AA278A Lecture 1: Introduction

Claire J. Tomlin

March 31, 2005

Hybrid systems are dynamical systems with interacting continuous-time dynamics (modeled, for example, by differential equations) and discrete-event dynamics (modeled, for example, by automata). The hybrid behavior arises in many different contexts:

- **Continuous systems with a phased operation:**
  - bouncing ball
  - walking robots
  - biological cell growth and division
- **Continuous systems controlled by discrete logic:** the revolution in digital technology has fueled a need for design techniques that can guarantee safety and performance specifications of *embedded systems*, or systems that couple discrete logic with the analog physical environment.
  - thermostat
  - chemical plants with valves, pumps
  - control modes for complex systems, eg. intelligent cruise control in automobiles, aircraft autopilot modes
- **Coordinating processes:** systems which are comprised of many interacting subsystems (called multiple agent, or multi-agent systems) typically feature continuous controllers to optimize performance of individual agents, and coordination among agents to compete for scarce resources, resolve conflicts, etc.
  - air and ground transportation systems
  - swarms of micro-air vehicles

In such systems, the continuous-time model describes such behavior as the motion of the mechanical systems, linear circuit behavior, chemical reactions; and the discrete-event model describes behavior like collisions in mechanical systems, circuit switching, valves and pumps in chemical plants.

Examples

Example 1: Bouncing Ball (Figure 1) [1]

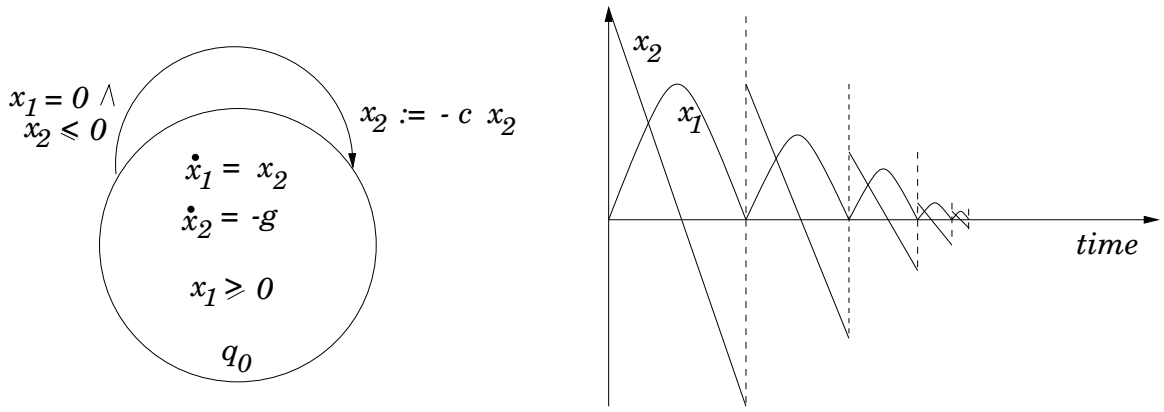


Figure 1: Bouncing ball

- x_1: vertical position of the ball; x_2 velocity of the ball
- g acceleration due to gravity
- $c \in [0, 1]$ coefficient of restitution
- continuous changes between bounces; discrete changes at bounce times
- easy to show the following of the *hybrid automaton* model of the bouncing ball:
  - it is **non-blocking** (meaning from any initial condition there exists at least one state trajectory (solution))
  - $x_1 \geq 0$ is an **invariant** (a property that is always true) of the hybrid automaton
  - if $c < 1$ the hybrid automaton is **Zeno** (takes an infinite number of discrete transitions in finite time)

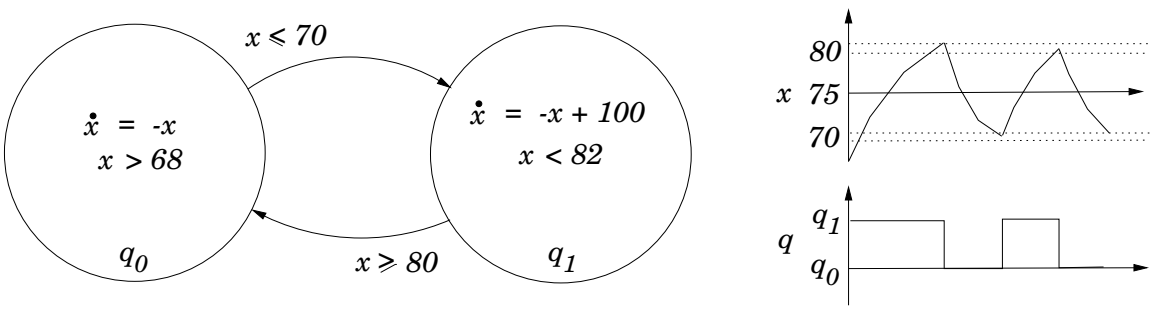


Figure 2: System consisting of a thermostat and the heating of a room

Example 2: Thermostat (Figure 2)

- temperature in a room (x) is controlled by switching a heater on and off
- thermostat regulates x around 75° by turning the radiator on when the temperature is between 68 and 70 and turning the radiator off when the temperature is between 80 and 82
- easy to show that the hybrid automaton model of the thermostat is **non-deterministic**, meaning that for a given initial condition there are a whole family of state trajectories (solutions)

Example 3: Control of an Inverted Pendulum (Figures 3 and 4) [2]

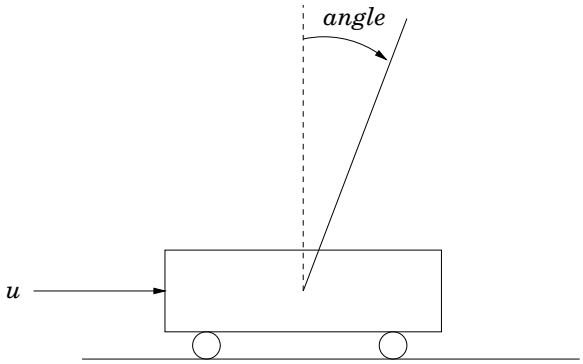


Figure 3: Inverted pendulum on a cart

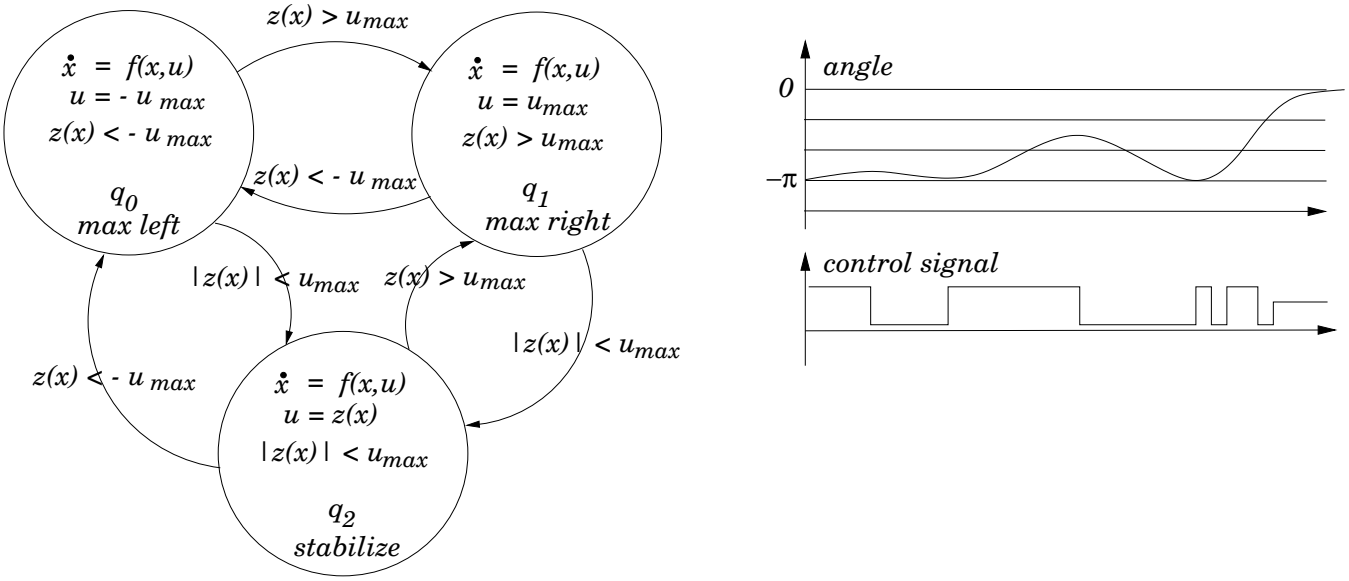


Figure 4: Hybrid control strategy and results for inverted pendulum

- regulate the inverted pendulum to an upright position using the linear acceleration of the pivot as control input u
- $f(x,u) = [x_2, g\sin x_1 - u\cos x_1]^T$, $z(x) = k[x_2^2/2 + g(\cos x_1 - 1)]\text{sgn}(x_2 \sin x_1)$ where x_1 is the pendulum angle and x_2 is the angular velocity
- g is the acceleration due to gravity, $k > 0$

Example 4: Autopilot of a Commercial Jet (Figure 5) [3, 4]

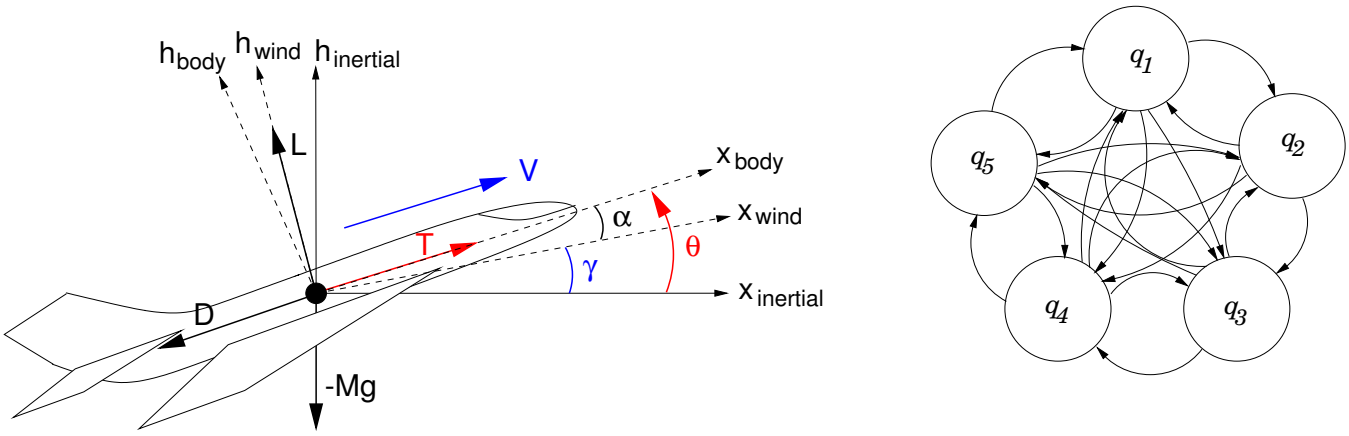


Figure 5: A planar aircraft in flight with attached axes about its center of mass; hybrid automaton of the autopilot.

- Dynamic equations of motion:

$$M \begin{bmatrix} \ddot{x} \\ \ddot{h} \end{bmatrix} = R(\theta) \left[R^T(\alpha) \begin{bmatrix} -D \\ L \end{bmatrix} + \begin{bmatrix} u_1 \\ 0 \end{bmatrix} \right] + \begin{bmatrix} 0 \\ -Mg \end{bmatrix} \tag{1}$$

  where $R(\alpha)$ and $R(\theta)$ are standard rotation matrices, M is the mass of the aircraft, g is gravitational acceleration, control inputs are thrust $u_1 = T$ and pitch $u_2 = \theta$, lift L, drag D

- The *flight path angle* γ and the *angle of attack* α are defined as: $\gamma = \tan^{-1}(\frac{\dot{h}}{\dot{x}})$, $\alpha = \theta - \gamma$; the groundspeed is V

- The system may be discretized into five flight modes, depending on the state variables being controlled:

  - q_1**:** (Speed, Flight Path), in which the thrust T is between its specified operating limits ($T_{min} < T < T_{max}$), the control inputs are T and θ, and the controlled outputs are the speed and the flight path angle of the aircraft $y = (V, \gamma)^T$;
  - q_2**:** (Speed), in which the thrust saturates ($T = T_{min} \vee T = T_{max}$) and thus it is no longer available as a control input; the only input is θ, and the only controlled output is V;
  - q_3**:** (Flight Path), in which the thrust saturates ($T = T_{min} \vee T = T_{max}$); the input is again θ, and the controlled output is γ;
  - q_4**:** (Speed, Altitude), in which the thrust T is between its specified operating limits ($T_{min} < T < T_{max}$), the control inputs are T and θ, and the controlled outputs are the speed and the vertical position of the aircraft $y = (V, h)^T$;
  - q_5**:** (Altitude), in which the thrust saturates $T = T_{min} \vee T = T_{max}$; the input is θ, and the controlled output is h.

Example 5: Automated Highway Systems (Figure 6) [5]

- **Large-scale multi-agent distributed system:** objective is to increase throughput of California's highway system without building new highways, while improving the system safety
- **Challenge 1:** It is unclear whether individual vehicle controllers which minimize travel time of each vehicle leads to maximum throughput of the system
- **Challenge 2:** Controllers which maximize throughput (high speed and close following) are in conflict with controllers designed for safety (low speed and large spacing between vehicles)
- **Solution [5]:** Organize vehicles in tightly spaced groups (called *platoons*) traveling quickly; continuous state for each vehicle (position, velocity, acceleration ...), discrete state for each vehicle (position in the platoon and lane number)
- **Challenge 3:** State explosion as number of vehicles increase
- **Solution:** coordinate vehicles using communication protocols (Figure 6)

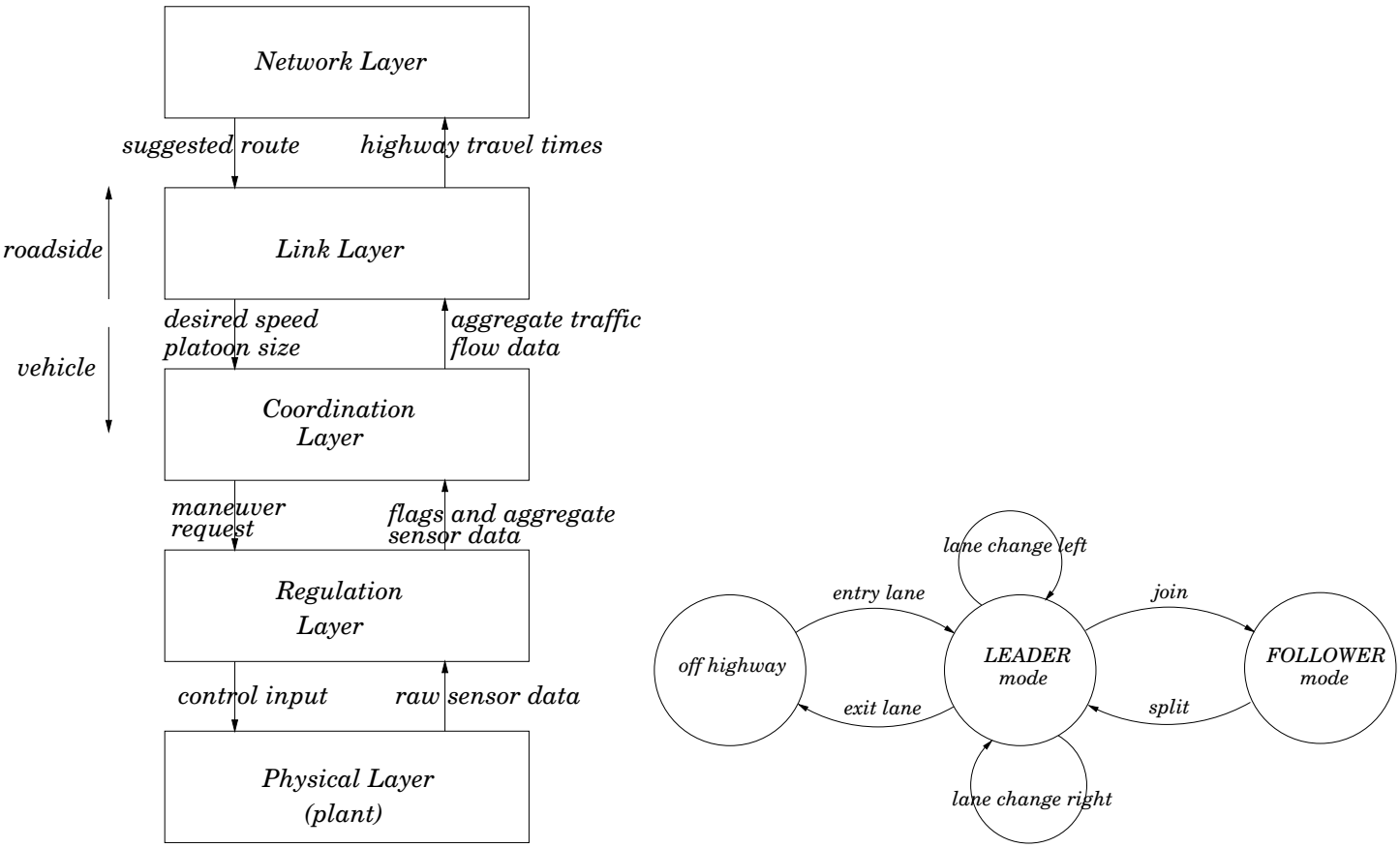


Figure 6: (a) AHS Control Hierarchy, and (b) Discrete state of an AHS vehicle.

Example 6: Free Flight (Figure 7) [6, 7, 8, 9]

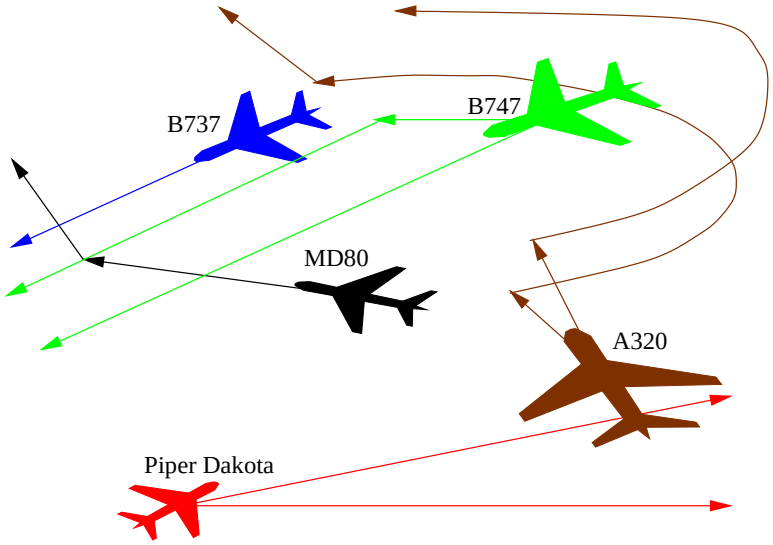
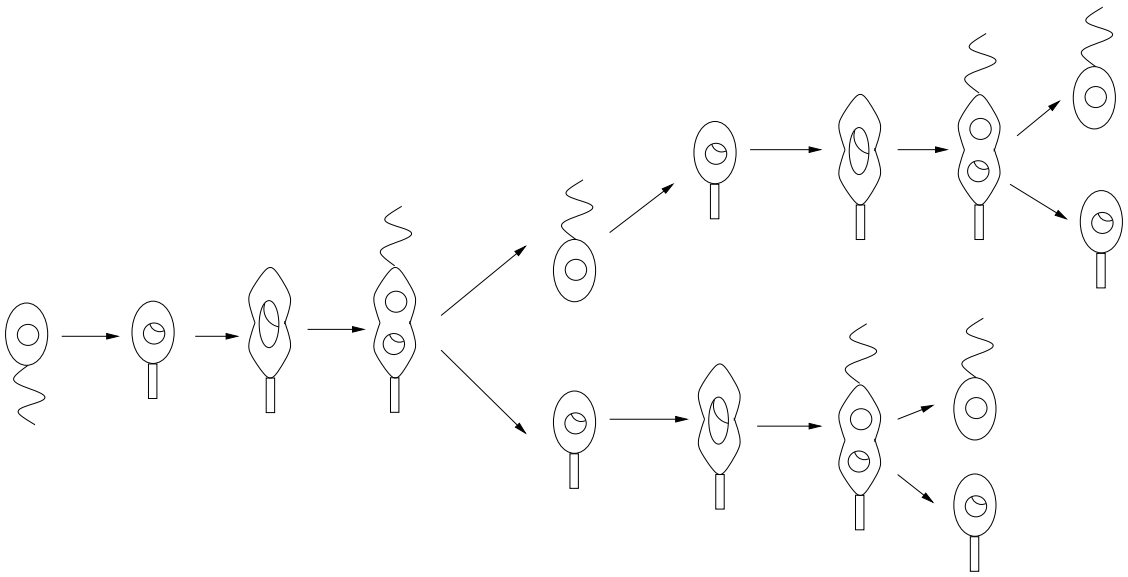


Figure 7: Free flight example

- **Challenge:** Each aircraft flies along a route which ensures that the flight time is short, the fuel consumption is minimized, and inclement weather is avoided, and at the same time will maintain safe distance from other aircraft
- **Solution:** Dynamic route structure and "protocol-based" maneuvers for conflict resolution between aircraft

Example 7: Biological cell growth and division (Figure 8) [10, 11, 12]



from:
H. McAdams and L. Shapiro "Systems Approach to Analysis of Genetic Regulatory Networks"
Stanford University School of Medicine, 2000

Figure 8: Asymmetric cell division in Caulobacter cell.

- The dynamics that govern the spatial and temporal increase or decrease of protein concentration inside a single cell are continuous differential equations derived from biochemistry, yet the activation or deactivation of these continuous dynamics are triggered by discrete switches which encode protein concentrations reaching given thresholds
- Caulobacter cell division: undergoes several mode changes, from a swarmer cell with a flagellum, to an immobile cell with stalk, through an asymmetric division process which creates a new swarmer and a new immobile cell

Hybrid Automaton

We summarize the dynamics of hybrid systems with a model called a *hybrid automaton*, which contains the following entities.

- discrete state $q \in Q$, where Q is a finite collection of variables
- continuous state $x \in \mathbb{R}^n$
- discrete inputs $\sigma \in \Sigma$, where Σ is a finite collection of variables
- continuous inputs $v \in V \subseteq \mathbb{R}^v$
- initial state $Init \subseteq Q \times \mathbb{R}^n$
- continuous dynamics $\dot{x} = f(q, x, v)$
- invariant $Inv \subseteq Q \times \mathbb{R}^n \times \Sigma \times V$ (this defines combinations of states and inputs for which continuous evolution is allowed)
- discrete dynamics $R : Q \times \mathbb{R}^n \times \Sigma \times V \to 2^{Q \times X}$

Example: bouncing ball

- $Q = \{q_0\}$
- $x = (x_1, x_2) \in \mathbb{R}^2$
- $\Sigma = \emptyset$
- $V = \emptyset$
- $Init = \{q_0\} \times \{x \in \mathbb{R}^2 : x_1 \geq 0\}$
- $\dot{x} = f(q_0, x) = [x_2, -g]^T$
- $Inv = \{q_0\} \times \{x \in \mathbb{R}^2 : x_1 \geq 0\}$
- $R(q_0, \{x : x_1 = 0 \wedge x_2 \leq 0\}) = (q_0, (x_1, -cx_2))$ where $c \in [0, 1]$

References

[1] K. J. Johansson, M. Egerstedt, J. Lygeros, and S. Sastry. On the regularization of zeno hybrid automata. *Systems and Control Letters*, 38:141–150, 1999.

[2] K. J. Astrom and K. Furuta. Swinging up a pendulum by energy control. *Automatica*, 1999.

[3] C. J. Tomlin, J. Lygeros, and S. Sastry. A game theoretic approach to controller design for hybrid systems. *Proceedings of the IEEE*, 88(7):949–970, July 2000.

[4] M. Oishi, I. Mitchell, A. M. Bayen, C. J. Tomlin, and A. Degani. Hybrid verification of an interface for an automatic landing. In *Proceedings of the IEEE Conference on Decision and Control*, pages 1607–1613, Las Vegas, NV, December 2002.

[5] Pravin Varaiya. Smart cars on smart roads: problems of control. *IEEE Transactions on Automatic Control*, AC-38(2):195–207, 1993.

[6] C. J. Tomlin, I. Mitchell, and R. Ghosh. Safety verification of conflict resolution maneuvers. *IEEE Transactions on Intelligent Transportation Systems*, 2(2):110–120, 2001. June.

[7] A. M. Bayen, P. Grieder, H. Sipma, C. J. Tomlin, and G. Meyer. Delay predictive models of the National Airspace System using hybrid control theory. In *Proceedings of the American Control Conference*, pages 767–772, Anchorage, AL, May 2002.

[8] I. Hwang and C. J. Tomlin. Multiple aircraft conflict resolution under finite information horizon. In *Proceedings of the American Control Conference*, Anchorage, AL, May 2002.

[9] R. Teo and C. J. Tomlin. Computing danger zones for provably safe closely spaced parallel approaches. *Journal of Guidance, Control, and Dynamics*, 26(3), May–June 2003.

[10] D. Hung, H. H McAdams, and L. Shapiro. Regulation of the caulobacter cell cycle. *Microbial Development*, 1999. ASM Press.

[11] R. Ghosh and C. J. Tomlin. Lateral inhibition through delta-notch signaling: a piecewise affine hybrid model. In M. D. Di Benedetto and A. Sangiovanni-Vincentelli, editors, *Hybrid Systems: Computation and Control*, LNCS 2034, pages 232–246. Springer Verlag, 2001.

[12] K. Amonlirdviman, N. A. Khare, D. R. P. Tree, W.-S. Chen, J. D. Axelrod, and C. J. Tomlin. Mathematical modeling of planar cell polarity to understand domineering nonautonomy. *Science*, 307(5708):423–426, January 2005.

AA278A Lecture Notes 2
Spring 2005
Mathematical Background: Discrete and Continuous Systems

Claire J. Tomlin

April 4, 2005

The formal definitions of hybrid systems build on a number of concepts from continuous state and discrete state dynamical systems.

A **continuous state dynamical system** in one in which the state x takes on values in $\mathbb{R}^n$, that is $x \in \mathbb{R}^n$. Continuous state systems may evolve in either continuous time (for example, $\dot{x} = f(x, u)$), or discrete time (for example, $x_{k+1} = f(x_k, u_k)$).

A **discrete state dynamical system** is one in which the state q takes on values in a finite set $\{q_1, q_2, q_3, \ldots\}$.

1 Finite Automaton models of discrete state systems

- **Examples:** digital switching circuit, lexical analyzers, digital computer ...
- **Definition [2, 3]:** A *finite automaton* M is a mathematical model of a system represented as

$$(Q, \Sigma, \text{Init}, R) \tag{1}$$

where

  - Q is a finite set of *discrete state variables*
  - $\Sigma = \Sigma_1 \cup \Sigma_2$ is a finite set of *discrete input variables*, where Σ_1 contains the controller's inputs and Σ_2 contains the environment's inputs, which cannot be controlled
  - $\text{Init} \subseteq Q$ is a set of *initial states*
  - $R : Q \times \Sigma \to 2^Q$ is a *transition relation* which maps the state and input space to subsets of the state space and thus describes the transition logic of the finite automaton

- **Definition:** An *execution* of the finite automaton M is defined to be a finite or infinite sequence of states and inputs, written as:

$$(q(\cdot), \sigma(\cdot)) \in (Q \times \Sigma)^\omega \tag{2}$$

where $(\cdot)^\omega$ indicates infinite strings, and for $i \in \mathbb{Z}$:

$$q(0) \in \text{Init and } q(i+1) \in R(q(i), \sigma(i)) \tag{3}$$

- Σ^* is the set of strings of arbitrary length of elements of Σ (we denote by ϵ a string of length 0)
- **Definition:** Consider a finite automaton M with a specified set of final states $F \subseteq Q$. M is said to *accept* a string $s = \sigma(0)\sigma(1)\sigma(2)\ldots\sigma(n) \in \Sigma^*$ if there exists a sequence of states $q = q(0)q(1)q(2)\ldots q(n+1) \in Q^*$ such that:
  - $q(0) \in \text{Init}$

- $q(i+1) \in R(q(i), \sigma(i))$
- $q(n+1) \in F$

- **Definition:** The *language accepted by* M, $L(M)$, is the set of all strings accepted by M

- **Definition:** Two automata, M_1 and M_2, are said to be *equivalent* if $L(M_1) = L(M_2)$

- **Definition:** The finite automaton M may *block* certain input strings if it does not accept certain inputs at certain states

- **Definition:** The finite automaton M may be *non-deterministic*, meaning that it could take different transitions from the same state in response to the same input symbol

- **Remark:** If the transition relation R is a function, that is $|R(q, \sigma)| = 1$ for all $q \in Q$ and all $\sigma \in \Sigma$, then M is deterministic

- **Example:** Consider the finite automaton M in Figure 1. In this case,

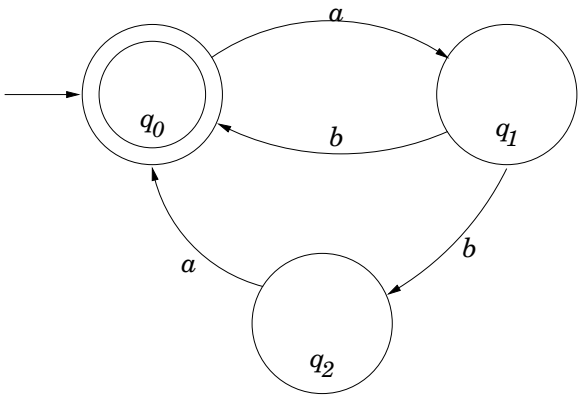


Figure 1: Finite Automaton Example.

  - $Q = \{q_0, q_1, q_2\}$
  - $\Sigma = \{a, b\}$
  - $q_1 = R(q_0, a), q_2 = R(q_1, b), q_0 = R(q_2, a), q_0 = R(q_1, b)$ (note that R is not a function)
  - Init $= q_0$ (indicated by the straight arrow in Figure 1)
  - $F = q_0$ (indicated by a double circle in Figure 1)
  - $L(M) = \{\epsilon, ab, aba, abab, abaaba, \ldots\} = ((ab)^*(aba)^*)^* \subseteq \Sigma^*$
  - non-deterministic

- **Example [2]: Coordinating Finite Automata.** Consider the crossroad traffic controller problem as illustrated in Figure 2. The system consists of three components: Avenue A, Boulevard B, and traffic Controller C. We model each of these components

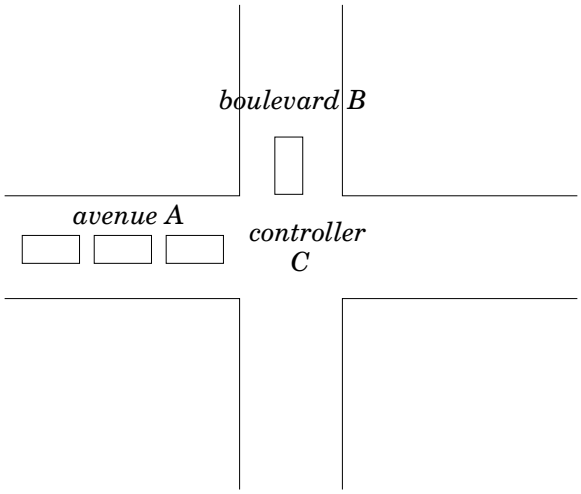


Figure 2: Crossroad Traffic Controller.

as a finite state machine as shown in Figure 3. Each of A, B, and C has 2 states, as shown in Figure 3.

The crossroad system is modeled by the *synchronous composition* of the 3 state machines, written as $M = A \otimes B \otimes C$:

- $Q = \{(stop, stop, go_A), \ldots, (go, go, go_B)\}$ (8 states)
- $\Sigma = \{a_1, a_2, a_3, b_1, b_2, b_3, c_1, c_2, c_3\}$
- The transition relation for M is given in term of Boolean conjunctions of the respective transition relations for the components, for example: $(go, stop, go_A) = R((stop, stop, go_A), a_2 \wedge (b_1 \vee b_2) \wedge c_1)$
- Init $= \{(stop, stop, go_A), (stop, stop, go_B)\}$
- $L(M)$ is the set of finite sequences of Boolean conjunctions and disjunctions of input symbols

Why is this modeling formalism interesting to us?

Because if we accept this model as reality, and we can prove (verify) that the system never transitions into a state in which $(A : cars_{going}) \wedge (B : cars_{going})$, then the system is *safe*. Likewise, if we can verify that system allows all cars on each road to eventually get through the intersection, then the system is *live*.

Consider the safety problem. It is possible to verify the safety of the system by constructing a *monitor automaton* as shown in Figure 4:

T_1 has the initial state *safe*; the self-loop transition *safe* $\rightarrow$ *safe* marked by a "+" is a *recur* edge, meaning that any infinite path through the transition structure of T_1 which crosses edge *safe* $\rightarrow$ *safe* infinitely often designates a behavior accepted by T_1. The *else* indicates that every behavior (of M) is accepted by T_1 unless it involves $(A : cars_{going}) \wedge (B : cars_{going})$, in which case T_1 moves to state *unsafe*, which disallows

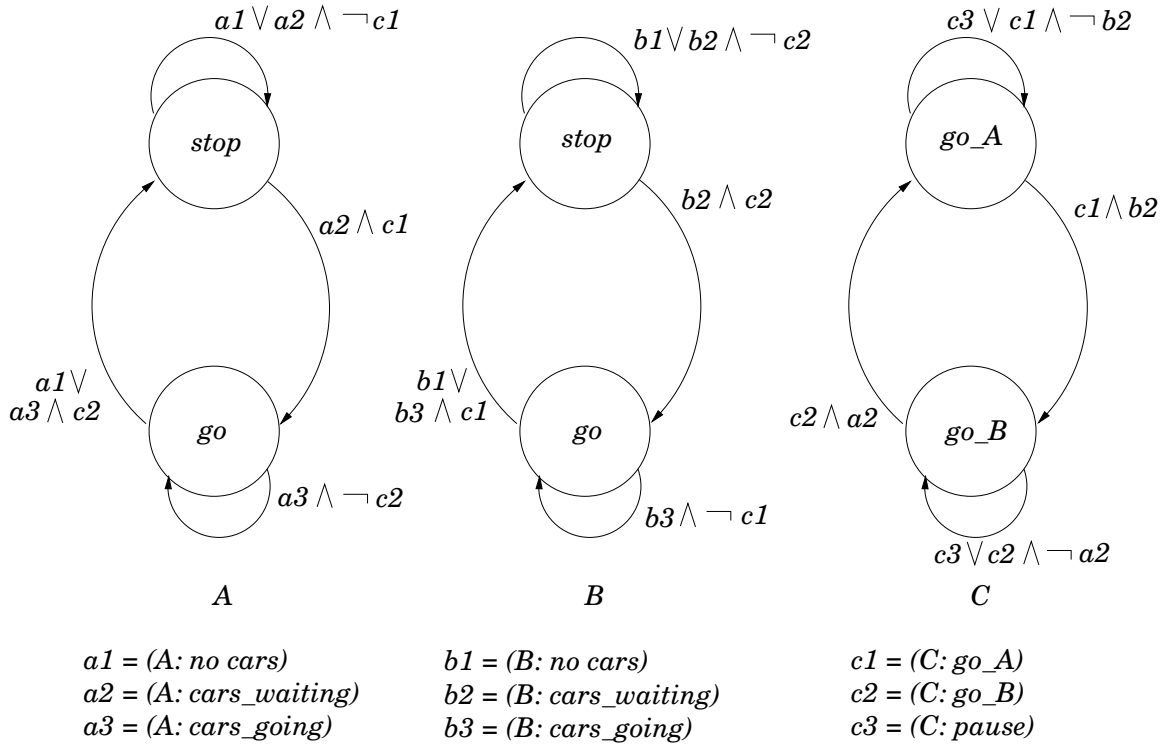


Figure 3: Finite automata models of the crossroad traffic controller example.

an infinite number of recur-edge crossings, and thus disallows the acceptance by T_1 of any behavior which includes that collision event. More formally, the system is safe if

$$L(M) \subset L(T_1) \tag{4}$$

2 Differential Equations

- **Definition:** *Continuous state, continuous time control systems* may be represented as differential equations evolving on a state space X [4, 5]:

$$\dot{x}(t) = f(x(t), v) \tag{5}$$

where

  - $x \in X$ is the state, here $X = R^n$
  - $v \in V = U \times D$ is the space of continuous input variables, where $U = \mathbb{R}^u$ is the set of control inputs and $D = \mathbb{R}^d$ is the set of disturbance inputs
  - $f : \mathbb{R}^n \times U \times D \rightarrow \mathbb{R}^n$ is a vector field, assumed to be Lipschitz continuous in x and piecewise continuous in v
  - the initial state $x(0) \in$ Init where Init $\subseteq X$

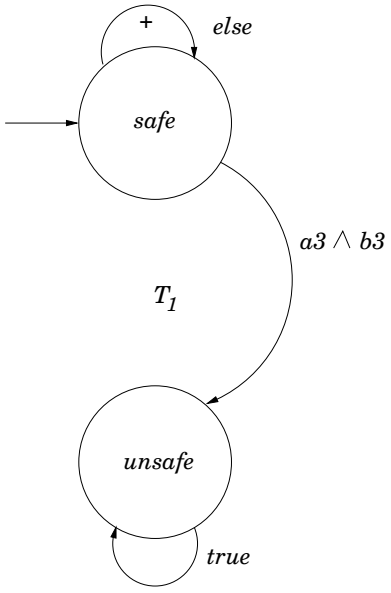


Figure 4: T_1: The monitor automaton which defines the state-invariant task "no cars collide".

- **Definition:** A *trajectory* (execution, solution in the sense of Caratheodory) of (5) over an interval $[\tau, \tau'] \subseteq \mathbb{R}$ is a map: $(x(\cdot), v(\cdot)) : [\tau, \tau'] \rightarrow \mathbb{R}^n \times V$ such that

$$x(\tau') = x(\tau) + \int_{\tau}^{\tau'} f(x(s), v(s))ds \tag{6}$$

Definition 1 (Lipschitz Continuous) *The function $f : \mathbb{R}^n \rightarrow \mathbb{R}^n$ is said to be Lipschitz continuous if there exists a $\lambda > 0$ such that for all $x_1, x_2 \in \mathbb{R}^n$:*

$$||f(x_1) - f(x_2)|| < \lambda ||x_1 - x_2|| \tag{7}$$

If f is Lipschitz continuous in x, then it is continuous is x. The converse is not necessarily true; however, if f has bounded partial derivative in x, then it is Lipschitz continuous.

- **Theorem 1: Local Existence and Uniqueness.** Assume $f(x, v)$ is piecewise continuous in v and Lipschitz continuous in x, for $x \in \{x \in \mathbb{R}^n :\| x - x_0 \| \leq r, r > 0\}$ and for $||f(x_0)|| \leq h, h \geq 0$. Then there exists $\delta > 0$ such the system (5) has a unique solution (6) on $[0, \delta]$.

- **Theorem 2: Global Existence and Uniqueness.** Assume $f(x, v)$ is piecewise continuous in v and Lipschitz continuous in x, and that $\| f(x_0) \| < h$ for $h > 0$, $x \in \mathbb{R}^n$. Then the system (5) has a unique solution (6) $\forall t$.

- **Examples:** Consider systems which violate some of the conditions:

  - f discontinuous in x: $\dot{x} = -\text{sgn}(x)$. The solution starting at $x_0 = 0$ is undefined for all $t > 0$.

- f continuous but not Lipschitz in x: $\dot{x} = x^{1/3}$. Solution is not unique ($x(t) = (2t/3)^{3/2}, x(t) = 0$ are both valid solutions for $x_0 = 0$)
- f Lipschitz but not globally Lipschitz in x: $\dot{x} = -x^2$. The solution for $x_0 = -1$ is $x(t) = 1/(t-1)$ and is not defined when $t = 1$.

- **Theorem 3: Continuous dependence on initial conditions.** Assume $f(x, v)$ satisfies the conditions of Theorem 2, and let x and y be two solutions of (5) starting at x_0 and y_0 respectively. Then for all $\epsilon > 0$ there exists $\delta(\epsilon, T) > 0$ such that

$$\| x_0 - y_0 \| \leq \delta \Rightarrow \| x(t) - y(t) \| \leq \epsilon \tag{8}$$

  for all $t \in [0, T]$, T finite.

- **Remarks:** These theorems provide conditions for a continuous system to be *non-blocking* (solutions exist locally), *deterministic* (solutions are unique), *non-Zeno* (solutions can be extended over arbitrarily long horizons).

References

[1] J. R. Munkres. *Topology: a first course*. Prentice-Hall, Englewood Cliffs, 1975.

[2] R. P. Kurshan. *Computer Aided Verification of Coordinating Processes: The Automata Theoretic Approach*. Princeton University Press, Princeton, N. J., 1994.

[3] J. E. Hopcroft and J. D. Ullman. *Introduction to Automata Theory, Languages, and Computation*. Addison-Wesley, Menlo Park, 1979.

[4] S. S. Sastry. *Nonlinear Systems: Analysis, Stability and Control*. Springer Verlag, 1999.

[5] J. Lygeros, S. S. Sastry, and students. Lecture notes for eecs291e (hybrid systems). UCB/ERL M99/34, University of California at Berkeley, 1999.

AA278A Lecture Notes 3
Spring 2005
Autonomous Hybrid Automata

Claire J. Tomlin

April 5, 2005

We develop definitions to formalize the notion of hybrid systems as dynamical systems that describe the evolution in time of the values of a set of discrete and continuous variables. In this chapter, we restrict our attention to autonomous hybrid automata, that is time invariant hybrid systems without inputs. In subsequent chapters we will extend this class of hybrid models and introduce input variables to allow us to state and solve control problems.

1 Hybrid Time Sets and Trajectories

Since we are interested in hybrid phenomena that involve both continuous and discrete dynamics, we introduce the hybrid time trajectory, which will encode the set of times over which the evolution of the system is defined. Hybrid time sets will be used to define the time horizon of executions of hybrid systems.

Definition 1 (Hybrid Time Set) *A hybrid time set is a finite or infinite sequence of intervals $\tau = \{I_i\}_{i=0}^{N}$, such that*

- *$I_i = [\tau_i, \tau_i']$ for all $i < N$;*
- *if $N < \infty$ then either $I_N = [\tau_N, \tau_N']$ or $I_N = [\tau_N, \tau_N')$; and*
- *$\tau_i \leq \tau_i' = \tau_{i+1}$ for all i.*

An example of a hybrid time trajectory is given in Figure 1. We will use $\mathcal{T}$ to denote the set of all hybrid time sets.

