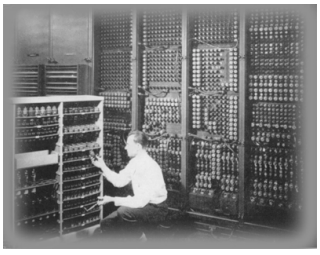
Hybrid Model Predictive Control

Alberto Bemporad

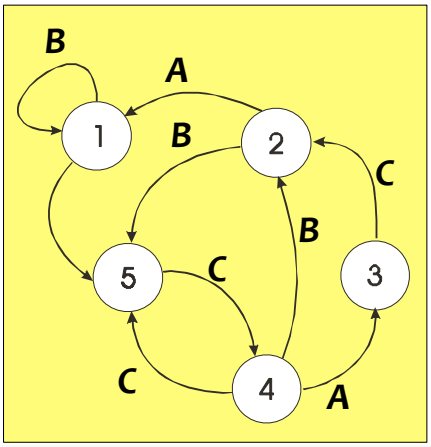
http://www.imtlucca.it/alberto.bemporad

IMT Institute for Advanced Studies Lucca

Hybrid systems



$\mathtt{x} \in \{1, 2, 3, 4, 5\}$
$\mathtt{u} \in \{A, B, C\}$



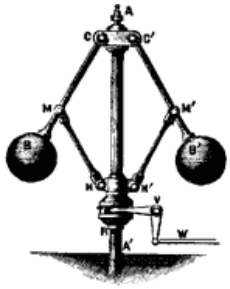
Computer Science

Control Theory

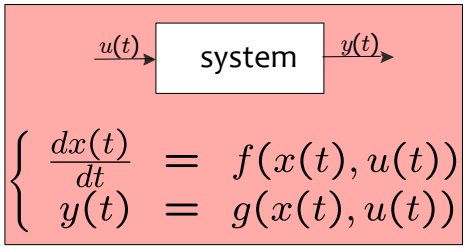
Finite state machines

Continuous dynamical systems

Hybrid systems



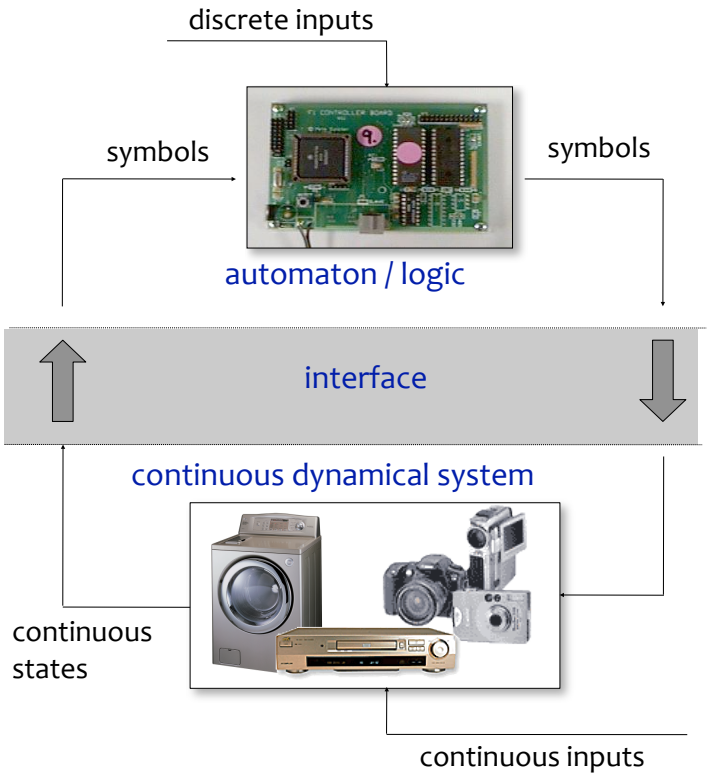
$x \in \mathbb{R}^n$
$u \in \mathbb{R}^m$
$y \in \mathbb{R}^p$



$$\begin{cases} \frac{dx(t)}{dt} = f(x(t), u(t)) \\ y(t) = g(x(t), u(t)) \end{cases}$$

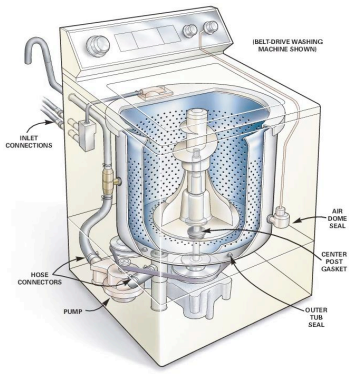
$$\begin{cases} x(k+1) = f(x(k), u(k)) \\ y(k) = g(x(k), u(k)) \end{cases}$$

Embedded systems



- Automobiles
- Industrial processes
- Consumer electronics
- Home appliances
- …

Example:



Example of hybrid control problem

Cruise control problem

GOAL:

command gear ratio, gas pedal, and brakes to **track** a desired speed and minimize consumptions

CHALLENGES:

- continuous **and** discrete inputs
- dynamics depends on gear
- nonlinear torque/speed maps

Key requirements for hybrid models

- **Descriptive** enough to capture the behavior of the system
  - continuous dynamics (physical laws)
  - logic components (switches, automata, software code)
  - interconnection between logic and dynamics

- **Simple** enough for solving *analysis* and *synthesis* problems

$$\begin{cases} x' = Ax + Bu \\ y = Cx + Du \end{cases}$$

linear systems

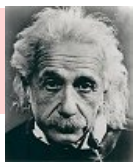
?

$$\begin{cases} x' = f(x,u,t) \\ y = g(x,u,t) \end{cases}$$

nonlinear systems

linear hybrid systems

"Make everything as simple as possible, but not simpler."
— Albert Einstein

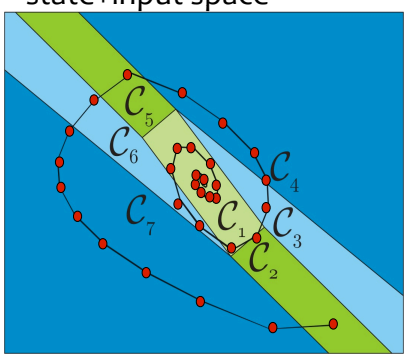


Piecewise affine systems

$$\begin{aligned} x(k+1) &= A_{i(k)}x(k) + B_{i(k)}u(k) + f_{i(k)} \\ y(k) &= C_{i(k)}x(k) + D_{i(k)}u(k) + g_{i(k)} \\ i(k) \quad &\text{s.t.} \quad H_{i(k)}x(k) + J_{i(k)}u(k) \le K_{i(k)} \end{aligned}$$

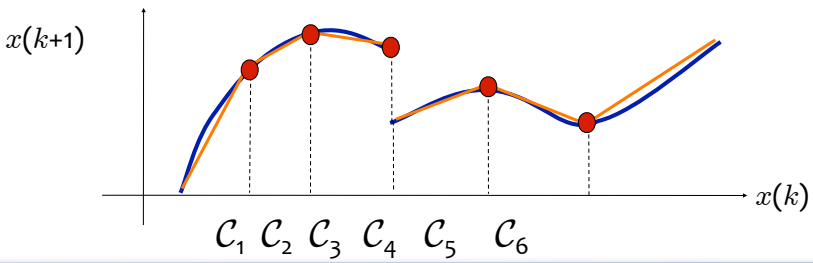
$x \in \mathbb{R}^n,\ u \in \mathbb{R}^m,\ y \in \mathbb{R}^p$
$i(k) \in \{1, \dots, s\}$

state+input space



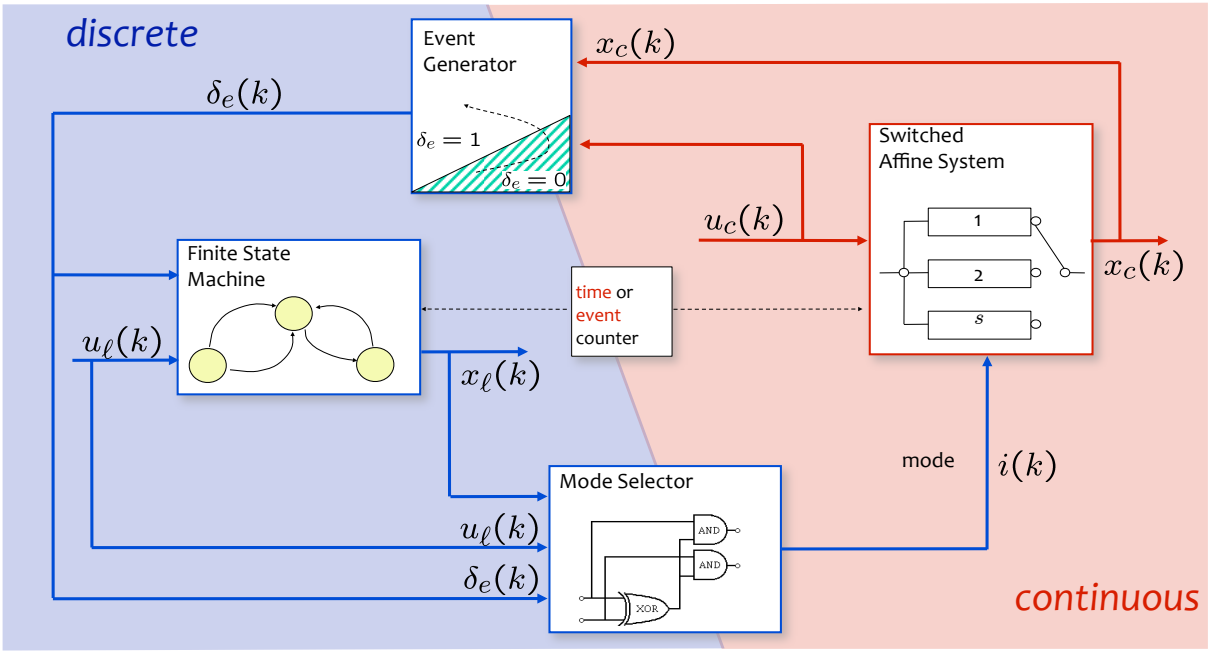
(Sontag 1981)

Can approximate nonlinear and/or discontinuous dynamics arbitrarily well



Discrete Hybrid Automaton (DHA)

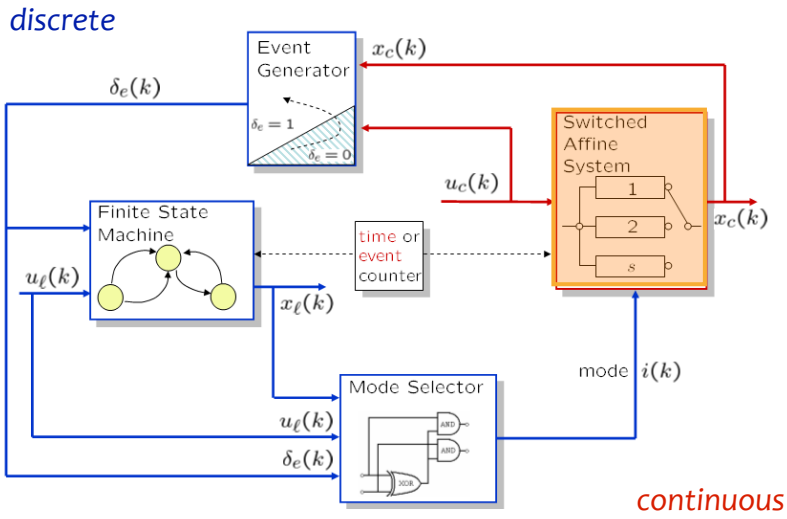
(Torrisi, Bemporad, 2004)



$x_\ell \in \{0,1\}^{n_b}$ = binary states
$u_\ell \in \{0,1\}^{m_b}$ = binary inputs
$\delta_e \in \{0,1\}^{n_e}$ = event variables

$x_c \in \mathbb{R}^{n_c}$ = continuous states
$u_c \in \mathbb{R}^{m_c}$ = continuous inputs
$i \in \{1,2,\dots,s\}$ = current mode

Switched affine system

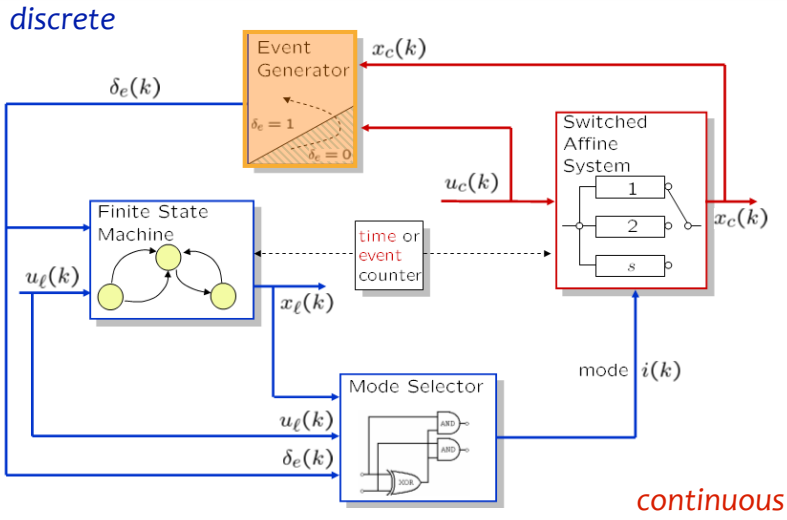


The affine dynamics depend on the current mode $i(k)$:

$$x_c(k+1) = A_{i(k)}x_c(k) + B_{i(k)}u_c(k) + f_{i(k)}$$

$x_c \in \mathbb{R}^{n_c}$, $u_c \in \mathbb{R}^{m_c}$

Event generator



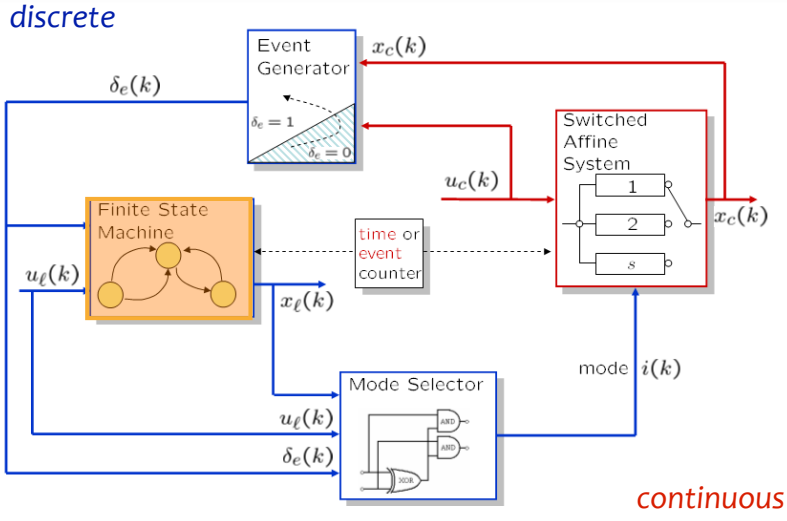
Event variables are generated by linear threshold conditions over continuous states, continuous inputs, and time:

$$[\delta_e^i(k) = 1] \iff [H^i x_c(k) + K^i u_c(k) \leq W^i]$$

$$x_c \in \mathbb{R}^{n_c},\ u_c \in \mathbb{R}^{m_c},\ \delta_e \in \{0,1\}^{n_e}$$

