Modeling, Control, and Reachability Analysis of Discrete-Time Hybrid Systems

Alberto Bemporad

DISC Course on
Modeling and Control of Hybrid Systems

March 31, 2003

Università degli Studi di Siena
(founded in 1240)

COHES Group
Control and Optimization of Hybrid and Embedded SystemS

http://www.dii.unisi.it/~bemporad

Outline

- What is a “hybrid system” ?
- Models for hybrid systems
- Control of hybrid systems
- Safety analysis of hybrid systems
- Examples (automotive)

Motivating 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

Hybrid Systems

(Witsenhausen, 1966)

Computer Science

Control Theory

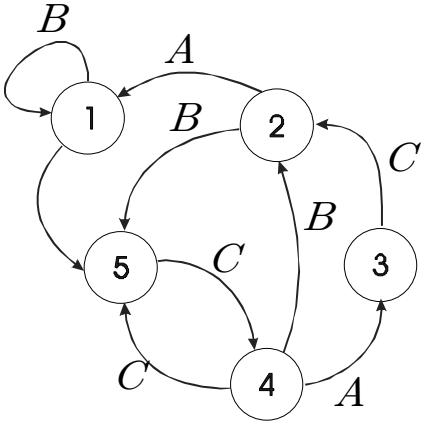
$X = \{1,2,3,4,5\}$
$U = \{A,B,C\}$

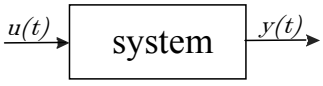
$x \in \mathbf{R}^n, u \in \mathbf{R}^m$
$y \in \mathbf{R}^p$

Finite state machines

Continuous dynamical systems

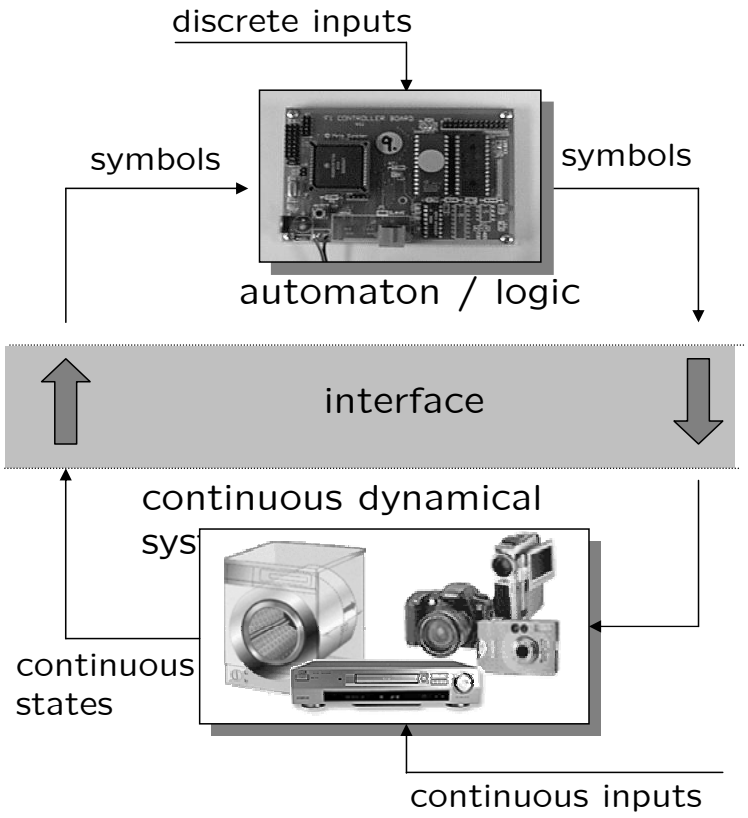
Hybrid systems





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

Motivation: Embedded Systems

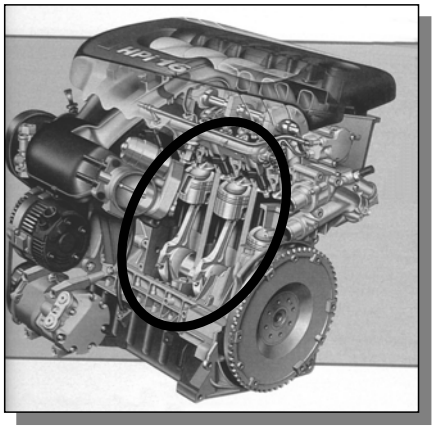


- Consumer electronics
- Home appliances
- Office automation
- Automobiles
- Industrial plants
- ...

Motivation: "Intrisically Hybrid"

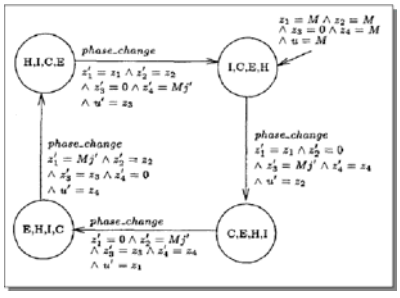
- Transmission

Discrete command (R,N,1,2,3,4,5) + Continuous dynamical variables (velocities, torques)



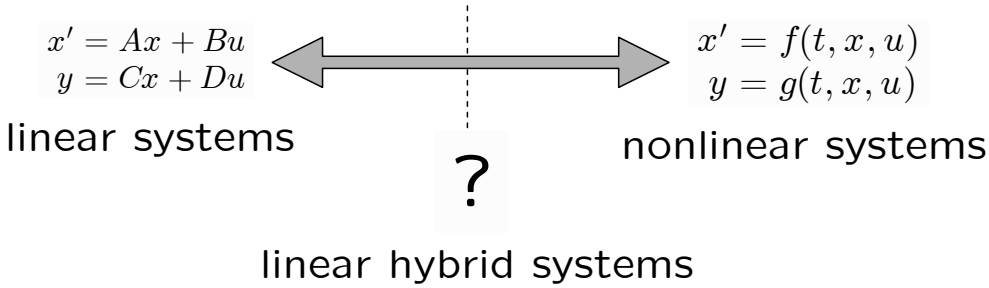
- Four-stroke engines

Automaton, dependent on power train motion →



Key Requirements for Hybrid Models

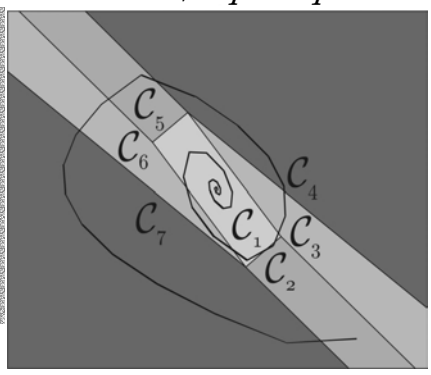
- **Descriptive** enough to capture the behavior of the system
  - continuous dynamics (physical laws)
  - logic components (switches, automata, software code)
  - interconnection between logic and dynamics
- **Simple** enough for solving *analysis* and *synthesis* problems



