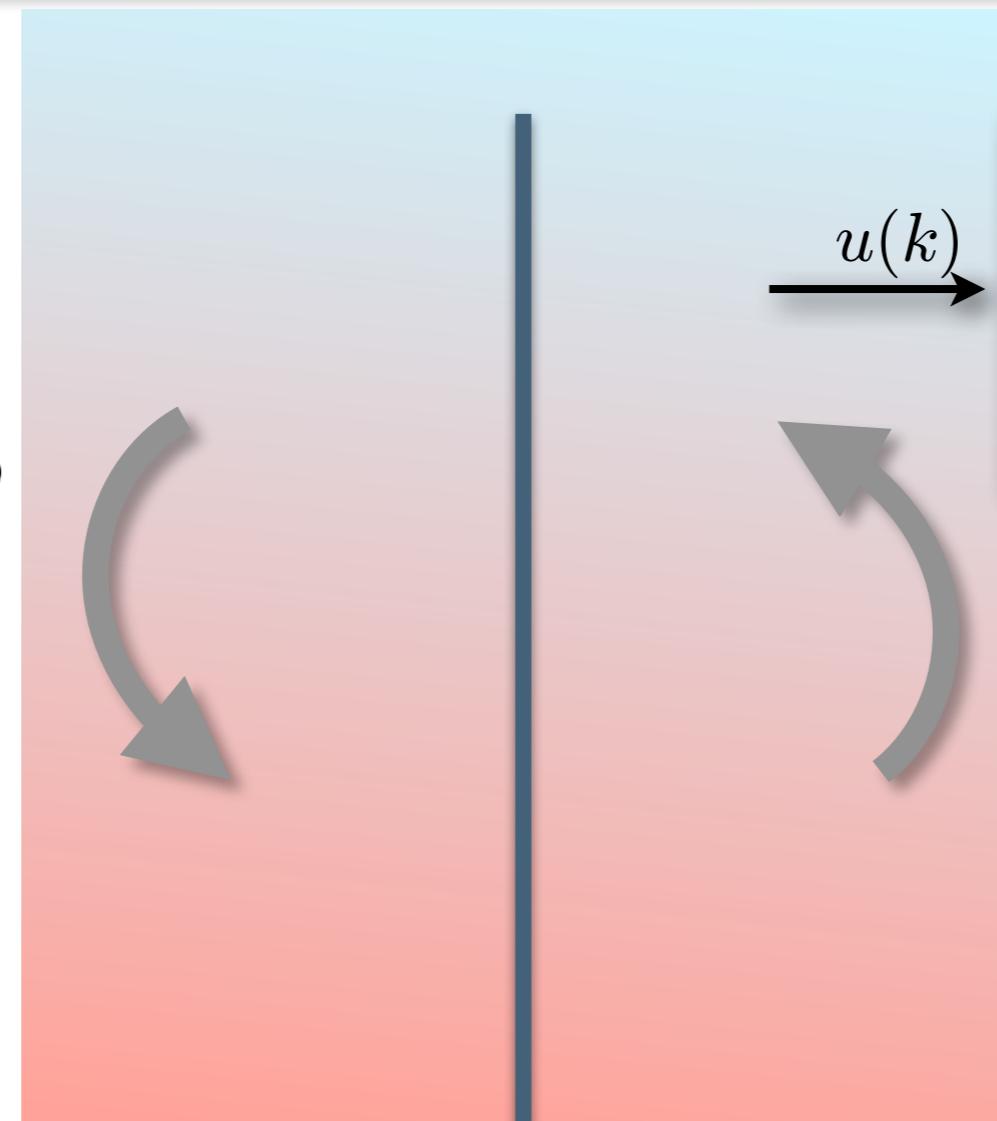
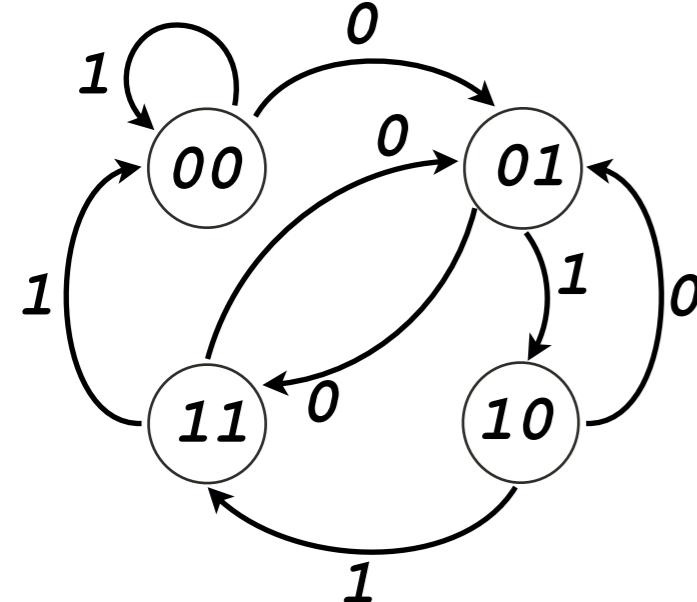


HYBRID SYSTEMS: MODELING

Alberto Bemporad - “Model Predictive Control” course - Academic year 2016/17

HYBRID DYNAMICAL SYSTEMS



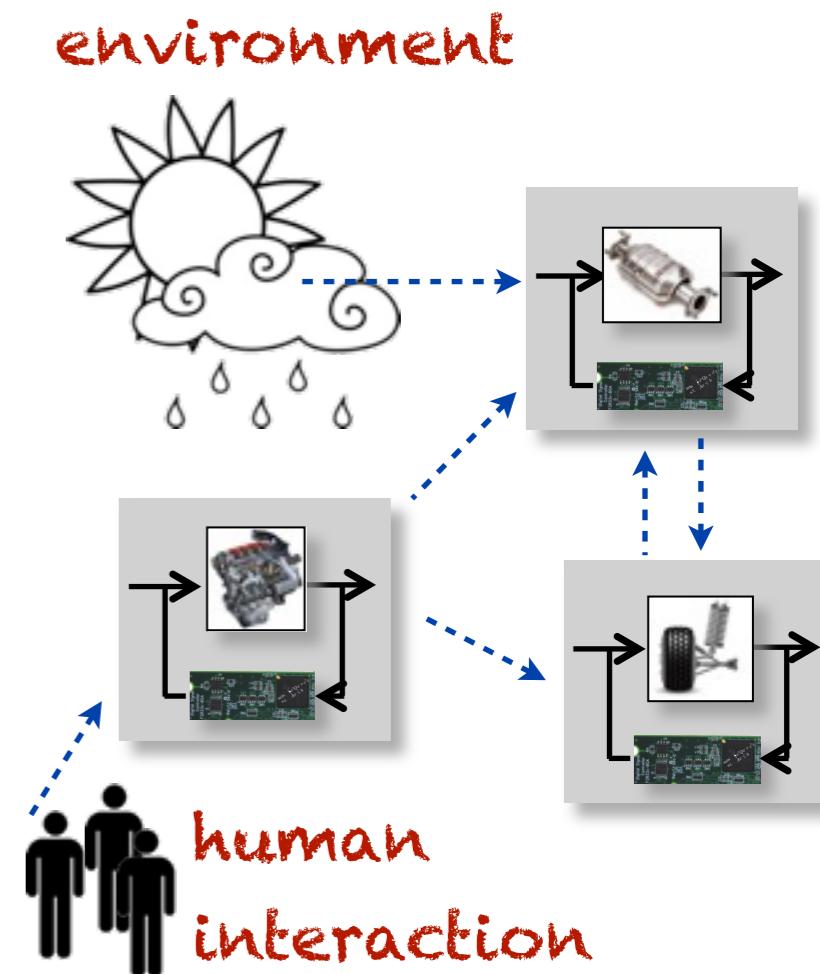
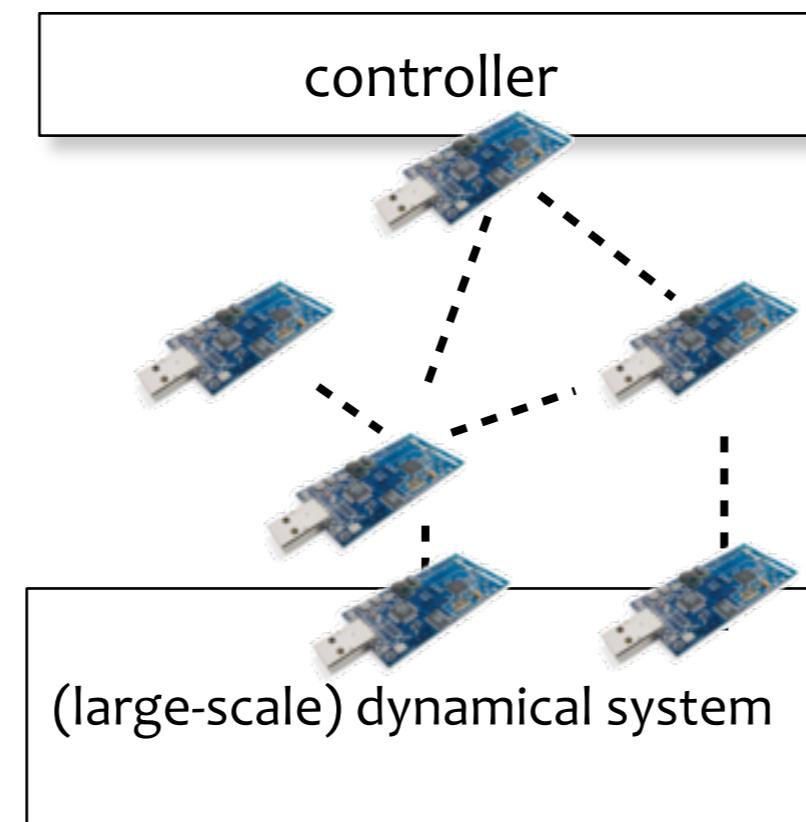
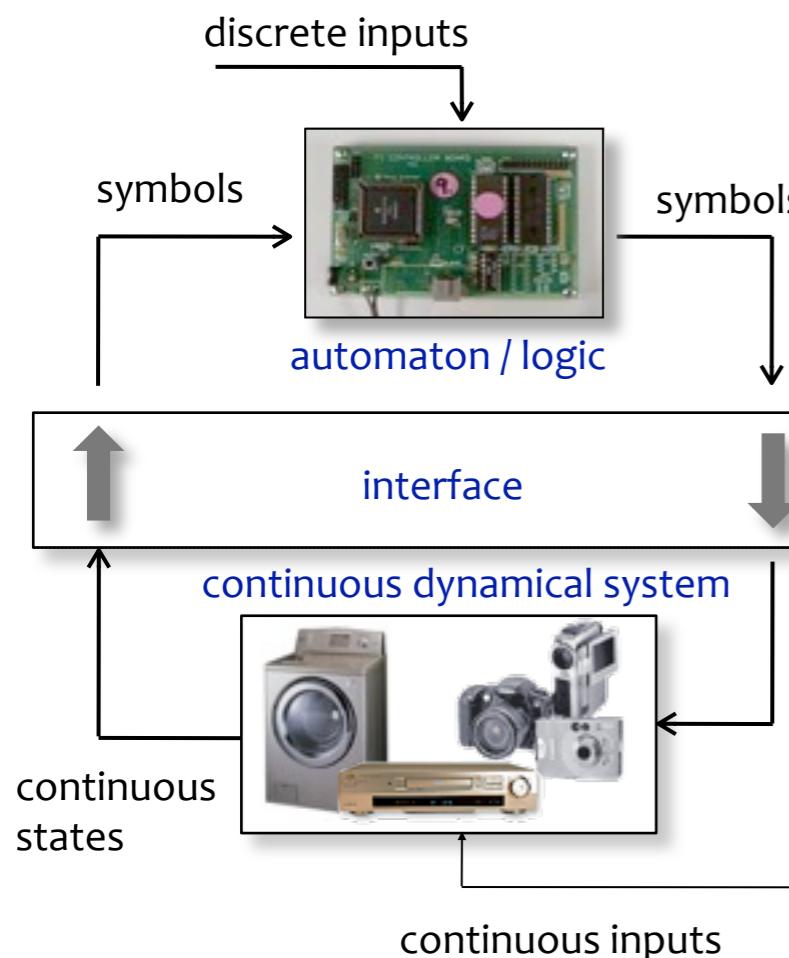
continuous
dynamical
system

hybrid
dynamical
system

- Variables are **discrete-valued**
 $x \in \{0, 1\}^{n_b}, u \in \{0, 1\}^{m_b}$
- Dynamics = **finite state machine**
- **Logic** constraints

- Variables are **real-valued**
 $x \in \mathbb{R}^{n_c}, u \in \mathbb{R}^{m_c}$
- **Difference/differential equations**
- **Linear inequality** constraints