Example: $[\delta(k)=1] \leftrightarrow [x_c(k) \geq 0]$

Finite state machine

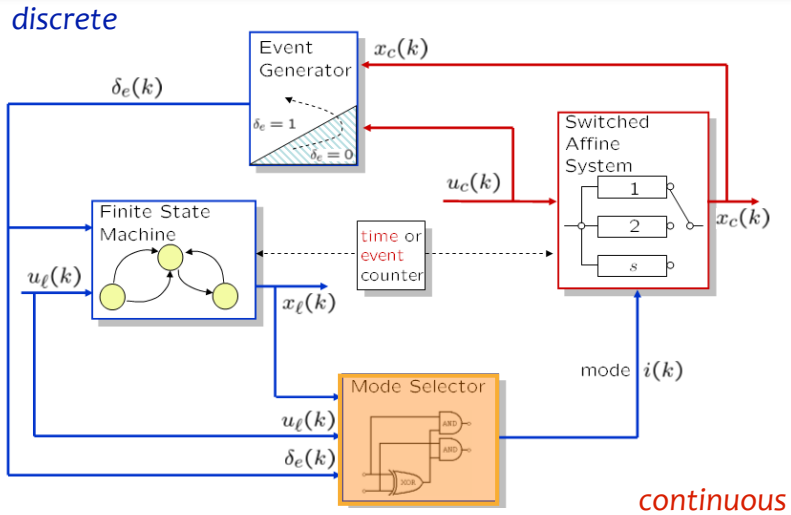


The binary state of the finite state machine evolves according to a Boolean state update function:

$$x_\ell(k+1) = f_B(x_\ell(k), u_\ell(k), \delta_e(k)) \qquad x_\ell \in \{0,1\}^{n_\ell},\ u_\ell \in \{0,1\}^{m_\ell},\ \delta_e \in \{0,1\}^{n_e}$$

Example: $x_\ell(k+1) = \neg\delta_e(k) \vee (x_\ell(k) \wedge u_\ell(k))$

Mode selector



The mode selector can be seen as the output function of the discrete dynamics

The active mode $i(k)$ is selected by a Boolean function of the current binary states, binary inputs, and event variables:

$$i(k) = f_{\mathrm{M}}(x_\ell(k), u_\ell(k), \delta_e(k)) \qquad x_\ell \in \{0,1\}^{n_\ell},\ u_\ell \in \{0,1\}^{m_\ell},\ \delta_e\{0,1\}^{n_e}$$

Example:

$$i(k) = \begin{bmatrix} \neg u_\ell(k) \vee x_\ell(k) \\ u_\ell(k) \wedge x_\ell(k) \end{bmatrix}$$

u_ℓ / x_ℓ	0	1
0	$i = \left[\begin{smallmatrix}1\\0\end{smallmatrix}\right]$	$i = \left[\begin{smallmatrix}1\\0\end{smallmatrix}\right]$
1	$i = \left[\begin{smallmatrix}0\\0\end{smallmatrix}\right]$	$i = \left[\begin{smallmatrix}1\\1\end{smallmatrix}\right]$

the system has 3 modes

Logic $\rightarrow$ inequalities

$$X_1 \vee X_2 = \text{TRUE} \quad \longrightarrow \quad \delta_1 + \delta_2 \geq 1, \qquad \delta_1, \delta_2 \in \{0,1\}$$

(Glover 1975, Williams 1977, Hooker 2000)

0. Given a Boolean statement

$$F(X_1, X_2, \ldots, X_n) = \text{TRUE}$$

1. Convert to Conjunctive Normal Form (CNF):

$$\bigwedge_{j=1}^{m} \left(\bigvee_{i \in P_j} X_i \bigvee_{i \in N_j} \bar{X}_i \right) = \text{TRUE}$$

2. Transform into inequalities:

$$\begin{aligned} \sum_{i \in P_1} \delta_i + \sum_{i \in N_1} (1 - \delta_i) &\geq 1 \\ \vdots \quad \vdots &\quad \vdots \\ \sum_{i \in P_m} \delta_i + \sum_{i \in N_m} (1 - \delta_i) &\geq 1 \end{aligned}$$

polyhedron

$$A\delta \leq b, \quad \delta \in \{0,1\}^n$$

Any logic proposition can be translated into linear integer inequalities

Logic → inequalities: symbolic approach

Example: $F(X_1, X_2, X_3) = [X_3 \leftrightarrow X_1 \wedge X_2]$

1. Convert to Conjunctive Normal Form (CNF):
(see e.g.: `http://www.oursland.net/aima/propositionApplet.html`
or just search [CNF + applet] on Google ...)

$$(X_3 \vee \neg X_1 \vee \neg X_2) \wedge (X_1 \vee \neg X_3) \wedge (X_2 \vee \neg X_3)$$

2. Transform into inequalities:

$$\begin{cases} \delta_3 + (1-\delta_1) + (1-\delta_2) \geq 1 \\ \delta_1 + (1-\delta_3) \geq 1 \\ \delta_2 + (1-\delta_3) \geq 1 \end{cases}$$

Big-M technique (iff)

- Consider the *if-and-only-if* condition

$$[\delta = 1] \leftrightarrow [a'x_c - b \leq 0]$$

$x_c \in \mathcal{X} \subset \mathbb{R}^{n_c}$
$\delta \in \{0, 1\}$

- Assume $\mathcal{X}$ bounded and let M and m such that

$$M > a'x_c - b, \ \forall x_c \in \mathcal{X}$$
$$m < a'x_c - b, \ \forall x_c \in \mathcal{X}$$

- The *if-and-only-if* condition is equivalent to

$$\begin{cases} a'x_c - b \ \leq \ M(1-\delta) \\ a'x_c - b \ > \ m\delta \end{cases}$$

Big-M technique (if-then-else)

- Consider the *if-then-else* condition

$$z = \begin{cases} a_1' x_c + f_1 & \text{if } \delta = 1 \\ a_2' x_c + f_2 & \text{otherwise} \end{cases}$$

$$x_c \in \mathcal{X} \subset \mathbb{R}^{n_c}$$
$$\delta \in \{0,1\}$$
$$z \in \mathbb{R}$$

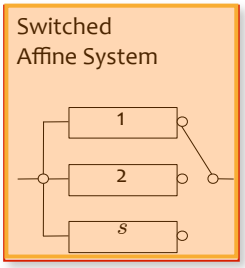
- Assume $\mathcal{X}$ bounded and let M_1, M_2 and m_1, m_2 such that

$$M_1 > a_1' x_c + f_1 > m_1, \ \forall x_c \in \mathcal{X}$$
$$M_2 > a_2' x_c + f_2 > m_2, \ \forall x_c \in \mathcal{X}$$

- The *if-then-else* condition is equivalent to

$$\begin{cases} (m_1 - M_2)(1-\delta) + z \leq a_1 x_c + f_1 \\ (m_2 - M_1)(1-\delta) - z \leq -a_1 x_c - f_1 \\ (m_2 - M_1)\delta + z \leq a_2 x_c + f_2 \\ (m_1 - M_2)\delta - z \leq -a_2 x_c - f_2 \end{cases}$$

Switched affine system



The state-update equation can be rewritten as a difference equation + *if-then-else* conditions:

$$z_1(k) = \begin{cases} A_1 x_c(k) + B_1 u_c(k) + f_1, & \text{if } i(k) = 1 \\ 0, & \text{otherwise,} \end{cases}$$

$$\vdots$$

$$z_s(k) = \begin{cases} A_s x_c(k) + B_s u_c(k) + f_s, & \text{if } i(k) = s, \\ 0, & \text{otherwise,} \end{cases}$$

$$x_c(k+1) = \sum_{i=1}^{s} z_i(k)$$

where $z_i(k) \in \mathbb{R}^{n_c}, i = 1, \ldots, s$

Output equations $y_c(k)=C_i x_c(k)+D_i u_c(k)+g_i$ admit a similar transformation.

Logic and inequalities

$X_1 \vee X_2 = \text{TRUE} \longrightarrow \delta_1 + \delta_2 \geq 1, \quad \delta_1, \delta_2 \in \{0,1\}$ (Glover 1975, Williams 1977, Hooker 2000)

Any logic statement
$f(X) = \text{TRUE}$

$$\bigwedge_{j=1}^{m} \left(\vee_{i \in P_j} X_i \vee_{i \in N_j} \neg X_i \right) \quad \text{(CNF)}$$
$$N_j, P_j \subseteq \{1, \dots, n\}$$

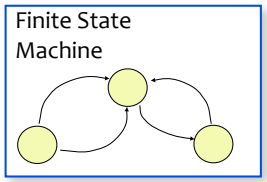
$$\begin{cases} 1 \leq \sum_{i \in P_1} \delta_i + \sum_{i \in N_1} (1 - \delta_i) \\ \vdots \\ 1 \leq \sum_{i \in P_m} \delta_i + \sum_{i \in N_m} (1 - \delta_i) \end{cases}$$

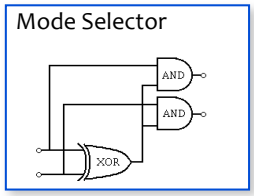
$$[\delta_e^i(k) = 1] \leftrightarrow [H^i x_c(k) \leq W^i]$$

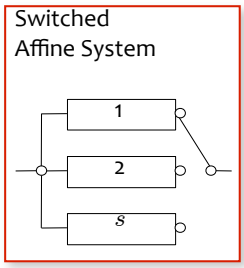
$$\begin{cases} H^i x_c(k) - W^i \leq M^i (1 - \delta_e^i) \\ H^i x_c(k) - W^i > m^i \delta_e^i \end{cases}$$

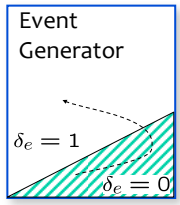
IF $[\delta = 1]$ THEN $z = a_1^T x + b_1^T u + f_1$
ELSE $z = a_2^T x + b_2^T u + f_2$

$$\begin{cases} (m_1 - M_2)(1 - \delta) + z \leq a_1 x + b_1 u + f_1 \\ (m_2 - M_1)(1 - \delta) - z \leq -a_1 x - b_1 u - f_1 \\ (m_2 - M_1)\delta + z \leq a_2 x + b_2 u + f_2 \\ (m_1 - M_2)\delta - z \leq -a_2 x - b_2 u - f_2 \end{cases}$$



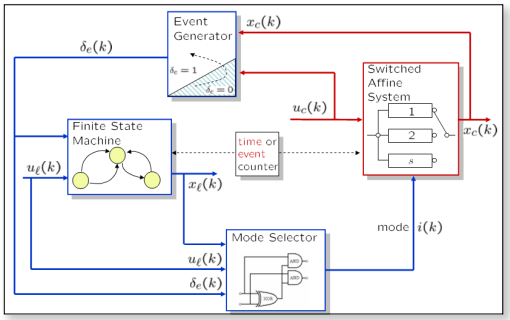






Mixed Logical Dynamical (MLD) systems

Discrete Hybrid Automaton



HYSDEL
(Torrisi, Bemporad, 2004)

Mixed Logical Dynamical (MLD) systems

$$\begin{aligned} x(k+1) &= Ax(k) + B_1 u(k) + B_2 \delta(k) + B_3 z(k) + B_5 \\ y(k) &= Cx(k) + D_1 u(k) + D_2 \delta(k) + D_3 z(k) + D_5 \\ E_2 \delta(k) + E_3 z(k) &\leq E_4 x(k) + E_1 u(k) + E_5 \end{aligned}$$

(Bemporad, Morari 1999)

Continuous and binary variables

$x \in \mathbb{R}^{n_r} \times \{0,1\}^{n_b}, \; u \in \mathbb{R}^{m_r} \times \{0,1\}^{m_b}$
$y \in \mathbb{R}^{p_r} \times \{0,1\}^{p_b}, \; \delta \in \{0,1\}^{r_b}, \; z \in \mathbb{R}^{r_r}$

- Computationally oriented (mixed-integer programming)
- Suitable for controller synthesis, verification, ...

A simple example

- System:

$$x(k+1) = \begin{cases} 0.8x(k) + u(k) & \text{if } x(k) \geq 0 \\ -0.8x(k) + u(k) & \text{if } x(k) < 0 \end{cases}$$

$$-10 \leq x(k) \leq 10$$

- Associate $[\delta(k) = 1] \leftrightarrow [x(k) \geq 0]$ and transform

$$\begin{aligned} x(k) &\geq m(1-\delta(k)) \\ x(k) &\leq -\epsilon + (M+\epsilon)\delta(k) \end{aligned} \qquad \begin{aligned} M &= -m = 10 \\ \epsilon &> 0 \text{ "small"} \end{aligned}$$

- Then $x(k+1) = 1.6\delta(k)x(k) - 0.8x(k) + u(k)$

$z(k) = \delta(k)x(k)$ →

$$\begin{aligned} z(k) &\leq M\delta(k) \\ z(k) &\geq m\delta(k) \\ z(k) &\leq x(k) - m(1-\delta(k)) \\ z(k) &\geq x(k) - M(1-\delta(k)) \end{aligned} \qquad \delta(k) \in \{0,1\}$$

- Rewrite as a linear equation

$$x(k+1) = 1.6z(k) - 0.8x(k) + u(k)$$

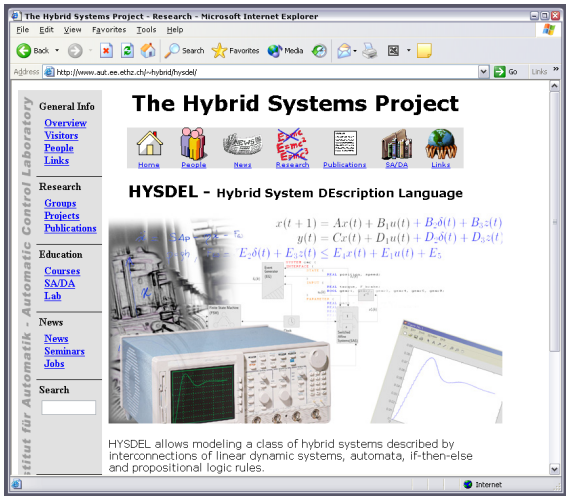
(Note: nonlinear system $x(k+1) = 0.8|x(k)| + u(k)$)

HYSDEL

(HYbrid Systems DEscription Language)

- Describe *hybrid systems:*

– Automata
– Logic
– Lin. Dynamics
– Interfaces
– Constraints



(Torrisi, Bemporad, 2004)

- Automatically generate MLD models in Matlab

Download: http://www.ing.unitn.it/~bemporad/hybrid/toolbox

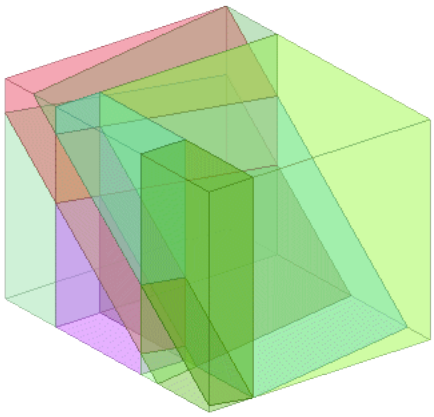
Reference: http://control.ethz.ch/~hybrid/hysdel