Consider $\tau = \{I_i\}_{i=0}^{N} \in \mathcal{T}$. Notice that, for $i < N$, the right endpoint, τ_i', of the interval I_i coincides with the left endpoint, τ_{i+1} of the interval I_{i+1}. The interpretation is that these are the times at which discrete transitions of the hybrid system take place. τ_i' corresponds to the time instant just before a discrete transition, whereas τ_{i+1} corresponds to the time instant just after the discrete transition. Discrete transitions are assumed to be instantaneous, therefore $\tau_i' = \tau_{i+1}$. The advantage of this convention is that it allows one to capture situations in which multiple discrete transitions take place one after the other at the same time instant, since it is possible for $\tau_{i-1}' = \tau_i = \tau_i' = \tau_{i+1}$.

For each hybrid time set, $\tau \in \mathcal{T}$, we define a precedence relation $\prec$ on τ. For $t_1 \in [\tau_i, \tau_i'] \in \tau$ and $t_2 \in [\tau_j, \tau_j'] \in \tau$ we say that t_1 *precedes* t_2 (denoted by $t_1 \prec t_2$) if $t_1 < t_2$ or $i < j$. It is easy to show that this relation is a linear order on τ. In Figure 1, we have $t_1 \prec t_2 \prec t_3 \prec t_4 \prec t_5 \prec t_6$.

On the set $\mathcal{T}$ of hybrid time sets we define a *prefix* relation. We say that $\tau = \{I_i\}_{i=0}^{N}$ is a prefix of $\hat{\tau} = \{\hat{I}_i\}_{i=0}^{M}$ (and write $\tau \sqsubseteq \hat{\tau}$) if either they are identical, or τ is finite, $N \leq M$, $I_i = \hat{I}_i$ for all $i = 0, \ldots, N-1$, and $I_N \subseteq \hat{I}_N$. We say that τ is a *strict prefix* of $\hat{\tau}$ (and write $\tau \sqsubset \hat{\tau}$) if $\tau \sqsubseteq \hat{\tau}$ and $\tau \neq \hat{\tau}$. It is easy to show that the prefix relation is a partial order on $\mathcal{T}$. In Figure 2, τ is a strict prefix of both $\hat{\tau}$ and $\tilde{\tau}$, but $\hat{\tau}$ is not a prefix of $\tilde{\tau}$ and $\tilde{\tau}$ is not a prefix of $\hat{\tau}$.

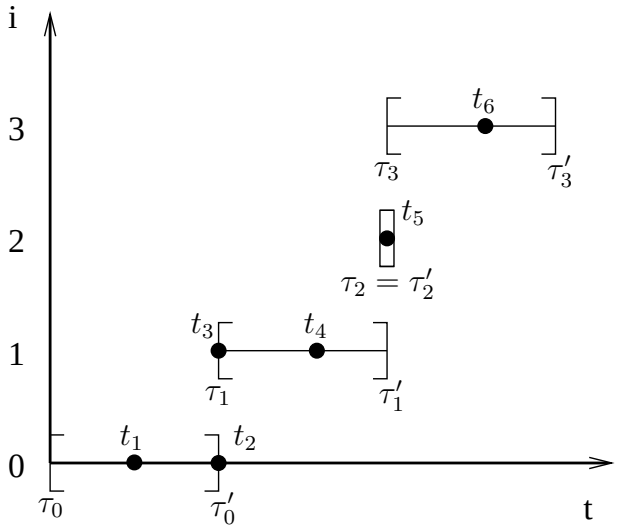


Figure 1: A hybrid time set $\tau = \{[\tau_i, \tau_i']\}_{i=0}^{3}$.

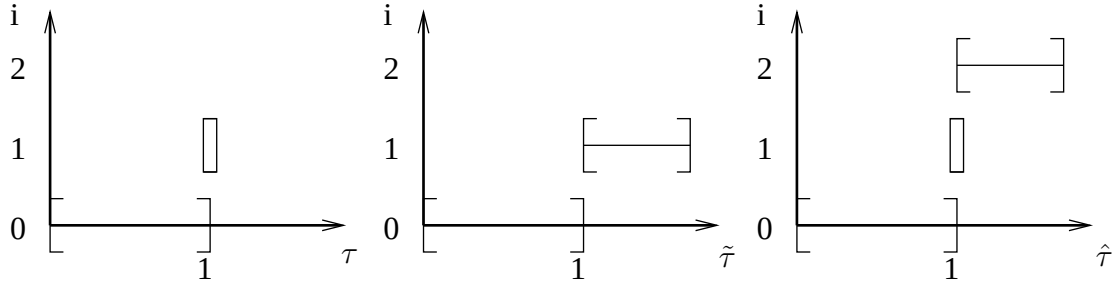


Figure 2: $\tau \sqsubset \hat{\tau}$ and $\tau \sqsubset \tilde{\tau}$.

Since hybrid time sets incorporate both discrete and continuous aspects one can consider two notions of "length" for them. One notion is discrete and reflects the number of intervals contained in the hybrid time set. The *discrete extent* can be defined as a function $\langle \cdot \rangle : \mathcal{T} \to \mathbb{N} \cup \{\infty\}$ that for $\tau \in \mathcal{T}$ returns

$$\langle \tau \rangle = \begin{cases} N & \text{if } \tau \text{ is a finite sequence} \\ \infty & \text{if } \tau \text{ is an infinite sequence} \end{cases}$$

The second notion of length is continuous, and reflects the length of time over which the hybrid time set is defined. More formally, the *continuous extent* can be defined as a function $\| \cdot \| : \mathcal{T} \to \mathbb{R}_+$ that for $\tau = \{I_i\}_{i=0}^{N} \in \mathcal{T}$ returns

$$\|\tau\| = \sum_{i=0}^{N} (\tau_i' - \tau_i)$$

Clearly, if $\tau \sqsubseteq \hat{\tau}$, $\langle \tau \rangle \leq \langle \hat{\tau} \rangle$ and $\|\tau\| \leq \|\hat{\tau}\|$.

Definition 2 (Classification of Hybrid Time Sets) *A hybrid time set $\tau \in \mathcal{T}$ is called*

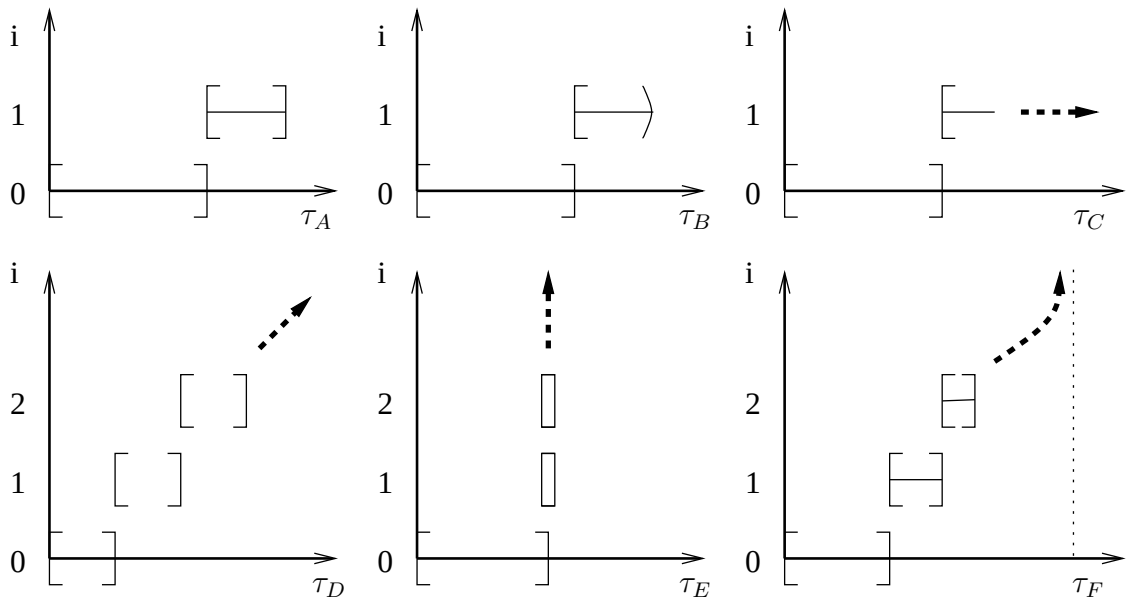


Figure 3: τ_A finite, τ_B finite-open, τ_C and τ_D infinite, τ_E and τ_F Zeno.

- finite*, if $\langle\tau\rangle$ is finite and the last interval in τ is closed;*
- finite-open*, if $\langle\tau\rangle$ is finite, and the last interval in τ is bounded and right open;*
- infinite*, if $\langle\tau\rangle = \infty$, or if $\|\tau\| = \infty$;*
- Zeno*, if it is infinite but $\|\tau\| < \infty$.*

Figure 3 shows examples of finite, finite open, infinite and Zeno hybrid time sets

Definition 3 (Hybrid Trajectory) *A hybrid trajectory (τ, q, x) consists of a hybrid time set $\tau = \{I_i\}_0^N \in \mathcal{T}$ and two sequences of functions $q = \{q_i(\cdot)\}_0^N$ and $x = \{x_i(\cdot)\}_0^N$ with $q_i(\cdot) : I_i \to Q$ and $x(\cdot) : I_i \to \mathbb{R}^n$.*

2 Autonomous Hybrid Automata

2.1 Fundamental Definitions

Definition 4 (Autonomous Hybrid Automaton) *An autonomous hybrid automaton H is a collection $H = (Q, X, \text{Init}, f, \text{Dom}, R)$, where*

- $Q = \{q_1, q_2, \ldots\}$ *is a set of* discrete states*;*
- $X = \mathbb{R}^n$ *is a set of* continuous states*;*
- $\text{Init} \subseteq Q \times X$ *is a set of* initial states*;*

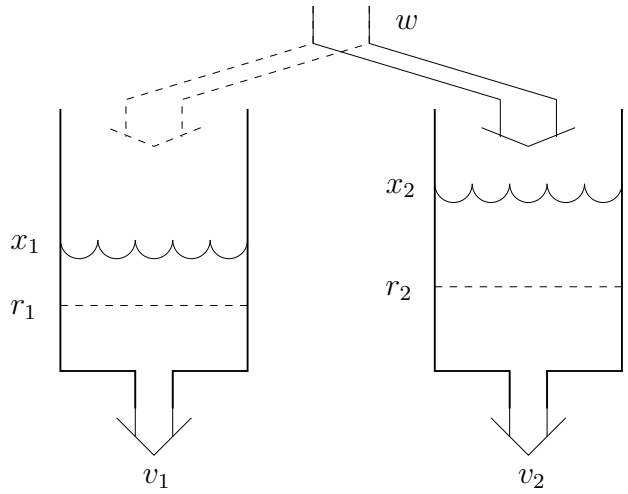


Figure 4: The water tank system.

- $f(\cdot,\cdot) : Q \times X \to \mathbb{R}^n$ *is a* vector field*;*
- $\mathrm{Dom}(\cdot) : Q \to 2^X$ *is a* domain*;*
- $R(\cdot,\cdot) : Q \times X \to 2^X$ *is a* reset relation.

Recall that 2^X denotes the power set (set of all subsets) of X. We refer to $(q, x) \in Q \times X$ as the state of H.

Roughly speaking, autonomous hybrid automata define possible ways that their state can evolve over time. These possible evolutions take the form of hybrid trajectories, (τ, q, x). Starting from an initial value $(q_0, x_0) \in \mathrm{Init}$, the continuous state, x, flows according to the differential equation $\dot{x} = f(q_0, x), x(0) = x_0$, while the discrete state, q, remains constant at q_0. Continuous evolution can go on as long as $x(t) \in \mathrm{Dom}(q_0)$. If at some point $R(q_0, x) \neq \emptyset$ a discrete transition can take place. During a discrete transition both the continuous and the discrete state may be reset to some value in $R(q_0, x)$. After the discrete transition, continuous evolution resumes and the whole process is repeated.

Before we give the formal definition of this process we give an example to illustrate how autonomous hybrid automata can be used to model physical systems.

Example (Water Tank) The two tank system, shown in Figure 4, consists of two tanks containing water. Both tanks are leaking at a constant rate. Water is added at a constant rate to the system through a hose, which at any point in time is dedicated to either one tank or the other. It is assumed that the hose can switch between the tanks instantaneously.

For $i \in \{1, 2\}$, let x_i denote the volume of water in Tank i and $v_i > 0$ denote the constant flow of water out of Tank i. Let w denote the constant flow of water into the system. The objective is to keep the water volumes above r_1 and r_2, respectively, assuming that the water volumes are above r_1 and r_2 initially. This is to be achieved by a controller that switches the inflow to Tank 1 whenever $x_1 \leq r_1$ and to Tank 2 whenever $x_2 \leq r_2$.

It is straightforward to define an autonomous hybrid automaton to describe this process:

- $Q = \{q_1, q_2\}$;
- $X = \mathbb{R}^2$;
- $Init = Q \times \{x \in \mathbb{R}^2 \mid x_1 \geq r_1 \wedge x_2 \geq r_2\}$;
- $f(q_1, x) = (w - v_1, -v_2)$ and $f(q_2, x) = (-v_1, w - v_2)$;
- $\mathrm{Dom}(q_1) = \{x \in \mathbb{R}^2 \mid x_2 \geq r_2\}$ and $\mathrm{Dom}(q_2) = \{x \in \mathbb{R}^2 \mid x_1 \geq r_1\}$;
- $R(q_1, x) = (q_2, x)$ if $x_2 \leq r_2$, $R(q_2, x) = (q_1, x)$ if $x_1 \leq r_1$ and $R(q, x) = \emptyset$ otherwise.

Though very simple, the water tank hybrid automaton has a number of interesting properties, and will be used throughout this section (and beyond) to illustrate various concepts. ■

An execution of an autonomous hybrid automaton is a hybrid trajectory, (τ, q, x), over its state variables. The elements listed in Definition 4 impose restrictions on the elements of the set of possible trajectories that the hybrid automaton finds "acceptable".

Definition 5 (Execution) *An execution of a hybrid automaton H is a hybrid trajectory, (τ, q, x), which satisfies the following conditions:*

- initial condition*:* $(q_0, x_0) \in \mathrm{Init}$
- discrete evolution*:* $(q_{i+1}(\tau_{i+1}), x_{i+1}(\tau_{i+1})) \in R(q_i(\tau_i'), x_i(\tau_i'))$*;*
- continuous evolution*: for all i,*
  1. *$q_i(\cdot) : I_i \to Q$ is constant over $t \in I_i$, that is, $q_i(t) = q_i(\tau_i)$ for all $t \in I_i$;*
  2. *$x_i(\cdot) : I_i \to X$ is the solution to the differential equation $\dot{x}_i = f(q_i(t), x_i(t))$ over I_i starting at $x_i(\tau_i)$; and,*
  3. *for all $t \in [\tau_i, \tau_i')$, $x_i(t) \in \mathrm{Dom}(q_i(t))$.*

Definition 5 specifies which subset of the hybrid trajectories over (Q, X) are executions of H by imposing a number of restrictions. The first restriction dictates that the executions should start at an acceptable initial state in Init. The second restriction determines when discrete transitions may take place and what the state after discrete transitions may be. The requirement is that the state after a discrete transition is related to the state before a discrete transition through the reset relation R. In this context, it is convenient to think of R as *enabling* discrete transitions: the execution *may* take a transition from a state s as long as $R(s) \neq \emptyset$. The third restriction determines what happens along continuous evolution, and when continuous evolution must give way to a discrete transition. The first part implies that along continuous evolution the discrete state remains constant. The second part requires that along continuous evolution the continuous state flows according to the vector field f. The third part requires that along continuous evolution the state must remain in the domain,

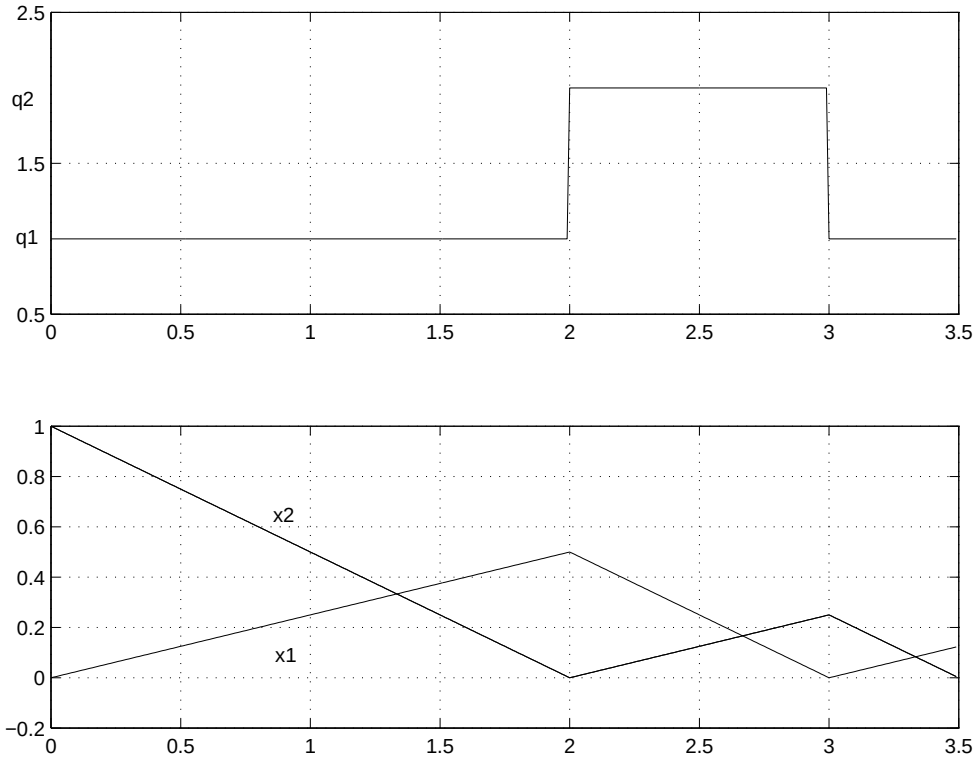


Figure 5: Example of an execution of the water tank hybrid automaton.

Dom. In this context, it is convenient to think of Dom as *forcing* discrete transitions: the execution *must* take a transition if the state is about to leave the domain. The definition is somewhat subtle here. Notice that the execution is required to remain in the domain throughput continuous evolution, *except* at the time instant just before a discrete transition. This assumption provides some additional generality, since one can define hybrid automata whose executions leave the domain for an instant. It will prove useful when studying the fundamental properties of hybrid automata in subsequent sections; for example, it will allow us to deal with hybrid automata whose domain is an open set.

Example (Water Tank Automaton) Figure 5 shows a finite execution of the water tank automaton. The hybrid time set τ of the execution in the figure consists of three intervals, $\tau = \{[0, 2], [2, 3], [3, 3.5]\}$. The evolution of the discrete state is shown in the upper plot, and the evolution of the continuous state is shown in the lower plot. The values chosen for the constants are $r_1 = r_2 = 0$, $v_1 = v_2 = 1/2$ and $w = 3/4$. The initial state is $(q_1, 0, 1)$. ∎

To simplify the notation, we will use $s_0 = s(\tau_0)$ to denote the initial state of an execution (τ, s). Since we restrict attention to autonomous hybrid automata, we can assume that $\tau_0 = 0$, without loss of generality.

Unlike continuous dynamical systems, the interpretation is that a hybrid automaton *accepts* (as opposed to generates) an execution. This difference in perspective allows one to consider, for example, hybrid automata that accept multiple executions for some initial states, a

property that can prove very useful when modeling uncertain systems, as the following example indicates.

Example (Thermostat) The thermostat system models the temperature variation in a room. The temperature is regulated by a thermostat, which switches on a heater when the temperature gets too low.

Let $x \in \mathbb{R}$ denote the temperature in the room. The heater is in one of two states, either on, or off. It is assumed that when the heater is on the temperature in the room rises towards 100 degrees according to the differential equation $\dot{x} = -x + 100$. When the heater is off on the other hand, the temperature decreases towards 0 degrees according to the differential equation $\dot{x} = -x$. The goal of the thermostat is to keep the temperature close to 75 degrees. To accomplice this it switches the heater on if the temperature reaches 70 degrees and off if the temperature reaches 80 degrees. There is, however some uncertainty in the switching process; the temperature may reach 68 degrees before the heating takes effect, and it may reach 82 degrees before the room begins to cool.

It is straight forward to define an autonomous hybrid automaton, T, to describe this process:

- $Q = \{\text{on}, \text{off}\}$;
- $X = \mathbb{R}$;
- $\text{Init} = X$;
- $f(\text{on}, x) = -x + 100$ and $f(\text{off}, x) = -x$;
- $\text{Dom}(\text{on}) = \{x \in \mathbf{X}_T \mid x \leq 82\}\text{Dom}(\text{off}) = \{x \in \mathbf{X}_T \mid x \geq 68\}$;
- $R(\text{on}, x) = (\text{off}, x)$ if $x \geq 80$, $R(\text{off}, x) = (\text{on}, x)$ if $x \leq 70$ and $R(q, x) = \emptyset$ otherwise.

It is easy to see that T accepts a whole family of executions for each initial condition. Some examples are given in Figure 6. ∎

2.2 Graphical Representation

It is often convenient to represent hybrid automata as directed graphs, $(\mathbf{Q}, E)$. The vertices, $\mathbf{Q}$, of the graph correspond to values of the discrete state, while the edges, $E \subseteq \mathbf{Q} \times \mathbf{Q}$, represent possible transitions between the discrete states,

$$E = \{(q, q') \in \mathbf{Q} \times \mathbf{Q} \mid (q', x') \in R(q, x) \text{ for some } x, x' \in \mathbf{X}\}.$$

With each vertex $q \in \mathbf{Q}$, we associate a set of continuous initial states

$$\text{Init}_q = \{x \in \mathbf{X} \mid (q, x) \in \text{Init}\} \subseteq \mathbf{X},$$

a differential equation, $f_q : \mathbf{X} \to 2^{\mathbf{X}}$, given by

$$F_q(x) = F(q, x),$$

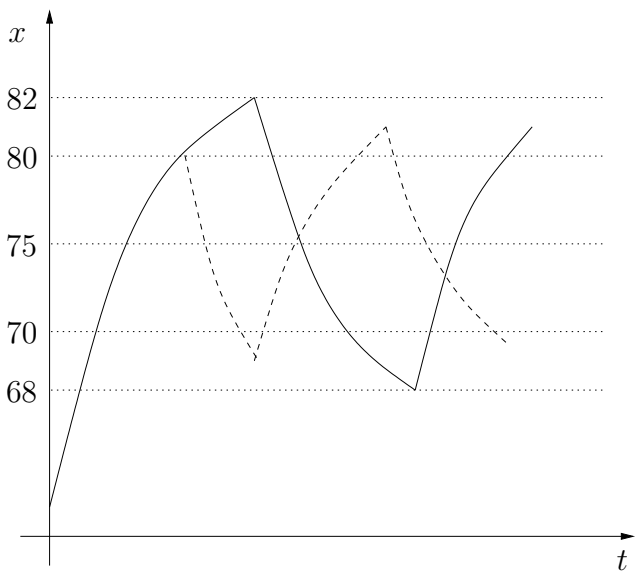


Figure 6: Examples of executions of the thermostat hybrid automaton.

and a domain,

$$\mathrm{Dom}_q = \{x \in \mathbf{X} \mid (q, x) \in \mathrm{Dom}\} \subseteq \mathbf{X}.$$

With each edge, $(q, q') \in E$, we associate a guard,

$$\mathrm{Guard}_{(q,q')} = \{x \in \mathbf{X} \mid (q', x') \in R(q, x) \text{ for some } x' \in \mathbf{X}\} \subseteq \mathbf{X},$$

and a reset relation, $\mathrm{Reset}_{(q,q')} : \mathbf{X} \to 2^{\mathbf{X}}$, given by

$$\mathrm{Reset}_{(q,q')}(x) = \{x' \in \mathbf{X} \mid (q', x') \in R(q, x)\}.$$

Example (Water Tank and Thermostat (continued)) The thermostat and the water tank automata can be represented by the directed graphs of Figure 7a and Figure 7b respectively. In each vertex of the graph, we specify the value of the discrete state, a differential equation (implied by the vector field), and the domain. We write the continuous part of the initial state on an arrow pointing to the vertex corresponding to the discrete part of the initial state. The edges are represented by arrows pointing from the starting state to the destination state. The guard is written near the beginning of the arrow, and the reset near the end. Assignment is denoted by := (deterministic assignment) or :∈ (non-deterministic assignment). ■

To simplify the figures, the reset will be suppressed for edges $(q, q') \in E$ with $\mathrm{Reset}_{(q,q')}(x) = \{x\}$ (as in the above examples). Likewise, the initial state arrow is suppressed for nodes $q \in \mathbf{Q}$ with $\mathrm{Init}_q = \emptyset$.

Notice that the graph associated with a hybrid automaton contains the same information as the definition of the automaton itself. The graphs, therefore, can also be treated as formal definitions of hybrid automata. We will take advantage of this fact in the examples

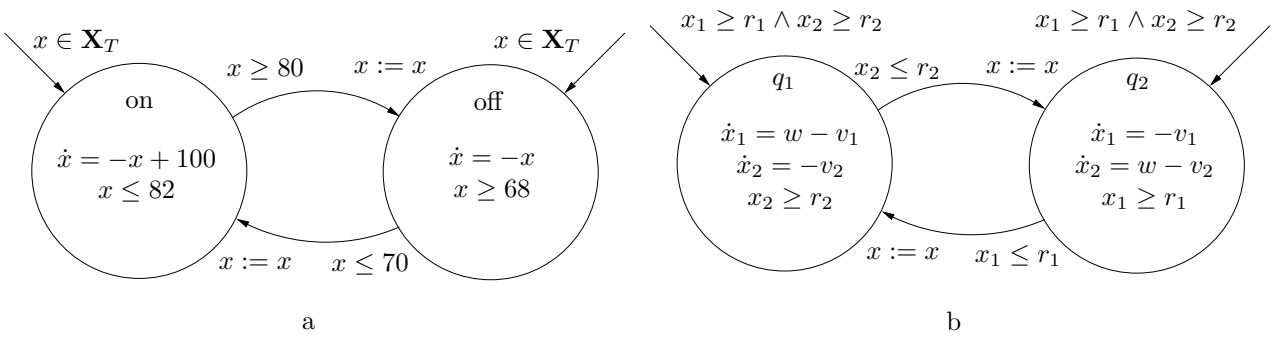


Figure 7: Graphical representation of the thermostat and water tank hybrid automata.

introduced in subsequent sections and use the more intuitive graphical representation to replace the formal (but occasionally cumbersome) definitions. In fact, graphs with guards, reset relations, etc. are commonly used in the hybrid systems literature as fundamental objects of modeling frameworks.

A subtle point here is that the correspondence of graphs to hybrid automata is not unique. The graphical representation may contain redundant elements. For example, one can define an edge $(q, q') \in E$ with $\text{Guard}_{(q,q')} = \emptyset$, or allow $\text{Reset}_{(q,q')}(x) = \emptyset$ for some $x \in \text{Guard}_{(q,q')}$. Situations like these are impossible in graphs generated from autonomous hybrid automata. Such graphs have the property that for all $(q, q') \in E$, $\text{Guard}_{(q,q')} \neq \emptyset$ and $\text{Reset}_{(q,q')}(x) \neq \emptyset$ for all $x \in \text{Guard}_{(q,q')}$. Graphs with these properties can be thought of as some form of "canonical" representations.

References

[1] A. F. Filippov. *Differential equations with discontinuous right hand sides*. Kluwer Academic Publishers, Boston, 1988.

[2] V. I. Utkin. *Sliding Modes in Control Optimization*. Springer Verlag, Berlin, 1992.

[3] K. J. Johansson, M. Egerstedt, J. Lygeros, and S. Sastry. On the regularization of zeno hybrid automata. *Systems and Control Letters*, 38:141–150, 1999.

[4] R. Alur and T. A. Henzinger. Modularity for timed and hybrid systems. In A. Mazurkiewicz and J. Winkowski, editors, *CONCUR 97: Concurrency Theory*, LNCS 1243, pages 74–88. Springer Verlag, 1997.

[5] M. S. Branicky. Multiple Lyapunov functions and other tools for switched and hybrid systems. *IEEE Transactions on Automatic Control*, 43(4):475–482, 1998.

[6] Claire J. Tomlin. *Hybrid Control of Air Traffic Management Systems*. PhD thesis, Department of Electrical Engineering, University of California, Berkeley, 1998.

AA278A Lecture Notes 4
Spring 2005
Existence of Executions

Claire J. Tomlin

April 7, 2005

1 Reachable States

Reachability is a crucial concept in the study of hybrid systems. Roughly speaking, a state, $(q, x) \in Q \times X$ is called reachable if the hybrid automaton H can "steer" its way to (q, x) while moving along one of its executions. The importance of the concept of reachability is difficult to overstate. In the next section we will show how reachability plays a central role in the derivation of existence and uniqueness conditions for executions. In subsequent sections, reachability will also turn out to be a key concept in the study of safety properties for hybrid systems. In this section we provide some basic definitions and results to motivate subsequent discussion.

Definition 1 (Reachable State) *A state $(q', x') \in Q \times X$ of an autonomous hybrid automaton H is called reachable if there exists a finite execution (τ, q, x) ending in (q', x'), i.e. $\tau = \{[\tau_i, \tau_i']\}_0^N$ with $N < \infty$ and $(q_N(\tau_N'), x_N(\tau_N')) = (q', x')$.*

We use $\text{Reach} \subseteq Q \times X$ to denote the set of all states reachable by H. Clearly, $\text{Init} \subseteq \text{Reach}$ (since we may choose $N = 0$ and $\tau_0' = \tau_0$).

Computing the set of reachable states is in general very difficult. Much of the rest of this course will be devoted to different methods that have been proposed for performing this computation.

Definition 2 (Invariant Set) *Consider an autonomous hybrid automaton H. A set of states $M \subseteq Q \times X$ is called invariant if for all $(q_0, x_0) \in M$, and all (τ, q, x) starting from (q_0, x_0), $(q_i(t), x_i(t)) \in M$ for all $i \leq \langle\tau\rangle$ and $t \in I_i$.*

One requirement imposed on hybrid automata is that the state never leave the domain $Q \times \text{Dom}$. This is often the case when modeling physical systems, where the domain typically encodes hard constraints on the state that should be satisfied along all executions. This requirement can be characterized in terms of the reachable states.

Definition 3 (Domain Preserving) *An autonomous hybrid automaton, H, is called domain preserving if* $\text{Reach} \subseteq Q \times \text{Dom}$.

A simple way to determine whether a hybrid automaton is domain preserving is using induction arguments along its executions: if the initial set of states is in the domain, and the reachable states from these initial states is in the domain, the hybrid automaton is domain preserving.

Example (Water Tank (continued)) Consider again the water tank automaton WT, and assume that $w > \max\{v_1, v_2\}$. Notice that $\text{Init}_{WT} \subseteq \text{Dom}_{WT}$. Here, we can show that WT is domain preserving since Init_{WT} is an invariant set.

Consider an arbitrary initial state $(q_0, x_0) \in \text{Init}_{WT}$ and an arbitrary execution (τ, q, x) starting in this initial state. We argue that $(q_i(t), x_i(t)) \in \text{Init}_{WT}$ for all $i \leq \langle\tau\rangle$, and all

$t \in I_i$ by induction. By assumption, $(q_0, x_0) \in \text{Init}_{WT}$. Assume $(q_i(\tau_i), x_i(\tau_i) \in \text{Init}_{WT}$ for some $i \leq \langle\tau\rangle$. If $\tau_i' > \tau_i$, $(q_i(t), x_i(t)) \in \text{Dom}_{WT}$ for all $t \in [\tau_i, \tau_i')$, by the definition of an execution. The only way the state can leave Init_{WT} along continuous evolution is if $x_1 = r_1$ or $x_2 = r_2$. Assume $q_i(\tau_i) = q_1$ (the case $q_i(\tau_i) = q_1$ is symmetric). Then, since $w > \max\{v_1, v_2\}$, $\dot{x}_1 > 0$. Therefore, the only way the state can leave Init_{WT} is if $x_2 = r_2$. But $x_2 = r_2$ implies a discrete transition takes place, since $\text{Dom}(q_1) = \{x \in \mathbb{R}^2 \mid x_2 \geq r_2\}$ and $\text{Dom}(q_2) = \{x \in \mathbb{R}^2 \mid x_1 \geq r_1\}$ and $\dot{x}_2 = -v_2 < 0$. Therefore, $(q_i(t), x_i(t)) \in \text{Init}_{WT}$ for all $t \in [\tau_i, \tau_i']$. Moreover, $R(q_1, x) = \{(q_2, x)\}$, therefore $(q_i(\tau_{i+1}), x_i(\tau_{i+1}) \in \text{Init}_{WT}$. By induction, Init_{WT} is an invariant set.

■

It is easy to see that the thermostat automaton T is not domain preserving, since the initial states are unconstrained, and therefore $\text{Reach}_T = \text{Init}_T = X_T \supset \text{Dom}_T$.

2 Transition States

Another important concept in the study of existence properties is the set of states from which continuous evolution is impossible, sometimes referred to as the *transition states*.

For $(\hat{q}, \hat{x}) \in Q \times X$ and some $\epsilon > 0$, consider the set of solutions, $x(\cdot) : [0, \epsilon) \to X$ to the differential equation:

$$\frac{dx}{dt} = f(\hat{q}, x) \text{ with } x(0) = \hat{x}. \tag{1}$$

To characterize the states from which continuous evolution is impossible, we define the set $\text{Trans} \subseteq Q \times X$ by

$$\text{Trans} = \{(\hat{q}, \hat{x}) \in Q \times X \mid \forall x(\cdot)\ \forall \epsilon > 0\ \exists t \in [0, \epsilon) \text{ such that } x(t) \notin \text{Dom}(\hat{q})\}.$$

In words, Trans is the set of states for which continuous evolution along the differential inclusion forces the system to exit the domain instantaneously.

The exact characterization of the set Trans may be quite involved. We conclude the section by giving some suggestions on how this can be done in certain simple cases. We restrict our attention to cases in which the domain is sufficiently smooth. If f is Lipschitz continuous in x, the following is an immediate consequence of the existence of solutions of ordinary differential equations.

Proposition 4 *If an autonomous hybrid automaton is such that f is locally Lipschitz continuous in x, then* $\text{Trans} \cap \overline{Q \times \text{Dom}} \subseteq \partial(q \times \text{Dom})$.

Here, the notation ∂ refers to the boundaries of the domains in each state q. The simplest case is the one in which the domain is an open set.

Proposition 5 *Consider an autonomous hybrid automaton. If $Q \times \text{Dom}$ is open and f is locally Lipschitz continuous in x, then* $\text{Trans} = (Q \times \text{Dom})^c$.

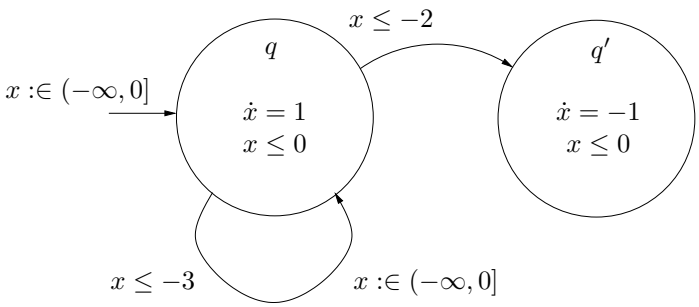


Figure 1: Examples of blocking and non-determinism.

3 Local Existence and Uniqueness

Next, we turn our attention to questions of existence of executions. We derive conditions under which all executions can be extended to infinite executions and conditions under which this extension can be done uniquely. An execution, (τ, q, x), of a hybrid automaton H is called *maximal* if it is not a strict prefix of any other execution of H.

Definition 6 (Non-Blocking and Deterministic) *An autonomous hybrid automaton H is called non-blocking if for all initial states $(q_0, x_0) \in \text{Init}$, there exists an infinite execution starting at (q_0, x_0). It is called deterministic if for all initial states $(q_0, x_0) \in \text{Init}$ there exists at most one maximal execution starting at (q_0, x_0).*

Roughly speaking, the non-blocking property implies that executions exist for all initial states, while the deterministic property implies that the infinite executions (if they exist) are unique. In continuous dynamical systems the existence property is satisfied when the vector field is continuous, while the uniqueness property is satisfied when it is locally Lipschitz continuous. In hybrid systems, however, more things can go wrong. Consider for example the autonomous hybrid automaton of Figure 1. Let (q_0, x_0) denote the initial state, and notice that $q_0 = q$. If $x_0 = -3$, executions starting at (q_0, x_0) can either flow along the vector field, or jump back to q resetting x anywhere in $(-\infty, 0]$, or jump to q' leaving x unchanged. If $x_0 = -2$ executions starting at (q_0, x_0) can either flow along the vector field, or jump to q'. If $x_0 = -1$ executions stating at (q_0, x_0) can only flow along the vector field. Finally, if $x_0 = 0$ there are no executions starting at (q_0, x_0), other than the trivial execution defined over $[\tau_0, \tau_0']$ with $\tau_0 = \tau_0'$. Therefore, the hybrid automaton of Figure 1 accepts no infinite executions for some initial states and multiple infinite executions for others.

Intuitively, a hybrid automaton is non-blocking if for all reachable states for which continuous evolution is impossible a discrete transition is possible. This fact is stated more formally in the following lemma.

Lemma 7 *H is non-blocking if for all $(q, x) \in \text{Trans} \cap \text{Reach}$, there exists $q' \in Q$ such that $(q, q') \in E$ and $x \in G(q, q')$. If H is deterministic, then it is non-blocking if and only if this condition holds.*

In the case in which H is non-deterministic the conditions of Lemma 7 are not necessary. The reason is that the conditions ensure that a blocking execution exists for some initial conditions, but are not sufficient to show that for some initial conditions all executions will block. For example, in the system of Figure 1 some executions starting at $s_0 = (q, 2)$ are cannot be extended to infinite executions (the ones that start by flowing) while others can (the ones that start by a transition to q').

Intuitively, a hybrid automaton may be non-deterministic if either there is a choice between continuous evolution and discrete transition, or if a discrete transition can lead to multiple destinations. More specifically, the following lemma states that a hybrid automaton is deterministic if and only if (1) each discrete transition has a unique destination, and (2) whenever a discrete transition is possible continuous evolution is impossible.

Lemma 8 *An autonomous hybrid automaton H is deterministic if and only if for all $(q, x) \in$ Reach: (1) if $x \in G(q, q')$ for some $(q, q') \in E$, then $(q, x) \in$ Trans; (2) if $(q, q') \in E$ and $(q, q'') \in E$, where $q' \neq q''$, then $x \neq G(q, q') \cap G(q, q'')$; and (3) if $(q, q') \in E$ and $x \in G(q, q')$ then $R(q, x) = (q', x')$ (ie. the set contains a single element).*

Theorem 9 *(Existence and Uniqueness) A hybrid automaton H accepts a unique infinite execution for each initial state if it satisfies all of the conditions of Lemmas 7 and 8.*

Example (Water Tank (continued)) Recall that

$$\begin{aligned}\text{Reach}_{WT} &= \{(q, x) \in \mathbf{Q} \times \mathbf{X} \mid x_1 \geq r_1 \wedge x_2 \geq r_2\},\\ \text{Trans}_{WT} &= \{q_1\} \times \{x \in \mathbf{X} \mid x_2 \leq r_2\} \cup \{q_2\} \times \{x \in \mathbf{X} \mid x_1 \leq r_1\}.\end{aligned}$$

Therefore,

$$\begin{aligned}\text{Reach}_{WT} \cap \text{Trans}_{WT} =& \{q_1\} \times \{x \in \mathbf{X} \mid x_1 \geq r_1 \wedge x_2 = r_2\} \cup\\ & \{q_2\} \times \{x \in \mathbf{X} \mid x_2 \geq r_2 \wedge x_1 = r_1\}.\end{aligned}$$

Notice that $R_{WT}(s) \neq \emptyset$ for all $s \in \text{Reach}_{WT} \cap \text{Trans}_{WT}$, therefore the conditions of Lemma 7 are satisfied. Moreover, for all $s \in \text{Reach}_{WT}$, $R_{WT}(s)$ contains either no elements (if $s \in \text{Trans}_{WT}^c$), or one element (if $s \in \text{Trans}_{WT}$). Therefore, the conditions of Lemma 8 are also satisfied, and the water tank automaton accepts a unique infinite execution for each initial state. ∎

4 Zeno Executions

The conditions of Theorem 9 ensure that a hybrid automaton accepts infinite executions for all initial states. They do not, however, ensure that the automaton accepts executions defined over arbitrarily long time horizons. We know by assumption of Lipschitz continuity that the state cannot escape to infinity in finite time along continuous evolution. However, the infinite executions may be such that the state takes an infinite number of discrete transitions ($\langle\tau\rangle = \infty$) in finite time (i.e. $|\tau| < \infty$). Executions with this property are known as Zeno executions.

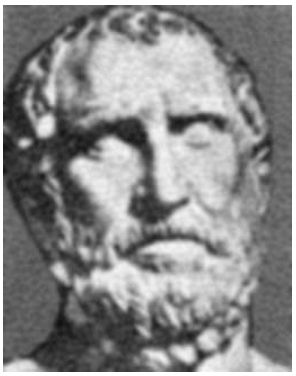


Figure 2: Zeno of Elea

4.1 Examples of Zeno Behavior

The name Zeno executions comes from the ancient Greek philosopher, Zeno of Elea (Figure 2). Born around 490BC, Zeno was a philosopher in the Eleatic school and a student of Parmenides. The teachings of Parmenides rejected the ideas of plurality and change as illusions generated by our senses. The main contribution of Zeno was a series of paradoxes designed to support the view of his mentor by showing that accepting plurality and motion leads to logical contradictions. One of the better known ones is the race of Achilles and the turtle.

> Achilles, a renowned runner, was challenged by the turtle to a race. Being a fair sportsman, Achilles decided to give the turtle a 100 meter head-start. To overtake the turtle, Achilles will have to first cover half the distance separating them, i.e. 50 meters. To cover the remaining 50 meters, he will first have to cover half that distance, i.e. 25 meters, and so on, ad infinitum. Covering each one of the segments in this series requires a non zero amount of time. Since there is an infinite number of segments, Achilles will never overtake the turtle.

This paradox may seem simple minded to the modern reader, but it was not until the beginning of the 20^{th} century that it was resolved satisfactorily by mathematicians and philosophers. And it was not until the end of the 20^{th} century that it turned out to be a practical problem, in the area of hybrid systems.

To motivate the subsequent discussion we list a few examples to illustrate different aspects of the Zeno phenomenon.

Example (Chattering System) Consider the autonomous hybrid automaton of Figure 3. It is easy to show that the hybrid automaton accepts a unique infinite execution for all initial states. However, all infinite executions are Zeno. An execution starting in x_0 at τ_0 reaches $x = 0$ in finite time $\tau_0' = \tau_0 + |x_0|$ and takes an infinite number of transitions from then on, without time progressing further. Thus $\|\tau\| = \tau_0 + |x_0|$. ∎

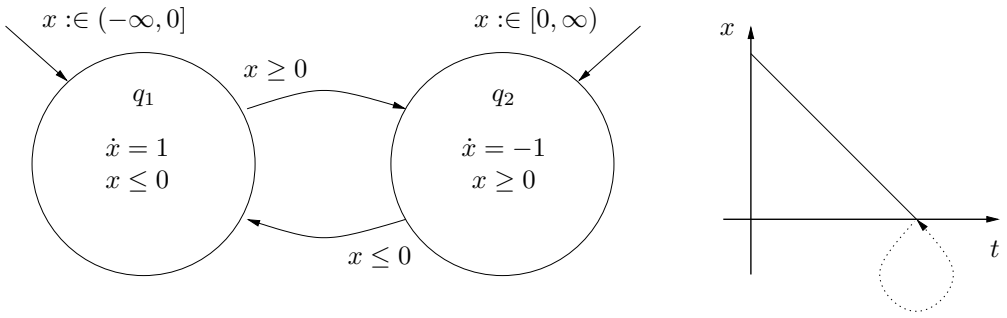


Figure 3: Chattering system.

