AA278A Homework 1: Hybrid System Modeling.

Assigned April 11; Due April 26

Problem 1: Consider the discontinuous differential equation

$$\dot{x}_1 = -\mathrm{sgn}(x_1) + 2\mathrm{sgn}(x_2) \quad (1)$$

$$\dot{x}_2 = -2\mathrm{sgn}(x_1) - \mathrm{sgn}(x_2) \quad (2)$$

where $x(0) \neq (0,0)$, and

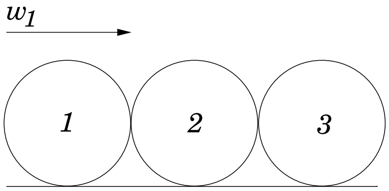
$$\mathrm{sgn}(z) = \begin{cases} 1 & \text{if } z > 0 \\ -1 & \text{if } z < 0 \\ \text{undefined} & \text{otherwise} \end{cases} \quad (3)$$

This system defines a hybrid automaton with four discrete modes having invariants corresponding to the four quadrants.

(a) Specify a non-blocking and deterministic hybrid automaton modeling the system.

(b) Does H accept Zeno executions for every initial state?

Problem 2: Consider three balls with unit masses and suppose that they are touching at time $t = 0$. The initial velocity of ball 1 is $w_1(0) = 1$ and balls 2 and 3 are at rest. Assume that the impact is a sequence of simple inelastic impacts. The first inelastic collision occurs between balls 1 and 2, resulting in $w_1(0+) = w_2(0+) = 0.5$ and $w_3(0+) = 0$. Since $w_2(0+) > w_3(0+)$, ball 2 hits ball 3 instantaneously giving $w_1(0++) = 0.5$ and $w_2(0++) = w_3(0++) = 0.25$. Now $w_1(0++) > w_2(0++)$ so ball 1 hits ball 2 again resulting in a new inelastic collision. This leads to an infinite sequence of collisions.



(a) Model the inelastic collisions of the three ball system as a hybrid automaton with a single discrete mode and three continuous variables (x_1, x_2, x_3) representing the velocities of the balls.

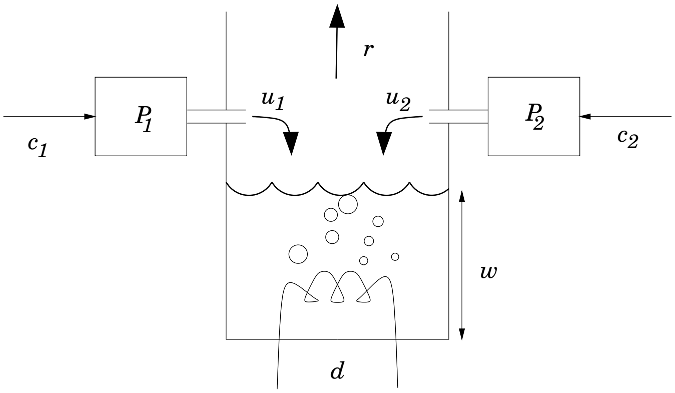
(b) Show that H accepts a Zeno execution corresponding to the sequence of collisions described above.

Problem 3: You may choose to do EITHER (a) or (b).

(a) A "benchmark" problem in hybrid systems analysis and control is that of the steam boiler [1].

The steam boiler consists of a tank containing water and a heating element that causes the water to boil and escape as steam. The water level in the boiler is denoted by w, and, for the sake of simplicity, consider only $w > 0$. The water is replenished by two pumps which at time t pump water into the boiler at rates $u_1(t)$ and $u_2(t)$ respectively. The water boils off at a rate r with d a variable that controls the rate of evaporation: $\dot{r} = d$. It is assumed the value of d at any given time is unknown, yet it is known to lie within given bounds. At every time t, pump i can be either be on ($u_i(t) = P_i$) or off ($u_i(t) = 0$). There is a delay T_i between the time pump i is ordered to switch on and the time u_i switches to P_i. There is no delay when the pumps are switched off.

The requirement is that the pumps are switched on and off so that the water level remains between two values M_1 and M_2.



(i) Derive a deterministic, non-blocking, hybrid automaton for the steam boiler.

(ii) Simulate the system for parameters: $P_i = 2.5$ and $T_i = 5$ (for $i = 1, 2$), $r \in [0, 4]$, $d \in [0, 0.5]$, $M_1 = 1, M_2 = 20$. Using your intuition, can you devise a pumping strategy that keeps the system within the allowable bounds, for possible worst case disturbance?

(b) Consider the inverted pendulum system in the paper handed out this week:
Kuipers and Ramamoorthy, "*Qualitative Modeling and Heterogeneous Control of Global System Behavior*", HSCC2002, LNCS 2289 pp 294-307.

Using the model and control scheme outlined in the paper, implement a controlled inverted pendulum simulation in Matlab. Is there a benefit to this hybrid control scheme against a continuous nonlinear control scheme?

Problem 4: Consider the water tank hybrid automaton given in Figure 4 of Lecture 3. Assume that $\max\{v_1, v_2\} < w < v_1 + v_2$, so that the water tank hybrid automaton is Zeno.

Temporal regularization of this system refers to the situation in which there is a delay $\epsilon > 0$ between the time the inflow is commanded to switch from one tank to the other, and the time the switch actually takes place.

Derive a new hybrid automaton for the water tank which incorporates this regularization, and show that this regularized automaton accepts a unique non-Zeno execution for each initial state.

References

[1] J. R. Abrial, E. Borger, and H. Langmaack. The steam boiler case study project: An introduction. In J. R. Abrial, E. Borger, and H. Langmaack, editors, *Formal Methods for Industrial Applications: Specifying and Programming the Steam Boiler Control*, LNCS 1165. Springer Verlag, 1996.

AA278A Homework 2: Stability of Hybrid Systems
Claire J. Tomlin.

Assigned May 3; Due May 19

If you make use of results from lecture notes or elsewhere, please state the result and the reference.

Problem 1: When Globally Quadratic Lyapunov Theory Fails.

Consider the linear hybrid system example from Lecture 6:

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

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

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

Simulation indicates the equilibrium $x_e = 0$ to be stable; yet there is no solution P to the LMI conditions:

$$\begin{aligned} P = P^T &> 0 \quad (1) \\ A_i^T P + P A_i &< 0 \quad (2) \end{aligned}$$

for $i = 1, 2$. As we saw in class, this makes sense, since when the system matrices A_1 and A_2 are interchanged, $x_e = 0$ is unstable.

Using the piecewise quadratic Lyapunov function theorem (Theorem 9) of Lecture 6, prove that $x_e = 0$ of the hybrid system described above is asymptotically stable. HINT 1: The same Lyapunov function actually works across both discrete states. HINT 2: This problem may be done very quickly using MATLAB's LMI toolbox (instructions at the end of Lecture 6).

