback controller of the form*

$$u(t) = k_i(x(t), r_i(t)) \tag{2.2}$$

associated with an output $y_i(t) = h_i(x(t))$ such that $y_i(t)$ shall track $r_i(t)$ where $y_i(t), r_i(t) \in \mathbb{R}^{m_i}$, $h_i : \mathbb{R}^n \to \mathbb{R}^{m_i}$, $k_i : \mathbb{R}^n \times \mathbb{R}^{m_i} \to \mathbb{R}^p$ for each $i \in \{1, \ldots, N\}$. We assume that $r_i \in \mathcal{R}_i$, the class of output trajectories associated with the control mode q_i, when the initial condition of the system (2.1) starts in the set $S_i(r_i) \subseteq X_i$, output tracking is guaranteed and the state satisfies a set of state constraints $X_i \subseteq \mathbb{R}^n$.

The trajectory $r_i(t)$ is the desired output trajectory, and $y_i(t)$ is the output vector which shall track $r_i(t)$. Notice that in general the initial set may be a function of the trajectory r_i, thus we denote it as $S_i(r_i)$. This is because even though trajectory tracking controllers are guaranteed to converge for any initial condition, trajectory tracking in the presence of state constraints or input constraints can be guaranteed only if the initial tracking error is sufficiently small. State constraints are specified by X_i for $i = 1, \ldots, N$. The state constraints are introduced due to the physical limits of the system and the control design.

In this paper we are interested in switching between controllers, rather than the design of output tracking controllers. We therefore make the following assumption.

Assumption 1 *For each control mode q_i, $i \in \{1, \ldots, N\}$, we assume that a controller of the form (2.2) has been designed which achieves output tracking such that $y_i(t)$ shall track $r_i(t)$ where $r_i \in \mathcal{R}_i \neq \emptyset$, while the state satisfies the set of state constraints $x(t) \in X_i \subseteq \mathbb{R}^n$, when the initial condition of the system (2.1) starts in the set $S_i(r_i) \subseteq X_i \subseteq \mathbb{R}^n$.*

The above assumption is justified given the maturity of output tracking controllers for large classes of linear and nonlinear systems [7, 23]. Based on different design methodologies, the notion of *output tracking* could be different (uniform, asymptotic, exponential, etc.)

Example 1. **Point Mass**. Consider the dynamics of a point mass that can be modeled as a double chain of integrators,

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= u \end{aligned} \tag{2.3}$$

where $x_1, x_2, u \in \mathbb{R}$ represent the position, velocity and acceleration of the point mass, respectively. Define the state as $x = [x_1 \ x_2]^T \in \mathbb{R}^2$. Assume that there are two controllers designed for (2.3) and the corresponding control modes are specified as:

Mode	Output	Reference	Constraint
q_1	$y_1 = x_1$	r_1	X_1
q_2	$y_2 = x_1$	r_2	X_2

where $X_2 \subset X_1 = \mathbb{R} \times (\underline{v}, \overline{v})$ with $\underline{v} < 0 < \overline{v}$. The given controllers are linear controllers and the design is simply based on pole-placement. In order to satisfy

Assumption 1, the controller in control mode q_1 is designed such that $\mathcal{R}_1 \subseteq \mathbb{R}^n$ and $S_1(r_1) = B([r_1\ 0]^T, \delta_1)$ with $\delta_1 < \frac{\min(\underline{v},\overline{v})}{M_1}$ where M_1 is the overshoot constant which can easily be obtained by Lyapunov theorems. Similarly, control mode q_2 are similarly defined but the poles are placed differently. It results in faster response but larger overshoot, *i.e.* $M_2 > M_1$. Hence, $\mathcal{R}_2 \subseteq \mathbb{R}^n$ and $S_2(r_2) = B([r_2\ 0]^T, \delta_2)$ with $\delta_2 < \frac{\min(\underline{v},\overline{v})}{M_2}$. Therefore, we have $S_2(r_2) \subset S_1(r_1)$ if $r_1 = r_2$.

Given two control modes as shown in Example 1, one cannot simply switch from one control mode to another due to incompatible constraints and trajectories. One can easily complicate the situation by introducing many more control modes to serve different performance objectives. A natural question is then whether this mode reachability task as defined Problem 1 can be achieved by a *finite sequence* of modes. Based on the discussion, we can now define the mode switching problem that we will address in this paper.

Problem 2 (Mode Switching Problem). Given an initial control mode q_0 with desired reference r_0, does there exist a sequence of control modes such that the system can reach a desired mode q_F with reference r_F? If so, then determine a mode sequence $q_0 \to \cdots q_i \to q_j \cdots \to q_F$ along with trajectories r_i for each control mode q_i, as well as conditions for switching between the control modes.

For the control modes defined in Example 1, one can define a task of having mode q_1 as an initial mode and ask for a finite control mode sequence to reach mode q_2. Note that Problem 2 is a reachability problem. In this simple example, the problem can be solved by examining the mode switching condition between modes.

In the above mode switching problem, there is enough structure to take advantage of in order to simplify the complexity of the synthesis task. First of all, the continuous controllers are assumed to have been designed, and therefore we do not have to design the continuous part of the system, but simply determine the mode switching conditions. Furthermore, by imposing certain conditions on the allowable mode switches, we reduce the complexity of the synthesis problem, by *maximally decoupling the discrete and continuous aspects of the synthesis.*

3 A Mode Switching Condition

Consider a mode switch from mode q_i to mode q_j. A mode switch from mode q_i to q_j could be allowed if during the operation of the system under mode q_i for some $r_i \in \mathcal{R}_i$, the state reaches the allowable set of initial conditions $S_j(r_j)$ for some $r_j \in \mathcal{R}_j$, *i.e.* there exist $r_i \in \mathcal{R}_i$ and $r_j \in \mathcal{R}_j$ such that

$$\exists x_0 \in S_i(r_i)\ \exists t \geq 0\ \exists x \in S_j(r_j)\ \text{ s.t. } x = \phi_i(t, r_i, x_0) \tag{3.1}$$

where $\phi_i(t, r_i, x_0)$ denotes the *flow* of system (2.1) operating in mode q_i with the controller defined by (2.2) for initial condition x_0, and desired output trajectory

r_i. If one allows this type of mode switching, then reachability critically depends on the particular choice of initial conditions since some initial conditions in $S_i(r_i)$ may reach the set $S_j(r_j)$ of mode q_j while others may not.