Hybrid Toolbox for MATLAB

Features:

(Bemporad, 2003-2010)

- Hybrid models: design, simulation, verification
- MPC design for linear systems w/ constraints and hybrid systems (on-line optimization via QP/MILP/MIQP)
- Explicit MPC (via multi-parametric programming)
- Automatic C-code generation
- Simulink library

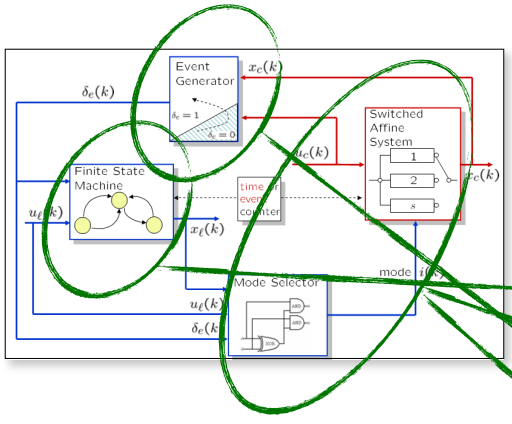


Supported by

3000+ download requests since October 2004

http://www.ing.unitn.it/~bemporad/hybrid/toolbox

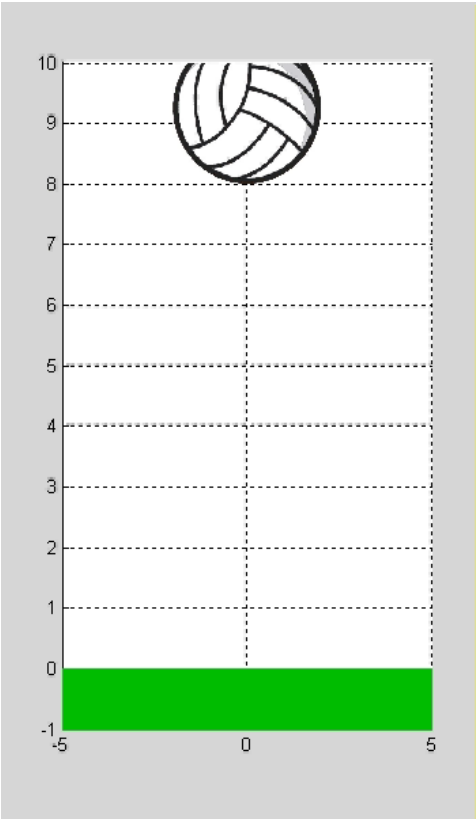
DHA and HYSDEL models




```
SYSTEM name {
   INTERFACE {
      STATE {
         REAL xc [xmin,xmax];
         BOOL xl; }
      INPUT {
         REAL uc [umin,umax];
         BOOL ul; }
      PARAMETER {
         REAL param1 = 1;}
   } /* end of interface */

   IMPLEMENTATION {
      AUX { BOOL d;
            REAL z; }

      AUTOMATA { xl = xl & ~ul; }

      DA { z = { IF d THEN 2*xc ELSE -xc };  }

      AD { d = xc - 1 <= 0; }

      CONTINUOUS {
         xc = z; }

      MUST {
         xc + uc <= 2;
         ~(xl & ul); }
   } /* end implementation */
} /* end system */
```

Additional relations constraining system's variables

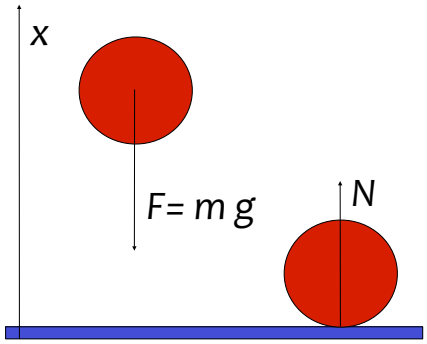
Bouncing ball example



$$\ddot{x} = -g$$

$$x \leq 0 \Rightarrow \dot{x}(t^+) = -(1-\alpha)\dot{x}(t^-)$$

$\alpha \in [0, 1]$

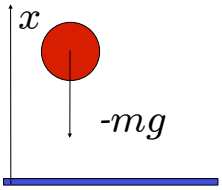


How to model the bouncing ball as a discrete-time hybrid system ?

Bouncing ball – Time discretization

$x(k) > 0$:

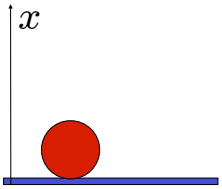
$$v(k) \approx \frac{x(k) - x(k-1)}{T_s} \qquad -g = \ddot{x}(k) \approx \frac{v(k) - v(k-1)}{T_s}$$



$$\Rightarrow \quad \begin{cases} v(k+1) &= v(k) - T_s g \\ x(k+1) &= x(k) + T_s v(k+1) \\ &= x(k) + T_s v(k) - T_s^2 g \end{cases}$$

$x(k) \leq 0$:

$$v(k) = -(1-\alpha)v(k-1) \qquad \begin{aligned} x(k+1) &= x(k-1) \\ &= x(k) - T_s v(k) \end{aligned}$$



$$\Rightarrow \quad \begin{cases} v(k+1) &= -(1-\alpha)v(k) \\ x(k+1) &= x(k) - T_s v(k) \end{cases}$$

go to demo **/demos/hybrid/bball.m**

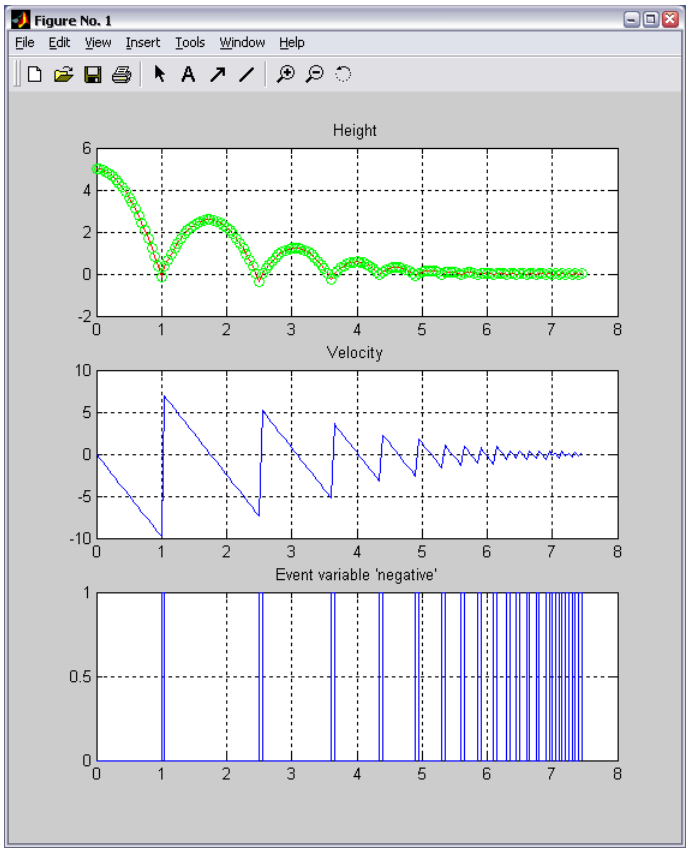
Bouncing ball - HYSDEL model

```
SYSTEM bouncing_ball {
INTERFACE {
/* Description of variables and constants */
        STATE { REAL height [-10,10];
                REAL velocity [-100,100];    }


        PARAMETER {
                REAL g;
                REAL alpha;   /* 0=elastic, 1=completely anelastic */
                REAL Ts; }
}
IMPLEMENTATION {
        AUX {   REAL hnext;
                REAL vnext;
                BOOL negative;    }


        AD {    negative = height <= 0; }


        DA {    hnext = {  IF negative THEN height-Ts*velocity
                           ELSE height+Ts*velocity-Ts*Ts*g};
                vnext = {  IF negative THEN -(1-alpha)*velocity
                           ELSE velocity-Ts*g};  }
        CONTINUOUS {
                height   = hnext;
                velocity = vnext;}
}}
```

go to demo **/demos/hybrid/bball.m**

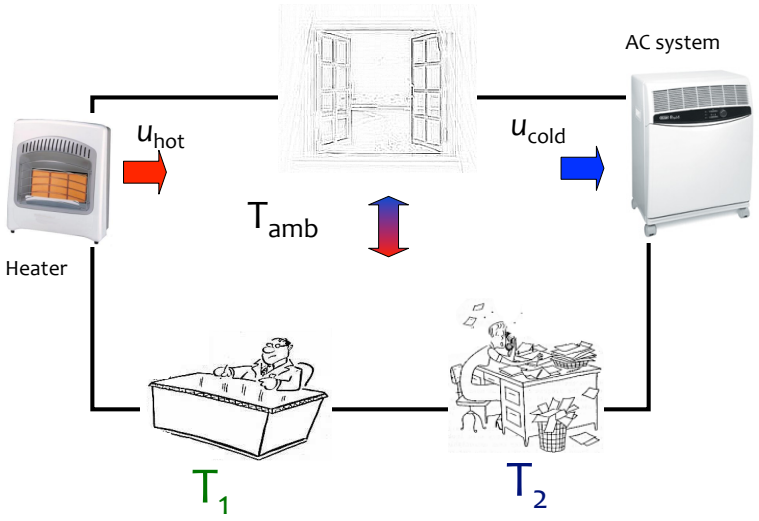
Bouncing ball - Simulation

```
>>Ts=0.05;
>>g=9.8;
>>alpha=0.3;

>>S=mld('bouncing_ball',Ts);

>>N=150;
>>U=zeros(N,0);
>>x0=[5 0]';

>>[X,T,D]=sim(S,x0,U);
```



Note: no Zeno effects in discrete time !

Example: Room temperature



Hybrid dynamics

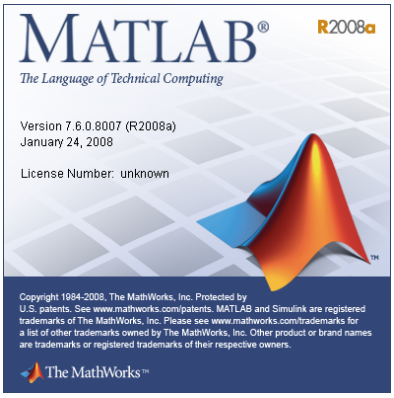
- #1 turns the heater (A/C) on whenever he is cold (hot)
- If #2 is cold he turns the heater on, unless #1 is hot
- If #2 is hot he turns A/C on, unless #2 is cold
- Otherwise, heater and A/C are off

- $\dot{T}_1 = -\alpha_1(T_1 - T_{amb}) + k_1(u_{hot} - u_{cold})$ (body temperature dynamics of #1)
- $\dot{T}_2 = -\alpha_2(T_2 - T_{amb}) + k_2(u_{hot} - u_{cold})$ (body temperature dynamics of #2)

go to demo **/demos/hybrid/heatcool.m**

HYSDEL Model

```
SYSTEM heatcool {

INTERFACE {
    STATE { REAL T1 [-10,50];
            REAL T2 [-10,50];
          }
    INPUT { REAL Tamb [-10,50];
          }
    PARAMETER {
        REAL Ts, alpha1, alpha2, k1, k2;
        REAL Thot1, Tcold1, Thot2, Tcold2, Uc, Uh;
        }
}
IMPLEMENTATION {
        AUX { REAL uhot, ucold;
              BOOL hot1, hot2, cold1, cold2;
            }
        AD  { hot1 = T1>=Thot1;
              hot2 = T2>=Thot2;
              cold1 = T1<=Tcold1;
              cold2 = T2<=Tcold2;
            }
        DA  { uhot = {IF cold1 | (cold2 & ~hot1) THEN Uh ELSE 0};
              ucold = {IF hot1 | (hot2 & ~cold1) THEN Uc ELSE 0};
            }
        CONTINUOUS { T1 = T1+Ts*(-alpha1*(T1-Tamb)+k1*(uhot-ucold));
                     T2 = T2+Ts*(-alpha2*(T2-Tamb)+k2*(uhot-ucold));
            }
    }
}
```



Hybrid Toolbox for Matlab

http://www.dii.unisi.it/hybrid/toolbox

```
>>S=mld('heatcoolmodel',Ts)
```

get the MLD model in Matlab

```
>>[XX,TT]=sim(S,x0,U);
```

simulate the MLD model

Hybrid MLD Model

- MLD model

$$\begin{aligned} x(k+1) &= Ax(k) + B_1u(k) + B_2\delta(k) + B_3z(k) \\ y(k) &= Cx(k) + D_1u(k) + D_2\delta(k) + D_3z(k) \\ E_2\delta(k) + E_3z(k) &\leq E_1u(k) + E_4x(k) + E_5 \end{aligned}$$

- 2 continuous states: (temperatures T_1,T_2)
- 1 continuous input: (room temperature T_{amb})
- 2 auxiliary continuous vars: (power flows u_{hot}, u_{cold})
- 6 auxiliary binary vars: (4 thresholds + 2 for OR condition)
- 20 mixed-integer inequalities

Possible combination of integer variables: $2^6 = 64$

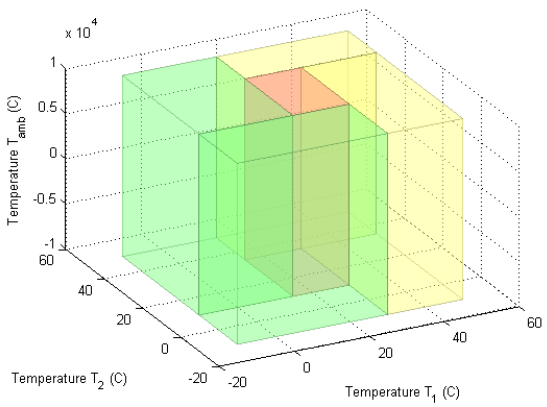
Hybrid PWA Model

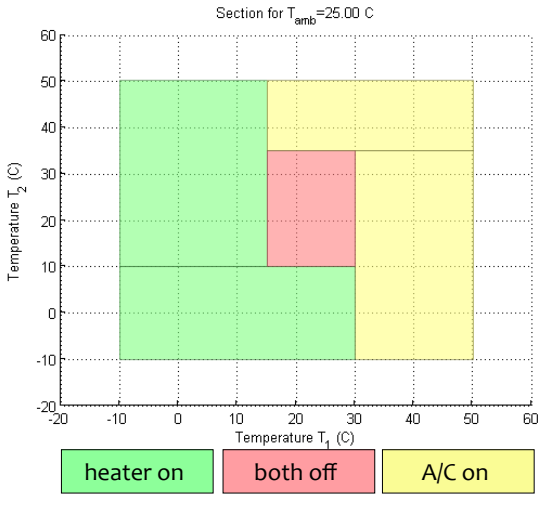
- PWA model

```
>>P=pwa(S);
```

$$\begin{aligned} x(k+1) &= A_{i(k)}x(k) + B_{i(k)}u(k) + f_{i(k)} \\ y(k) &= C_{i(k)}x(k) + D_{i(k)}u(k) + g_{i(k)} \\ i(k) \quad &\text{s.t.} \quad H_{i(k)}x(k) + J_{i(k)}u(k) \leq K_{i(k)} \end{aligned}$$