Piecewise Affine Systems *(Sontag 1981)*

$$
\begin{aligned}
x'(k) &= A_{i(k)}x(k) + B_{i(k)}u(k) + f_{i(k)} \\
y(k) &= C_{i(k)}x(k) + D_{i(k)}u(k) + g_{i(k)} \\
\\
i(k) \quad &\text{s.t.} \quad H_{i(k)}x(k) + J_{i(k)}u(k) \leq K_{i(k)}
\end{aligned}
$$

state+input space



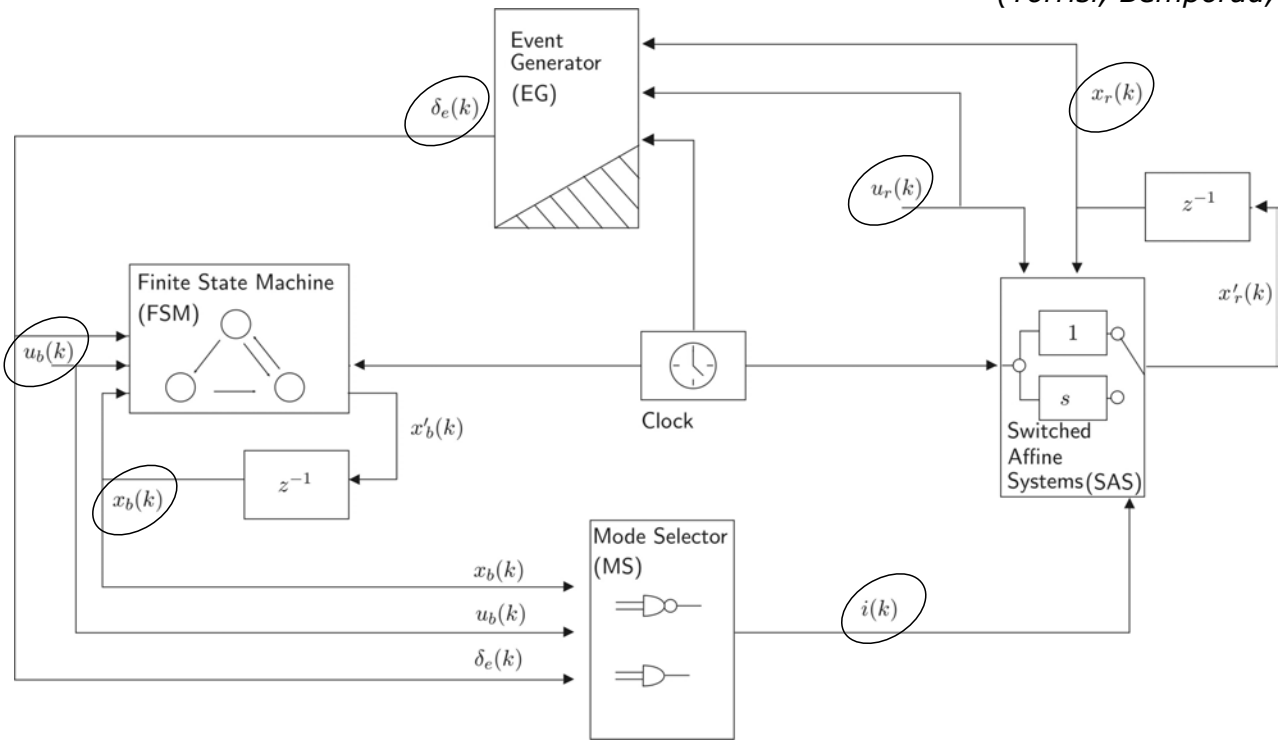
$x \in \mathcal{X} \subseteq \mathbb{R}^n$, $u \in \mathcal{U} \subseteq \mathbb{R}^m$, $y \in \mathcal{Y} \subseteq \mathbb{R}^p$

$i(k) \in \{1, \ldots, s\}$

- Approximates nonlinear dynamics arbitrarily well
- Suitable for stability analysis, reachability analysis (verification), controller synthesis, ...

Discrete Hybrid Automata

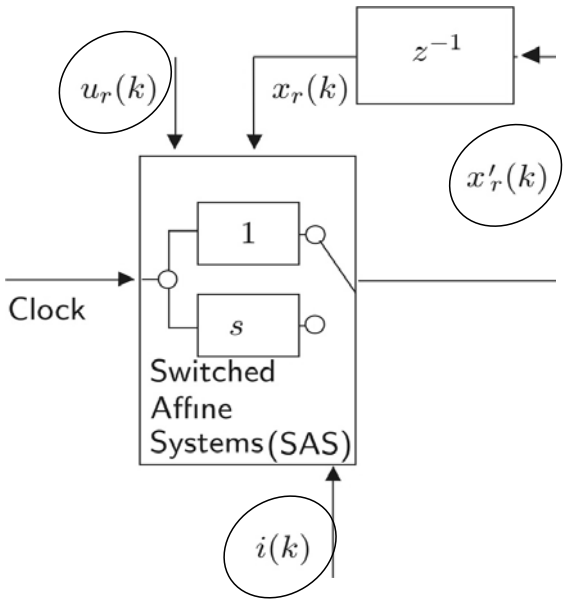
(Torrisi, Bemporad, 2003)



$x_r \in \mathbb{R}^{n_r}$ = continuous states
$x_b \in \{0,1\}^{n_b}$ = binary states
$i(k) \in \{1,\ldots,s\}$ = mode

$u_r \in \mathbb{R}^{m_r}$ = continuous inputs
$u_b \in \{0,1\}^{m_b}$ = binary inputs
$\delta_e \in \{0,1\}^{n_e}$ = event conditions

Switched Affine Systems



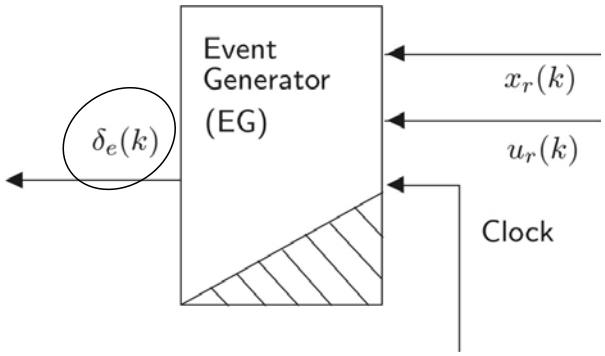
Linear affine dynamics depends upon the mode $i(k)$

$$x'_r(k) = A_{i(k)}x_r(k) + B_{i(k)}u_r(k) + f_{i(k)}$$

$$y_r(k) = C_{i(k)}x_r(k) + D_{i(k)}u_r(k) + g_{i(k)}$$

$x_r \in \mathbb{R}^{n_r},\ u_r \in \mathbb{R}^{m_r},\ y_r \in \mathbb{R}^{p_r}$

Event Generator

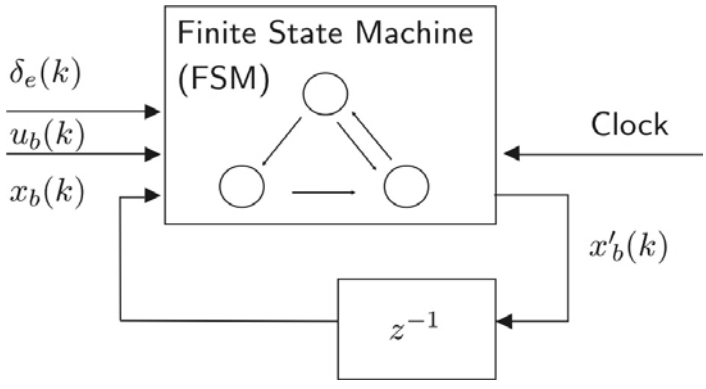


Generates a logic signal according to the satisfaction of a linear affine constraint

$$[\delta_e^i(k) = 1] \leftrightarrow [H^i x_r(k) + K^i u_r(k) + J^i k \leq W^i], \;\; \delta_e^i \in \{0, 1\}$$

$$x_r \in \mathbb{R}^{n_r}, \;\; u_r \in \mathbb{R}^{m_r}, \;\; \delta_e^i \in \{0, 1\}$$