In general, nested reachability computations are required for solving such a reachability problem. Furthermore, since loops can be considered in feasible sequences, the number of possible mode sequences could be infinite. Therefore, for this type of mode switching, decidability becomes a central issue [2]. We now characterize the reachable set within each mode.

Definition 2 (Predecessor set). *Given a set $P \subseteq X_i$, a trajectory $r_i \in \mathcal{R}_i$, the reach set $Pre_i(P, r_i)$ in mode q_i is defined by*

$$Pre_i(P, r_i) = \{x_0 \in X_i | \ \exists t \geq 0 \ \exists x \in P \\ s.t.\ x = \phi_i(t, r_i, x_0)\} \tag{3.2}$$

$Pre_i(P, r_i)$ consists of all states that can reach the set P in mode q_i for a given output trajectory r_i, at *some* future time. Furthermore, because of Assumption 1, we have a guarantee that throughout the whole trajectory, the state constraints are satisfied, that is $\phi_i(t, r_i, x_0) \in X_i$ for all $t \geq 0$. Hence, using (3.2), condition (3.1) can be rewritten as

$$S_i(r_i) \cap Pre_i(S_j(r_j), r_i) \neq \emptyset. \tag{3.3}$$

In order to avoid the nested computations mentioned above, as well as break free of restricted decidability results, we constrain our allowable mode switches.

Definition 3 (Consistent mode switching). *Assume that control mode q_i satisfies Assumption 1, that is $\phi_i(t, r_i, x_0) \in X_i$ for all $t \geq 0$ with initial conditions starting from $S_i(r_i)$ where $r_i \in \mathcal{R}_i$. A transition from mode q_i to mode q_j is allowed only if there exist $r_i \in \mathcal{R}_i$ and $r_j \in \mathcal{R}_j$ such that*

$$S_i(r_i) \subseteq Pre_i(S_j(r_j), r_i) \tag{3.4}$$

Therefore, if there exist trajectories r_i (in mode q_i) and r_j (in mode q_j) such that, if the system starts at *any* $x_0 \in S_i(r_i)$, then switching from mode q_i to q_j can occur at some time t such that $\phi_i(t, r_i, x_0) \in S_j(r_j)$. The consistent mode switching condition is shown in Figure 1. The condition expressed in Definition 3 is a consistency condition that guarantees that our ability to get from mode q_i to mode q_j for the particular trajectory pair (r_i, r_j) is independent of the choice of initial condition in $S_i(r_i)$. Hence, a mode switching from mode q_i to mode q_j is possible, if there exists a trajectory $r_i \in \mathcal{R}_i$ that will steer the system state to an initial set $S_j(r_j)$ with $r_j \in \mathcal{R}_j$ independently of where we start in $S_i(r_i)$. The collection of the trajectory pair (r_i, r_j) is specified by the following definition

$$\mathcal{R}^{ij} = \{(r_i, r_j) \in \mathcal{R}_i \times \mathcal{R}_j \mid \text{Cond. (3.4) is satisfied } \} \tag{3.5}$$

Therefore, *every* trajectory pair $(r_i, r_j) \in \mathcal{R}^{ij}$ will steer the system from mode q_i to mode q_j. For each $(r_i, r_j) \in \mathcal{R}^{ij}$, the only thing that depends on the initial

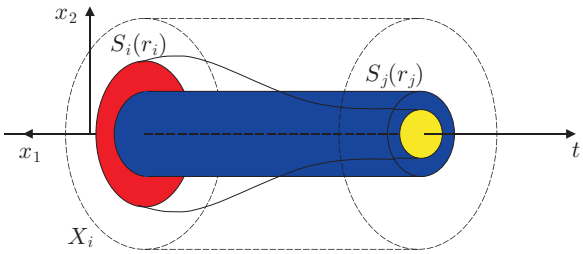


Fig. 1. Consistent mode switching condition

condition is *when* the state will reach $S_j(r_j)$, but not *if* the state will reach $S_j(r_j)$. In our problem we apply the existing results in computing the reachable sets [3, 5, 13, 14, 19] to test the mode switching condition (3.4) and compute the sets $\mathcal{R}^{ij}$. Since most of these reachability computations are approximate, one must consider an over-approximation of $S_i(r_i)$ and an under-approximation of $Pre_i(S_j(r_j), r_i)$, in order to satisfy condition (3.4), that is

$$\overline{S}_i(r_i) \subseteq \underline{Pre}_i(\underline{S}_j(r_j), r_i) \tag{3.6}$$

where $\overline{P}$ and $\underline{P}$ denote the over-approximation and under-approximation of a set P, respectively.

4 Mode Sequence Synthesis

By focusing on the trajectory sets $\mathcal{R}^{ij}$ rather than the initial sets, the mode switching condition (3.4) makes the mode switching problem much more tractable. Furthermore, the construction presented in this section will abstract the mode switching logic into a purely discrete graph. Therefore one can first determine the sequence of modes using standard algorithms for discrete graph reachability, and then determine the continuous parameters r_i for each mode. This will decouple the discrete from the continuous aspects of the problem, and allow continuous techniques for continuous problems, and discrete techniques for discrete problems.