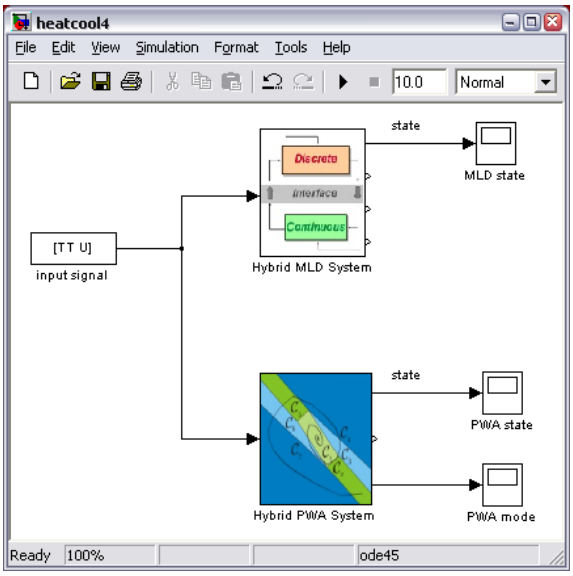
- 2 continuous states: (temperatures T_1,T_2)
- 1 continuous input: (room temperature T_{amb})

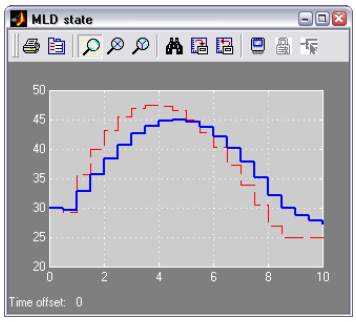


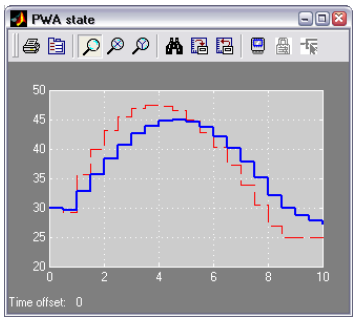


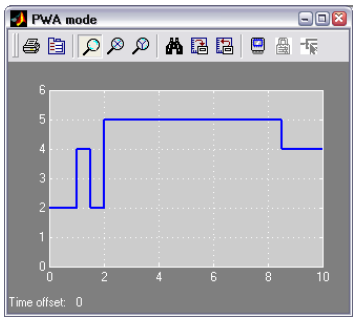
- 5 polyhedral regions
  (partition does not depend on input)

Simulation in Simulink





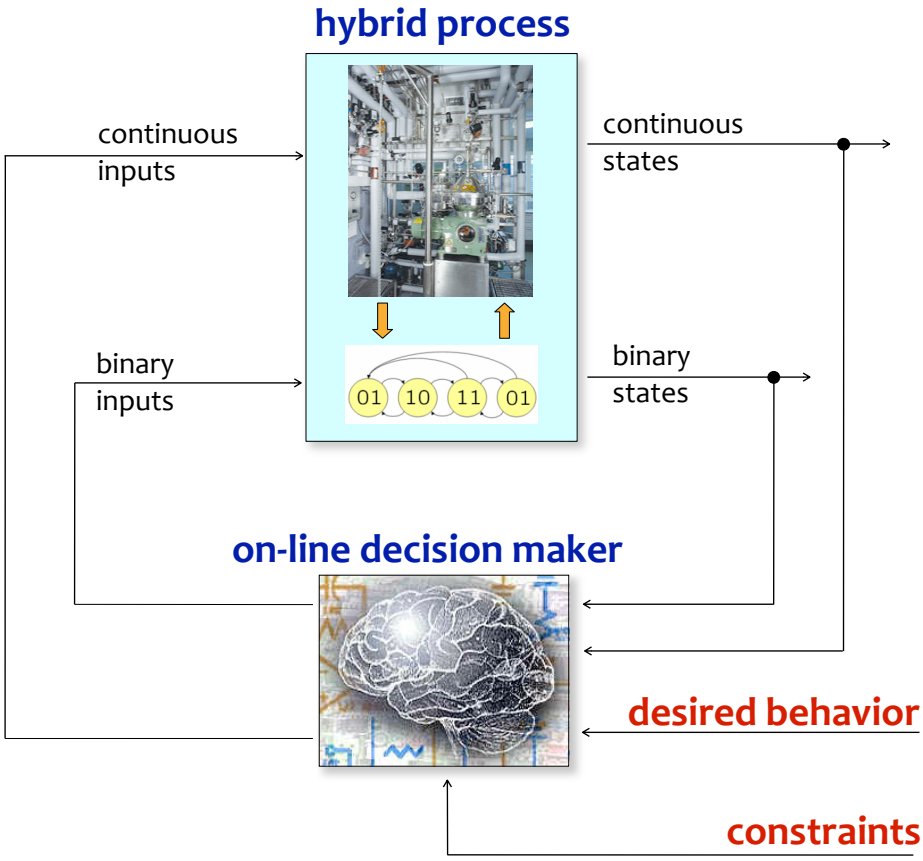




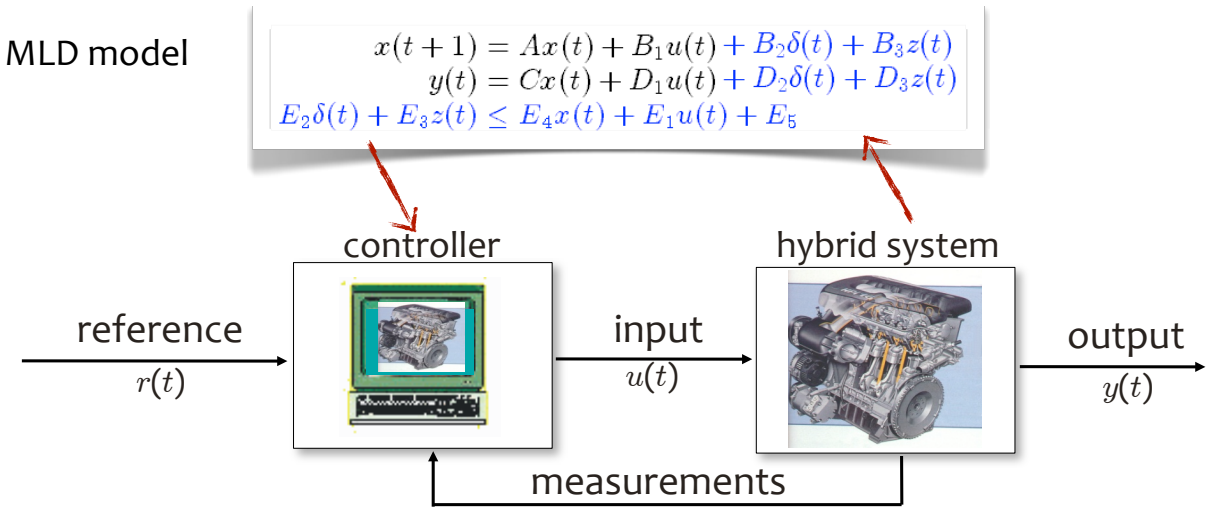
MLD and PWA models are equivalent

Model predictive control of hybrid systems

Hybrid control problem



MPC of hybrid systems



- MODEL: use an MLD (or PWA) model of the plant to predict the future behavior of the hybrid system
- PREDICTIVE: optimization is still based on the predicted future evolution of the hybrid system
- CONTROL: the goal is to control the hybrid system

MPC for hybrid systems

- At time t solve the finite-horizon optimal control problem w.r.t. $U \triangleq \begin{bmatrix} u_0 \\ \vdots \\ u_{N-1} \end{bmatrix}$

$$\begin{aligned} \min_{u_0,\dots,u_{N-1}} \quad & \sum_{k=0}^{N-1} \|y_k - r\|_Q^2 + \|u_k - u_r\|_R^2 \\ & +\sigma\left(\|\delta_k - \delta_r\|^2 + \|z_k - z_r\|^2 + \|x_k - x_r\|^2\right) \\ \text{subject to} \quad & \text{MLD equations} \\ & x_0 = x(t) \\ & x_N = x_r \end{aligned}$$

notation: $\|v\|_Q^2 \triangleq v'Qv$

where the equilibrium condition $(x_r, u_r, \delta_r, z_r)$ is obtained by solving the following mixed-integer feasibility problem

$$\begin{cases} x_r &= Ax_r + B_1u_r + B_2\delta_r + B_3z_r \\ r &= Cx_r + D_1u_r + D_2\delta_r + D_3z_r \\ E_2\delta_r + E_3z_r &\le E_4x_r + E_1u_r + E_5 \end{cases}$$

- Apply only $u(t)$=u_0* (discard the remaining optimal inputs)
- At time $t+1$: get new measurements, repeat optimization

Closed-loop convergence

Theorem Let $(x_r, u_r, \delta_r, z_r)$ be the equilibrium values corresponding to set point r. Assume $x(0)$ is such that the MPC problem is feasible at time t=0.

Then $\forall Q, R \succ 0$, $\forall \sigma > 0$ the closed-loop hybrid MPC loop converges asymptotically

$$\lim_{t\to\infty} y(t) = r \qquad \lim_{t\to\infty} x(t) = x_r$$
$$\lim_{t\to\infty} \delta(t) = \delta_r$$
$$\lim_{t\to\infty} u(t) = u_r \qquad \lim_{t\to\infty} z(t) = z_r$$

and all constraints are fulfilled at each time $t \ge 0$.

(Bemporad, Morari 1999)

Proof: Easily follows from standard Lyapunov arguments

More stability results: see (Lazar, Heemels, Weiland, Bemporad, 2006)

Convergence proof

Main idea: Use **value function** $V^*(x(t))$ as a **Lyapunov** function

- Let U_t = optimal control sequence at time t, $U_t = [u_0^t \ \dots \ u_{N-1}^t]'$
- By construction $\bar{U}_{t+1} = [u_1^t \ \dots \ u_{N-1}^t \ 0]'$ is a feasible sequence at time $t+1$
- The cost of $\bar{U}_{t+1}$ is $V^*(x(t)) - \|y(t) - r\|_Q^2 - \|u - u_r\|_R^2$
  $-\sigma \left(\|\delta(t) - \delta_r\|^2 + \|z(t) - z_r\|^2 + \|x(t) - x_r\|^2 \right) \geq V^*(x(t+1))$
- $V^*(x(t))$ is monotonically decreasing and ≥ 0, so $\exists \lim_{t\to\infty} V^*(x(t)) \triangleq V_\infty$
- Hence $\|y(t) - r\|_Q^2$, $\|u(t) - u_r\|_R^2$ and $\|\delta(t) - \delta_r\|^2$, $\|z(t) - z_r\|^2$, $\|x(t) - x_r\|^2 \to 0$
- Since $R, Q > 0$, $\lim_{t\to\infty} y(t) = r$, $\lim_{t\to\infty} u(t) = u_r$, and all other variables converge. □

Global optimum is not needed to prove convergence !

MIQP Formulation of MPC

(Bemporad, Morari, 1999)

$$\min_\xi J(\xi, x(0)) = \sum_{t=0}^{T-1} y'(t)Qy(t) + u'(t)Ru(t)$$

$$\text{subject to} \begin{cases} x(t+1) = Ax(t) + B_1u(t) + B_2\delta(t) + B_3z(t) + B_5 \\ y(t) = Cx(t) + D_1u(t) + D_2\delta(t) + D_3z(t) + D_5 \\ E_2\delta(t) + E_3z(t) \leq E_4x(t) + E_1u(t) + E_5 \end{cases}$$

- Optimization vector:

$$\xi = [u(0), \dots, u(T-1), \delta(0), \dots, \delta(T-1), z(0), \dots z(T-1)]'$$

$$\Rightarrow \quad \min_\xi \frac{1}{2}\xi' H \xi + x(0)' F \xi + \frac{1}{2} x'(0) Y x(0)$$
$$\text{subj. to} \quad G\xi \leq W + Sx(0)$$

Mixed Integer Quadratic Program (MIQP)

$u \in \mathbb{R}^{n_u}$, $\delta \in \{0,1\}^{n_\delta}$, $z \in \mathbb{R}^{n_z}$ $\Rightarrow$ $\xi \in \mathbb{R}^{(n_u+n_z)T} \times \{0,1\}^{n_\delta T}$

ξ has both real and $\{0,1\}$ components

MILP formulation of MPC

(Bemporad, Borrelli, Morari, 2000)

$$\begin{aligned} \min_{\xi} \quad & J(\xi, x(0)) = \sum_{t=0}^{T-1} \|Qy(t)\|_\infty + \|Ru(t)\|_\infty \\ \text{subject to} \quad & \text{MLD model} \end{aligned}$$

- Basic trick: introduce slack variables

$$\min_x |x| \quad \Rightarrow \quad \begin{aligned} \min_{x,\epsilon} \quad & \epsilon \\ \text{s.t.} \quad & \epsilon \geq x \\ & \epsilon \geq -x \end{aligned}$$

$$\begin{cases} \epsilon_k^x \geq \|Qy(t+k|t)\|_\infty \\ \epsilon_k^u \geq \|Ru(t+k)\|_\infty \end{cases} \Rightarrow \begin{cases} \epsilon_k^x \geq Q^i y(t+k|t) & i=1,\dots,p \quad k=0,\dots,T-1 \\ \epsilon_k^x \geq -Q^i y(t+k|t) & i=1,\dots,p \quad k=0,\dots,T-1 \\ \epsilon_k^u \geq R^i u(t+k) & i=1,\dots,m \quad k=0,\dots,T-1 \\ \epsilon_k^u \geq -R^i u(t+k) & i=1,\dots,m \quad k=0,\dots,T-1 \end{cases}$$

Q^i = ith row of matrix Q

- Optimization vector:

$$\xi = [\epsilon_1^x, \dots, \epsilon_{T-1}^x, \epsilon_0^u, \dots, \epsilon_{T-1}^u, u(0), \dots, u(T-1), \delta(0), \dots, \delta(T-1), z(0), \dots z(T-1)]'$$

$$\Rightarrow \quad \begin{aligned} \min_{\xi} \quad & J(\xi, x(0)) = \sum_{k=0}^{T-1} \epsilon_k^x + \epsilon_k^u \\ \text{s.t.} \quad & G\xi \leq W + Sx(0) \end{aligned}$$

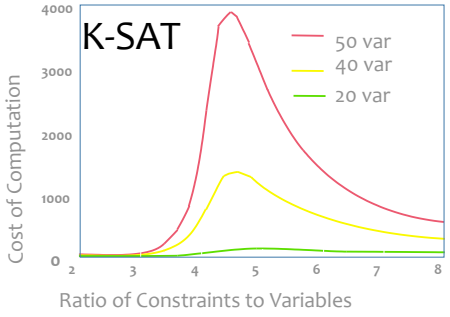
Mixed Integer Linear Program (MILP)

ξ has both real and $\{0,1\}$ components

Mixed-Integer Program (MIP) solvers

- Mixed-Integer Programming is *NP*-complete

Phase transitions have been found in computationally hard problems.



(Monasson et al., *Nature*, 1999)

BUT

- General purpose branch & bound / branch & cut solvers available for MILP and MIQP (CPLEX, GLPK, Xpress-MP, CBC, Gurobi, ...)