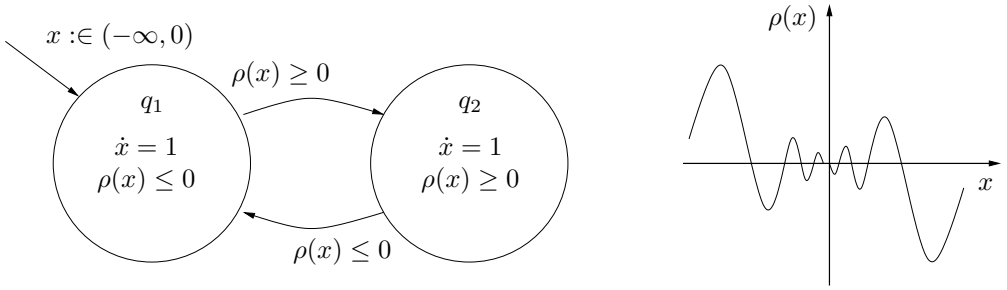


Figure 4: System with a smooth, non-analytic domain.

Example (Non-analytic Domain) Consider the hybrid automaton of Figure 4. Assume that the function $\rho : \mathbb{R} \to \mathbb{R}$ that determines the boundary of the domain is of the form

$$\rho(x) \begin{cases} \sin\left(\frac{1}{x^2}\right) \exp\left(-\frac{1}{x^2}\right) & \text{if } x \neq 0 \\ 0 & x = 0 \end{cases}$$

The function ρ is smooth, but is not analytic in a neighborhood of the origin. It is easy to check that the automaton is non-blocking and deterministic.

For any $\epsilon > 0$, ρ has an infinite number of zero crossings in the interval $(-\epsilon, 0]$. Therefore, the execution of the hybrid automaton with initial state (q_1, x_0) will take an infinite number of discrete transitions in the finite interval $[\tau_0, \tau_0 + |x_0|)$ (notice that $x_0 < 0$).

■

Example (Water Tank (continued)) We have already shown that the water tank hybrid automaton accepts a unique infinite execution for each initial state. In addition, if the inflow is greater than each of the outflows but is less than their sum ($\max\{v_1, v_2\} < w < v_1 + v_2$), then all infinite executions are Zeno. One can show that

$$\|\tau\| = \tau_0 + \frac{x_1(\tau_0) + x_2(\tau_0) - r_1 - r_2}{v_1 + v_2 - w}$$

■

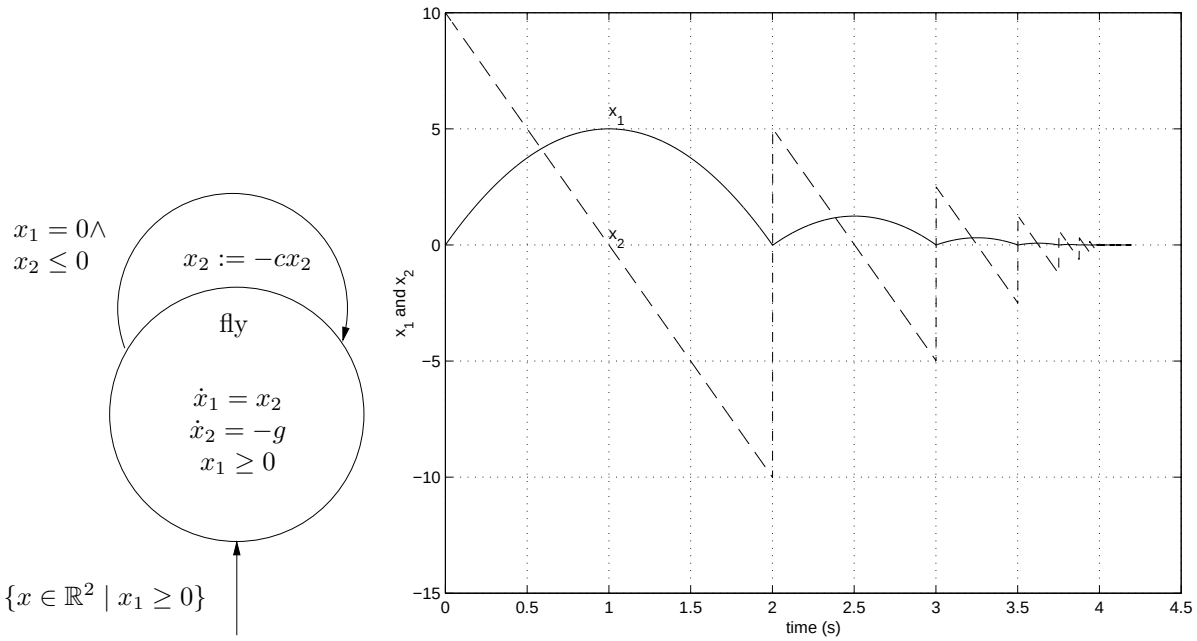


Figure 5: The bouncing ball automaton and one of its Zeno executions.

Example (Bouncing Ball) The boucing ball automaton (Figure 5) is a model of an elastic ball bouncing on a level surface. x_1 denotes the hight of the ball above the surface and x_2 its vertical velocity. It is assumed that the ball loses a fraction $c^2 \in [0, 1]$ of its energy at each bounce. g is the acceleration due to gravity and the mass of the ball is assumed to be 1..

One can show that the bouncing ball hybrid automaton is non-blocking and deterministic. Moreover, if $c < 1$ all infinite executions are Zeno. The first bounce occurs at time:

$$\tau_0' = \tau_1 = \tau_0 + \frac{x_2(\tau_0) + \sqrt{x_2^2(\tau_0) + 2gx_1(\tau_0)}}{g}$$

The second bounce occurs at time:

$$\tau_2 = \tau_1' = \tau_0 + \tau_1 + \frac{2x_2(\tau_1)}{g}$$

More generally, the N^{th} bounce occurs at time:

$$\tau_N = \tau_1' = \tau_0 + \tau_1 + \frac{2x_2(\tau_1)}{g} \sum_{k=1}^{N} c^{k-1}$$

where $x_2(\tau_1) = -cx_2(\tau_0') = c\sqrt{x_2^2(\tau_0) + 2gx_1(\tau_0)}$. Since for $c \in [0, 1)$,

$$\sum_{k=1}^{N} c^{k-1} \to \frac{1}{1-c} \text{ as } N \to \infty \tag{2}$$

we have that

$$\begin{aligned}\|\tau\| &= \tau_0 + \frac{x_2(\tau_0) + \sqrt{x_2^2(\tau_0) + 2gx_1(\tau_0)}}{g} + \frac{2x_2(\tau_1)}{g(1-c)} \\ &= \tau_0 + \frac{x_2(\tau_0)}{g} + \frac{(1+c)\sqrt{x_2^2(\tau_0) + 2gx_1(\tau_0)}}{g(1-c)}\end{aligned}$$

Figure 5 shows a Zeno execution in which $\|\tau\| = 4$. ∎

It can be argued that Zeno behavior is little more than a mathematical curiosity and should not be an issue when dealing with physical systems. However, modeling abstraction, often employed by engineers to simplify models for the purpose of analysis and control, can easily lead to Zeno hybrid models of physical systems, as the water tank and bouncing ball examples illustrate. Zeno executions turn out to be a source of a number of problems in the simulation and analysis of hybrid automata. When faced with Zeno executions, simulation algorithms are likely to stall or produce incorrect results. Analysis algorithms may result in misleading claims as the following example illustrates.

Example (Water Tank (continued)) Consider the water tank automaton and assume that $\max\{v_1, v_2\} < w < v_1 + v_2$. We have already shown that under this assumption Init_{WT} is an invariant set. Therefore, along all executions, the water in both tanks remains above the corresponding levels ($x_1 \geq r_1$ and $x_2 \geq r_2$ for all times). ∎

Though mathematically sound, the above argument is misleading from a practical point of view. If the amount of water coming in to the system is less that the amount of water going out, we know that sooner or later at least one of the tanks will have to drain (just as we know that sooner or later Achilles will overtake the turtle). The point is that one would never be able to infer this fact by analyzing the hybrid automaton model of the water tank system.

4.2 Zeno Hybrid Automata

Zeno executions are a fundamentally hybrid phenomenon. They can not appear in purely discrete or purely continuous systems, since they require the interaction of continuous dynamics (in the form of time) and discrete dynamics (in the form of discrete transitions).

Definition 10 *(Zeno Automaton and Zeno Time) An autonomous hybrid automaton H is called Zeno if for some $(q_0, x_0) \in$ Init all infinite executions are Zeno ($\langle\tau\rangle = \infty$ and $\|\tau\| < \infty$). The continuous extent, $\|\tau\|$, of a Zeno execution is called the Zeno time.*

Definition 11 *(Zeno State) Consider an autonomous hybrid automaton H that accepts a Zeno execution. A state (q, x) is called a Zeno state of this execution if there exists a sequence $\{\theta_j\}_{j=1}^{\infty}$ with $\theta_j \in I_{i_j} \in \tau$ and $\lim_{j\to\infty} \theta_j = \|\tau\|$ such that for all $N > 0$ and all $\epsilon > 0$ and $d((q_{i_j}(\theta_j), x_{i_j}(\theta_j)), (q, x)) < \epsilon$ for some $i > N$.*

The Zeno state is a special case of an ω limit point of an infinite execution. The discrete part of the Zeno state consists of a discrete state that is visited infinitely often by a Zeno execution.

4.3 Resolving the Zeno Phenomenon

In both the bouncing ball and water tank automata, the Zeno behavior is due to modeling simplifications. In the water tank example, the switching dynamics associated with the inflow have been abstracted with an ideal switch, and in the bouncing ball example, the bounce dynamics have been replaced with a simple reset map. In these two examples, an infinite number of transitions takes place in the time interval $(\tau_\infty - \epsilon, \tau_\infty)$ for any $\epsilon > 0$.

The first example above (hybrid automaton modeling chatter) is an example of a Zeno hybrid automaton for which there exists an interval $(\tau_\infty - \epsilon, \tau_\infty)$ in which no transitions take place, while an infinite number of transitions take place at τ_∞. The classical way of analyzing (and controlling!) such systems is by introducing the concept of sliding modes [1, 2].

For the bouncing ball and water tank, we consider an approach known as *regularization*. In the study of differential equations, regularization is a fairly standard process in which one attempts to slightly alter a differential equation whose solution is not well defined, to one whose solution is well defined. The approach can be extended to Zeno hybrid automata, to extend Zeno executions beyond the Zeno time. The general idea and some examples are given below.

- Let H be a non-blocking and deterministic hybrid automaton.
- Assume that for every $(q_0, x_0) \in \text{Init}$ the execution starting at (q_0, x_0) is Zeno.
- We *regularize* H by constructing a family of deterministic, non-blocking, and non-Zeno automata H_ϵ, parameterized by a real valued parameter $\epsilon > 0$, and a continuous map:

$$\phi : Q_\epsilon \times X_\epsilon \to Q \times X \tag{3}$$

  which relates the state of each H_ϵ to the state of H.
- In general $\phi((\tau_\epsilon, q_\epsilon, x_\epsilon)$ will not be an execution of H. However, the construction of H_ϵ is such that H_ϵ tends to H as ϵ tends to 0.

Example (Regularization of bouncing ball): Consider again the bouncing ball automaton of Figure 5.

1. Temporal Regularization: Assume each bounce of the ball takes time $\epsilon > 0$. The regularized automaton is shown in Figure 6. One can shown that the automaton of Figure 6 accepts a unique, non-Zeno execution for each initial state. The state of the regularized system is related to the state of the system by

$$\phi(q_0, (x_1, x_2, x_3)) = \phi(q_1, (x_1, x_2, x_3)) = (q, (x_1, x_2)) \tag{4}$$

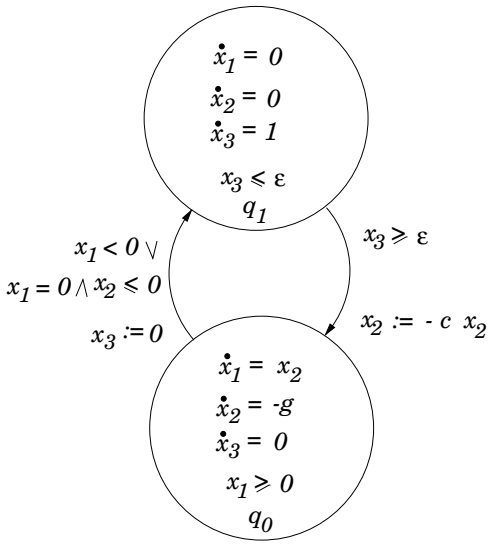


Figure 6: Temporal regularization of bouncing ball automaton.

Figure 7 shows simulation results for the regularized system: x_1, x_2 and q are plotted as functions of time for $\epsilon = 0.1$ and $\epsilon = 0.01$. As ϵ decreases, the execution of the regularized automaton converges to the execution of the actual automaton for $t \in (0, \tau_\infty)$. For $t > \tau_\infty$, the execution converges to the constant $x_1(t) = x_2(t) = 0$.

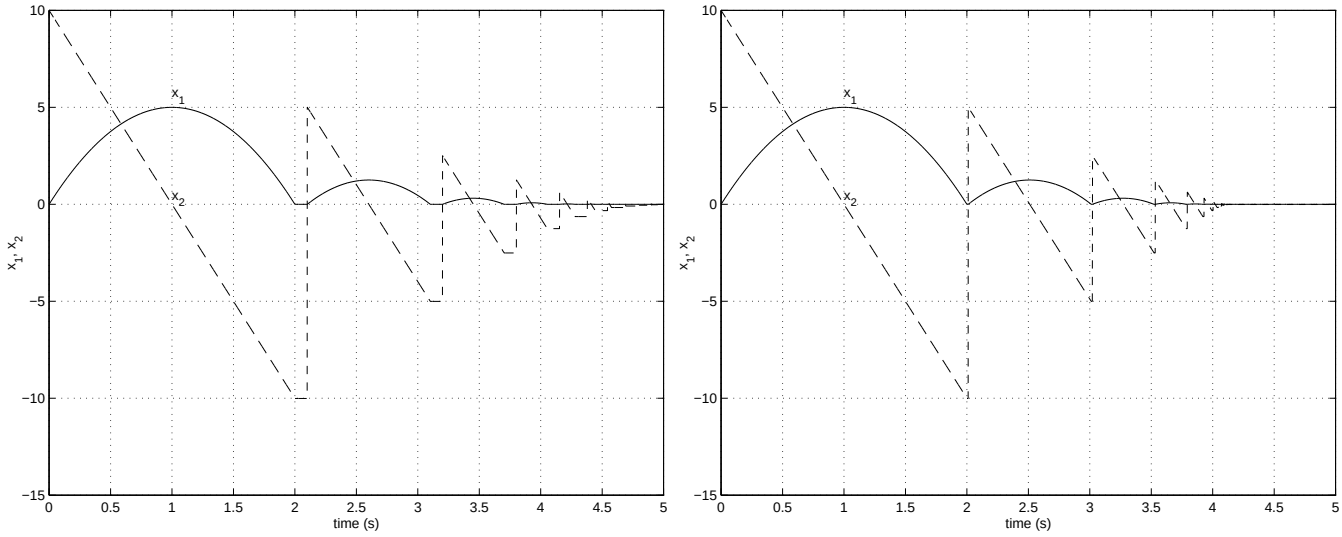


Figure 7: Temporal regularization of bouncing ball automaton ($\epsilon = 0.1$ and $\epsilon = 0.01$).

2. Dynamic Regularization: Assume that the ground is modeled as a stiff spring with spring constant $1/\epsilon$ and no damping. The regularized automaton is shown in Figure 8. One can shown that the automaton of Figure 8 accepts a unique, non-Zeno execution for each initial state. The state of the regularized system is related to the state of the system by

$$\phi(q_0, (x_1, x_2)) = \phi(q_1, (x_1, x_2)) = (q, (x_1, x_2)) \tag{5}$$

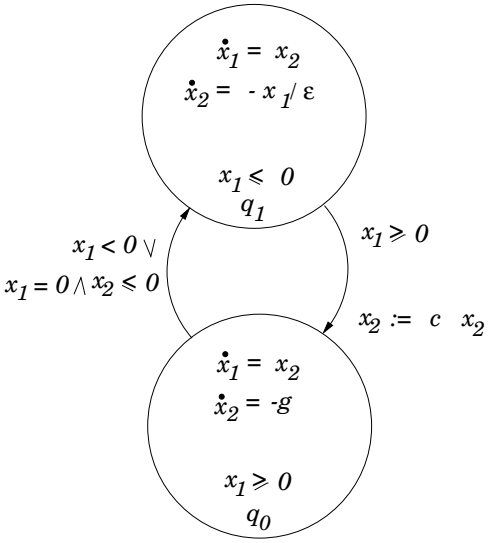


Figure 8: Dynamic regularization of bouncing ball automaton.

Figure 9 shows simulation results for the regularized system: x_1, x_2 and q are plotted as functions of time for $\epsilon = 0.01$ and $\epsilon = 0.001$.

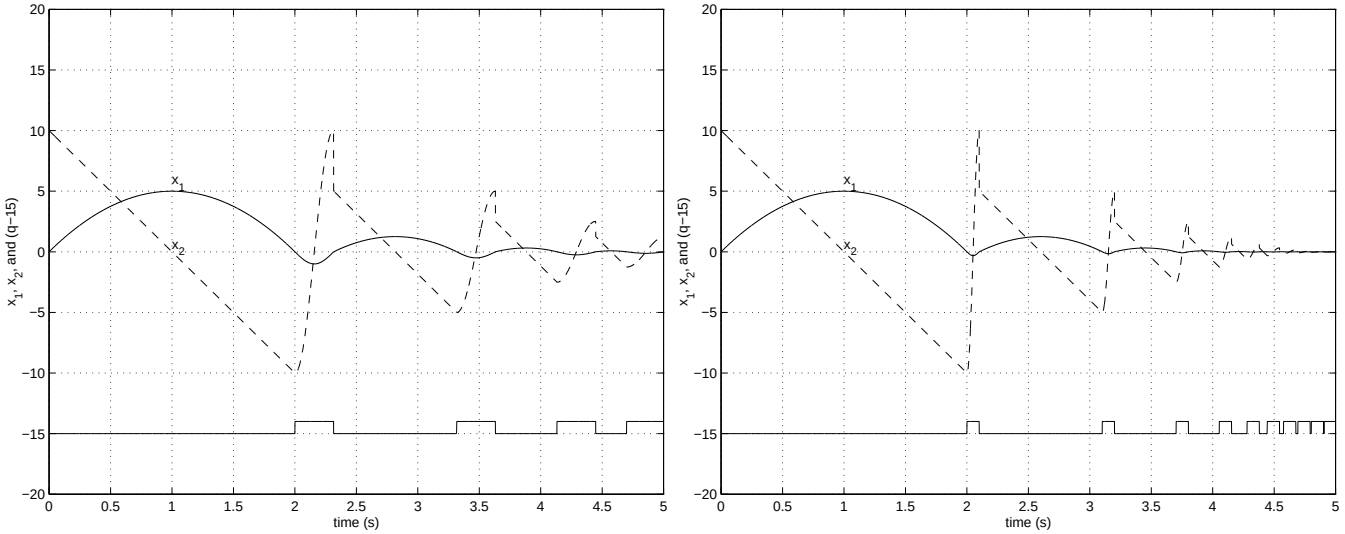


Figure 9: Dynamic regularization of bouncing ball automaton ($\epsilon = 0.01$ and $\epsilon = 0.001$).

Notes and Bibliography

Local existence is arguably the most important of the fundamental properties studied above (and the easiest to check!). One can argue that Zeno is also important, since the presence of Zeno executions makes hybrid automata difficult to simulate, and may lead to misleading claims of safety. However, Zeno hybrid automata can in some sense be very useful (even

realistic) models of physical systems. Therefore, instead of requiring that hybrid automata do not accept Zeno executions, it may be better to deal with Zeno problems when they arise, for example by defining appropriate extensions of Zeno execution beyond the Zeno time. This is the approach taken in [3] motivated by the classical definition of "sliding" flows for discontinuous vector fields.

Uniqueness of execution may also be desirable, since it makes simulation and analysis easier. However, requiring that hybrid automata are deterministic is usually an unnecessarily strict restriction. It is sometimes very desirable to work with non-deterministic hybrid automata, since they allow one to model uncertainty in a physical process and its environment. Moreover, analysis of non-deterministic hybrid automata is not much more difficult. Instead of arguing about *the* execution of the system, one simply has to ensure that *all* possible executions of the system are considered.

References

[1] A. F. Filippov. *Differential equations with discontinuous right hand sides.* Kluwer Academic Publishers, Boston, 1988.

[2] V. I. Utkin. *Sliding Modes in Control Optimization.* Springer Verlag, Berlin, 1992.

[3] K. J. Johansson, M. Egerstedt, J. Lygeros, and S. Sastry. On the regularization of zeno hybrid automata. *Systems and Control Letters*, 38:141–150, 1999.

AA278A Lecture Notes 5. Sequence Properties and Verification.

Claire J. Tomlin

April 11, 2005

One of our goals in this course is to develop tools for analyzing whether or not a hybrid system satisfies certain desirable properties. A second goal is to design controllers for hybrid systems, such that the closed loop hybrid system satisfies such properties. Thus, we would like to be able to answer the questions of:

- **Verification:** Does the hybrid automaton meet the specification?
- **Synthesis:** Can a controller be designed such that the closed loop hybrid system satisfies the specification?

In this chapter, we will discuss the **safety** property; in the next chapter, we will discuss the **stability** property.

1 Reachability and Sequence Properties

- The problem we will address is, given a hybrid automaton H compute Reach(H).
- If we can solve this problem we can also answer questions about safety properties. A safety property can generally be posed as:
  Does the state (q, x) always remain in a set of states $G \subseteq Q \times X$?
  The set G can be used to encode "good" or "safe" states. For example, G could be the set of states for which two aircraft always maintain the minimum required separation distance. Using notation from temporal logic, the above safety property can be written as
  $$\Box((q, x) \in G)$$

Here, $\Box$ stands for "always", and the way to read the above formula is "the property that (q, x) is in G is always satisfied". We will also use the notation $(Q \cup X, \Box G)$ to indicate that for the hybrid automaton H with states $Q \cup X$, it is always the case that the state stays in G.

- The dual property of safety is known as **liveness**. A liveness property can be posed as:
  Does the state (q, x) eventually reach a set of states $G \subseteq Q \times X$?,
  This reflects the fact that something good should eventually happen to our system. In temporal logic notation this property can be written as

$$\Diamond((q, x) \in G)$$

  Here, $\Diamond$ stands for "eventually", and the way to read the above formula is "the property that (q, x) eventually reaches G is satisfied".

- Using concepts of "always" and "eventually", one can encode arbitrarily complex properties, such as:

$$\Box\Diamond((q, x) \in G)$$

  which means that the state visits the set G "infinitely often", and

$$\Diamond\Box((q, x) \in G)$$

  which means that the state reaches G at some point and stays there for ever after.

- In this course, we will focus on safety properties, and not liveness properties. This is because, for most of the practical systems we deal with, we are either interested in safety properties directly, or in the property that the state reaches a given set in a finite amount of time. *How would you pose this second property as a safety property?*

Proposition 1 *H satisfies $(Q \cup X, \Box G)$ for $G \subseteq Q \times X$ if and only if* $\mathrm{Reach}(H) \subseteq G$.

- Different methods have been proposed for solving the reachability problem:
  1. **Optimal Control**: The role of the "control" is often played by the non-determinism of the system.
  2. **Deductive Techniques**: Establish invariants to bound $\mathrm{Reach}(H)$.
  3. **Model Checking Techniques**: Automatically compute $\mathrm{Reach}(H)$. Requires one to be able to "compute" with sets of states. Class of systems to which it applies is inherently limited.
  4. **Approximation**: Works with all of the above
     - For optimal control, approximate by sets and dynamics for which optimal control problems are easier to solve (e.g. ellipsoidal sets and linear dynamics)
     - Deductive techniques are inherently based on over-approximation

  - For model checking, approximate by a system you can compute with.
  - Also possible to do "brute force" over-approximation, based, for example, on gridding.

- In all cases you stop once your question has been answered. For example, if your question was "does H satisfy $(Q \cup X, \Box G)$", you could stop if you found a reachable state outside G.

- For approximation, you typically "over-approximate".

- All methods are supported by computational tools:

  1. typically uses optimal control and convex optimization tools
  2. typically uses theorem provers
  3. typically uses model checkers
  4. done using "gridding" or polynomial manipulation packages.

- **Simulation** can also be used for reachability investigations. It is not a formal method however since:

  - It is a shot in the dark: we simulate. If G^C, the complement of G, is reached, fine, else we have to choose another initial condition and try again.
  - No termination guarantee.

  Still it is the best we can do for many classes of hybrid systems.

2 General Transition Systems

Definition 2 (Transition System) *A transition system is a collection* $T = (S, \Sigma, \rightarrow, S_0, S_F)$*, where*

- S *is a set of states;*
- Σ *is an alphabet of events;*
- $\rightarrow$: $S \times \Sigma \rightarrow 2^S$ *is a transition relation;*
- $S_0 \subseteq S$ *is a set of initial states; and,*
- $S_F \subseteq S$ *is a set of final states.*

Example: A finite automaton $M = (Q, \Sigma, \text{Init}, R)$, with final states F specified, is a transition system with:

- $S = Q$

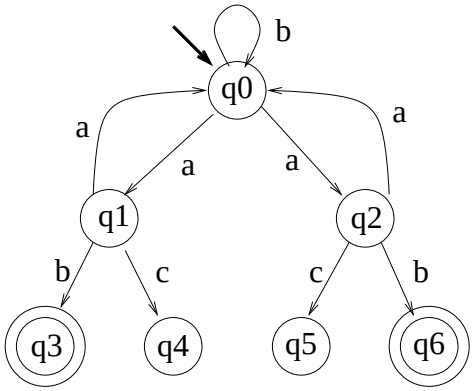


Figure 1: Reachability example for finite automata

- Σ the same
- $\rightarrow = R$
- $S_0 = \text{Init}$
- $S_F = F$

Example: An autonomous hybrid automaton $H = (Q,\ X,\ \text{Init},\ f,\ \text{Dom},\ R)$ and a safety property $(Q \cup X, \Box G)$ form a transition system with:

- $S = Q \times X$
- Σ not specified
- $\rightarrow = \{$ discrete transitions $\} \cup \{$ continuous evolution$\}$, all of which are characterized by f, Dom, and R
- $S_0 = \text{Init}$
- $S_F = G^C$

Problem 1 (Reachability) *Given a transition system T, is any state $s_f \in S_F$ reachable from a state $s_0 \in S_0$ by a sequence of transitions?*

Remark: For finite automata we can always "decide" reachability problems by brute force.

Example: Consider for example the finite automaton of Figure 1. We can start "exploring" from q_0 and keep going until we either visit a state in F or have nowhere else to go (have visited all reachable states).

More formally, we can approach this problem using the **predecessor operator**:

$$\text{Pre} : 2^S \rightarrow 2^S$$

which is defined as follows (for a set $S' \subseteq S$):

$$\text{Pre}(S') = \{s \in S : \exists s' \in S' \exists \sigma \in \Sigma \text{ such that } s' \rightarrow_\sigma s\}$$

In the above, $\rightarrow_\sigma$ indicates that $(s, \sigma) \rightarrow s'$. Thus, the **predecessor** of a set of states S' is the set of states that can reach S' in one transition. The set of reachable states for a transition system can be computed using the following algorithm:

Algorithm 1 (Reachability)

Initialization:
    $W_0 = S_F$, $i = 0$
repeat
    if $W_i \cap S_0 \neq \emptyset$
        return "S_F reachable"
    end if
    $W_{i+1} = \text{Pre}(W_i) \cup W_i$
    $i = i + 1$
until $W_i = W_{i-1}$
return "S_F not reachable"

- For the example of Figure 1, if event a happens at iteration 1, and events b, c happen at iteration 2, then $W_0 = \{q_0\}$, $W_1 = \{q_0, q_1, q_2\}$ and $W_2 = Q$.
- For finite automata the algorithm *can be implemented* and *always terminates*.
- What properties of finite automata allow us to do this?
    1. We can represent states in a finite way (by enumeration).
    2. We can represent transitions among states in a finite way (by enumeration)
    3. We are guaranteed that if we start exploring a particular execution, after a finite number of steps we will either reach a state we visited already, or a state from which we have nowhere else to go, or a final state.
- Enumeration is a fairly naive way of going about reachability analysis. In practice, sets of states are not enumerated, but are represented more compactly, using, for example, Binary Decision Diagrams, or BDDs.
- For general transition systems, this algorithm is conceptually useful, but not directly implementable on a computer. To effectively implement this algorithm, on has to devise effective ways to:
    - store sets of states
    - compute the Pre of a set of states
    - take the union and intersection of sets of states
    - check whether a set of states is empty

– check whether two sets of states are equal

If the number of states is finite, one can do all of these relatively easy using enumeration; for non-finite state spaces, these are all difficult problems, which will be addressed in theis chapter and later chapters. One way of addressing this is through **bisimulation**.

3 Bisimulation

- In the example of Figure 1, q_1 and q_2 have very similar properties, since they can both be reached from q_0 by a and all subsequent executions look similar.
- Suggests that these states are in some sense "equivalent".
- Try to make this statement more precise by introducing the notion of bisimulation.
- Consider the set of states S. A relation on S is a subset of $S \times S$.

Definition 3 (Equivalence Relation) *A relation $\sim \subseteq S \times S$ is called an equivalence relation if it is:*

1. *Reflexive: $(s, s) \in \sim$ for all $s \in S$;*
2. *Symmetric: $(s, s') \in \sim$ implies that $(s', s) \in \sim$; and,*
3. *Transitive: $(s, s') \in \sim$ and $(s', s'') \in \sim$ imply $(s, s'') \in \sim$.*

- For simplicity we write $s \sim s'$ instead of $(s, s') \in \sim$ and say s is equivalent to s'.
- An example of an equivalence relation: Equality.
- An equivalence relation partitions S to a number of *equivalence classes*:

$$S = \bigcup_i S_i$$

such that for all $s, s' \in S$, $s, s' \in S_i$ if and only if $s \sim s'$.

- Equivalence classes cover S (by symmetry)
- Equivalence classes are disjoint (by transitivity)
- For equality the equivalence classes are the singletons, for $S \times S$ there is only one equivalence class, S itself.
- Given an equivalence relation $\sim$, let $S/\sim = \{S_i\}$ denote the quotient space, i.e. the set consisting of all equivalence classes.

- Given a set $P \subseteq S$, let $P/\sim$ represent the part of the quotient space with which P overlaps:
$$P/\sim = \{S_i \ : \ S_i \cap P = \emptyset\} \subseteq S/\sim$$

- If S are the states of a transition system, $T = (S, \Sigma, \rightarrow, S_0, S_F)$, define the *quotient transition system* as
$$T/\sim = (S/\sim, \Sigma, \rightarrow_\sim, S_0/\sim, S_F/\sim)$$
where for $S_1, S_2 \in S/\sim$, $S_1 \times \sigma \rightarrow_\sim S_2$ if and only if there exist $s_1 \in S_1$ and $s_2 \in S_2$ such that $(s_1 \times \sigma \rightarrow s_2)$.

- Notice that the quotient transition system may be "non-deterministic", even if the original system is.

- Finally, we denote the predecessor under an event σ as Pre_σ:
$$\mathrm{Pre}_\sigma(P) = \{s \in S : \exists s' \in P \text{ such that } (s, \sigma) \rightarrow s'\}$$

Definition 4 (Bisimulation) *Given $T = (S, \Sigma, \rightarrow, S_0, S_F)$, and $\sim$ an equivalence relation over S, $\sim$ is called a bisimulation if:*

1. *S_0 is a union of equivalence classes;*
2. *S_F is a union of equivalence classes;*
3. *if one state (say s) in one equivalence class (say S_i) can transition to another equivalence class (say S_j), then all other states, $s' \in S_i$ must be able to transition to some state in S_j. More formally, for all i, j and for all states $s, s' \in S_i$, if $s \rightarrow S_j$, then $s' \rightarrow S_j$.*

Note that the third condition above is equivalent to "for all $\sigma \in \Sigma$, if P is a union of equivalence classes $\mathrm{Pre}_\sigma(P)$ is also a union of equivalence classes".

- If $\sim$ is a bisimulation, T and $T/\sim$ are called bisimilar.
- Equality is a bisimulation.
- In a sense bisimilar transition systems generate the same sequences of transitions (language).
- Therefore, if $\sim$ is a bisimulation, we need not distinguish between the elements of an equivalence class.
- More specifically, if a state in S_F is reachable from a state in S_0, a state in $S_F/\sim$ is reachable from a state in $S_F/\sim$.

Proposition 5 *$\sim$ is a bisimulation if and only if:*

1. $(s_1 \sim s_2) \wedge (s_1 \in S_0) \Rightarrow (s_2 \in S_0)$;

2. $(s_1 \sim s_2) \wedge (s_1 \in S_F) \Rightarrow (s_2 \in S_F)$; *and,*

3. $(s_1 \sim s_2) \wedge ((s_1, \sigma) \rightarrow s_1') \Rightarrow \exists s_2'$ *such that* $(s_1' \sim s_2') \wedge ((s_2, \sigma) \rightarrow s_2')$.

Proof: ($\Rightarrow$):

1. If for all $s_1 \in S_0$, $s_1 \sim s_2$ implies $s_2 \in S_0$, S_0 must be a union of equivalence classes.
2. Similarly for S_F.
3. If P is an equivalence class and $s_1 \in \text{Pre}_\sigma(P)$, then $s_2 \in \text{Pre}_\sigma(P)$ for all $s_2 \sim s_1$. Hence $\text{Pre}_\sigma(P)$ must be a union of equivalence classes.

($\Leftarrow$): similar ∎

Aside: More generally, two transition systems, $T = (S, \Sigma, \rightarrow, S_0, S_F)$ and $T' = (S', \Sigma, \rightarrow', S_0', S_F')$ are called *bisimilar* if there exists a relation $\sim \subseteq S \times S'$ such that:

1. $(s_1 \sim s_2) \wedge (s_1 \in S_0) \Rightarrow (s_2 \in S_0')$;
2. $(s_1 \sim s_2) \wedge (s_2 \in S_0') \Rightarrow (s_1 \in S_0)$;
3. $(s_1 \sim s_2) \wedge (s_1 \in S_F) \Rightarrow (s_2 \in S_F')$;
4. $(s_1 \sim s_2) \wedge (s_2 \in S_F') \Rightarrow (s_1 \in S_F)$;
5. $(s_1 \sim s_2) \wedge ((s_1, \sigma) \rightarrow s_1') \Rightarrow \exists s_2'$ such that $(s_1' \sim s_2') \wedge ((s_2, \sigma) \rightarrow' s_2')$.

4 Computing Bisimulations

- Bisimulations look useful since they preserve the language of the transition system.
- How does one find a bisimulation?

Algorithm 2 (Bisimulation)

Initialization:
$S/\sim = \{S_0, S_F, S \setminus (S_0 \cup S_F)\}$
while $\exists P, P' \in S/\sim$ and $\sigma \in \Sigma$ such that $P \cap \text{Pre}_\sigma(P') \neq P$ and $P \cap \text{Pre}_\sigma(P') \neq \emptyset$ **do**
begin
$P_1 = P \cap \text{Pre}_\sigma(P')$
$P_2 = P \setminus \text{Pre}_\sigma(P')$
$S/\sim = (S/\sim \setminus \{P\}) \cup \{P_1, P_2\}$
end

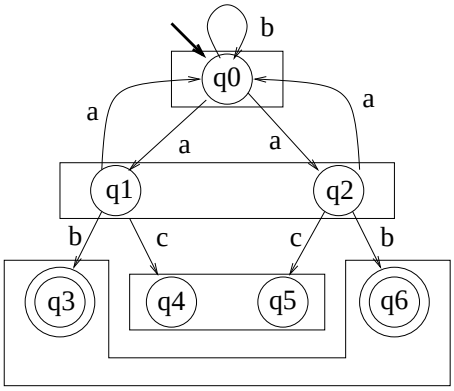


Figure 2: Bisimulation of previous example

- If the algorithm terminates, $\sim$ is a bisimulation, since:
  1. S_0 is a union of equivalence classes (note that $S_0 \in S/\sim$ initially, and we only split sets)
  2. S_F is a union of equivalence classes (for the same reason).
  3. Termination implies that for all $P' \in S/\sim$ and for all σ, $P \cap \mathrm{Pre}_\sigma(P')$ is either equal to P or equal to $\emptyset$, therefore, $\mathrm{Pre}_\sigma(P')$ is a union of equivalence classes.
- Again, implementation and termination of this algorithm for general transition systems are not obvious. For finite state systems we can implement the algorithm and guarantee that it terminates because we can enumerate the states for the finite state system.
- Figure 2 shows the results of applying this algorithm to our previous finite state example.
- Why is this an improvement?
  1. No need to enumerate all the states, therefore may have a computational advantage.
  2. Extends to systems with infinite states. If the bisimulation quotient can be computed and is finite, then the reachability computation is decidable.

5 Bisimulations of Timed Automata

- Consider $X = \{x_1, \ldots, x_n\}$ a finite collection of variables, each of which takes values in $\mathbb{R}$. and let $x = (x_1, \ldots, x_n) \in \mathbb{R}^n$

Definition 6 (Clock Constraints) *The set, $\Phi(X)$, of clock constraints of X, is a set of logical expressions defined inductively by $\delta \in \Phi(X)$ if:*

$$\delta := (x_i \leq c) | (x_i \geq c) | \neg\delta | \delta_1 \wedge \delta_2$$

where $x_i \in X$ and $c \geq 0$ is a rational number.

- Examples: let $X = \{x_1, x_2\}$.
  - $(x_1 \leq 1) \in \Phi(X)$
  - $(0 \leq x_1 \leq 1) \in \Phi(X)$, since $(0 \leq x_1 \leq 1) \Leftrightarrow (x_1 \geq 0) \wedge (x_1 \leq 1)$
  - $(x_1 = 1) \in \Phi(X)$, since $(x_1 = 1) \Leftrightarrow (x_1 \geq 1) \wedge (x_1 \leq 1)$
  - $(x_1 < 1) \in \Phi(X)$, since $(x_1 < 1) \Leftrightarrow (x_1 \leq 1) \wedge \neg(x_1 \geq 1)$
  - $\text{True} \in \Phi(X)$, since $\text{True} \Leftrightarrow \neg((x_1 = 1) \wedge \neg(x_1 = 1))$
  - $(x_1 \leq x_2) \notin \Phi(X)$
- Given $\delta \in \Phi(X)$, we say $x \in \mathbf{X}$ *satisfies* δ if $\delta(x) = \text{True}$.
- To each $\delta \in \Phi(X)$ we can associate a set:

$$\hat{\delta} = \{x \in \mathbf{X} \ : \ \delta(x) = \text{True}\}$$

- The original definition of a timed automaton, found in [1] is given below.

Definition 7 (Timed Automaton) *A timed automaton is a hybrid automaton $H = (Q, X, \text{Init}, f, \text{Dom}, R)$, where*

- *Q is a set of discrete variables, $Q = \{q_1, \ldots, q_m\}$;*
- *$X = \{x_1, \ldots, x_n\}$, $X = \mathbb{R}^n$;*
- *$\text{Init} = \{\{q_i\} \times \widehat{\text{Init}_{q_i}}\}_{i=1}^m$ where $\text{Init}_{q_i} \in \Phi(X)$;*
- *$f(q, x) = (1, \ldots, 1)$ for all (q, x);*
- *$\text{Dom}(q) = X$ for all $q \in Q$;*
- *$R : Q \times \Phi(X) \to Q \times X$ where $R(q, x)$ either leaves x_i unaffected or resets it to 0 (notice that R is single valued).*

Example: As an example consider the timed automaton of Figure 3.

- $Q = \{q_1, q_2\}$;
- $X = \{x_1, x_2\}$, $\mathbf{X} = \mathbb{R}^2$;
- $\text{Init} = \{(q_1, 0, 0)\}$;
- $f(q, x) = (1, 1)$ for all (q, x);
- $\text{Dom}(q) = \mathbb{R}^2$ for all $q \in Q$;

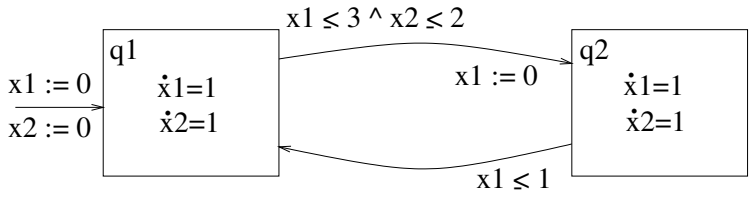


Figure 3: Example of a timed automaton

- $E = \{(q_1, q_2), (q_2, q_1)\}$;
- $G(q_1, q_2) = \{x \in \mathbb{R}^2 \ : \ (x_1 \leq 3) \wedge (x_2 \geq 2)\}$, $G(q_2, q_1) = \{x \in \mathbb{R}^2 \ : \ (x_1 \leq 1)\}$;
- $R(q_1, q_2, x) = \{(0, x_2)\}$, $R(q_2, q_1, x) = \{(x_1, 0)\}$

where the graphical notation for hybrid automata has been used (E edge, G guard).

6 Timed Automata are Bisimilar to Finite Systems

- Without loss of generality, all constants can be assumed to be integers.
- Let T be the transition system defined by a timed automaton, H.
- Consider an arbitrary $\lambda > 0$, rational.
- Let H_λ denote the timed automaton obtained by replacing all constants, c, in H by λc.
- Let T_λ denote the transition system associated with H_λ.