Consider a collection of control modes $Q = \{q_1, \ldots, q_N\}$. If there exist trajectory pairs $(r_i, r_j) \in \mathcal{R}^{ij}$ that can transfer the system from mode q_i to q_j, there would be a transition $q_i \rightarrow q_j$. However, given $q_i \rightarrow q_j$ and $q_j \rightarrow q_k$, if $\mathcal{R}^{ij} \cap \mathcal{R}^{jk} = \emptyset$ there does not exist a trajectory r_j, which will take a point $x \in S_i(r_i)$ to $S_k(r_k)$ via $S_j(r_j)$. In order to construct a consistent *control mode graph* such that the high level mode switching logic is implementable at the lower level by the continuous controllers, transitivity should be preserved.

As shown in [11], each control mode q_i gets refined to $2N$ *submodes*, where N submodes stand for entering mode q_i from any other mode q_j, and N more copies for exiting mode q_i towards any other mode q_j. Therefore, this control

mode graph has some discrete memory, in the sense that each state represents not only which mode the system is in, but also which mode will either precede it or has preceded it. If the set $\mathcal{R}^{ij}$ can be expressed as a decoupled product of the form $\mathcal{R}^{ij} = \mathcal{R}_i^{ij} \times \mathcal{R}_j^{ij}$ where $\mathcal{R}_i^{ij} = \{r_i \in \mathcal{R}_i \mid (r_i, r_j) \in \mathcal{R}^{ij}\}$ and $\mathcal{R}_j^{ij} = \{r_j \in \mathcal{R}_j \mid (r_i, r_j) \in \mathcal{R}^{ij}\}$, then the choice of trajectory $r_i \in \mathcal{R}_i^{ij}$ in mode q_i would work for any trajectory $r_j \in \mathcal{R}_j^{ij}$ in mode q_j, *i.e.*

$$\forall r_i \in \mathcal{R}_i^{ij} \ \forall r_j \in \mathcal{R}_j^{ij} \qquad \text{condition (3.4) is satisfied.} \tag{4.1}$$

This decoupling allows us to consider switching via submodes. Within each mode, we can check for submode consistency by simply performing set intersections. Since there are maximally $2N$ submodes of N modes, a total of N^2 pairwise reachability computations and $N(N)^2 = N^3$ intersections must be computed.

Algorithm 1:(Consistent Control Mode Graph)

Input Control Modes $Q = \{q_1, \ldots, q_N\}$
Output Control Mode Graph $(Q_c, \rightarrow_c)$
Initialize $Q_c := \emptyset$, $\rightarrow_c = \emptyset$
Determine Mode Interconnections
for $i = 1 : N$; **for** $j = 1 : N$
  Compute sets $\mathcal{R}^{ij}$ using (3.4) and (3.5)
  if $\mathcal{R}^{ij} = \mathcal{R}_i^{ij} \times \mathcal{R}_j^{ij}$;
  $q_i^{ij} := q_i$, $q_j^{ij} := q_j$,
  $Q_c := Q_c \cup \{q_i^{ij}, q_j^{ij}\}$,
  $\rightarrow_c := \rightarrow_c \cup \{(q_i^{ij}, q_j^{ij})\}$
  end if
end for; end for
Determine Submode Interconnections
for $j = 1 : N$
  $\check{Q} := \{q_j^{nj} \in Q_c | \exists n \ s.t. \ (q_n^{nj}, q_j^{nj}) \in \rightarrow_c\}$
  $\hat{Q} := \{q_j^{jm} \in Q_c | \exists m \ s.t. \ (q_j^{jm}, q_k^{jm}) \in \rightarrow_c\}$
  for all $q_j^{ij} \in \check{Q}$; **for all** $q_j^{jk} \in \hat{Q}$
  if $\mathcal{R}_j^{ij} \cap \mathcal{R}_j^{jk} \neq \emptyset$;
  $\rightarrow_c := \rightarrow_c \cup \{(q_j^{ij}, q_j^{jk})\}$
  end if
  end for; end for
end for

We now summarize the ideas and present an algorithm for constructing the consistent control mode graph. The algorithm starts with the pairwise reachability computations (3.4,3.5), and performs the submode interconnections. After applying the algorithm, we obtain a finite *control mode graph* $(Q_c, \rightarrow_c)$ which has

been shown in [11] to be *consistent*. Without loss of generality, in the following discussion, we assume that the given initial and final control mode in Q can be represented by $q_0 \in Q_c$ and $q_F \in Q_c$ respectively. Given an initial control mode $q_0 \in Q_c$, the problem of whether we can reach control mode $q_F \in Q_c$, can be efficiently solved using standard reachability algorithms.

Furthermore, one can determine the shortest path (minimum number of mode switches) between mode q_S and q_F, in the control mode graph. The structure that we have imposed on our control mode graph, immediately results in the following solution to the mode switching problem.

Theorem 1 (Mode Switching Solution). *Given a collection of control modes Q, consider the mode switching Problem 2. Construct the consistent control mode graph $(Q_c, \rightarrow_c)$ as described in Algorithm 1. If there exists a path in the consistent control mode graph between q_S and q_F with feasible trajectories r_0 and r_0, then Problem 2 is solvable.*

Having determined the sequence of modes that can steer our system from q_0 to q_F, we are left with the problem of determining the parameters r_i for each mode of the sequence. By construction, such parameters exist and may be selected from the computed sets. Furthermore, it is reasonable to pose the problem of choosing r_i within mode q_i as an optimization or an optimal control problem.