TECHNOLOGICAL PUSH FOR STUDYING HYBRID SYSTEMS



embedded
systems

networked
control systems

cyber-physical
systems

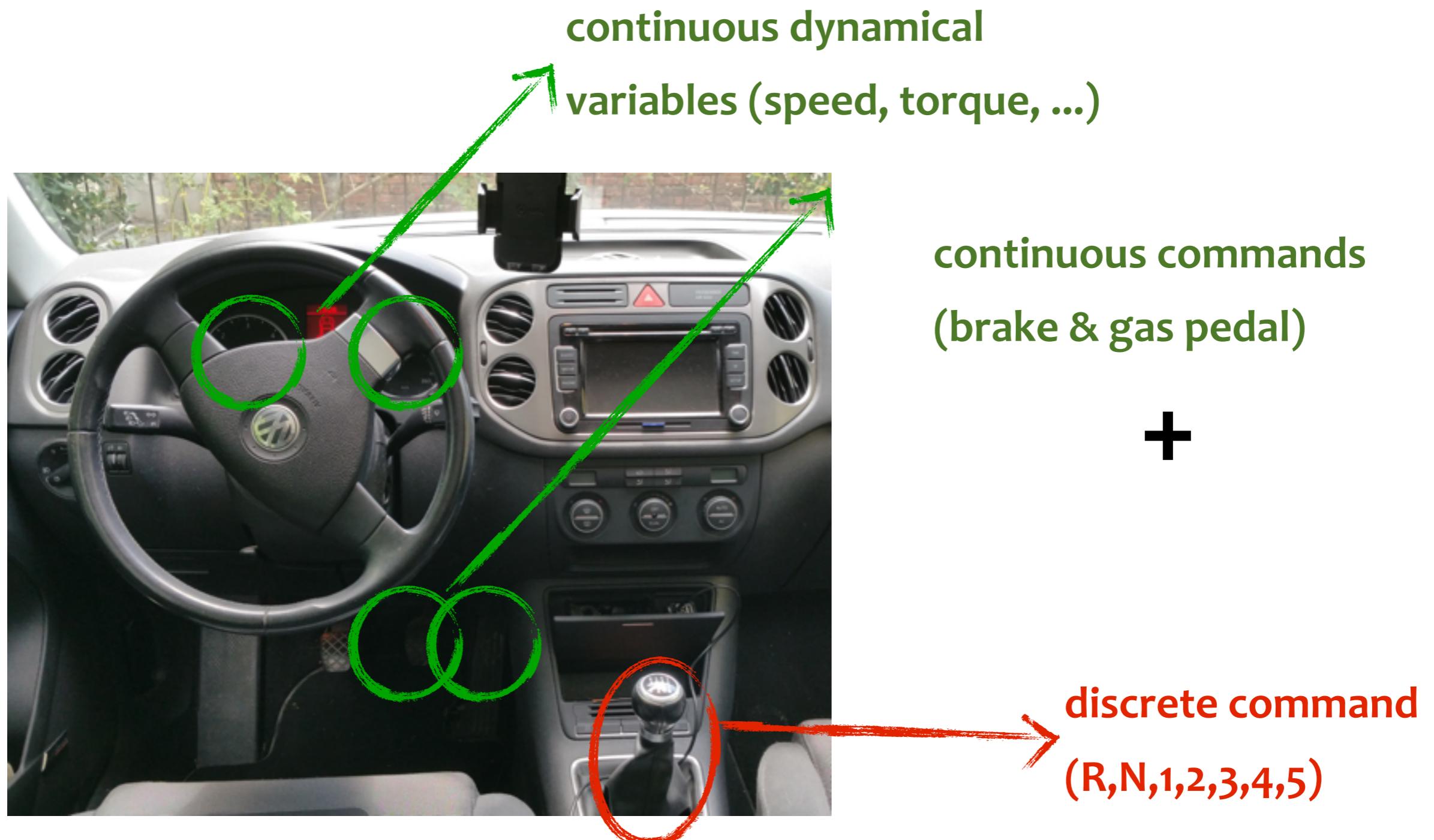
>1995

>2005

>2010

AN EXAMPLE OF “INTRINSICALLY HYBRID” SYSTEM

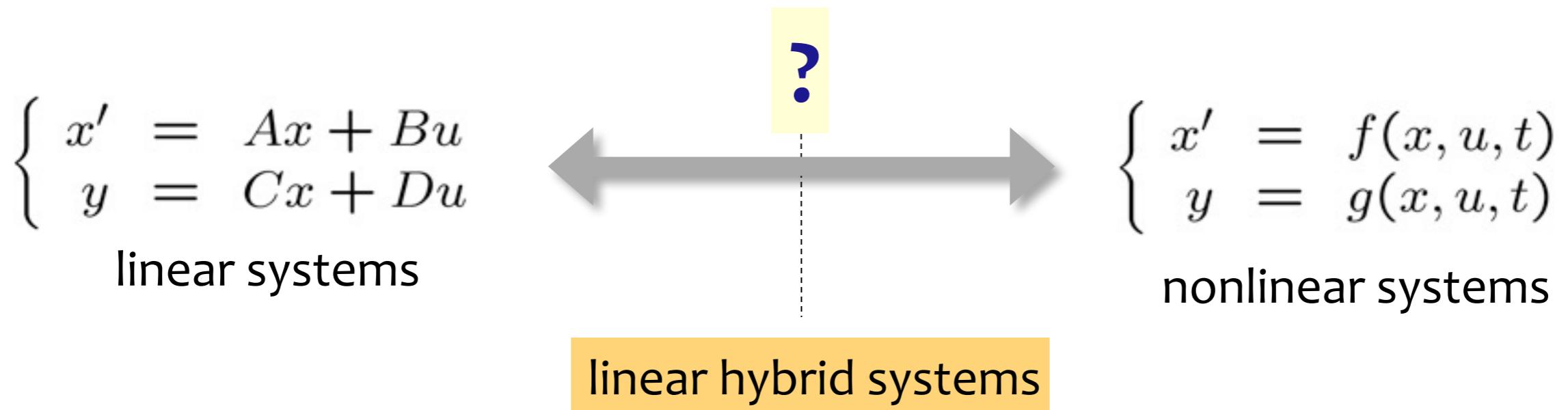
- Gear-shift



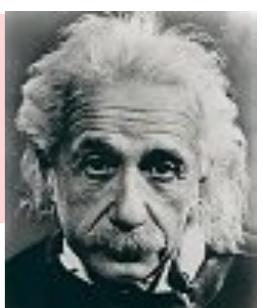
KEY REQUIREMENTS FOR HYBRID MODELS

- **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



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

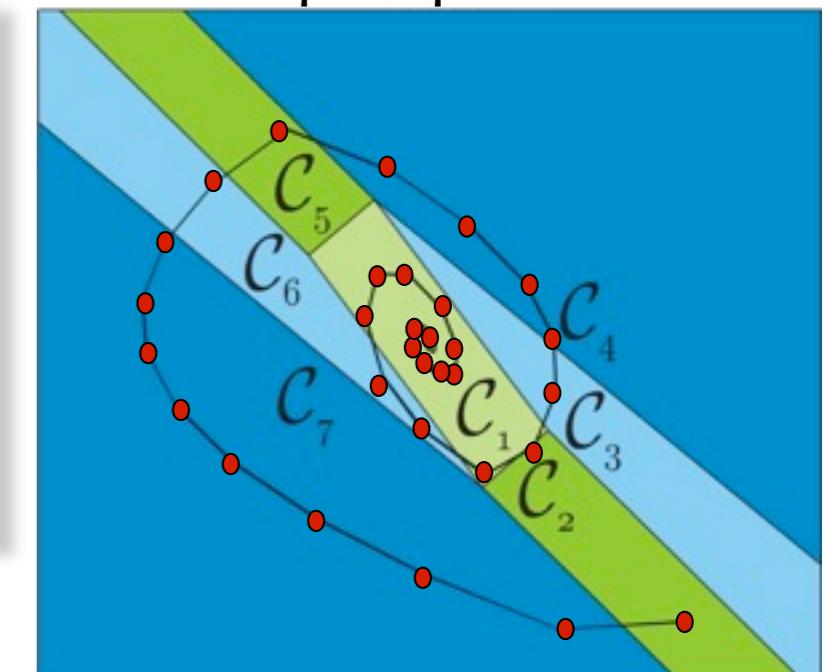


PIECEWISE AFFINE SYSTEMS

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

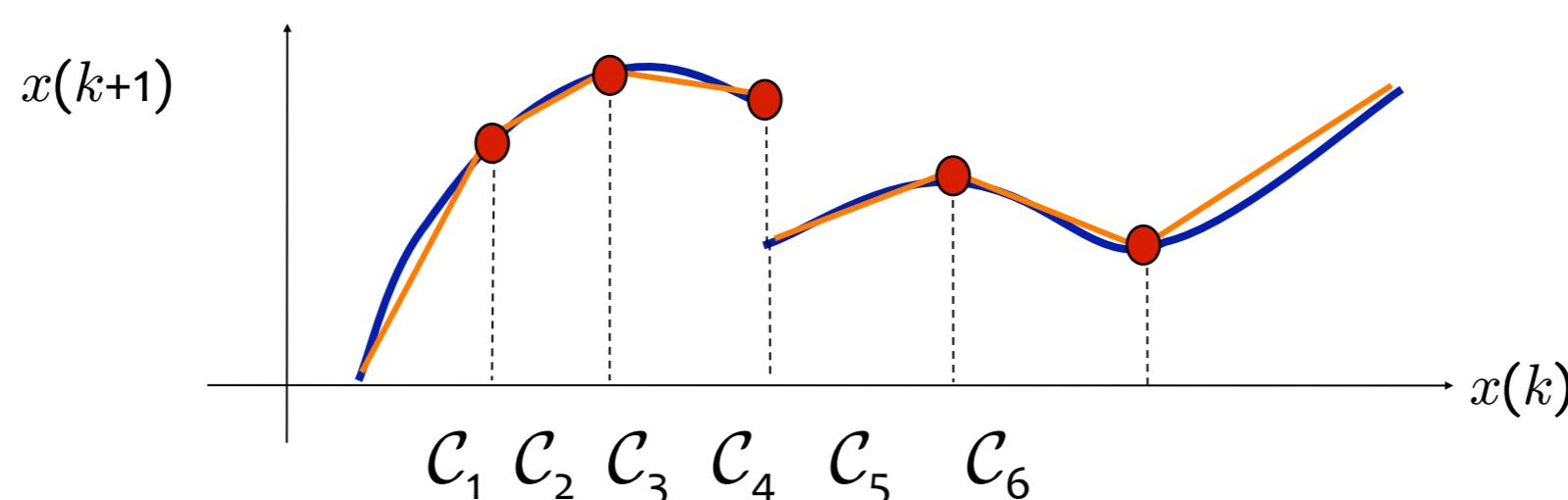
$$x \in \mathbb{R}^n, u \in \mathbb{R}^m, y \in \mathbb{R}^p$$
$$i(k) \in \{1, \dots, s\}$$

state+input space



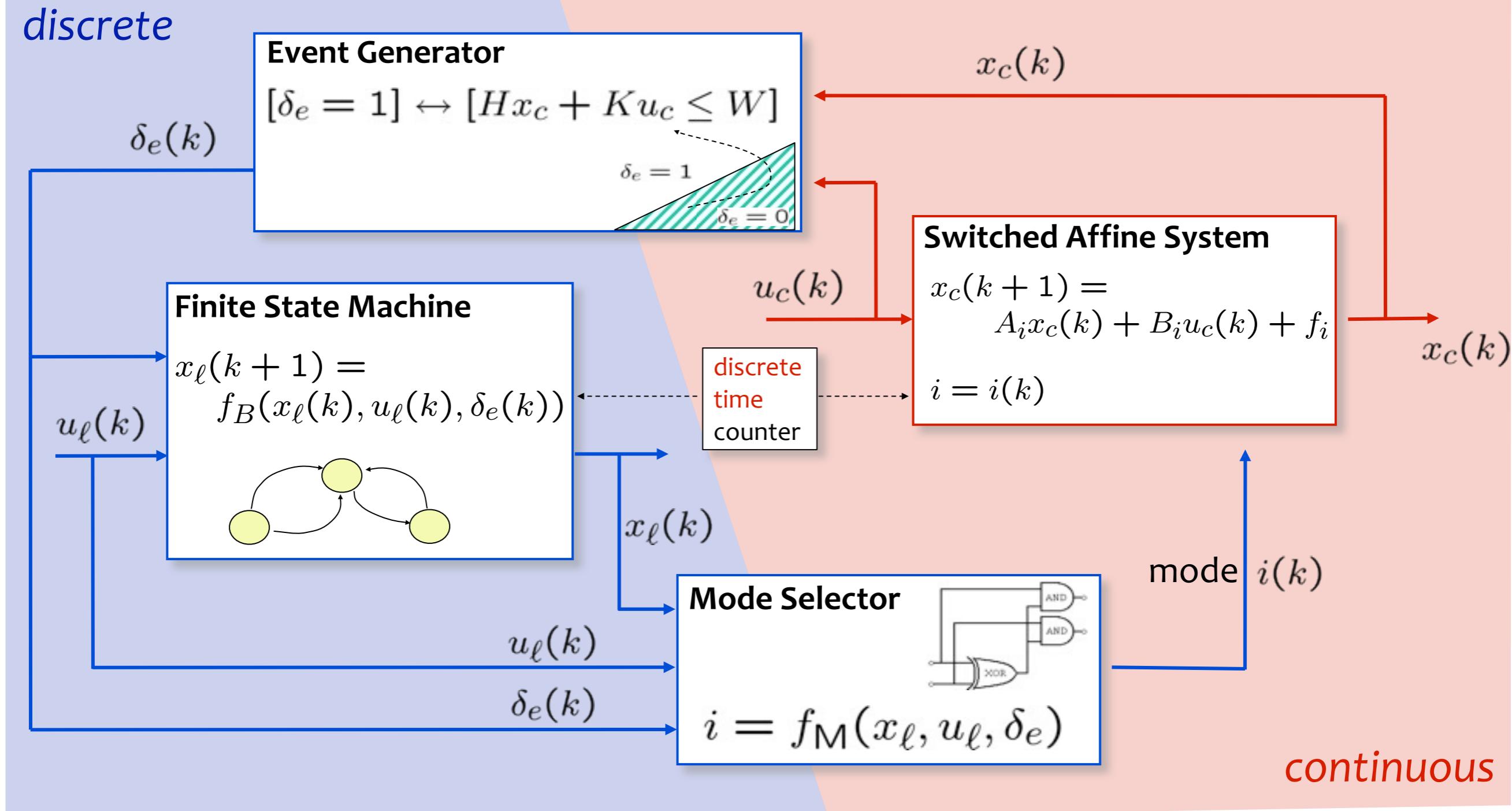
(Sontag 1981)

Can approximate nonlinear and/or discontinuous dynamics arbitrarily well



DISCRETE HYBRID AUTOMATON (DHA)

(Torrisi, Bemporad, 2004)

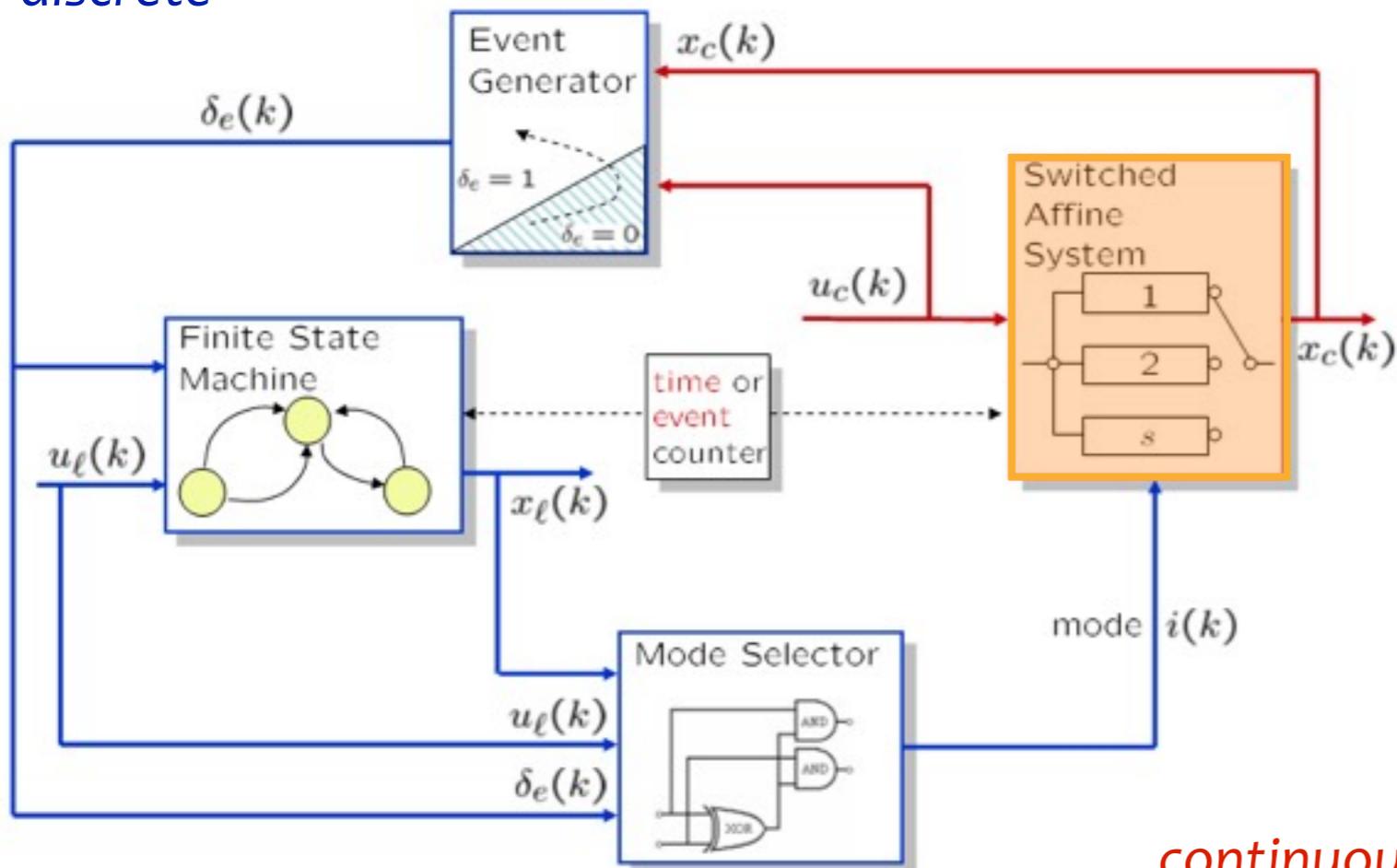


$x_\ell \in \{0, 1\}^{n_\ell}$ = binary state
 $u_\ell \in \{0, 1\}^{m_\ell}$ = binary input
 $\delta_e \in \{0, 1\}^{n_e}$ = event variable

$x_c \in \mathbb{R}^{n_c}$ = real-valued state
 $u_c \in \mathbb{R}^{m_c}$ = real-valued input
 $i \in \{1, \dots, s\}$ = current mode

SWITCHED AFFINE SYSTEM

discrete



continuous

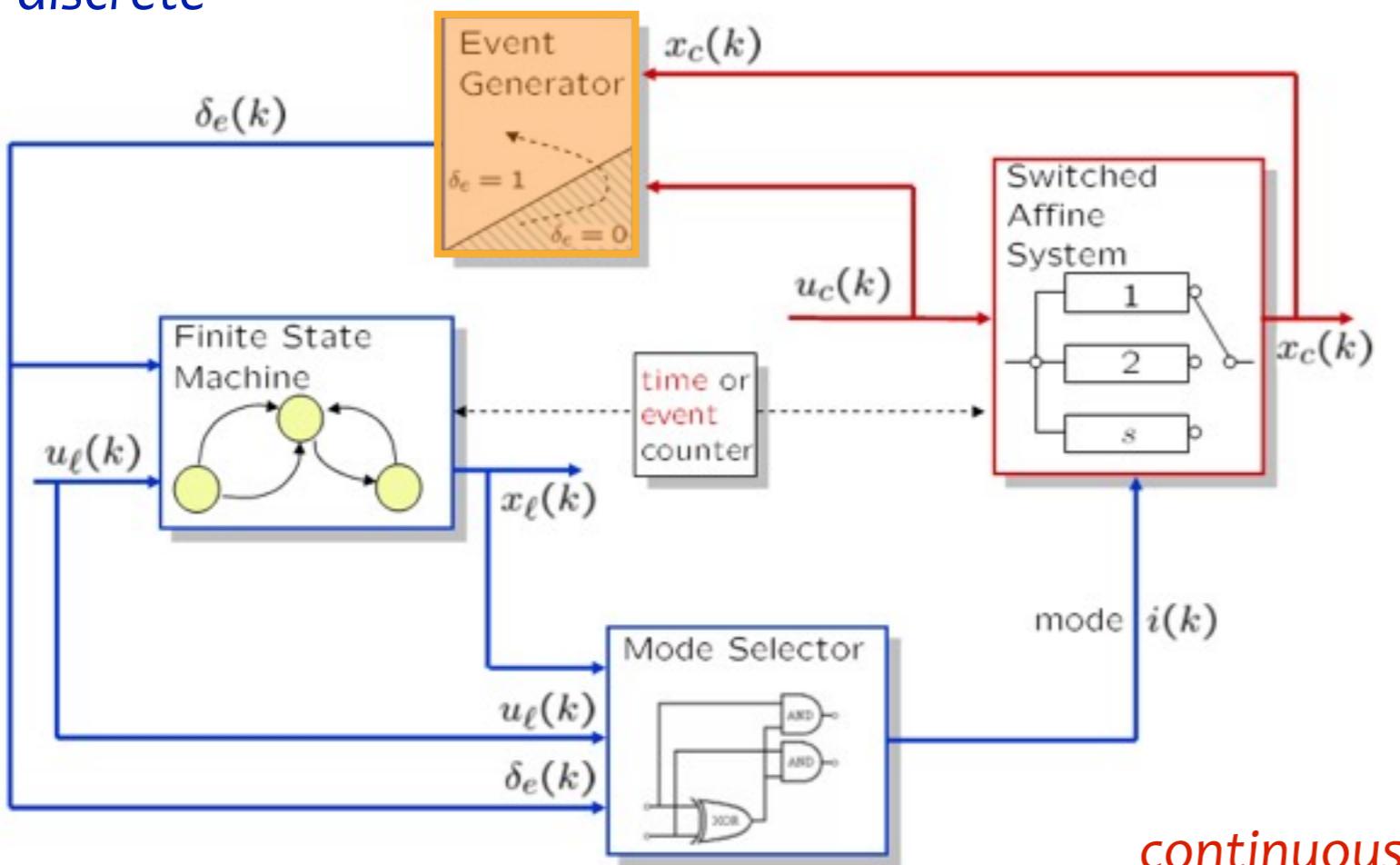
The affine dynamics depend on the current mode $i(k)$:

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

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

EVENT GENERATOR

discrete



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

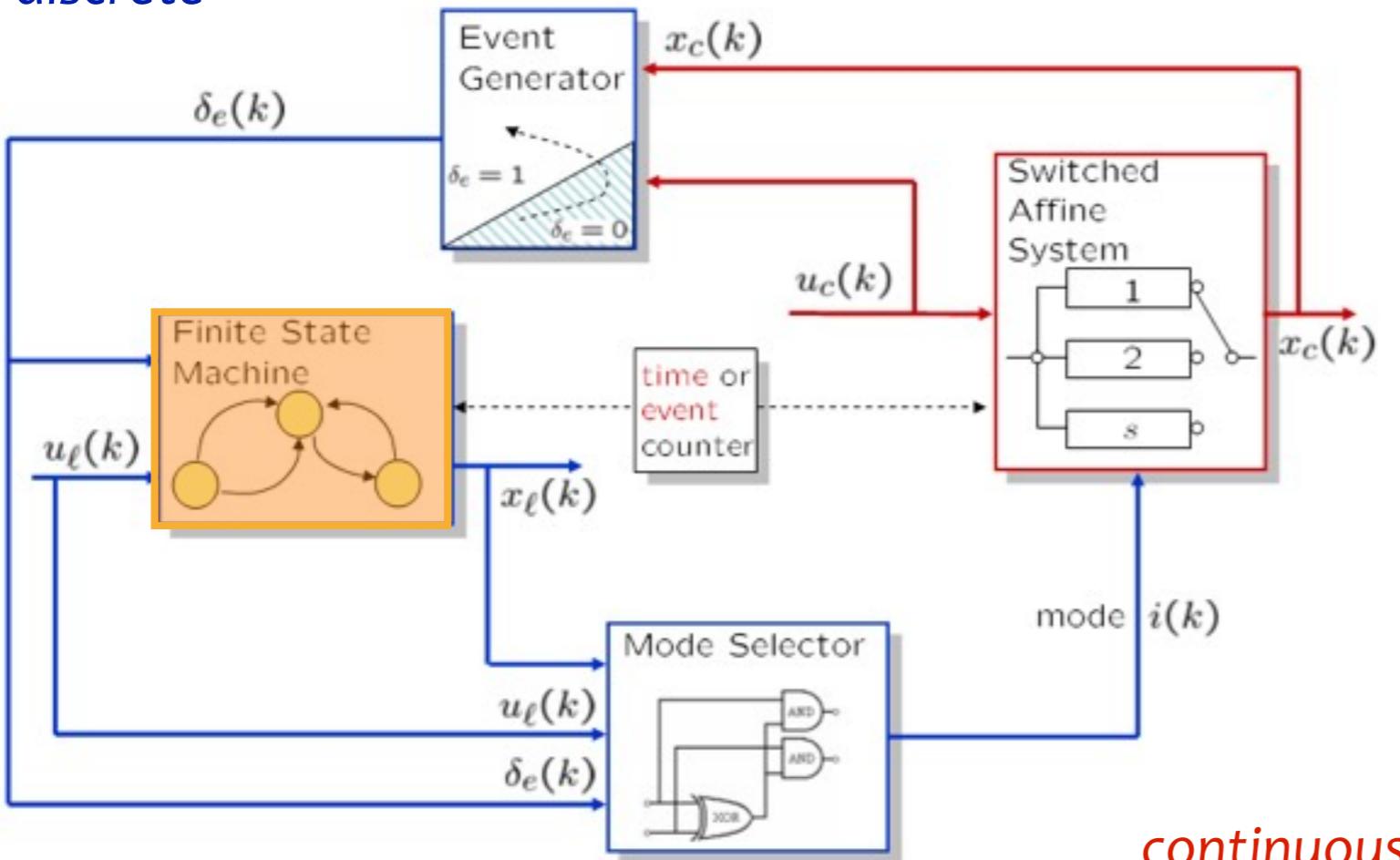
$$[\delta_e^i(k) = 1] \iff [H^i x_c(k) + K^i u_c(k) \leq W^i]$$

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

$$\text{Example: } [\delta(k) = 1] \iff [x_c(k) \geq 0]$$

FINITE STATE MACHINE

discrete



continuous

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

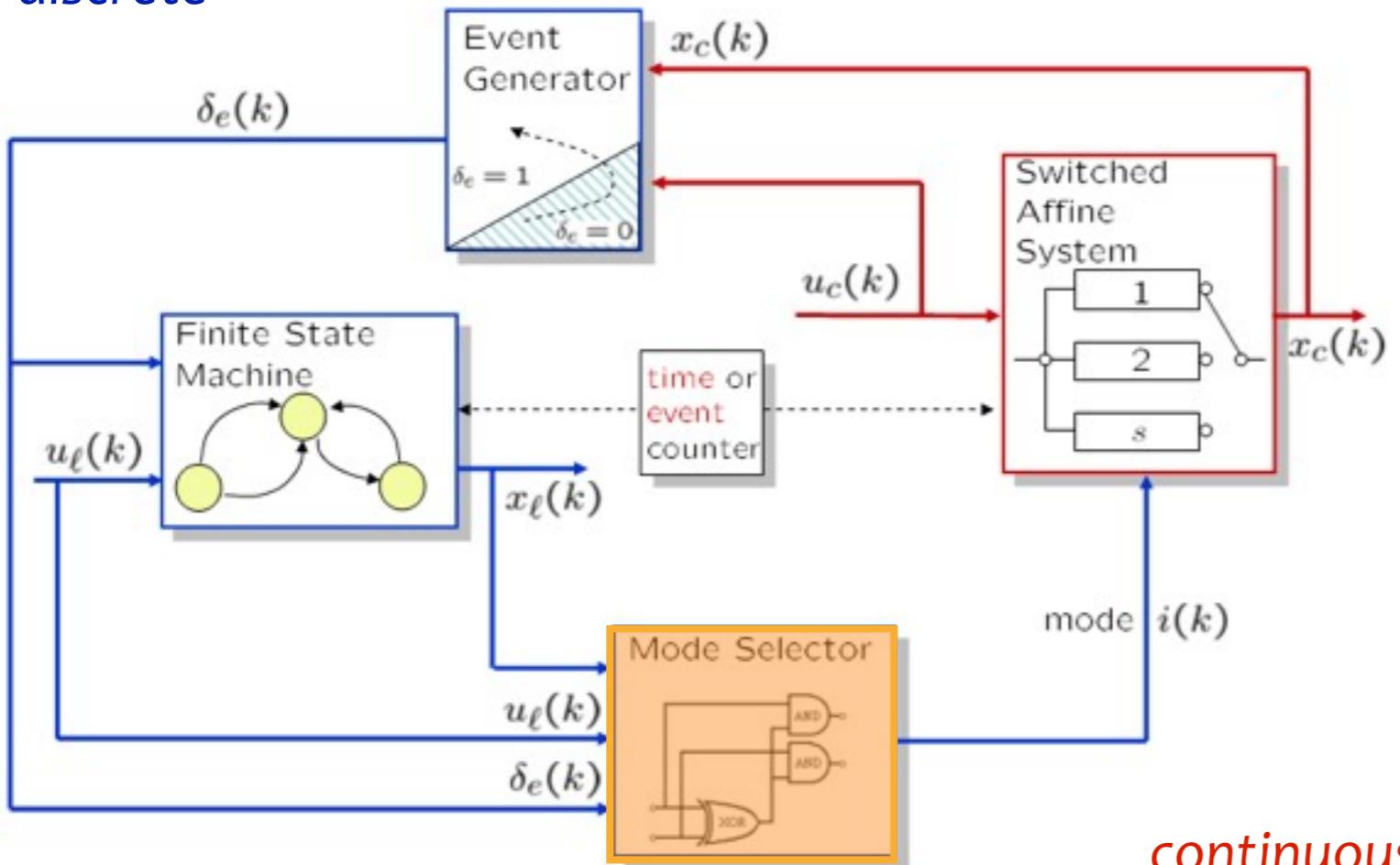
$$x_\ell(k + 1) = f_B(x_\ell(k), u_\ell(k), \delta_e(k))$$

$$x_\ell \in \{0, 1\}^{n_\ell}, \quad u_\ell \in \{0, 1\}^{m_\ell}, \quad \delta_e \in \{0, 1\}^{n_e}$$

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

MODE SELECTOR

discrete



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

continuous

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

$$i(k) = f_M(x_\ell(k), u_\ell(k), \delta_e(k))$$

$$x_\ell \in \{0, 1\}^{n_\ell}, \quad u_\ell \in \{0, 1\}^{m_\ell}, \quad \delta_e \in \{0, 1\}^{n_e}$$

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	$i = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$	$i = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$
1	$i = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$	$i = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$

the system has 3 modes

LOGIC -> INEQUALITIES

$$X_1 \vee X_2 = \text{TRUE}$$



$$\delta_1 + \delta_2 \geq 1, \quad \delta_1, \delta_2 \in \{0, 1\}$$

(Glover 1975,
Williams 1977,
Hooker 2000)

0. Given a Boolean statement

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

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:

$$\sum_{i \in P_1} \delta_i + \sum_{i \in N_1} (1 - \delta_i) \geq 1$$

: : :

$$\sum_{i \in P_m} \delta_i + \sum_{i \in N_m} (1 - \delta_i) \geq 1$$

$$P_j \cup N_j \subseteq \{1, n\}$$

polyhedron

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

Any logic proposition can be translated into integer linear 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. formal.cs.utah.edu:8080/pbl/PBL.php or just google “CNF + converter” ...)

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

2. Transform into inequalities:

$$\left\{ \begin{array}{l} \delta_3 + (1 - \delta_1) + (1 - \delta_2) \geq 1 \\ \delta_1 + (1 - \delta_3) \geq 1 \\ \delta_2 + (1 - \delta_3) \geq 1 \end{array} \right.$$

LOGIC -> INEQUALITIES: GEOMETRIC APPROACH

Boolean statement

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



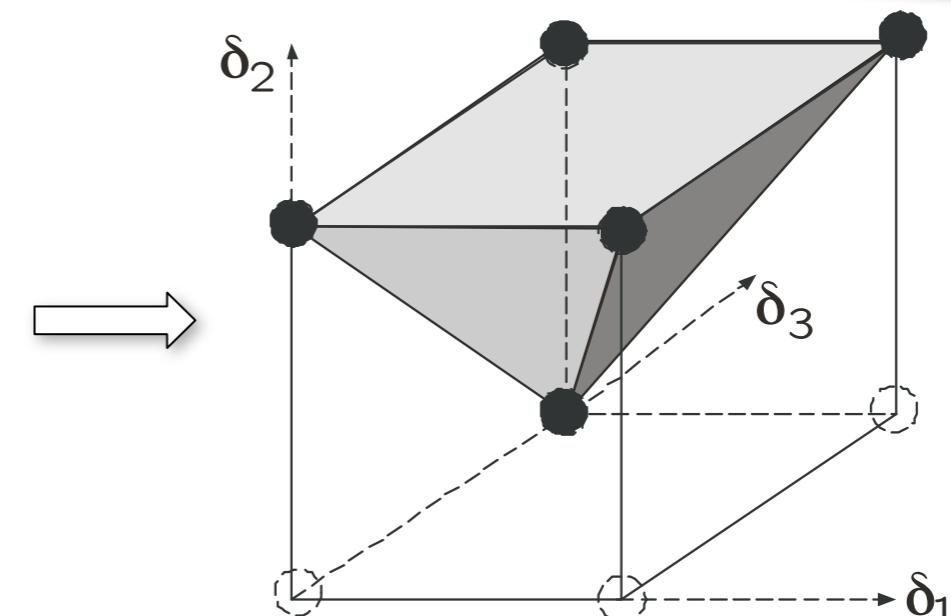
polyhedron

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

The polytope $P = \{\delta : A\delta \leq b\}$ is the convex hull of the rows of the truth table T associated with formula $F(X_1, \dots, X_N)$

$T:$

	X_1	X_2	\dots	X_N
	0	0	\dots	1
	0	1	\dots	0
\vdots	\vdots	\vdots	\vdots	\vdots
	1	1	\dots	0



Convex hull algorithms: [cdd](#), [lrs](#), [qhull](#), [chD](#), [Hull](#), [Porto](#)

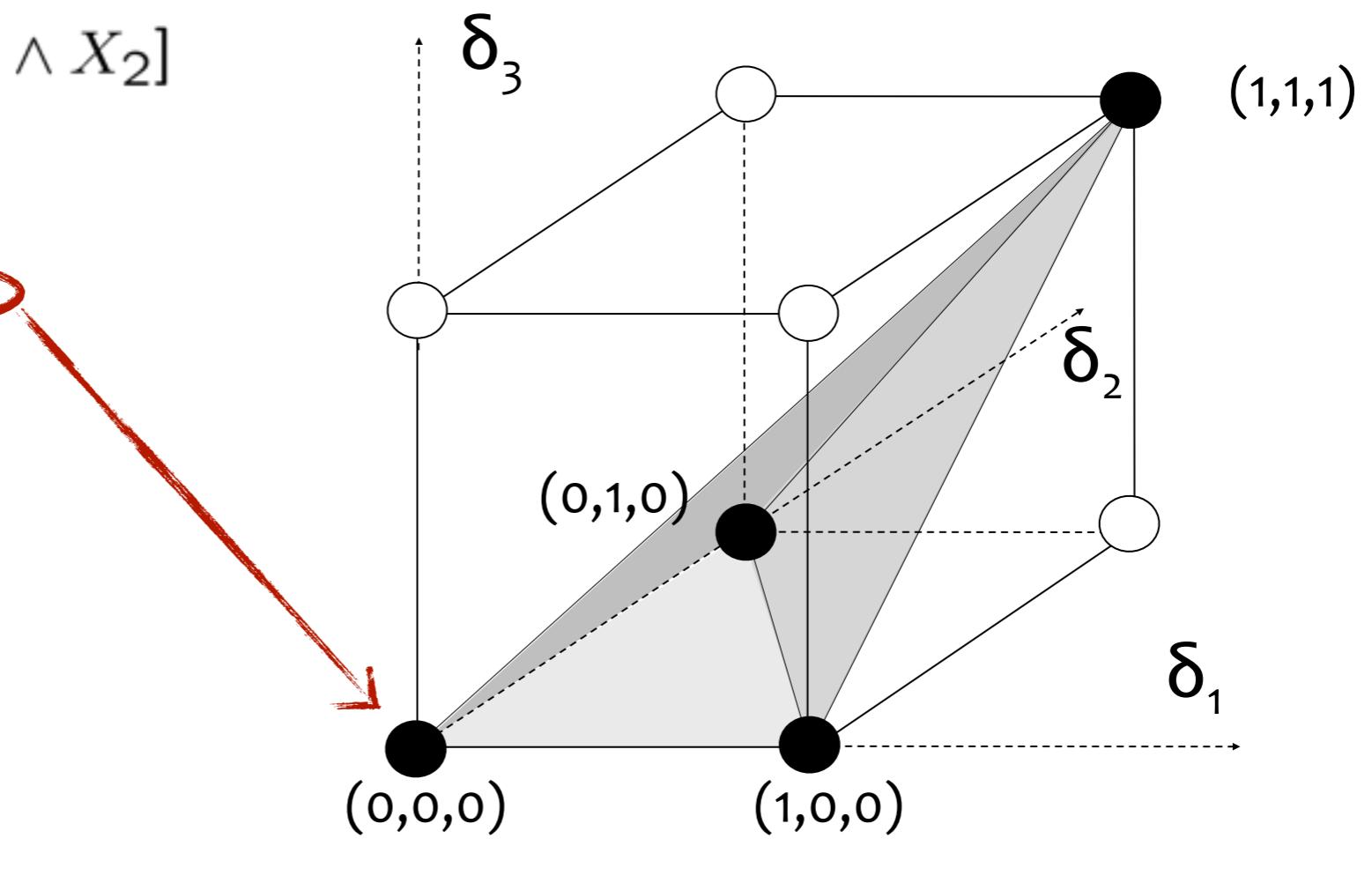
CDDMEX package included in the Hybrid Toolbox

LOGIC -> INEQUALITIES: GEOMETRIC APPROACH

Example: logic “AND”

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

X_1	X_2	X_3
0	0	0
0	1	0
1	0	0
1	1	1



Key idea:

White points cannot be
in the hull of black points

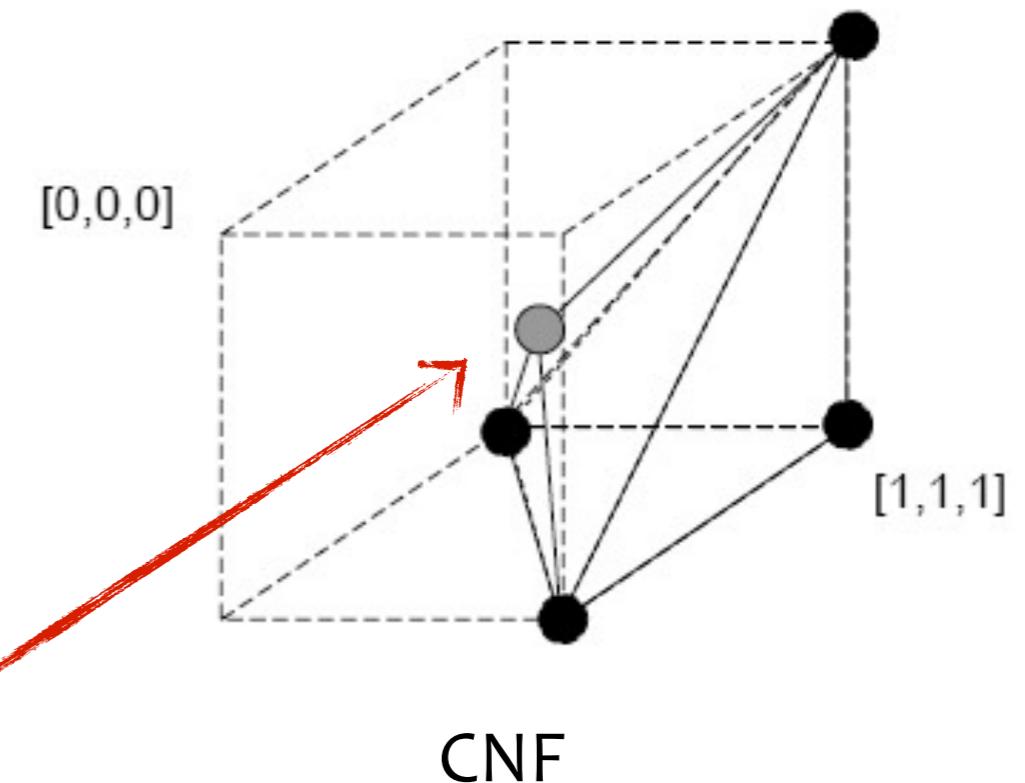
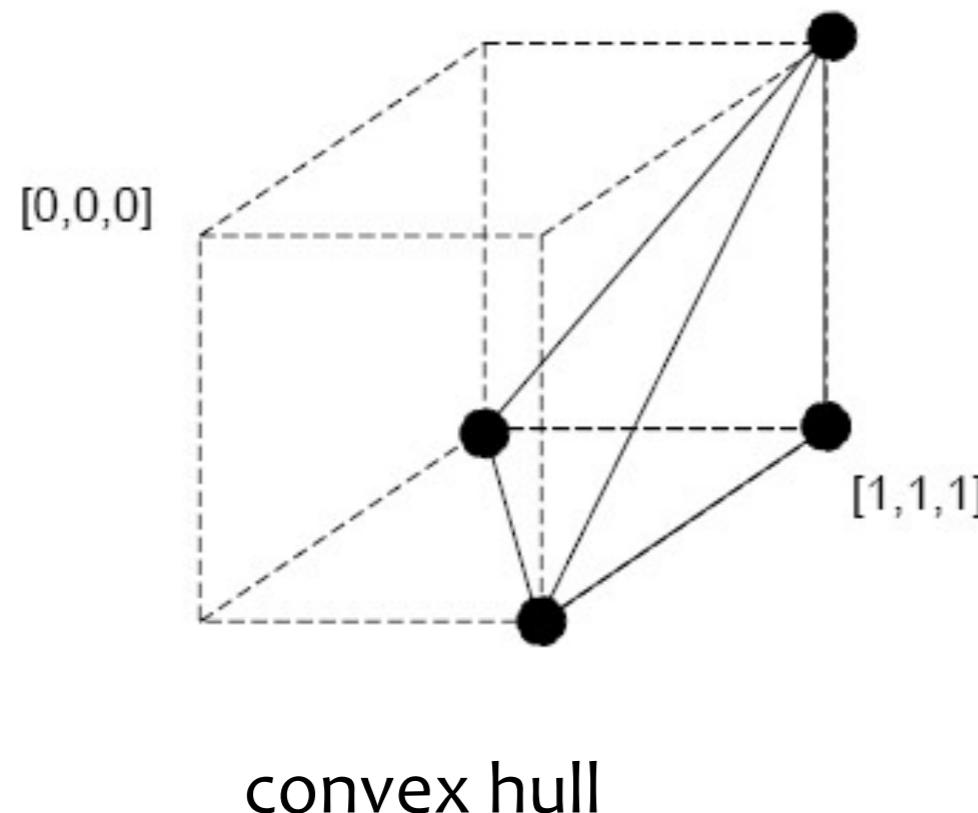
$$\text{conv} \left(\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \right) = \left\{ \delta \in \mathbb{R}^3 : \begin{array}{l} -\delta_1 + \delta_3 \leq 0 \\ -\delta_2 + \delta_3 \leq 0 \\ \delta_1 + \delta_2 - \delta_3 \leq 1 \end{array} \right\}$$

```
>> V=struct('v',[0 0 0;0 1 0;1 0 0;1 1 1]);
>> H=cddmex('hull',V);A=H.A,b=H.B
```