More solvers and benchmarks: http://plato.la.asu.edu/bench.html

- No need to reach global optimum (see proof of the theorem), although performance deteriorates

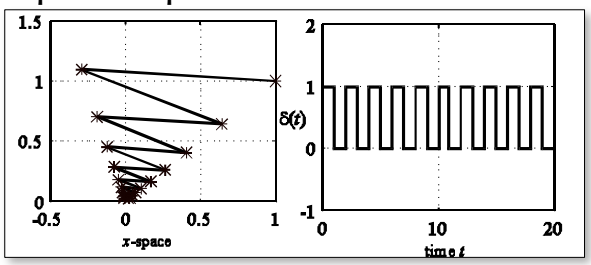
Hybrid MPC example

PWA system:

$$
\begin{aligned}
x(t+1) &= 0.8 \begin{bmatrix} \cos\alpha(t) & -\sin\alpha(t) \\ \sin\alpha(t) & \cos\alpha(t) \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(t) \\
y(t) &= x_2(t) \\
\alpha(t) &= \begin{cases} \frac{\pi}{3} & \text{if } x_1(t) > 0 \\ -\frac{\pi}{3} & \text{if } x_1(t) \le 0 \end{cases}
\end{aligned}
$$

Constraint: $-1 \le u(t) \le 1$

Open loop behavior



go to demo **/demos/hybrid/bm99sim.m**

Hybrid MPC example

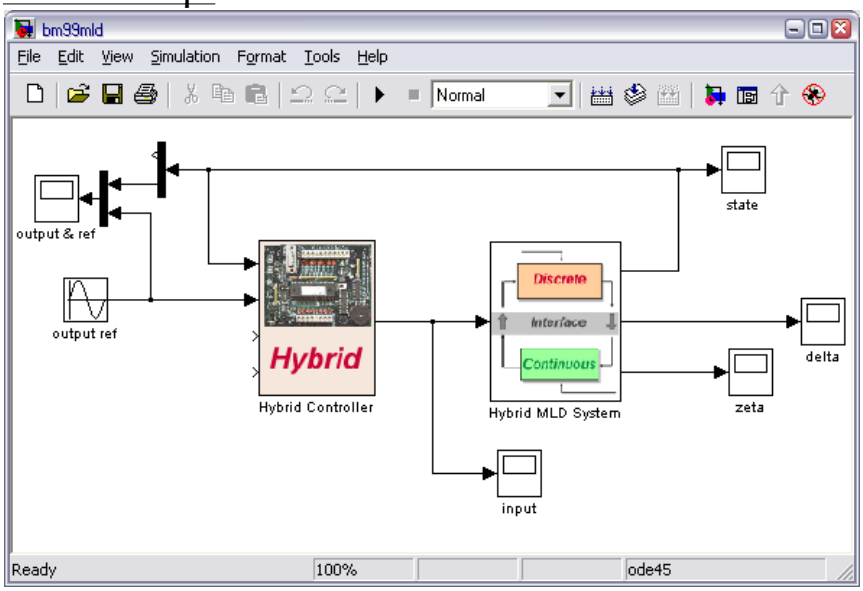
HYSDEL model

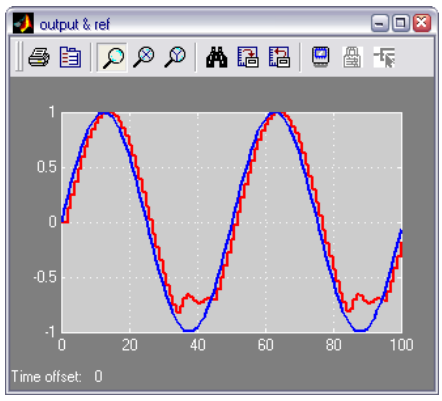
```
/* 2x2 PWA system  - Example from the paper
   A. Bemporad and M. Morari, ``Control of systems integrating logic, dynamics,
   and constraints,'' Automatica, vol. 35, no. 3, pp. 407-427, 1999.
   (C) 2003 by A. Bemporad, 2003 */

SYSTEM pwa {

INTERFACE {
            STATE { REAL x1 [-10,10];
                    REAL x2 [-10,10];}

            INPUT { REAL u [-1.1,1.1];}

            OUTPUT{ REAL y;}

            PARAMETER {
              REAL alpha = 1.0472; /* 60 deg in radiants */
              REAL C = cos(alpha);
              REAL S = sin(alpha);}
           }

IMPLEMENTATION {
            AUX { REAL z1,z2;
                  BOOL sign; }
            AD  { sign = x1<=0; }

            DA  { z1 = {IF sign THEN 0.8*(C*x1+S*x2)
                        ELSE 0.8*(C*x1-S*x2) };
                  z2 = {IF sign THEN 0.8*(-S*x1+C*x2)
                        ELSE 0.8*(S*x1+C*x2) };  }

            CONTINUOUS {x1 = z1;
                        x2 = z2+u; }

            OUTPUT { y = x2;  }
              }
}
```

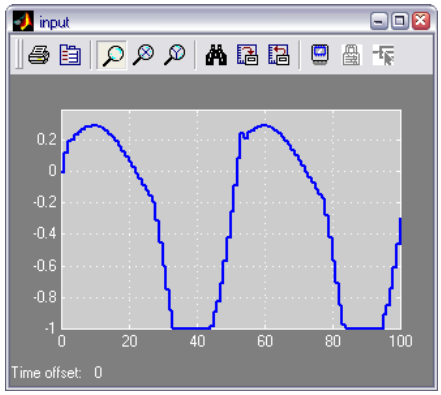
/demos/hybrid/bm99.hys

Hybrid MPC example

Closed-loop:







Performance index:

$$\min \sum_{k=1}^{2} |y(t+k|t) - r(t)|$$

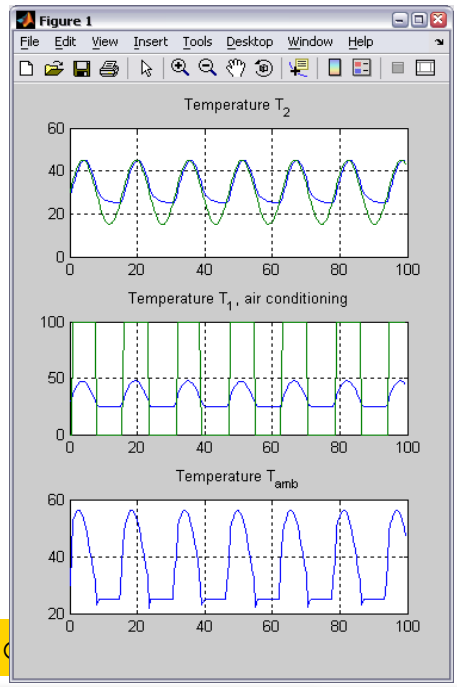
Hybrid MPC – Temperature control

```
>>refs.x=2;    % just weight state #2
>>Q.x=1;
>>Q.rho=Inf;   % hard constraints
>>Q.norm=2;    % quadratic costs
>>N=2;         % optimization horizon
>>limits.xmin=[25;-Inf];
```

$$\min \sum_{k=1}^{2} (x_2(k) - r)^2$$
$$\text{s.t. } x_1(k) \geq 25 \quad k = 1,2$$
$$\text{MLD model}$$

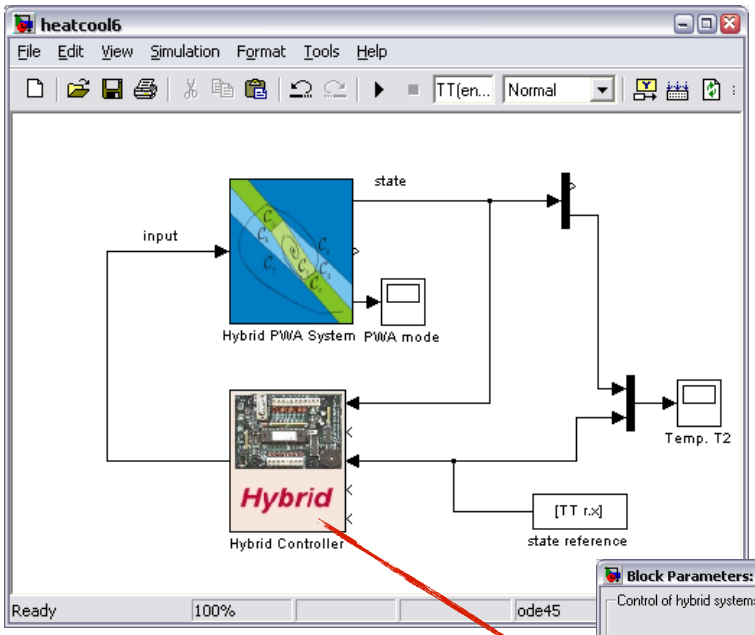
```
>>C=hybcon(S,Q,N,limits,refs);
```

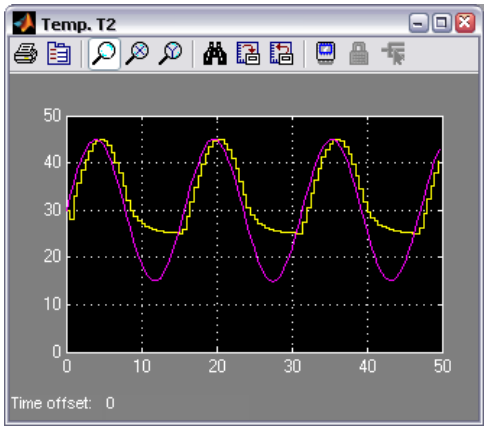
```
>> C

Hybrid controller based on MLD model S <heatcoolmodel.hys>

  2 state measurement(s)
  0 output reference(s)
  0 input reference(s)
  1 state reference(s)
  0 reference(s) on auxiliary continuous z-variables

 20 optimization variable(s) (8 continuous, 12 binary)
 46 mixed-integer linear inequalities
sampling time = 0.5, MILP solver = 'glpk'

Type "struct(C)" for more details.
>>
```

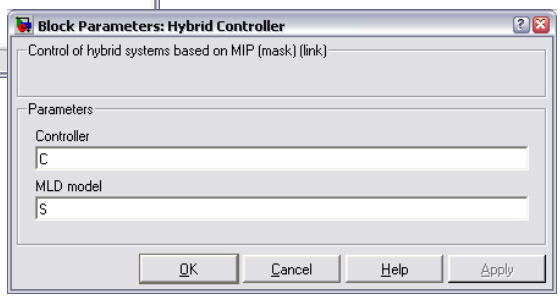



```
>>[XX,UU,DD,ZZ,TT]=sim(C,S,r,x0,Tst
```

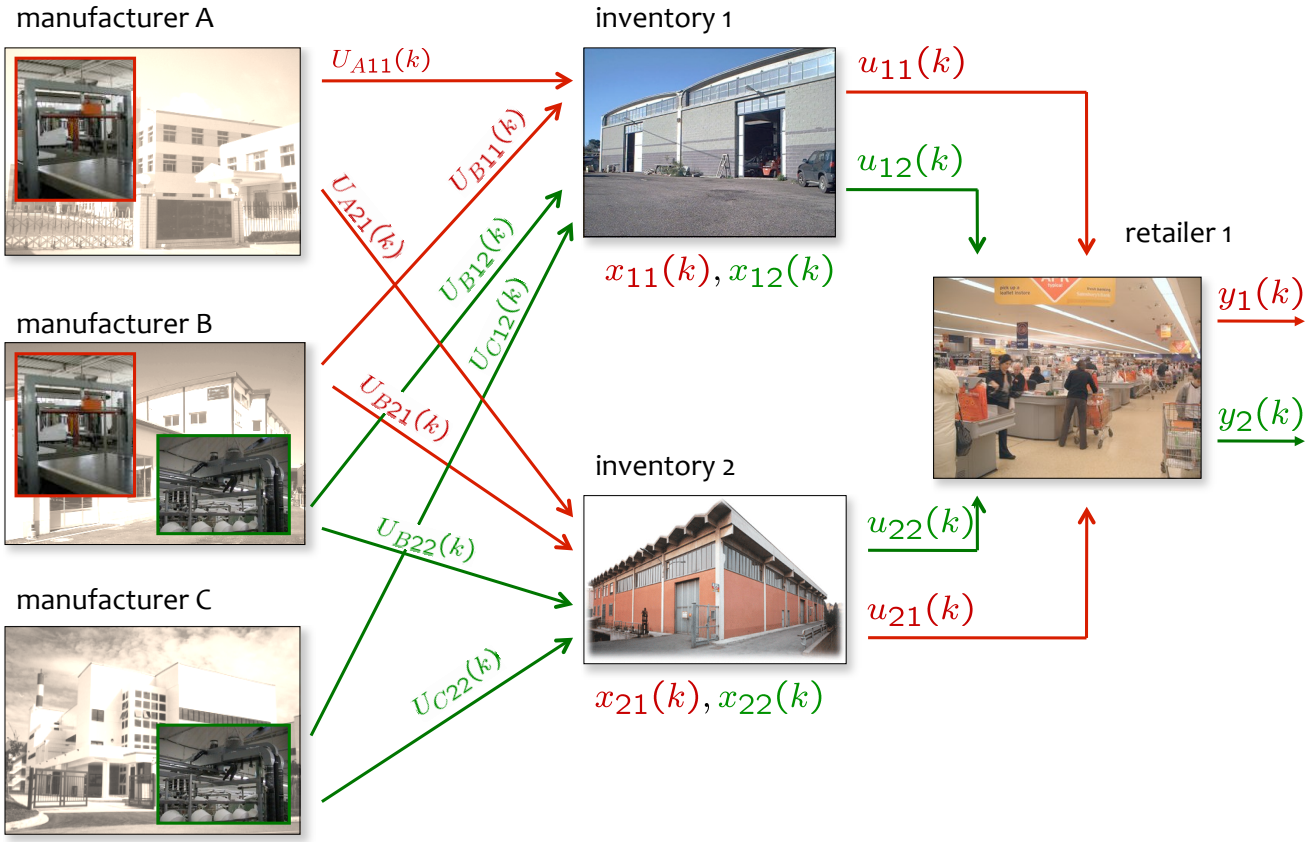
Hybrid MPC – Temperature control







A Simple Example in Supply Chain Management



System Variables

- continuous states:

$x_{ij}(k)$ = amount of j hold in inventory i at time$k$ (i=1,2, j=1,2)

- continuous outputs:

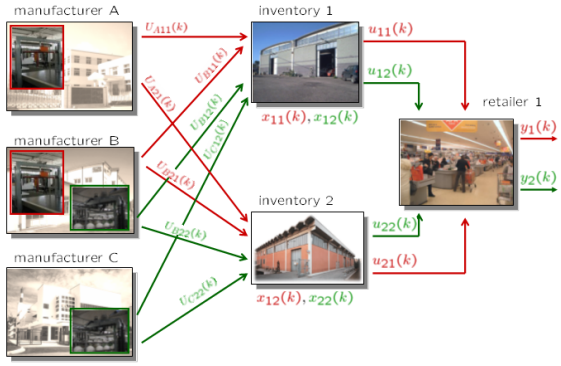
$y_j(k)$ = amount of j sold at time k (j=1,2)

- continuous inputs:

$u_{ij}(k)$ = amount of j taken from inventory i at time k(i=1,2, j=1,2)