5 Multi-Modal Control of a Helicopter based UAV

The design framework presented in previous sections has been applied to the control of a a helicopter model [17] described in longitudinal and vertical axes with simplified force and moment generation processes. We first present the controller design to illustrate how to compute the reachable sets, then we show how to check the consistent mode switching condition between control modes. The x, z-axes of the spatial frame are pointing to north and down directions. The body x-axis is defined from the center of gravity to the nose of the helicopter, and body z-axis is pointing down from the center of gravity. The motion of the helicopter is controlled by main rotor thrust, T_M and longitudinal tilt path angle, a_M. The pitch angle is defined by θ. The equations of motion can be expressed as:

$$\begin{bmatrix} \ddot{p}_x(t) \\ \ddot{p}_z(t) \end{bmatrix} = \frac{1}{m} \begin{bmatrix} \cos\theta(t) & \sin\theta(t) \\ -\sin\theta)(t) & \cos\theta(t) \end{bmatrix} \begin{bmatrix} -T_M(t)\sin a_M(t) \\ -T_M(t)\cos a_M(t) \end{bmatrix} + \begin{bmatrix} 0 \\ g \end{bmatrix} \tag{5.1}$$

$$\ddot{\theta}(t) = \frac{1}{I_y}(M_M a_M(t) + h_M T_M(t)\sin a_M(t)) \tag{5.2}$$

The state vector and input vector are defined as $x = [p_x, \dot{p}_x, p_z, \dot{p}_z, \theta, \dot{\theta}]^T \in \mathbb{R}^6$ and $u = [T_M, a_M]^T \in \mathbb{R}^2$, respectively.

Control Mode	Output	Reference	Constraint
q_1: Hover	$y_1 = [p_x, p_z]^T$	r_1	X_1
q_2: Cruise	$y_2 = [\dot{p}_x, \dot{p}_z]^T$	r_2	X_2
q_3: Ascend	$y_3 = [\dot{p}_x, \dot{p}_z]^T$	r_3	X_3
q_4: Descend	$y_4 = [\dot{p}_x, \dot{p}_z]^T$	r_4	X_4

Define $X_1 = X_2 = \mathbb{R} \times (\underline{v}_x, \overline{v}_x) \times \mathbb{R} \times (\underline{v}_z, \overline{v}_z) \times (-\pi/2, \pi/2) \times \mathbb{R}$, $X_3 = \mathbb{R} \times (\underline{v}_x^{cr}, \overline{v}_x) \times \mathbb{R} \times (\underline{v}_z, \overline{v}_z^{as}) \times (-\pi/2, \pi/2) \times \mathbb{R}$, and $X_4 = \mathbb{R} \times (\underline{v}_x^{cr}, \overline{v}_x) \times \mathbb{R} \times (\underline{v}_z^{de}, \overline{v}_z) \times (-\pi/2, \pi/2) \times \mathbb{R}$ where $\underline{v}_x < 0 < \underline{v}_x^{cr} < \overline{v}_x$ and $\underline{v}_z < \underline{v}_z^{de} < 0 < \overline{v}_z^{as} < \overline{v}_z$. To satisfy Assumption 1, several control design methodologies can be used to design a controller for each discrete control mode q_i where $i \in \{1, 2, 3, 4\}$. Each controller implementation can be specified as $u = k_i(x, r_i)$ with $r_i \in \mathcal{R}_i$ where $\mathcal{R}_i$ defines the class of admissible output trajectories in mode q_i, and the performance of the closed-loop system can be specified by *initial set*, $S_i(r_i)$, and *flow*, $\phi_i(t, r_i, x_0)$ where $x_0 \in S_i(r_i)$. We assume that all output trajectories are *constant* trajectories, therefore, all controllers are setpoint regulators. Choosing computable classes of trajectories makes the reachability computations simpler.

Given the specifications for the control modes, a nonlinear control scheme [9] based on outer flatness is applied for the design of the controllers. For each mode, the closed-loop dynamics with states defined by $x_{ex} = [p_x, \dot{p}_x, p_z, \dot{p}_z, \theta, \dot{\theta}, T_M, a_M]^T \in \mathbb{R}^8$ can be decoupled into an inner system and two outer subsystems which specify the dynamics in x and z directions. In the following presentation, the Hover mode is presented to illustrate how the reachable set can be computed.

For q_1, the output tracking controller is designed such that $y_1(t)$ shall track $r_1 = [r_{1x}, r_{1z}]^T$ and the output tracking error is uniformly ultimately bounded. Furthermore, because of satisfying Assumption 1, the controller is designed with initial set $S_1(r_1) = B^1([r_{1x}, 0]^T, \epsilon_{1x}) \times B([r_{1z}, 0]^T, \epsilon_{1z}) \times S_{in}$ where $r_1 \in \mathcal{R}_1 = \mathbb{R}^2$, $\epsilon_{1x}, \epsilon_{1z} > 0$ and $S_{in} \subseteq (-\pi/2, \pi/2) \times \mathbb{R}^3$ such that for $x(t_0) \in S_1(r_1)$ then

$$\begin{cases} \|e_{1x}(t)\| \le M_{1x} \exp(-\alpha_{1x} t)(\|e_{1x}(t_0)\| + \delta_{1inx}), \\ \|e_{1z}(t)\| \le M_{1z} \exp(-\alpha_{1z} t)(\|e_{1z}(t_0)\| + \delta_{1inz}), \\ x_{in} \in X_{in}, \ \forall t_0 \le t < t_0 + T_1; \end{cases} \text{ and } \begin{cases} \|e_{1x}(t)\| \le \delta_{1x}, \\ \|e_{1z}(t)\| \le \delta_{1z}, \\ x_{in} \in S_{in}, \ \forall t \ge t_0 + T_1 \end{cases} \tag{5.3}$$