GEOMETRIC VS. SYMBOLIC APPROACH

- The polyhedron obtained via convex hull is the smallest one
- The one obtained via CNF may be larger. Example:

$$(X_1 \vee X_2) \wedge (X_1 \vee X_3) \wedge (X_2 \vee X_3) = \text{TRUE}$$



spurious vertex
in $(0.5, 0.5, 0.5)$

Note: no other example with 3 vars but $(X_1 \vee X_2) \wedge (X_1 \vee X_3) \wedge (X_2 \vee X_3) \wedge (\bar{X}_1 \vee \bar{X}_2 \vee \bar{X}_3)$

BIG-M TECHNIQUE (IFF)

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

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

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

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

$$\begin{aligned}M &> a'x_c - b, \quad \forall x_c \in \mathcal{X} \\ m &< a'x_c - b, \quad \forall x_c \in \mathcal{X}\end{aligned}$$

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

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

- To avoid strict inequalities, replace by $a'x_c - b \geq \epsilon + (m - \epsilon)\delta$ where $\epsilon > 0$ is a small number (e.g., machine precision)

BIG-M TECHNIQUE (IF-THEN-ELSE)

- Consider the *if-then-else* condition

$$z = \begin{cases} a'_1 x_c + f_1 & \text{if } \delta = 1 \\ a'_2 x_c + f_2 & \text{otherwise} \end{cases}$$

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

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

$$M_1 > a'_1 x_c + f_1 > m_1, \quad \forall x_c \in \mathcal{X}$$

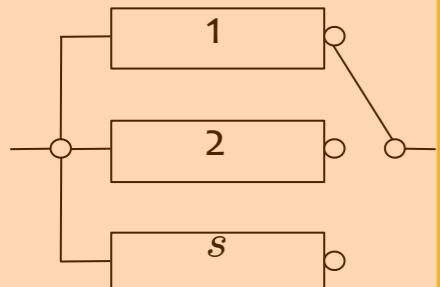
$$M_2 > a'_2 x_c + f_2 > m_2, \quad \forall x_c \in \mathcal{X}$$

- 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

Switched
Affine System



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

$$z_1(k) = \begin{cases} A_1 x_c(k) + B_1 u_c(k) + f_1 & \text{if } \delta_1(k) = 1 \\ 0 & \text{otherwise} \end{cases}$$

$$\vdots$$

$$z_s(k) = \begin{cases} A_s x_c(k) + B_s u_c(k) + f_s & \text{if } \delta_s(k) = 1 \\ 0 & \text{otherwise} \end{cases}$$

$$\sum_{i=1}^s \delta_i(k) = 1 \quad \rightarrow \quad \begin{cases} \sum_{i=1}^s \delta_i(k) \geq 1 \\ \sum_{i=1}^s \delta_i(k) \leq 1 \end{cases}$$

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

where $z_i(k) \in \mathbb{R}^{n_c}$, $\delta_i(k) \in \{0, 1\}$, $i = 1, \dots, s$

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

LOGIC AND INEQUALITIES

$$X_1 \vee X_2 = \text{TRUE}$$



$$\delta_1 + \delta_2 \geq 1, \quad \delta_1, \delta_2 \in \{0, 1\}$$

(Glover 1975, Williams 1977,
Hooker 2000)

Any logic statement

$$f(X) = \text{TRUE}$$

$$\bigwedge_{j=1}^m \left(\bigvee_{i \in P_j} X_i \vee \bigvee_{i \in N_j} \neg X_i \right)$$

$N_j, P_j \subseteq \{1, \dots, n\}$

(CNF)

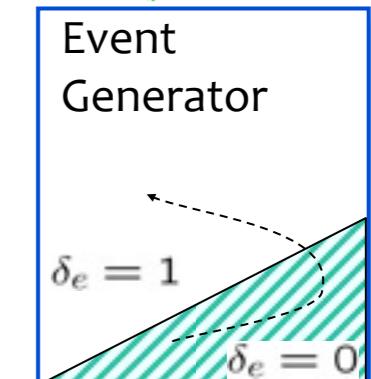
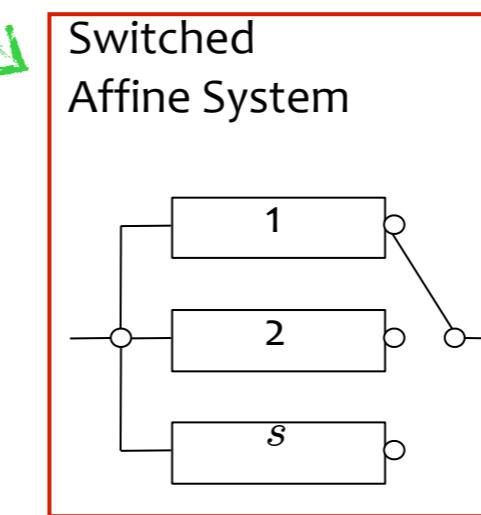
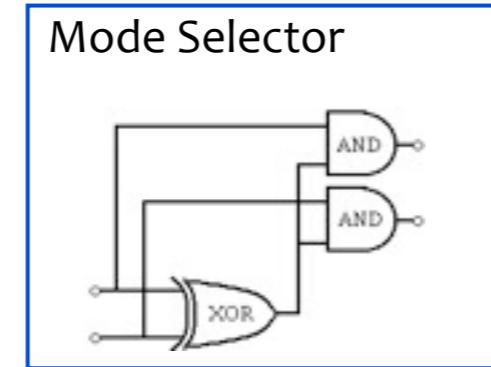
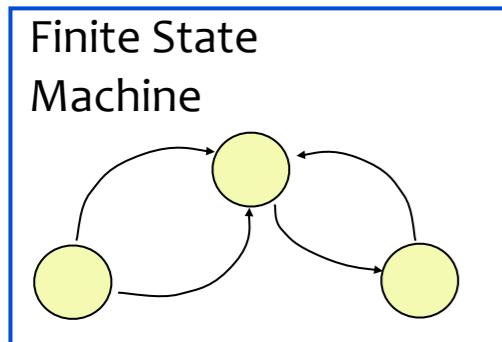
$$\left\{ \begin{array}{l} 1 \leq \sum_{i \in P_1} \delta_i + \sum_{i \in N_1} (1 - \delta_i) \\ \vdots \\ 1 \leq \sum_{i \in P_m} \delta_i + \sum_{i \in N_m} (1 - \delta_i) \end{array} \right.$$

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

$$\left\{ \begin{array}{l} H^i x_c(k) - W^i \leq M^i (1 - \delta_e^i) \\ H^i x_c(k) - W^i > m^i \delta_e^i \end{array} \right.$$

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

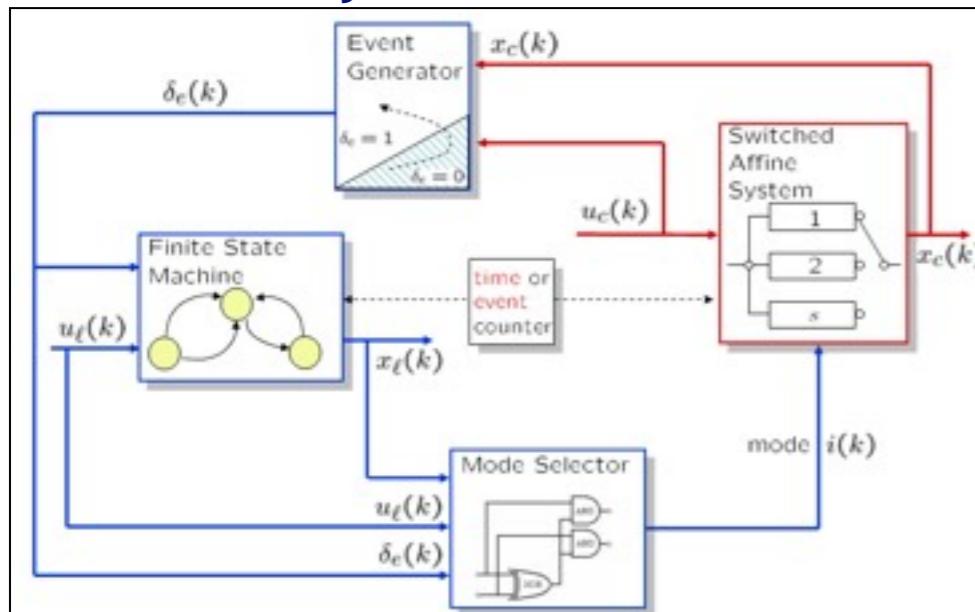
$$\left\{ \begin{array}{l} (m_1 - M_2)(1 - \delta) + z \leq a_1 x + b_1 u + f_1 \\ (m_2 - M_1)(1 - \delta) - z \leq -a_1 x - b_1 u - f_1 \\ (m_2 - M_1)\delta + z \leq a_2 x + b_2 u + f_2 \\ (m_1 - M_2)\delta - z \leq -a_2 x - b_2 u - f_2 \end{array} \right.$$



MIXED LOGICAL DYNAMICAL (MLD) SYSTEMS

(Bemporad, Morari 1999)

Discrete Hybrid Automaton



HYSDEL

(Torrisi, Bemporad, 2004)

convert logic propositions to
mixed-integer linear inequalities

Mixed Logical Dynamical (MLD) systems

$$\begin{cases} x_{k+1} = Ax_k + B_1u_k + B_2\delta_k + B_3z_k + B_5 \\ y_k = Cx_k + D_1u_k + D_2\delta_k + D_3z_k + D_5 \\ E_2\delta_k + E_3z_k \leq E_4x_k + E_1u_k + E_5 \end{cases}$$

Continuous and binary variables $x \in \mathbb{R}^{n_c} \times \{0, 1\}^{n_b}$, $u \in \mathbb{R}^{m_c} \times \{0, 1\}^{m_b}$
 $y \in \mathbb{R}^{p_c} \times \{0, 1\}^{p_b}$, $\delta \in \{0, 1\}^{r_b}$, $z \in \mathbb{R}^{r_c}$

- Computationally oriented model (mixed-integer programming)
- Suitable for **MPC control, verification, state estimation, fault detection,**

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

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

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

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

\nearrow

$$z(k) = \delta(k)x(k) \longrightarrow$$

- Rewrite as a linear equation

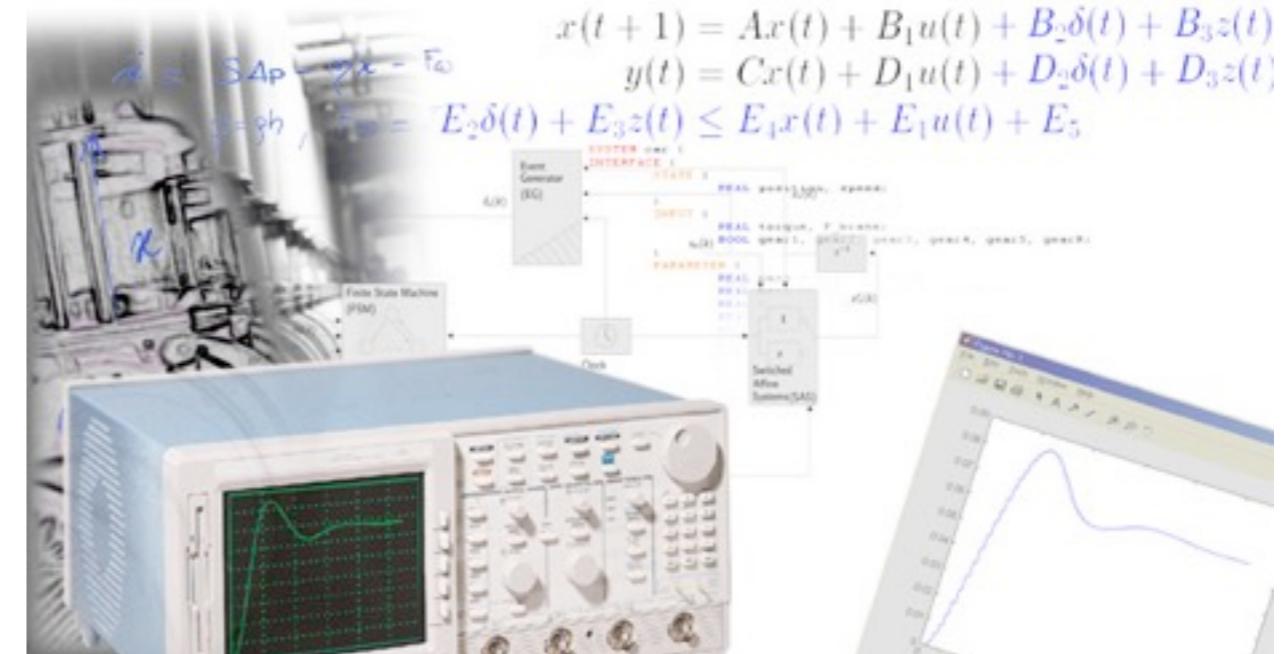
$$\longrightarrow x(k+1) = 1.6z(k) - 0.8x(k) + u(k)$$

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

(HYbrid Systems DEscription Language)

- Describe *hybrid systems*:

- Automata
- Logic
- Lin. Dynamics
- Interfaces
- Constraints



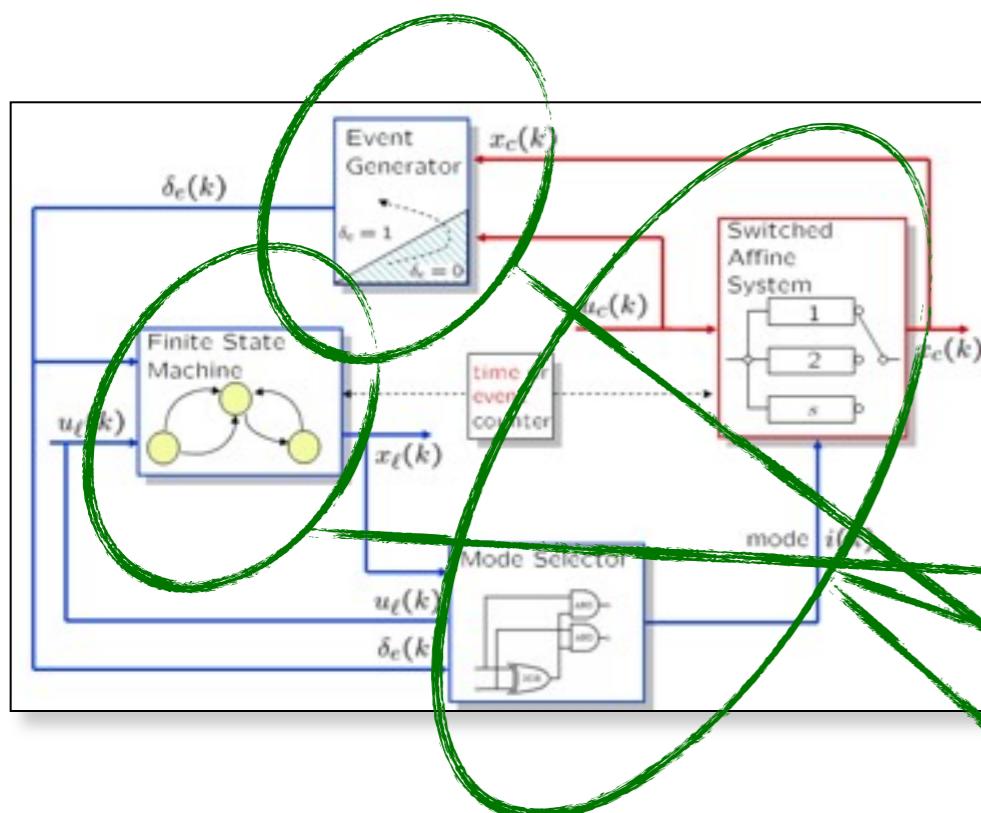
(Torrisi, Bemporad, 2004)