Finite State Machine

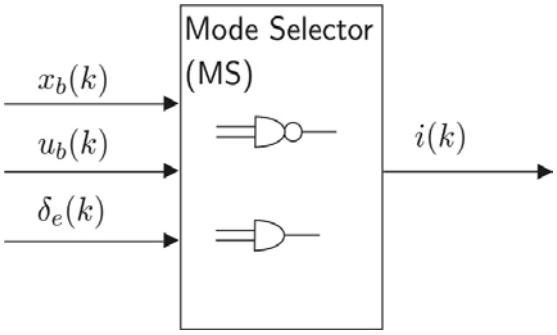


Discrete dynamics evolving according to a Boolean state update function

$$\begin{aligned} x'_b(k) &= f_{\mathsf{B}}(x_b(k), u_b(k), \delta_e(k)) \\ y_b(k) &= g_{\mathsf{B}}(x_b(k), u_b(k), \delta_e(k)) \end{aligned}$$

$$x_b \in \{0, 1\}^{n_b}, \;\; y_b \in \{0, 1\}^{p_b}$$

Mode Selector



a Boolean function selects the active mode $i(k)$ of the SAS

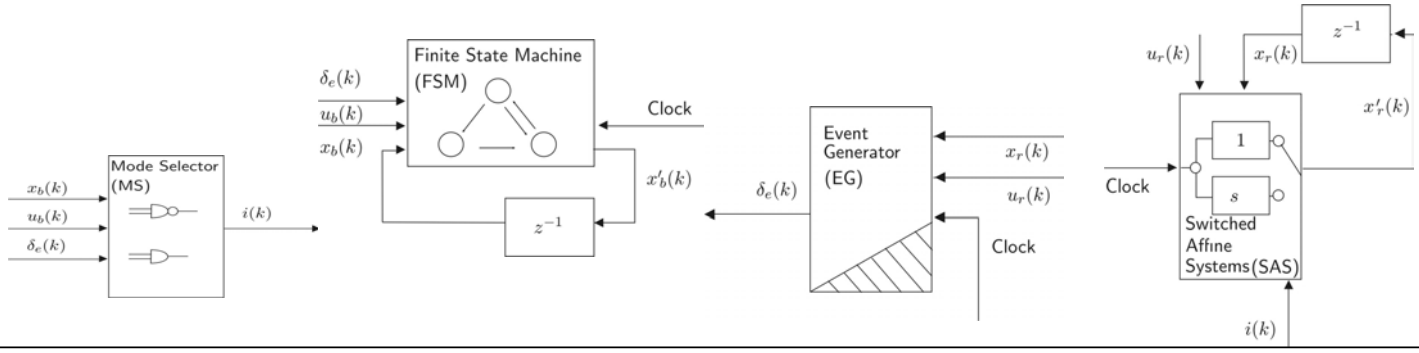
$$i(k) \;=\; f_{\mathsf{M}}(x_b(k), u_b(k), \delta_e(k))$$

$$i(k) \in \left\{ \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ \vdots \\ 0 \end{bmatrix}, \ldots, \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \end{bmatrix} \right\} \subset \{0,1\}^s$$

Logic and Inequalities

Glover 1975, Williams 1977

$X_1 \vee X_2$	$X_1 + X_2 \geq 1$
Any logic statement $f(X)$ =TRUE $\bigwedge_{j=1}^{m} \left(\vee_{i \in P_j} X_i \vee_{i \in N_j} \neg X_i \right)$ (CNF) $N_j, P_j \subseteq \{1, \ldots, n\}$	$A\delta \leq B$ $\begin{cases} 1 \leq \sum_{i \in P_1} X_i + \sum_{i \in N_1} (1 - X_i) \\ \vdots \\ 1 \leq \sum_{i \in P_m} X_i + \sum_{i \in N_m} (1 - X_i) \end{cases}$
$[\delta_e^i(k) = 1] \leftrightarrow [H^i x_r(k) \leq W^i]$	$\begin{array}{rcl} H^i x_r(k) - W^i & \leq & M^i(1 - \delta_e^i) \\ H^i x_r(k) - W^i & > & m^i \delta_e^i \end{array}$
IF δ THEN $z = a_1^T x + b_1^T u + f_1$ ELSE $z = a_2^T x + b_2^T u + f_2$	$\begin{array}{rcl} (m_2 - M_1)\delta + z & \leq & a_2 x + b_2 u + f_2 \\ (m_1 - M_2)\delta - z & \leq & -a_2 x - b_2 u - f_2 \\ (m_1 - M_2)(1 - \delta) + z & \leq & a_1 x + b_1 u + f_1 \\ (m_2 - M_1)(1 - \delta) - z & \leq & -a_1 x - b_1 u - f_1 \end{array}$



Transformation of Logic into Linear Integer Inequalities

0. Given a Boolean statement $F(X_1, X_2, \ldots, X_n)$

1. Convert to Conjunctive Normal Form (CNF):

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

2. Transform into inequalities:

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

$$A\delta \leq B, \quad \delta \in \{0,1\}$$

➡ Every logic proposition can be translated into linear integer inequalities

General Transformation into Linear Integer Inequalities

Geometric approach: The polytope $P \triangleq \{\delta : A\delta \leq B\}$ is the convex hull of the rows of the truth table T

T:

x_1	x_2	$\ldots$	x_{n-1}	$x_n = F(x_1, \ldots, x_{n-1})$
0	0		0	1
0	0		1	0
⋮	⋮		⋮	
1	1		1	1

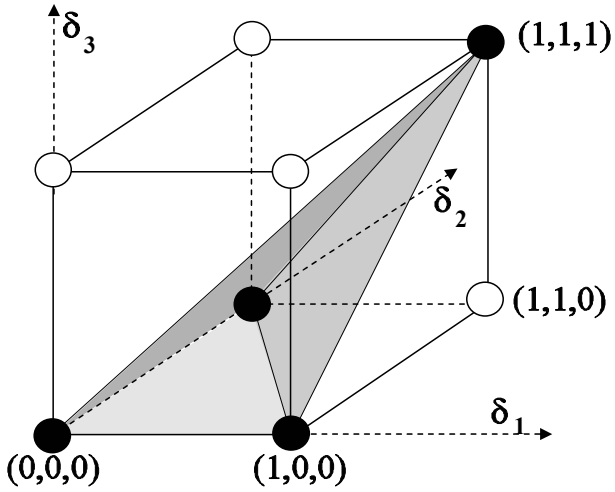
⇨ $A\delta \leq B, \delta \in \{0,1\}^n$

➡ Every logic proposition can be translated into linear integer inequalities

Truth Tables ➡ Linear Integer Inequalities

Example: logic "AND"

δ_1	δ_2	δ_3
0	0	0
0	1	0
1	0	0
1	1	1

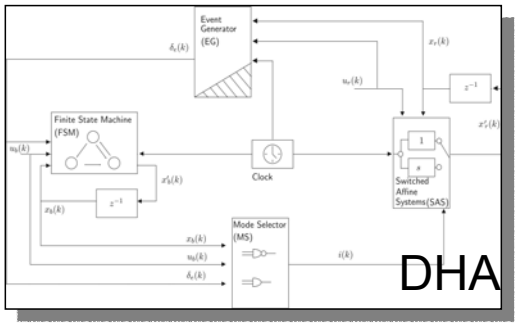


Key idea:
White points cannot be in the hull of black points

$$\text{conv}\left(\begin{bmatrix}0\\0\\0\end{bmatrix},\begin{bmatrix}0\\1\\0\end{bmatrix},\begin{bmatrix}1\\0\\0\end{bmatrix},\begin{bmatrix}1\\1\\1\end{bmatrix}\right)=\left\{\delta:\begin{array}{r}-\delta_1+\delta_3\le 0\\-\delta_2+\delta_3\le 0\\\delta_1+\delta_2-\delta_3\le 1\end{array}\right\}$$

Convex hull algorithms: **`cdd`, `lrs`, `qhull`, `chD`, `Hull`, `Porto`**

Mixed Logical Dynamical Systems



Mixed Logical Dynamical (MLD) Systems

$$\begin{aligned}x(t+1) &= Ax(t)+B_1u(t)+B_2\delta(t)+B_3z(t)+B_5\\ y(t) &= Cx(t)+D_1u(t)+D_2\delta(t)+D_3z(t)+D_5\\ E_2\delta(t)+E_3z(t) &\le E_4x(t)+E_1u(t)+E_5\end{aligned}$$

(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 optimal controller synthesis, verification, ...