for some T_1 M_{1x}, M_{1z}, α_{1x}, α_{1z}, δ_{1inx}, δ_{1inz}, δ_{1x}, $\delta_{1z} > 0$. In above, $e_{1x} = [p_x - r_{1x}, \dot{p}_x]^T$, $e_{1z} = [p_z - r_{1z}, \dot{p}_z]^T$, and $x_{in} = [\theta, \dot{\theta}, T_M, a_M]^T$. Equation (5.3) explicitly over specifies the reachable set of the mode q_1 by examining the stability property. For other modes, although the control designs are slightly modified for tracking different outputs, the reachable sets of other modes are similarly computed. In Figure 2, we show the initial sets of all the control modes by projecting them onto $p_x - \dot{p}_x$ plane and $p_z - \dot{p}_z$ plane where the projection operator is defined as $\Pi_i : x_{ex} \mapsto (p_i, \dot{p}_i)$ for $i \in \{x, z\}$. In summary, the control modes can be specified by

[1] $B(r, \epsilon) = \{\eta | \ \|\eta - r\| < \epsilon\}$.

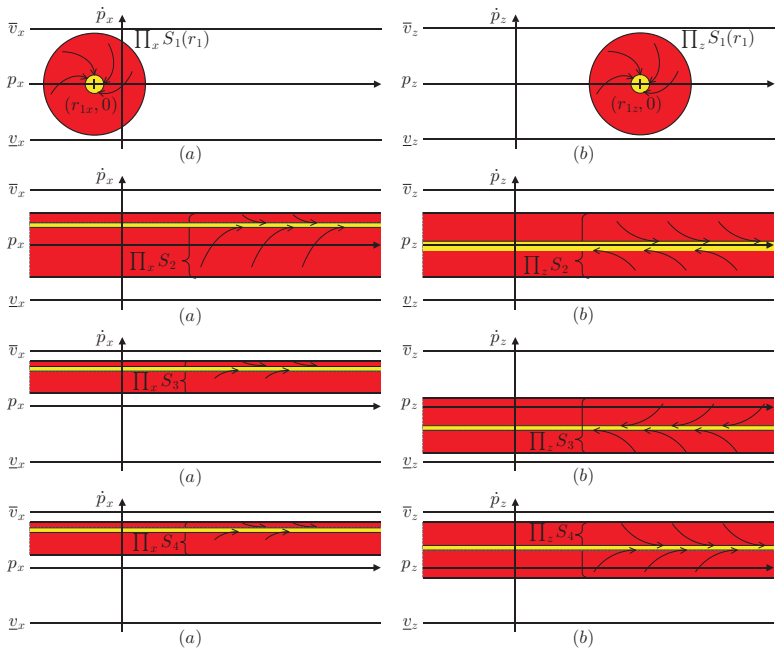


Fig. 2. Projection of $S_1(r_1)$, $S_2(r_2)$, $S_3(r_3)$, and $S_4(r_4)$ onto: (a) $p_x - \dot{p}_x$ plane; (b) $p_z - \dot{p}_z$ plane

Control Mode	Trajectory Set	Initial Set
q_1	$\mathcal{R}_1 = \mathbb{R}^2$	$S_1 = B(0,4) \times B(0,4) \times S_{in}$
q_2	$\mathcal{R}_2 = [-3,3] \times \{0\}$	$S_2 = \mathbb{R} \times (-3.5, 3.5) \times (-3.5, 3.5) \times S_{in}$
q_3	$\mathcal{R}_3 = [2,4] \times [-3,0]$	$S_3 = \mathbb{R} \times (1.5, 4.5) \times (-3.5, 0.5) \times S_{in}$
q_4	$\mathcal{R}_4 = [2,4] \times [0,3]$	$S_4 = \mathbb{R} \times (1.5, 4.5) \times (-0.5, 3.5) \times S_{in}$

where $S_{in} = B(0, 0.2)$, $X_{in} = (-\pi/2, \pi/2) \times \mathbb{R}^3$ and the associated parameters are defined as $\underline{v}_x = -6$, $\underline{v}^{cr} = 1$, $\overline{v}_x = 6$, $\underline{v}_z = -6$, $\overline{v}_z^{as} = 1$, $\underline{v}_z^{de} = -1$, $\overline{v}_z = 6$.

Given the set of control modes, we generated the consistent control mode graph by applying Algorithm 1. In Figure 3, we illustrate the idea of computing the reachable sets on $p_x - \dot{p}_x$ plan. One can easily see the advantage of using feedback, since it is straight forward not only to check the consistent mode switching condition but also to determine the feasible range of trajectory, that is compute the sets $\mathcal{R}^{ij}$. In particular consider the pair (q_1, q_2), that is the transition from hover to cruise. As can be seen from the left side of Figure 3, the consistency condition is trivially satisfied since the ball $S_1(r_1)$ will *eventually* shrink towards the setpoint $(r_{1x}, 0)$, and as a result, will be totally contained inside $S_2(r_2)$ for any r_2. Therefore, in this case $\mathcal{R}^{12} = \mathcal{R}_1 \times \mathcal{R}_2$. Therefore, feedback allows us to check very easily the consistency condition and compute the sets $\mathcal{R}^{ij}$. The right side of Figure 3 shows the similar graphical computation for the mode transition (q_2, q_3), from cruise to ascend. In a similar manner, we

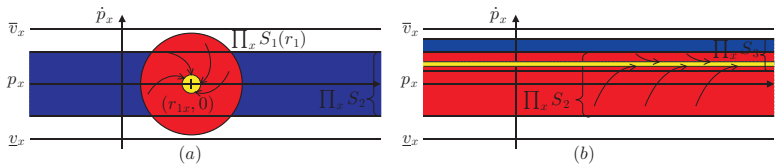


Fig. 3. Graphical illustration of performing reachability computation for checking consistent mode switching condition on $p_x - \dot{p}_x$ plane: (a) $q_1 \to q_2$; (b) $q_2 \to q_3$

have checked the following pairs,

$$\{(q_1, q_2), (q_2, q_2), (q_2, q_3), (q_2, q_4), (q_3, q_2), (q_3, q_3), (q_3, q_4), (q_4, q_2), (q_4, q_3), (q_4, q_4)\}$$

All of the above reachability computations were extremely simple to check. The result of applying Algorithm 1 is summarized in the control mode graph that is shown in Figure 5.