Problem 2: Stabilizing unstable systems through switching. Give an example of a hybrid automaton that has unstable dynamics in each discrete mode, but for which the equilibrium $x_e = 0$ is stable. Illustrate your example through Matlab simulation. Prove, using one of the stability theorems from class, that your example is stable.

Problem 3.

Consider the following switching system

$$\dot{x} = A_q x$$

where $q \in \{1, 2\}$ and

$$\begin{aligned} A_1 &= \begin{bmatrix} -a_1 & b_1 \\ 0 & -c_1 \end{bmatrix} \\ A_2 &= \begin{bmatrix} -a_2 & b_2 \\ 0 & -c_2 \end{bmatrix} \end{aligned}$$

Assume that a_i, b_i, and c_i, for $i = 1, 2$ are real numbers and that $a_i, c_i > 0$. Show that the switched system is asymptotically stable.

Problem 4: Common Lyapunov Function for Commuting A-matrices. Assume that $K > 1$ matrices A_q, $q \in \{1, \ldots K\}$ are given. Consider the switched linear system $\dot{x} = A_q x$ where $\sigma : \mathbb{R}^n \to Q$ such that for all $(q, x) \in Q \times \mathbb{R}^n$, $f(q, x) = A_q x$. Assume that the matrices A_q are stable (ie. with eigenvalues in the open left half of the complex plane). Also, assume that, for all $i, j \in \{1, \ldots K\}$, $A_i A_j = A_j A_i$. Now, let $P_1, \ldots, P_m$ be the unique symmetric positive definite matrices that satisfy the Lyapunov equations:

$$\begin{aligned} A_1^T P_1 + P_1 A_1 &= -I && (3) \\ A_i^T P_i + P_i A_i &= -P_{i-1}, \qquad i = 2, \ldots, m && (4) \end{aligned}$$

Derive an explicit integral formula for P_m which only depends on the A_i, $i = 1, \ldots m$. Then show that the function $V(q, x) = x^T P_m x$ is a common Lyapunov function for the systems $\dot{x} = A_i x$, $i = 1, \ldots m$.

AA278A Homework 3: Control of Hybrid Systems.

Assigned May 20; Due June 7

We consider the motivational example given in Lecture Notes 7.

Problem 1: Two-aircraft collision avoidance, no mode switching.

Consider the case in which the aircraft follow straight paths only (mode 1 of the motivational example), and collision avoidance is achieved using linear velocity control only. Thus, the continuous inputs are the airspeeds of the aircraft ($u = v_1, d = v_2$) and assume that the airspeeds are known to 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

$$\begin{aligned} \dot{x}_r &= -u + d\cos\psi_r \\ \dot{y}_r &= d\sin\psi_r \\ \dot{\psi}_r &= 0 \end{aligned} \tag{1}$$

Design a MATLAB program which plots the subset of states which is doomed (whatever the controller does) to enter the 5-mile relative protected zone in T seconds. You can choose T to be anything you like; what happens as $T \to -\infty$?

For your code, use $[\underline{v}_1, \overline{v}_1] = [2, 4]$, $[\underline{v}_2, \overline{v}_2] = [1, 5]$, and consider four different values of ψ_r: $\pi/2$, 0, $-\pi/4$, and $-\pi/2$.

Problem 2: Two-aircraft collision avoidance, mode switching

Now consider the three mode example of Lectures Notes 7.

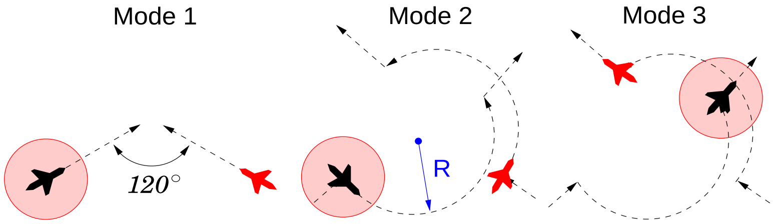


Figure 1: Two aircraft in three modes of operation: in modes 1 and 3 the aircraft follow a straight course and in mode 2 the aircraft follow a half circle. The initial relative heading (120°) is preserved throughout.

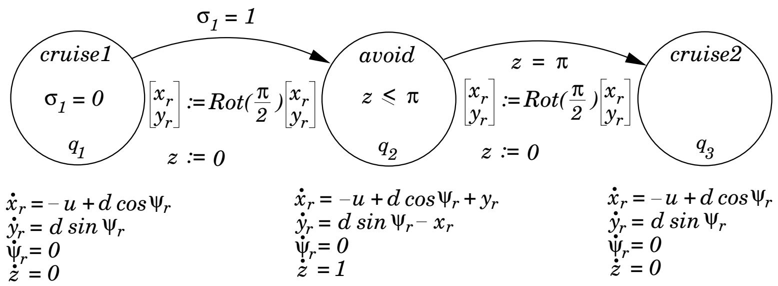
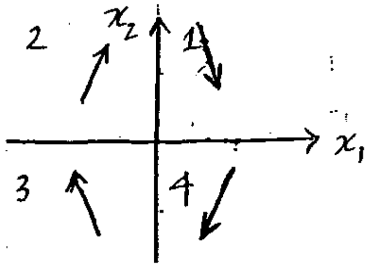

$\sigma_1 = 1$
cruise1
$\sigma_1 = 0$
q_1
$\begin{bmatrix} x_r \\ y_r \end{bmatrix} := Rot(\frac{\pi}{2})\begin{bmatrix} x_r \\ y_r \end{bmatrix}$
$z := 0$
avoid
$z \leq \pi$
q_2
$z = \pi$
$\begin{bmatrix} x_r \\ y_r \end{bmatrix} := Rot(\frac{\pi}{2})\begin{bmatrix} x_r \\ y_r \end{bmatrix}$
$z := 0$
cruise2
q_3
$\dot{x}_r = -u + d\cos\Psi_r$
$\dot{y}_r = d\sin\Psi_r$
$\dot{\Psi}_r = 0$
$\dot{z} = 0$
$\dot{x}_r = -u + d\cos\Psi_r + y_r$
$\dot{y}_r = d\sin\Psi_r - x_r$
$\dot{\Psi}_r = 0$
$\dot{z} = 1$
$\dot{x}_r = -u + d\cos\Psi_r$
$\dot{y}_r = d\sin\Psi_r$
$\dot{\Psi}_r = 0$
$\dot{z} = 0$

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.

Assume that, in the straight modes, $\omega_1 = \omega_2 = 0$, and in the circular arc mode, $\omega_1 = \omega_2 = 1$; and assume that, in all modes, $v_1 = v_2 = 5$. Assume that in all modes, $\psi_r = 2\pi/3$.