A Simple Example

- System:

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

$$-10 \leq x(t) \leq 10, -1 \leq u(t) \leq 1$$

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

$\longrightarrow$

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

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

$$z(t) = \delta(t)x(t) \longrightarrow \begin{aligned} z(t) &\leq M\delta(t) \\ z(t) &\geq m\delta(t) \\ z(t) &\leq x(t) - m(1-\delta(t)) \\ z(t) &\geq x(t) - M(1-\delta(t)) \end{aligned}$$

- Rewrite as linear equation

$\longrightarrow$

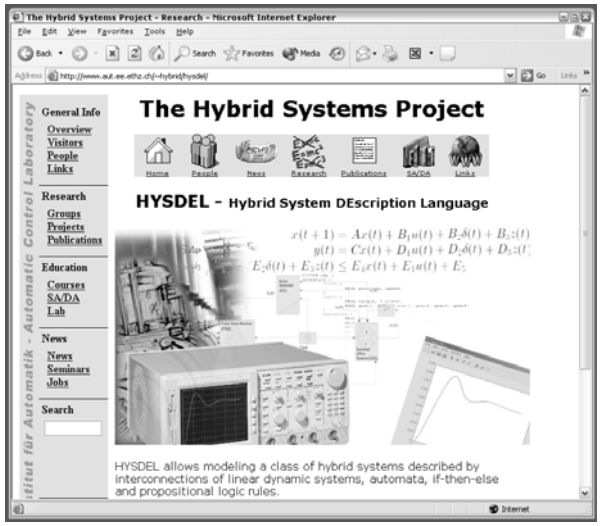
$$x(t+1) = 1.6z(t) - 0.8x(t) + u(t)$$

HYSDEL

(HYbrid Systems DEscription Language)

- Describe *hybrid systems:*

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



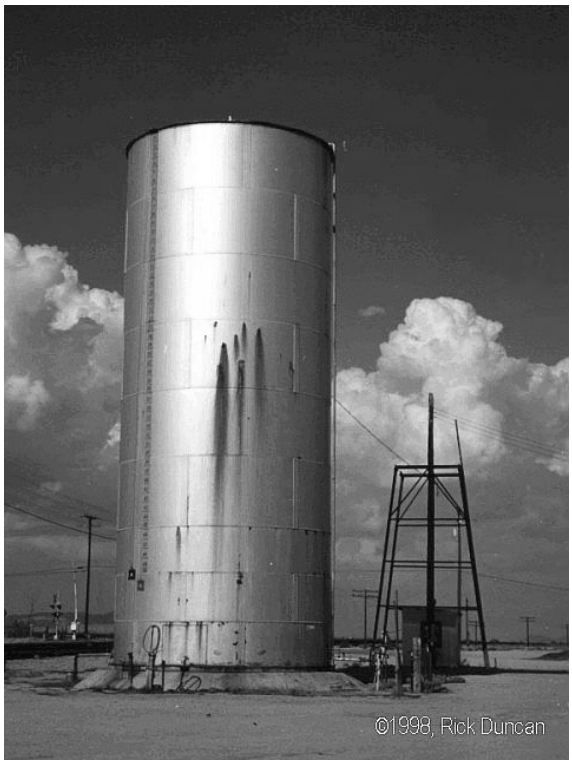
(Torrisi, Bemporad, 2003)

- Automatically generate MLD models in Matlab

- MLD model is not unique in terms of the number of auxiliary variables $\longrightarrow$ optimize model (minimize # binary variables !)

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

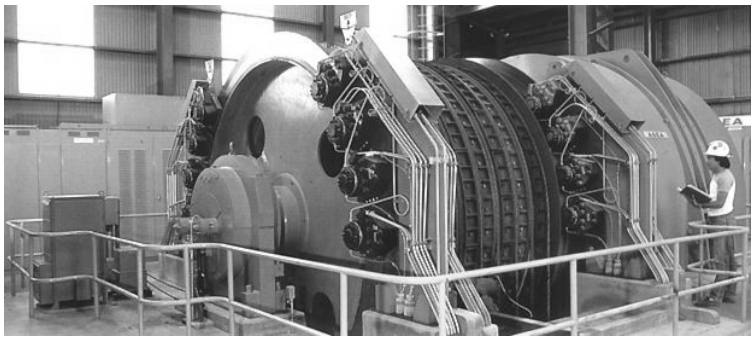
Example 1: AD section



$$[s = \mathsf{T}] \leftrightarrow [h \geq h_{\max}]$$

```
SYSTEM tank {
   INTERFACE {
      STATE {
         REAL h  [-0.3,0.3]; }
      INPUT {
         REAL Q  [-10,10]; }
      PARAMETER {
         REAL k    = 1;
         REAL e    = 1e-6; }
   } /* end interface */

   IMPLEMENTATION {
      AUX {
         BOOL s; }

      AD {
         s = hmax - h <= 0; }

      CONTINUOUS {
         h = h + k * Q; }
   } /* end implementation */
} /* end system */
```

Example 2: DA section



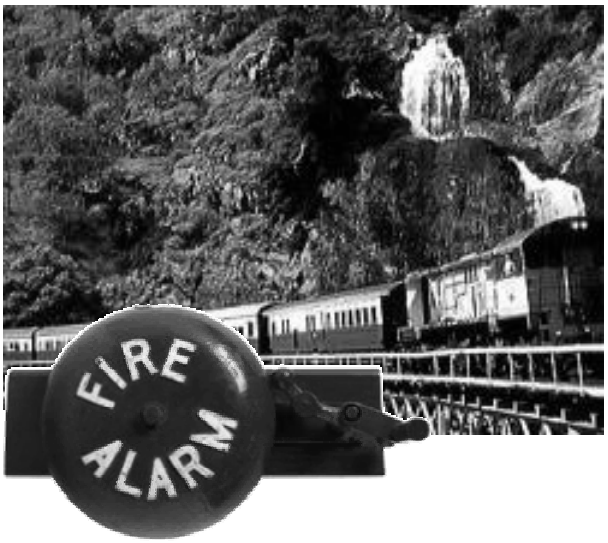
Nonlinear amplification unit

$$u_{comp} = \begin{cases} u & (u < u_t) \\ 2.3u - 1.3u_t & (u \geq u_t) \end{cases}$$

```
SYSTEM motor {
   INTERFACE {
      STATE {
         REAL ucomp; }
      INPUT {
         REAL u [0,10]; }
      PARAMETER {
         REAL ut   = 1;
         REAL e    = 1e-6;}
   } /* end interface */
```

```
IMPLEMENTATION {
      AUX {
         REAL unl;
         BOOL th; }
      AD {
         th = ut - u <= 0; }

      DA {
         unl = { IF th THEN 2.3*u - 1.3*ut
                       ELSE u };  }

      CONTINUOUS {
         ucomp = unl; }
   } /* end implementation */
} /* end system */
```

Example 3: LOGIC section

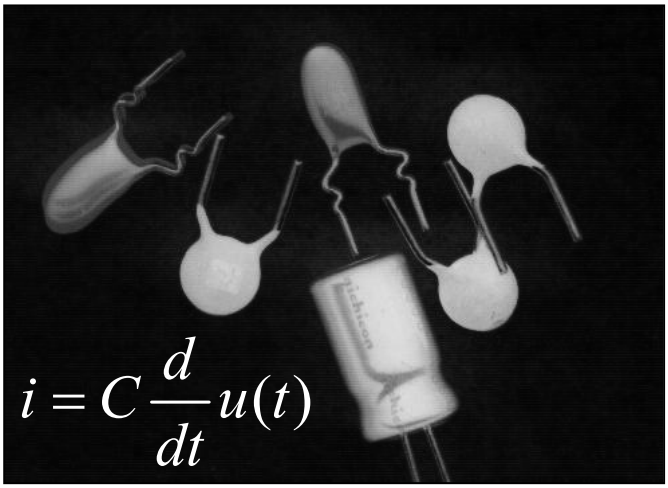


$$u_{brake} = u_{alarm} \wedge (\neg s_{tunnel} \vee \neg s_{fire})$$

$$s_{fire} \rightarrow u_{alarm}$$

```
SYSTEM train {
   INTERFACE {
      STATE {
         BOOL brake; }
      INPUT {
         BOOL alarm, tunnel, fire; }

   } /* end interface */


   IMPLEMENTATION {
      AUX {
         BOOL decision; }

      LOGIC {
         decision =
            alarm & (~tunnel | ~fire); }

      AUTOMATA {
         brake = decision; }
      MUST {
         fire -> alarm; }
   } /* end implementation */
} /* end system */
```