- binary inputs:

$U_{Xij}(k)$ = 1 if manufacturer X produces and send j to inventory i at time k

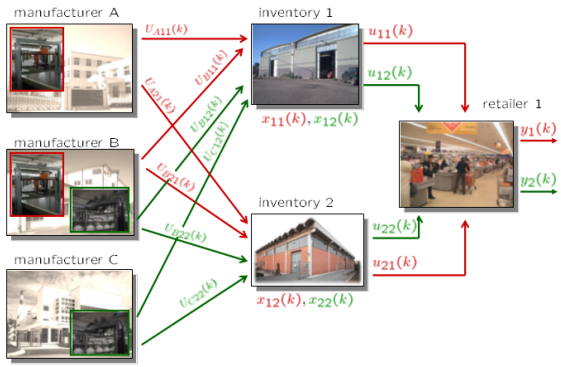


Constraints

- Max capacity of inventory i:

$0 \cdot \sum_j x_{ij}(k) \cdot x_{Mi}$

Numerical values:
x_{M1}=10, x_{M2}=10



- Max transportation from inventories:

$0 \cdot u_{ij}(k) \cdot u_M$

- A product can only be sent to one inventory:

UA11(k) and UA21(k) cannot be =1 at the same time
UB11(k) and UB21(k) cannot be =1 at the same time UB12(k) and UB22(k) cannot be =1 at the same time
UC12(k) and UC22(k) cannot be =1 at the same time

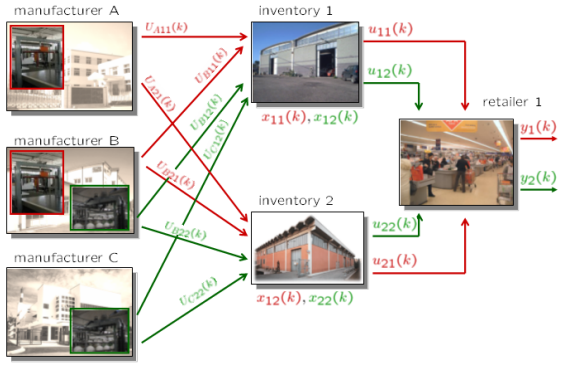
- A manufacturer can only produce one type of product at one time:

[UB11(k)=1 or UB21(k)=1] and [UB12(k)=1 or UB22(k)=1] cannot be true

Dynamics

$P_{A1}, P_{B1}, P_{B2}, P_{C2}$ = amount of type 1(2) produced by A (B,C) in one time interval

Numerical values:
P_{A1}=4, P_{B1}=6, P_{B2}=7, P_{C2}=3

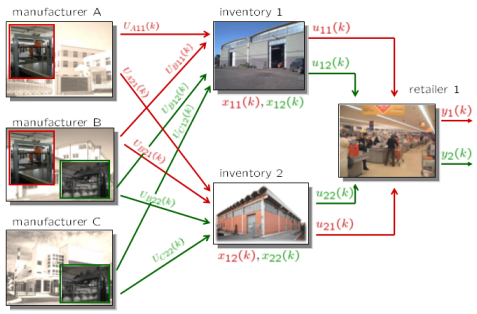


- Level of inventories:

$$\begin{cases} x_{11}(k+1) = x_{11}(k) + P_{A1}U_{A11}(k) + P_{B1}U_{B11}(k) - u_{11}(k) \\ x_{12}(k+1) = x_{12}(k) + P_{B2}U_{B12}(k) + P_{C2}U_{C12}(k) - u_{12}(k) \\ x_{21}(k+1) = x_{21}(k) + P_{A1}U_{A21}(k) + P_{B1}U_{B21}(k) - u_{21}(k) \\ x_{22}(k+1) = x_{22}(k) + P_{B2}U_{B22}(k) + P_{C2}U_{C22}(k) - u_{22}(k) \end{cases}$$

Hybrid Dynamical Model

```
SYSTEM supply_chain{
INTERFACE {
        STATE { REAL x11   [0,10];
                REAL x12   [0,10];
                REAL x21   [0,10];
                REAL x22   [0,10]; }

        INPUT { REAL u11 [0,10];
                REAL u12 [0,10];
                REAL u21 [0,10];
                REAL u22 [0,10];
                BOOL UA11,UA21,UB11,UB12,UB21,UB22,UC12,UC22; }

        OUTPUT {REAL y1,y2;}

        PARAMETER { REAL PA1,PB1,PB2,PC2,xM1,xM2;}
}
IMPLEMENTATION {

        AUX { REAL zA11, zB11, zB12, zC12, zA21, zB21, zB22, zC22;}


        DA { zA11 = {IF UA11 THEN PA1 ELSE 0};
             zB11 = {IF UB11 THEN PB1 ELSE 0};
             zB12 = {IF UB12 THEN PB2 ELSE 0};
             zC12 = {IF UC12 THEN PC2 ELSE 0};
             zA21 = {IF UA21 THEN PA1 ELSE 0};
             zB21 = {IF UB21 THEN PB1 ELSE 0};
             zB22 = {IF UB22 THEN PB2 ELSE 0};
             zC22 = {IF UC22 THEN PC2 ELSE 0}; }
```

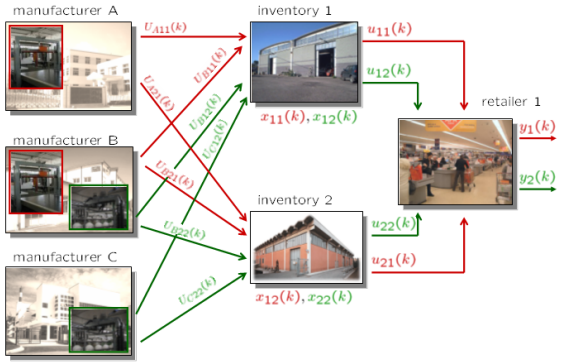



```
        CONTINUOUS {x11 = x11 + zA11 + zB11 - u11;
                    x12 = x12 + zB12 + zC12 - u12;
                    x21 = x21 + zA21 + zB21 - u21;
                    x22 = x22 + zB22 + zC22 - u22;  }

        OUTPUT {    y1 = u11 + u21;
                    y2 = u12 + u22; }

        MUST {  ~(UA11 & UA21);
            ~(UC12 & UC22);
            ~((UB11 | UB21) & (UB12 | UB22));
             ~(UB11 & UB21);
            ~(UB12 & UB22);
            x11+x12 <= xM1;
            x11+x12 >=0;
            x21+x22 <= xM2;
            x21+x22 >=0; }
} }
```

/demos/hybrid/supply_chain.m

Objectives

- Meet customer demand as much as possible: $y_1 \approx r_1, y_2 \approx r_2$



- Minimize transportation costs

- Fulfill all constraints

Performance Specs

penalty on demand tracking error

$$\min \sum_{k=0}^{N-1} 10(|y_1(k) - r_1(k)| + |y_2(k) - r_2(k)|) +$$

cost for shipping from inv.#1 to market

$$4(|u_{11}(k)| + |u_{12}(k)|) +$$

$$2(|u_{21}(k)| + |u_{22}(k)|) +$$

cost for shipping from inv.#2 to market

$$1(|U_{A11}(k)| + |U_{A21}(k)|) +$$

cost from A to inventories

$$4(|U_{B11}(k)| + |U_{B12}(k)| + |U_{B21}(k)| + |U_{B22}(k)|) +$$

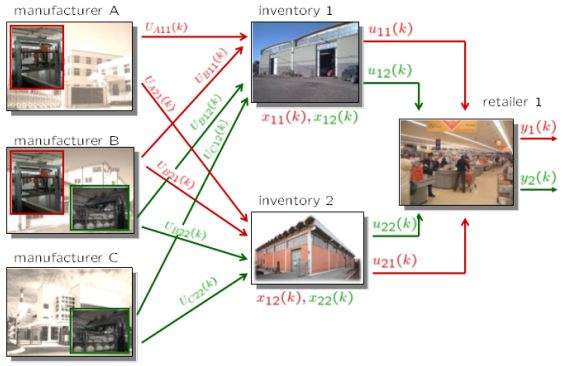
$$10(|U_{C12}(k)| + |U_{C22}(k)|)$$

cost from B to inventories

cost from C to inventories

Simulation setup

```
>>refs.y=[1 2];       % weights output2 #1,#2
>>Q.y=diag([10 10]); % output weights
…
>>Q.norm=Inf;         % infinity norms
>>N=2;                % optimization horizon
>>limits.umin=umin;   % constraints
>>limits.umax=umax;
>>limits.xmin=xmin;
>>limits.xmax=xmax;
```



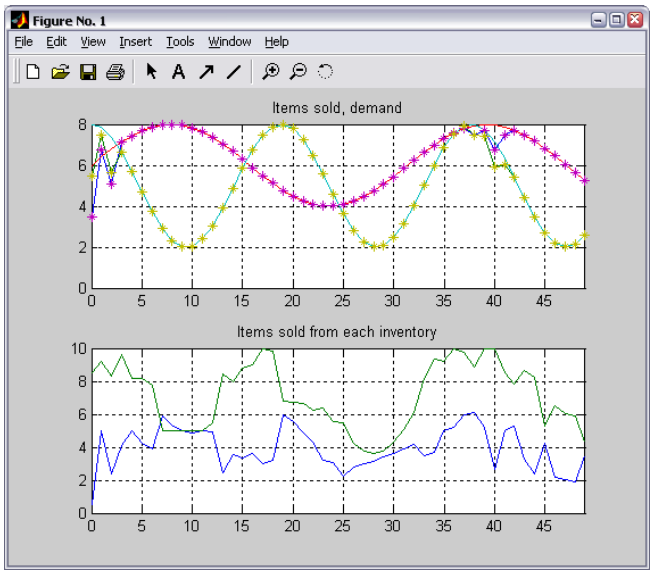

```
>>C=hybcon(S,Q,N,limits,refs);
```

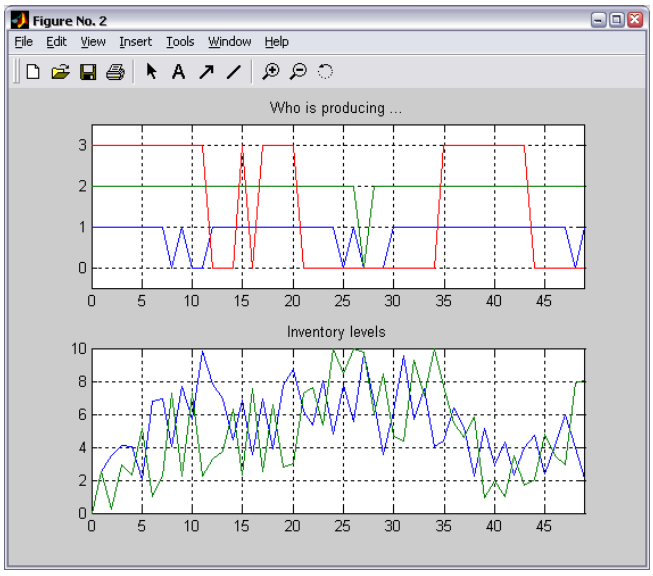
```
>> C

Hybrid controller based on MLD model S <supply_chain.hys> [Inf-norm]

  4 state measurement(s)
  2 output reference(s)
 12 input reference(s)
  0 state reference(s)
  0 reference(s) on auxiliary continuous z-variables

 44 optimization variable(s) (28 continuous, 16 binary)
176 mixed-integer linear inequalities
sampling time = 1, MILP solver = 'glpk'

Type "struct(C)" for more details.

>>
```


Simulation results

```
>>x0=[0;0;0;0];                    % Initial condition
>>r.y=[6+2*sin((0:Tstop-1)'/5)     % Reference trajectories
       5+3*cos((0:Tstop-1)'/3)];
```

```
>>[XX,UU,DD,ZZ,TT]=sim(C,S,r,x0,Tstop);
```





CPU time: ≈ 13 ms per time step (using GLPK on this machine)

Explicit hybrid MPC

Explicit hybrid MPC (MLD formulation)

$$\min_{\xi} J(\xi, x(t)) = \sum_{k=0}^{T-1} \|Q(y_k)\|_\infty + \|Ru_k\|_\infty$$

$$\text{subject to} \begin{cases} x_{k+1} = Ax_k + B_1u_k + B_2\delta_k + B_3z_k + B_5 \\ y_k = Cx_k + D_1u_k + D_2\delta_k + D_3z_k + D_5 \\ E_2\delta_k + E_3z_k \le E_4x_k + E_1u_k + E_5 \\ x_0 = x(t) \end{cases}$$

- On-line optimization: solve the problem for a given state $x(t)$

Mixed-Integer Linear Program (MILP)

- Off-line optimization: solve the MILP in advance **for all** $x(t)$

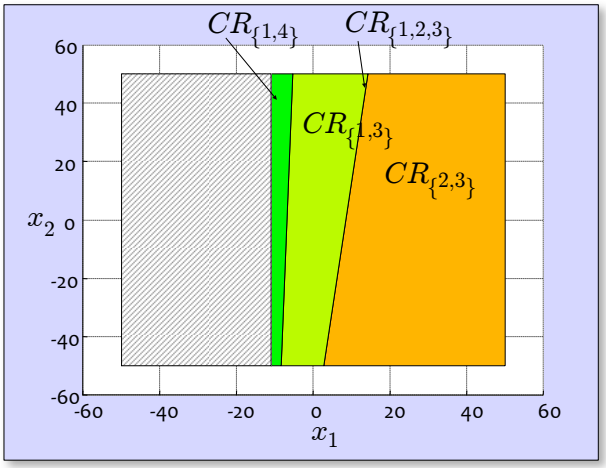
$$\min_{\xi} \sum_{k=0}^{T-1} \epsilon_k^x + \epsilon_k^u$$
$$\text{s.t. } G\xi \le W + Sx(t)$$

multi-parametric Mixed Integer Linear Program (mp-MILP)

Example of multiparametric solution

Multiparametric LP

$$\begin{aligned}\min_{\xi} \quad & -3\xi_1 - 8\xi_2 \\ \text{s.t.} \quad & \begin{cases} \xi_1 + \xi_2 \le 13 + x_1 \\ 5\xi_1 - 4\xi_2 \le 20 \\ -8\xi_1 + 22\xi_2 \le 121 + x_2 \\ -4\xi_1 - \xi_2 \le -8 \\ -\xi_1 \le 0 \\ -\xi_2 \le 0 \end{cases}\end{aligned}$$



$$\xi(x) = \begin{cases} \begin{bmatrix} 0.00 & 0.05 \\ 0 & 0.06 \end{bmatrix} x + \begin{bmatrix} 11.85 \\ 9.80 \end{bmatrix} & \text{if } \begin{bmatrix} 0.02 & 0.00 \\ 0.00 & 0.02 \\ 0.00 & -0.02 \\ -0.12 & 0.01 \end{bmatrix} x \le \begin{bmatrix} 1.00 \\ 1.00 \\ 1.00 \\ -1.00 \end{bmatrix} & \text{CR}_{\{2,3\}} \\ \begin{bmatrix} 0.73 & -0.03 \\ 0.27 & 0.03 \end{bmatrix} x + \begin{bmatrix} 5.50 \\ 7.50 \end{bmatrix} & \text{if } \begin{bmatrix} 0.00 & 0.02 \\ 0.00 & -0.02 \\ 0.12 & -0.01 \\ -0.15 & 0.00 \end{bmatrix} x \le \begin{bmatrix} 1.00 \\ 1.00 \\ 1.00 \\ 1.00 \end{bmatrix} & \text{CR}_{\{1,3\}} \\ \begin{bmatrix} -0.33 & 0.00 \\ 1.33 & 0 \end{bmatrix} x + \begin{bmatrix} -1.67 \\ 14.67 \end{bmatrix} & \text{if } \begin{bmatrix} 0.00 & 0.02 \\ 0.00 & -0.02 \\ 0.15 & -0.00 \\ -0.09 & 0.00 \end{bmatrix} x \le \begin{bmatrix} 1.00 \\ 1.00 \\ -1.00 \\ 1.00 \end{bmatrix} & \text{CR}_{\{1,4\}} \end{cases}$$