Proposition 8 *T and T_λ are bisimilar.*

Proof: Consider the relation $(q, x) \sim (q, \lambda x)$. Note that, since $\lambda > 0$, $(x_i \leq c) \Leftrightarrow (\lambda x \leq \lambda c)$ and $(x_i \geq c) \Leftrightarrow (\lambda x \geq \lambda c)$. Therefore:

$$
\begin{aligned}
(q, x) \in \text{Init} &\Leftrightarrow (q, \lambda x) \in \text{Init}_\lambda && (1)\\
(q, x) \in F &\Leftrightarrow (q, \lambda x) \in F_\lambda && (2)\\
(q, x) \xrightarrow{e} (q', x') &\Leftrightarrow (q, \lambda x) \xrightarrow{e} (q', \lambda x') && (3)\\
(q, x) \xrightarrow{\tau} (q', x') &\Leftrightarrow (q, \lambda x) \xrightarrow{\tau} (q', \lambda x') && (4)
\end{aligned}
$$

For the discrete transition, $(q, x) \xrightarrow{e} (q', x')$ if $e = (q, q') \in E$, $Ge(x) = \text{True}$ and $x' \in R(e, x)$. Therefore, $(q, \lambda x) \xrightarrow{e} (q', \lambda x')$, since $e = (q, q') \in E$, $G_{e_\lambda}(\lambda x) = \text{True}$ and $\lambda x' \in R_\lambda(e, \lambda x)$. For the continuous transition, recall that $(q, x) \xrightarrow{\tau} (q', x')$ if $q = q'$, and there exists $t \geq 0$ such that $x' = x + t(1, \ldots, 1)$. Therefore, $(q, \lambda x) \xrightarrow{\tau} (q', \lambda x')$ since $q = q'$ and $\lambda x' = \lambda x + \lambda t(1, \ldots, 1)$. ∎

- We can therefore assume all constants are integers. If they are not, we let λ be a common multiple of their denominators and consider the bisimilar system T_λ.
- Let c_i denote the largest constant with which x_i is compared.
- In the above example, $c_1 = 3$ and $c_2 = 2$.
- Let $\lfloor x_i \rfloor$ denote the integer part of x_i and $\langle x_i \rangle$ denote the fractional part of x_i. In other words, $x_i = \lfloor x_i \rfloor + \langle x_i \rangle$, $\lfloor x_i \rfloor \in \mathbb{Z}$ and $\langle x_i \rangle \in [0, 1)$.
- Consider the relation $\sim \subseteq \mathbf{Q} \times \mathbf{X}$ with $(q, x) \sim (q', x')$ if:
  1. $q = q'$;
  2. for all x_i, $\lfloor x_i \rfloor = \lfloor x_i' \rfloor$ or $(\lfloor x_i \rfloor > c_i) \vee (\lfloor x_i' \rfloor > c_i)$
  3. for all x_i, x_j with $x_i \leq c_i$ and $x_j \leq c_j$
$$(\langle x_i \rangle \leq \langle x_j \rangle) \Leftrightarrow (\langle x_i' \rangle \leq \langle x_j' \rangle)$$
  4. for all x_i with $x_i \leq c_i$,
$$(\langle x_i \rangle = 0) \Leftrightarrow (\langle x_i' \rangle = 0)$$

Proposition 9 $\sim$ *is an equivalence relation.*

Proof: Probably part of homework 3! ∎

- What do the equivalence classes look like?
- The equivalence classes are either open triangles, open line segments, open parallelograms or points.
- For the example introduced above, they are shown in Figure 4.
- Notice that the number of classes is $2 \times (12 \text{ points} + 30 \text{ lines} + 18 \text{ open sets})$. Quite a few, but definitely finite!

Proposition 10 $\sim$ *is a bisimulation.*

Proof: We need to show:

1. Init is a union of equivalence classes.
2. F is a union of equivalence classes.
3. If P is an equivalence class and $e \in E$, $\text{Pre}_e(P)$ is a union of equivalence classes.
4. If P is an equivalence class, $\text{Pre}_\tau(P)$ is a union of equivalence classes.

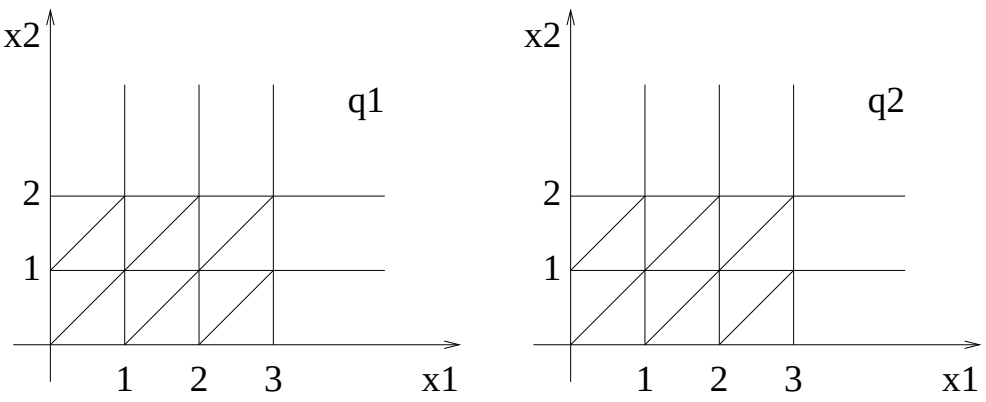


Figure 4: Equivalence classes for the example

The proof will be somewhat informal. ∎

- First, note that if $\delta \in \Phi(X)$,

  $$\hat{\delta} = \{x \in \mathbf{X} \ : \ \delta(x) = \text{True}\}$$

  is "a union of equivalence classes" (for the X variables only). Recall that $\hat{\delta}$ can be written as the product of unions and intersections of sets of the form:

  $$\{x_i \geq c\}, \{x_i \leq c\}, \{x_i < c\}, \{x_i > c\}, \{x_i = c\}$$

  where c is an integer constant. All these sets are unions of equivalence classes for the X variables.

- This takes care of the requirements on Init and F.

- To deal with $\text{Pre}_e(P)$, let:

  $$R^{-1}(e, P) = \{(q, x) \in \mathbf{Q} \times \mathbf{X} \ : \ \exists (q', x') \in P \text{ with } e = (q, q'), x' \in R(e, x)\}$$

- Notice that:

  $$\text{Pre}_{(q,q')}(P) = R^{-1}(q, q', P) \cap (\{q\} \times G(q, q'))$$

  Proposition 11 *If P is an equivalence class, $R^{-1}(e, P)$ is a union of equivalence classes. Hence* $\text{Pre}_e(P)$ *is a union of equivalence classes.*

- Easier to demonstrate by examples. For the timed automaton of Figure 3, consider

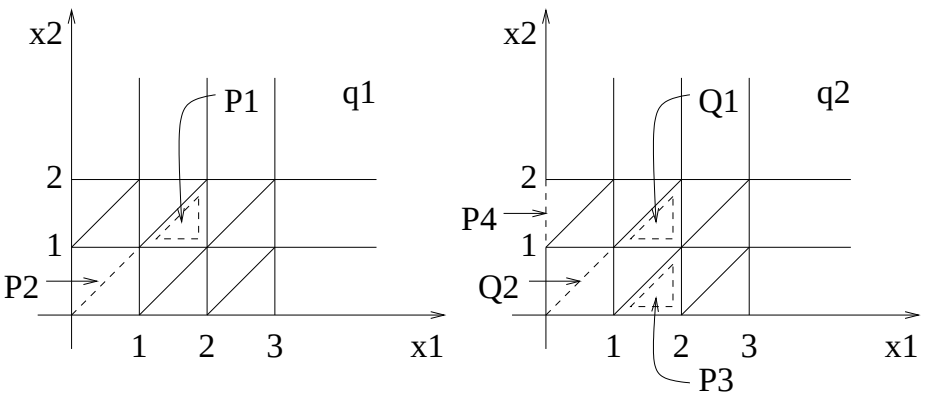


Figure 5: Examples of Pre_e computation

the equivalence classes $P_1, \ldots, P_4$ shown in Figure 5. Notice that:

$$\begin{aligned}
\mathrm{Pre}_{e_1}(P_1) &= \emptyset, \\
\mathrm{Pre}_{e_2}(P_1) &= Q_1 \cap (\{q_2\} \times \{x_1 \leq 1\}) = \emptyset \\
\mathrm{Pre}_{e_1}(P_2) &= \emptyset, \\
\mathrm{Pre}_{e_2}(P_2) &= Q_2 \cap (\{q_2\} \times \{x_1 \leq 1\}) = Q_2 \\
\mathrm{Pre}_{e_1}(P_3) &= \emptyset \cap (\{q_1\} \times \{x_1 \leq 1 \wedge x_2 \leq 2\}) = \emptyset, \\
\mathrm{Pre}_{e_2}(P_3) &= \emptyset \\
\mathrm{Pre}_{e_1}(P_4) &= \{q_1\} \times (\{x_1 \geq 0 \wedge 1 < x_2 < 2\} \cap \{x_1 \leq 1 \wedge x_2 \leq 2\}) \\
&= \{q_1\} \times \{0 \leq x_1 \leq 3 \wedge 1 < x_2 < 2\}, \\
\mathrm{Pre}_{e_2}(P_4) &= \emptyset
\end{aligned}$$

In all cases the result is a union of equivalence classes.

- Finally, notice that

$$\mathrm{Pre}_\tau(P) = \{(q, x) \in \mathbf{Q} \times \mathbf{X} \ : \ \exists (q', x') \in P, t \geq 0 \text{ with } q = q', x' = x + t(1, \ldots, 1)\}$$

These are all points that if we move in the $(1, \ldots, 1)$ direction we will eventually reach P. If P is an equivalence class, this set is also a union of equivalence classes.

- For example, in Figure 6, $\mathrm{Pre}_\tau(P) = P \cup P_1 \cup P_2 \cup P_3 \cup P_4 \cup P_5$

7 Complexity Estimates and Generalizations

- The above discussion indicates that reachability questions for timed automata can be answered on a finite state system, defined on the quotient space of the bisimulation $\sim$ (also known as the *region graph*).
- What is the "size" of this finite state system?

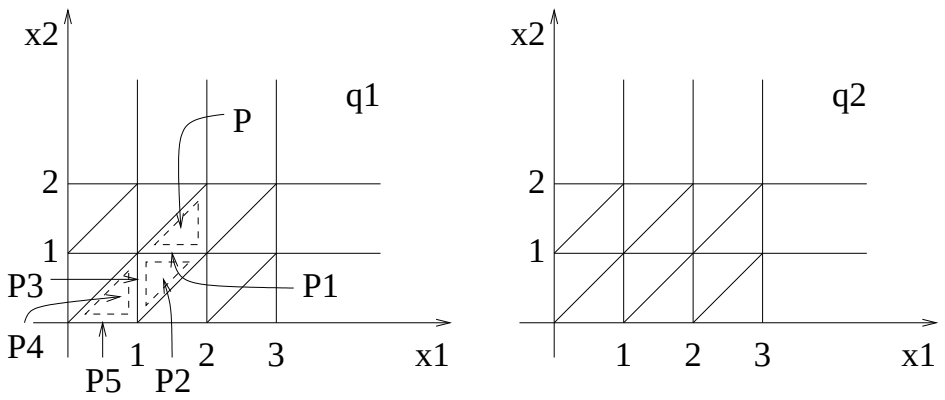


Figure 6: Examples of Pre_τ computation

- For the example, the number of equivalence classes is 2(12points+30lines+18open sets) = 120. Quite a few!
- For an arbitrary system, one can expect up to $m(n!)(2^n)\prod_{i=1}^{n}(2c_i+2)$ discrete states.
- Of course, in general, one need not construct the entire region graph:
  1. On the fly reachability: run the reachability algorithm. Typically it will terminate, without constructing the entire region graph.
  2. Construct coarser bisimulations: run the bisimulation algorithm. Typically the bisimulation generated will have fewer equivalence classes than the region graph.
- Still the problem is PSPACE complete!
- The finite bisimulation result is preserved under some simple extensions:
  1. $I(q) = \hat{I}_q$ for some $I_q \in \Phi(X)$.
  2. $f(q,x) = (k_1, \ldots, k_n)$ for some rational $k_1, \ldots, k_n$ and all (q,x).
  3. R mapping equivalence classes to equivalence classes.

8 Rectangular Hybrid Automata

- The largest class of systems of this form that is known to be decidable is the class of *initialized rectangular automata* [2, 3, 4, 5].
- A set $R \subset \mathbb{R}^n$ is called a rectangle if $R = \prod_{i=1}^{n} R_i$ where R_i are intervals whose finite end points are rational.

Definition 12 (Rectangular Automaton) *A rectangular automaton is a hybrid automaton $H = (Q, X, \mathrm{Init}, f, \mathrm{Dom}, R)$, where*

- *Q is a set of discrete variables, $\mathbf{Q} = \{q_1, \ldots, q_m\}$;*

- $X = \{x_1, \ldots, x_n\}$, $X = \mathbb{R}^n$;
- $\text{Init} = \cup_{i=1}^{m} \{q_i\} \times \text{Init}(q_i)$ *where* $\text{Init}(q_i)$ *is a rectangle;*
- $f(q, x) = F(q)$ *for all* (q, x), *where* $F(q)$ *is a rectangle;*
- $\text{Dom}(q)$ *is a rectangle for all* $q \in \mathbf{Q}$;
- $R(q, x)$ *either leaves* x_i *unaffected or resets to an arbitrary value in a rectangle.*

References

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

[2] R. Alur, C. Courcoubetis, T.A. Henzinger, P.-H. Ho, X. Nicollin, A. Olivero, J. Sifakis, and S. Yovine. The algorithmic analysis of hybrid systems. In G. Cohen and J.-P. Quadrat, editors, *Proceedings of the 11th International Conference on Analysis and Optimization of Systems: Discrete-event Systems*, number 199 in LNCS, pages 331–351. Springer Verlag, 1994.

[3] T.A. Henzinger. Hybrid automata with finite bisimulations. In Z. Fülöp and F. Gécseg, editors, *ICALP 95: Automata, Languages, and Programming*, number 944 in LNCS, pages 324–335. Springer Verlag, 1995.

[4] R. Alur, C. Courcoubetis, N. Halbwachs, T. A. Henzinger, et al. The algorithmic analysis of hybrid systems. *Theoretical Computer Science*, Vol.138(1):3–4, 1995.

[5] R. Alur, C. Courcoubetis, T. A. Henzinger, and P.-H. Ho. Hybrid automata: An algorithmic approach to the specification and verification of hybrid systems. In Robert L. Grossman, Anil Nerode, Anders P. Ravn, and Hans Rischel, editors, *Hybrid Systems*, LNCS, pages 366–392. Springer Verlag, New York, 1993.

AA278A Lecture Notes 6
Spring 2005
Stability of Hybrid Systems

Claire J. Tomlin

April 21, 2005

Figure 1: Aleksandr Mikhailovich Lyapunov, 1857 - 1918

For the purpose of studying stability of hybrid automata, we will again drop reference to inputs (and outputs) of the system and focus on the state trajectory. Later, we will bring the inputs (and outputs) back in when we talk about stabilizing controllers.

1 Review of Stability for Continuous Systems

For reference to the following material, see [1, 2], Chapters 3 and 2 respectively.

Consider the following continuous system:

$$\dot{x} = f(x), \qquad x(0) = x_0 \tag{1}$$

where $f : \mathbb{R}^n \to \mathbb{R}^n$ is globally Lipschitz continuous.

- **Definition (Equilibrium of (1)):** $x = x_e$ is an *equilibrium point* of (1) if $f(x_e) = 0$. Without loss of generality in the following we will assume that $x_e = 0$.
- **Definition (Stability of (1)):** The equilibrium point $x_e = 0$ of (1) is *stable* (in the sense of Lyapunov) if for all $\epsilon > 0$ there exists a $\delta > 0$ such that

  $$||x_0|| < \delta \Rightarrow ||x(t)|| < \epsilon, \forall t \geq 0 \tag{2}$$

  where $x : [0, \infty) \to \mathbb{R}^n$ is the solution to (1), starting at x_0.

  The equilibrium point $x_e = 0$ of (1) is *asymptotically stable* if it is stable and δ can be chosen so that

  $$||x_0|| < \delta \Rightarrow \lim_{t \to \infty} ||x(t)|| = 0 \tag{3}$$

- More definitions for stability: exponentially stable, globally (asymptotically, exponentially) stable, locally (asymptotically, exponentially) stable, unstable ...
- Note that the definition of stability allows $x_e = 0$ to be stable without $x(t)$ converging to 0; note also that for system (1) with a single unstable equilibrium point, for all x_0 the solution can be bounded.

Consider a continuously differentiable (C^1) function $V : \mathbb{R}^n \to \mathbb{R}$. The rate of change of V along solutions of (1) is computed as:

$$\dot{V}(x) = \sum_{i=1}^{n} \frac{\partial V}{\partial x_i} \dot{x}_i = \sum_{i=1}^{n} \frac{\partial V}{\partial x_i} f_i(x) = \frac{\partial V}{\partial x} f(x) \tag{4}$$

This function is denoted the *Lie derivative* of V with respect to f.

Theorem 1 (Lyapunov Stability Theorem) *Let $x_e = 0$ be an equilibrium point of $\dot{x} = f(x), x(0) = x_0$ and $D \subset \mathbb{R}^n$ a set containing $x_e = 0$. If $V : D \to \mathbb{R}$ is a C^1 function such that*

$$V(0) = 0 \tag{5}$$
$$V(x) > 0, \forall x \in D \backslash \{0\} \tag{6}$$
$$\dot{V}(x) \leq 0, \forall x \in D \tag{7}$$

then x_e is stable. Furthermore, if $x_e = 0$ is stable and

$$\dot{V}(x) < 0, \forall x \in D \backslash \{0\} \tag{8}$$

then x_e is asymptotically stable.

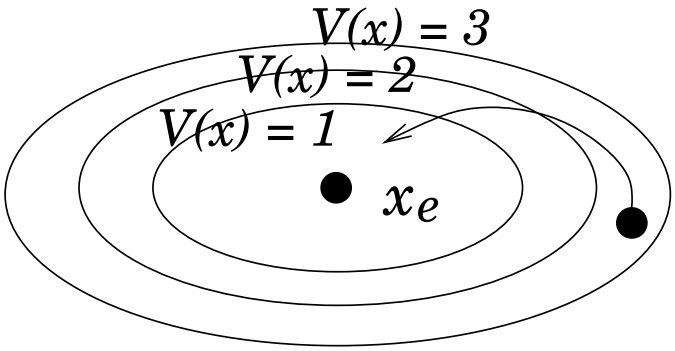


Figure 2: Level sets $V(x) = 1$, $V(x) = 2$, and $V(x) = 3$ for a Lyapunov function V; thus if a state trajectory enters one of these sets, it has to stay inside it since $\dot{V}(x) \leq 0$.

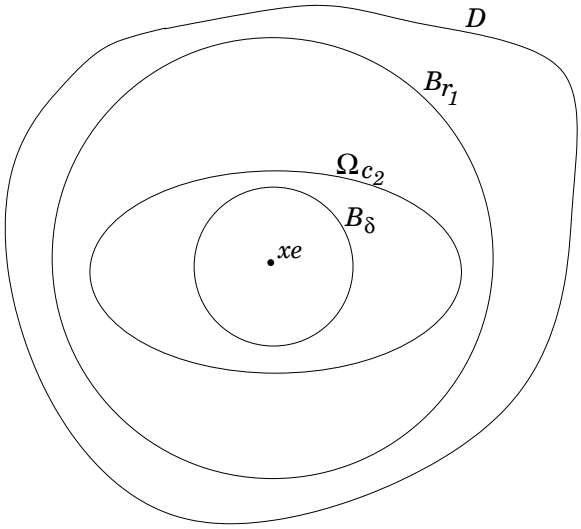


Figure 3: Figure for Proof of Lyapunov Stability Theorem (for continuous systems); WLOG $x_e = 0$.

Note that the Lyapunov function defines level sets $\{x \in \mathbb{R}^n : V(x) \leq c\}$ for $c > 0$ (see Figure 2). If a state trajectory enters one of these sets, it has to stay inside it, since $\dot{V}(x) \leq 0$ implies that if $V(x) = c$ at $t = t_0$, then $V(x(t)) \leq V(x(0)) \leq c$, $\forall t \geq t_0$.

Proof: For stability, we need to prove that for all $\epsilon > 0$ there exists $\delta > 0$ such that:

$$||x_0|| < \delta \Rightarrow ||x(t)|| < \epsilon, \forall t \geq 0 \tag{9}$$

We will use the following notation: for any $r > 0$, let $B_r = \{x \in \mathbb{R}^n : ||x|| < r\}$, $S_r = \{x \in \mathbb{R}^n : ||x|| = r\}$, and $\Omega_r = \{x \in \mathbb{R}^n : V(x) < r\}$.

See Figure 3. Choose $r_1 \in (0, \epsilon)$ such that $B_{r_1} \subseteq D$ (we do this because there is no guarantee that $B_\epsilon \subseteq D$). Let $c_1 = \min_{x \in S_{r_1}} V(x)$. Choose $c_2 \in (0, c_1)$. Note that there is no guarantee that $\Omega_{c_2} \subset B_{r_1}$. *Why not?* However, if $\delta > 0$ is chosen so that $B_\delta \subseteq \Omega_{c_2}$, then $V(x_0) < c_2$. Since V is non-increasing along system executions, executions that start inside B_δ cannot leave Ω_{c_2}. Thus for all $t > 0$ we have $x(t) \in B_{r_1} \subset B_\epsilon$. Thus $||x(t)|| \leq \epsilon$ for all $t > 0$. ∎

- **Example (Pendulum)** Consider the pendulum, unit mass, unit length, where x_1 is the angle of the pendulum with the vertical, and x_2 is the angular velocity of the pendulum.

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= -g \sin x_1 \end{aligned}$$

To show that $x_e = 0$ (pendulum pointing downwards) is a stable equilibrium, consider the candidate Lyapunov function:

$$V(x) = g(1 - \cos x_1) + \frac{x_2^2}{2} \tag{10}$$

defined over the set $\{x \in \mathbb{R}^n : -\pi < x_1 < \pi\}$. Clearly, $V(0) = 0$, and $V(x) > 0, \forall x \in \{x \in \mathbb{R}^n : -\pi < x_1 < \pi\} \backslash \{0\}$. Also,

$$\dot{V}(x) = [g \sin x_1 \;\; x_2] \begin{bmatrix} x_2 \\ -g \sin x_1 \end{bmatrix} = 0$$

so the equilibrium point $x_e = 0$ is stable. Is it asymptotically stable?

- Finding Lyapunov functions in general is HARD. Often a solution is to try to use the *energy* of the system as a Lyapunov function (as in the example above). However, for linear systems, finding Lyapunov functions is easy: For a stable linear system $\dot{x} = Ax$, a Lyapunov function is given by $V(x) = x^T P x$, where P is a positive definite symmetric matrix which solves the *Lyapunov Equation* $A^T P + PA = -I$. (Recall that a matrix P is said to be positive definite if $x^T P x > 0$ for all $x \neq 0$. It is called positive semidefinite if $x^T P x \geq 0$ for all $x \neq 0$.)

2 Stability of Hybrid Systems

Consider an autonomous hybrid automaton $H = (S, \text{Init}, f, \text{Dom}, R)$.

Definition 2 (Equilibrium of a Hybrid Automaton) *The continuous state $x_e = 0 \in \mathbb{R}^n$ is an equilibrium point of H if:*

1. *$f(q, 0) = 0$ for all $q \in \mathbf{Q}$, and*
2. *$R(q, 0) \subset Q \times 0$.*

- Thus, discrete transitions are allowed out of $(q, 0)$, as long as the system jumps to a (q', x) in which $x = x_e = 0$.
- It follows from the above definition that if $(q_0, 0) \in \text{Init}$ and $(\tau, (q, x))$ represents the hybrid execution starting at $(q_0, 0)$, then $x(t) = 0$ for all $t \in \tau$.

As we did for continuous systems, we would like to characterize the notion that if the continuous state starts close to the equilibrium point, it stays close, or converges, to it.

- **Definition (Stable Equilibrium of a Hybrid Automaton):** Let $x_e = 0$ be an equilibrium point of the hybrid automaton H. Then $x_e = 0$ is stable if for all $\epsilon > 0$ there exists $\delta > 0$ such that for all $(\tau, (q, x))$ starting at (q_0, x_0),

$$||x_0|| < \delta \Rightarrow ||x(t)|| < \epsilon, \forall t \in \tau \tag{11}$$

- **Definition (Asymptotically Stable Equilibrium of a Hybrid Automaton):** Let $x_e = 0 \in X$ be an equilibrium point of the hybrid automaton H. Then $x_e = 0$ is asymptotically stable if it is stable and δ can be chosen so that for all $(\tau, (q, x))$ starting at (q_0, x_0),

$$||x_0|| < \delta \Rightarrow \lim_{t \to \tau_\infty} ||x(t)|| = 0 \tag{12}$$

- **Remark:** In the above, $(\tau, (q, x))$ is considered to be an infinite execution, with $\tau_\infty = \sum_i (\tau_i' - \tau_i)$. Notice that $\tau_\infty < \infty$ if χ is Zeno and $\tau_\infty = \infty$ otherwise.

One would expect that a hybrid system for which each discrete state's continuous system is stable would be stable, at least if $R(q, x) \in Q \times \{x\}$ for all x. But this is NOT NECESSARILY the case:

- **Example:** Consider the hybrid automaton H with:
  - $Q = \{q_1, q_2\}$, $X = \mathbb{R}^2$
  - Init $= Q \times \{x \in X : ||x|| > 0\}$
  - $f(q_1, x) = A_1 x$ and $f(q_2, x) = A_2 x$, with:

$$A_1 = \begin{bmatrix} -1 & 10 \\ -100 & -1 \end{bmatrix}, A_2 = \begin{bmatrix} -1 & 100 \\ -10 & -1 \end{bmatrix}$$

  - Dom $= \{q_1, \{x \in \mathbb{R}^2 : x_1 x_2 \leq 0\}\} \cup \{q_2, \{x \in \mathbb{R}^2 : x_1 x_2 \geq 0\}\}$
  - $R(q_1, \{x \in \mathbb{R}^2 : x_1 x_2 \geq 0\}) = (q_2, x)$ and $R(q_2, \{x \in \mathbb{R}^2 : x_1 x_2 \leq 0\}) = (q_1, x)$
- **Remark 1:** Since $f(q_1, 0) = f(q_2, 0) = 0$ and $R(q_1, 0) = (q_2, 0), R(q_2, 0) = (q_1, 0)$, $x_e = 0$ is an equilibrium of H.
- **Remark 2:** Since the eigenvalues of both systems are at $-1 \pm j\sqrt{1000}$, the continuous systems $\dot{z} = A_i x$ for $i = 1, 2$ are asymptotically stable. (See phase portraits for each in Figure 4.)
- **Remark 3:** $x_e = 0$ is unstable for H! If the switching is flipped, then $x_e = 0$ is stable! (See phase portraits for each in Figure 5.)

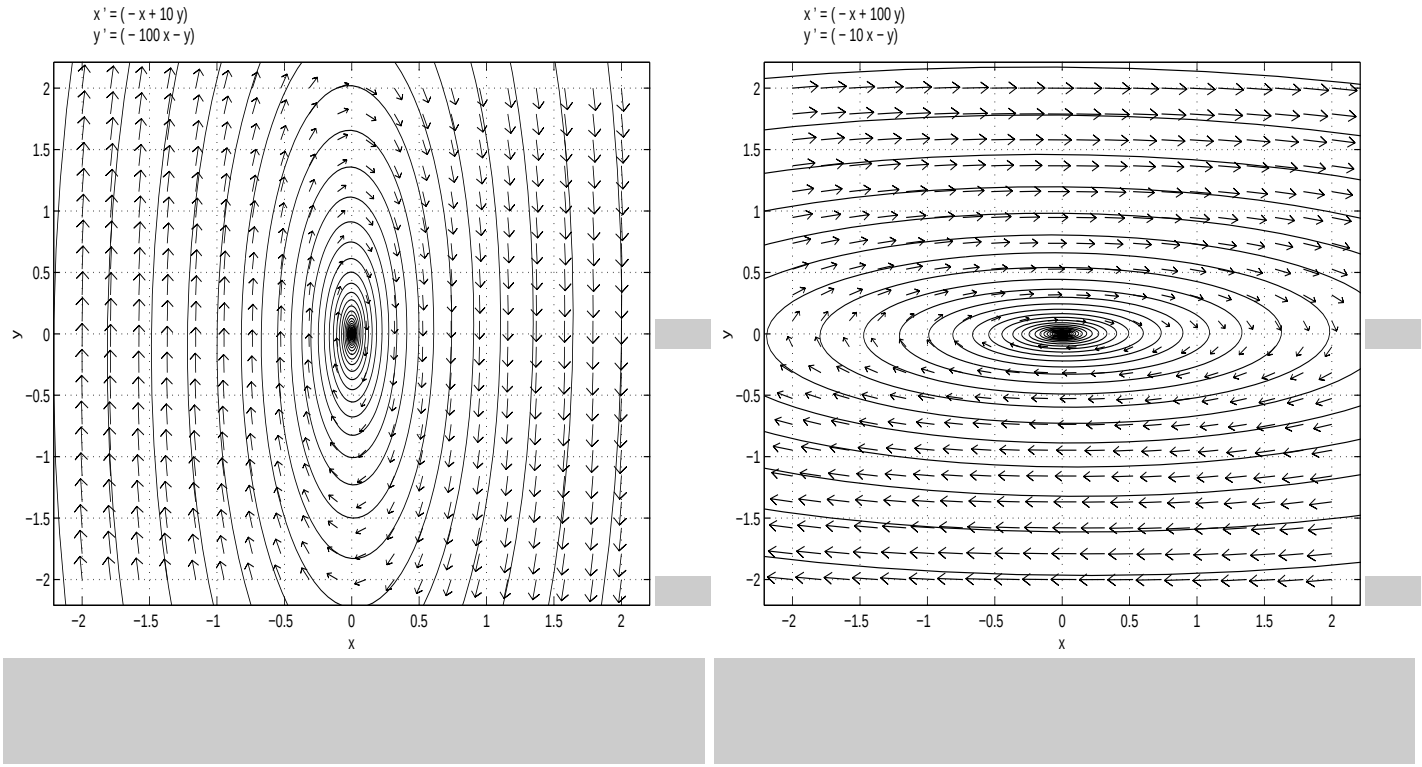


Figure 4: (a) Phase portrait of $\dot{x} = A_1 x$; (b) Phase portrait of $\dot{x} = A_2 x$. Figure generated using phase plane software from `http://math.rice.edu/polking/odesoft/`, freely downloadable.

- **Remark 4:** This simple example (drawn from [3]) shows that in general we cannot expect to analyze the stability of a hybrid system by studying the continuous components separately.

Theorem 3 (Lyapunov Stability Theorem (for hybrid systems)) *Consider a hybrid automaton H with $x_e = 0$ an equilibrium point, and $R(q, x) \in Q \times \{x\}$. Assume that there exists an open set $D \subset Q \times \mathbb{R}^n$ such that $(q, 0) \in D$ for some $q \in Q$. Let $V : D \to \mathbb{R}$ be a C^1 function in its second argument such that for all $q \in Q$:*

1. *$V(q, 0) = 0$;*
2. *$V(q, x) > 0$ for all x, $(q, x) \in D \backslash \{0\}$, and*
3. *$\frac{\partial V(q,x)}{\partial x} f(q, x) \leq 0$ for all x, $(q, x) \in D$*

If for all (τ, q, x) starting at (q_0, x_0) where $(q_0, x_0) \in \text{Init} \cap D$, and all $q' \in Q$, the sequence $\{V(q(\tau_i), x(\tau_i)) : q(\tau_i) = q'\}$ is non-increasing (or empty), then $x_e = 0$ is a stable equilibrium of H.

- We call such function "Lyapunov-like" (see Figure 6).

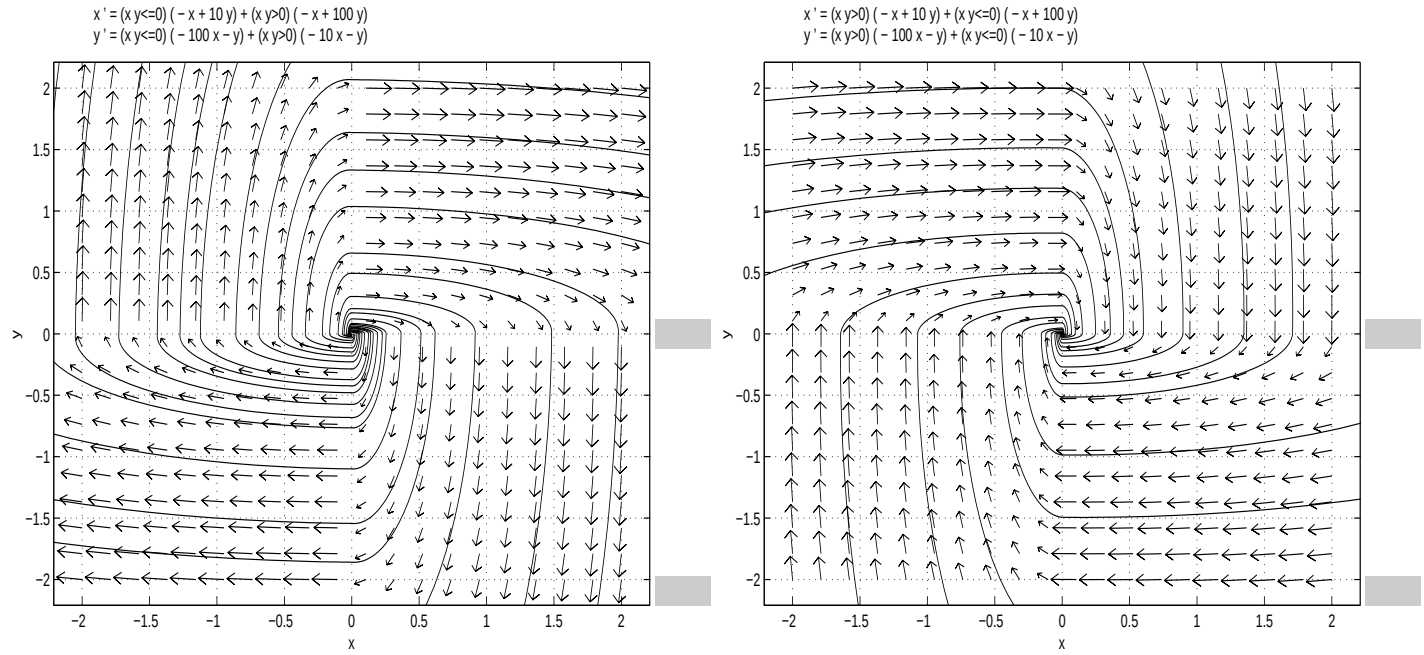


Figure 5: (a) Phase portrait of H; (b) Phase portrait of H, switching conditions flipped.

- A drawback of this Theorem is that the sequence $\{V(q(\tau_i), x(\tau_i))\}$ must be evaluated (which may require integrating the vector field and then we lose the fundamental advantage of Lyapunov theory)
- Also, it is in general difficult to derive such a function V
- HOWEVER, for certain classes of hybrid automata, which have vector fields linear in x, computationally attractive methods exist to derive Lyapunov-like functions V

Proof: We sketch the proof for $Q = \{q_1, q_2\}$ and $(q, \cdot) \notin R(q, \cdot)$. Define the sets in Figure 7 similar to the previous proof, ie.

$$\Omega_{c_{2_i}} = \{x \in B_{r_{1_i}} : V(q_i, x) < c_{2_i}\} \tag{13}$$

where $c_{2_i} \in (0, c_{1_i})$ where $c_{1_i} = \min_{x \in S_{r_{1_i}}} V(q_i, x)$. Now let $r = \min\{\delta_1, \delta_2\}$, and inside B_r, in each of q_1 and q_2, repeat the construction above, ie.

$$\Omega_{\overline{c}_{2_i}} = \{x \in B_r : V(q_i, x) < \overline{c}_{2_i}\} \tag{14}$$

where $\overline{c}_{2_i} \in (0, \min_{x \in S_r} V(q_i, x))$. Also, let $B_{\overline{\delta}_i} \subset \Omega_{\overline{c}_{2_i}}$. Let $\delta = \min\{\overline{\delta}_1, \overline{\delta}_2\}$. Consider the hybrid trajectory (τ, q, x) starting at $x_0 \in B_\delta$, and assume that the initial discrete state q_0 is equal to q_1. By the corresponding continuous Lyapunov theorem, $x(t) \in \Omega_{\overline{c}_{2_1}}$ for $t \in [\tau_0, \tau_0']$. Therefore, $x(\tau_1) = x(\tau_0') \in \Omega_{\overline{c}_{2_2}}$ (where equality holds because of the restricted

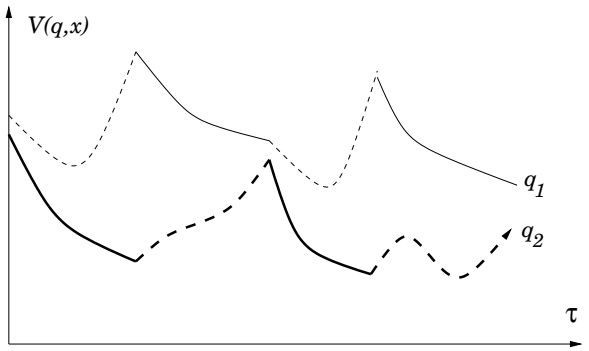


Figure 6: Showing $V(q_1, x)$ and $V(q_2, x)$. Solid segments on q_i mean that the system is in q_i at that time, dotted segments mean the system is in q_j, $j \neq i$.

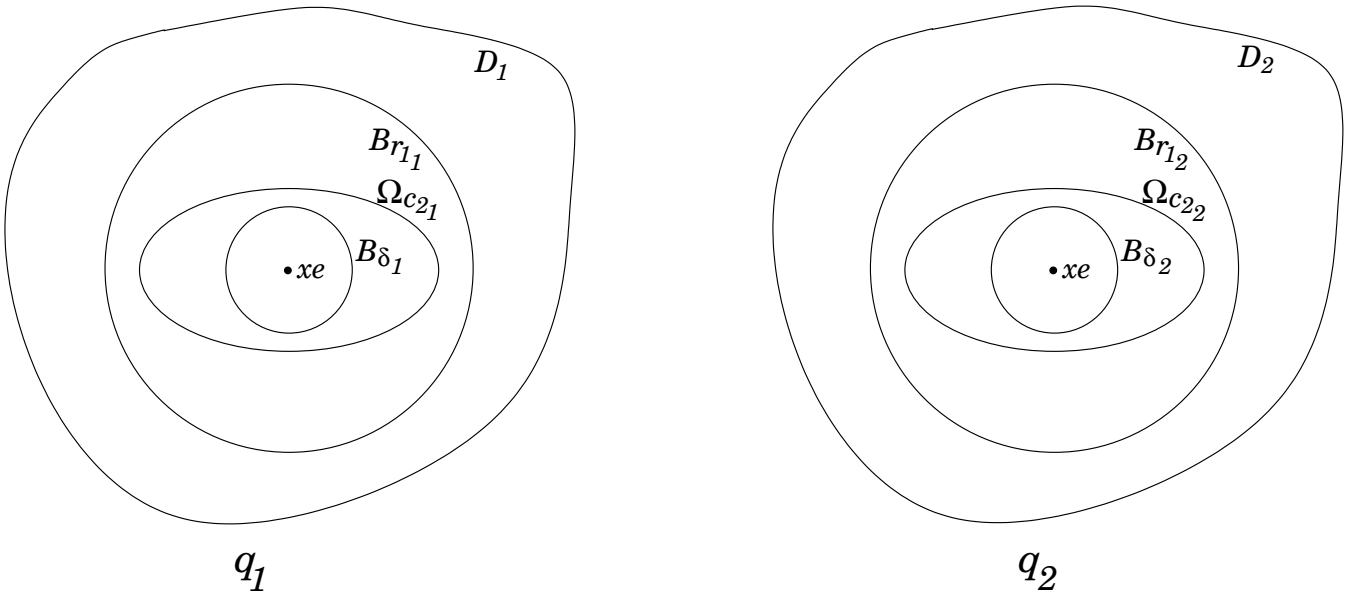


Figure 7: Figure for Proof of Lyapunov Stability Theorem (for hybrid systems).

definition of the transition map). By the continuous Lyapunov Theorem again, $x(t) \in \Omega_{\bar{c}_{2_2}}$ and thus $x(t) \in B_\epsilon$ for $t \in [\tau_1, \tau_1']$. By the assumption of the non-increasing sequence, $x(\tau_1') = x(\tau_2) \in \Omega_{\bar{c}_{2_1}}$. The result follows by induction. ∎

Corollary 4 (A more restrictive Lyapunov Stability Theorem (for hybrid systems)) *Consider a hybrid automaton H with $x_e = 0$ an equilibrium point, and $R(q, x) \in Q \times \{x\}$. Assume that there exists an open set $D \subset \mathbb{R}^n$ such that $0 \in D$. Let $V : D \to \mathbb{R}$ be a C^1 function such that for all $q \in Q$:*

1. *$V(0) = 0$;*
2. *$V(x) > 0$ for all $x \in D \backslash \{0\}$, and*
3. *$\frac{\partial V(x)}{\partial x} f(q, x) \leq 0$ for all $x \in D$*

Then $x_e = 0$ is a stable equilibrium of H.

Proof: Define $\hat{V} : \mathbf{Q} \times \mathbb{R}^n \to \mathbb{R}$ by $\hat{V}(q, x) = V(x)$ for all $q \in \mathbf{Q}$, $x \in \mathbb{R}^n$ and apply Theorem 3. ∎

3 Lyapunov Stability for Piecewise Linear Systems

Theorem 5 (Lyapunov Stability for Linear Systems) *The equilibrium point $x_e = 0$ of $\dot{x} = Ax$ is asymptotically stable if and only if for all matrices $Q = Q^T > 0$ there exists a unique matrix $P = P^T > 0$ such that*

$$PA + A^T P = -Q \tag{15}$$

Proof: For the "if" part of the proof, consider the Lyapunov function (from Lecture Notes 4) $V(x) = x^T P x$. Thus,

$$\dot{V} = x^T P \dot{x} + \dot{x}^T P x = x^T (PA + A^T P) x) = -x^T Q x < 0 \tag{16}$$

For the "only if" part of the proof, consider

$$P = \int_0^\infty e^{A^T t} Q e^{At} dt \tag{17}$$

which is well-defined since $\mathrm{Re}(\lambda_i(A)) < 0$. Clearly,

$$\begin{aligned} PA + A^T P &= \int_0^\infty e^{A^T t} Q e^{At} A dt + \int_0^\infty A^T e^{A^T t} Q e^{At} dt & (18) \\ &= \int_0^\infty \frac{d}{dt} e^{A^T t} Q e^{At} dt = -Q & (19) \end{aligned}$$

Also, P is unique: to prove this, assume there exists another solution $\hat{P} \neq P$. Then,

$$\begin{aligned} 0 &= e^{A^T t}(Q - Q)e^{At} & (20) \\ &= e^{A^T t}[(P - \hat{P})A + A^T(P - \hat{P})]e^{At} & (21) \\ &= \frac{d}{dt} e^{A^T t}(P - \hat{P})e^{At} & (22) \end{aligned}$$

which means that $e^{A^T t}(P - \hat{P})e^{At}$ is constant for all $t \geq 0$. Thus,

$$e^{A^T 0}(P - \hat{P})e^{A0} = \lim_{t \to \infty} e^{A^T t}(P - \hat{P})e^{At} \tag{23}$$

thus $P = \hat{P}$, which contradicts the assumption and thus concludes the proof. ∎

- Equation (15) is called a Lyapunov equation. We may also write the matrix condition in the Theorem of Lyapunov Stability for Linear Systems as the following inequality:

$$A^T P + PA < 0 \tag{24}$$

which is called a *linear matrix inequality* (LMI), since the left hand side is linear in the unknown P.