Show that by increasing the radius of the circular arc in the "avoid" mode, the set of states which is doomed (whatever the controller does) to enter the 5-mile relative protected zone decreases in size. You can use the code that you wrote for Problem 1 and the "overlapping set" argument presented in class, and answer this question using a set of illustrations.

1 (a) An example is $H = (Q, X, \text{Init}, f, \text{Dom}, R)$ with $Q = \{1,2,3,4\}$, $X = \mathbb{R}^2$



$$f(1,\cdot) = \begin{bmatrix} 1 \\ -3 \end{bmatrix} \qquad f(3,\cdot) = \begin{bmatrix} -1 \\ 3 \end{bmatrix}$$

$$f(2,\cdot) = \begin{bmatrix} 3 \\ 1 \end{bmatrix} \qquad f(4,\cdot) = \begin{bmatrix} -3 \\ -1 \end{bmatrix}$$

Init (away from $x_1 = 0$ or $x_2 = 0$)

$$\begin{aligned} \text{Inv} = &(1, \{x : x_1 > 0, x_2 > 0\}) \cup \\ &(2, \{x : x_1 < 0, x_2 > 0\}) \cup \\ &(3, \{x : x_1 < 0, x_2 < 0\}) \cup \\ &(4, \{x : x_1 > 0, x_2 < 0\}) \end{aligned}$$

$$\begin{aligned} &R(1, \{x : x_2 \leq 0\}) = (4, \begin{bmatrix} x_1 \\ x_2 := 0^- \end{bmatrix}) \\ &R(2, \{x : x_1 \geq 0\}) = (1, \begin{bmatrix} x_1 := 0^+ \\ x_2 \end{bmatrix}) \\ &R(3, \{x : x_2 \geq 0\}) = (2, \begin{bmatrix} x_1 \\ x_2 := 0^+ \end{bmatrix}) \\ &R(4, \{x : x_1 \leq 0\}) = (3, \begin{bmatrix} x_1 := 0^- \\ x_2 \end{bmatrix}) \\ &R = \emptyset \text{ otherwise} \end{aligned}$$

Since $R(q,x) \neq \emptyset$ for all $(q,x) \in \text{Trans}$ H is non-blocking. Since $|R(q,x)| \leq 1$ for all (q,x), H is deterministic.

1 (b) FIRST, NOTE that for every execution there exists i such that $x(\tau_i) = (a, b)^T$ where either $a = 0$ or $b = 0$. Then $x(\tau_i') = (c, d)^T$ where $(c, d) = (\frac{b}{3}, 0)$ if $a = 0$ and $(0, -\frac{a}{3})$ if $b = 0$.
It thus follows that H accepts a unique infinite execution for every initial state. The execution is zeno because for all $t \in \tau$

$$\dot{W}(t) = \frac{d}{dt}\left(\|x_1(t)\| + \|x_2(t)\|\right) = -2$$

where $W(t) = \|x_1(t)\| + \|x_2(t)\|$ (the l_1-norm)

so $W(t) = 0$ for some finite t and thus the origin is reached in finite time.

2 (a) An example is $H = (Q, X, \text{Init}, f, \text{Dom}, R)$ where $Q = \{q\}$, $X = \mathbb{R}^3$

$$f(q, \cdot) = [0, 0, 0]^T$$
$$\text{Init} = (q, (1, 0, 0)^T)$$
$$\text{Dom} = (q, (0, 0, 0)^T)$$

and

$$R(q, (x_1, x_2, x_3)) = \begin{cases} (q, ((x_1 + x_2)/2, (x_1 + x_2)/2, x_3)) & \text{if } x_1 > x_2 \\ (q, (x_1, (x_2 + x_3)/2, (x_2 + x_3)/2)) & \text{if } x_2 > x_3 \\ \emptyset & \text{otherwise} \end{cases}$$

2(b). Define $\Delta_i = x_1(\tau_i) - x_2(\tau_i) + x_2(\tau_i) - x_3(\tau_i)$ for all i with $\chi = (\tau, q, x) \in \mathcal{E}_H(q, (100)^T$.

Then $\Delta_i = 2^{-i} > 0$ so χ is an infinite execution. It is Zeno because $\tau_i = \tau_0$ for all i.

Since $x_1(\tau_i) > x_2(\tau_i)$ and $x_2(\tau_i) > x_3(\tau_i)$ it follows from $\lim_{i\to\infty} \Delta_i = 0$ that

$$\lim_{i\to\infty} x_1(\tau_i) = \lim_{i\to\infty} x_2(\tau_i) = \lim_{i\to\infty} x_3(\tau_i)$$

But $x_1(\tau_i) + x_2(\tau_i) + x_3(\tau_i) = 1$, so the Zeno state of χ is thus equal to $(\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$.

4

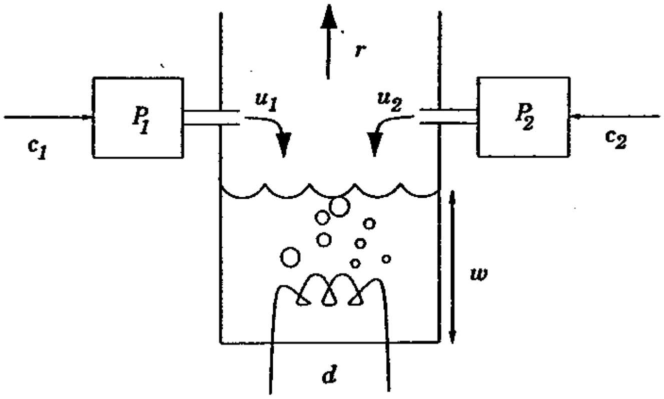


Figure 1: The Steam Boiler

Problem 3: Steam Boiler (Steam Boiler System, Figure 1)

The steam boiler consists of a tank containing water and a heating element that causes the water to boil and escape as steam. The water is replenished by two pumps which at time t pump water into the boiler at rates $u_1(t)$ and $u_2(t)$ respectively. At every time t, pump i can either be on ($u_i(t) = \overline{P}_i$) or off ($u_i(t) = 0$). There is a delay $\overline{T}_i$ between the time pump i is ordered to switch on and the time q_i switches to P_i. There is no delay when the pumps are switched off. We will use three hybrid automata to describe this system, one for the boiler, $B = (Q_B, X_B, V_B, Y_B, Init_B, f_B, h_B, Inv_B, E_B, G_B, R_B)$, and one for each of the pumps, $P_i = (Q_i, X_i, V_i, Y_i, Init_i, f_i, h_i, Inv_i, E_i, G_i, R_i)$.