Example 4: CONTINUOUS section

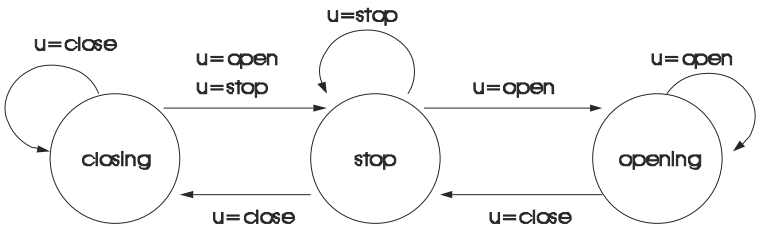


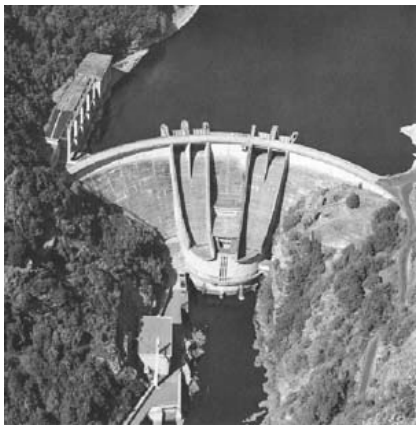
Apply forward difference rule:

$$u(k+1) = u(k) + \frac{T}{C} i(k)$$

```
SYSTEM capacitorD {
   INTERFACE {
      STATE {
         REAL u; }
      PARAMETER {
         REAL R = 1e4;
         REAL C = 1e-4;
         REAL T = 1e-1; }
   } /* end interface */


   IMPLEMENTATION {

      CONTINUOUS {
        u = u - T/C/R*i;  }

   } /* end implementation */
} /* end system */
```

Example 5: AUTOMATA section





```
SYSTEM outflow {
   INTERFACE {
      STATE {
         BOOL closing, stop, opening; }
      INPUT {
         BOOL uclose, uopen, ustop; }
   } /* end of interface */

   IMPLEMENTATION {
```

```
      AUTOMATA {
         closing = (uclose & closing) | (uclose & stop);
         stop    = ustop | (uopen & closing) | (uclose & opening);
         opening = (uopen & stop) | (uopen & opening);  }
```

```
      MUST {
         ~(uclose & uopen);
         ~(uclose & ustop);
         ~(uopen  & ustop); }
   } /* end implementation */
} /* end system */
```

Example 6: MUST section

$$0 \le h \le h_{\max}$$

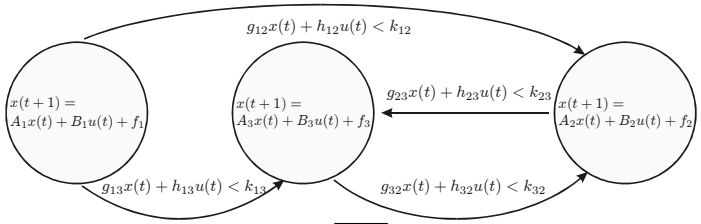
```
SYSTEM watertank {
   INTERFACE {
      STATE {
         REAL h; }
      INPUT {
         REAL Q; }
      PARAMETER {
         REAL hmax = 0.3;
         REAL k    = 1; }
   } /* end interface */


   IMPLEMENTATION {
      CONTINUOUS {
         h = h + k*Q; }
```

```
      MUST {
          h - hmax <= 0;
         -h        <= 0; }
```

```
   } /* end implementation */
} /* end system */
```

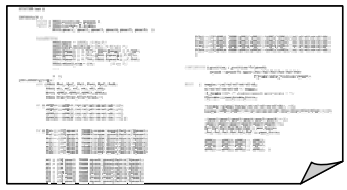
Modeling Flow

Process



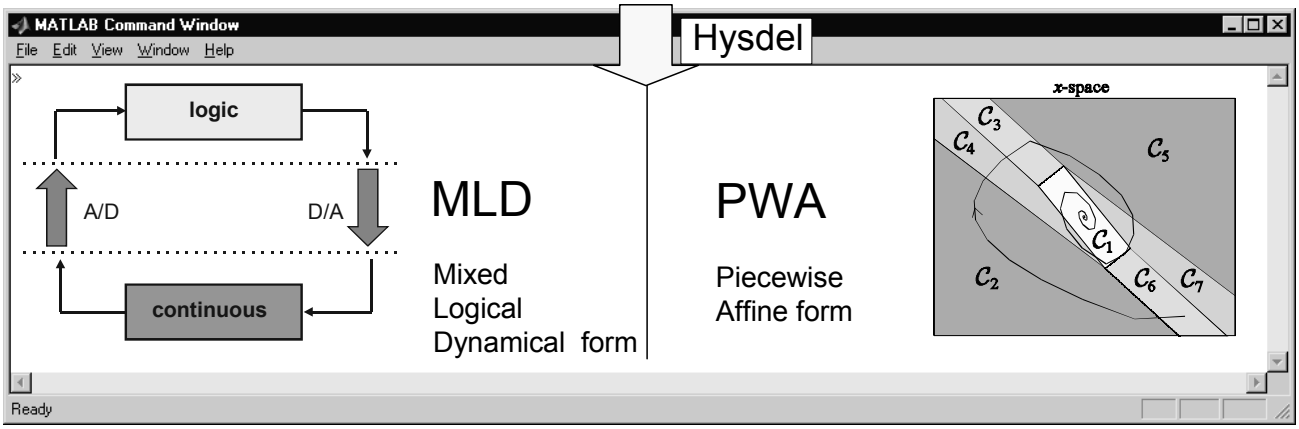
Designer

Text

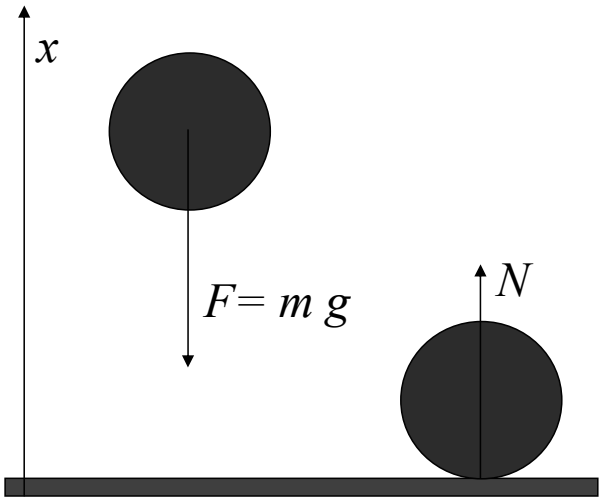


HYSDEL Source

Matlab



Example: Bouncing Ball



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

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

How to model this system in MLD form?

HYSDEL - Bouncing Ball

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

        PARAMETER {
                REAL g=9.8;
                REAL dissipation=.4;  /* 0=elastic, 1=completely anelastic */
                REAL Ts=.05; }
}
IMPLEMENTATION {
        AUX {   REAL z1;
                REAL z2;
                BOOL negative;   }