Example: Switched Linear System, Revisited.

Consider the linear hybrid system example from page 5 (with the switching flipped):

- $Q = \{q_1, q_2\}$, $X = \mathbb{R}^2$
- Init $= Q \times \{x \in X : ||x|| > 0\}$
- $f(q_1, x) = A_1 x$ and $f(q_2, x) = A_2 x$, with:

$$A_1 = \begin{bmatrix} -1 & 10 \\ -100 & -1 \end{bmatrix}, A_2 = \begin{bmatrix} -1 & 100 \\ -10 & -1 \end{bmatrix}$$

- Dom $= \{q_1, \{x \in \mathbb{R}^2 : x_1 x_2 \geq 0\}\} \cup \{q_2, \{x \in \mathbb{R}^2 : x_1 x_2 \leq 0\}\}$
- $R(q_1, \{x \in \mathbb{R}^2 : x_1 x_2 \leq 0\}) = (q_2, x)$ and $R(q_2, \{x \in \mathbb{R}^2 : x_1 x_2 \geq 0\}) = (q_1, x)$

Proposition 6 *$x = 0$ is an equilibrium of H.*

Proof: $f(q_1, 0) = f(q_2, 0) = 0$ and $R(q_1, 0) = (q_2, 0), R(q_2, 0) = (q_1, 0)$. ∎

Proposition 7 *The continuous systems $\dot{x} = A_i x$ for $i = 1, 2$ are asymptotically stable.*

Recall that $P_i > 0$ (positive definite) if and only if $x^T P_i x > 0$ for all $x \neq 0$, and I is the identity matrix.

Consider the candidate Lyapunov function:

$$V(q, x) = \begin{cases} x^T P_1 x & \text{if } q = q_1 \\ x^T P_2 x & \text{if } q = q_2 \end{cases}$$

Check that the conditions of the Theorem hold. For all $q \in \mathbf{Q}$:

1. $V(q, 0) = 0$
2. $V(q, x) > 0$ for all $x \neq 0$ (since the P_i are positive definite)

3. $\frac{\partial V}{\partial x}(q,x)f(q,x) \leq 0$ for all x since:

$$\begin{aligned}\frac{\partial V}{\partial x}(q,x)f(q,x) &= \frac{d}{dt}V(q,x(t)) \\ &= \dot{x}^T P_i x + x^T P_i \dot{x} \\ &= x^T A_i^T P_i x + x^T P_i A_i x \\ &= x^T (A_i^T P_i + P_i A_i) x \\ &= -x^T I x \\ &= -\|x\|^2 \leq 0\end{aligned}$$

It remains to test the non-increasing sequence condition. Notice that the level sets of $x^T P_i x$ are ellipses centered at the origin. Therefore each level set intersects the switching line $x_i = 0$ (for $i = 1, 2$) at exactly two points, $\hat{x}$ and $-\hat{x}$. Assume that $x(\tau_i) = \hat{x}$ and $q(\tau_i) = q_1$. The fact that $V(q_1, x(t))$ does not increase for $t \in [\tau_i, \tau_i']$ (where $q(t) = q_1$) implies that the next time a switching line is reached, $x(\tau_i') = \alpha(-\hat{x})$ for some $\alpha \in (0, 1]$. Therefore, $\|x(\tau_{i+1})\| = \|x(\tau_i')\| \leq \|x(\tau_i)\|$. By a similar argument, $\|x(\tau_{i+2})\| = \|x(\tau_{i+1}')\| \leq \|x(\tau_{i+1})\|$. Therefore, $V(q(\tau_i), x(\tau_i)) \leq V(q(\tau_{i+2}), x(\tau_{i+2}))$.

3.1 Globally Quadratic Lyapunov Function

The material in this section and the next is drawn from the work of Mikael Johansson [4, 5], and also from [6, 7, 8].

Theorem 8 (Globally Quadratic Lyapunov Function) *Consider a hybrid automaton $H = (S, \text{Init}, f, \text{Dom}, R)$ with equilibrium $x_e = 0$. Assume that for all i:*

- $f(q_i, x) = A_i x, A_i \in \mathbb{R}^{n \times n}$
- $\text{Init} \subseteq \text{Dom}$
- *for all $x \in \mathbb{R}^n$*

$$|R(q_i, x)| = \begin{cases} 1 & \text{if } (q_i, x) \in \partial\text{Dom} \\ 0 & \text{otherwise} \end{cases} \tag{25}$$

and $\{(q', x') \in R(q_i, x)\} \Rightarrow \{(q', x') \in \text{Dom}, x' = x\}$. Furthermore, assume that for all $\chi \in \mathcal{E}_H^\infty$, $\tau_\infty(\chi) = \infty$. Then, if there exists $P = P^T > 0$ such that

$$A_i^T P + P A_i < 0, \forall i \tag{26}$$

$x_e = 0$ is asymptotically stable.

Proof: First note that there exists $\gamma > 0$ such that

$$A_i^T P + P A_i + \gamma I \leq 0, \forall i \tag{27}$$

Also, note that with the given assumptions there exists a unique, infinite, and non-Zeno execution $\chi = (\tau, q, x)$ for every $(q_0, x_0) \in \text{Init}$. For all $i \geq 0$, the continuous evolution $x : \tau \rightarrow \mathbb{R}^n$ of such an execution satisfies the following time-varying linear differential equation:

$$\dot{x}(t) = \sum_i \mu_i(t) A_i x(t), t \in [\tau_i, \tau_i'] \tag{28}$$

where $\mu_i : \tau \rightarrow [0,1]$ is a function such that for $t \in [\tau_i, \tau_i']$, $\sum_i \mu_i(t) = 1$. Letting $V(q,x) = x^T P x$, we have that for $t \in [\tau_i, \tau_i']$,

$$\begin{aligned} \dot{V}(q(t), x(t)) &= \sum_i [\mu_i(t) x(t)^T (A_i^T P + P A_i) x(t)] & (29) \\ &\leq -\gamma ||x(t)||^2 \sum_i \mu_i(t) & (30) \\ &= -\gamma ||x(t)||^2 & (31) \end{aligned}$$

Now, since $V(q,x) = x^T P x$, we have that

$$\lambda_{min} ||x||^2 \leq V(q,x) \leq \lambda_{max} ||x||^2 \tag{32}$$

where $0 < \lambda_{min} \leq \lambda_{max}$ are the smallest and largest eigenvalues of P respectively. It follows that

$$\dot{V}(q(t), x(t)) \leq -\frac{\gamma}{\lambda_{max}} V(q(t), x(t)), t \in [\tau_i, \tau_i'] \tag{33}$$

and hence

$$V(q(t), x(t)) \leq V(q(\tau_i), x(\tau_i)) e^{-\gamma(t-\tau_i)/\lambda_{max}}, t \in [\tau_i, \tau_i'] \tag{34}$$

Thus, from (32):

$$\lambda_{min} ||x(t)||^2 \leq \lambda_{max} ||x(\tau_i)||^2 e^{-\gamma(t-\tau_i)/\lambda_{max}}, t \in [\tau_i, \tau_i'] \tag{35}$$

Since the execution χ by assumption is non-Zeno, we have that $\tau_i \rightarrow \infty$ as $i \rightarrow \infty$. Hence, $||x(t)||$ goes to zero exponentially as $t \rightarrow \tau_\infty$, which implies that the equilibrium point $x_e = 0$ is asymptotically (actually exponentially) stable. ∎

Example 1: Consider the hybrid automaton H of Figure 8, with

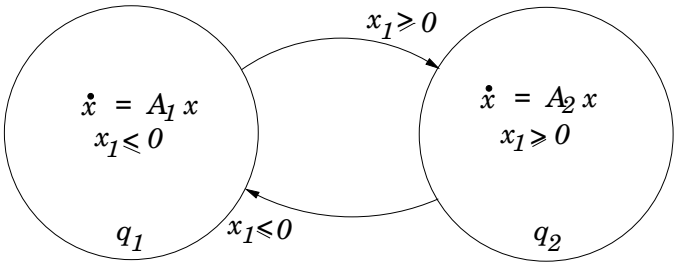


Figure 8: Example 1

$$A_1 = \begin{bmatrix} -1 & 1 \\ -1 & -1 \end{bmatrix}, A_2 = \begin{bmatrix} -2 & 1 \\ -1 & -2 \end{bmatrix} \tag{36}$$

Since the eigenvalues of A_1 are $\lambda(A_1) = \{-1 \pm i\}$ and of A_2 are $\lambda(A_2) = \{-2 \pm i\}$, both $\dot{x} = A_1 x$ and $\dot{x} = A_2 x$ have an asymptotically stable focus. H satisfies the assumptions of the previous theorem; indeed, $A_1^T + A_1 < 0$ and $A_2^T + A_2 < 0$, so the inequalities in the theorem are satisfied for $P = I$. Hence, the origin is an asymptotically stable equilibrium point for H.

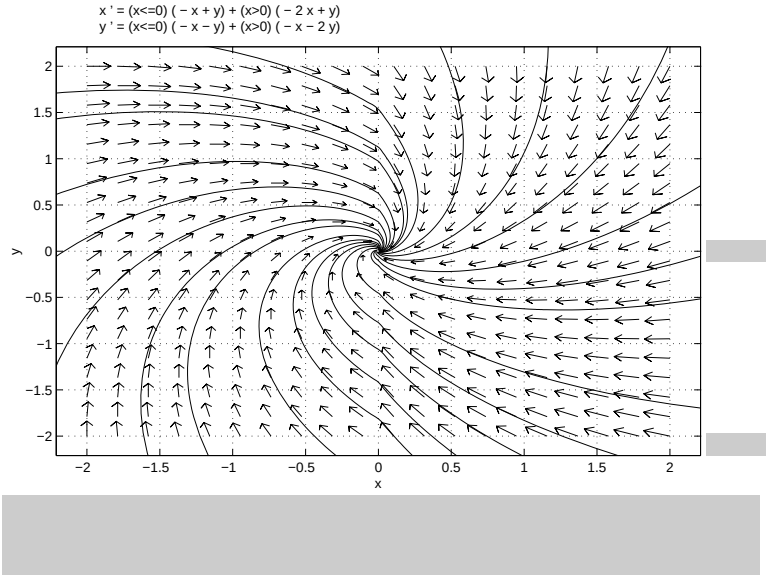


Figure 9: Example 1

Example 2 [5]: Consider the hybrid automaton from Figure 1 again, but let

$$A_1 = \begin{bmatrix} -5 & -4 \\ -1 & -2 \end{bmatrix}, A_2 = \begin{bmatrix} -2 & -4 \\ 20 & -2 \end{bmatrix} \tag{37}$$

The eigenvalues of A_1 are $\lambda(A_1) = \{-6, -1\}$ and of A_2 are $\lambda(A_2) = \{-2 \pm 4\sqrt{5}i\}$ so that $\dot{x} = A_1 x$ has an asymptotically stable node and $\dot{x} = A_2 x$ has an asymptotically stable focus. The evolution of the continuous state is shown in Figure 10 for four different initial states. The origin seems to be a stable equilibrium point – indeed, the Lyapunov function indicated by the dashed level sets proves asymptotic stability of the origin. Yet from the shape of its level sets, we see that the Lyapunov function is not globally quadratic, but piecewise quadratic, in the sense that it is quadratic in each discrete mode. (In fact, we can show for this example that it is not possible to find a quadratic Lyapunov function).

3.2 Piecewise Quadratic Lyapunov Function

If we assume that the hybrid automaton is restricted further so that the domains are given by polyhedra, then we can make some more general statements about the stability of the hybrid system:

$$\text{Dom} = \cup_i \{q_i\} \times \{x \in \mathbb{R}^n : E_{i1} x \geq 0, \ldots, E_{in} x \geq 0\} \tag{38}$$

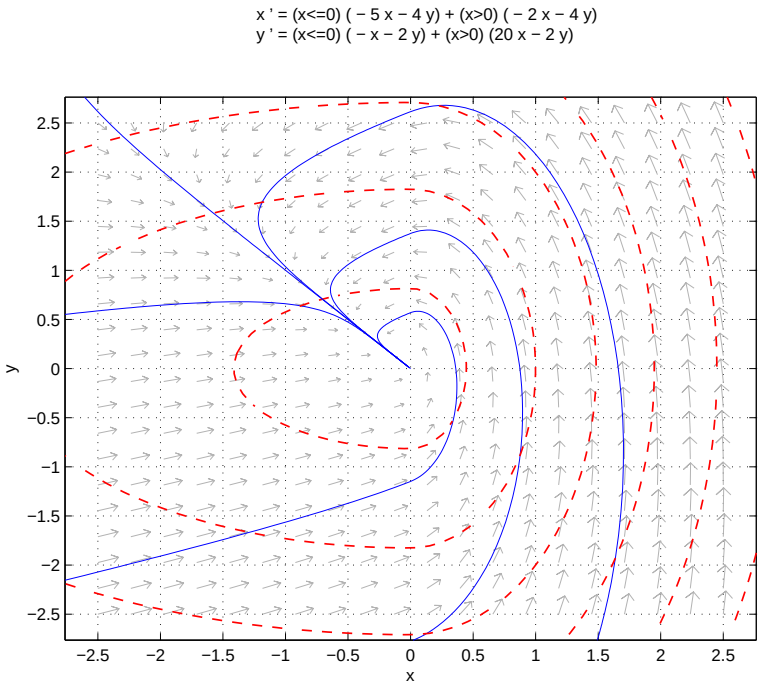


Figure 10: Continuous evolution for a hybrid automaton that does not have a globally quadratic Lyapunov function. Still, the origin is an asymptotically stable equilibrium point, which can be proved by using a Lyapunov function quadratic in each discrete state.

where

$$E_i = \begin{bmatrix} E_{i1} \\ \vdots \\ E_{in} \end{bmatrix} \in \mathbb{R}^{n \times n} \tag{39}$$

It then follows that $(q_i, 0) \in \text{Dom}$ for all i. Suppose that the reset relation R satisfies:

$$|R(q_i, x)| = \begin{cases} 1 & \text{if } (q_i, x) \in \partial\text{Dom} \\ 0 & \text{otherwise} \end{cases} \tag{40}$$

such that

$$(q_k, x') \in R(q_i, x) \Rightarrow F_k x = F_i x, q_k \neq q_i, x' = x \tag{41}$$

where $F_k, F_i \in \mathbb{R}^{m \times n}$ are given matrices (which hence define the boundaries of Dom). The LMI condition in Theorem 1 would require that

$$x^T (A_i^T P + P A_i) x < 0, \forall x \neq 0, (q_i, x) \in Q \times \mathbb{R}^n \tag{42}$$

It is however sufficient to require that

$$x^T (A_i^T P + P A_i) x < 0, \forall x \neq 0, (q_i, x) \in \text{Dom} \tag{43}$$

This can be done by specifying a matrix S_i such that $x^T S_i x \geq 0$ for all x with $(q_i, x) \in \text{Dom}$. Then,

$$A_i^T P + P A_i + S_i < 0 \tag{44}$$

still implies that

$$x^T(A_i^T P + PA_i)x < 0, \forall x \neq 0, (q_i, x) \in \text{Dom} \tag{45}$$

but $x^T(A_i^T P + PA_i)x < 0$ does not have to hold for $x \neq 0$ with $(q_k, x) \in \text{Inv}$ and $i \neq k$. The matrix S_i may be given as $S_i = E_i^T U_i E_i$, where E_i is given by the representation of H and $U_i = U_i^T \in \mathbb{R}^{n \times n}$ is chosen to have non-negative elements.

We can also let V depend on the discrete state, ie. $V(q_i, x) = x^T P_i x$ for $(q_i, x) \in \text{Inv}$. We choose $P_i = F_i^T M F_i$, where F_i is given by the representation of H, and $M = M^T \in \mathbb{R}^{n \times n}$ is to be chosen.

Theorem 9 (Piecewise Quadratic Lyapunov Function) *$H = (S, \text{Init}, f, \text{Dom}, R)$ with equilibrium $x_e = 0$. Assume that for all i:*

- $f(q_i, x) = A_i x, A_i \in \mathbb{R}^{n \times n}$
- $\text{Dom} = \cup_i \{q_i\} \times \{x \in \mathbb{R}^n : E_{i1} x \geq 0, \ldots, E_{in} x \geq 0\}$
- $\text{Init} \subseteq \text{Dom}$
- *for all $x \in \mathbb{R}^n$*

$$|R(q_i, x)| = \begin{cases} 1 & \text{if } (q_i, x) \in \partial\text{Dom} \\ 0 & \text{otherwise} \end{cases} \tag{46}$$

such that

$$(q_k, x') \in R(q_i, x) \Rightarrow F_k x = F_i x, q_k \neq q_i, x' = x \tag{47}$$

where $F_k, F_i \in \mathbb{R}^{n \times n}$.

Furthermore, assume that for all $\chi \in \mathcal{E}_H^\infty$, $\tau_\infty(\chi) = \infty$. Then, if there exists $U_i = U_i^T$, $W_i = W_i^T$, and $M = M^T$ such that $P_i = F_i^T M F_i$ satisfies:

$$A_i^T P_i + P_i A_i + E_i^T U_i E_i < 0 \tag{48}$$

$$P_i - E_i^T W_i E_i > 0 \tag{49}$$

where U_i, W_i are non-negative, then $x_e = 0$ is asymptotically stable.

Example 3: Consider the hybrid automaton of Figure 11 with

$$A_1 = A_3 = \begin{bmatrix} -0.1 & 1 \\ -5 & -0.1 \end{bmatrix}, A_2 = A_4 = \begin{bmatrix} -0.1 & 5 \\ -1 & -0.1 \end{bmatrix} \tag{50}$$

Here, we may choose

$$E_1 = -E_3 = \begin{bmatrix} -1 & 1 \\ -1 & -1 \end{bmatrix}, E_2 = -E_4 = \begin{bmatrix} -1 & 1 \\ 1 & 1 \end{bmatrix} \tag{51}$$

and

$$F_i = \begin{bmatrix} E_i \\ I \end{bmatrix} \forall i \in \{1, 2, 3, 4\} \tag{52}$$

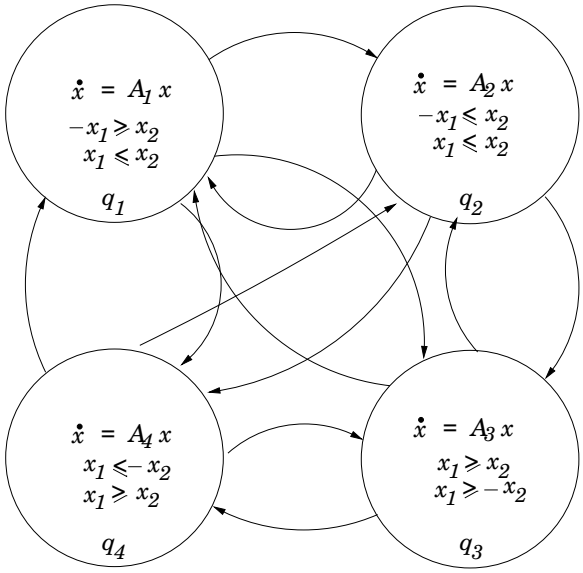


Figure 11: Example 3

The eigenvalues of A_i are $-1/10 \pm \sqrt{5}i$. The evolution of the continuous state is shown in Figure 12. We can use a Lyapunov function given by:

$$P_1 = P_3 = \begin{bmatrix} 5 & 0 \\ 0 & 1 \end{bmatrix}, P_2 = P_4 = \begin{bmatrix} 1 & 0 \\ 0 & 5 \end{bmatrix} \tag{53}$$

to prove asymptotic stability of the hybrid automaton.

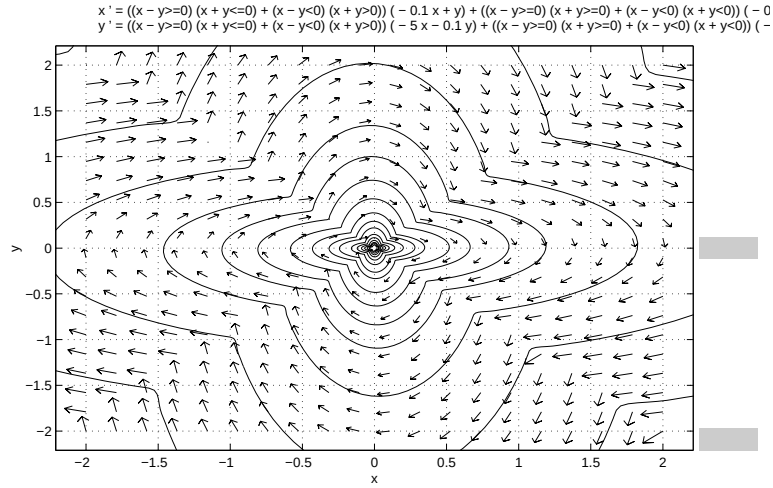


Figure 12: Example 3

3.3 Linear Matrix Inequalities

LMIs appear in many problems in systems and control theory (for example, see the reference [9]). For example, in the last Theorem we would like to solve

$$E_i^T M E_i > 0 \tag{54}$$
$$A_i^T P_i + P_i A_i + E_i^T U_i E_i < 0 \tag{55}$$
$$P_i - E_i^T W_i E_i > 0 \tag{56}$$

for the unknowns M, U_i, and W_i. This problem may be cast as an optimization problem, which turns out to be convex, so that it can be efficiently solved. In MATLAB, there is a toolbox called LMI Control Toolbox:

```
>> help lmilab
```

Try the demo:

```
>> help lmidem
```

and you'll see there is a graphical user interface to specify LMIs; you can enter this directly through:

```
lmiedit
```

After specifying an LMI, ie.:

$$P = P^T \succ 0 \tag{57}$$
$$A^T P + P A \prec 0 \tag{58}$$

where A is a matrix that you've already entered in MATLAB's workspace in `lmisys`, a feasible solution P is found (if it exists) by running the commands:

```
[tmin, pfeas] = feasp(lmisys)
p = dec2mat(lmisys,pfeas,p)
```

The feasibility problem is solved as a convex optimization problem, which has a global minimum. This means that if the feasibility problem has a solution, the optimization algorithm will (at least theoretically) always find it.

References

[1] S. S. Sastry. *Nonlinear Systems: Analysis, Stability and Control*. Springer Verlag, New York, 1999.

[2] H. K. Khalil. *Nonlinear Systems*. Prentice-Hall, 3rd edition, 2002.

[3] M. S. Branicky. Stability of switched and hybrid systems. In *Proceedings of the IEEE Conference on Decision and Control*, pages 3498–3503, 1994.

[4] M. Johansson and A. Rantzer. Computation of piecewise quadratic lyapunov functions for hybrid systems. *IEEE Transactions on Automatic Control*, 43(4):555–559, 1998.

[5] Mikael Johansson. *Piecewise linear control systems*. PhD thesis, Lund University, 1998.

[6] Karl Henrick Johansson. Lecture notes for eecs291e (hybrid systems). University of California at Berkeley, 2000.

[7] H. Ye, A. Michel, and L. Hou. Stability theory for hybrid dynamical systems. *IEEE Transactions on Automatic Control*, 43(4):461–474, April 1998.

[8] R. A. Decarlo, M. S. Branicky, S. Pettersson, and B. Lennartson. Perspectives and results on the stability and stabilizability of hybrid systems. *Proceedings of the IEEE*, 88(7):1069–1082, July 2000.

[9] S. Boyd, L. El Ghaoui, E. Feron, and V. Balakrishnan. *Linear Matrix Inequalities in System and Control Theory*. SIAM, Philadelphia, 1994.

AA278A Lecture Notes 7. Controller Synthesis for Hybrid Systems I: Introduction to Discrete Games

Claire J. Tomlin

May 18, 2005

This lecture is first in a series of lectures on designing control laws for hybrid systems. What is a control law? It is a method by which we steer the automaton using input variables, so that the closed loop exhibits certain desirable properties. A control problem involves:

1. A plant
2. A specification
3. A controller

Controller synthesis involves coming up with a methodology for designing controllers that meet the specification. The discussion in this lecture is based on [1] and [2, 3]. Here, we will deal mostly with *safety* specifications; although the methodology in this part of the course can be extended to a much more general class of properties.

1 Controlled Hybrid Automata

1.1 Fundamental Definitions

We augment the autonomous hybrid automaton definition of Lecture Notes 3 (page 4) with inputs and outputs:

Definition 1 (Hybrid Automaton) *A hybrid automaton H is a collection $H = (Q, X,$ Init, In, $f,$ Dom, $R,$ Out), where*

- $Q = \{q_1, q_2, \ldots\}$ *is a set of* discrete states*;*
- $X = \mathbb{R}^n$ *is a set of* continuous states*;*
- Init $\subseteq Q \times X$ *is a set of* initial states*;*

- In *is a finite collection of* input variables. *We assume that* $\text{In} = \Sigma \cup W$*:* $\Sigma = \Sigma_U \times \Sigma_D$ *where* Σ_U *contains discrete input variables and* Σ_D *contains discrete disturbance input variables;* $W = U \cup D$ *where* U *contains continuous control input variables and* D *contains continuous disturbance input variables;*
- $f : Q \times X \times \text{In} \to \mathbb{R}^n$ *is a* vector field*;*
- $\text{Dom} : Q \to 2^{X \times \text{In}}$ *is a* domain*;*
- $R : Q \times X \times \text{In} \to 2^{Q \times X}$ *is a* reset relation.
- Out *is a finite collection of* output variables. *We assume that* $\text{Out} = P \cup Y$*, where* P *contains discrete output variables, and* Y *contains continuous output variables.*

We refer to $(\sigma, w) \in \text{In}$ as the *input* of H, and $(p, y) \in \text{Out}$ as the *output* of H.

The control objective is assumed to be encoded by means of a safety property $\Box F$ (always F) with $F \subseteq Q \times X$.

- The inputs are chosen by a "controller"; typically some input variables can be controlled (their values can be assigned at will by a controller), while others cannot be controlled (their values are determined by the environment);
- Uncontrollable variables (or *disturbances*) typically represent:
  - noise in the sensors, numerical computations, etc.
  - external forces, wind for the aircraft etc.
  - unmodeled dynamics
  - uncertainty about the actions of other agents (such as in the air traffic control and highway system examples)

2 Controllers

A controller can be defined as a map from state executions to sets of allowable inputs:

$$C : (\text{executions of } H) \to 2^{\Sigma_U \cup U}$$

The interpretation is that the controller restricts the valuations of the control input variables that are allowed along the trajectory. The controller may still be able to "cheat" by forcing the execution to be Zeno. Typically one assumes that all loops among discrete states require a non zero amount of time. This assumption may be somewhat restrictive and is difficult to enforce by manipulating the primitives of the model.

Memoryless Controllers

A controller, C, is called *memoryless* (or sometimes pure feedback) if for all executions of the hybrid system ending at the same state, the controller would be the same. A memoryless controllers can be characterized by a feedback map:

$$g : Q \times X \to 2^{\Sigma_U \times U}$$

Given a plant, H, and a memoryless controller, g, we can define the closed loop hybrid automaton, $H = (Q, X, \text{Init}, \text{In}, f, \text{Dom}, R, \text{Out}, g)$.

Controlled Invariant Sets

Typically, for a controller synthesis problem one treats the set of initial conditions, Init, as variable and attempts to establish the largest set of states for which there exists a controller that satisfies the specification. This set of initial conditions turns out to be a "controlled invariant set".

Definition 2 (Controlled Invariant) *A set $W \subseteq \mathbf{Q} \times \mathbf{X}$ is called controlled invariant if there exists a controller that solves the controller synthesis problem $(H', \Box W)$ where H' is identical to H except for* Init' *which is equal to W.*

A controlled invariant set W is called *maximal* if it is not a proper subset of another controlled invariant set. We say a controller *renders W invariant* if it solves the controller synthesis problem $(H', \Box W)$ where $\text{Init}' = W$.

Least Restrictive Controllers

We would like to derive a memoryless controller that solves the problem while imposing minimal restrictions on the controls it allows. There are at least two reasons why such a controller is desirable:

1. As discussed above, safety properties can sometimes be satisfied using trivial controllers (that cause deadlocks or zeno executions for example). Imposing as few restrictions as possible allows us to find a meaningful controller whenever possible.
2. In many cases multiple, prioritized specifications are given for a particular problem. Imposing fewer restrictions on the controls when designing controllers for higher priority specifications allows us greater flexibility when trying to satisfy lower priority specifications.

Memoryless controllers that solve the synthesis problem $(H, \Box F)$ can be partially ordered by the relation:

$$g_1 \preceq g_2 \Leftrightarrow g_1(x) \subseteq g_2(x) \text{ for all } x \in \mathbf{X}$$

Definition 3 *A memoryless controller that solves $(H, \Box F)$ is called* least restrictive *if it is maximal among the controllers that solve $(H, \Box F)$.*

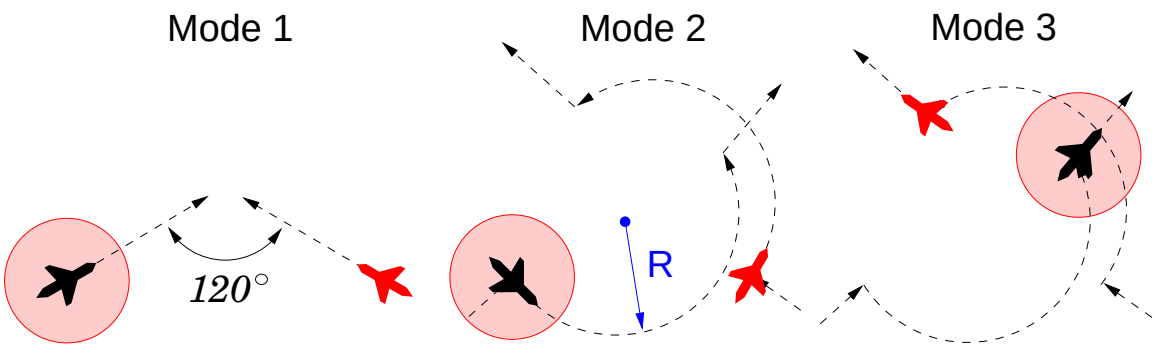


Figure 1: Two aircraft in three modes of operation: in modes 1 and 3 the aircraft follow a straight course and in mode 2 the aircraft follow a half circle. The initial relative heading (120°) is preserved throughout.

Some Remarks on Implementation

The notion of a controller introduced above may be inadequate when it comes to implementation. For one thing, the set valued map g allows non-deterministic choices of control inputs. Since in practice only one input can be applied to the system at any time, this non-determinism has to somehow be resolved when it comes time to implement such a controller. The set valued map can in this sense be thought of as a family of single valued controllers; implementation involves choosing one controller from this family.

Normally, one would implement a controller by using another hybrid automaton, which, when composed with the plant automaton yields the desired behavior.

We assume that the entire state is available to the controller. In general this will not be the case. If a controller is to be implemented by a hybrid automaton, the information the controller has about the plant is obtained through the valuations of the output variables of the plant, which are not necessarily in one to one correspondence with the valuations of the state variables. The controller synthesis problem under partial observation (output feedback) is much more complicated than the full observation (state feedback) problem addressed here and is a topic of current research.

3 Motivating Example

We present as motivating example a model for the kinematic motions of two aircraft, labeled 1 and 2, at a fixed altitude. Let $(x_r, y_r, \psi_r) \in \mathbb{R}^2 \times [-\pi, \pi)$ represent the relative position and orientation of aircraft 2 with respect to aircraft 1. In terms of the absolute positions and orientations of the two aircraft x_i, y_i, ψ_i for $i = 1, 2$, it may be verified that $x_r = \cos\psi_1(x_2 - x_1) + \sin\psi_1(y_2 - y_1), y_r = -\sin\psi_1(x_2 - x_1) + \cos\psi_1(y_2 - y_1), \psi_r = \psi_2 - \psi_1$, and

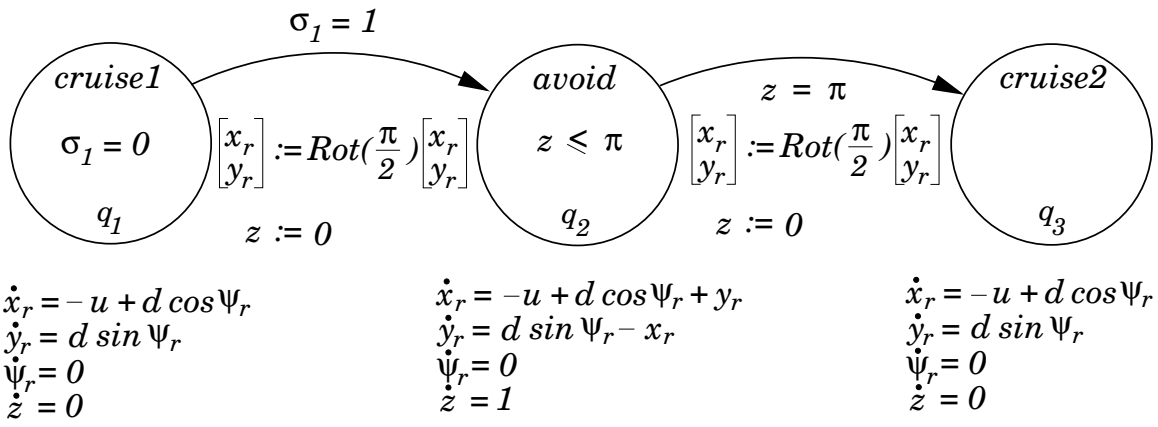


Figure 2: In q_1 both aircraft follow a straight course, in q_2 a half circle, and in q_3 both aircraft return to a straight course.

it is easy to derive that

$$\begin{aligned} \dot{x}_r &= -v_1 + v_2 \cos\psi_r + \omega_1 y_r \\ \dot{y}_r &= v_2 \sin\psi_r - \omega_1 x_r \\ \dot{\psi}_r &= \omega_2 - \omega_1 \end{aligned} \tag{1}$$

where v_i is the linear velocity of aircraft i and ω_i is its angular velocity. The protected zone of aircraft 2 may be translated to the origin of this relative frame, and thus the relative position (x_r, y_r) must remain outside of the disk $\{(x_r, y_r, \psi_r) : x_r^2 + y_r^2 < 5^2\}$. The flight modes for this system of two aircraft are based on the linear and angular velocities of the aircraft. We consider two possibilities: $\omega_i = 0$, meaning that aircraft i follows a straight line, and $\omega_i = 1$, meaning that aircraft i follows an arc of a circle if v_i is kept constant. These maneuvers approximate closely the behavior of pilots flying aircraft: straight line segments (constant heading) and arcs of circles (constant bank angle) are easy to fly both manually and on autopilot. Consider a maneuver in which there are three modes in sequence: a *cruise* mode in which both aircraft follow a straight path; an *avoid* mode in which both aircraft follow a circular arc path; and a second *cruise* mode in which the aircraft return to the straight path. The protocol of the maneuver is that as soon as the aircraft are within a certain distance of each other, each aircraft turns 90° to its right and follows a half circle. Once the half circle is complete, each aircraft returns to its original heading and continues on its straight path (Figure 1). We assume that both aircraft switch modes simultaneously, so that the relative orientation ψ_r is constant, and we assume that both aircraft fly an arc with the same radius at the same velocity. These assumptions simply allow us to display the evolution of the continuous state in two dimensions, making the results easier to present: in a true conflict resolution scenario these assumptions would be removed. This maneuver generalizes to n-aircraft as a "roundabout" maneuver, discussed in [4].

The dynamics of the maneuver can be encoded by the hybrid automaton of Figure 2, where q_1 corresponds to cruising before the avoid maneuver, q_2 corresponds to the avoid mode, and q_3 corresponds to cruising after the avoid maneuver has been completed. There is one discrete control input σ_1, such that the switch from $\sigma_1 = 0$ to $\sigma_1 = 1$ triggers the transition from q_1 to

q_2. The transition from q_2 to q_3 is required to take place after the aircraft have completed a half circle: note that with $\omega_i = 1$, for $i = 1, 2$, it takes π time units to complete a half circle. The continuous state space is augmented with a timer $z \in \mathbb{R}^+$ to force this transition. Let $x = (x_r, y_r, \psi_r, z)^T$. At each transition, both aircraft change heading instantaneously by $\pi/2$ radians; we represent this with the standard rotation matrix $Rot(\frac{\pi}{2})$. Assuming computation in the flight management system of aircraft 1, we assume that v_1 is controllable, and v_2 is known to within some uncertainty. Safety is defined in terms of the relative distance between the two aircraft:

$$F^c = G = \{q_1, q_2, q_3\} \times \{x \in X \ : \ x_r^2 + y_r^2 \leq 5^2\} \tag{2}$$

Thus the state space of this two aircraft system is $Q \times X = \{q_1, q_2, q_3\} \times (\mathbb{R}^2 \times [-\pi, \pi) \times \mathbb{R}^+)$. The discrete input space is $\Sigma = \Sigma_U = \{0, 1\}$ ($\Sigma_D = \emptyset$), and the continuous input space is $W = U \times D$, where $U = [v_{1_{min}} v_{1_{max}}]$ and $D = [v_{2_{min}} v_{2_{max}}]$. We assume Init $= q_1 \times (G^\circ)^c$, that f is described by the relative aircraft dynamics (1) augmented with a timer, as shown in Figure 2, and that Dom is given as follows:

$$\text{Dom}(q_1) = (X, \{\text{In} : \sigma_1 = 0\}); \text{Dom}(q_2) = (\{x \in X \mid 0 \leq z \leq \pi\}, \text{In}); \text{Dom}(q_3) = (X, \text{In}) \tag{3}$$

The map R which resets x_r, y_r in transitions from q_1 to q_2 and q_2 to q_3 is described in Figure 2. Output variables are the same as the state variables, in this example.

The controller synthesis problem is therefore to generate continuous control inputs for aircraft 1, as well as the conditions under which aircraft 1 is allowed to switch modes, so that that the 5 nautical mile separation is maintained, despite the disturbance actions of aircraft 2 (unknown but within known bounds).

Controllers for Safety for Discrete Systems

Before solving this hybrid control problem, let's start with something simpler: a finite automaton model of a discrete system. Recall from Lecture 2:

- **Definition:** A *finite automaton* M is a mathematical model of a system represented as

$$(Q, \Sigma, \text{Init}, R) \tag{4}$$

where

  - Q is a finite set of *discrete state variables*
  - $\Sigma = \Sigma_1 \cup \Sigma_2$ is a finite set of *discrete input variables*, where Σ_1 contains the controller's inputs and Σ_2 contains the environment's inputs, which cannot be controlled
  - Init $\subseteq Q$ is a set of *initial states*
  - $R : Q \times \Sigma \to 2^Q$ is a *transition relation* which maps the state and input space to subsets of the state space and thus describes the transition logic of the finite automaton

- Now consider a set of states $F \subseteq Q$.

  Problem statement: Design a control law to guarantee $\Box F(\chi)$ (where χ represents executions of the discrete system) despite possible worst case action of the disturbance inputs.

  There could be many such control laws: we would like to establish the largest set of states for which there exists a controller that manages to keep all executions inside F. This is called *least restrictive* control; its practical importance is that one can compose any other control design with this least restrictive control law, and the least restrictive control law will only come into effect when one is in danger of violating $\Box F(\chi)$.

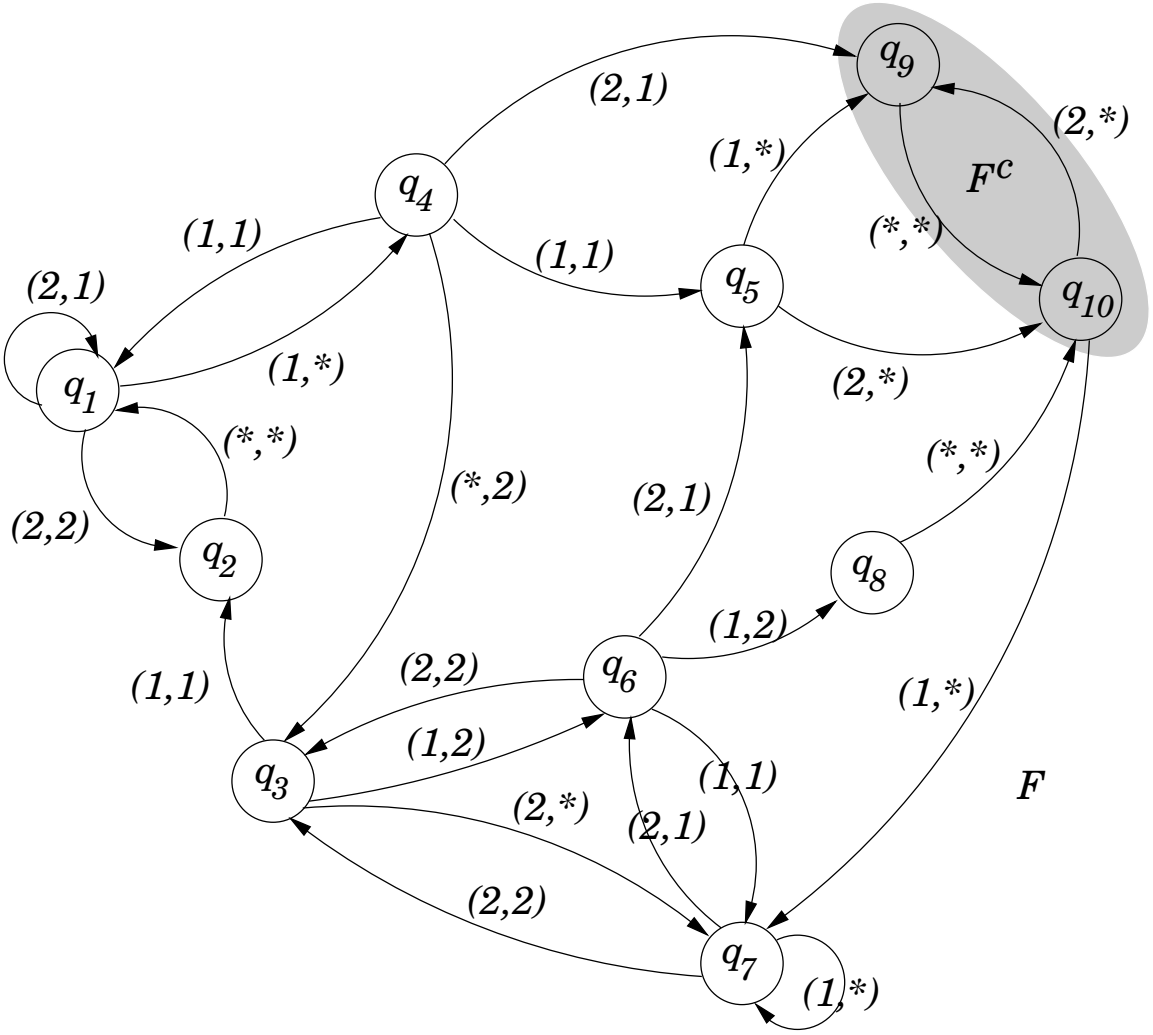


Figure 3: Example for controller synthesis for discrete systems.