The boiler automaton is defined by:

- $Q_B = \{q_B\}$, $\mathrm{Q}_B = \{BOILING\}$;
- $X_B = \{w, r\}$, $\mathrm{X}_B = \mathrm{R}^2$;

you can ignore here for now

- $V_B = \{u_1, u_2, d\}$, $\mathrm{V}_B = [0, \overline{P}_1] \times [0, \overline{P}_2] \times [-D_1, D_2]$, where $\overline{P}_1, \overline{P}_2, D_1, D_2 > 0$;
- $Y_B = \{y_1, y_2\}$, $\mathrm{Y}_B = \mathrm{R}^2$;
- $Init = \{BOILING\} \times [0, W] \times [0, R]$, where $W, R > 0$;
- $P_1 + P_2 - r$

$$f(BOILING, w, r, u_1, u_2, d) = \begin{bmatrix} \dot{w} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} u_1 + u_2 - r \\ d \end{bmatrix}$$

- $$h(BOILING, w, r) = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} w \\ r \end{bmatrix}$$

- $Inv(BOILING) = \mathrm{X}_B \times \mathrm{V}_B$ $=: Dom$
- edges $E = \emptyset$

Notice that since $E = \emptyset$, G and R are trivial and need not be explicitly defined.

The automaton for pump i is defined by:

- $Q_i = \{q_i\}$, $\mathbf{Q}_i = \{OFF, GOING\_ON, ON\}$;
- $X_i = \{T_i\}$, $\mathbf{X}_i = \overset{\mathbb{R}}{\mathrm{R}}$; reals
- $V_i = \{c_i\}$, $\mathbf{V}_i = \{0, 1\}$;
- $Y_i = \{u_i\}$, $\mathbf{Y}_i = [0, \overline{P}_i]$;
- $Init = \{OFF\} \times \{0\}$,
- $$f(q_i, T_i, c_i) = \begin{cases} 1 & \text{if } q_i = GOING\_ON \vee q_i = ON \\ 0 & \text{if } q_i = OFF \end{cases}$$
- $$h(q_i, T_i) = \begin{cases} 0 & \text{if } q_i = OFF \vee q_i = GOING\_ON \\ \overline{P}_i & \text{if } q_i = ON \end{cases}$$
- $$Dom := Inv(q_i) = \begin{cases} \mathbf{X}_i \times \{c_i = 0\} & \text{if } q_i = OFF \\ \{T_i \leq \overline{T}_i\} \times \{c_i = 1\} & \text{if } q_i = GOING\_ON \\ \mathbf{X}_i \times \{c_i = 1\} & \text{if } q_i = ON \end{cases}$$

edges
- $E = \{(OFF, GOING\_ON), (GOING\_ON, OFF), (GOING\_ON, ON), (ON, OFF)\}$

guards
- $$G(e) = \begin{cases} \mathbf{X}_i \times \{c_i = 1\} & \text{if } e = (OFF, GOING\_ON) \\ \mathbf{X}_i \times \{c_i = 0\} & \text{if } e = (GOING\_ON, OFF) \\ \{T_i \geq \overline{T}_i\} \times \{c_i = 1\} & \text{if } e = (GOING\_ON, ON) \\ \mathbf{X}_i \times \{c_i = 0\} & \text{if } e = (ON, OFF) \end{cases}$$

resets.
- $$R(e, T_i, c_i) = \begin{cases} \{0\} & \text{if } e = (OFF, GOING\_ON) \\ \{0\} & \text{if } e = (GOING\_ON, OFF) \\ \{T_i\} & \text{if } e = (GOING\_ON, ON) \\ \{0\} & \text{if } e = (ON, OFF) \end{cases}$$

⑥

The controller synthesis for this model can be found in [4]. The continuous dynamics of the boiling process are summarized by the differential equations:

$$\begin{aligned} \dot{w} &= p_1 + p_2 - r \\ \dot{r} &= d \end{aligned}$$

The dynamics of pump i are summarized by the open hybrid automaton of Figure 2. Notice that p_i is both an output variable of the pump and an input variable of the boiling process. For a formal definition of the model (with slight differences in the notation) ~~please refer to Lecture 8~~.

The composite automaton has 4 continuous, real valued state variables:

$$x = (w, r, T_1, T_2) \in \mathbb{R}^4$$

SOLUTION FROM "Controllers for Reachability Specifications for Hybrid Systems" Lygeros, Tomlin, Sastry Automatica 1999.

Figure 1: The Steam Boiler

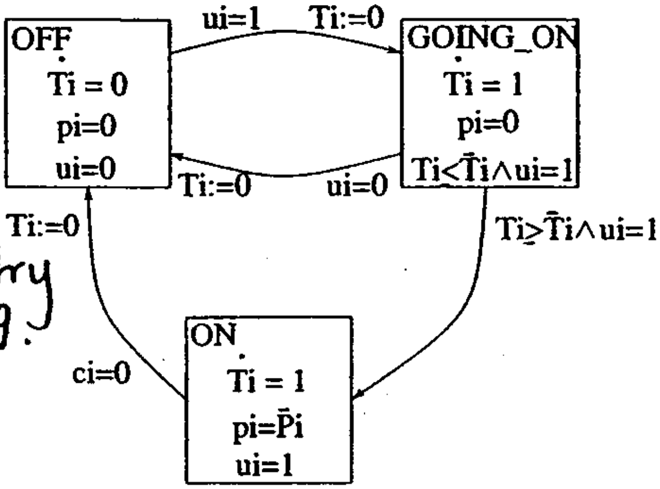


Figure 2: The pump hybrid automaton

9 discrete states:

$$q \in \{(OFF, OFF), (OFF, GOING\_ON), \ldots, (ON, ON)\}$$

2 discrete input variables:

$$(u_1, u_2) \in \{0,1\} \times \{0,1\} = \mathbf{U}$$

and one continuous input variable:

$$d \in [-D_1, D_2] = \mathbf{D}$$

As the notation suggests, u_1 and u_2 will play the role of controls and d will play the role of the disturbance. The additional requirement that $r \in [0, R]$ can be encoded by a state dependent input constraint:

$$\phi(q, x) = \begin{cases} \mathbf{U} \times [0, D_2] & \text{if } r \leq 0 \\ \mathbf{U} \times \mathbf{D} & \text{if } r \in (0, R) \\ \mathbf{U} \times [-D_1, 0] & \text{if } r \geq R \end{cases}$$

Proposition 3 *If* Init $\subseteq \mathbf{Q} \times \mathbb{R} \times [0, R] \times \mathbb{R}^2$, *then for all* $\chi = (\tau, q, x, u_1, u_2, d)$ *and for all* $t \in \tau$, $r(t) \in [0, R]$.