- Automatically generate MLD models in MATLAB

Download: <http://cse.lab.imtlucca.it/~bemporad/hybrid/toolbox>

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

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; }

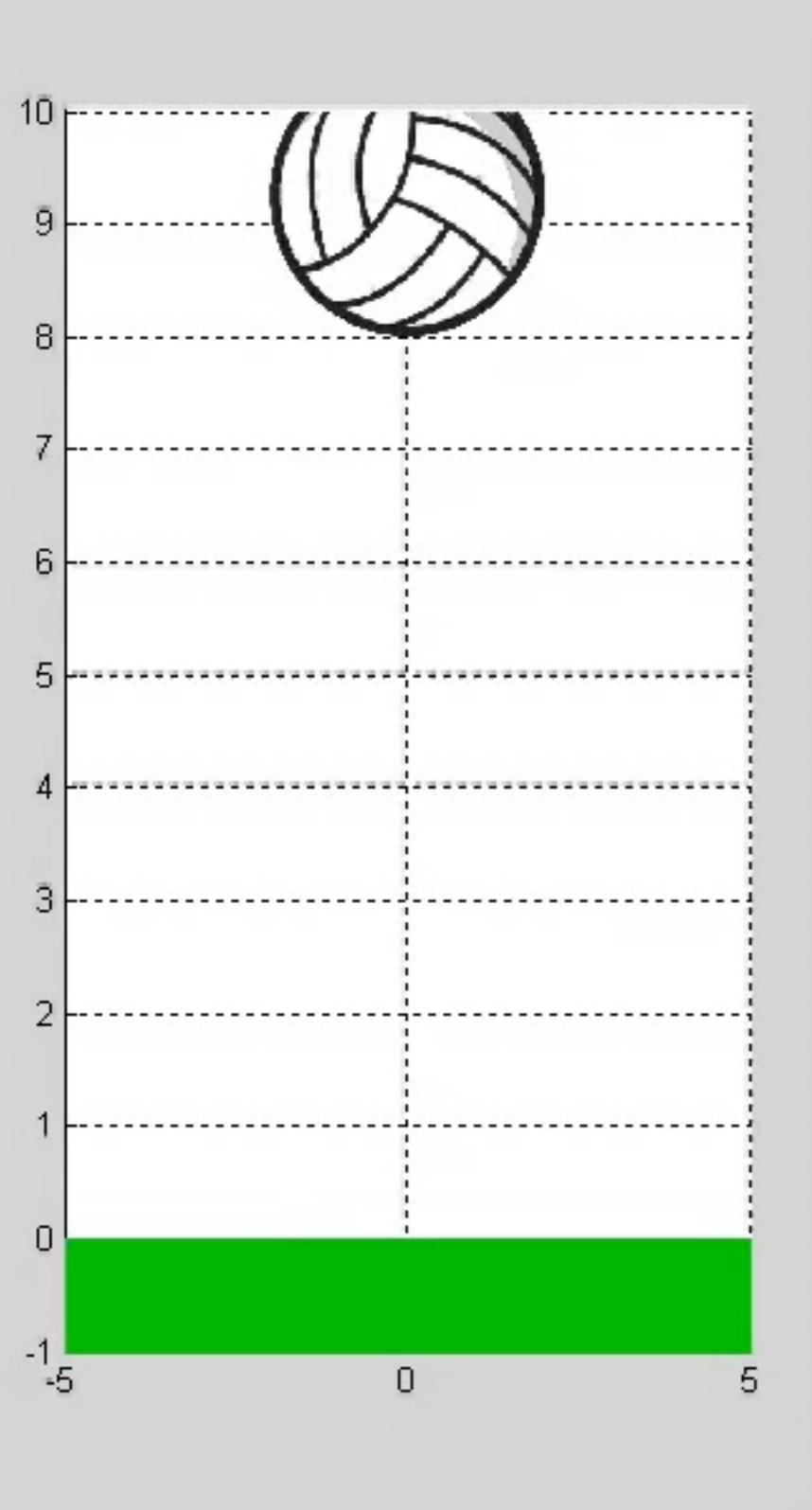
    MUST {
      xc + uc <= 2;
      ~(xl & ul); }
  } /* end implementation */

} /* end system */

```

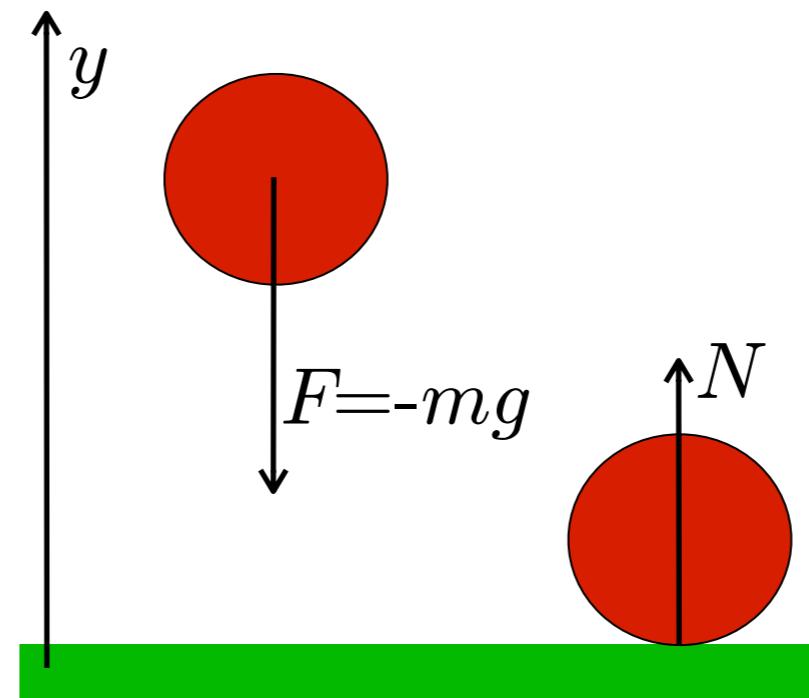
Additional relations constraining system's variables

BOUNCING BALL EXAMPLE



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

$$\alpha \in [0, 1]$$



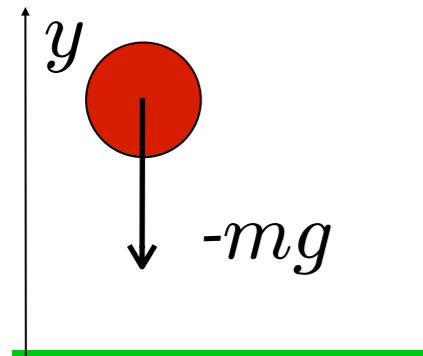
How to model the bouncing ball as a discrete-time hybrid system ?

BOUNCING BALL – TIME DISCRETIZATION

$$\underline{y(t) > 0}$$

$$v(t) \approx \frac{y(t) - y(t-1)}{T_s}$$

$$-g = \ddot{y}(t) \approx \frac{v(t) - v(t-1)}{T_s}$$

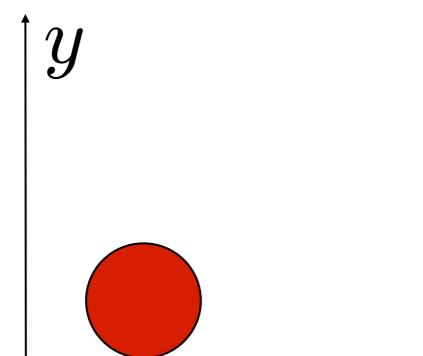


$$\begin{cases} v(t+1) &= v(t) - T_s g \\ y(t+1) &= y(t) + T_s v(t+1) \\ &= y(t) + T_s v(t) - T_s^2 g \end{cases}$$

$$\underline{y(t) \leq 0}$$

$$v(t) = -(1 - \alpha)v(t-1)$$

$$\begin{aligned} y(t+1) &= y(t-1) \\ &= y(t) - T_s v(t) \end{aligned}$$



$$\begin{cases} v(t+1) &= -(1 - \alpha)v(t) \\ y(t+1) &= y(t) - T_s v(t) \end{cases}$$

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 { BOOL negative;
      REAL hnnext;
      REAL vnnext; }

AD { negative = height <= 0; }

DA { hnnext = { IF negative THEN height-Ts*velocity
                ELSE height+Ts*velocity-Ts*Tg};
     vnnext = { IF negative THEN -(1-alpha)*velocity
                ELSE velocity-Ts*g}; }

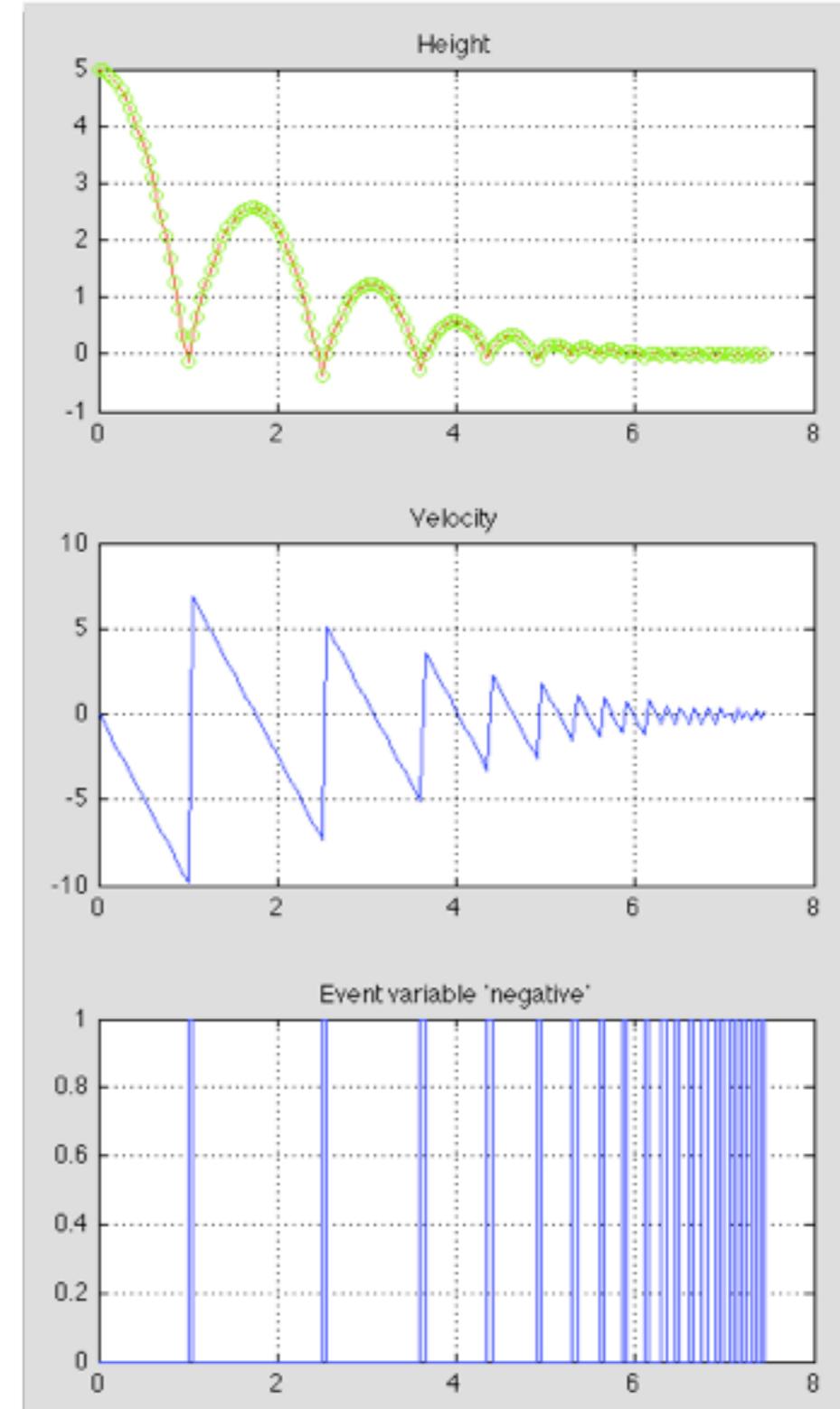
CONTINUOUS {
height = hnnext;
velocity = vnnext; }

}
}
```