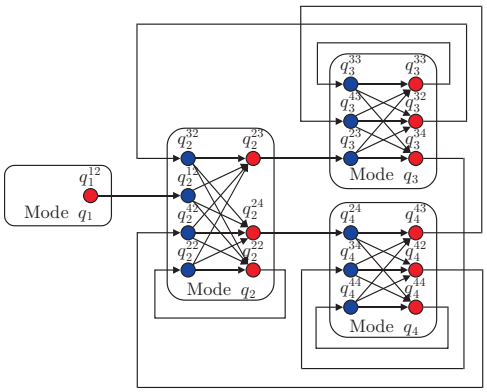


Fig. 4. Consistent control mode graph for the multi-modal helicopter control example

Recall the *high-altitude takeoff* task, which is the task of having the Hover mode q_1 as an initial mode and ask for a finite control mode sequence to reach the Ascend mode q_3. We can now see that from Figure 5 that q_1 has $\{q_1^{12}\}$ as a submode, and $q_3 = \{q_3^{23}, q_3^{43}, q_3^{32}, q_3^{34}\}$, and there exit many paths which are feasible for achieving the task. However, $q_1^{12} q_2^{12} q_2^{23} q_3^{23}$ gives a solution to the task with the minimum number of mode switches, *i.e.*, $q_1 \to q_2 \to q_3$. Given a cost function with respect to the continuous variables, the performance of the sequence can now be optimized with respect to the feasible trajectories. We have therefore decoupled the problem in a purely discrete graph search problem, and a collection of continuous designs within each mode.

Simulation results of the controlled system based on the selected sequence are shown in Figure 5. In the simulation, we can choose $r_1 = [0\ 0]^T \in \mathcal{R}_1^{12}$, $r_2 = [2\ 0]^T \in \mathcal{R}_2^{12} \cap \mathcal{R}_2^{23}$ and $r_3 = [3\ -1]^T \in \mathcal{R}_3^{23}$. The initial conditions of the outer system are $p_x(0) = -2$, $\dot{p}_x(0) = -0.2$, $p_z(0) = 1$, $\dot{p}_z(0) = 0.5$. The initial

condition of the inner system, $x_{in}(0) \in S_{in}$. Mode switchings occur at $t = 20$ for $q_1 \rightarrow q_2$ and at $t = 45$ for $q_2 \rightarrow q_3$.

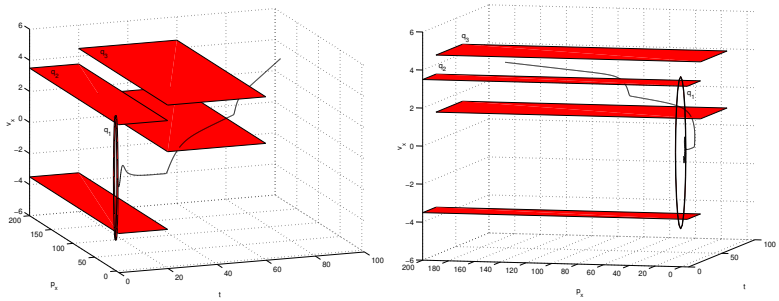


Fig. 5. Projected trajectories of the helicopter along with the initial sets of the next control modes from different view angles are shown. Notice that immediate transition $q_2 \rightarrow q_3$ after $q_1 \rightarrow q_2$ is not allowed until $x(t)$ enters the initial set $S_3(r_3)$.

6 Hybrid and Embedded System Models

Our approach presented in previous sections consists of extracting a finite graph which refines the original collections of modes, but is consistent with the physical system, in the sense that high level design has feasible implementation. A three-level system architecture for implementing the approach is proposed and it is depicted in Figure 6.

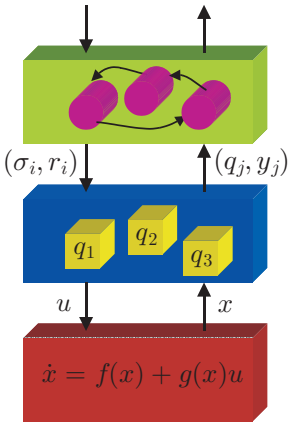


Fig. 6. Hierarchical system architecture for multi-modal control

The consistent control mode graph is contained in the highest level, and it interacts with the lower-level with output events (σ_i, r_i) and input events (q_j, y_j) where symbol $\sigma_i \in \Sigma$ with $|\Sigma| = |Q|$ is introduced for enabling discrete transition towards control mode q_i. The indexes i, j indicates that the automaton is in discrete state q_j and it is going to switch to q_i if the corresponding guard condition is satisfied.

The bottom level contains the continuous dynamics which is represented by differential equations and it interacts with the higher-level with continuous state vector x and continuous input vector u. If we combine the mid-level with the bottom-level together, it can be modeled as a hybrid automaton as shown in Figure 7. In the following, we present a definition of hybrid automata [18] which captures all of the essential features of hybrid automata.