Our goal is to design a controller that keeps the water level in a given range, $[M_1, M_2]$, with $0 \leq M_1 < M_2]$. This requirement can easily be encoded by a safety property $(Q \cup X, \Box F)$ with

$$F = \mathbf{Q} \times [M_1, M_2] \times \mathbb{R}^3$$

We will try to achieve this goal by treating the situation as a game between (u_1, u_2) and d over the cost function J. Recall that this involves solving the equation:

$$J^*(q_0, x_0) = \max_{g} \min_{d} \left(\min_{\chi=(\tau,q,x,(u,d))\in\mathcal{H}_g} J(\chi) \right)$$

Fortunately, for this example, the equation simplifies considerably.

First, notice that the steam boiler system is deterministic, in the sense that for each initial state and each input sequence consistent with ϕ the automaton accepts a unique execution. In this case, we can represent an execution more compactly by $((q_0, x_0), (u_1, u_2), d)$ with the interpretation that (u_1, u_2) and d represent the entire sequence for these variables. Moreover, if the memoryless controller we pick is single valued, this implies we need not worry about the innermost minimization.

Next notice that J can be encoded by means of two real valued cost functions:

$$J_1(x^0, u_1, u_2, d) = \inf_{t\geq 0} w(t) \quad \text{and} \quad J_2(x^0, u_1, u_2, d) = -\sup_{t\geq 0} w(t) \tag{3}$$

Clearly:

$$J = 1 \Leftrightarrow (J_1 \geq M_1) \wedge (J_2 \geq -M_2)$$

The problem of finding a solution to the game over the discrete cost function (known as *qualitative game* or *game of kind*) reduces to finding solutions to two real valued games (known as *quantitative games* or *games of degree*). Even though there is no obvious benefit to doing this, it allows us to use tools from continuous optimal control to address the problem.

Start with the game over J_1. Guess a possible solution:

$$u_i^*(q, x) = 1 \text{ for all } (q, x), \quad \text{and} \quad d^*(q, x) = \begin{cases} D_2 & \text{if } r < R \\ 0 & \text{if } r = R \end{cases} \tag{4}$$

Notice that both players resort to a feedback strategy (a trivial one).

Lemma 1 *(u_1^*, u_2^*, d^*) is globally a saddle solution for the game between (u_1, u_2) and d over J_1.*

Proof: See [4]. ∎

A *saddle solution* is a solution to equation (1) for which the order in which the players make their decisions turns out to be unimportant. In other words, a solution for which for all (q, x):

$$J_1^*(q, x) = \max_{(u_1,u_2)} \min_{d} J_1((q, x), (u_1, u_2), d) = \min_{d} \max_{(u_1,u_2)} J_1((q, x), (u_1, u_2), d)$$

Or, in other words, a solution for which for all $(q, x), u_1, u_2$ and d:

$$J_1((q, x), (u_1, u_2), d^*) \leq J_1((q, x), (u_1^*, u_2^*), d^*) \leq J_1((q, x), (u_1^*, u_2^*), d)$$

The last definition is usually somewhat easier to to work with. It is in fact used to prove the lemma.

The saddle cost:

$$J_1^*(q, x) = J_1((q, x), (u_1^*, u_2^*), d^*)$$

Can be computed in closed form. This allows us then to compute the set of states for which there exists a control that for all actions of the disturbance prevents draining. This set turns out to be of the form:

$$W_1^* = \{(q, x) \ : \ J_1^*(q, x) \geq M_1\} = \{(q, x) \ : \ w \geq \hat{w}(r, T_1, T_2)\}$$

Two level sets of this function are shown in Figure 3.

The expression for J^* also allows us to compute the least restrictive controller that renders the set W_1^* invariant. It turns out to be unique:

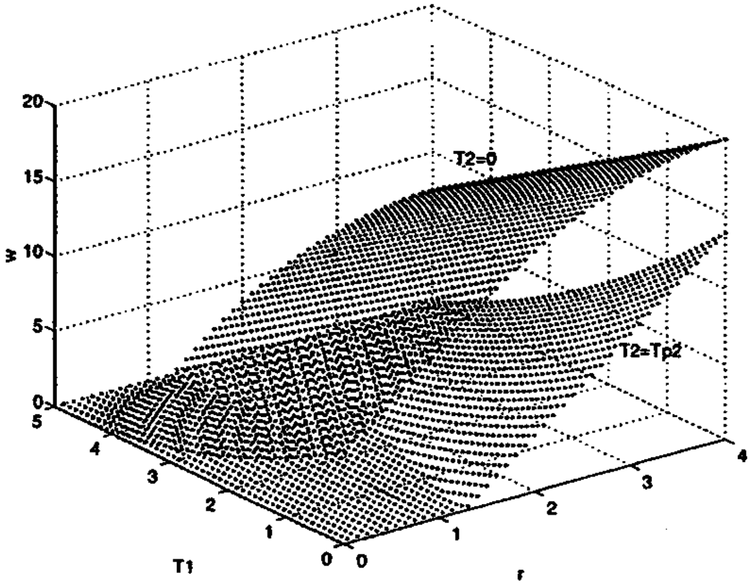


Figure 3: Lower limit on w to avoid draining

Lemma 2 *The feedback controller g_1^1 given by:*

$$\begin{aligned}
&u_1 \in \{0,1\} \textit{ and } u_2 \in \{0,1\} \textit{ if } [w > \hat{w}(r,0,0)] \vee [w < \hat{w}(r,T_1,T_2)]\\
&u_1 = 1 \textit{ and } u_2 \in \{0,1\} \textit{ if } \hat{w}(r,0,0) \geq w > \hat{w}(r,T_1,0)\\
&u_1 \in \{0,1\} \textit{ and } u_2 = 1 \textit{ if } \hat{w}(r,0,0) \geq w > \hat{w}(r,0,T_2)\\
&u_1 = 1 \textit{ and } u_2 = 1 \textit{ if } w = \hat{w}(r,T_1,T_2)
\end{aligned}$$

is the unique, least restrictive, non-blocking, feedback controller that renders W_1^{1*} invariant.

Proof: See [4]. ■

Note that the first term applies to states in the interior of the safe set ($w > \hat{w}(r,0,0)$) as well as all the states outside the safe set ($w < \hat{w}(r,T_1,T_2)$). The expression for $\hat{w}$ (see [4]) suggests that $\hat{w}$ is monotone in T_1 and T_2. Therefore, the condition on the last case is enabled if and only if all other conditions fail. The two middle conditions may overlap, however. Therefore there is some nondeterminism in the choice of safe controls (some states may be safe with either one or the other pump on, but not neither).