        AD {    negative = height <= 0; }

        DA {    z1 = {  IF negative THEN height-Ts*velocity
                        ELSE height+Ts*velocity-Ts*Ts*g};
                z2 = {  IF negative THEN -(1-dissipation)*velocity
                        ELSE velocity-Ts*g};  }
        CONTINUOUS {
                height = z1;
                velocity=z2;}
}}
```

System Theory for Hybrid Systems

- Analysis
  - Well-posedness
  - Realization & Transformation
  - Stability
  - Reachability (=Verification)
  - Observability
- Synthesis
  - Control
  - State estimation
  - Identification
  - Modeling language

Well-posedness

Are state and output trajectories defined ?
Uniquely defined ? Persistently defined ?

- MLD well-posedness :

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

$$\delta(t) = F(x(t), u(t))$$
$$z(t) = G(x(t), u(t))$$

$$\{x(t), u(t)\} \rightarrow \{x(t+1)\}$$
$$\{x(t), u(t)\} \rightarrow \{y(t)\}$$

are single valued

Definition 1 *Let $\Omega \subseteq \mathbb{R}^n \times \mathbb{R}^m$ be a set of input+state pairs. A hybrid MLD system is called* well-posed *on Ω, if for all pairs $(x(t), u(t)) \in \Omega$ there exists a solution $x(t+1), y(t), \delta(t), z(t)$ and moreover, $x(t+1), y(t)$ are uniquely determined.*

Numerical test based on mixed-integer programming available

(Bemporad, Morari, *Automatica,* 1999)

Realization and Transformation (State-Space Hybrid Models)

Other Existing Hybrid Models

- Linear complementarity (LC) systems (Heemels, 1999)

$$\begin{aligned} x(t+1) &= Ax(t) + B_1u(t) + B_2w(t) \\ y(t) &= Cx(t) + D_1u(t) + D_2w(t) \\ v(t) &= E_1x(t) + E_2u(t) + E_3w(t) + e_4 \\ 0 &\le v(t) \perp w(t) \ge 0 \end{aligned}$$

Ex: mechanical systems
circuits with diodes etc.

- Extended linear complementarity (ELC) systems
  Generalization of LC systems (De Schutter, De Moor, 2000)

- Min-max-plus-scaling (MMPS) systems
  (De Schutter, Van den Boom, 2000)

$$\begin{aligned} x(t+1) &= M_x(x(t), u(t), d(t)) \\ y(t) &= M_y(x(t), u(t), d(t)) \\ 0 &\ge M_c(x(t), u(t), d(t)) \end{aligned}$$

MMPS function: defined by the grammar

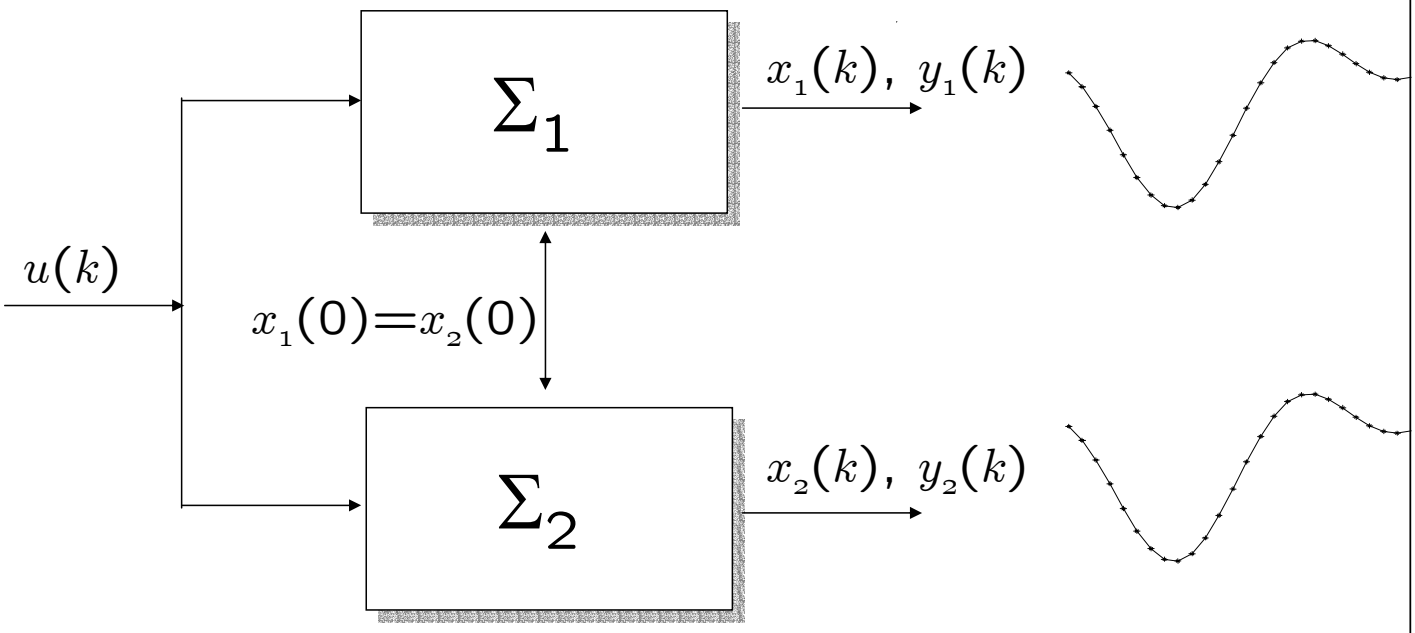
$$M := x_i | \alpha | \max(M_1, M_2) | \min(M_1, M_2) | M_1 + M_2 | \beta M_1$$

Example: $x(t+1) = 2\max(x(t), 0) + \min(-\frac{1}{2}u(t), 1)$

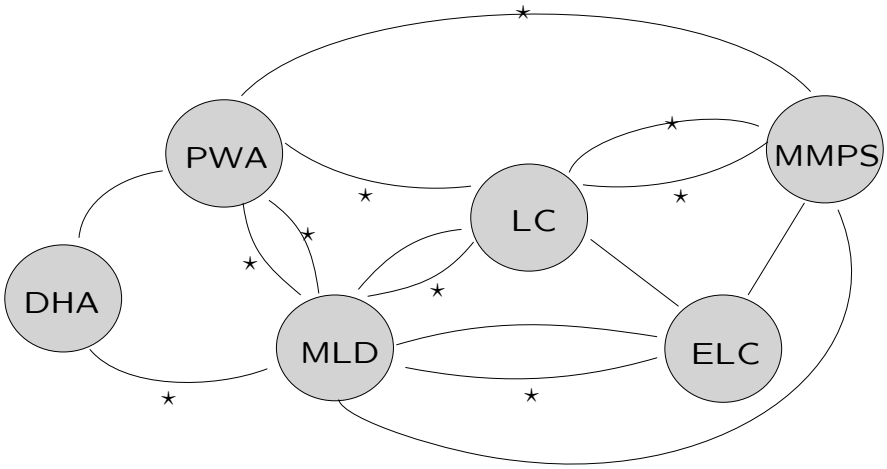
Used for modeling discrete-event systems (t=event counter)

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 Results



Theorem 1 *All the above six classes of discrete-time hybrid models are equivalent (possibly under some additional assumptions, such as boundedness of input and state variables)*

(Heemels, De Schutter, Bemporad, *Automatica,* 2001)
(Torrisi, Bemporad, *IEEE CST,* 2003)
(Bemporad and Morari, *Automatica*, 1999)
(Bemporad, Ferrari-T.,Morari, *IEEETAC*, 2000)

Theoretical properties and analysis/synthesis tools can be transferred from one class to another

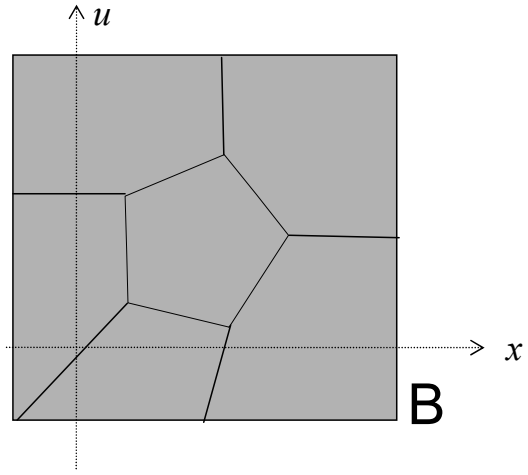
Efficient MLD to PWA Conversion

- Proof is constructive: given an MLD system it returns its equivalent PWA form
- Drawback: it needs the enumeration of all possible combinations of binary states, binary inputs, and δ variables
- Most of such combinations lead to empty regions Ω
- IDEA: use techniques from multiparametric programming to avoid such an enumeration

MLD to PWA Conversion Algorithm

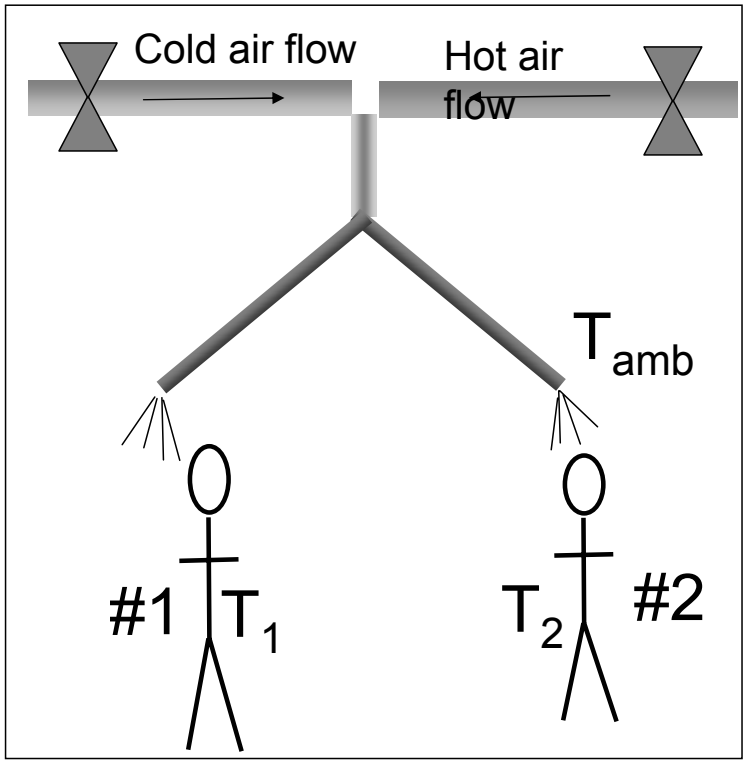
(Bemporad, 2002)

GOAL: split a given set B of states+inputs into polyhedral cells and find the affine dynamics in each cell, therefore determining the PWA system which is equivalent to the given MLD system



Note: the cells Ω_i are embedded in R^{n+m} by treating integer states and integer inputs as continuous vars:
$x_b, u_b \in [0,1] \rightarrow x_b, u_b \in [-1/2,1/2) \cup [1/2,3/2]$

Example: Heat and Cool



Rules of the game:

- #1 turns the heater (air conditioning) 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 the air conditioning on, unless #2 is cold
- Otherwise, heater and air conditioning are off

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

Hybrid HYSDEL Model