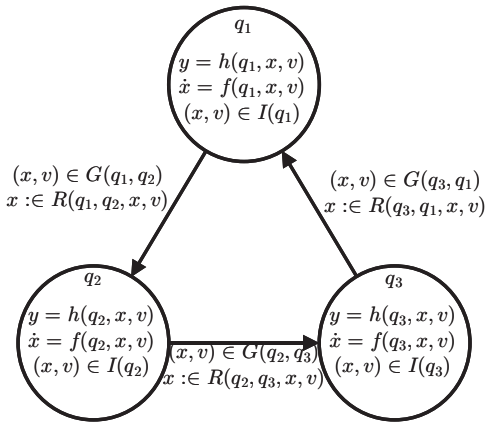


Fig. 7. Hybrid Automaton

Definition 4 (Hybrid Automaton). *A hybrid automaton H is a collection $H = (Q, X, V, Y, Init, f, h, I, E, G, R)$, where*

- *Q is a finite collection of discrete state variables;*
- *X is a finite collection of continuous state variables;*
- *V is a finite collection of continuous input variables;*
- *Y is a finite collection of continuous output variables;*
- *$Init \subseteq Q \times X$ is a set of initial states;*
- *$f : Q \times X \times V \to TX$ is an input dependent smooth vector field;*
- *$h : Q \times X \times V \to 2^Y$ is an input dependent output map;*
- *$I : Q \to 2^{X \times V}$ assigns each $q \in Q$ an input dependent invariant set;*
- *$E \subset Q \times Q$ is a collection of discrete transitions;*
- *$G : E \to 2^{X \times V}$ assigns to each $e = (q, q') \in E$ a guard; and*
- *$R : E \times X \times V \to 2^X$ assigns to each $e = (q, q') \in E$, $x \in X$ and $v \in V$ a reset relation.*

Such systems are essentially a finite state machine with a closed-loop dynamics embedded in each discrete location. The continuous state in a discrete location evolves continuously according to differential equations, as long as the location's invariant remain true; then, when a transition guard becomes true, the discrete state proceeds to another discrete location, and reset some of the continuous state to new values. For the multi-modal control problem, input v is defined as $v = (\sigma, r)$. Consider $e = (q_1, q_2)$, we have $G(q_1, q_2) = S_2(r_2) \times \{r_2\} \times \{\sigma_2\}$, $R(q_1, q_2, x, v) = \{x\}$ and $I(q_2) = X_2$. Therefore, simulation and verification tools developed for hybrid systems can be used to study the behaviors of the multi-modal systems.

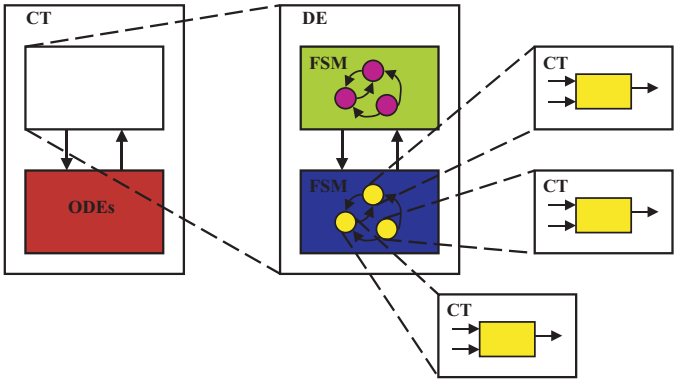


Fig. 8. The multi-modal system is modeled as a hierarchical nesting of parallel and serial composition of discrete and continuous components. At each level of the hierarchy, a Model of Computation (MOC) governs the behaviors and interactions of components

For embedded system implementation, component-based design[16] provides a clean way to integrate different models by hierarchically nesting of parallel and serial composition of heterogeneous components. Depending on the level of abstraction and domain-specificity of design practice, different MOC which govern the interaction among components can be chosen and mixed for composing hybrid systems. The proposed system architecture can be implemented as an embedded system by mixing various MOCs. There are two levels of hierarchy introduced in the design and it is shown in Figure 8. At the first level, in the CT domain the multi-modal controller is modeled as a composite component and the vehicle dynamics is modeled as ordinary differential equations (ODEs). The multi-modal controller is hierarchically refined by two components in the DE domain due to the asynchronous interaction between them. In DE models of computation, an event consists of a value and time stamp. At the second level, the component, on the top, models the mode switching logic and the other component, on the bottom, models the control switches by FSM. In each discrete of the FSM, the component is further refined to model the continuous-time con-

trol laws. Notice that domain DE is chosen due to the asynchronous behaviors exhibited by the models.

7 Conclusion

In this paper, we have focused on formal approach of hybrid system design for multi-modal control. In multi-modal control paradigm, a set of controllers of satisfactory performance have already been designed and must be used.Each controller may be designed for a different set of outputs in order to meet the given performance objectives and system constraints. Our approach consists of extracting a finite graph which refines the original collections of modes, but is consistent with the physical system, in the sense that high level design has feasible implementation. Extracting a finite graph critically depends on the fact the closed loop, output tracking controllers reduce the complexity of the model to the complexity of the output trajectories. Therefore, by choosing simple trajectories, such as constant trajectories, we can solve meaningful problems, such as the multi-modal control of a helicopter based UAV. Given an initial mode and a final mode, if there exist any path connecting the two modes, it shows that the given task is feasible to be solved. Furthermore, each path on the graph basically encodes a specific set of performance criteria. By choosing different paths, different performance can be achieved for executing the same task.