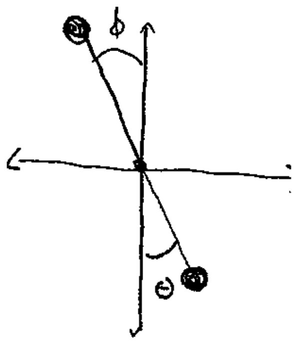
References

[1] J.-R. Abrial, E. Börger, and H. Langmaack, "The steam-boiler case study project, an introduction", in *Formal Methods for Industrial Applications: Specifying and Programming the Steam Boiler Control*, J.-R. Abrial, E. Börger, and H. Langmaack, Eds., number 1165 in LNCS. Springer Verlag, 1996.

[2] J.-R. Abrial, "The steam-boiler control specification problem", in *Formal Methods for Industrial Applications: Specifying and Programming the Steam Boiler Control*, J.-R. Abrial, E. Börger, and H. Langmaack, Eds., number 1165 in LNCS. Springer Verlag, 1996.

[3] Tomas A. Henzinger and Howard Wong-Toi, "Using HYTECH to synthesize control parameters for a steam boiler", in *Formal Methods for Industrial Applications: Specifying and Programming the Steam Boiler Control*, J.-R. Abrial, E. Börger, and H. Langmaack, Eds., number 1165 in LNCS, pp. 265–282. Springer Verlag, 1996.

3b The pendulum dynamics are



$$\ddot{\phi} + f(\dot{\phi}) - K\sin\phi + u(\phi,\dot{\phi}) = 0$$

note: $\phi = \theta - \pi$

The controller is designed with 3 modes

q_1 = Balance: when $\dfrac{\phi^2}{\phi_{max}^2} + \dfrac{\dot{\phi}^2}{\dot{\phi}_{max}^2} \leq 1$

$$u = (c_{11} + K)\phi + c_{12}\dot{\phi}$$

q_2 = Pumping: when not in Balance and

$$S = \tfrac{1}{2}\dot{\theta}^2 - K(1-\cos(\theta)) < 0$$

$$u = -(c + c_3)\dot{\theta}$$

q_3 = Spinning: otherwise

$$u = c_2\dot{\theta}$$

The coefficients given are $c = 0.01$, $K = 10$, $u_{max} = 4$

$c_{11} = 0.4$, $c_{12} = 0.3$ $c_2 = 0.5$ $c_3 = 0.5$

For systems such as the inverted pendulum, it is quite natural to design a hybrid controller to achieve the distinct goals of adding sufficient energy to the system and balancing the pendulum. The resulting control laws are simple and easy to understand intuitively. The issue of chatter between modes arises, however, and must be dealt with in any real implementati

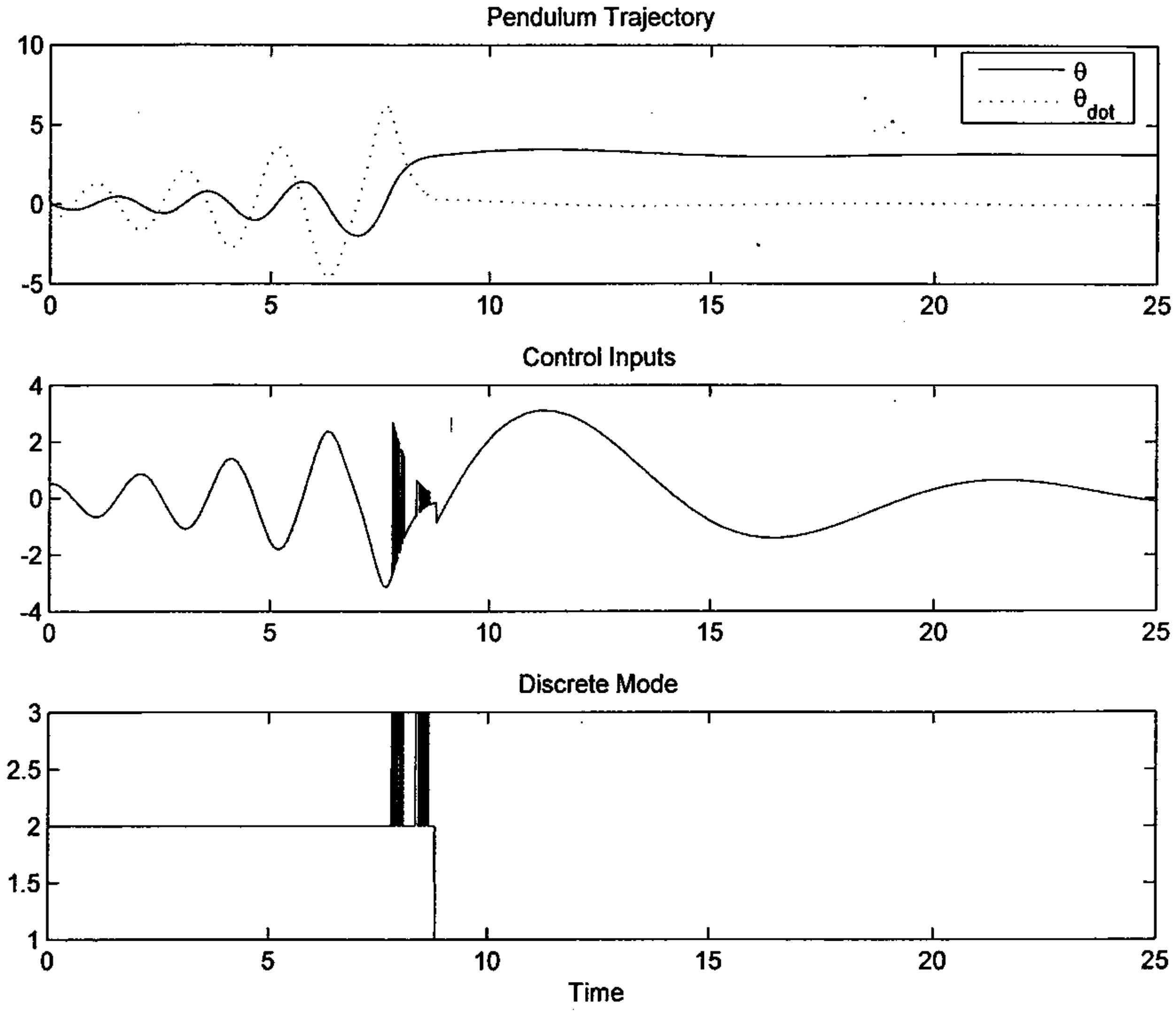
5/5/05 10:07 AM C:\Documents and Settings\Steve\My Documents\School ...\testrun.m 1 of 1

```
clear;

% Simulation Parameters
y0 = [0 -1];
t_max = 25;

% Call solver
[T,Y] = ode45('pendulum',[0 t_max], y0);

% Recalculate control inputs
for i=1:length(T)
  [u(i), q(i)] = controller(Y(i,:));
end

%Plot
figure(1);clf;
subplot(3,1,1);
plot(T,Y(:,1),T,Y(:,2),'r:')
title('Pendulum Trajectory')
legend('\theta', '\theta_{dot}')

subplot(3,1,2);
plot(T,u)
title('Control Inputs')

 ubplot(3,1,3);
plot(T,q)
title('Discrete Mode')
xlabel('Time')
```