- This is easiest to demonstrate by the following example, illustrated in Figure 3. Consider the finite automaton shown in Figure 3 where $Q = \{q_1, \ldots, q_{10}\}$, $\Sigma_1 = \Sigma_2 = \{1, 2\}$, $F = \{q_1, q_2, \ldots q_8\}$, and the transition relation is as shown in the figure, where the dis-

crete inputs are listed in the order (σ_1, σ_2) which represents the "order of play"[1], and $*$ denotes a "wild-card" (either 1 or 2).

- Let us denote by W^* the largest set of states such that there exists a feedback controller that keeps the execution inside F. Clearly:

$$W^* \subseteq F = \{q_1, q_2, \ldots, q_8\} \tag{5}$$

- Next, look at states that can end up in F^c in one transition: (q_4, q_5, and q_8). Now, from q_4, if $\sigma_1 = 1$ then whatever σ_2 chooses to do the system will remain in F; however from q_5 and q_8, whatever σ_1 chooses to do, the system leaves F. Thus

$$W^* \subseteq \{q_1, q_2, q_3, q_4, q_6, q_7\} \tag{6}$$

- Now, we continue to iterate backwards. Consider states that can leave the above set in one transition: q_4 and q_6. Note that from q_4, whatever σ_1 chooses to do, if $\sigma_2 = 1$ then the system leaves the set, and from q_6, whatever σ_1 chooses to do, σ_2 can play a value to force the system out of the set. Thus

$$W^* \subseteq \{q_1, q_2, q_3, q_7\} \tag{7}$$

- From these remaining states, if σ_1 chooses according to the following feedback map:

$$g(q) = \begin{cases} \{1\} & \text{if } q = q_7 \\ \{2\} & \text{if } q \in \{q_1, q_3\} \\ \{1, 2\} & \text{if } q = q_2 \end{cases} \tag{8}$$

  then the system is guaranteed to remain in $\{q_1, q_2, q_3, q_7\}$ forever.

- Thus,

$$W^* = \{q_1, q_2, q_3, q_7\} \tag{9}$$

  and g is called the *least restrictive control scheme* that renders W^* invariant.

- More generally, we may encode the above as the following algorithm:

  We define the *winning states* W^* for the controller as the subset of F from which the system has a sequence of control actions $\sigma_1(\cdot)$ which can force the system to remain in F despite the actions of the environment $\sigma_2(\cdot)$. The set W^* can be calculated as the fixed point of the following iteration (where a negative index $i \in \mathbb{Z}_-$ is used to indicate that each step is a predecessor operation):

  Algorithm 1 (Maximal Controlled Invariant Set for Finite State Automata)

[1] As we shall see, this order gives the disturbance action the benefit of knowing what the control has just played, and thus represents a conservative solution.

initialization: $W^0 = F$, $W^{-1} = \emptyset$, $i = 0$.
while $W^i \neq W^{i-1}$ **do**
$W^{i-1} = W^i \cap \{q \in Q \mid \exists \sigma_1 \in \Sigma_1 \; \forall \sigma_2 \in \Sigma_2 \; R(q, \sigma_1, \sigma_2) \subseteq W^i \}$
$i = i - 1$
end while

The iteration terminates when $W^i = W^{i-1} \triangleq W^*$. At each step of the iteration, $W^{i-1} \subseteq W^i$. Since $|Q|$ is finite the iteration terminates in a finite number of steps. The set W^i contains those states for which the controller has a sequence of actions which will ensure that the system remains in F for at least i steps, for all possible sequences $\sigma_2(\cdot) \in \Sigma_2$.

This problem can be formulated as a game.

In order to characterize this iteration mathematically, we associate a *value function* $J(q, i)$ to each state at each iteration, representing the future reward or cost to be incurred by the system given that its current state is q and iteration i:

$$J(q,i) : Q \times \mathbb{Z}_- \to \{0,1\} \text{ such that } J(q,i) = \begin{cases} 1 & q \in W^i \\ 0 & q \in (W^i)^c \end{cases} \tag{10}$$

Therefore, $W^i = \{q \in Q \mid J(q,i) = 1\}$. Since the most logical action of the controller is to keep the system inside F in the face of unknown and therefore possibly hostile actions of the environment:

$$\max_{\sigma_1 \in \Sigma_1} \min_{\sigma_2 \in \Sigma_2} \min_{q' \in R(q,\sigma_1,\sigma_2)} J(q', i) = \begin{cases} 1 & \text{if } \exists \sigma_1 \in \Sigma_1 \; \forall \sigma_2 \in \Sigma_2, R(q, \sigma_1, \sigma_2) \subseteq W^i \\ 0 & \text{otherwise} \end{cases} \tag{11}$$

The "$\min_{q' \in R(q,\sigma_1,\sigma_2)}$" in the above compensates for the nondeterminism in R; the order of operations $\max_{\sigma_1} \min_{\sigma_2}$ means that the controller *plays first*, trying to maximize the minimum value of $J(\cdot)$. This representation gives the environment the advantage, since it has "prior" knowledge of the controller's action when making its own choice. Therefore, in general,

$$\max_{\sigma_1 \in \Sigma_1} \min_{\sigma_2 \in \Sigma_2} \min_{q' \in R(q,\sigma_1,\sigma_2)} J(\cdot) \leq \min_{\sigma_2 \in \Sigma_2} \max_{\sigma_1 \in \Sigma_1} \min_{q' \in R(q,\sigma_1,\sigma_2)} J(\cdot) \tag{12}$$

with equality occurring when the action (σ_1, σ_2) is a *saddle solution*, or a *no regret* solution for each player. Here, we do not need to assume the existence of a saddle solution, rather we always give advantage to the environment, the player doing its worst to drive the system out of F, in order to ensure a conservative solution. Strictly speaking this is a *Stackelberg solution* of the game with the controller as leader.

The iteration process in Algorithm 1 may be summarized by the difference equation:

$$J(q, i-1) - J(q,i) = \min\{0, \max_{\sigma_1 \in \Sigma_1} \min_{\sigma_2 \in \Sigma_2} [\min_{q' \in R(q,\sigma_1,\sigma_2)} J(q', i) - J(q,i)]\} \tag{13}$$

We refer to equation (13) as a "discrete Hamilton-Jacobi equation." The first "min" in the equation ensures that states outside W^i that can be forced by the controller to transition into W^i are prevented from appearing in W^{i-1}. This means that once a state has associated to it a value of zero, the value stays at zero for all subsequent iterations: enforcing the requirement that "once a state becomes unsafe, it remains unsafe".

Proposition 4 (Winning States W^*) *For finite sets Q and Σ, a fixed point $J^*(q)$ of (13) is reached in a finite number of steps. The set of winning states for the controller is $W^* = \{q \in Q \mid J^*(q) = 1\}$. W^* is the largest controlled invariant subset of F.*

Additional References

The problem of synthesizing control laws $g : Q \to 2^{\Sigma_1}$ in the presence of uncertain actions $\sigma_2(\cdot) \in \Sigma_2^\omega$ for the finite automaton described by (4) was first posed by Church in 1962 [5], who was studying problems in digital circuit design, and was solved by Büchi and Landweber [6] and Rabin [7] in the late 1960's and early 1970's using a version of the von Neumann-Morgenstern discrete game [8]. More recently, Ramadge and Wonham [9] added new insight into the structure of the control law. A temporal logic for modeling such games is introduced in [10]. The "discrete Hamilton-Jacobi equation" to model such a process was introduced in [2].

References

[1] J. Lygeros, C. Tomlin, and S. Sastry. Controllers for reachability specifications for hybrid systems. *Automatica*, 35(3):349–370, 1999.

[2] C. Tomlin, J. Lygeros, and S. Sastry. Synthesizing controllers for nonlinear hybrid systems. In T. Henzinger and S. Sastry, editors, *Hybrid Systems: Computation and Control*, number 1386 in LNCS, pages 360–373. Springer Verlag, New York, 1998.

[3] C. J. Tomlin, J. Lygeros, and S. Sastry. A game theoretic approach to controller design for hybrid systems. *Proceedings of the IEEE*, 88(7):949–970, July 2000.

[4] C. Tomlin, G. J. Pappas, and S. Sastry. Conflict resolution for air traffic management: A case study in multi-agent hybrid systems. *IEEE Transactions on Automatic Control*, 43(4):509–521, April 1998.

[5] A. Church. Logic, arithmetic, and automata. In *Proceedings of the International Congress of Mathematicians*, pages 23–35. 1962.

[6] J. R. Büchi and L. H. Landweber. Solving sequential conditions by finite-state operators. In *Proceedings of the American Mathematical Society*, pages 295–311, 1969.

[7] M. O. Rabin. Automata on infinite objects and Church's problem. In *Regional Conference Series in Mathematics*, 1972.

[8] J. von Neumann and O. Morgenstern. *Theory of Games and Economic Behavior*. Princeton University Press, 1947.

[9] P. J. G. Ramadge and W. M. Wonham. The control of discrete event dynamical systems. *Proceedings of the IEEE*, Vol.77(1):81–98, 1989.

[10] R. Alur, T.A. Henzinger, and O. Kupferman. Alternating-time temporal logic. In *Proceedings of the 38th Annual Symposium on Foundations of Computer Science*, pages 100–109. IEEE Computer Society Press, 1997.

AA278A Lecture Notes 8. Optimal Control and Dynamic Games

Claire J. Tomlin

May 11, 2005

These notes represent an introduction to the theory of optimal control and dynamic games; they were written by S. S. Sastry [1].

There exist two main approaches to optimal control and dynamic games:

1. via the Calculus of Variations (making use of the Maximum Principle);
2. via Dynamic Programming (making use of the Principle of Optimality).

Both approaches involve converting an optimization over a function space to a pointwise optimization. The methods are based on the following simple observations:

1. For the calculus of variations, the optimal curve should be such that neighboring curves to not lead to smaller costs. Thus the 'derivative' of the cost function about the optimal curve should be zero: one takes small variations about the candidate optimal solution and attempts to make the change in the cost zero.
2. For dynamic programming, the optimal curve remains optimal at intermediate points in time.

In these notes, both approaches are discussed for optimal control; the methods are then extended to dynamic games.

1 Optimal Control based on the Calculus of Variations

There are numerous excellent books on optimal control. Commonly used books which we will draw from are Athans and Falb [2], Berkovitz [4], Bryson and Ho [5], Pontryagin et al [6], Young [7], Kirk [8], Lewis [9] and Fleming and Rishel[10]. The history of optimal control is quite well rooted in antiquity, with allusion being made to Dido, the first Queen of Carthage, who when asked to take as much land as could be covered by an ox-hide, cut the ox-hide into a tiny strip and proceeded to enclose the entire area of what came to be

know as Carthage in a circle of the appropriate radius[1]. The calculus of variations is really the ancient precursor to optimal control. Iso perimetric problems of the kind that gave Dido her kingdom were treated in detail by Tonelli and later by Euler. Both Euler and Lagrange laid the foundations of mechanics in a variational setting culminating in the Euler Lagrange equations. Newton used variational methods to determine the shape of a body that minimizes drag, and Bernoulli formulated his brachistochrone problem in the seventeenth century, which attracted the attention of Newton and L'Hôpital. This intellectual heritage was revived and generalized by Bellman [3] in the context of dynamic programming and by Pontryagin and his school in the so-called Pontryagin principle for optimal control ([6]).

Consider a nonlinear possibly time varying dynamical system described by

$$\dot{x} = f(x, u, t) \tag{1}$$

with state $x(t) \in \mathbb{R}^n$ and the control input $u \in \mathbb{R}^{n_i}$. Consider the problem of minimizing the performance index

$$J = \phi(x(t_f), t_f) + \int_{t_0}^{t_f} L(x(t), u(t), t)dt \tag{2}$$

where t_0 is the initial time, t_f the final time (free), $L(x, u, t)$ is the running cost, and $\phi(x(t_f), t_f)$ is the cost at the terminal time. The initial time t_0 is assumed to be fixed and t_f variable. Problems involving a cost only on the final and initial state are referred to as Mayer problems, those involving only the integral or running cost are called Lagrange problems and costs of the form of equation (2) are referred to as Bolza problems. We will also have a constraint on the final state given by

$$\psi(x(t_f), t_f) = 0 \tag{3}$$

where $\psi : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}^p$ is a smooth map. To derive necessary conditions for the optimum, we will perform the calculus of variations on the cost function of (2) subject to the constraints of equations (1), (3). To this end, define the modified cost function, using the Lagrange multipliers $\lambda \in \mathbb{R}^p, p(t) \in \mathbb{R}^n$,

$$\tilde{J} = \phi(x(t_f), t_f) + \lambda^T \psi(x(t_f), t_f) + \int_{t_0}^{t_f} \left[L(x, u, t) + p^T(f(x, u, t) - \dot{x})\right] dt \tag{4}$$

Defining the *Hamiltonian* $H(x, u, t)$ using what is referred to as a *Legendre transformation*

$$H(x, p, u, t) = L(x, u, t) + p^T f(x, u, t) \tag{5}$$

The variation of (4) is given by assuming independent variations in $\delta u(), \delta x(), \delta p(), \delta\lambda$, and δt_f:

$$\begin{array}{rcl} \delta\tilde{J} = & (D_1\phi + D_1\psi^T\lambda)\delta x|_{t_f} + (D_2\phi + D_2\psi^T\lambda)\delta t|_{t_f} + \psi^T\delta\lambda \\ + & (H - p^T\dot{x})\delta t|_{t_f} \\ + & \int_{t_0}^{t_f} \left[D_1 H\delta x + D_3 H\delta u - p^T\delta\dot{x} + (D_2 H^T - \dot{x})^T\delta p\right] dt \end{array} \tag{6}$$

[1]The optimal control problem here is to enclose the maximum area using a closed curve of given length.

The notation D_iH stands for the derivative of H with respect to the i th argument. Thus, for example,

$$D_3H(x,p,u,t) = \frac{\partial H}{\partial u} \quad D_1H(x,p,u,t) = \frac{\partial H}{\partial x}$$

Integrating by parts for $\int p^T \delta\dot{x} dt$ yields

$$\begin{array}{rl} \delta\tilde{J} = & (D_1\phi + D_1\psi^T\lambda - p^T)\delta x(t_f) + (D_2\phi + D_2\psi^T\lambda + H)\delta t_f + \psi^T\delta\lambda \\ + & \int_{t_0}^{t_f} \left[(D_1H + \dot{p}^T)\delta x + D_3H\delta u + (D_2^TH - \dot{x})^T\delta p\right] dt \end{array} \tag{7}$$

An extremum of $\tilde{J}$ is achieved when $\delta\tilde{J} = 0$ for all independent variations $\delta\lambda, \delta x, \delta u, \delta p$. These conditions are recorded in the following

Table of necessary conditions for optimality:

Table 1

Description	Equation	Variation
Final State constraint	$\psi(x(t_f), t_f) = 0$	$\delta\lambda$
State Equation	$\dot{x} = \frac{\partial H}{\partial p}^T$	δp
Costate equation	$\dot{p} = -\frac{\partial H}{\partial x}^T$	δx
Input stationarity	$\frac{\partial H}{\partial u} = 0$	δu
Boundary conditions	$D_1\phi - p^T = -D_1\psi^T\lambda\|_{t_f}$	$\delta x(t_f)$
	$H + D_2\phi = -D_2\psi^T\lambda\|_{t_f}$	δt_f

The conditions of Table (1) and the boundary conditions $x(t_0) = x_0$ and the constraint on the final state $\psi(x(t_f), t_f) = 0$ constitute the necessary conditions for optimality. The end point constraint equation is referred to as the transversality condition:

$$\begin{array}{rl} D_1\phi - p^T = & -D_1\psi^T\lambda \\ H + D_2\phi = & -D_2\psi^T\lambda \end{array} \tag{8}$$

The optimality conditions may be written explicitly as

$$\begin{array}{rl} \dot{x} = & \frac{\partial H}{\partial p}^T(x, u^*, p) \\ \dot{p} = & -\frac{\partial H}{\partial x}^T(x, u^*, p) \end{array} \tag{9}$$

with the stationarity condition reading

$$\frac{\partial H}{\partial u}(x, u^*, p) = 0$$

and the endpoint constraint $\psi(x(t_f), t_f) = 0$. *The key point to the derivation of the necessary conditions of optimality is that the Legendre transformation of the Lagrangian to be minimized into a Hamiltonian converts a functional minimization problem into a static optimization problem on the function* $H(x, u, p, t)$.

The question of when these equations also constitute sufficient conditions for (local) optimality is an important one and needs to be ascertained by taking the second variation of $\tilde{J}$. This is an involved procedure but the input stationarity condition in Table (1) hints at the **sufficient condition for local minimality** of a given trajectory $x^*(\cdot), u^*(\cdot), p^*(\cdot)$ being a local minimum as being that the Hessian of the Hamiltonian,

$$D_2^2 H(x^*, u^*, p^*, t) \tag{10}$$

being positive definite along the optimal trajectory. A sufficient condition for this is to ask simply that the $n_i \times n_i$ Hessian matrix

$$D_2^2 H(x, u, p, t) \tag{11}$$

be positive definite. As far as conditions for **global minimality** are concerned, again the stationarity condition hints at a sufficient condition for global minimality being that

$$u^*(t) = \underset{\{ \text{ min over u } \}}{\text{argmin}} \quad H(x^*(t), u, p^*(t), t) \tag{12}$$

Sufficient conditions for this are, for example, the convexity of the Hamiltonian $H(x, u, p, t)$ in u.

Finally, there are instances in which the Hamiltonian $H(x, u, p, t)$ is not a function of u at some values of x, p, t. These cases are referred to as *singular extremals* and need to be treated with care, since the value of u is left unspecified as far as the optimization is concerned.

1.1 Fixed Endpoint problems

In the instance that the final time t_f is fixed, the equations take on a simpler form, since there is no variation in δt_f. Then, the boundary condition of equation (8) becomes

$$p^T(t_f) = D_1\phi + D_1\psi^T\lambda|_{t_f} \tag{13}$$

Further, if there is no final state constraint the boundary condition simplifies even further to

$$p(t_f) = D_1\phi^T|_{t_f} \tag{14}$$

1.2 Time Invariant Systems

In the instance that $f(x, u, t)$ and the running cost $L(x, u, t)$ are not explicitly functions of time, there is no final state constraint and the final time t_f is fixed, the formulas of Table (1) can be rewritten as

State Equation	$\dot{x} = \frac{\partial H}{\partial p}^T = f(x, u^*)$
Costate Equation	$\dot{p} = -\frac{\partial H}{\partial x}^T = -D_1 f^T p + D_1 L^T$
Stationarity Condition	$0 = \frac{\partial H}{\partial u} = D_2 L^T + D_2 f^T p$
Transversality Conditions	$D_1\phi - p^T = -D_1\psi^T\lambda$
	$H(t_f) = 0$

In addition, it may be verified that

$$\frac{dH^*}{dt} = \frac{\partial H^*}{\partial x}(x,p)\dot{x} + \frac{\partial H^*}{\partial p}\dot{p} = 0 \tag{15}$$

thereby establishing that $H^*(t) \equiv 0$.

1.3 Connections with Classical Mechanics

Hamilton's principle of least action states (under certain conditions [2]) that a conservative system moves so as to minimize the time integral of its "action", defined to be the difference between the kinetic and potential energy. To make this more explicit we define $q \in \mathbb{R}^n$ to be the vector of generalized coordinates of the system and denote by $U(q)$ the potential energy of the system and $T(q, \dot{q})$ the kinetic energy of the system. Then Hamilton's principle of least action asks to solve an optimal control problem for the system

$$\dot{q} = u$$

with Lagrangian

$$L(q, u) = T(q, u) - U(q)$$

The equations (9) in this context have $H(q, u, p) = L(q, u) + p^T u$. $u^* = u^*(p, q)$ is chosen so as to minimize the Hamiltonian H. A necessary condition for stationarity is that $u^*(p, q)$ satisfies

$$0 = \frac{\partial H}{\partial u} = \frac{\partial L}{\partial u} + p^T \tag{16}$$

The form of the equations (9) in this context is that of the familiar *Hamilton Jacobi equations.* The costate p has the interpretation of momentum.

$$\begin{aligned} \dot{q} &= \frac{\partial H^*}{\partial p}(p,q) &&= u^*(p,q) \\ \dot{p} &= -\frac{\partial H^*}{\partial q}(p,q) \end{aligned} \tag{17}$$

Combining the second of these equations with (16) yields the familiar *Euler Lagrange equations*

$$\frac{\partial L}{\partial q} - \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}}\right) = 0 \tag{18}$$

[2] For example, there is no dissipation or no nonholonomic constraints. Holonomic or integrable constraints are dealt with by adding appropriate Lagrange multipliers. If nonholonomic constraints are dealt with in the same manner, we get equations of motion, dubbed vakonomic by Arnold [11] which do not correspond to experimentally observed motions. On the other hand, if there are only holonomic constraints, the equations of motion that we derive from Hamilton's principle of least action is equivalent to Newton's laws.

2 Optimal Control based on Dynamic Programming

To begin this discussion, we will embed the optimization problem which we are solving in a larger class of problems, more specifically we will consider the original cost function of equation (2) from an initial time $t \in [t_0, t_f]$ by considering the cost function on the interval $[t, t_f]$:

$$J(x(t), t) = \phi(x(t_f), t_f) + \int_t^{t_f} L(x(\tau), u(\tau), \tau) d\tau$$

Bellman's principle of optimality says that if we have found the optimal trajectory on the interval from $[t_0, t_f]$ by solving the optimal control problem on that interval, the resulting trajectory is also optimal on all subintervals of this interval of the form $[t, t_f]$ with $t > t_0$, provided that the initial condition at time t was obtained from running the system forward along the optimal trajectory from time t_0. The optimal value of $J(x(t), t)$ is referred to as the "cost-to go". To be able to state the following key theorem of optimal control we will need to define the "optimal Hamiltonian" to be

$$H^*(x, p, t) := H(x, u^*, p, t)$$

Theorem 1 The Hamilton Jacobi Bellman equation

Consider, the time varying optimal control problem of (2) with fixed endpoint t_f and time varying dynamics. If the optimal value function, i.e. $J^*(x(t_0), t_0)$ is a smooth function of x, t, then $J^*(x, t)$ satisfies the **Hamilton Jacobi Bellman** *partial differential equation*

$$\frac{\partial J^*}{\partial t}(x, t) = -H^*(x, \frac{\partial J^*}{\partial x}(x, t), t) \tag{19}$$

with boundary conditions given by $J^*(x, t_f) = \phi(x, t_f)$ for all $x \in \{x : \psi(x, t_f) = 0\}$.

Proof: The proof uses the principle of optimality. This principle says that if we have found the optimal trajectory on the interval from $[t, t_f]$ by solving the optimal control problem on that interval, the resulting trajectory is also optimal on all subintervals of this interval of the form $[t_1, t_f]$ with $t_1 > t$, provided that the initial condition at time t_1 was obtained from running the system forward along the optimal trajectory from time t. Thus, from using $t_1 = t + \Delta t$, it follows that

$$J^*(x, t) = \min_{\substack{u(\tau) \\ t \le \tau \le t + \Delta t}} \left[\int_t^{t+\Delta t} L(x, u, \tau) d\tau + J^*(x + \Delta x, t + \Delta t) \right] \tag{20}$$

Taking infinitesimals and letting $\Delta t \to 0$ yields that

$$-\frac{\partial J^*}{\partial t} = \min_{u(t)} \left(L + (\frac{\partial J^*}{\partial x}) f \right) \tag{21}$$

with the boundary condition being that the terminal cost is

$$J^*(x, t_f) = \phi(x, t_f)$$

on the surface $\psi(x) = 0$. Using the definition of the Hamiltonian in equation (5), it follows from equation (21) that the Hamilton Jacobi equation of equation (19) holds.

□

Remarks:

1. The preceding theorem was stated as a necessary condition for extremal solutions of the optimal control problem. As far as minimal and global solutions of the optimal control problem, the Hamilton Jacobi Bellman equations read as in equation (21). In this sense, the form of the Hamilton Jacobi Bellman equation in (21) is more general.

2. The **Eulerian conditions** of Table (1) are easily obtained from the Hamilton Jacobi Bellman equation by proving that $p^T(t) := \frac{\partial J^*}{\partial x}(x,t)$ satisfies the costate equations of that Table. Indeed, consider the equation (21). Since $u(t)$ is unconstrained, it follows that it should satisfy
$$\frac{\partial L}{\partial u}(x^*,u^*) + \frac{\partial f}{\partial u}^T p = 0 \tag{22}$$
Now differentiating the definition of $p(t)$ above with respect to t yields
$$\frac{dp}{dt}^T = \frac{\partial^2 J^*}{\partial t \partial x}(x^*,t) + \frac{\partial^2 J^*}{\partial x^2} f(x^*,u^*,t) \tag{23}$$
Differentiating the Hamilton Jacobi equation (21) with respect to x and using the relation (22) for a stationary solution yields
$$-\frac{\partial^2 J^*}{\partial t \partial x}(x^*,t) = \frac{\partial L}{\partial x} + \frac{\partial^2 J^*}{\partial x^2} f + p^T \frac{\partial f}{\partial x} \tag{24}$$
Using equation (24) in equation (23) yields
$$-\dot{p} = \frac{\partial f}{\partial x}^T p + \frac{\partial L}{\partial x}^T \tag{25}$$
establishing that p is indeed the co-state of Table 1. The boundary conditions on $p(t)$ follow from the boundary conditions on the Hamilton Jacobi Bellman equation.

2.1 Constrained Input Problems

In the instance that there are no constraints on the input, the extremal solutions of the optimal control problem are found by simply extremizing the Hamiltonian and deriving the stationarity condition. Thus, if the specification is that $u(t) \in U \subset \mathbb{R}^{n_i}$ then, the optimality condition is that
$$H(x^*,u^*,p^*,t) \leq H(x^*,u,p^*,t) \quad \forall u \in U \tag{26}$$
If the Hamiltonian is convex in u and U is a convex set, there are no specific problems with this condition. In fact, when there is a single input and the set U is a single closed interval, there are several interesting examples of Hamiltonians for which the optimal inputs switch between the endpoints of the interval, resulting in what is referred to as **bang bang control**. However, problems can arise when U is either not convex or compact. In these cases, a concept of a **relaxed control** taking values in the convex hull of U needs to be introduced. As far as an implementation of a control $u(t) \in \text{conv}U$, but not in U, a probabilistic scheme involving switching between values of U whose convex combination u is needs to be devised.

2.2 Free end time problems

In the instance that the final time t_f is free, the transversality conditions are that

$$\begin{aligned} p^T(t_f) = &\quad D_1\phi + D_1\psi^T\lambda \\ H(t_f) = &\quad -(D_2\phi + D_2\psi^T\lambda) \end{aligned} \tag{27}$$

2.2.1 Minimum time problems

A special class of minimum time problems of especial interest is minimum time problems, where t_f is to be minimized subject to the constraints. This is accounted for by setting the Lagrangian to be 1, and the terminal state cost $\phi \equiv 0$, so that the Hamiltonian is $H(x,u,p,t) = 1 + p^T f(x,u,t)$. Note that by differentiating $H(x,u,p,t)$ with respect to time, we get

$$\frac{dH^*}{dt} = D_1H^*\dot{x} + D_2H^*\dot{u} + D_3H^*\dot{p} + \frac{\partial H^*}{\partial t} \tag{28}$$

Continuing the calculation using the Hamilton Jacobi equation,

$$\frac{dH^*}{dt} = (\frac{\partial H^*}{\partial x} + \dot{p})f(x,u^*,t) + \frac{\partial H^*}{\partial t} = \frac{\partial H^*}{\partial t} \tag{29}$$

In particular, if H^* is not an explicit function of t, it follows that $H^*(x,u,p,t) \equiv H$. Thus, for minimum time problems for which $f(x,u,t)$ and $\psi(x,t)$ are not explicitly functions of t, it follows that $0 = H(t_f) \equiv H(t)$.

3 Two person zero sum dynamical games

The theory of games also has a long and distinguished, if not as long a history. Borel encountered saddle points in a matrix in 1915, but it really took von Neumann to prove his fundamental theorem about the existence of mixed strategies for achieving a saddle solution in games in 1936. In a classic book, von Neumann and Morgenstern ([12]) laid out the connections between static games and economic behavior. Nash, von Stackelberg and others extended the work to N person non-cooperative games. This work has continued in many important new directions in economics. Differential or dynamical games showed up in the work of Isaacs in 1969 and rapidly found fertile ground in the control community where it has progressed. There are several nice books on the subject of dynamical games, our treatment is drawn heavily from Basar and Olsder ([13]).

Consider a so-called dynamical, two person zero sum, perfect state information game modeled by

$$\dot{x} = f(x,u,d,t) \qquad x(t_0) = x_0 \tag{30}$$

on a fixed duration $[t_0, t_f]$. Here $x \in \mathbb{R}^n$ models the state of the system and $u \in \mathbb{R}^{n_1}, d \in \mathbb{R}^{n_2}$ represent the actions of the two players. The game is said to be *zero sum* if one player seeks

to minimize and the other to maximize the same cost function taken to be of the form

$$J = \phi_(x(t_f), t_f) + \int_{t_0}^{t_f} L(x, u, d, t)dt \tag{31}$$

We will assume that player 1 (u) is trying to minimize J and player 2 (d) is trying to maximize J. For simplicity we have omitted the final state constraint and also assumed the end time t_f to be fixed. These two assumptions are made for simplicity but we will discuss the t_f free case when we study pursuit evasion games. The game is said to have perfect information if both players have access to the full state $x(t)$. The solution of two person zero sum games proceeds very much along the lines of the optimal control problem by setting up the Hamiltonian

$$H(x, u, d, p, t) = L(x, u, d, t) + p^T f(x, u, d, t) \tag{32}$$

Rather than simply minimizing $H(x, u, d, p, t)$ the game is said to have a *saddle point solution* if the following analog of the saddle point condition for two person zero sum static games holds:

$$\min_{\mathrm{u}} \max_{\mathrm{d}} H(x, u, d, p, t) = \max_{\mathrm{d}} \min_{\mathrm{u}} H(x, u, d, p, t) \tag{33}$$

If the minmax is equal to the maxmin, the resulting optimal Hamiltonian is denoted $H^*(x, p, t)$ and the optimal inputs u^*, d^* are determined to be respectively,

$$u^*(t) = \operatorname*{argmin}_{\mathrm{u}} \left(\max_{\mathrm{d}} H(x, u, d, p, t) \right) \tag{34}$$

and

$$d^*(t) = \operatorname*{argmax}_{\mathrm{d}} \left(\min_{\mathrm{u}} H(x, u, d, p, t) \right) \tag{35}$$

The equations for the state and costate and the transversality conditions are given as before by

$$\begin{aligned} \dot{x} &= \tfrac{\partial H^*}{\partial p}^T(x, p) \\ \dot{p} &= -\tfrac{\partial H^*}{\partial x}^T(x, p) \end{aligned} \tag{36}$$

with boundary conditions $x(t_0) = x_0$ and $p^T(t_f) = D_1\phi(x_{t_f})$, and the equation is the familiar *Hamilton Jacobi equation.* As before, one can introduce the optimal cost to go $J^*(x(t), t)$ and we have the following analog of Theorem (1):

Theorem 2 The Hamilton Jacobi Isaacs equation

Consider, the two person zero sum differential game problem of (31) with fixed endpoint t_f. If the optimal value function, i.e. $J^*(x(t_0), t_0)$ is a smooth function of x, t, then $J^*(x, t)$ satisfies the **Hamilton Jacobi Isaacs** *partial differential equation*

$$\frac{\partial J^*}{\partial t}(x, t) = -H^*(x, \frac{\partial J^*}{\partial x}(x, t), t) \tag{37}$$

with boundary conditions given by $J^*(x, t_f) = \phi(x)$ for all x.

Remarks

1. We have dealt with saddle solutions for unconstrained input signals u, d thus far in the development. If the inputs are constrained to lie in sets U, D respectively the saddle solutions can be guaranteed to exist if

$$\min_{u \in U} \max_{d \in D} H(x, u, d, p, t) = \max_{d \in D} \min_{u \in U} H(x, u, d, p, t) \tag{38}$$

   Again, if the input sets are not convex, relaxed controls may be needed to achieve the min-max.

2. The sort of remarks that were made about free endpoint optimal control problems can also be made of games.

3. In our problem formulation for games, we did not include explicit terminal state constraints of the form $\psi(x(t_f), t_f) = 0$. These can be easily included, and we will study this situation in greater detail under the heading of pursuit evasion games.

4. The key point in the theory of dynamical games is that the Legendre transformation of the Lagrangian cost function into the Hamiltonian function converts the solution of the "dynamic" game into a "static" game, where one needs to find a saddle point of the Hamiltonian function $H(x, u, d, p, t)$. This is very much in the spirit of the calculus of variations and optimal control.

4 N person dynamical games

When there are N persons playing a game, many new and interesting new possibilities arise. There is a scenario in which the N agents are non-cooperative, and another in which they cooperate in teams. Of course, if the information available to each of them is different, this makes the solution even more interesting. In this section, we will assume that each of the agents has access to the full state of the system. In this section, we will only discuss non-cooperative solution concepts: first the Nash solution concept and then briefly the Stackelberg solution concept. Cooperative games with total cooperation are simply optimal control problems. If there is cooperation among teams, this can be viewed as a noncooperative game between the teams. When however there are side payments between teams the scope of the problem increases quite considerably.

4.1 Non-cooperative Nash solutions

When there are N players each able to influence the process by controls $u_i \in \mathbb{R}^{n_i}, i = 1, \ldots, N$, modeled by

$$\dot{x} = f(x, u_1, \ldots, u_N, t) \tag{39}$$

and each cost functional (to be minimized) is of the form

$$J_i(u_1(\cdot), \ldots, u_N(\cdot)) = \phi_i(x(t_f), t_f) + \int_{t_0}^{t_f} L_i(x, u_1, \ldots, u_N, t)dt \tag{40}$$

different solution concepts need to be invoked. The simplest non-cooperative solution strategy is a so-called *non-cooperative Nash equilibrium*. A set of controls $u_i^*, i = 1, \ldots, N$ is said to be a Nash strategy if for each player modifying that strategy, and assuming that the others play their Nash strategies, results in an increase in his payoff, that is for $i = 1, \ldots, N$

$$J_i(u_1^*, \ldots, u_i, \ldots, u_N^*) \geq J_i(u_1^*, \ldots, u_i^*, \ldots, u_N^*) \qquad \forall u_i(\cdot) \tag{41}$$

It is important to note that Nash equilibria may not be unique. It is also easy to see that for 2 person zero sum games, a Nash equilibrium is a saddle solution.

As in the previous section on saddle solutions, we can write Hamilton Jacobi equations for Nash equilibria by defining Hamiltonians $H_i(x, u_1, \ldots, u_N, p, t)$ according to

$$H_i(x, u_1, \ldots, u_N, p, t) = L_i(x, u_1, \ldots, u_N) + p^T f(x, u_1, \ldots, u_N, t) \tag{42}$$

The conditions for a Nash equilibrium of equation (41) are there exist $u_i^*(x, p, t)$ such that

$$H_i(x, u_1^*, \ldots, u_i, \ldots, u_N^*, p, t) \geq H_i(x, u_1^*, \ldots, u_i^*, \ldots, u_N^*, p, t) \tag{43}$$

Then, we have N sets of Hamilton Jacobi equations for the N players satisfying the Hamilton Jacobi equations with $H_i^* = H_i^*(x, u_1^*, \ldots, u_N^*, p_i, t)$. Note that we have changed the costate variables to p_i to account for different Hamiltonians and boundary conditions.

$$\begin{aligned} \dot{x} &= \frac{\partial H_i^*}{\partial p_i}^T \\ \dot{p}_i &= -\frac{\partial H_i^*}{\partial x}^T \end{aligned} \tag{44}$$

with transversality conditions $p_i^T(t_f) = -D_1\phi_i(x(t_f), t_f)$.

4.2 Noncooperative Stackelberg solutions

Stackelberg or hierarchical equilibria refer to noncooperative solutions, where one or more of the players act as leaders. We will illustrate the solution concept for a two person game where player 1 is the leader and player 2 the follower. Once player 1 announces a strategy $u_1^o(\cdot)$, if player 2 is rational he choose his strategy so as to minimize his cost J_2 with the dynamics

$$\dot{x} = f(x, u_1^o, u_2, t) \tag{45}$$

and

$$J_2(u_2) = \phi_2(x(t_f), t_f) + \int_{t_0}^{t_f} L_2(x, u_1^o(t), u_2, t)dt$$

Thus, $u_2^*(u_1^o)$ is chosen to minimize $H_2(x, u_1^o, u_2, p_2, t)$, where p_2 satisfies the differential equation

$$\dot{p}_2 = -\frac{\partial H_2}{\partial x}^T (x, u_1^o, u_2(u_1^o), p, t) \quad p_2(t_f) = D_1^T\phi_2(x(t_f), t_f)$$

In turn the leader chooses his strategy to be that u_1^* which minimizes J_1 subject to the assumption that player 2 will rationally play $u_2^*(u_1^*)$. Thus, the system of equations that he has to solve to minimize J_1 subject to

$$\begin{array}{rll} \dot{x} = & f(x, u_1, u_2, t) & x(t_0) = x_0 \\ \dot{p}_2 = & -\frac{\partial H_2}{\partial x}^T(x, u_1^o, u_2(u_1^o), p, t) & p_2(t_f) = D_1^T \phi_2(x(t_f), t_f) \\ 0 = & D_3 H_2(x, u_1, u_2, p_2, t) & \end{array} \tag{46}$$

The last equation in (46) is the stationarity condition for minimizing H_2. The optimization problem of the system in (46) is not a standard optimal control in $\mathbb{R}^{2n}$ because there is an equality to be satisfied. Thus, Lagrange multipliers (co-states) taking values in $\mathbb{R}^{2n+n_2}$ for $t \in [t_0, t_f]$ are needed. We will omit the details in these notes.

References

[1] S. S. Sastry, *Lectures in Optimal Control and Dynamic Games*, Notes for the course EECS290A, Advanced Topics in Control Theory, University of California, Berkeley, 1996.

[2] M. Athans and P. Falb, *Optimal Control*, Mc Graw Hill, 1966.

[3] R. Bellman, *Dynamic Programming*, Princeton University Press, 1957.

[4] L. D. Berkovitz, *Optimal Control Theory*, Springer-Verlag, 1974.

[5] A. E. Bryson and Y-C. Ho, *Applied Optimal Control*, Blaisdell Publishing Company, Waltham, 1969.

[6] L. Pontryagin, V. Boltyanskii, R. Gamkrelidze, and E. Mischenko, *The Mathematical Theory of Optimal Processes*, Wiley-New York, 1962.

[7] L. C. Young, *Optimal Control Theory*, Cambridge University Press, 2nd edition, 1980.

[8] D. Kirk, *Optimal Control Theory: An Introduction*, Prentice Hall, 1970.

[9] F. Lewis, *Optimal Control*, Wiley-New York, 1986.

[10] W. Fleming and R. Rishel, *Deterministic and Stochastic Optimal Control*, Springer Verlag, 1975.

[11] V. Arnold, *Mathematical Methods of Classical Mechanics*, Springer Verlag, 2nd edition, 1989.

[12] J. von Neumann and O. Morgenstern, *Theory of Games and Economic Behavior*, Princeton University Press, 1947.

[13] T. Basar and G. Olsder, *Dynamic Noncooperative Games*, Academic Press, 2nd Edition, 1995.

AA278A Lecture Notes 9. Controllers for Safety for Continuous Systems

Claire J. Tomlin

May 19, 2005

The material in this lecture is based on the presentations in [1, 2, 3].

In the last lectures we have discussed optimal control and game theory for continuous systems: we highlighted mathematical tools (calculus of variations and dynamic programming) that may be used to solve these problems. In this lecture, we show how these methods may be applied to the problem of solving the reachability problem and synthesizing controllers for continuous systems:

$$\dot{x}(t) = f(x(t), u(t), d(t)) \tag{1}$$

where $x \in X$ where $X = \mathbb{R}^n$, $u \in U$ is the set of control inputs and $d \in D$ is the set of disturbance inputs, where U and D are convex and compact subsets of $\mathbb{R}^u$ and $\mathbb{R}^d$ respectively; f is a vector field, assumed to be globally Lipschitz in x and continuous in u and d; and the initial state $x(0) \in \text{Init}$ where $\text{Init} \subseteq \mathbb{R}^n$. We will use $\mathcal{U}$ to denote the set of piecewise continuous functions from $\mathbb{R}$ to U and $\mathcal{D}$ to denote the set of piecewise continuous functions from $\mathbb{R}$ to D.

Consider a set of states $F \subseteq \mathbb{R}^n$. Try to establish the maximal controlled invariant subset of F, i.e. the largest set of initial states for which there exists a controller that manages to keep all executions inside F. Somewhat informally this set can be characterized as:

$$W^* = \{x_0 \in \mathbb{R}^n \ : \ \exists u \in \mathcal{U} \ \forall d \in \mathcal{D}, \ \Box F(x_0, u, d)\}$$

To eliminate technical complications we assume that:

Assumption 1 *There exists a continuously differentiable function $l : \mathbb{R}^n \to \mathbb{R}$ such that:*

$$\begin{array}{lll} l(x) > 0 & \textit{if} & x \in F^o \\ l(x) = 0 & \textit{if} & x \in \partial F \\ l(x) < 0 & \textit{if} & x \in F^c \\ l(x) = 0 & \Rightarrow & \frac{\partial l}{\partial x}(x) \neq 0 \end{array}$$

The assumption implies that F is a closed set with non-empty interior, whose boundary is a $n-1$ dimensional manifold.

1 Dynamic Programming Solution

To apply the optimal control and differential game tools introduced in the previous lecture let $t_f = 0$, consider an arbitrary $t \leq 0$ and introduce the value function:

$$\hat{J}(x,t) = \max_{u \in \mathcal{U}_{[t,0]}} \min_{d \in \mathcal{D}_{[t,0]}} l(x(0))$$

Notice that this control problem involves no Lagrangian ($L \equiv 0$), just a terminal cost. The game obtained in this setting falls in the class of *pursuit-evasion* games [4]. The (u,d) obtained by the above optimal control problem is a Stackelberg equilibrium for the game between control and disturbance, with the control playing the role of the leader.

$\hat{J}$ can be computed using the dynamic programming tools discussed in the last two lectures. Introduce a Hamiltonian:

$$\begin{array}{rcl} H : \mathbb{R}^n \times \mathbb{R}^n \times \mathbb{R}^u \times \mathbb{R}^d & \longrightarrow & \mathbb{R} \\ (x,p,u,d) & \longmapsto & p^T f(x,u,d) \end{array}$$

Consider the optimal Hamiltonian:

$$H^*(x,p) = \max_{u \in U} \min_{d \in D} H(x,p,u,d)$$

Notice again that the minimization over u and d is pointwise, as opposed to over functions of time. Then, if $\hat{J}$ is continuously differentiable it satisfies:

$$\begin{array}{rcl} \frac{\partial \hat{J}}{\partial t}(x,t) & = & -H^*\left(x, \frac{\partial \hat{J}}{\partial x}(x,t)\right) \\ \hat{J}(x,0) & = & l(x) \end{array} \tag{2}$$

Notice that the evolution of the partial differential equation is "backwards" in time.