```
/* Heat and Cool example
   (C) 2002 by A. Bemporad, Las Vegas, December 9, 2002 */

SYSTEM heatcool {

INTERFACE {
    STATE { REAL T1 [-10,50];
            REAL T2 [-10,50];
        }
    INPUT { REAL Tamb [-10,50];
        }
    PARAMETER {
        REAL Ts = .5; /* sampling time, second
        REAL alpha1 = 1;
        REAL alpha2 = 0.5;
        REAL k1 = .8;
        REAL k2 = .4;
        REAL Thot1 = 30;
        REAL Tcold1 = 15;
        REAL Thot2 = 35;
        REAL Tcold2 = 10;
        REAL Uc = 2; /* cold water flow */
        REAL Uh = 2; /* hot water flow */
    }
}

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 MLD Model

- MLD model

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

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

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

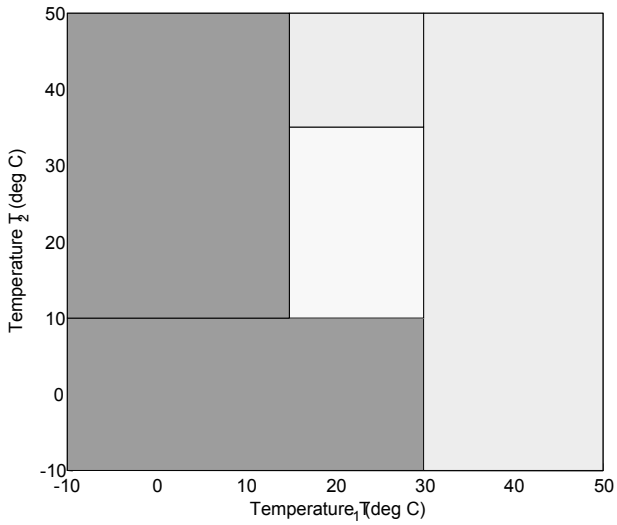
Hybrid PWA Model

- PWA model

$$\begin{array}{rcl} x'(k) & = & A_{i(k)}x(k) + B_{i(k)}u(k) + f_{i(k)} \\ y(k) & = & C_{i(k)}x(k) + D_{i(k)}u(k) + g_{i(k)} \\ i(k) & \text{s.t.} & H_{i(k)}x(k) + J_{i(k)}u(k) \leq K_{i(k)} \end{array}$$

- 2 continuous states:
  (temperatures T_1,T_2)
- 1 continuous input:
  (room temperature T_{amb})
- 5 polyhedral regions
  (partition does not depend on input)

Computation time: 0.72 s in Matlab

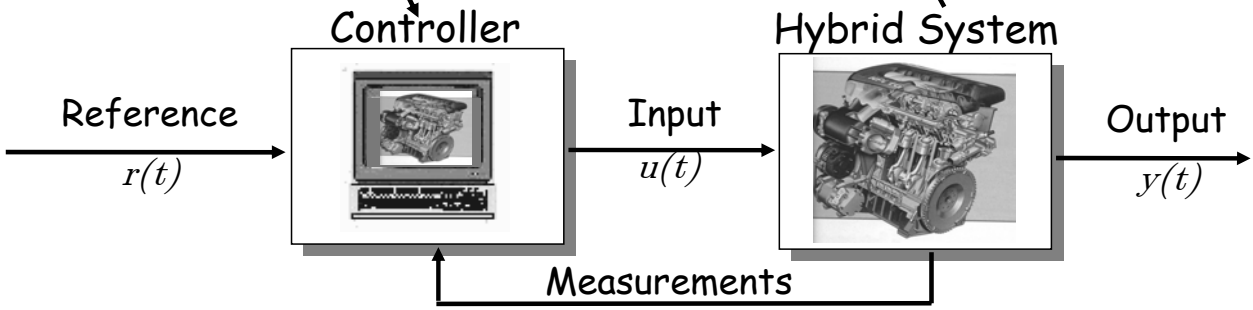


$$\begin{array}{rcl} u_{\text{hot}} & = & 0 \\ u_{\text{cold}} & = & 0 \end{array} \qquad \begin{array}{rcl} u_{\text{hot}} & = & 0 \\ u_{\text{cold}} & = & \bar{U}_C \end{array} \qquad \begin{array}{rcl} u_{\text{hot}} & = & \bar{U}_H \\ u_{\text{cold}} & = & 0 \end{array}$$

Controller Synthesis

Model Predictive Control of Hybrid Systems

$$x(t+1) = Ax(t) + B_1u(t) + B_2\delta(t) + B_3z(t)$$
$$y(t) = Cx(t) + D_1u(t) + D_2\delta(t) + D_3z(t)$$
$$E_2\delta(t) + E_3z(t) \leq E_4x(t) + E_1u(t) + E_5$$



- MODEL: a model of the plant is needed to predict the future behavior of the plant
- PREDICTIVE: optimization is based on the predicted future evolution of the plant
- CONTROL: control complex constrained multivariable plants