5/5/05 10:07 AM C:\Documents and Settings\Steve\My Documents\School...\pendulum.m 1 of 1

```
function [dydt] = plant(T,Y)

c = 0.01;
k = 10;

% System Models, using both coordinate systems
[u, q] = controller(Y);
thetad(1) = Y(2);
thetad(2) = - c*Y(2) - k*sin(Y(1))- u;

% Derivative output for ode23 solver
dydt = thetad';
```

```
function [u_out,q_out] = controller(Y)

% This model has three modes: Pumping, Spinning and Balancing, with the only difference
% between the three being the control input.


%System Constants
c = 0.01;
k = 10;
umax = 4;
c11 = 0.4;
c12 = 0.3;
c2 = 0.5;
c3 = 0.5;

%Input states
theta = Y;
phi(1) = Y(1) - pi;
phi(2) = Y(2);

%Controller selection
phimax = 0.4;
phidmax = 0.3;

balance = phi(1)^2/phimax^2 + phi(2)^2/phidmax^2;
  = 1/2*theta(2)^2-k*(1+cos(theta(1)));

if balance <= 1
    % Balance Control
    u = (c11+k)*sin(phi(1)) + c12*phi(2);
    q = 1;
elseif s < 0
    %Pumping Control
    u = -(c+c3)*theta(2);
    q = 2;
else
    %Spinning Control
    u =  c2*theta(2);
    q = 3;
end

if abs(u) > umax
      u = umax * sign(u);
end

u_out = u;
q_out = q;
```



Pendulum Trajectory

Control Inputs

Discrete Mode

~~this kind of Zeno behavior. The classical way of analyzing such systems is by introducing the notion of sliding modes [8,18].~~

~~4 Regularization~~ SOLUTION TO PROBLEM 4. FROM "ON The Regularization of Zeno Hybrid Automata", Johansson, Egerstedt, Lygeros, Sastry SCL 1999.

Regularization is a standard technique for dealing with differential equations whose solutions are not well defined. We propose a similar approach to extend Zeno executions beyond the Zeno time, primarily for the purpose of simulation. The formal treatment of how to regularize general Zeno hybrid automata is the topic of current research. Here we limit ourselves to specific regularizations of the water tank and bouncing ball automata introduced above. All regularizations are motivated by physical considerations of the underlying systems. For the water tank automaton, it is interesting to notice that different regularizations suggest different extensions of the executions. For the bouncing ball automaton, all extensions considered here are consistent with one another and physical intuition. The regularizations are only presented graphically in this section; see [10] for formal definitions.

Consider a non-blocking and deterministic hybrid automaton H and assume that for every $(q_0, x_0) \in \text{Init}$ the execution $\chi \in \mathcal{H}^{\infty}_{(q_0,x_0)}$ is Zeno. Regularization of H involves 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, $\phi : \mathbf{Q}_\epsilon \times \mathbf{X}_\epsilon \to \mathbf{Q} \times \mathbf{X}$, relating the state of each H_ϵ to the state of H. Given an execution $\chi_\epsilon = (\tau_\epsilon, q_\epsilon, x_\epsilon)$, we use $\phi(\chi_\epsilon)$ as a shorthand notation for the collection (τ, q, x) with $\tau = \tau_\epsilon$, and $\big(q(t), x(t)\big) = \phi\big(q_\epsilon(t), x_\epsilon(t)\big)$ for all $t \in \tau$. Note that in general $\phi(\chi_\epsilon)$ will not be an execution of H. However, the construction of the family H_ϵ should be such that H_ϵ tends to H as ϵ tends to 0, in the sense that if $(q_{\epsilon_0}, x_{\epsilon_0}) \in \text{Init}_\epsilon$, then $\phi(q_{\epsilon_0}, x_{\epsilon_0}) \in \text{Init}$, and if χ_ϵ is the execution of H_ϵ with initial condition $(q_{\epsilon_0}, x_{\epsilon_0})$, then $\phi(\chi_\epsilon)$ converges to $\chi \in \mathcal{H}^{\infty}_{\phi(q_{\epsilon_0}, x_{\epsilon_0})}$ over all compact subintervals of $[\tau_0, \tau_\infty)$, where the convergence is taken in the Skorohod metric [4].[3]

[3] Formally, we need to eliminate all "inert" transitions from τ, that is, replace all $[\tau_i, \tau_i'][\tau_{i+1}, \tau_{i+1}']$ for which $\phi\big(q_\epsilon(\tau_i'), x_\epsilon(\tau_i')\big) = \phi\big(q_\epsilon(\tau_{i+1}), x_\epsilon(\tau_{i+1})\big)$ by a single interval $[\tau_i, \tau_{i+1}']$.

Water Tank Automaton

We first study temporal and spatial regularizations of the water tank automaton. Throughout, we assume that $\max\{v_1, v_2\} < w < v_1 + v_2$, so that WT is Zeno.

Physically, temporal regularization represents a situation where there is a delay, $\epsilon > 0$, between the time the inflow is commanded to switch from one tank to the other and the time the switch actually takes place. The temporal regularization of the water tank automaton, WT_ϵ^{T}, is shown in Figure 3. It is easy to show that WT_ϵ^{T} accepts a unique non-Zeno execution for each initial state. Overloading the notation somewhat, we can express the relation between the states of WT_ϵ^{T} and the states of WT through the map $\phi\big(q_i, (x_1, x_2, x_3)\big) = \phi\big(q_i', (x_1, x_2, x_3)\big) = \big(q_i, (x_1, x_2)\big)$, for $i = 1, 2$. If we set $r_1 = r_2 = 1$, $v_1 = 2$, $v_2 = 3$, and $w = 4$, and assume that initially $x_1(0) = x_2(0) = 2$ and $q(0) = q_1$, then $\tau_\infty = 2$. Figure 4 shows simulation results for WT_ϵ^{T}; x_1 and x_2 are plotted as functions of time for two values of ϵ, 0.1 and 0.01. Note that as ϵ decreases, the execution of WT_ϵ^{T} converges over the interval $(\tau_0, \tau_\infty) = (0, 2)$ to the execution of WT, in the sense discussed above. For $t > \tau_\infty$, the continuous part of the execution of WT_ϵ^{T} tends to $\big(x_1(t), x_2(t)\big) = \big(1, 1 - (t - \tau_\infty)\big)$.