Consider the set:

$$\widehat{W}_t = \{x_0 \in X \ : \ \hat{J}(x_0,t) \geq 0\}$$

This is the set of all states for which starting at $x(t) = x_0$, there exists a controller for u such that for all disturbance trajectories $d \in \mathcal{D}_{[t,0]}$, $l(x(0)) \geq 0$ or, in other words, $x(0) \in F$. This is not quite what we need yet. We would like the set of all states for which there exists a u such that for all d and for all $t' \in [t,0]$, $x(t') \in F$. This excludes points in $\widehat{W}_t$ which leave F at some point in $[t,0]$ but re-enter it before time 0. This requirement can be encoded either after the computation of $\hat{J}$, or by modifying the Hamilton-Jacobi equation:

$$\begin{array}{rcl} \frac{\partial J}{\partial t}(x,t) & = & -\min\left\{0, H^*\left(x, \frac{\partial J}{\partial x}(x,t)\right)\right\} \\ J(x,0) & = & l(x) \end{array} \tag{3}$$

Compare this with the discrete Hamilton-Jacobi equation from Lecture 11:

$$\begin{array}{rcl} J(q,0) & = & \left\{ \begin{array}{ll} 1 & q \in F \\ 0 & q \in F^c \end{array} \right. \\ J(q,i-1) - J(q,i) & = & \min\{0, \max_{\sigma_1 \in \Sigma_1} \min_{\sigma_2 \in \Sigma_2} [\min_{q' \in R(q,\sigma_1,\sigma_2)} J(q',i) - J(q,i)]\} \end{array} \tag{4}$$

$R(q, \sigma_1, \sigma_2)$ essentially implements the spatial partial derivative of J along the dynamics of the system. The innermost minimization is not needed in the continuous case, as the continuous system is "deterministic".

As before, we seek a stationary solution to this equation. Assume that as $t \to -\infty$, $J(x,t)$ converges to a continuously differentiable function $J^* : X \to \mathbb{R}$.

Proposition 2 *The set $W^* = \{x \in X \ : \ J^*(x) \geq 0\}$ is the largest controlled invariant set contained in F.*

The solution to the partial differential equation also leads to a least restrictive controller that renders W^* invariant. Consider:

$$g(x) = \begin{cases} \left\{u \in U \ : \ \min_{d \in D} \left(\frac{\partial J^*(x)}{\partial x}\right)^T f(x,u,d) \geq 0\right\} & \text{if } x \in \partial W^* \\ U & \text{if } x \in (W^*)^\circ \cup (W^*)^c \end{cases} \tag{5}$$

2 Geometric interpretation

For an arbitrary time $t \leq 0$ define:

$$W_t = \{x \in X \ : \ J(x,t) \geq 0\}$$

Consider an $x \in \partial W_t$ and assume that:

$$\begin{aligned} & H^*\left(x, \frac{\partial J}{\partial x}(x,t)\right) < 0 \\ \Leftrightarrow \quad & \max_{u \in U} \min_{d \in D} H\left(x, \frac{\partial J}{\partial x}(x,t), u, d\right) < 0 \\ \Leftrightarrow \quad & \max_{u \in U} \min_{d \in D} \frac{\partial J}{\partial x}(x,t) f(x,u,d) < 0 \\ \Leftrightarrow \quad & \forall u \in U \ \ \exists d \in D \text{ such that } \frac{\partial J}{\partial x}(x,t) f(x,u,d) < 0 \end{aligned}$$

But $\frac{\partial J}{\partial x}(x,t)$ is the normal to the boundary of W_t at x, pointing inside W_t. Moreover, $\frac{\partial J}{\partial x}(x,t) f(x,u,d)$ is the inner product between this normal and the vector $f(x,u,d)$. Let θ be the angle between $\frac{\partial J}{\partial x}(x,t)$ and $f(x,u,d)$. Then:

$$\begin{aligned} \frac{\partial J}{\partial x}(x,t) f(x,u,d) > 0 \quad & \text{if} \quad \theta < \pi/2 \\ \frac{\partial J}{\partial x}(x,t) f(x,u,d) = 0 \quad & \text{if} \quad \theta = \pi/2 \\ \frac{\partial J}{\partial x}(x,t) f(x,u,d) < 0 \quad & \text{if} \quad \theta > \pi/2 \end{aligned}$$

Therefore, the above statement is equivalent to:

for all $u \in U$ there exists $d \in D$ such that the normal to ∂W_t at x pointing towards the interior of W_t makes an angle greater than $\pi/2$ with $f(x, u, d)$,

or, equivalently:

for all $u \in U$ there exists $d \in D$ such that $f(x, u, d)$ points outside W_t.

These are points at which whatever u does d can force them to leave the set W_t instantaneously. Notice that the order of the quantifiers in the above expression implies that d may depend on u, in addition to x and t. The part of the boundary of W_t where $H^* < 0$ is known as the "usable part" in the pursuit-evasion game literature.

Returning to the Hamilton-Jacobi equation, we see that for these points:

$$\begin{aligned} \frac{\partial J}{\partial t}(x,t) &= -\min\left\{0, H^*\left(x, \frac{\partial J}{\partial x}(x,t)\right)\right\} \\ &= -H^*\left(x, \frac{\partial J}{\partial x}(x,t)\right) \\ &> 0 \end{aligned}$$

Therefore, as t decreases, J also decreases. For these points on the boundary of W_t, J becomes negative instantaneously, and they "fall out of" W_t.

What if $H^* \geq 0$? A similar argument shows that in this case:

there exists $u \in U$ such that for all $d \in D$ the normal to ∂W_t at x pointing towards the interior of W_t makes an angle at most $\pi/2$ with $f(x, u, d)$,

or, equivalently:

there exists $u \in U$ such that for all $d \in D$, $f(x, u, d)$ either points inside W_t or is tangent to ∂W_t.

These are points for which there exists a choice of u that for all d forces the state to remain in W_t. Notice that the order of the quantifiers implies that u may only depend x and t, and not d. For these points:

$$\begin{aligned} \frac{\partial J}{\partial t}(x,t) &= -\min\left\{0, H^*\left(x, \frac{\partial J}{\partial x}(x,t)\right)\right\} \\ &= 0 \end{aligned}$$

Therefore, as t decreases, J remains constant. These are points that want to move towards the interior of W_t. The role of the outermost minimum is to ensure that the value of J does not increase for these points, so that W_t does not grow. This is to prevent states that have been labeled as unsafe (can reach F^c) from being relabeled as safe later on.

3 Examples: Two-Aircraft Collision Avoidance (no mode switching)

3.1 Angular velocities as control inputs

Consider the relative model of two aircraft (From Lecture 11, with no mode switching) for the case in which the linear velocities of both aircraft are fixed, $v_1, v_2 \in \mathbb{R}$, and the control inputs of the aircraft are the angular velocities, $u = \omega_1$ and $d = \omega_2$:

$$\begin{aligned} \dot{x}_r &= -v_1 + v_2 \cos\psi_r + u y_r \\ \dot{y}_r &= v_2 \sin\psi_r - u x_r \\ \dot{\psi}_r &= d - u \end{aligned} \tag{6}$$

with state variables $(x_r, y_r, \psi_r) \in \mathbb{R}^2 \times [-\pi, \pi)$ and control and disturbance inputs $u \in U = [\underline{\omega}_1, \overline{\omega}_1] \subset \mathbb{R}$, $d \in D = [\underline{\omega}_2, \overline{\omega}_2] \subset \mathbb{R}$. Without loss of generality (we scale the coefficients of u and d if this is not met), assume that $\underline{\omega}_i = -1$ and $\overline{\omega}_i = 1$, for $i = 1, 2$.

The set $F^c = G$ is defined in the relative frame:

$$G = \{(x_r, y_r) \in \mathbb{R}^2, \psi_r \in [-\pi, \pi) \mid x_r^2 + y_r^2 \leq 5^2\} \tag{7}$$

and the unsafe (or capture) set is defined as the interior of G

$$G^\circ = \{(x_r, y_r) \in \mathbb{R}^2, \psi_r \in [-\pi, \pi) \mid x_r^2 + y_r^2 < 5^2\} \tag{8}$$

which is a 5-mile-radius cylindrical block in the (x_r, y_r, ψ_r) space denoting the protected zone in the relative frame. The function $l(x)$ is defined as

$$l(x) = x_r^2 + y_r^2 - 5^2 \tag{9}$$

The optimal Hamiltonian is

$$H^*(x, p) = \max_{u \in U} \min_{d \in D} [-p_1 v_1 + p_1 v_2 \cos\psi_r + p_2 v_2 \sin\psi_r + (p_1 y_r - p_2 x_r - p_3) u + p_3 d] \tag{10}$$

Defining the *switching functions* $s_1(t)$ and $s_2(t)$, as

$$\begin{aligned} s_1(t) &= p_1(t) y_r(t) - p_2(t) x_r(t) - p_3(t) \\ s_2(t) &= p_3(t) \end{aligned} \tag{11}$$

the optimal control and disturbance u^* and d^* exist when $s_1 \neq 0$ and $s_2 \neq 0$ and are calculated as

$$\begin{aligned} u^* &= sgn(s_1) \\ d^* &= -sgn(s_2) \end{aligned} \tag{12}$$

The equations for $\dot{p}$ are obtained through Hamilton's equations and are

$$\begin{aligned} \dot{p}_1 &= u^* p_2 \\ \dot{p}_2 &= -u^* p_1 \\ \dot{p}_3 &= p_1 v_2 \sin\psi_r - p_2 v_2 \cos\psi_r \end{aligned} \tag{13}$$

with $p(0) = (x_r, y_r, 0)^T = \nu$, the outward pointing normal to ∂G at any point (x_r, y_r, ψ_r) on ∂G.

The usable part UP of ∂G is calculated with $\nu = (x_r, y_r, 0)^T$:

$$UP = \{(x_r, y_r, \psi_r) \in \partial G \mid -v_1 x_r + v_2(x_r \cos\psi_r + y_r \sin\psi_r) < 0\} \tag{14}$$

with boundary

$$\{(x_r, y_r, \psi_r) \in \partial G \mid -v_1 x_r + v_2(x_r \cos\psi_r + y_r \sin\psi_r) = 0\} \tag{15}$$

To solve for $p^*(t)$ and $x^*(t)$ for $t < 0$, we must first determine $u^*(0)$ and $d^*(0)$. Equations (12) are not defined at $t = 0$, since $s_1 = s_2 = 0$ on ∂G, giving rise to "abnormal extremals" [5] (meaning that the optimal Hamiltonian loses dependence on u and d at these points). Analogously to [4] (pages 442-443), we use an indirect method to calculate $u^*(0)$ and $d^*(0)$: at any point (x_r, y_r, ψ_r) on ∂G, the derivatives of the switching functions s_1 and s_2 are

$$\dot{s}_1 = y_r v_1 \tag{16}$$

$$\dot{s}_2 = x_r v_2 \sin\psi_r - y_r v_2 \cos\psi_r \tag{17}$$

For points $(x_r, y_r, \psi_r) \in \partial G$ such that $\psi_r \in (0, \pi)$ it is straightforward to show that $\dot{s}_1 > 0$ and $\dot{s}_2 > 0$, meaning that for values of t slightly less than 0, $s_1 < 0$ and $s_2 < 0$. Thus for this range of points along ∂G, $u^*(0) = -1$ and $d^*(0) = 1$. These values for u^* and d^* remain valid for $t < 0$ as long as $s_1(t) < 0$ and $s_2(t) < 0$. When $s_1(t) = 0$ and $s_2(t) = 0$, the optimal solution (u^*, d^*) switches and the computation of the boundary continues with the new values of u^* and d^*, thus introducing "kinks" into the boundary. These points correspond to loss of smoothness in the Hamilton-Jacobi equation. Figure 1 displays the resulting boundary $\{x \in X \mid J^*(x, t) = 0\}$, computed by solving the Hamilton-Jacobi equation locally using Hamilton's equations, The global solution to the Hamilton-Jacobi equation is shown in Figure 2, using the Hamilton-Jacobi equation solver of [3].

3.2 Linear velocities as control inputs

In modes in which the aircraft do not change their heading, only their linear velocities, the airspeeds vary over specified ranges: $u \in U = [\underline{v}_1, \overline{v}_1] \subset \mathbb{R}^+$, $d \in D = [\underline{v}_2, \overline{v}_2] \subset \mathbb{R}^+$, and model reduces to

$$\begin{aligned} \dot{x}_r &= -u + d\cos\psi_r \\ \dot{y}_r &= d\sin\psi_r \\ \dot{\psi}_r &= 0 \end{aligned} \tag{18}$$

Using the same set G as in the previous subsection, we can form the optimal Hamiltonian for this problem, and solve Hamilton's equations for $x(t)$ and $p(t)$ to compute the boundary of the reachable set, as before. This is for you to do in Problem 1 of Homework 3.

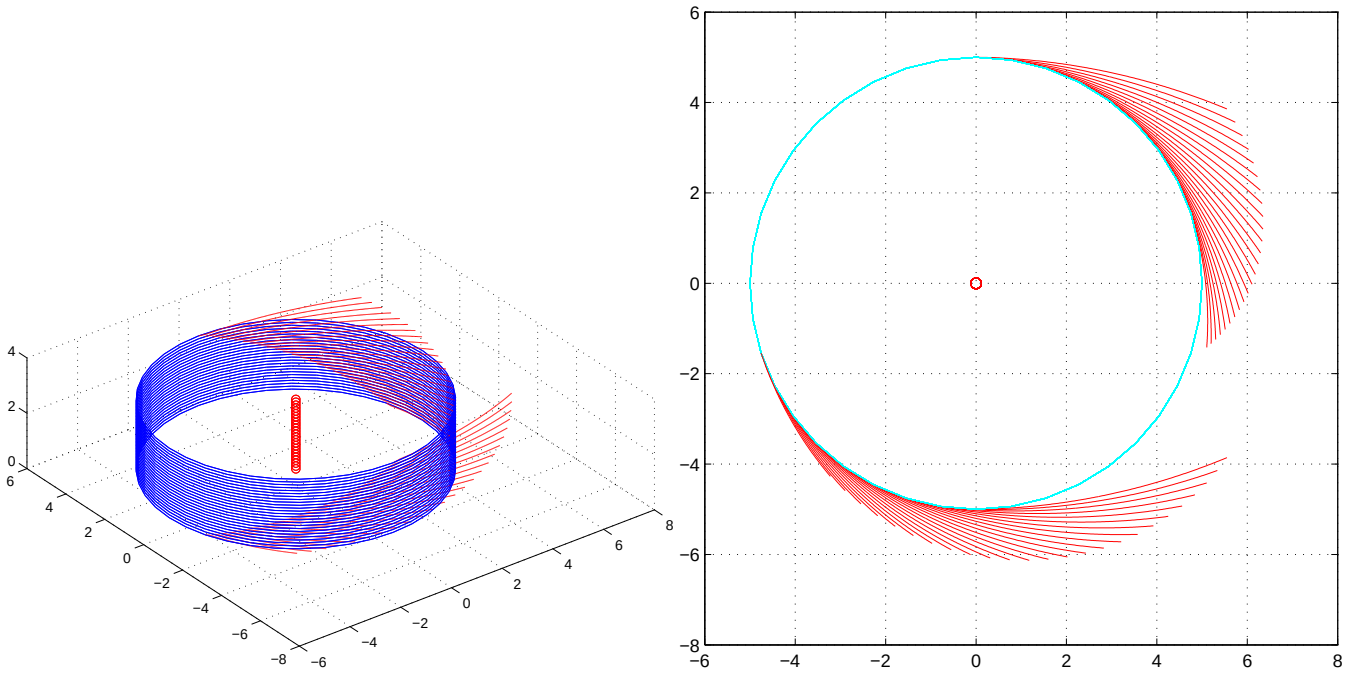


Figure 1: The set $G = \{(x_r, y_r), \psi_r \in (pi/4, \pi) \mid x_r^2 + y_r^2 \leq 5^2\}$ (cylinder) and the local representation of the boundary of the set $\{x \in X \mid J^*(x,t) = 0\}$, for fixed $t < 0$, using the solution of Hamilton's equations from the boundary of the usable part on G. The picture on the right is a top view of the one on the left.

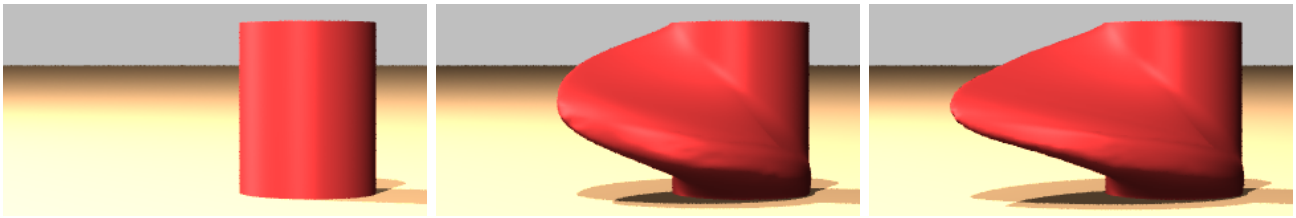


Figure 2: Level set solution to the collision avoidance example. Axes are (x_r, y_r, ψ_r), three plots are displayed to show the growth of the set, the set in the right subplot illustrates the fixed point, and it encloses all states which could eventually lead to collision under disturbance action. Animations are available at `http://cherokee.stanford.edu/~mitchell`.

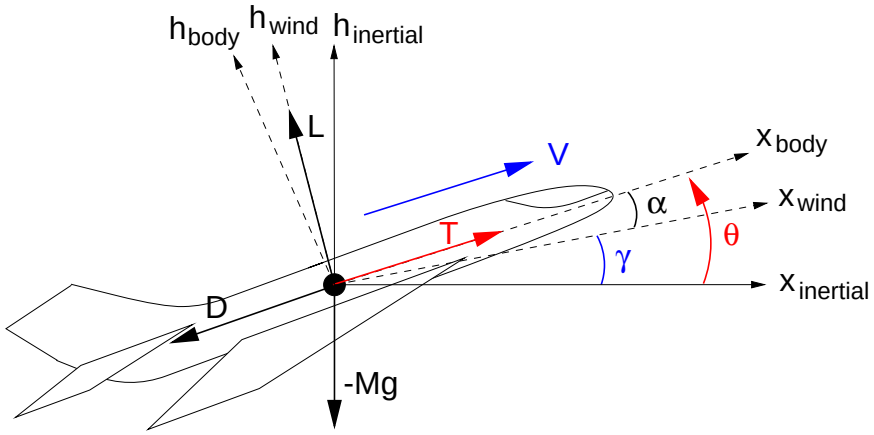


Figure 3: A planar aircraft in flight with attached axes about its center of mass.

4 Example: Aerodynamic Envelope Protection

This example was first inspired by the work of Charlie Hynes at NASA Ames [6], who recognized the problems in designing 'discrete' flight management systems for the continuous dynamics of an aircraft. At Ames, we solved the following 'aerodynamic envelope protection example', which seeks the maximal controlled invariant set contained within a given flight envelope. Subsequently, the guidance and control group at Honeywell Technology Center became interested in this work in the context of the 'flight mode switching' problem (as outlined in Lecture 1), and we extended the continuous problem below to the multiple mode hybrid system. The example presented below is based on our presentations in [7, 1].

We consider a nonlinear model of the longitudinal axis dynamics of a conventional take-off and landing (CTOL) aircraft in normal aerodynamic flight in still air [8] shown in Figure 3. The horizontal and vertical axes are respectively the $(x_{inertial}, h_{inertial})$ (denoted x, h) axes and the *pitch angle* θ is the angle made by the aircraft body axis, x_{body} with the x axis. The *flight path angle* γ and the *angle of attack* α are defined as: $\gamma = \tan^{-1}(\frac{\dot{h}}{\dot{x}})$, $\alpha = \theta - \gamma$. Expressions for the lift (L) and drag (D) forces are given by

$$\begin{aligned} L &= a_L(\dot{x}^2 + \dot{h}^2)(1 + c\alpha) \\ D &= a_D(\dot{x}^2 + \dot{h}^2)(1 + b(1 + c\alpha)^2) \end{aligned} \tag{19}$$

where a_L, a_D are dimensionless *lift* and *drag coefficients*, and b and c are positive constants. We assume that the autopilot has direct control over both the forward thrust T (throttle) and the aircraft pitch θ (through the elevators), thus there are two continuous control inputs $(u_1, u_2) = (T, \theta)$. Physical considerations impose constraints on the inputs:

$$u \in [T_{min}, T_{max}] \times [\theta_{min}, \theta_{max}] \tag{20}$$

The longitudinal dynamics may be modeled by the Newton-Euler equations:

$$M \begin{bmatrix} \ddot{x} \\ \ddot{h} \end{bmatrix} = R(\theta) \left[R^T(\alpha) \begin{bmatrix} -D \\ L \end{bmatrix} + \begin{bmatrix} T \\ 0 \end{bmatrix} \right] + \begin{bmatrix} 0 \\ -Mg \end{bmatrix} \tag{21}$$

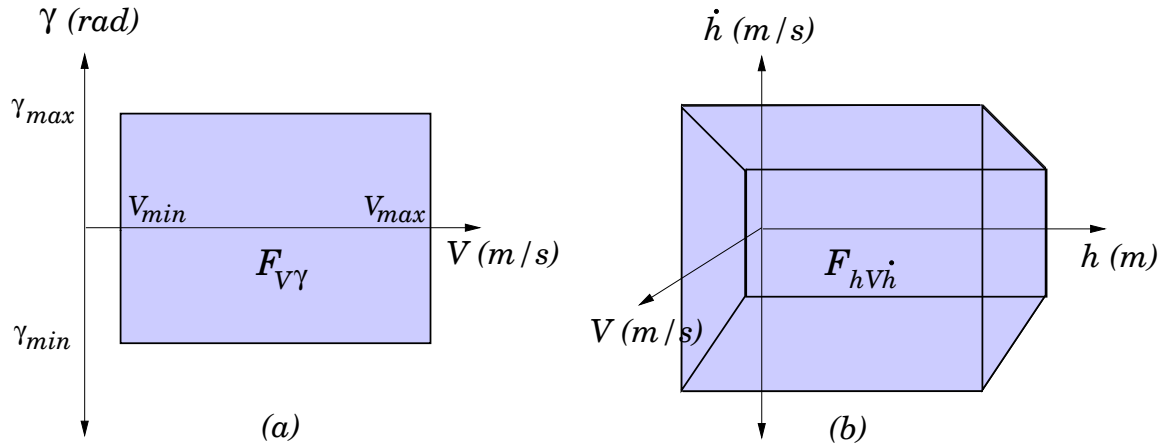


Figure 4: (a) Simplified Aerodynamic Flight Envelope in (V, γ)-space: axes are speed V, flight path angle γ; (b) Simplified Aerodynamic Flight Envelope in $(h, V, \dot{h})$-space: axes are altitude h, speed V, vertical speed $\dot{h}$.

where $R(\alpha)$ and $R(\theta)$ are standard rotation matrices, M is the mass of the aircraft, and g is gravitational acceleration. The state of the system is $\mathbf{x} = (x, \dot{x}, h, \dot{h})^T$.

The speed of the aircraft is defined as $V = \sqrt{\dot{x}^2 + \dot{h}^2}$. The simplified FMS studied here uses control inputs T and θ to control combinations of the speed V, flight path angle γ, and altitude h. The linear and angular accelerations $(\dot{V}, V\dot{\gamma})$ may be derived directly from (21):

$$\dot{V} = -\frac{D}{M} - g\sin\gamma + \frac{T}{M}\cos\alpha \tag{22}$$

$$V\dot{\gamma} = \frac{L}{M} - g\cos\gamma + \frac{T}{M}\sin\alpha \tag{23}$$

Note that these dynamics are expressed solely in terms of (V, γ) and inputs (T, θ), where $\alpha = \theta - \gamma$; thus equations (22), (23) are a convenient way to represent the dynamics for modes in which h is not a controlled variable.

Safety regulations for the aircraft dictate that V, γ, and h must remain within specified limits:

$$\begin{array}{c} V_{min} \le V \le V_{max} \\ \gamma_{min} \le \gamma \le \gamma_{max} \\ h_{min} \le h \le h_{max} \end{array} \tag{24}$$

where $V_{min}, V_{max}, \gamma_{min}, \gamma_{max}, h_{min}, h_{max}$ are functions of such factors as airspace regulations, type of aircraft, and weather. For aircraft flying in en-route airspace, we assume that these limits are constants, and thus the aerodynamic flight envelope F is as illustrated in Figure 4, in (V, γ)-space and $(h, V, \dot{h})$-space, where $\dot{h} = V\sin\gamma$. The state trajectory must remain within F at all times during en-route flight. We also impose a *secondary criterion*, that the state trajectory must satisfy constraints on the linear and angular acceleration:

$$|\dot{V}| \le 0.1g, \quad |V\dot{\gamma}| \le 0.1g \tag{25}$$

imposed for passenger comfort.

In our calculations we use the following parameter values, which correspond to a DC-8 at cruising speed: $M = 85000$kg, $b = 0.01$, $c = 6$, $a_L = 30$, $a_D = 2$, $T_{min} = 40000$ N, $T_{max} = 80000$ N, $\theta_{min} = -22.5°$, $\theta_{max} = 22.5°$, $V_{min} = 180$ m/s, $V_{max} = 240$ m/s, $\gamma_{min} = -22.5°$ and $\gamma_{max} = 22.5°$. The bounds on the pitch angle θ and the flight path angle γ are chosen to be symmetric about zero for ease of computation. In actual flight systems, the positive bound on these angles is greater than the negative bound. Also, the angles chosen for this example are greater than what are considered acceptable for passenger flight ($\pm 10°$). Since we are interested in en route flight, the limits on the altitudes are: $h_{min} = 15,000$ feet, $h_{max} = 51,000$ feet.

In this lecture, we consider the specification $\Box F_{V\gamma}$: the airspeed V and flight path angle γ must remain in the envelope $F_{V\gamma}$ at all times. We derive the maximal controlled invariant set contained in $F_{V\gamma}$, using the (V, γ)-dynamics (22), (23):

$$\begin{aligned} \dot{V} &= -\frac{D}{M} - g\sin\gamma + \frac{T}{M}\cos\alpha \\ V\dot{\gamma} &= \frac{L}{M} - g\cos\gamma + \frac{T}{M}\sin\alpha \end{aligned}$$

where $\alpha = \theta - \gamma$. Let

$$F_{V\gamma} = \{(V,\gamma) \mid \forall i \in \{1,2,3,4\},\ l_i(V,\gamma) \geq 0\} \tag{26}$$

where

$$l_1(V,\gamma) = V - V_{min} \tag{27}$$
$$l_2(V,\gamma) = -\gamma + \gamma_{max} \tag{28}$$
$$l_3(V,\gamma) = -V + V_{max} \tag{29}$$
$$l_4(V,\gamma) = \gamma - \gamma_{min} \tag{30}$$

$\partial F_{V\gamma}$ is only piecewise smooth, contradicting the assumption of existence of a differentiable function $l : (V,\gamma) \to \mathbb{R}$ such that $\partial F_{V\gamma} = \{(V,\gamma) \mid l(V,\gamma) = 0\}$. We show that, for this example, the calculation can in fact be performed one edge of the boundary at a time: we derive a Hamilton-Jacobi equation for each l_i, and prove that the intersection of the resulting sets is the maximal controlled invariant subset of $F_{V\gamma}$. The subscript i in each J_i, H_i will indicate that the calculation is for boundary l_i.

Starting with $l_1(V,\gamma)$, consider the system (22), (23) over the time interval $[t, 0]$, where $t < 0$, with cost function

$$J_1((V,\gamma), u(\cdot), t) : \mathbb{R}^+ \times \mathbb{R} \times \mathcal{U} \times \mathbb{R}_- \to \mathbb{R} \tag{31}$$

such that $J_1((V,\gamma), u(\cdot), t) = l_1(V(0), \gamma(0))$. Since there are no disturbances in our model, the dynamic game of Lecture 12 reduces to an optimal control problem. The optimal cost is found by maximizing with respect to u:

$$J_1^*((V,\gamma), t) = \max_{u(\cdot)\in\mathcal{U}} J_1((V,\gamma), u(\cdot), t) \tag{32}$$

We seek to compute $W_1^* = \{(V,\gamma) \mid J_1^*(V,\gamma) \geq 0\}$, which are those (V,γ) for which there exists a control input which keeps the system to the right of $l_1(V,\gamma) = 0$. The optimal Hamiltonian is given by the following, where we have substituted into the dynamics the expressions for the lift L and drag D forces (19) (neglecting the quadratic term in D):

$$H_1^*((V,\gamma),p) = \max_{u\in U}[p_1(-\frac{a_D V^2}{M} - g\sin\gamma + \frac{1}{M}T) + p_2(\frac{a_L V(1-c\gamma)}{M} - \frac{g\cos\gamma}{V} + \frac{a_L cV}{M}\theta)] \quad (33)$$

where $p = (p_1, p_2) \in \mathbb{R}^2$. The Hamilton-Jacobi equation describing the evolution of $J_1^*((V,\gamma),t)$ is obtained from the Hamilton-Jacobi equation of Lecture 13:

$$-\frac{\partial J_1^*(x,t)}{\partial t} = \min\{0, H_1^*((V,\gamma), \frac{\partial J_1^*((V,\gamma),t)}{\partial (V,\gamma)})\} \quad (34)$$

with boundary condition $J_1^*((V,\gamma),0) = l_1((V,\gamma))$.

The optimal control at $t = 0$ is computed from equation (33). The optimal throttle input T may be calculated directly from this equation: $u_1^*(0) = T_{max}$ (since $p_1 > 0$ for the inward pointing normal). The optimal pitch input must be calculated indirectly[1]. Define $(V_{min}, \gamma_a) = \{(V,\gamma) \mid l_1(V,\gamma) = 0 \cap H_1^*(V,\gamma) = 0\}$. Then:

$$\gamma_a = \sin^{-1}(\frac{T_{max}}{Mg} - \frac{a_D V_{min}^2}{Mg}) \quad (35)$$

Integrate the system dynamics (22), (23) with $(V(0),\gamma(0)) = (V_{min},\gamma_a)$, $u = (u_1^*, u_2^*)$, backwards from $t = 0$ to $t = -T$, where T is chosen to be large enough so that the solution intersects $\{(V,\gamma) \mid l_2(V,\gamma) = 0\}$. The optimal control u_2^* is required for this calculation. At the abnormal extremal (V_{min},γ_a), any $u_2 \in [\theta_{min}, \theta_{max}]$ may be used. However, as we integrate the system, we leave the abnormal extremal regardless of the choice of u_2 instantaneously, and u_2^* is uniquely determined. For all $u_2 \in [\theta_{min}, \theta_{max}]$, the inward pointing normal to the solution $(V(t),\gamma(t))$ of the system (22), (23), starting at (V_{min},γ_a) and proceeding backwards in time for small $t < 0$ using $u_1 = u_1^*$, is such that p_2 is negative. Thus, $u_2^* = \theta_{min}$. Denote the point of intersection of the solution of (22), (23) with $\{(V,\gamma) \mid l_2(V,\gamma) = 0\}$ as (V_a,γ_{max}), and the solution to (22), (23) between (V_{min},γ_a) and (V_a,γ_{max}) as ∂J^a, as shown in Figure 5. Repeat this calculation for the remaining three boundaries. Of the remaining three, only $\{(V,\gamma) \mid l_3(V,\gamma) = 0\}$ contains a point at which the associated optimal Hamiltonian, $H_3^*((V,\gamma),p)$, becomes zero. We denote this point as (V_{max},γ_b) where:

$$\gamma_b = \sin^{-1}(\frac{T_{min}}{Mg} - \frac{a_D V_{max}^2}{Mg}) \quad (36)$$

and similarly calculate ∂J^b and V_b, as shown in Figure 7.

Lemma 3 *For the aircraft dynamics (22), (23) with flight envelope $F_{V\gamma}$ given by (26), and input constraints (20), the maximal controlled invariant subset of $F_{V\gamma}$, denoted $W_{V\gamma}^*$, is the*

[1]Since $H_1^*((V,\gamma),p)$ loses dependence on u_2 on the set $\{(V,\gamma) \mid l_1(V,\gamma) = 0\}$, the calculations involve computing the so-called *abnormal extremals* [5].

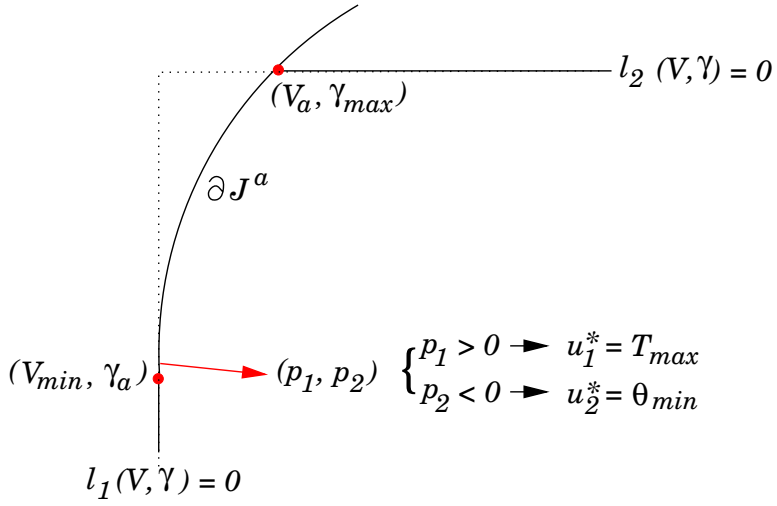


Figure 5: Computing the boundary ∂J^a.

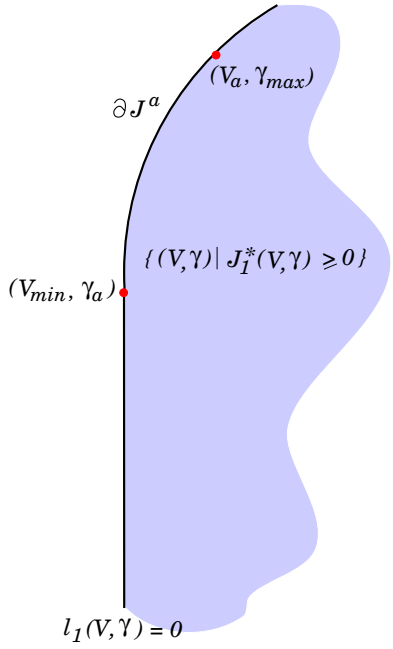


Figure 6: Computing the set $\{(V, \gamma) \mid J_1^*(V, \gamma) = 0\}$.

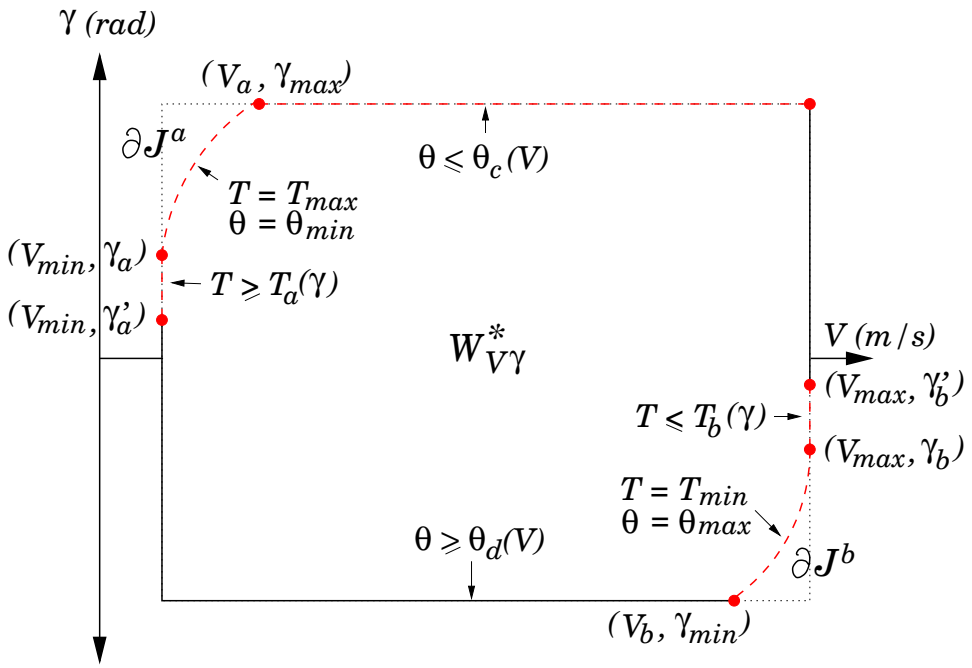


Figure 7: The set $W^*_{V\gamma}$ in (V,γ)-space, with control law as indicated. Values used are for a DC-8: $\gamma_{min} = -\pi/8$ rad, $\gamma_{max} = \pi/8$ rad, $V_{min} = 180$ m/s, $V_{max} = 240$ m/s, $\theta_{min} = -\pi/8$ rad, $\theta_{max} = \pi/8$ rad, $T_{min} = 40$ kN, $T_{max} = 80$ kN.

set enclosed by

$$\begin{aligned}\partial W^*_{V\gamma} = \{(V,\gamma) \mid\quad & (V = V_{min}) \wedge (\gamma_{min} \le \gamma \le \gamma_a) \quad \vee \\ & (V,\gamma) \in \partial J^a \quad \vee \\ & (\gamma = \gamma_{max}) \wedge (V_a \le V \le V_{max}) \quad \vee \\ & (V = V_{max}) \wedge (\gamma_b \le \gamma \le \gamma_{max}) \quad \vee \\ & (V,\gamma) \in \partial J^b \quad \vee \\ & (\gamma = \gamma_{min}) \wedge (V_{min} \le V \le V_b)\}\end{aligned} \tag{37}$$

Proof: We first prove that the boundary of the set $\cap_{i\in\{1,2,3,4\}}\{(V,\gamma) \mid J^*_i(V,\gamma) \ge 0\}$ is the boundary constructed in equation (37). We then prove that this set is equal to $W^*_{V\gamma}$, the maximal controlled invariant set contained in $F_{V\gamma}$.

Consider first the edge $\{(V,\gamma) \mid l_1(V,\gamma) = 0\}$ in ∂F. We will show that

$$\{(V,\gamma) \mid J^*_1(V,\gamma) = 0\} = \{(V,\gamma) \mid (V = V_{min}) \wedge (\gamma_{min} \le \gamma \le \gamma_a)\} \cup \{(V,\gamma) \in \partial J^a\} \tag{38}$$

The optimal Hamiltonian $H^*_1((V,\gamma),p)$ satisfies:

$$H^*_1((V,\gamma),p) \begin{cases} < 0 & (V,\gamma) \in F_{V\gamma} \cap l_1(V,\gamma) = 0 \cap \gamma > \gamma_a \\ = 0 & (V,\gamma) \in F_{V\gamma} \cap l_1(V,\gamma) = 0 \cap \gamma = \gamma_a \\ > 0 & (V,\gamma) \in F_{V\gamma} \cap l_1(V,\gamma) = 0 \cap \gamma < \gamma_a \end{cases} \tag{39}$$

Thus, the set $\{(V,\gamma) \mid (V = V_{min}) \wedge (\gamma_{min} \le \gamma \le \gamma_a)\}$ remains unchanged under the evolution of the Hamilton-Jacobi equation (34), since $H^*_1 > 0$ for this set. We now prove

that for $(V,\gamma) \in \partial J^a$, $J_1^*(V,\gamma) = 0$. $J_1^*(V,\gamma)$ satisfies:

$$\begin{bmatrix} \frac{\partial J_1^*(V,\gamma)}{\partial V} \\ \frac{\partial J_1^*(V,\gamma)}{\partial \gamma} \end{bmatrix} [-\frac{a_D V^2}{M} - g\sin\gamma + \frac{1}{M}T_{max}, \frac{a_L V(1-c\gamma)}{M} - \frac{g\cos\gamma}{V} + \frac{a_L cV}{M}\theta_{min}] = 0 \quad (40)$$

Since

$$\begin{bmatrix} \frac{\partial J_1^*(V,\gamma)}{\partial V} \\ \frac{\partial J_1^*(V,\gamma)}{\partial \gamma} \end{bmatrix} \quad (41)$$

is the inward pointing normal to $\{(V,\gamma) \mid J_1^*(V,\gamma) = 0\}$, then for each (V,γ) in $\{(V,\gamma) \mid J_1^*(V,\gamma) = 0\}$, the vector field

$$\begin{bmatrix} -\frac{a_D V^2}{M} - g\sin\gamma + \frac{1}{M}T_{max} \\ \frac{a_L V(1-c\gamma)}{M} - \frac{g\cos\gamma}{V} + \frac{a_L cV}{M}\theta_{min} \end{bmatrix} \quad (42)$$

is tangent to $\{(V,\gamma) \mid J_1^*(V,\gamma) = 0\}$. Thus the solution $(V(t),\gamma(t))$ to equations (22), (23) with $u = (T_{max}, \theta_{min})$ evolves along $J_1^*(V,\gamma) = 0$. Since, by construction, $(V,\gamma) \in \partial J^a$ satisfies equations (22), (23) with $u = (T_{max}, \theta_{min})$, then $(V,\gamma) \in \partial J^a$ satisfies $J_1^*(V,\gamma) = 0$.

Repeating this analysis for $\{(V,\gamma) \mid l_3(V,\gamma) = 0\}$, we can show that

$$\{(V,\gamma) \mid J_3^*(V,\gamma) = 0\} = \{(V,\gamma) \mid (V = V_{max}) \wedge (\gamma_b \le \gamma \le \gamma_{max})\} \cup \{(V,\gamma) \in \partial J^b\} \quad (43)$$

On the remaining boundaries, $H_2^*((V,\gamma),p) > 0$ and $H_4^*((V,\gamma),p) > 0$, so these boundaries remain unchanged under the evolution of their respective Hamilton-Jacobi equations.