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);
```

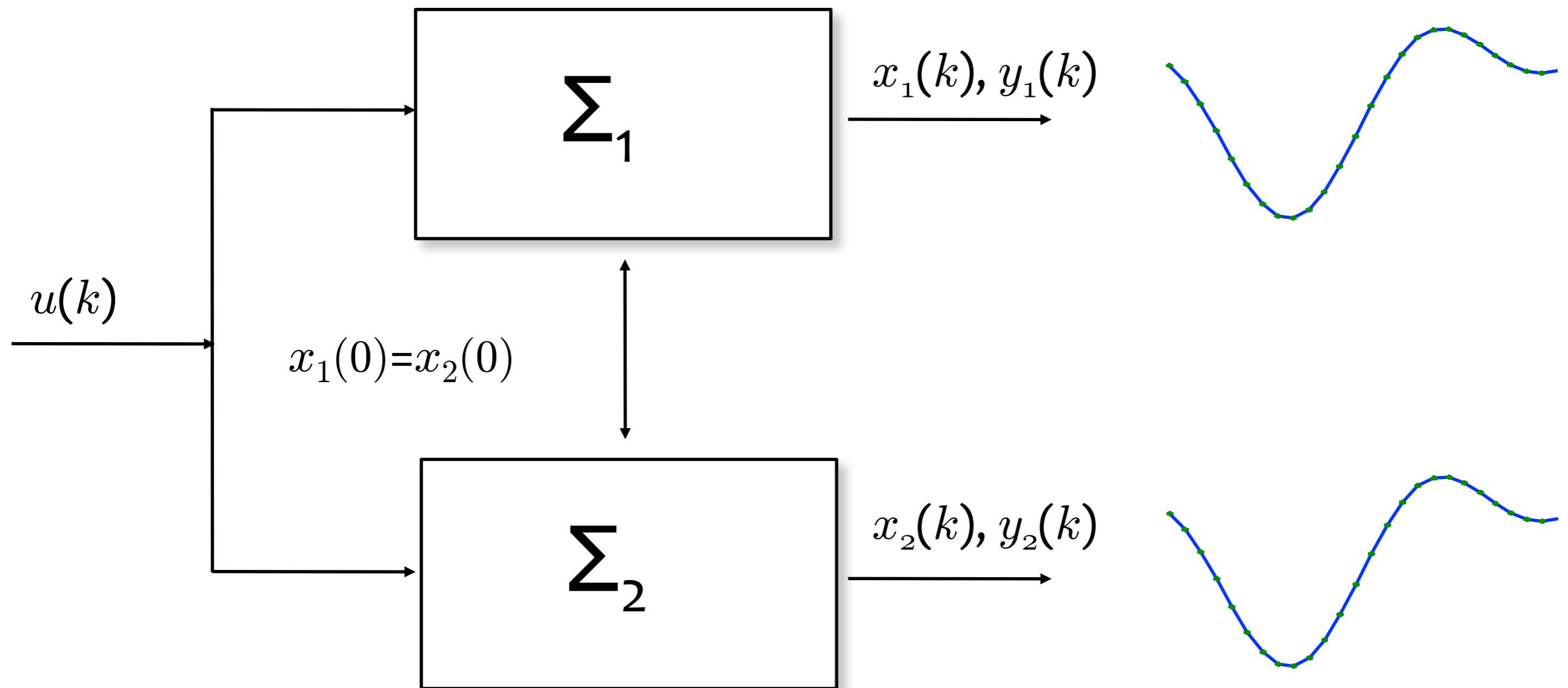


Note: no Zeno effect in discrete time !

REALIZATION AND TRANSFORMATIONS (STATE-SPACE HYBRID MODELS)

EQUIVALENCES OF HYBRID MODELS

Definition 1 Two hybrid systems Σ_1, Σ_2 are equivalent if for all initial conditions $x_1(0) = x_2(0)$ and input $\{u_1(k)\}_{k \in \mathbb{Z}_+} = \{u_2(k)\}_{k \in \mathbb{Z}_+}$ then $x_1(k) = x_2(k)$ and $y_1(k) = y_2(k)$, for all $k \in \mathbb{Z}_+$.



EQUIVALENCE OF HYBRID MODELS

- **MLD systems and PWA systems are equivalent**

(Bemporad, Ferrari-Trecate, Morari, *IEEE TAC*, 2000)

- Efficient algorithms for **converting** MLD models into PWA form exist

(Bemporad, *IEEE TAC*, 2004) (Geyer, Torrisi, Morari, *HSCC*, 2003)

- **Further equivalences** exist with other classes of hybrid dynamical systems, such as Linear Complementarity (**LC**) systems

(Heemels, De Schutter, Bemporad, *Automatica*, 2001)

MODELING A PWA SYSTEM IN MLD FORM

- Given the PWA system with bounded states and inputs
- $$\begin{aligned} x(t+1) &= A_{i(t)}x(t) + B_{i(t)}u(t) + f_{i(t)} \\ y(t) &= C_{i(t)}x(t) + D_{i(t)}u(t) + g_{i(t)} \\ i(t) &= i \in \{1, \dots, s\} : H_i x(t) + J_i u(t) \leq K_i \end{aligned}$$

$$C_i = \{[\begin{matrix} x \\ u \end{matrix}] : H_i x + J_i u \leq K_i\}$$

$$\overset{\circ}{C}_i \cap \overset{\circ}{C}_j = \emptyset, \quad \forall i \neq j \quad \text{polyhedral partition}$$

- Introduce s binary variables $\delta_i, i = 1, \dots, s$, subject to the constraints

$$\begin{aligned} [\delta_i = 1] \rightarrow [H_i x + J_i u \leq K_i] &\longrightarrow H_i x + J_i u \leq K_i + M_i(1 - \delta_i) \\ i = 1, \dots, s &\qquad\qquad\qquad i = 1, \dots, s \end{aligned}$$

$$\oplus_{i=1}^s [\delta_i = 1] = \text{TRUE} \quad \longrightarrow \quad \sum_{i=1}^s \delta_i = 1$$

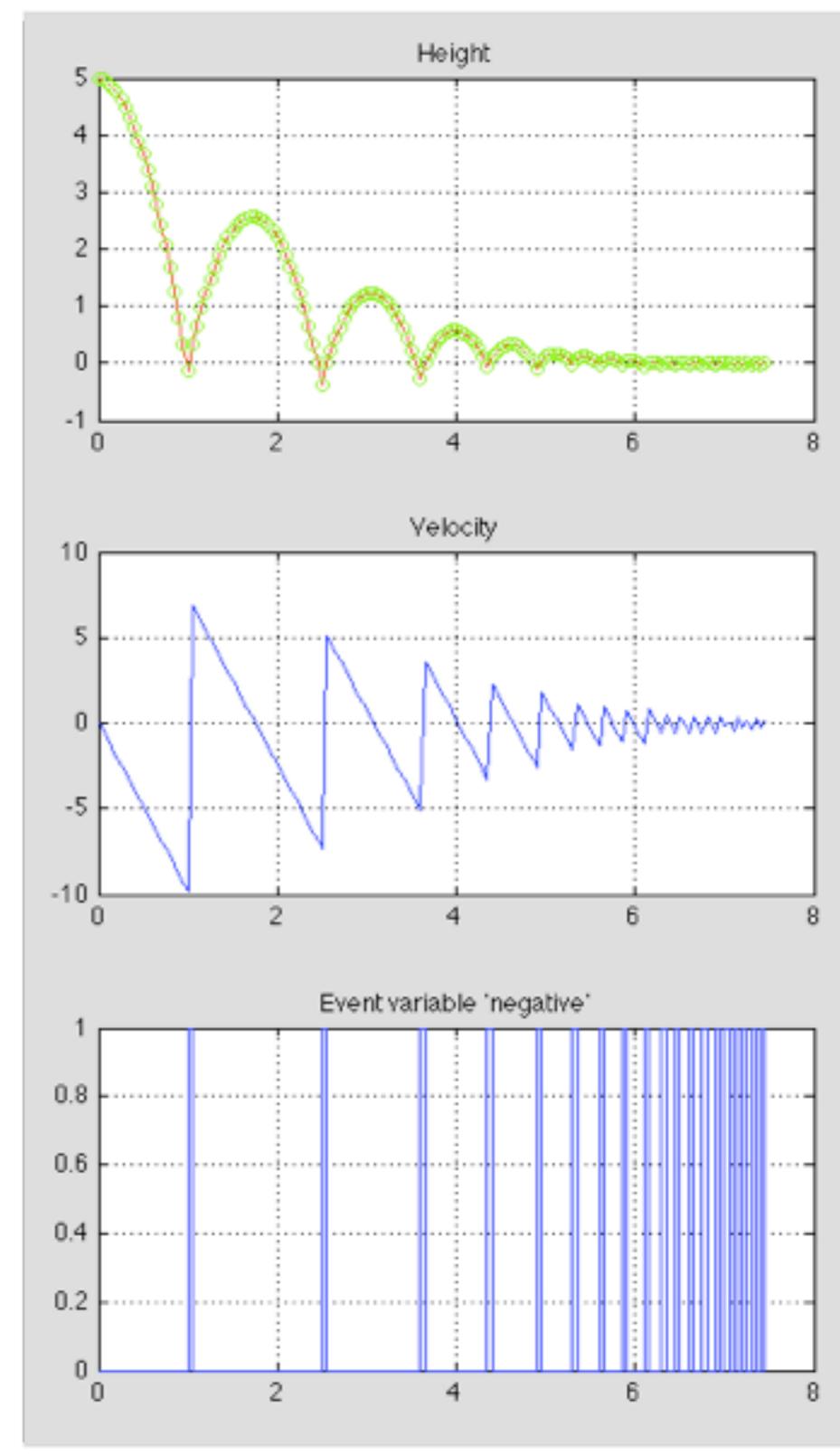
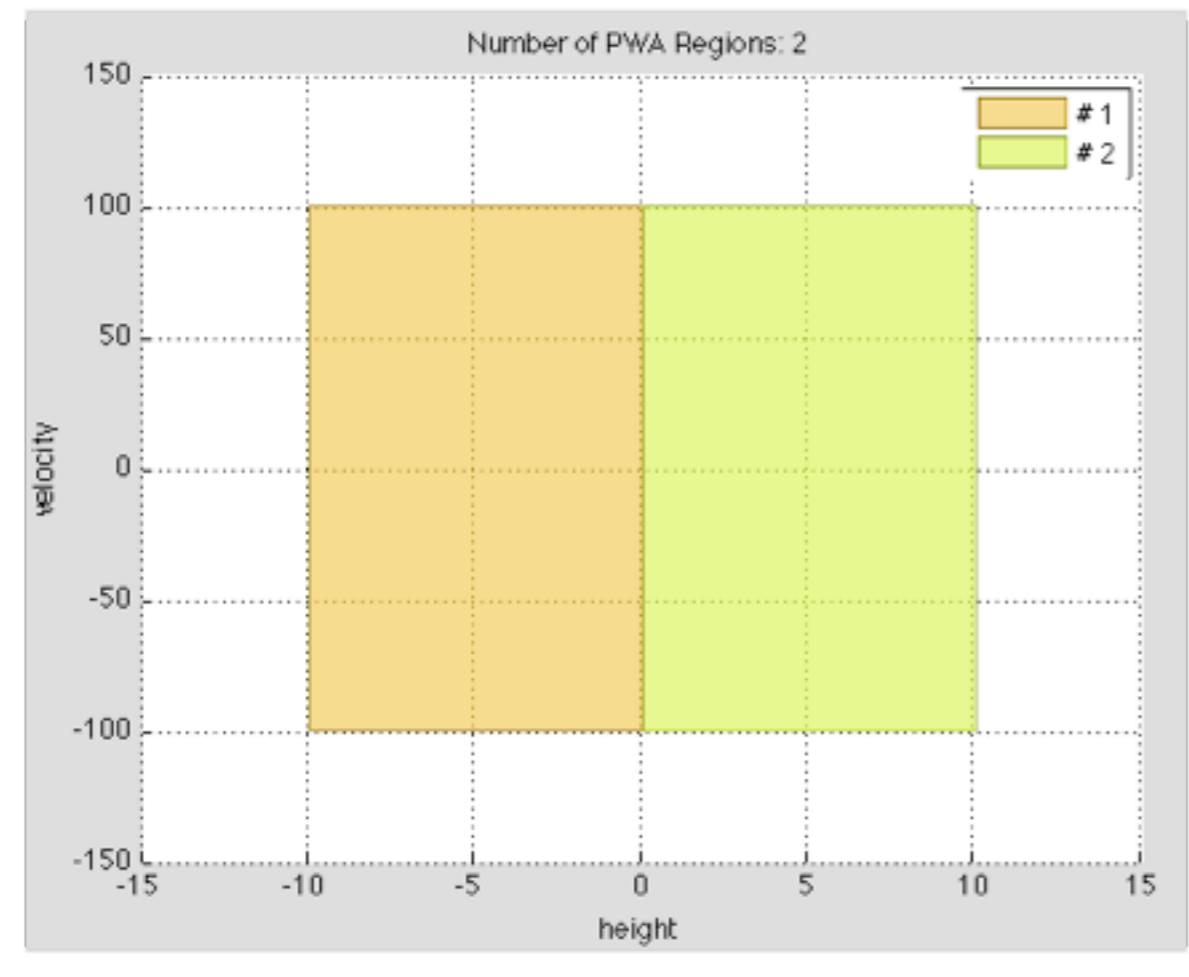
were the vector M_i of upper-bounds can be computed, e.g., via LP

- Introduce also auxiliary continuous vectors z_i, w_i defined by if-then-else rules

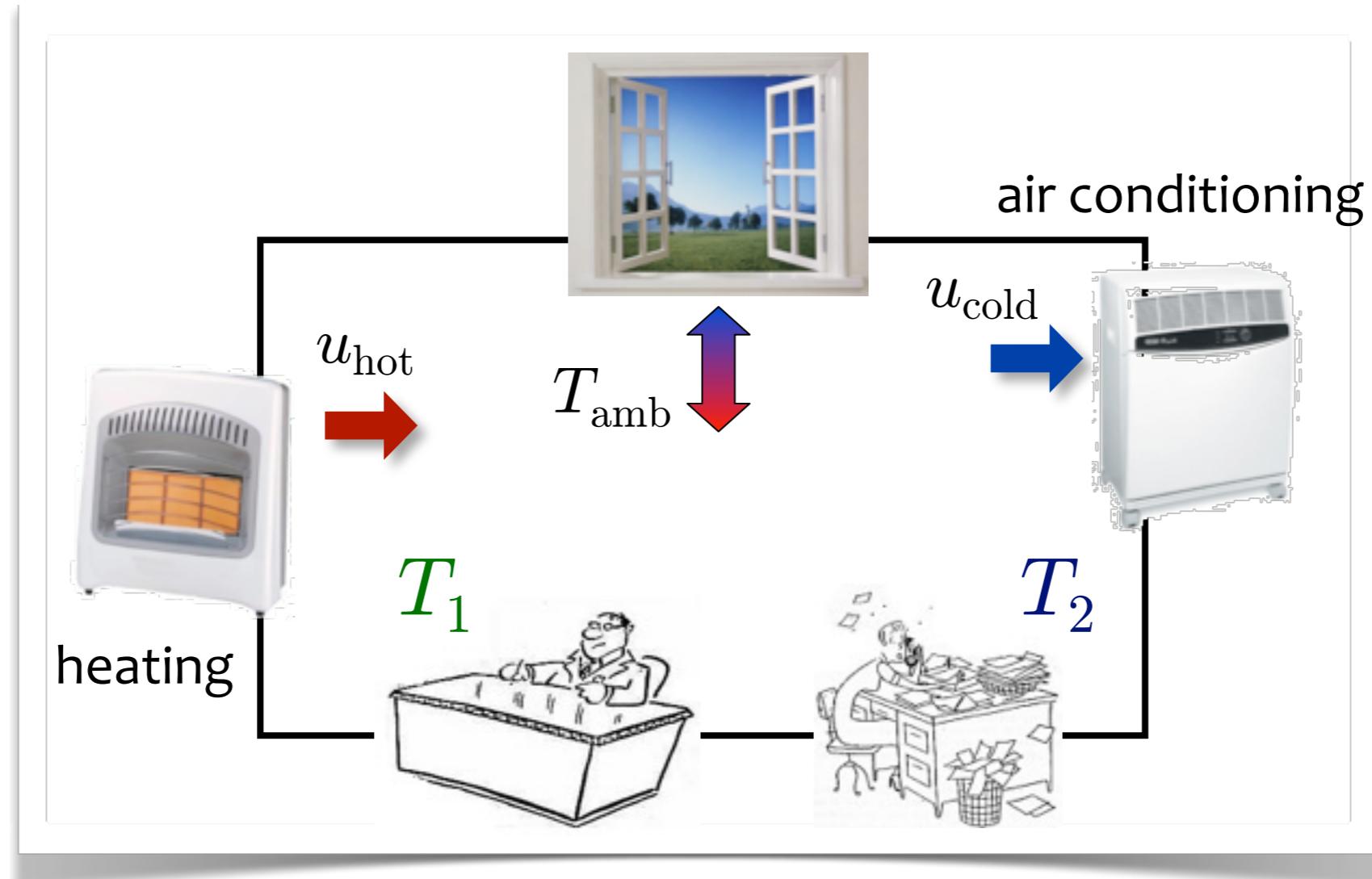
$$\begin{aligned} z_i &= A_i x + B_i u + f_i \text{ if } \delta_i = 1, \quad 0 \text{ otherwise} \\ w_i &= C_i x + D_i u + g_i \text{ if } \delta_i = 1, \quad 0 \text{ otherwise} \end{aligned} \longrightarrow x(t+1) = \sum_{i=1}^s z_i(t), \quad y(t) = \sum_{i=1}^s w_i(t)$$

BOUNCING BALL - PWA EQUIVALENT

```
>>P=pwa(S);  
>>plot(P)  
  