Multiparametric MILP

$$\begin{aligned}\min_{\xi_c,\xi_d} \quad & f_c'\xi_c + f_d'\xi_d \\ \text{s.t.} \quad & G_c\xi_c + G_d\xi_d \le W + Sx\end{aligned}$$

$\xi_c \in \mathbb{R}^n$

$\xi_d \in \{0,1\}^m$

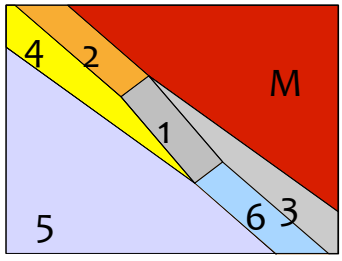
- mp-MILP can be solved (by alternating MILPs and mp-LPs)

(Dua, Pistikopoulos, 1999)

- **Theorem:** The multiparametric solution $\xi^*(x)$ is piecewise affine
- **The MPC controller is piecewise affine in** $x=x(t)$

$$u(x) = \begin{cases} F_1x + g_1 & \text{if } H_1x \le K_1 \\ \vdots & \vdots \\ F_Mx + g_M & \text{if } H_Mx \le K_M \end{cases}$$

x-space



(more generally, the MPC controller is a PWA function of x and of the reference signals)

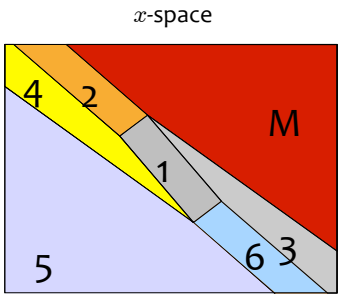
Explicit hybrid MPC (PWA formulation)

$$\min_U J(U, x(t)) = \sum_{k=0}^{T-1} \|Q(y_k)\|_\infty + \|Ru_k\|_\infty$$

$$\text{subject to} \begin{cases} x_{k+1} = A_{i(k)}x_k + B_{i(k)}u_k + f_{i(k)} \\ y_k = C_{i(k)}x_k + D_{i(k)}u_k + g_{i(k)} \\ i(k) \text{ such that } H_{i(k)}x_k + W_{i(k)}u_k \le K_{i(k)} \\ x_0 = x(t) \end{cases}$$

- **The MPC controller is piecewise affine in $x=x(t)$**

$$u(x) = \begin{cases} F_1 x + g_1 & \text{if} \quad H_1 x \le K_1 \\ \vdots & \vdots \\ F_M x + g_M & \text{if} \quad H_M x \le K_M \end{cases}$$



(more generally, the MPC controller is a PWA function of $x$and of the reference signals)

Note: With quadratic costs, partition may not be fully polyhedral (see next slide)

Computation of explicit hybrid MPC (PWA)

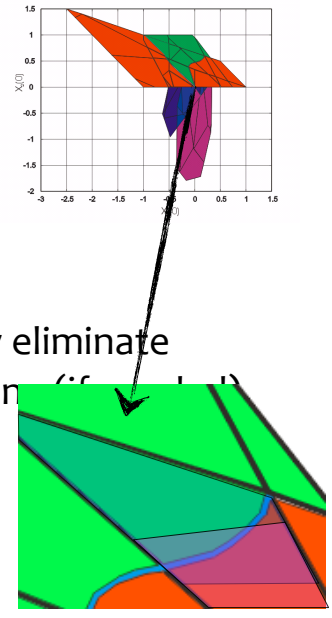
Method A: (Borrelli, Baotic, Bemporad, Morari, *Automatica*, 2005)

Use a combination of DP (dynamic programming) and mpLP (1-norm, ∞-norm), or mpQP (quadratic forms)

Method B: (Bemporad, *Hybrid Toolbox*, 2003) (Alessio, Bemporad, ADHS 2006)(Mayne, ECC 2001)

1 - Use backwards (=DP) reachability analysis for enumerating all feasible mode sequences $I=\{i(0),i(1),\ldots,i(T)\}$;

2 - For each fixed sequence I, solve the explicit finite-time optimal control problem for the corresponding linear time-varying system (mpQP or mpLP);



3 - Case 1/∞-norm: Compare value functions and split regions.

Quadratic case: keep overlapping regions (possibly eliminate overlaps that are never optimal) and compare on-lin

Note: With quadratic costs, partition may not be fully polyhedral ⟹ Better keep overlapping polyhedra

Hybrid control examples (revisited)

Hybrid control example

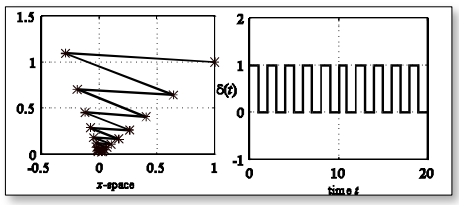
PWA system:

$$\begin{aligned} x(t+1) &= 0.8 \begin{bmatrix} \cos\alpha(t) & -\sin\alpha(t) \\ \sin\alpha(t) & \cos\alpha(t) \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(t) \\ y(t) &= x_2(t) \\ \alpha(t) &= \begin{cases} \frac{\pi}{3} & \text{if } x_1(t) \geq 0 \\ -\frac{\pi}{3} & \text{if } x_1(t) < 0 \end{cases} \end{aligned}$$

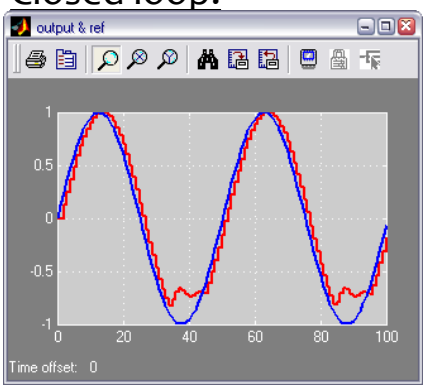
Constraints: $-1 \leq u(t) \leq 1$

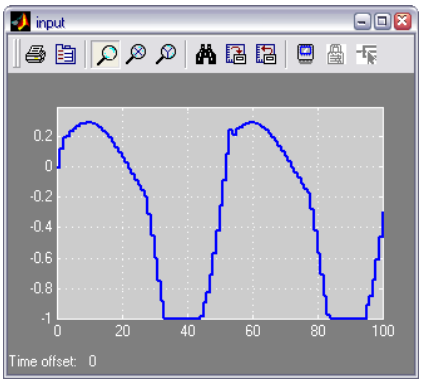
Objective: $\min \sum_{k=1}^{2} |y(t+k|t) - r(t)|$

Open loop behavior:



Closed loop:

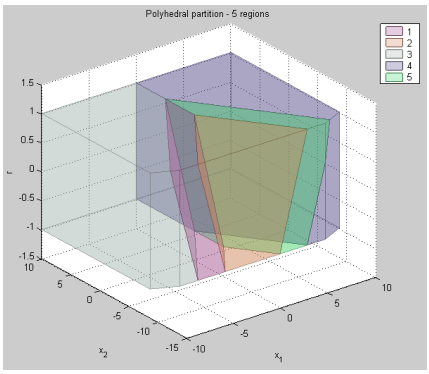




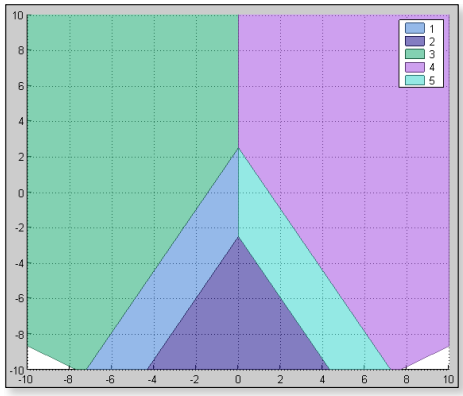
HybTbx: **/demos/hybrid/bm99sim.m**

Explicit PWA controller

$$u(x,r)=\begin{cases}\begin{bmatrix}0.6928 & -0.4 & 1\end{bmatrix}\begin{bmatrix}x\\ r\end{bmatrix} & \text{if } \begin{bmatrix}0.6928 & -0.4 & 1\\ -0.4 & -0.6928 & 0\\ 0 & -1 & 0\\ 0 & 0 & -1\\ -0.6928 & 0.4 & -1\\ 0 & 0 & 1\\ 1 & 0 & 0\end{bmatrix}\begin{bmatrix}x\\ r\end{bmatrix}\le\begin{bmatrix}1\\ 10\\ 10\\ 1\\ 1\\ 1\\ 1e{-}006\end{bmatrix} \\ \text{(Region \#1)} \\ 1 & \text{if } \begin{bmatrix}-0.6928 & 0.4 & -1\\ 0.6928 & 0.4 & -1\\ 0 & 0 & 1\\ 0 & 0 & -1\\ 0 & -1 & 0\end{bmatrix}\begin{bmatrix}x\\ r\end{bmatrix}\le\begin{bmatrix}-1\\ -1\\ 1\\ 1\\ 10\end{bmatrix} \\ \text{(Region \#2)} \\ -1 & \text{if } \begin{bmatrix}-0.4 & -0.6928 & 0\\ 0 & -1 & 0\\ 1 & 0 & 0\\ 0 & 1 & 0\\ 0.6928 & -0.4 & 1\\ 0 & 0 & 1\\ 0 & 0 & -1\\ -1 & 0 & 0\end{bmatrix}\begin{bmatrix}x\\ r\end{bmatrix}\le\begin{bmatrix}10\\ 10\\ 1e{-}006\\ 10\\ -1\\ 1\\ 1\\ 10\end{bmatrix} \\ \text{(Region \#3)} \\ -1 & \text{if } \begin{bmatrix}-1 & 0 & 0\\ 0 & 1 & 0\\ 0.4 & -0.6928 & 0\\ 0 & -1 & 0\\ -0.6928 & -0.4 & 1\\ 0 & 0 & 1\\ 0 & 0 & -1\\ 1 & 0 & 0\end{bmatrix}\begin{bmatrix}x\\ r\end{bmatrix}\le\begin{bmatrix}0\\ 10\\ 10\\ 10\\ -1\\ 1\\ 1\\ 10\end{bmatrix} \\ \text{(Region \#4)} \\ \begin{bmatrix}-0.6928 & -0.4 & 1\end{bmatrix}\begin{bmatrix}x\\ r\end{bmatrix} & \text{if } \begin{bmatrix}-0.6928 & -0.4 & 1\\ 0.4 & -0.6928 & 0\\ 0 & -1 & 0\\ 0 & 0 & -1\\ 0.6928 & 0.4 & -1\\ 0 & 0 & 1\\ -1 & 0 & 0\end{bmatrix}\begin{bmatrix}x\\ r\end{bmatrix}\le\begin{bmatrix}1\\ 10\\ 10\\ 1\\ 1\\ 1\\ 0\end{bmatrix} \\ \text{(Region \#5)}\end{cases}$$



Section with r=0



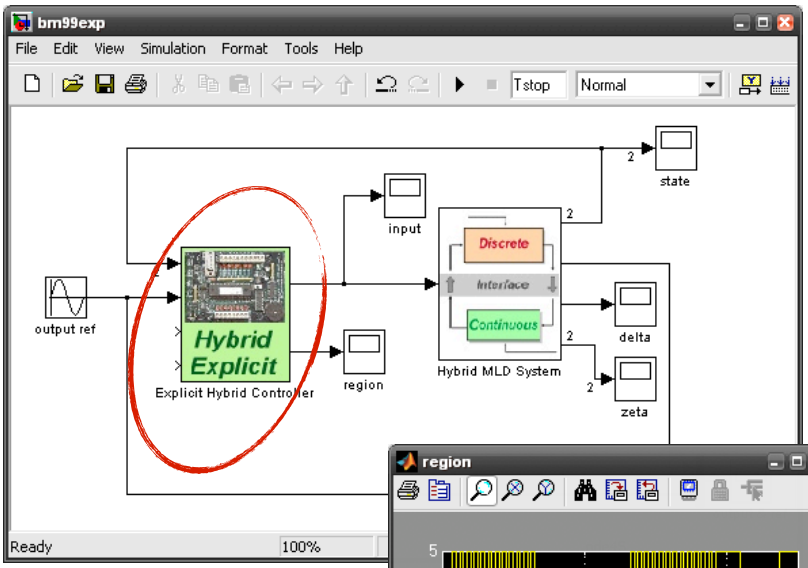
HybTbx: **/demos/hybrid/bm99sim.m**

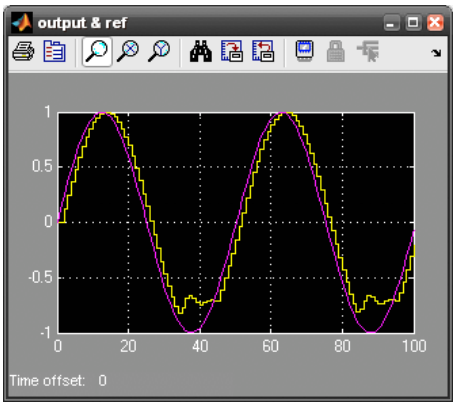
(CPU time: 1.51 s, Pentium M 1.4GHz)

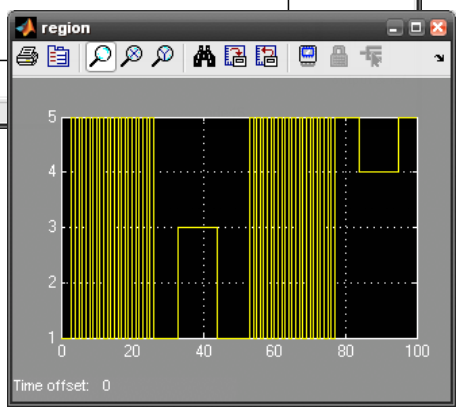
PWA law ≡ MPC law !