The spatial regularization of the water tank automaton corresponds to a situation where the measurement of x_1 and x_2 is based on floats, which have to move a certain distance ϵ to register a change. It can be implemented by introducing a minimum deviation in the continuous state variables between the discrete transitions. The regularized automaton, WT_ϵ^{S}, is presented in Figure 5. Again one can show that WT_ϵ^{S} accepts a unique non-Zeno execution for each initial state. We can relate the state of WT_ϵ^{S} to the state of WT through $\phi\big(q_i, (x_1, x_2, x_3, x_4)\big) = \big(q_i, (x_1, x_2)\big)$, for $i = 1, 2$. Figure 6 shows simulation results for WT_ϵ^{S} with $\epsilon = 0.1$ and 0.01 and the parameters given above. As for the temporal regularization, the execution of WT_ϵ^{S} converges to the execution of WT over the interval (τ_0, τ_∞). For $t > \tau_\infty$, however, the execution converges to $x_1(t) = x_2(t) = -(t - \tau_\infty)/2 + 1$, which is different from the limit in the case of temporal regularization.

Bouncing Ball Automaton

Next, we consider temporal and dynamic regularizations of the bouncing ball automaton. Throughout we assume $c > 1$ so that BB is Zeno.

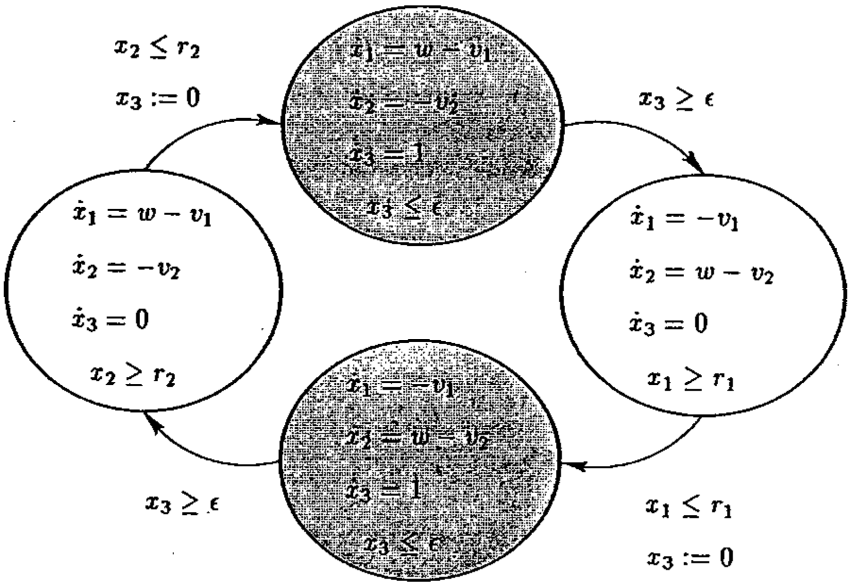


Fig. 3. Temporal regularization of the water tank automaton.

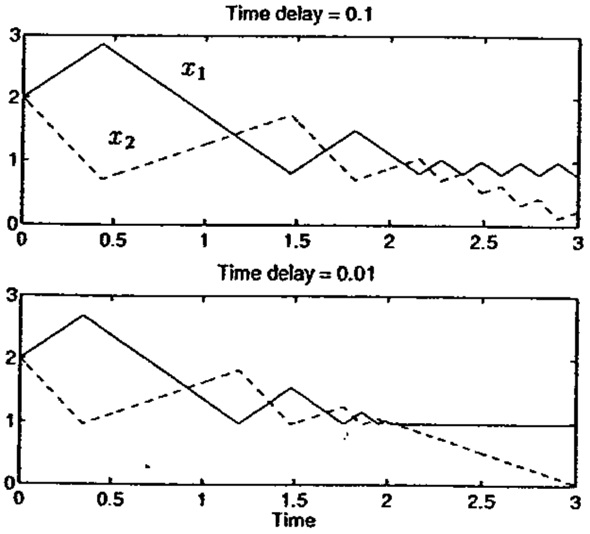


Fig. 4. Simulation of the temporally regularized water tank automaton.

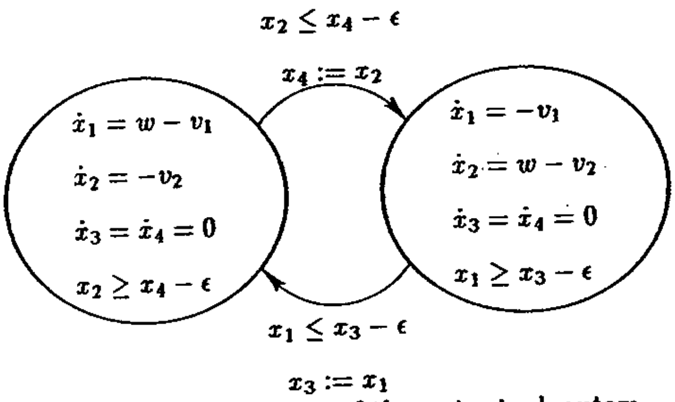

$x_2 \leq x_4 - \epsilon$
$x_4 := x_2$
$\dot{x}_1 = w - v_1$
$\dot{x}_2 = -v_2$
$\dot{x}_3 = \dot{x}_4 = 0$
$x_2 \geq x_4 - \epsilon$
$\dot{x}_1 = -v_1$
$\dot{x}_2 = w - v_2$
$\dot{x}_3 = \dot{x}_4 = 0$
$x_1 \geq x_3 - \epsilon$
$x_1 \leq x_3 - \epsilon$
$x_3 := x_1$

Fig. 5. Spatial regularization of the water tank autom

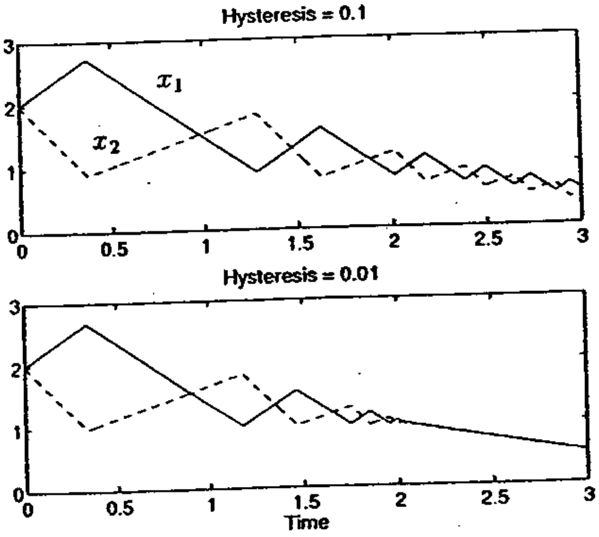


Fig. 6. Simulation of the spatially regularized water tank automat