>>[X,T,I]=sim(P,x0,U);
```



EXAMPLE: ROOM TEMPERATURE CONTROL



discrete dynamics

- #1=cold \rightarrow heater=on
- #2=cold \rightarrow heater=on unless #1=hot
- A/C activation has similar rules

continuous dynamics

$$\frac{dT_i}{dt} = -\alpha_i(T_i - T_{\text{amb}}) + k_i(u_{\text{hot}} - u_{\text{cold}}) \quad i = 1, 2$$

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));
     }
}
}
```

```
>>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 $u_{\text{hot}}, u_{\text{cold}}$)
- 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

$$x(k+1) = A_i(k)x(k) + B_i(k)u(k) + f_i(k)$$

$$y(k) = C_i(k)x(k) + D_i(k)u(k) + g_i(k)$$

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

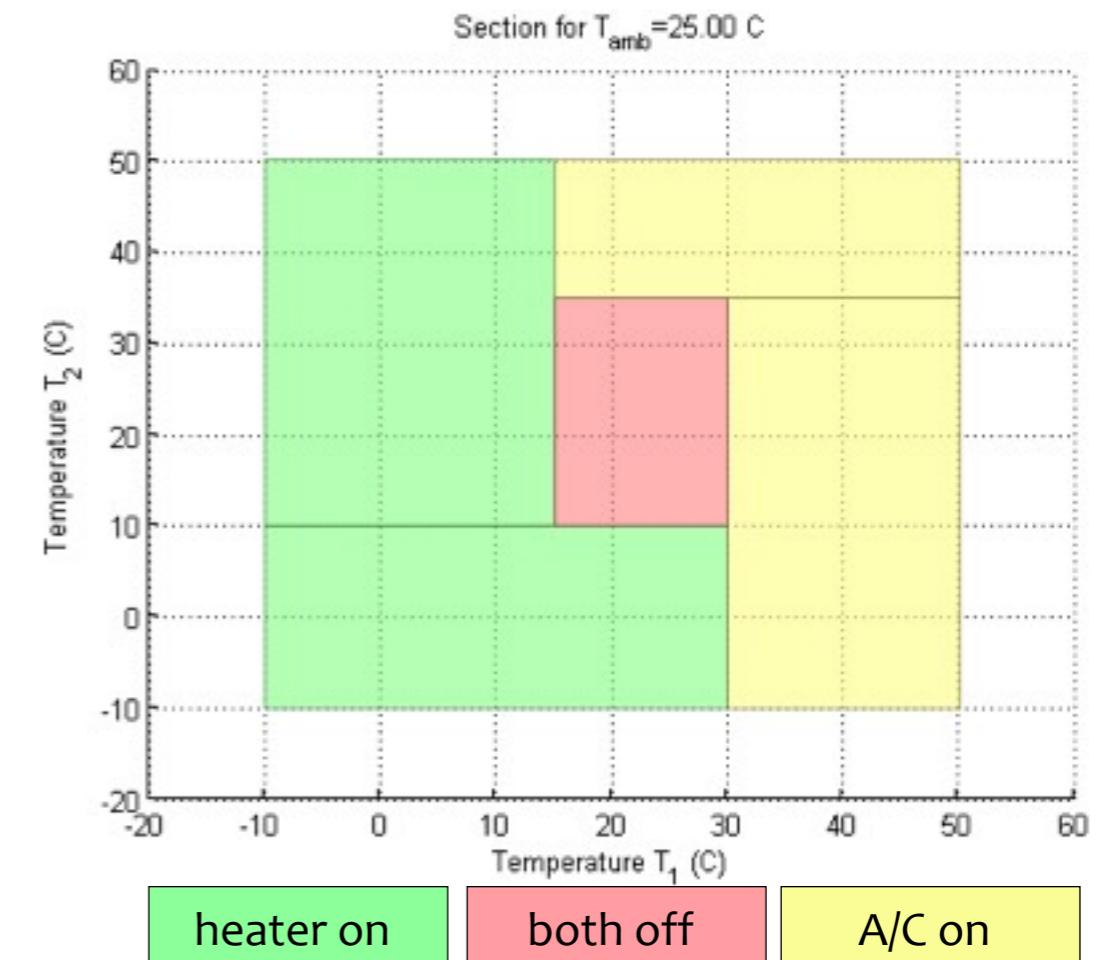
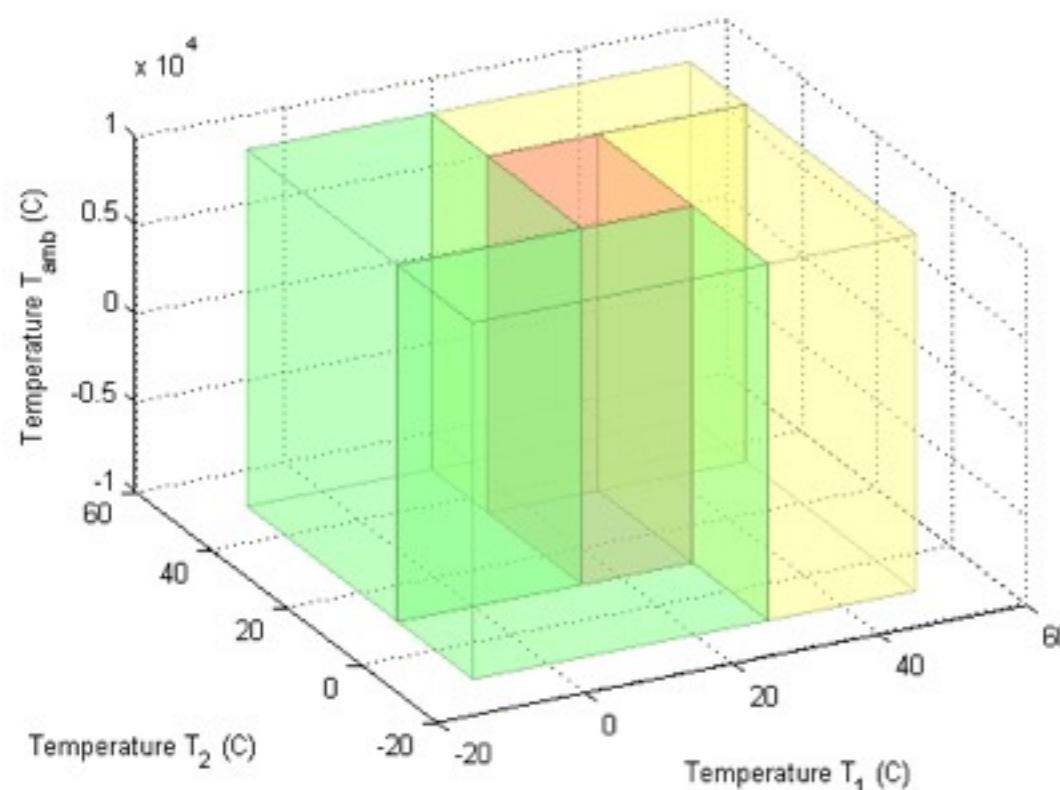
$$i(k) \text{ s.t. } H_i(k)x(k) + J_i(k)u(k) \leq K_i(k)$$

- 2 continuous states:

(temperatures T_1, T_2)

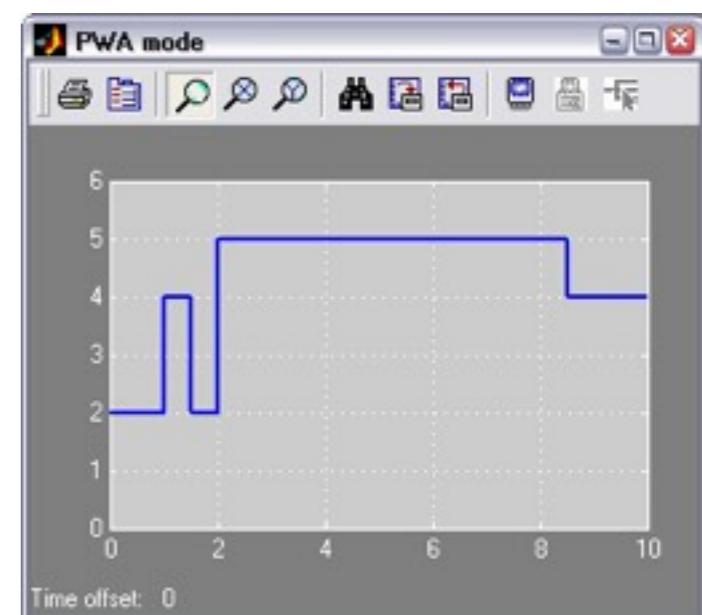
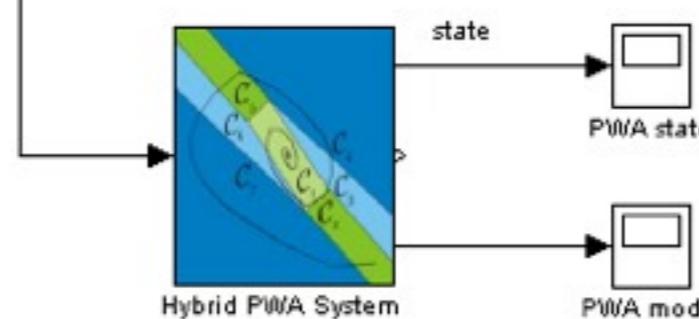
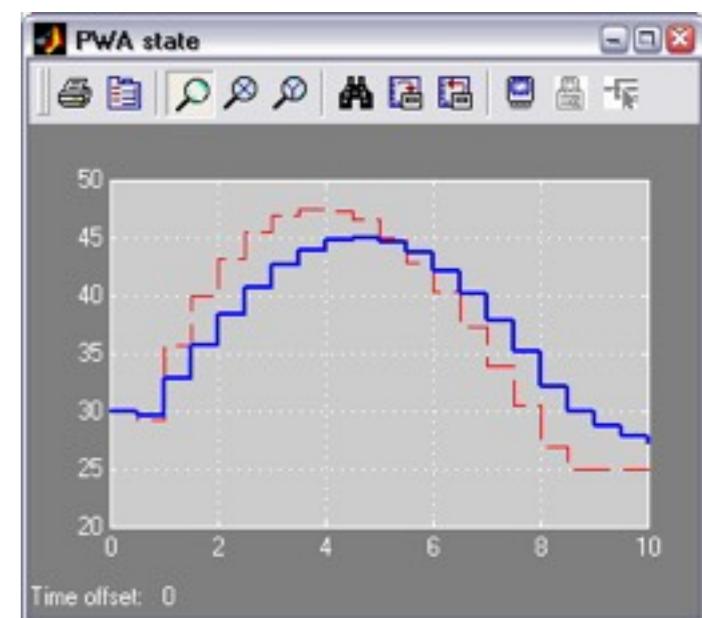
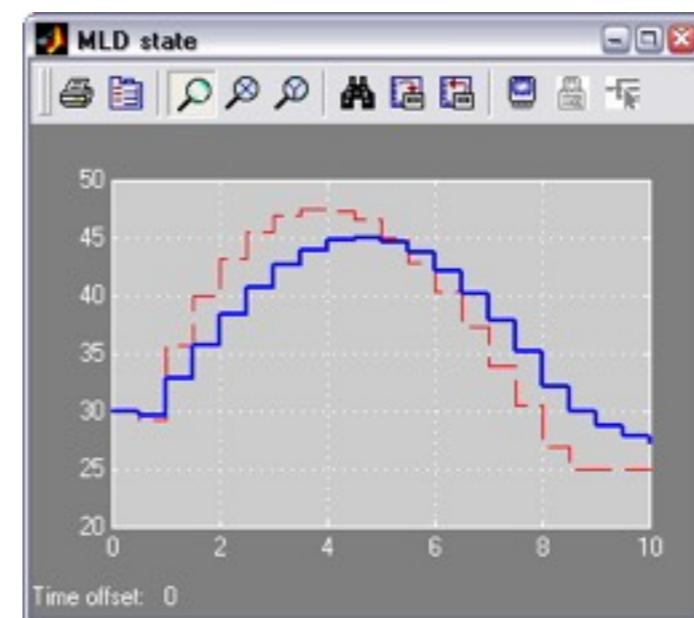
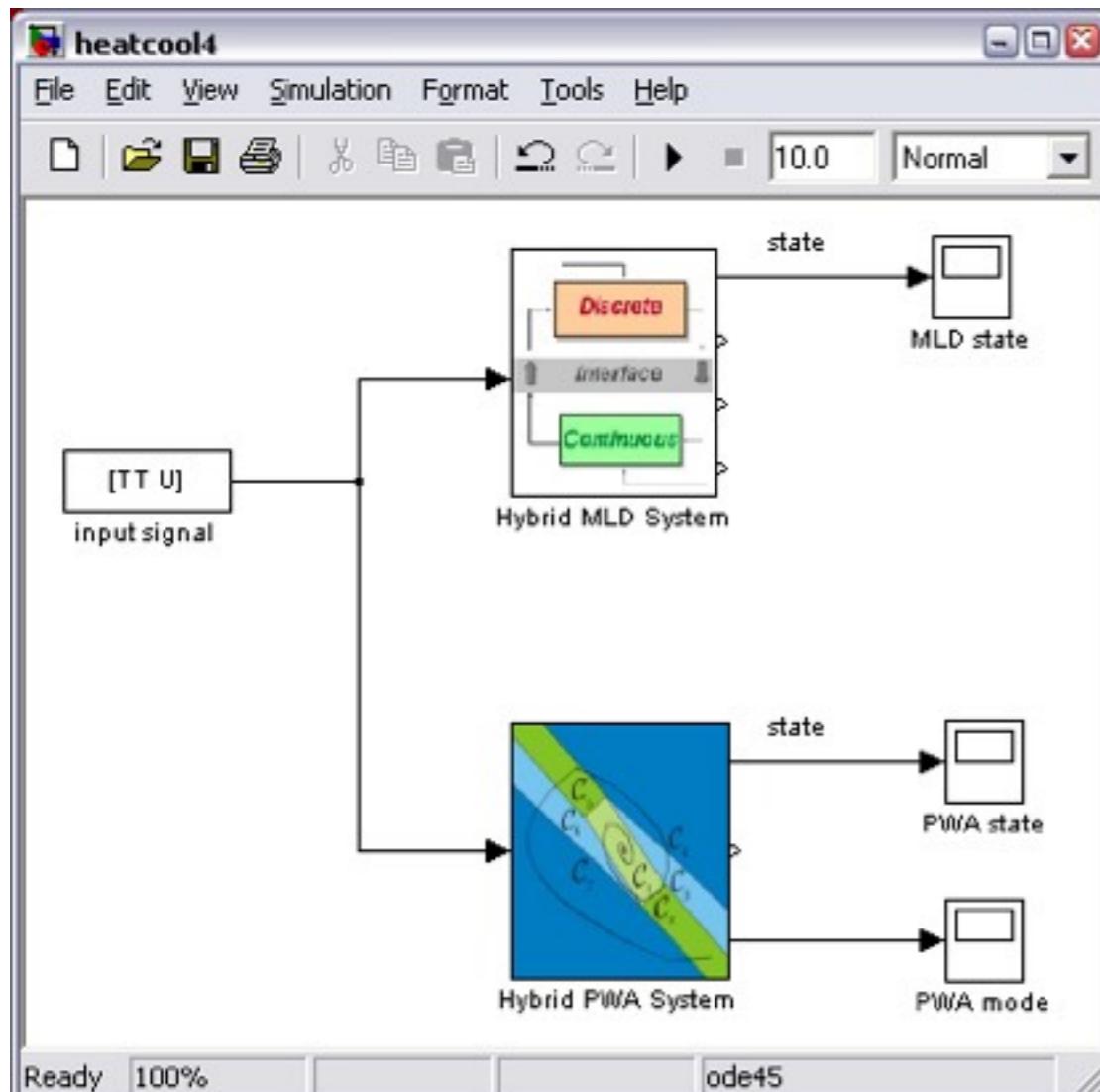
- 1 continuous input:

(room temperature T_{amb})



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

SIMULATION IN SIMULINK



MLD and PWA models are equivalent

USING MLD TO PWA FOR MODEL CHECKING

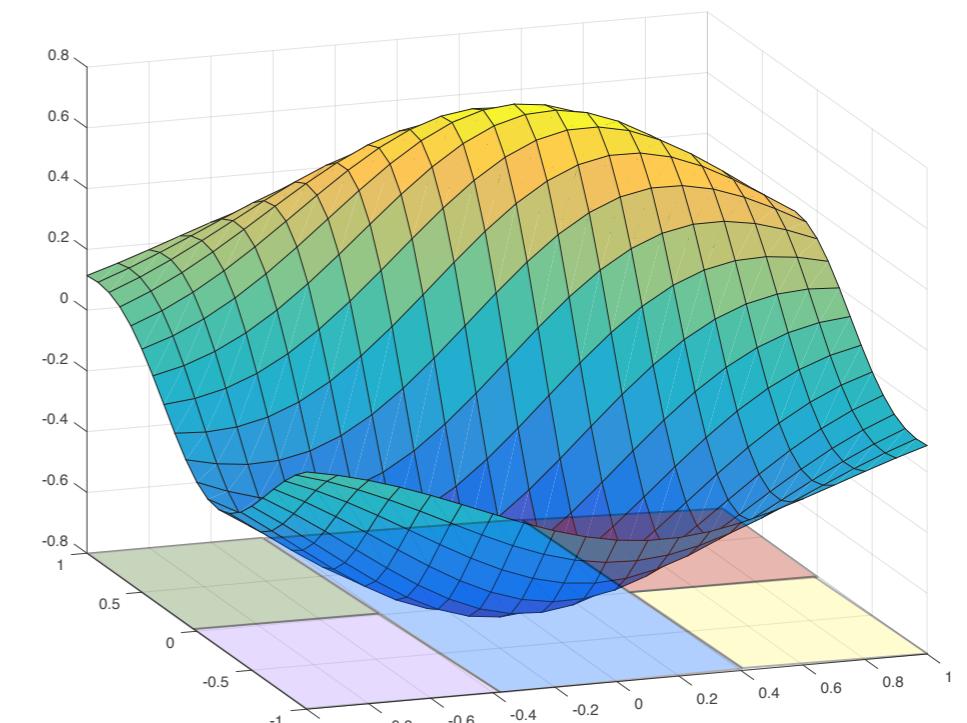
- Assume plant + controller can be modeled as DHA:
 - **Plant** = PWA approximation (e.g.: nonlinear switched model)
 - **Controller** = switched linear controller (e.g: a combination of threshold conditions, logic, linear feedback laws, ...)
- Write HYSDEL model, convert to MLD, then to PWA
- The resulting PWA map tells how the closed-loop system behaves in different regions of the state-space

IDENTIFICATION OF HYBRID SYSTEMS

HYBRID SYSTEM IDENTIFICATION

- A *hybrid model* of the process may not be available from physical principles
- Therefore, a model must be either
 - Estimated from data ([model unknown](#))
 - or *hybridized* ([model known but nonlinear](#))
- If one linear model is enough: easy problem (see Ljung's SYS-ID TBX)
- If switching sequence known: easy, just identify one linear model per mode
- If modes & dynamics must be identified simultaneously, we need
hybrid system identification

In industrial MPC applications, most of the effort is spent in **identifying (multiple) linear prediction models from data**



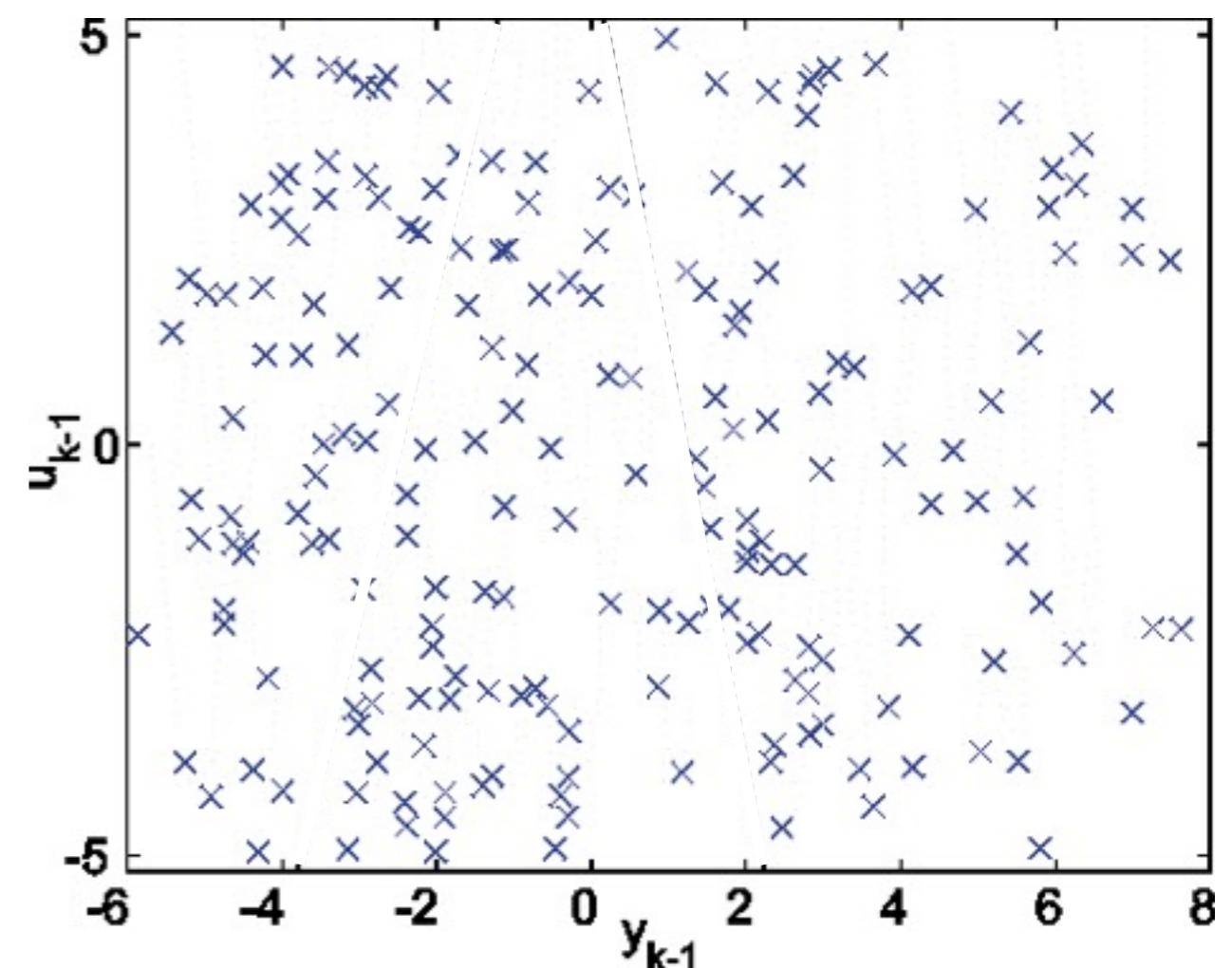
PWA IDENTIFICATION PROBLEM

Estimate from data **both the parameters** of the affine submodels and the **partition** of the PWA map

Example Let the data be generated by the PWARX system

$$y_k = \begin{cases} \begin{bmatrix} -0.4 & 1 & 1.5 \end{bmatrix} \varphi_k + \varepsilon_k \\ \text{if } \begin{bmatrix} 4 & -1 & 10 \end{bmatrix} \varphi_k < 0 \end{cases}$$
$$\begin{cases} \begin{bmatrix} 0.5 & -1 & -0.5 \end{bmatrix} \varphi_k + \varepsilon_k \\ \text{if } \begin{bmatrix} -4 & 1 & -10 \end{bmatrix} \varphi_k \leq 0 \end{cases}$$
$$\begin{cases} \begin{bmatrix} -0.3 & 0.5 & -1.7 \end{bmatrix} \varphi_k + \varepsilon_k \\ \text{if } \begin{bmatrix} -5 & -1 & 6 \end{bmatrix} \varphi_k < 0 \end{cases}$$

with $\varphi_k = [y_{k-1} \ u_{k-1} \ 1]'$, $|u_k| \leq 5$
and $|\varepsilon_k| \leq 0.1$



APPROACHES TO PWA IDENTIFICATION

- Mixed-integer linear or quadratic programming (Roll, Bemporad, Ljung, 2004)
- Partition of infeasible set of inequalities (Bemporad, Garulli, Paoletti, Vicino, 2005)
- K-means clustering in a feature space (Ferrari-Trecate, Muselli, Liberati, Morari, 2003)
- Bayesian approach (Juloski, Wieland, Heemels, 2004)
- Kernel-based approaches (Pillonetto, 2016)
- Hyperplane clustering in data space (Münz, Krebs, 2002)
- Recursive multiple least squares & PWL separation (Breschi, Piga, Bemporad, 2016)

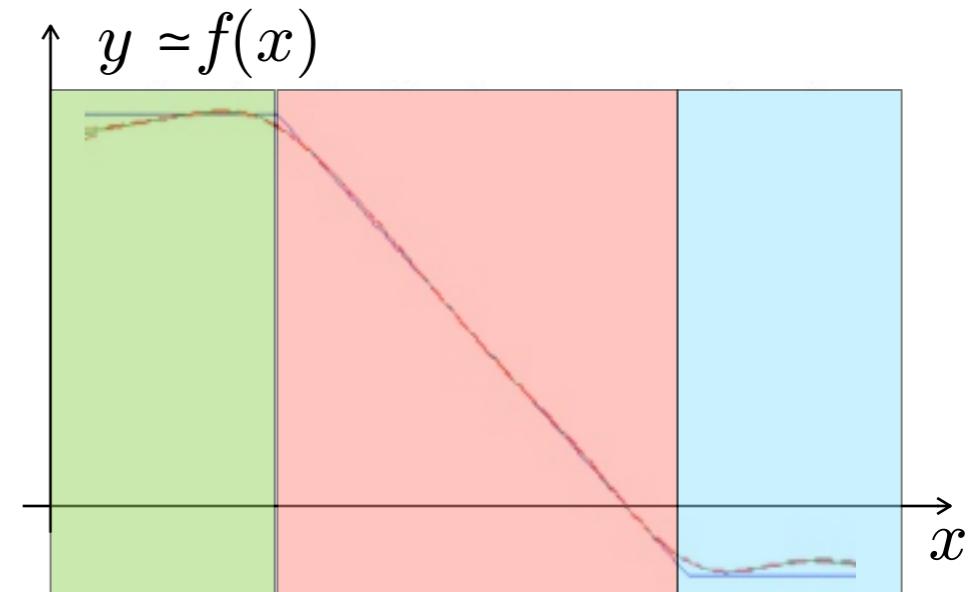
PWA REGRESSION PROBLEM

- **Problem:** given input/output pairs $\{x(k), y(k)\}$, $k=1, \dots, N$ and number s of models, compute an approximation $y \approx f(x)$

$$f(x) = \begin{cases} F_1 x + g_1 & \text{if } H_1 x \leq K_1 \\ \vdots \\ F_s x + g_s & \text{if } H_s x \leq K_s \end{cases}$$

**PWA model
(PieceWise Affine)**

- Need to learn **both** the parameters (F_i, g_i) of the affine submodels **and** the partition (H_i, K_i) of the PWA map from data (off-line learning)
- Possibly need to update model and partition as new data are collected (on-line learning)



PWA REGRESSION PROBLEM

- **Special case #1:** hybrid (PWARX) model

$$x(k) = [y'(k-1) \ y'(k-2) \ \cdots \ y'(k-n_a) \\ u'(k-1) \ u'(k-2) \ \cdots \ u'(k-n_b)]'$$

$$y(k) = f(x(k))$$

f = piecewise affine function of regressor x

- **Special case #2:** switched parameter-dependent ARX model

$$y(k) = \sum_{i=1}^{n_a} a_i(p(k))y(k-i) + \sum_{j=0}^{n_b} b_j(p(k))u(k-j) + e(k)$$