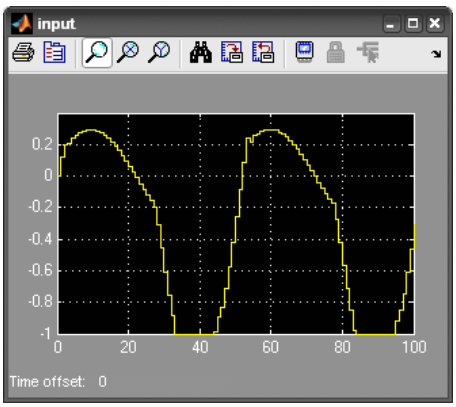
Hybrid control example

Closed loop:





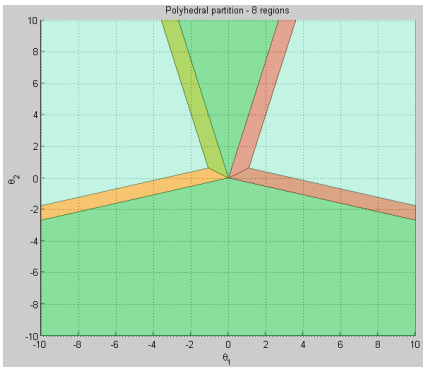




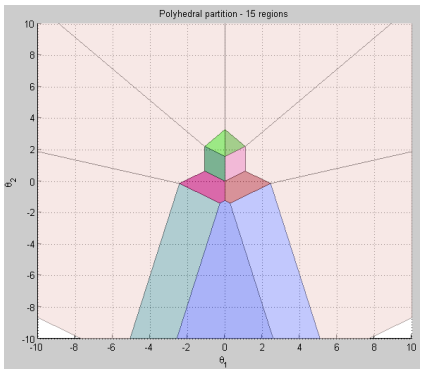
Explicit PWA regulator

Objective: $\min \sum_{k=1}^{N} \|x(t+k|t)\|_\infty$

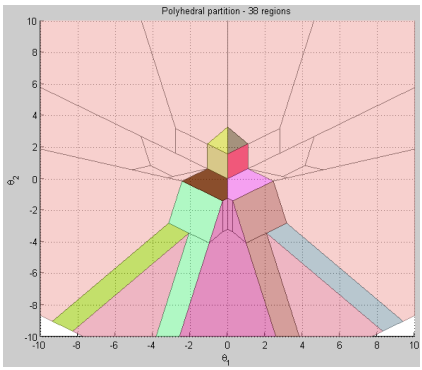
Prediction horizon N=1



Prediction horizon N=2



Prediction horizon N=3

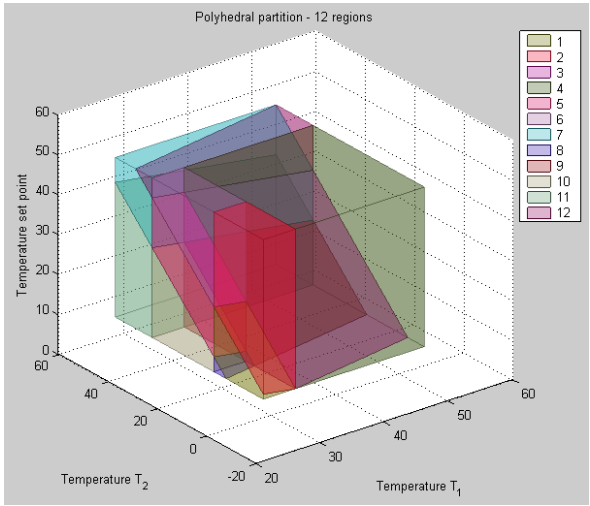


HybTbx: **/demos/hybrid/bm99benchmark.m**

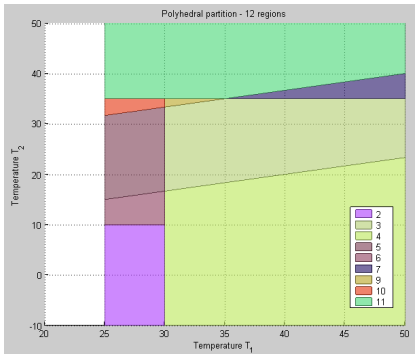
Explicit MPC – Temperature control

```
>>E=expcon(C,range,options);
```

```
>> E

Explicit controller (based on hybrid controller C)
   3 parameter(s)
   1 input(s)
  12 partition(s)
sampling time = 0.5

The controller is for hybrid systems (tracking)
This is a state-feedback controller.

Type "struct(E)" for more details.
>>
```

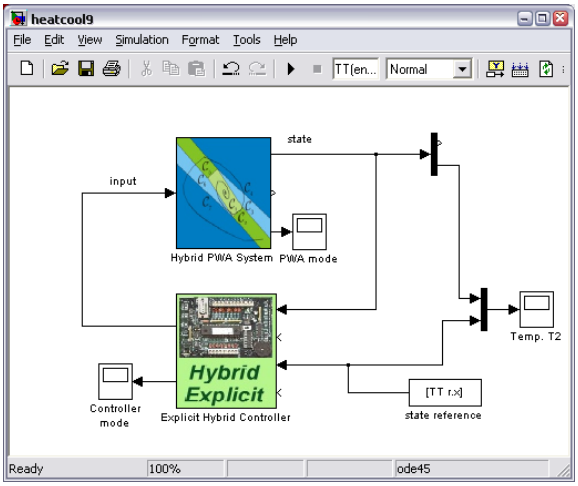


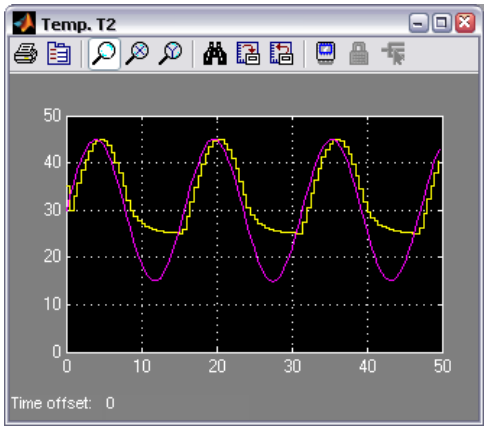
$$\min \sum_{k=1}^{2} (x_2(k) - r)^2$$
$$\text{s.t. } x_1(k) \geq 25 \quad k = 1,2$$
PWA model

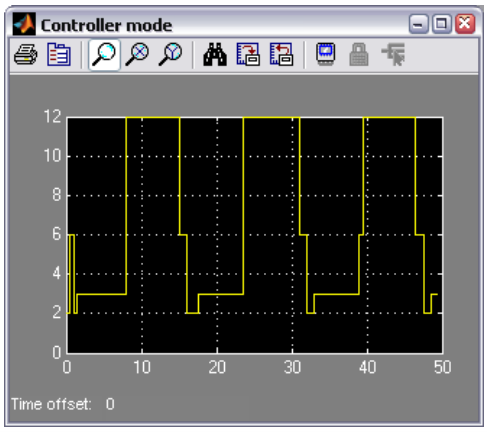


Section in the (T_1,T_2)-space for T_{ref} = 30

Explicit MPC – Temperature control







Generated C-code

utils/expcon.h

```
#define EXPCON_REG 12
#define EXPCON_NTH 3
#define EXPCON_NYM 2
#define EXPCON_NH 72
#define EXPCON_NF 12
static double EXPCON_F[]={
    -1,0,0,0,-1,0,
    -1,-1,-1,-1,-1,-1,0,-3,-3,
    -3,0,-3,0,0,0,0,0,
    0,0,4,4,4,0,4,0,0,
    0,0,0,0};

static double EXPCON_G[]={
    101.6,1.6,1.6,-1.6,98.4001,0,100,51.6,
    101.6,51.6,48.4,50};

static double EXPCON_H[]={
    0,0,0,-0.00999999,0,-0.0333333,
    0.02,0.00999999,-0.02,0,0,-0.0333333,0.02,0.00999999,
    0,0,-0.02,0.02,0,-1,0.00999999,0,
```

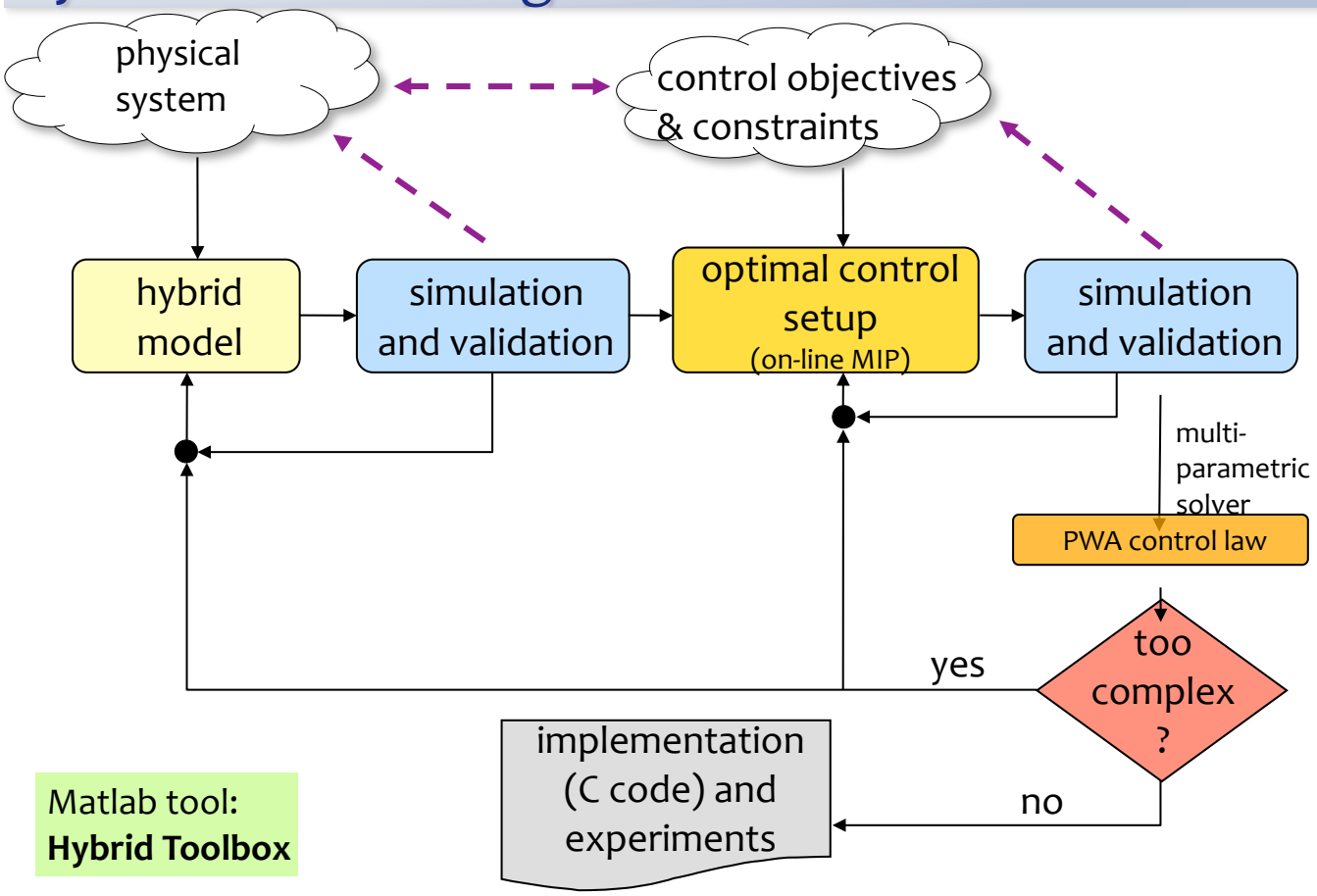

Implementation aspects of hybrid MPC

- **Alternatives:** (1) solve MIP on-line
  (2) evaluate a PWA function

- Small problems (short horizon N=1,2, one or two inputs): explicit PWA control law preferable
  - time to evaluate the control law is shorter than MIP
  - control code is simpler (no complex solver must be included in the control software !)
  - more insight in controller's behavior

- Medium/large problems (longer horizon, many inputs and binary variables): MIP preferable

Hybrid control design flow



General remarks about MIP modeling

The complexity of solving a mixed-integer program largely depends on the number of integer (binary) variables involved in the problem.

Henceforth, when creating a hybrid model one has to

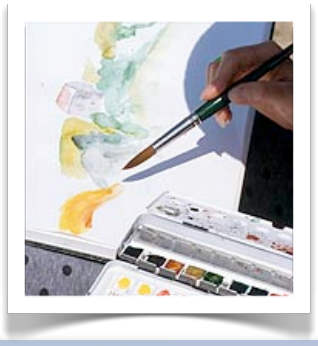
Be thrifty with integer variables !

Adding logical constraints usually helps ...

Generally speaking:

modeling is an art

(a unifying general theory does not exist)



Bibliography

[1] A. Bemporad and M. Morari, "Control of systems integrating logic, dynamics, and constraints," Automatica, vol. 35, no. 3, pp. 407–427, 1999.

[2] M. Lazar, W.P.M.H. Heemels, S. Weiland, and A. Bemporad, "Stabilizing model predictive control of hybrid systems," IEEE Trans. Automatic Control, vol. 51, no. 11, pp. 1813–1818, 2006.

[3] F. Borrelli, M. Baotic, A. Bemporad, and M. Morari, "Dynamic programming for constrained optimal control of discrete-time linear hybrid systems," *Automatica*, vol. 41, no. 10, Oct. 2005

[4] A. Bemporad, G. Ferrari-Trecate, and M. Morari, "Observability and controllability of piecewise affine and hybrid systems," IEEE Trans. Automatic Control, vol. 45, no. 10, pp. 1864–1876, 2000.

[5] W.P.M.H. Heemels, B. De Schutter, and A. Bemporad, "Equivalence of hybrid dynamical models," Automatica, vol. 37, no. 7, pp. 1085–1091, July 2001.

[6] A. Bemporad, A. Garulli, S. Paoletti, and A. Vicino, "A bounded-error approach to piecewise affine system identification," IEEE Trans. Automatic Control, vol. 50, no. 10, pp. 1567–1580, Oct. 2005.

[7] A. Bemporad and S. Di Cairano, "Model-predictive control of discrete hybrid stochastic automata," IEEE Trans. Automatic Control, vol. 56, no. 6, pp. 1307–1321, 2011.

[8] S. Di Cairano, A. Bemporad, and J. Júlvez, "Event-driven optimization-based control of hybrid systems with integral continuous-time dynamics," Automatica, vol. 45, pp. 1243–1251, 2009.

[9] A. Bemporad, G. Bianchini, and F. Brogi, "Passivity analysis and passification of discrete-time hybrid systems," IEEE Trans. Automatic Control, vol. 54, no. 4, pp. 1004–1009, 2008.

[10] A. Bemporad and N. Giorgetti, "Logic-based methods for optimal control of hybrid systems," IEEE Trans. Automatic Control, vol. 51, no. 6, pp. 963–976, 2006.

Bibliography

Toolbox

[11] F.D. Torrisi and A. Bemporad, "HYSDEL — A tool for generating computational hybrid models," IEEE Trans. Contr. Systems Technology, vol. 12, no. 2, pp. 235–249, Mar. 2004.

[12] A. Bemporad, "Efficient conversion of mixed logical dynamical systems into an equivalent piecewise affine form," IEEE Trans. Automatic Control, vol. 49, no. 5, pp. 832–838, 2004.

[13] A. Bemporad, *Hybrid Toolbox – User's Guide*, Jan. 2004, http://www.ing.unitn.it/~bemporad/hybrid/toolbox

Book

[14] F. Borrelli, A. Bemporad, and M. Morari, *Predictive control for linear and hybrid systems*, Cambridge University Press, 2011, In press.

Publications available for download at **http://www.ing.unitn.it/~bemporad/publications**