It remains to prove that $W_{V\gamma}^* = \cap_{i\in\{1,2,3,4\}}\{(V,\gamma) \mid J_i^*(V,\gamma) \ge 0\}$. Clearly, any state (V,γ) for which there exists an i such that $J_i^*(V,\gamma) < 0$ must be excluded from $W_{V\gamma}^*$, since a trajectory exists which starts from this state and drives the system out of $\cap_{i\in\{1,2,3,4\}}\{(V,\gamma) \mid J_i^*(V,\gamma) \ge 0\}$. Thus $W_{V\gamma}^* \subset \cap_{i\in\{1,2,3,4\}}\{(V,\gamma) \mid J_i^*(V,\gamma) \ge 0\}$. To prove equality, we need only show that at the points of intersection of the four boundaries: $\{(V_a,\gamma_{max}), (V_{max},\gamma_{max}), (V_b,\gamma_{min}), (V_{min},\gamma_{min})\}$ there exists a control input in U which keeps the system state inside $\cap_{i\in\{1,2,3,4\}}\{(V,\gamma) \mid J_i^*(V,\gamma) \ge 0\}$. Consider the point (V_a,γ_{max}). At this point, the set of control inputs which keeps the system state inside the set $\{(V,\gamma) \mid J_1^*(V,\gamma) \ge 0\}$ is $\{(T_{max},\theta_{min})\}$, and the set of control inputs which keeps the system state inside $\{(V,\gamma) \mid J_2^*(V,\gamma) \ge 0\}$ is the set $\{(T,\theta) | T \in [T_{min}, T_{max}], \theta \in [\theta_{min}, \frac{M}{a_L V_a c}(\frac{g\cos\gamma_{max}}{V_a} - \frac{a_L V_a(1-c\gamma_{min})}{M})]\}$. Since these two sets have non-empty intersection, the intersection point $(V_a,\gamma_{max}) \in W_{V\gamma}^*$. Similar analysis holds for the remaining three intersection points. Thus $W_{V\gamma}^* = \cap_{i\in\{1,2,3,4\}}\{(V,\gamma) \mid J_i^*(V,\gamma) \ge 0\}$. ∎

Lemma 4 *The least restrictive controller that renders $W_{V\gamma}^*$ controlled invariant is $g(V,\gamma) = U \cap \hat{g}(V,\gamma)$, where:*

$$\begin{array}{lll} \hat{g}(V,\gamma) = \{ & \emptyset & \text{if } (V,\gamma) \in (W_{V\gamma}^*)^c \\ & T \ge T_a(\gamma) & \text{if } (V = V_{min}) \wedge (\gamma_{min} \le \gamma \le \gamma_a) \\ & \theta = \theta_{min} \wedge T = T_{max} & \text{if } (V,\gamma) \in \partial J^a \\ & \theta \le \theta_c(V) & \text{if } (\gamma = \gamma_{max}) \wedge (V_a \le V \le V_{max}) \\ & T \le T_b(\gamma) & \text{if } (V = V_{max}) \wedge (\gamma_b \le \gamma \le \gamma_{max}) \\ & \theta = \theta_{max} \wedge T = T_{min} & \text{if } (V,\gamma) \in \partial J^b \\ & \theta \ge \theta_d(V) & \text{if } (\gamma = \gamma_{min}) \wedge (V_{min} \le V \le V_b)\} \end{array} \quad (44)$$

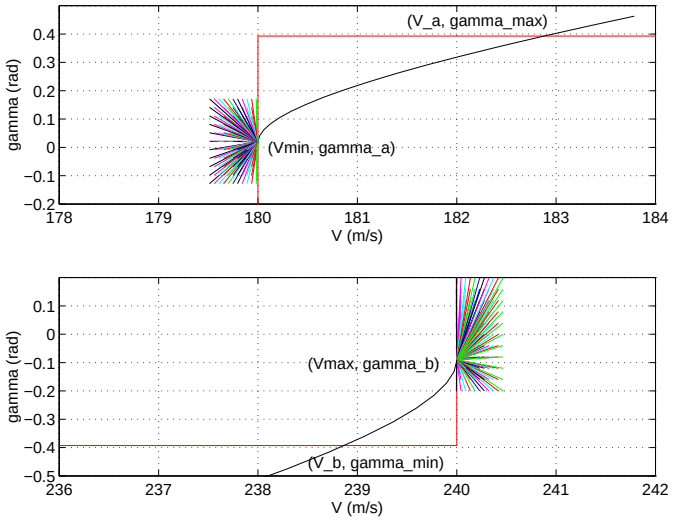


Figure 8: Upper left boundary and lower right boundary of $F_{V\gamma}$.

with

$$T_a(\gamma) = a_D V_{min}^2 + Mg\sin\gamma \tag{45}$$

$$T_b(\gamma) = a_D V_{max}^2 + Mg\sin\gamma \tag{46}$$

$$\theta_c(V) = \frac{M}{a_L V c}\left(\frac{g\cos\gamma_{max}}{V} - \frac{a_L V(1-c\gamma_{max})}{M}\right) \tag{47}$$

$$\theta_d(V) = \frac{M}{a_L V c}\left(\frac{g\cos\gamma_{min}}{V} - \frac{a_L V(1-c\gamma_{min})}{M}\right) \tag{48}$$

Proof: Consider the set $\{(V,\gamma) \mid (V = V_{min}) \wedge (\gamma_{min} \le \gamma \le \gamma_a)\}$. For each (V,γ) in this set, denote by $(T_a(\gamma), \theta_a(\gamma))$ the values of (T,θ) for which the vector field $(\dot{V},\dot{\gamma})$ becomes tangent to this set. These are the (T,θ) for which $\dot{V} = 0$: setting $\dot{V} = 0$ leads to equation (45) for all $\theta_a(\gamma) \in [\theta_{min}, \theta_{max}]$. Thus, $\{[T_a(\gamma), T_{max}] \times [\theta_{min}, \theta_{max}]\} \subseteq U$ keeps the system either tangent to or to the right side of the boundary $\{(V,\gamma) \mid (V = V_{min}) \wedge (\gamma_{min} \le \gamma \le \gamma_a)\}$. At the point (V_{min}, γ_a'), where $T_a(\gamma_a') = T_{min}$ the vector field cone $(\dot{V},\dot{\gamma})$ for $(T,\theta) \in U$ points completely inside $F_{V\gamma}$. At γ_a, the cone points completely outside $F_{V\gamma}$, and $T = T_{max}$ is the unique value of throttle which keeps the system trajectory $(V(t),\gamma(t))$ tangent to $F_{V\gamma}$. This is illustrated in Figure 8, which shows the upper left boundary of $F_{V\gamma}$, and the cone of controls at the point (V_{min}, γ_a).

The calculation may be repeated for the set $\{(V,\gamma) \mid (V = V_{max}) \wedge (\gamma_b \le \gamma \le \gamma_{max})\}$. Here, denote by $(T_b(\gamma), \theta_b(\gamma))$ the values of (T,θ) for which the vector field $(\dot{V},\dot{\gamma})$ becomes tangent to this set. Setting $\dot{V} = 0$ leads to equation (46) for all $\theta_b(\gamma) \in [\theta_{min}, \theta_{max}]$. Therefore, $\{[T_{min}, T_b(\gamma)] \times [\theta_{min}, \theta_{max}]\} \subseteq U$ keeps the system either tangent to or to the left side of the boundary $\{(V,\gamma) \mid (V = V_{max}) \wedge (\gamma_b \le \gamma \le \gamma_{max})\}$. At the point (V_{max}, γ_b), where $T_b(\gamma_b) = T_{min}$, T_{min} is the unique thrust which keeps the system trajectory tangent to $F_{V\gamma}$ (lower right boundary of $F_{V\gamma}$ in Figure 8).

Similar calculations along the upper and lower sides of $\partial F_{V\gamma}$ yield that the values of θ for

which the vector field becomes tangent to $\partial F_{V\gamma}$ are $\theta_c(V)$ and $\theta_d(V)$ of equations (47) and (48). ∎

In Figure 7, the portions of $W^*_{V\gamma}$ for which all control inputs are safe ($g(V,\gamma) = U$) are indicated with solid lines; those for which only a subset are safe ($g(V,\gamma) \subset U$) are indicated with dashed lines. The map defines the *least restrictive safe control scheme* and determines the mode switching logic. On ∂J^a and ∂J^b, the system must be in **Mode 2** or **Mode 3**. Anywhere else in $W^*_{V\gamma}$, any of the three modes is valid as long as the input constraints of equation (44) are satisfied. In the regions $F_{V\gamma} \backslash W^*_{V\gamma}$ (the upper left and lower right corners of $F_{V\gamma}$), no control inputs will keep the system inside of $F_{V\gamma}$.

Additional Constraints for Passenger Comfort

Cost functions involving the linear and angular accelerations can be used to encode the requirement for passenger comfort (we use J_5, J_6 in the following, after J_1 to J_4 of the previous section):

$$J_5((V,\gamma),u(\cdot),t)) = -\max_{t\geq 0} |\dot{V}(t)|, \quad J_6((V,\gamma),u(\cdot),t)) = -\max_{t\geq 0} |V(t)\dot{\gamma}(t)| \tag{49}$$

The requirement that the linear and angular accelerations remain within the limits determined for comfortable travel are encoded by thresholds:

$$J_5((V,\gamma),u(\cdot),t)) \geq -0.1g, \quad J_6((V,\gamma),u(\cdot),t)) \geq -0.1g \tag{50}$$

Within the class of safe controls, a control scheme which addresses the passenger comfort requirement can be constructed. To do this, we solve the optimal control problem:

$$J_5^*((V,\gamma)) = \max_{u(\cdot)\in g(V,\gamma)} J_5, \quad J_6^*((V,\gamma)) = \max_{u(\cdot)\in g(V,\gamma)} J_6 \tag{51}$$

From this calculation, it is straightforward to determine the set of "comfortable" states:

$$\{(V,\gamma) \in W^*_{V\gamma} \mid J_5^*(V,\gamma) \geq -0.1g \wedge J_6^*(V,\gamma) \geq -0.1g\} \tag{52}$$

The set of comfortable controls may be calculated by substituting the bounds on the accelerations into equation (22), (23) to get

$$\begin{array}{rcccl} -0.1Mg + a_D V^2 + Mg\sin\gamma & \leq & T & \leq & 0.1Mg + a_D V^2 + Mg\sin\gamma \\ -\frac{0.1Mg}{a_L V^2 c} - \frac{1-c\gamma}{c} + \frac{Mg\cos\gamma}{a_L V^2 c} & \leq & \theta & \leq & \frac{0.1Mg}{a_L V^2 c} - \frac{1-c\gamma}{c} + \frac{Mg\cos\gamma}{a_L V^2 c} \end{array} \tag{53}$$

These constraints provide lower and upper bounds on the thrust and the pitch angle which may be applied at any point (V,γ) in $W^*_{V\gamma}$ while maintaining comfort.

References

[1] J. Lygeros, C. Tomlin, and S. Sastry. Controllers for reachability specifications for hybrid systems. *Automatica*, 35(3):349–370, 1999.

[2] C. J. Tomlin, J. Lygeros, and S. Sastry. A game theoretic approach to controller design for hybrid systems. *Proceedings of the IEEE*, 88(7):949–970, July 2000.

[3] I. M. Mitchell, A. M. Bayen, and C. J. Tomlin. A Time-Dependent Hamilton-Jacobi Formulation of Reachable Sets for Continuous Dynamic Games. *IEEE Transactions on Automatic Control*, June 2005.

[4] T. Başar and G. J. Olsder. *Dynamic Non-cooperative Game Theory.* Academic Press, second edition, 1995.

[5] R. Montgomery. Abnormal minimizers. *SIAM Journal of Control and Optimization*, 32(6):1605–1620, 1994.

[6] C. Hynes and L. Sherry. Synthesis from design requirements of a hybrid system for transport aircraft longitudinal control. Preprint, NASA Ames Research Center, Honeywell Air Transport Division, 1996.

[7] C. Tomlin, J. Lygeros, and S. Sastry. Aerodynamic envelope protection using hybrid control. In *Proceedings of the American Control Conference*, pages 1793–1796, Phildelphia, PA, 1998.

[8] C. Tomlin, J. Lygeros, L. Benvenuti, and S. Sastry. Output tracking for a non-minimum phase dynamic CTOL aircraft model. In *Proceedings of the IEEE Conference on Decision and Control*, pages 1867–1872, New Orleans, LA, 1995.

AA278A Lecture Notes 10. Controller Synthesis for Hybrid Systems

Claire J. Tomlin

May 23, 2005

In the last two weeks, we have discussed controller synthesis for:

- Discrete Systems: design is characterized by a fixed point of a difference equation
- Continuous Systems: design is characterized by a fixed point of a partial differential equation

In each case, the solution is characterized as a fixed point of an equation, which we refer to as the Hamilton-Jacobi equation. In these lecture notes, we bring the discrete and continuous parts together, and discuss the problem of designing controllers for hybrid systems. Our treatment follows that of [1, 2, 3, 4].

1 Problem Formulation

Recall that the plant is modeled by a hybrid automaton $H = (Q, X, \text{Init}, In, f, \text{Dom}, R, Out)$ as described in Lecture Notes 7.

To avoid technical problems we assume that:

1. f is Lipschitz continuous in x and continuous in $w \in In$;
2. for all $q \in Q$ and for all $\sigma \in \Sigma, w \in W$, $\text{Dom}(q)$ is an open set.
3. for all $(q, x) \in Q \times X$, and for all $(\sigma_1, u) \in \Sigma_1 \times U$ there exists $(\sigma_2, d) \in \Sigma_2 \times D$ such that:
$$[(x, (\sigma_1, \sigma_2), (u, d)) \in \text{Dom}(q)] \vee [R(q, x, (\sigma_1, \sigma_2), (u, d)) \neq \emptyset]$$

Part 1 is standard, and is needed for existence of continuous evolution. Part 2 implies that we need not worry about what happens on the boundary of the invariant set. Finally, part 3 (together with part 2) implies that the controller cannot block the system execution. Notice that this assumption is not symmetric for (σ_1, u) and (σ_2, d). The reason is that, since we

are dealing with safety specifications, it will never be to the benefit of (σ_2, d) to stop the execution.

As in the discrete and continuous cases, we will try to establish the largest controlled invariant subset of a given set $F \subseteq Q \times X$, and design a controller that renders this set invariant. Again, to avoid technical problems, we will assume that F is a closed set.

We will again take an adversarial approach and treat the design as a game between (σ_1, u) and (σ_2, d). Whenever possible we will give the advantage to (σ_2, d), in particular:

1. In the order of play ((σ_1, u) will be the "leader" of the game).
2. When resolving non-determinism.

2 Definitions of Operators

Notice that the disturbance has two choices. It can:

1. Try to make the system "jump" outside F.
2. Try to "steer" the system outside F along continuous evolution.

The control also has two choices:

1. Try to "jump" to another state in F when the disturbance tries to steer out of F.
2. Try to "steer" the system and keep it in F along continuous evolution.

To characterize alternative 1 for the control, introduce the *controllable predecessor operator*, $\mathrm{Pre}_1 : 2^{Q \times X} \to 2^{Q \times X}$, which, given a set $K \subseteq Q \times X$ returns:

$$\begin{aligned} \mathrm{Pre}_1(K) \;=\; & \{(q,x) \in K : \exists (\sigma_1, u) \in \Sigma_1 \times U,\ \forall (\sigma_2, d) \in \Sigma_2 \times D,\ (x, \sigma_1, \sigma_2, u, d) \notin \mathrm{Dom}(q) \\ & \wedge R(q, x, \sigma_1, \sigma_2, u, d) \subseteq K\} \end{aligned}$$

To characterize alternative 1 for the disturbance, introduce the *uncontrollable predecessor operator*, $\mathrm{Pre}_2 : 2^{Q \times X} \to 2^{Q \times X}$, which, given a set $K \subseteq Q \times X$ returns:

$$\begin{aligned} \mathrm{Pre}_2(K) \;=\; & \{(q,x) \in K : \forall (\sigma_1, u) \in \Sigma_1 \times U\ \exists (\sigma_2, d) \in \Sigma_2 \times D \\ & R(q, x, \sigma_1, \sigma_2, u, d) \cap K^c \neq \emptyset\} \cup K^c \end{aligned}$$

Therefore $\mathrm{Pre}_1(K)$ contains all states in K for which controllable actions (σ_1, u) can force the state to remain in K for at least one step in the discrete evolution. $\mathrm{Pre}_2(K)$, on the other hand, contains all states in K^c, the complement of K, as well as all states from which uncontrollable actions (σ_2, d) may be able to force the state outside of K. In the definition of Pre_1, the controllable actions are required to be able to *force* a transition (hence the Inv in the formula). In contrast, for Pre_2, we simply require that a transition be possible, giving the

advantage to the uncontrollable actions. The controllable and uncontrollable predecessors will form the discrete part of the algorithm for computing controlled invariant sets.

Some simple facts about these two operators:

Proposition 1 *For all* $K \subseteq Q \times X$, $\mathrm{Pre}_1(K) \subseteq K$, $\mathrm{Pre}_2(K) \supseteq K^c$ *and* $\mathrm{Pre}_1(K) \cap \mathrm{Pre}_2(K) = \emptyset$

Remarks:

- The two operators are asymmetric.
- The order of the quantifiers is consistent with the control being the leader in the game.
- Since all non-determinism is resolved in favor of the disturbance, the control has to work harder:
  - it has to ensure a transition back into K exists (second condition in the definition of $\mathrm{Pre}_1(K)$),
  - it has to be able to "force" the transition (no mention of *Dom* in the definition of $\mathrm{Pre}_2(K)$),
  - it has to ensure all possible transitions stay in K (first condition in the definition of $\mathrm{Pre}_1(K)$).

Finally, to characterize alternative 2 for both control and disturbance, we define the Reach : $2^{Q \times X} \times 2^{Q \times X} \to 2^{Q \times X}$ operator:

Definition 2 (Reach) *Consider two subsets* $G \subseteq Q \times X$ *and* $E \subseteq Q \times X$ *such that* $G \cap E = \emptyset$. *The Reach operator is defined as*

$$\begin{aligned} \mathrm{Reach}(G, E) = \; & \{(q, x) \in Q \times X \mid \forall u \in \mathcal{U} \; \exists d \in \mathcal{D} \text{ and } t \geq 0 \text{ such that} \\ & (q(t), x(t)) \in G \text{ and } (q(s), x(s)) \in \Pi(\mathrm{Dom}) \setminus E \text{ for } s \in [0, t]\} \end{aligned} \tag{1}$$

where $(q(s), x(s))$ *is the continuous state trajectory of* $\dot{x} = f(q(s), x(s), u(s), d(s))$ *starting at* (q, x) *and* $\Pi(\mathrm{Dom})$ *represents the state space components of* Dom. *The set* $\mathrm{Reach}(G, E)$ *describes those states from which, for all* $u(\cdot) \in \mathcal{U}$, *there exists a* $d(\cdot) \in \mathcal{D}$, *such that the state trajectory* $(q(s), x(s))$ *can be driven to* G *while avoiding an "escape" set* E.

3 Basic Algorithm

Using the above definitions, the following algorithm can now be formulated for computing the largest controlled invariant subset of a given set F.

Algorithm 1 (Controlled Invariant Set)

Initialization:
$W^0 = F$, $W^1 = \emptyset$, $i = 0$
while $W^i \neq W^{i+1}$ **do**
begin
$W^{i-1} = W^i \setminus \mathrm{Reach}(\mathrm{Pre}_2(W^i), \mathrm{Pre}_1(W^i))$
$i = i - 1$
end

In the first step of this algorithm, we remove from F all states from which there is a disturbance $d(\cdot) \in \mathcal{D}$ forcing the system either outside F or to states from which an environment action $\sigma_2 \in \Sigma_2$ may cause transitions outside F, without first touching the set of states from which there is a control action $\sigma_1 \in \Sigma_1$ keeping the system inside F. Since at each step, $W^{i-1} \subseteq W^i$, the set W^i decreases monotonically as i decreases. If the algorithm terminates, we denote the fixed point as W^*.

In order to implement this algorithm, we need to calculate Pre_1, Pre_2, and Reach. The computation of Pre_1 and Pre_2 requires inversion of the transition relation R subject to the quantifiers $\exists$ and $\forall$; existence of this inverse can be guaranteed subject to well understood conditions on the map R. The computation of Reach, for each discrete state q, may be formulated as a constrained Hamilton-Jacobi equation or variational inequality.

Recall that along continuous evolution the value of the discrete state remains constant. Therefore, since the computation of the Reach operator involves only continuous evolution it can be carried out for each discrete state separately. Fix the value of $q \in Q$ and let $l_G : X \to \mathbb{R}$ and $l_E : X \to \mathbb{R}$ be differentiable functions such that $G \triangleq \{x \in X : l_G(x) \leq 0\}$ and $E \triangleq \{x \in X : l_E(x) \leq 0\}$.

Then the set of states which reaches G without entering E is

$$G(t) = \{x \in X : J_G(x,t) \leq 0\} \quad (2)$$
$$E = \{x \in X : J_E(x) \leq 0\} \quad (3)$$
$$(4)$$

where

$$\frac{\partial J_G(x,t)}{\partial t} + \min(0, H(x, \frac{\partial J_G(x,t)}{\partial x})) = 0 \quad (5)$$
$$\text{subject to } J_G(x,t) \geq J_E(x) \quad (6)$$

This is a constrained Hamilton-Jacobi equation – the constraint $J_G(x,t) \geq J_E(x)$ ensures that the evolution of $J_G^*(x,t)$ is frozen in set E. Figure 1 illustrates a sample evolution.

Remarks

In general, one cannot expect to solve for W^* using a finite computation. The class of hybrid systems for which algorithms like the one presented here are guaranteed to terminate

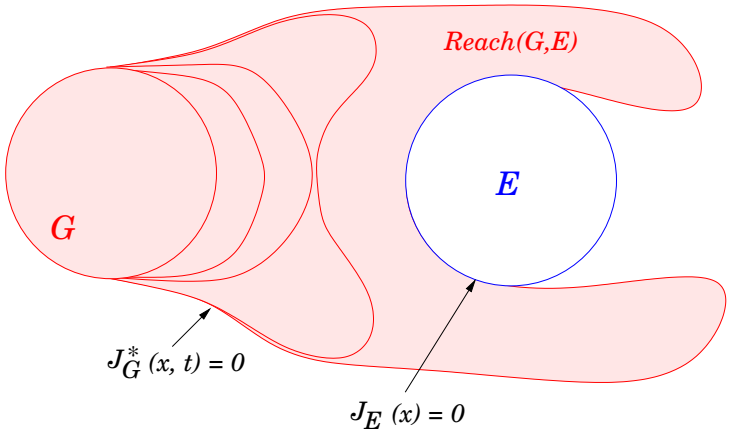


Figure 1: The computation of Reach(G, E) in a single discrete state q.

is known to be restricted [5]. In general, the basic algorithm of the previous section is semi-decidable when the operators $\mathrm{Pre}_1, \mathrm{Pre}_2, \mathrm{Reach}$ are computable. For example, when the continuous state dynamics are constant and the guards and resets are polyhedra, then the operators $\mathrm{Pre}_1, \mathrm{Pre}_2, \mathrm{Reach}$ map polyhedral sets back into polyhedral sets. These hybrid systems are referred to as *linear hybrid automata*.

Consider the three-mode aircraft conflict resolution example from Lecture 11.

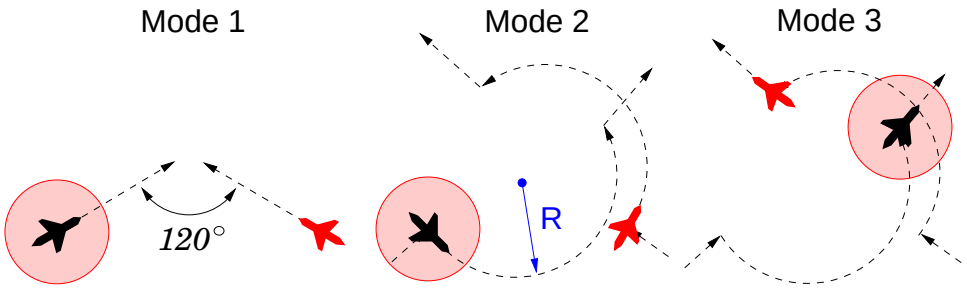


Figure 2: Two aircraft in three modes of operation: in modes 1 and 3 the aircraft follow a straight course and in mode 2 the aircraft follow a half circle. The initial relative heading (120°) is preserved throughout.

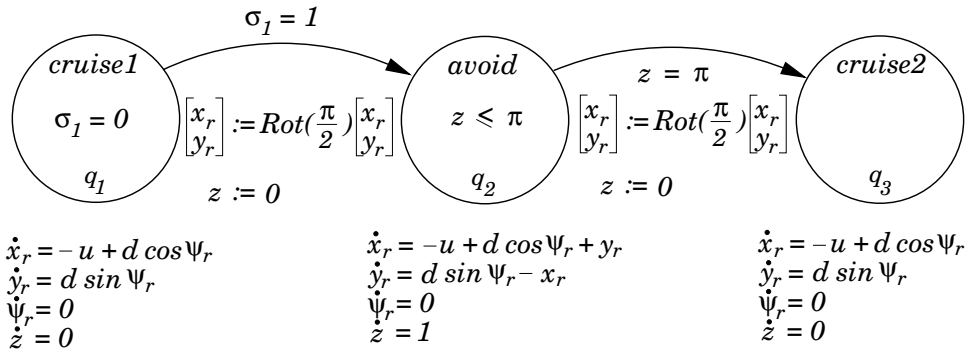


Figure 3: In q_1 both aircraft follow a straight course, in q_2 a half circle, and in q_3 both aircraft return to a straight course.

We assume that for this example the speeds (v_1, v_2) of both aircraft are constant even in the straight modes, so that the input and disturbance sets are singletons $(U = v_1, D = v_2)$ and $u^* = v_1, d^* = v_2$. The general case, in which U and D are ranges of possible speeds, is considered in the examples in [6, 7]. Recall that our goal is to calculate the relative distance at which the system may safely switch from mode 1 to mode 2, and the minimum turning radius R in mode 2, to ensure that separation between aircraft is maintained. The evolution of the protected zone in each mode, assuming no switches, is computed using the continuous-time Hamilton-Jacobi method. The unsafe set G is defined as: $G = \{q_1, q_2, q_3\} \times \{x \in X \mid l(x) \leq 0\}$ where $l(x) = x_r^2 + y_r^2 - 5^2$. Let $G_i = (q_i, \{x \in X \mid l(x) \leq 0\})$ represent the unsafe set in mode i. Thus the set $\{x \in X \mid J^*_{G_i}(x) \leq 0\}$ where $J^*_{G_i}$ is the optimal cost, is the backwards evolution of the protected zone in mode i, assuming no switches between modes.

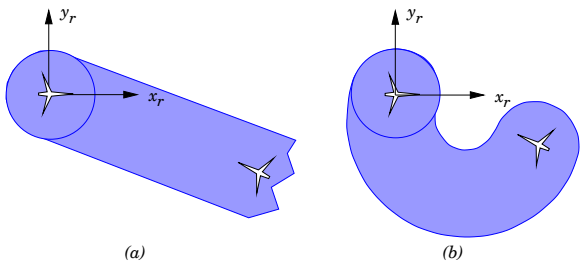


Figure 4: $J^*_{G_i}(x) \leq 0$ for (a) Modes 1 and 3 ($i = 1, 3$), $\omega_1 = \omega_2 = 0$ (the jagged edge means the set extends infinitely), (b) Mode 2 ($i = 2$), $\omega_1 = \omega_2 = 1$. In both cases, $\psi_r = 2\pi/3$, and $v_1 = v_2 = 5$.

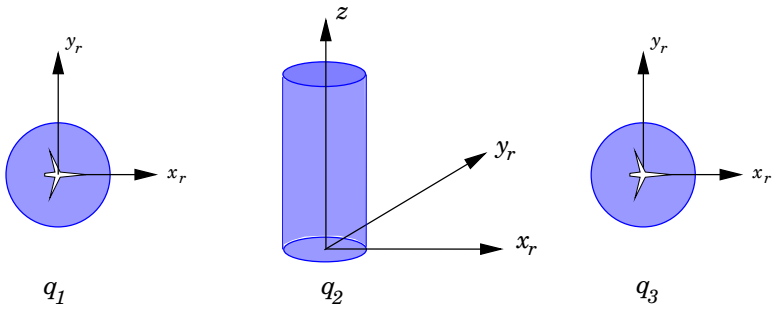


Figure 5: $(W^0)^c$.

These sets are shown in Figure 4. In both cases, the relative heading between aircraft is assumed fixed at $\psi_r = 2\pi/3$ (because of our assumption that aircraft switch modes instantaneously).

We implement Algorithm 1 for this example, at each step computing the sets Pre_1, Pre_2, and $\text{Reach}(\text{Pre}_2, \text{Pre}_1)$. In the first step, $W^0 = F \triangleq G^c$, the complement of G:

$$\begin{aligned} W^0 &= ((q_1, \{x \in X \mid l(x) \leq 0\}^c \cap \{x \in X \mid z = 0\}) \cup (q_2, \{x \in X \mid l(x) \leq 0\}^c) \\ &\quad \cup (q_3, \{x \in X \mid l(x) \leq 0\}^c \cap \{x \in X \mid z = 0\})) \end{aligned} \tag{7}$$

as shown in Figure 5 (the complement is shown in the figure).

$$\text{Pre}_1(W^0) = (q_1, \{x \in X \mid l(x) \leq 0\}^c \cap \{x \in X \mid z = 0\}) \tag{8}$$

$$\text{Pre}_2((W^0)^c) = G \tag{9}$$

Note that $\text{Pre}_1(W^i) \subseteq \{(q_1, X)\}$ for all i, since σ_1 labels transitions from q_1. The set W^{-1} (Figure 6) is

$$W^{-1} = W^0 \backslash \text{Reach}(\text{Pre}_2((W^0)^c), \text{Pre}_1(W^0)) \tag{10}$$

The set W^{-2} involves computing $\text{Reach}(\text{Pre}_2((W^{-1})^c), \text{Pre}_1(W^{-1}))$, this computation is illustrated in Figure 7(a) and the set is shown in Figure 7(b) as the shaded region. Continuing, a fixed point is reached after 3 iterations: Figure 8 illustrates this fixed point $W^* = W^{-3}$ in q_1. Since we assumed in this example that the continuous control input $u = v_1$ is fixed, we need only design the discrete part of the controller σ_1 and the radius of the maneuver R. The

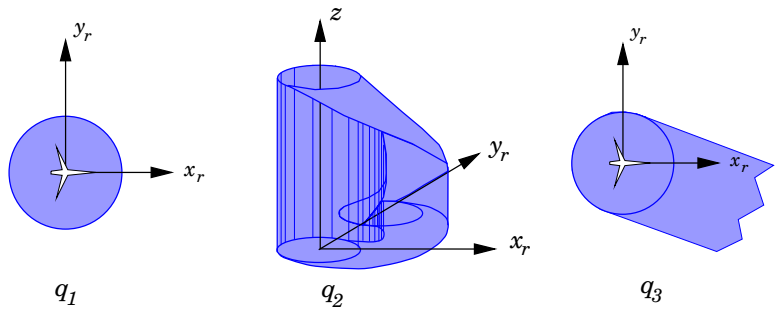


Figure 6: $(W^{-1})^c$. The jagged edge in q_3 means that the set extends infinitely.

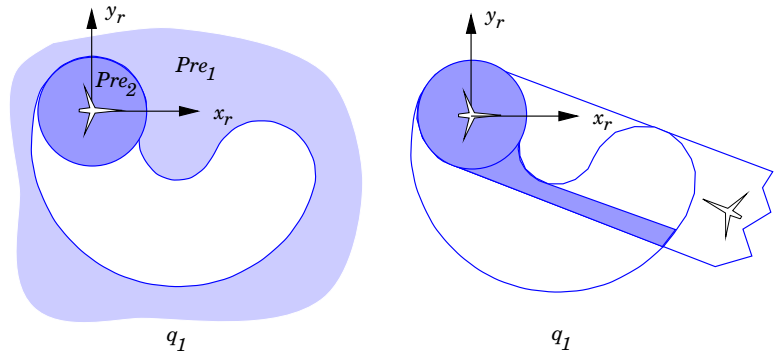


Figure 7: (a) $\mathrm{Pre}_1(W^{-1})$ and $\mathrm{Pre}_2(W^{-1})$ in q_1; (b) $\mathrm{Reach}(\mathrm{Pre}_2(W^{-1}), \mathrm{Pre}_1(W^{-1}))$ in q_1.

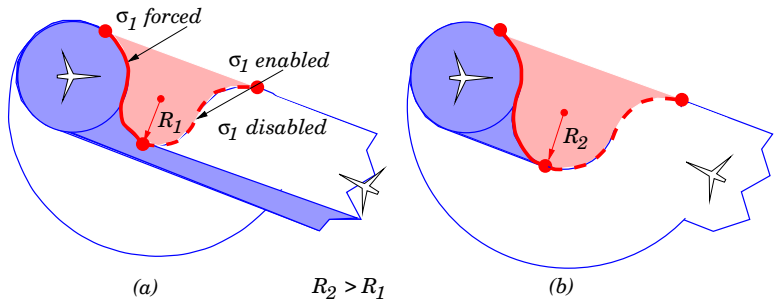


Figure 8: Showing the enabling and forcing boundaries for σ_1 in state q_1; and the result of increasing the radius of the turn in the avoid maneuver to increase W^*.

design is as illustrated in Figure 8(a): the enabling and forcing of σ_1 occurs at the boundary of W^* as shown, as explained below. The transition from q_1 to q_2, governed by σ_1, must be disabled until the relative position of the two aircraft reach the dashed line as shown, otherwise the aircraft will lose separation with each other either during the maneuver or after the maneuver is complete. At the dashed line, σ_1 is enabled, meaning the transition from q_1 to q_2 may occur at any time. σ_1 remains enabled until the dynamics reach the solid line (boundary of W^*), at which point it must be both enabled and forced: otherwise the aircraft lose separation immediately. Note that there are states (x_r, y_r) which are not rendered safe by the maneuver. Indeed, if the initial state is in the darker shaded region shown in Figure 8(a), then the aircraft are doomed to collide. Figure 8(b) displays the result of increasing the radius of the turn in q_2. Notice that the set W^* (the complement of the shaded region) increases as the turning radius increases. This implies that the maneuver renders a larger subset of the state space safe. Figure 8(b) shows the critical value of the turning radius, for which the maneuver is guaranteed to be safe, provided the conflict is detected early enough.

We have recently designed a tool for computing reachable sets for hybrid systems based on this level set technique [8, 3], have implemented it in C, and we have used it to compute reachable sets for several examples, including the first example in this paper. Using a grid spacing of $\Delta x = 0.1$ (or about 90000 grid points) each iteration of this example required about 1400 timesteps on a Sun UltraSparc 10 (a 300 MHz UltraSparc processor with 512 KB cache and 128 MB main memory).

Using this for the computation of the continuous evolution in the hybrid system algorithm, the three-mode conflict resolution example may be computed automatically as shown in Figure 9 below.

4 Other Computational Methods involving Approximations

Other methods have been presented for approximating the reach set calculation. One idea has been to use rectangular hybrid automata to approximate conservatively the reach set of general hybrid automata. This procedure consists of sub-dividing the state space into regions where one can find upper and lower bounds for each component of the right hand side of the continuous dynamics and using the reach set analysis for the resulting rectangular hybrid system. The package HyTech does precisely this computation provided that the guards and invariants are polyhedra [9]. A synthesis procedure based on this appears in the paper of Wong-Toi [10]. The main advantage of this approximation procedure is that it deals with a class of systems for which the synthesis algorithm is semi-decidable. The main drawback is that there is an exponential growth in the number of discrete states in approximating the continuous dynamics. The successor to HyTech is a package called HyperTech [11] which reduces the conservativeness of HyTech by using interval arithmetic with some systematic checks to reduce the divergence of interval arithmetic estimates to approximate reach sets. A controller design procedure using HyperTech has yet to be completed.

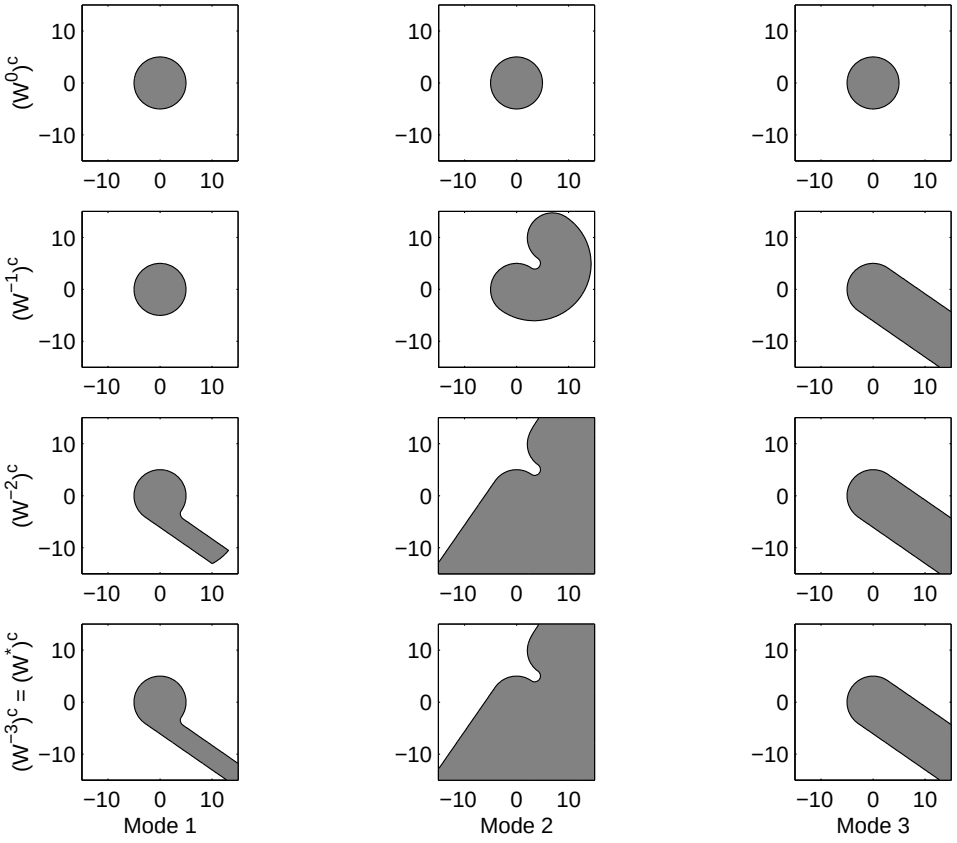


Figure 9: Unsafe Sets for Three Mode Example

Approximating Dynamics with Differential Inclusions. Suppose the continuous dynamics in the nonlinear hybrid system were approximated with the differential inclusion

$$\dot{x} \in g(q, x) \tag{11}$$

where $g(q, x) = \{f(q, x, u, d) \mid \forall u \in U, d \in D\}$. A computationally efficient method for approximating the reach set of $g(q, x)$ is to conservatively approximate $g(q, x)$ by a set of constant inclusions, each of the form

$$\dot{x} \in [g_{\min}, g_{\max}] \tag{12}$$

and then to compute the reach set of the constant inclusions. This method is presented in [12], [13] where it is proved that the approximation error can be made arbitrarily small by approximating the differential inclusion arbitrarily closely (ϵ-approximation). An advantage of this method is that the class of constant inclusions used to approximate the differential inclusion is known to be decidable, thus one can guarantee that the reachable set as $t \to -\infty$ can be computed in a finite number of steps. The amount of preprocessing required to initially approximate the dynamics may be quite formidable however, especially to achieve a close approximation of the true reach set.

Approximating non-smooth sets with smooth sets. We have shown that the reach set at any time $t \in (-\infty, 0]$ may have a non-smooth boundary due to switches in (u^*, d^*), non-smooth initial data, or the formation of shocks. The level set scheme propagates these discontinuities, yet its implementation may require a very small time step to do this accurately. In [14] we present a method for over-approximating such non-smooth sets with sets for which the boundary is continuously differentiable, by using smoothing functions to derive smooth inner and outer approximations. By applying Algorithm 2 to smooth inner and outer approximations of the sets G and E, we calculate smooth inner and outer approximations to the true reach set.

Ellipsoidal Methods. A similar idea is to use ellipsoids as inner and outer approximations to the reach set [15], [16]. To preserve the propagation of ellipsoids the continuous dynamics in each of the discrete locations needs to be approximated by linear dynamics. Bounds on the conservativeness of this approximation and their validity have not yet been worked out. However, [16] presents efficient algorithms for calculating both the minimum volume ellipsoid containing given points, and the maximum volume ellipsoid in a polyhedron, using a matrix determinant maximization procedure subject to linear matrix inequality constraints.

References

[1] J. Lygeros, C. Tomlin, and S. Sastry. Controllers for reachability specifications for hybrid systems. *Automatica*, 35(3):349–370, 1999.

[2] C. J. Tomlin, J. Lygeros, and S. Sastry. A game theoretic approach to controller design for hybrid systems. *Proceedings of the IEEE*, 88(7):949–970, July 2000.

[3] I. Mitchell, A. M. Bayen, and C. J. Tomlin. Validating a Hamilton-Jacobi approximation to hybrid system reachable sets. In M. D. Di Benedetto and A. Sangiovanni-Vincentelli, editors, *Hybrid Systems: Computation and Control*, LNCS 2034, pages 418–432. Springer Verlag, 2001.

[4] I. M. Mitchell, A. M. Bayen, and C. J. Tomlin. A Time-Dependent Hamilton-Jacobi Formulation of Reachable Sets for Continuous Dynamic Games. *IEEE Transactions on Automatic Control*, June 2005.

[5] T. A. Henzinger, P. W. Kopke, A. Puri, and P. Varaiya. What's decidable about hybrid automata. In *Proceedings of the 27th Annual ACM Symposium on Theory of Computing*, 1995.

[6] C. Tomlin, G. J. Pappas, and S. Sastry. Conflict resolution for air traffic management: A case study in multi-agent hybrid systems. *IEEE Transactions on Automatic Control*, 43(4):509–521, April 1998.

[7] Claire J. Tomlin. *Hybrid Control of Air Traffic Management Systems*. PhD thesis, Department of Electrical Engineering, University of California, Berkeley, 1998.

[8] I. Mitchell and C. Tomlin. Level set methods for computation in hybrid systems. In B. Krogh and N. Lynch, editors, *Hybrid Systems: Computation and Control*, LNCS 1790, pages 310–323. Springer Verlag, 2000.

[9] T. A. Henzinger, P. H. Ho, and H. Wong-Toi. A user guide to HYTECH. In E. Brinksma, W. Cleaveland, K. Larsen, T. Margaria, and B. Steffen, editors, *TACAS 95: Tools and Algorithms for the Construction and Analysis of Systems*, number 1019 in LNCS, pages 41–71. Springer Verlag, 1995.

[10] H. Wong-Toi. The synthesis of controllers for linear hybrid automata. In *Proceedings of the IEEE Conference on Decision and Control*, San Diego, CA, 1997.

[11] T. A. Henzinger, B. Horowitz, and R. Majumdar. Rectangular hybrid games. In *Proceedings of the 10th International Conference on Concurrence Theory (CONCUR)*. 1999.

[12] A. Puri. *Theory of Hybrid Systems and Discrete Event Systems*. PhD thesis, Department of Electrical Engineering, University of California, Berkeley, 1995.

[13] A. Puri, P. Varaiya, and V. Borkar. ϵ-approximation of differential inclusions. In *Proceedings of the IEEE Conference on Decision and Control*, pages 2892–2897, New Orleans, LA, 1995.

[14] J. Lygeros, C. Tomlin, and S. Sastry. On controller synthesis for nonlinear hybrid systems. In *Proceedings of the IEEE Conference on Decision and Control*, pages 2101–2106, Tampa, FL, 1998.

[15] A. B. Kurzhanski and I. Valyi. *Ellipsoidal calculus for estimation and control*. Birkhauser, Boston, 1997.

[16] L. Vandenberghe, S. Boyd, and S.-P. Wu. Determinant maximization with linear matrix inequality constraints. *SIAM Journal on Matrix Analysis and Applications*, 19(2):499–533, 1998.