(a_i, b_j) = piecewise affine functions of scheduling parameter p

PWA REGRESSION ALGORITHM (1/2)

(Breschi, Piga, Bemporad, 2016)

- Two-stage algorithm for PWA regression
 - **Stage 1:** Cluster the regressors $x(k)$ and simultaneously estimate the parameter matrices (F_i, g_i) recursively (=one sample at the time)

$$i(k) \leftarrow \arg \min_{i=1, \dots, s} e_i(k)' \Lambda_e^{-1} e_i(k) + (x(k) - c_i)' R_i^{-1} (x(k) - c_i)$$

↓ ↓
 one-step prediction error
 of model #i proximity to centroid
 of cluster #i

- Only update $(F_{i(k)}, g_{i(k)})$ using recursive least squares based on inverse QR decomposition (Alexander, Ghirnikar, 1993)
 - Iterate the procedure M times for improved results

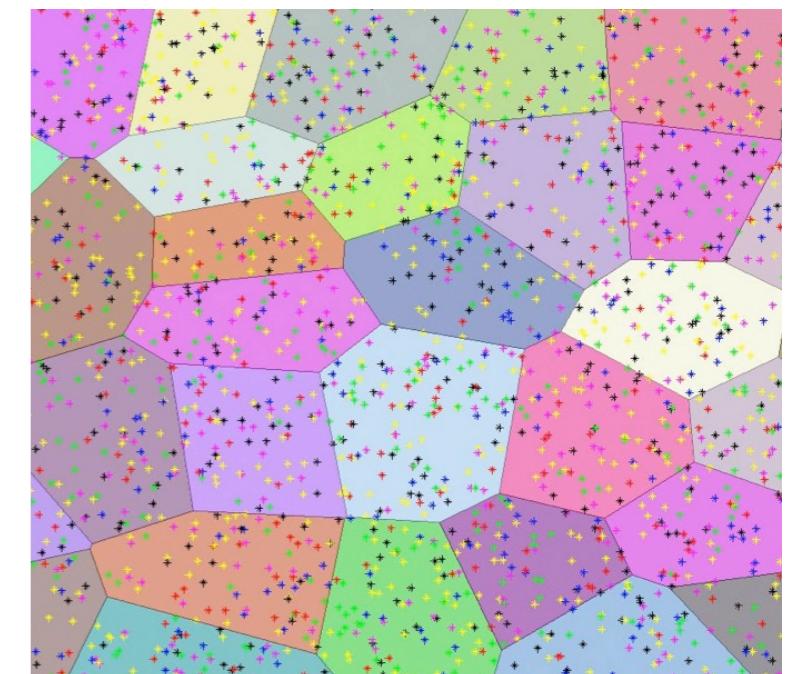
PWA REGRESSION ALGORITHM (2/2)

(Breschi, Piga, Bemporad, 2016)

- **Stage 2:** Compute a polyhedral partition (H_i, K_i) of the regressor space via multi-category linear separation (batch or incrementally):

$$\phi(x) = \max_{i=1,\dots,s} \{w_i'x - \gamma_i\}$$

$$X_i = \left\{x \in \mathbb{R}^n : \phi(x) = w_i'x - \gamma_i\right\}$$



- Alternative ways to compute (w_i, γ_i) :

→ **Robust Linear Programming (batch)** (Bennett, Mangasarian, 1994)

→ **Piecewise-smooth Newton method (batch)** (Bemporad, Bernardini, Patrinos, 2015)

→ **Averaged stochastic gradient descent (online)** (Bottou, 2012)

PWA REGRESSION EXAMPLES

(Breschi, Piga, Bemporad, 2016)

- Identification of piecewise-affine ARX model

$$\begin{bmatrix} y_1(k) \\ y_2(k) \end{bmatrix} = \begin{bmatrix} -0.83 & 0.20 \\ 0.30 & -0.52 \end{bmatrix} \begin{bmatrix} y_1(k-1) \\ y_2(k-1) \end{bmatrix} + \begin{bmatrix} -0.34 & 0.45 \\ -0.30 & 0.24 \end{bmatrix} \begin{bmatrix} u_1(k-1) \\ u_2(k-1) \end{bmatrix} \\ + \begin{bmatrix} 0.20 \\ 0.15 \end{bmatrix} + \max \left\{ \begin{bmatrix} 0.20 & -0.90 \\ 0.10 & -0.42 \end{bmatrix} \begin{bmatrix} y_1(k-1) \\ y_2(k-1) \end{bmatrix} \right. \\ \left. + \begin{bmatrix} 0.42 & 0.20 \\ 0.50 & 0.64 \end{bmatrix} \begin{bmatrix} u_1(k-1) \\ u_2(k-1) \end{bmatrix} + \begin{bmatrix} 0.40 \\ 0.30 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \end{bmatrix} \right\} + e_o(k),$$

Results: *quality of fit*

		$N = 4000$	$N = 20000$	$N = 100000$
BFR ₁	(Off-line) RLP [8]	96.0 %	96.5 %	99.0 %
	(Off-line) RPSN	96.2 %	96.4 %	98.9 %
	(On-line) ASGD	86.7 %	95.0 %	96.7 %
BFR ₂	(Off-line) RLP [8]	96.2 %	96.9 %	99.0 %
	(Off-line) RPSN	96.3 %	96.8 %	99.0 %
	(On-line) ASGD	87.4 %	95.2 %	96.4 %

$$BFR_i = \max \left\{ 1 - \frac{\|y_{o,i} - \hat{y}_i\|_2}{\|y_{o,i} - \bar{y}_{o,i}\|_2}, 0 \right\}$$

(Best Fit Rate)

RLP = robust linear programming
RPSN = piecewise-smooth Newton
ASGD = (one-pass) averaged stochastic gradient

CPU time for computing the partition

	$N = 4000$	$N = 20000$	$N = 100000$
(Off-line) RLP [8]	0.308 s	3.227 s	112.435 s
(Off-line) RPSN	0.016 s	0.086 s	0.365 s
(On-line) ASGD	0.013 s	0.023 s	0.067 s

PWA REGRESSION EXAMPLES

(Breschi, Piga, Bemporad, 2016)

- Identification of piecewise-linear LPV-ARX model

$$\begin{bmatrix} y_1(k) \\ y_2(k) \end{bmatrix} = \begin{bmatrix} \bar{a}_{1,1}(p(k)) & \bar{a}_{1,2}(p(k)) \\ \bar{a}_{2,1}(p(k)) & \bar{a}_{2,2}(p(k)) \end{bmatrix} \begin{bmatrix} y_1(k-1) \\ y_2(k-1) \end{bmatrix} + \begin{bmatrix} \bar{b}_{1,1}(p(k)) & \bar{b}_{1,2}(p(k)) \\ \bar{b}_{2,1}(p(k)) & \bar{b}_{2,2}(p(k)) \end{bmatrix} \begin{bmatrix} u_1(k-1) \\ u_2(k-1) \end{bmatrix} + e_o(k)$$

$$\bar{a}_{1,1}(p(k)) = \begin{cases} -0.3 & \text{if } 0.4(p_1(k) + p_2(k)) \leq -0.3, \\ 0.3 & \text{if } 0.4(p_1(k) + p_2(k)) \geq 0.3, \\ 0.4(p_1(k) + p_2(k)) & \text{otherwise,} \end{cases}$$

$$\bar{a}_{1,2}(p(k)) = 0.5(|p_1(k)| + |p_2(k)|), \quad \bar{a}_{2,1}(p(k)) = p_1(k) - p_2(k)$$

$$\bar{a}_{2,2}(p(k)) = \begin{cases} 0.5 & \text{if } p_1(k) < 0, \\ 0 & \text{if } p_1(k) = 0, \\ -0.5 & \text{if } p_1(k) > 0, \end{cases}$$

$$\bar{b}_{1,1}(p(k)) = 3p_1(k) + p_2(k),$$

$$\bar{b}_{1,2}(p(k)) = \begin{cases} 0.5 & \text{if } 2(p_1^2(k) + p_2^2(k)) \geq 0.5, \\ 2(p_1^2(k) + p_2^2(k)) & \text{otherwise,} \end{cases}$$

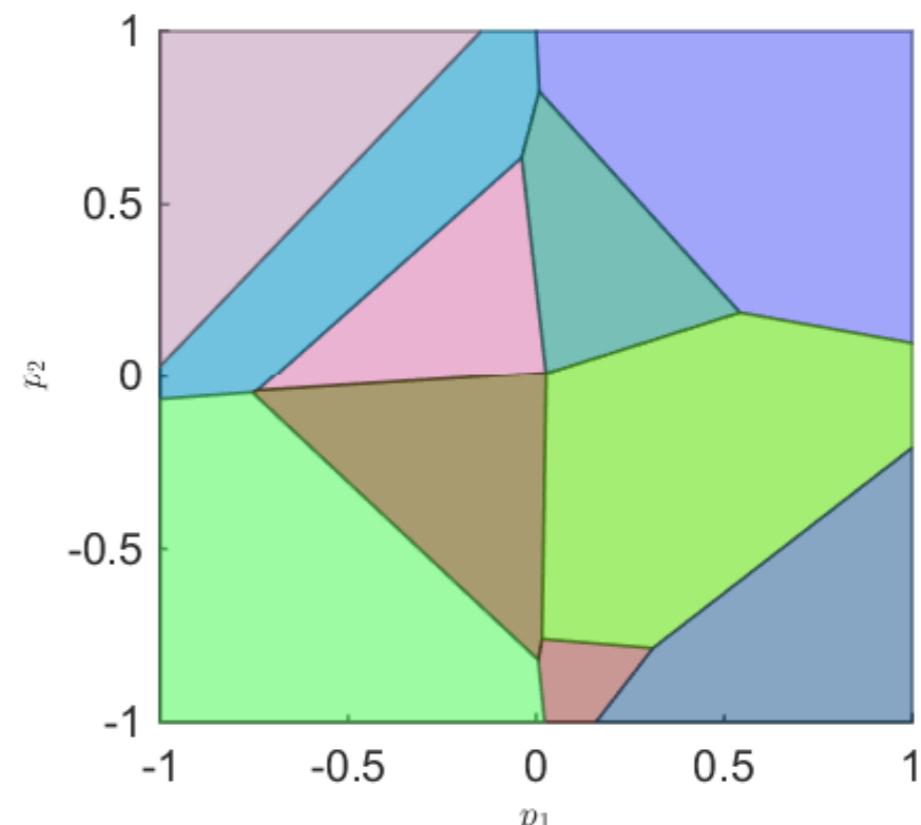
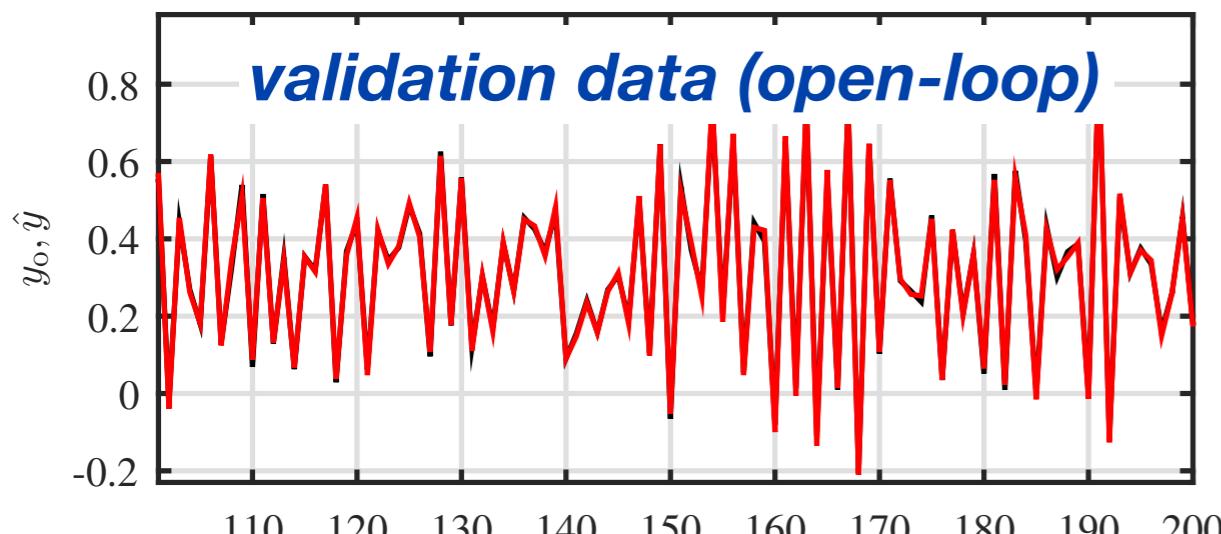
$$\bar{b}_{2,1}(p(k)) = 2 \sin\{p_1(k) - p_2(k)\}, \quad \bar{b}_{2,2}(p(k)) = 0.$$

Results:

quality of fit

	BFR ₁	BFR ₂
PWA regression	87 %	84 %
parametric LPV [3]	80 %	70 %

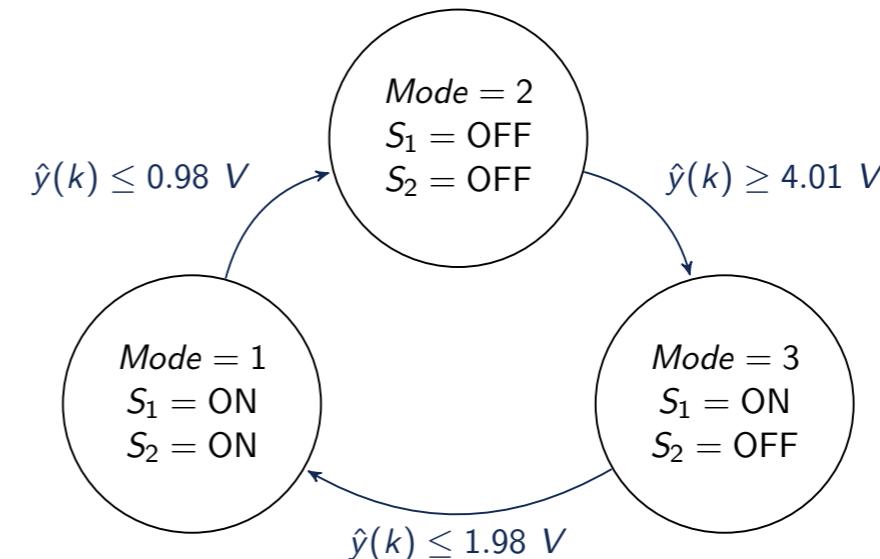
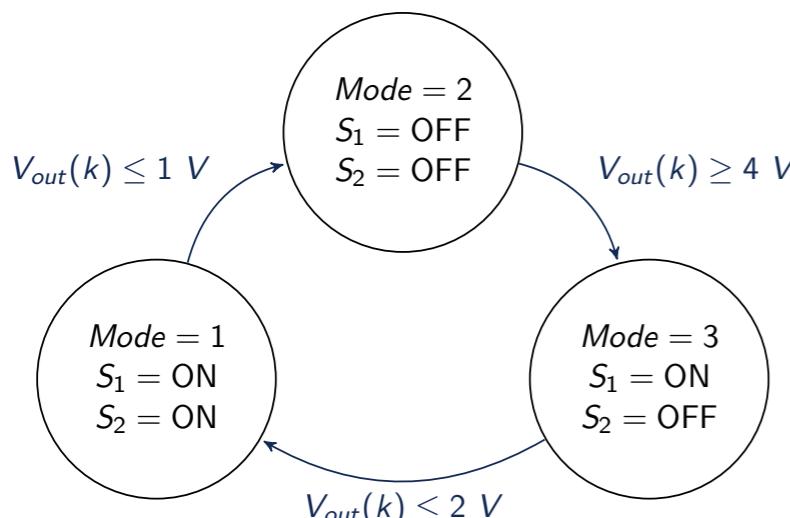
[3] = Bamieh, Giarré (2002)



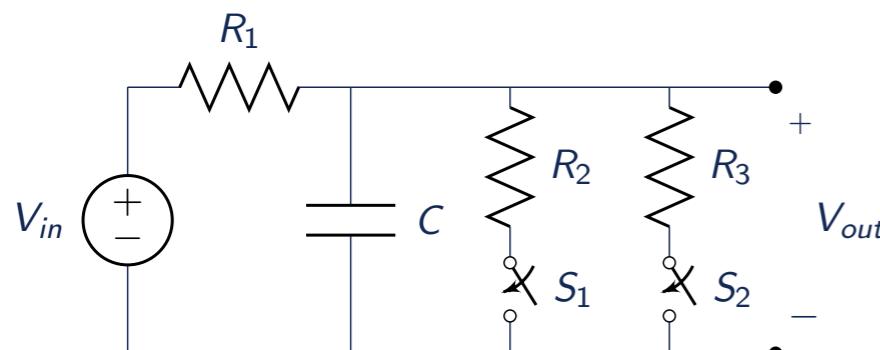
IDENTIFICATION OF HYBRID MODELS WITH LOGIC STATES

(Breschi, Piga, Bemporad, CDC 2016)

- Identification of hybrid models from data



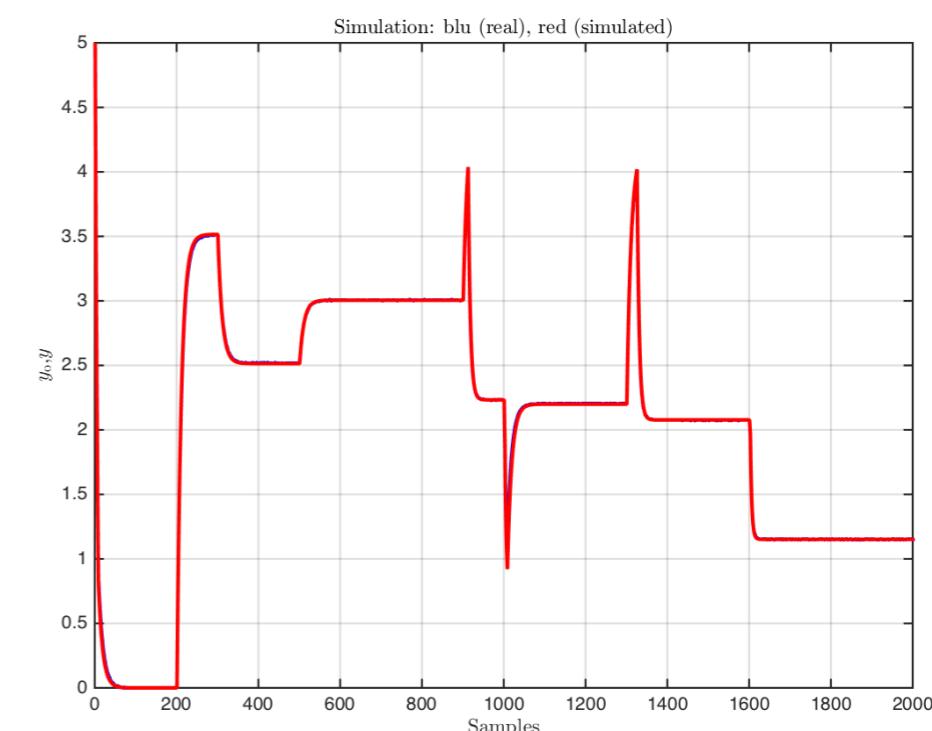
identified system



true system



CPU time to compute the partitions: **0.033 s**
CPU required to identify the DHA: **0.078 s***



Validation set of $N_V = 2000$ samples:
 $\text{BFR} = 98.64 \%$