For a complex control system, it is clear that a computational tool is necessary to generate the consistent mode graph. Such a mode switching tool can be used off-line for synthesizing the mode switching logic every time a new mode is designed. The control mode graph can then be used on-line for efficient and dependable real-time mode switching. To enable such functionalities, we have studied the design problem of embedded software for multi-vehicle multi-modal systems. A hierarchical architecture which promote verification is presented in [10] for the construction of embedded systems. Hybrid control design techniques are important for rapid prototyping of system components for real-time embedded systems. Motivated by our design experience on the development of embedded software for our helicopter-based unmanned aerial vehicles, we believe that at the level closest to the environment under control, the embedded software needs to be time-triggered for guaranteed safety; at the higher levels, we advocate asynchronous hybrid controller design.

Acknowledgments

This work is supported by the DARPA SEC grant F33615-98-C-3614.

References

1. R. Alur, T. Henzinger. Modularity for Timed and Hybrid Systems. *CONCUR97: Concurrency Theory,* A. Mazurkiewicz and J. Winkowski, Eds., Lecture Notes in Computer Science 1243, pp. 74-88, Springer-Verlag, 1997.

2. R. Alur, T. Henzinger, G. Lafferriere, and G. J. Pappas. Discrete Abstractions of Hybrid Systems. *Proceedings of the IEEE*, 88(2):971-984, July 2000.
3. E. Asarin, O. Bournez, T. Dang, O. Maler, and A. Pnueli, Effective Synthesis of Switching Controllers for Linear Systems, Proceedings of the IEEE, 88(2):1011-1025.
4. R. W. Brockett. Hybrid Models for Motion Control Systems. Essays in Control: Perspectives in the Theory and Its Applications, H. Trentelman and J. Willems, Eds., pp. 29-53, Birkhäuser, Boston, 1993.
5. A. Chutinan, B.H. Krogh, Verification of polyhedral-invariant hybrid systems using polygonal flow pipe approximations, Hybrid Systems : Computation and Control, Lecture Notes in Computer Science, 1999.
6. J. P. Hespanha, A. Stephen Morse. Switching Between Stabilizing Controllers, Submitted to *Auotmatica*, 2000.
7. A. Isidori, C. I. Byrnes. Output Regulation of Nonlinear systems. In *IEEE Transactions on Automatic Control*, vol.35(2), pp. 131-140, February 1990.
8. I. Kolmanovsky, E. G. Gilbert. Multimode Regulators for Systems with State & Control Constraints and Disturbance Inputs. In *Lecture Notes in Control and Information Sciences 222, Control Using Logic-Based Switching*, A. Stephen Morse(Ed.), pp. 104-117, Springer-Verlag, London, 1997.
9. T. J. Koo, S. Sastry. Differential Flatness Based Full Authority Helicopter Control Design. In *Proceedings of IEEE Conference on Decision and Control*, pp. 1982-87, Phoenix, Arizona, December 1999.
10. T. J. Koo, J. Liebman, C. Ma, and S. Sastry. Hierarchical Approach for Design of Multi-Vehicle Multi-Modal Embedded Software. Embedded Software, Lectures Notes in Computer Science, volume 2211, Springer-Verlag, 2001
11. T. J. Koo, G. J. Pappas, and S. Sastry. Mode Switching Synthesis for Reachability Specifications. Hybrid Systems : Computation and Control, Lecture Notes in Computer Science, volume 2034, Springer-Verlag, 2001.
12. T. J. Koo, B. Sinopoli, A. Sangiovanni-Vincentelli, and S. Sastry. A Formal Approach to Reactive System Design: A UAV Flight Management System Design Example. In *Proceedings of IEEE International symposium on Computer-Aided Control System Design*, Kohala Coast, Hawaii, September 1999.
13. A.B. Kurzhanski, P.Varaiya. Ellipsoidal Techniques for Reachability Analysis. Hybrid Systems : Computation and Control, Lecture Notes in Computer Science, 2000.
14. G. Lafferriere, G.J. Pappas, and S. Yovine. Symbolic Reachability Computations for Families of Linear Vector Fields, Journal of Symbolic Computation, 32(3):231-253, September 2001.
15. E. A. Lee and A. Sangiovanni-Vincentelli. A Framework for Comparing Models of Computation, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 17(12):1217-1229, December 1998.
16. E. A. Lee. Overview of the Ptolemy Project, *Technical Memorandum UCB/ERL*, M01/11, University of California, Berkeley, March 2001.
17. J. Liu, X. Liu, T. J. Koo, B. Sinopoli, S. S. Sastry, and E. A. Lee. Hierarchical Hybrid System Simulation. In *Proceedings of the 38th Conference on Decision and Control*, Phoenix, Arizona. December 1999.
18. J. Lygeros. Hybrid Systems: Modeling, Analysis and Control. Memorandum UCB/ERL M99/34, University of California, Berkeley, Spring 1999.
19. J. Lygeros, C. Tomlin, S. Sastry. Controllers for Reachability Specifications for Hybrid Systems, Automatica, Volume 35, Number 3, March 1999.
20. R. Milner. Communication and Concurrency. Prentice Hall, 1989.
21. G.J. Pappas, G. Lafferriere, and S. Sastry. Hierarchically Consistent Control Systems. IEEE Transactions on Automatic Control, 45(6):1144-1160, June 2000.
22. C. Tomlin, G. Pappas, J. Lygeros, D. Godbole, and S. Sastry. Hybrid Control Models of Next Generation Air Traffic Management, *Hybrid Systems IV*, volume 1273 of Lecture Notes in Computer Science, pages 378-404, Springer Verlag, Berlin, Germany, 1997.
23. S. Sastry. *Nonlinear Systems: Analysis, Stability, and Control*. Springer-Verlag, New York, 1999.
24. P. Varaiya. Smart Cars on Smart Roads: Problems of Control, *IEEE Transactions on Automatic Control*, 38(2):195-207, February 1993.