Receding Horizon Philosophy

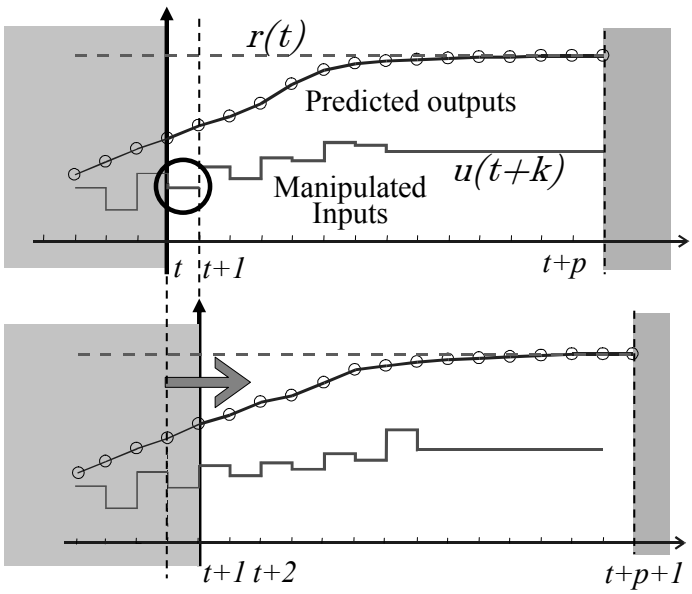
- At time t :

  Solve an optimal control problem over a finite future horizon p :

  - minimize $|y - r| + \rho|u|$

  – subject to constraints

$$u_{min} \leq u \leq u_{max}$$
$$y_{min} \leq y \leq y_{max}$$



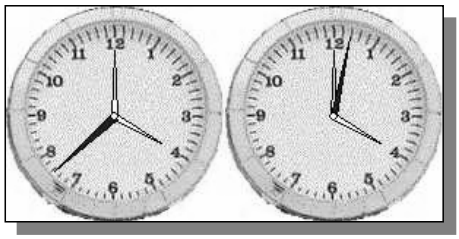
- Only apply the first optimal move $u^*(t)$

- Get new measurements, and repeat the optimization at time $t+1$

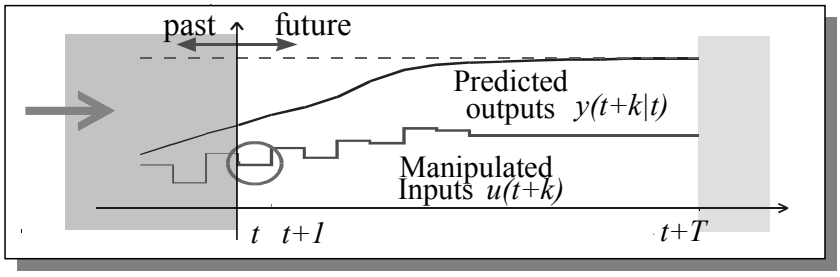
Advantage of on-line optimization: *FEEDBACK!*

Receding Horizon - Example

MPC is like playing chess !



MPC for Hybrid Systems



Model Predictive (MPC) Control

- At time t solve with respect to $U \triangleq \{u(t), u(t+1) \ldots, u(t+T-1)\}$ the finite-horizon open-loop, optimal control problem:

$$\min_U J(U, x(t), r) \triangleq \sum_{k=0}^{T-1} \|Q(y(t+k+1|t) - r)\| + \|R(u(t+k) - u_r)\|$$
$$+ \sigma \left(\|\delta(t+k|t) - \delta_r\| + \|z(t+k|t) - z_r\| + \|x(t+k|t) - x_r\| \right)$$

$$\text{subj. to} \begin{cases} \text{MLD model} \\ x(t|t) = x(t) \\ x(t+T|t) = x_r \end{cases}$$

- Apply only $u(t) = u^*(t)$ (discard the remaining optimal inputs)
- Repeat the whole optimization at time $t+1$

Closed-Loop Stability

Theorem 1 *Let $(x_r, u_r, \delta_r, z_r)$ be the equilibrium values corresponding to the set point r, and 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 MPC controller stabilizes the MLD system*

$$\lim_{t\to\infty} y(t) = r$$
$$\lim_{t\to\infty} u(t) = u_r$$

$\lim_{t\to\infty} x(t) = x_r$, $\lim_{t\to\infty} \delta(t) = \delta_r$, $\lim_{t\to\infty} z(t) = z_r$, *and all constraints are fulfilled.*

(Bemporad, Morari 1999)

Proof: Easily follows from standard Lyapunov arguments

Stability Proof

- Assume we set the terminal constraint $x(T) = x_r$ in the optimal control problem
- Let $\mathcal{U}_t^*$ denote the optimal control sequence $\{u_t^*(0), \ldots, u_t^*(T-1)\}$
- Let $V(t) \triangleq J(\mathcal{U}_t^*, x(t))$=value function
- By construction, $\mathcal{U}_1 = \{u_t^*(1), \ldots, u_t^*(T-1), u_r\}$ is feasible @$t+1$
- Hence,

$$V(t+1) \le J(\mathcal{U}_1, x(t+1)) = V(t) - \|y(t) - r\|_Q - \|u(t) - u_r\|_R - \sigma(\|\delta(t) - \delta_r\| - \|z(t) - z_r\| - \|x(t) - x_r\|)$$

- Hence $V(t)$ is decreasing and lower-bounded by 0 $\Rightarrow \exists V_\infty = \lim_{t\to\infty} V(t)$ $\Rightarrow V(t+1) - V(t) \to 0$
- Hence, $\|y(t) - r\|_Q \to 0$,$\|u(t) - u_r\|_R \to 0$, $\ldots$, $\|x(t) - x_r\| \to 0$

Hybrid MPC - Example

Switching System:

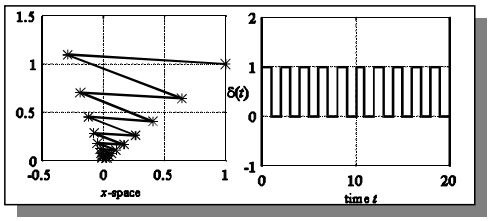
$$x(t+1) = 0.8 \begin{bmatrix} \cos\alpha(t) & -\sin\alpha(t) \\ \sin\alpha(t) & \cos\alpha(t) \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(t)$$

$$y(t) = [\,0 \quad 1\,]\, x(t)$$

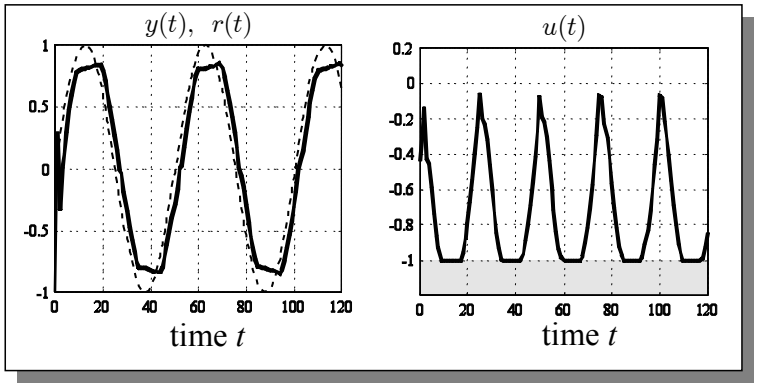
$$\alpha(t) = \begin{cases} \frac{\pi}{3} \text{ if } [1\ \ 0]x(t) \geq 0 \\ -\frac{\pi}{3} \text{ if } [1\ \ 0]x(t) < 0 \end{cases}$$

Constraint: $\quad -1 \leq u(t) \leq 1$

Open loop:



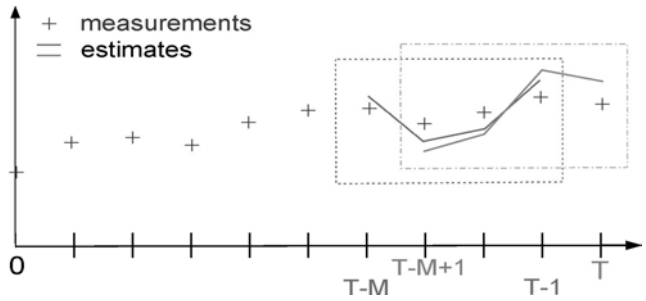
Closed loop:



State Estimation / Fault Detection

State Estimation / Fault Detection

- Problem: given past output measurements and inputs, estimate the current state/faults
- Solution: Use Moving Horizon Estimation for MLD systems (dual of MPC)



Augment the MLD model with:

- Input disturbances $\xi \in \mathbf{R}^n$
- Output disturbances $\zeta \in \mathbf{R}^p$

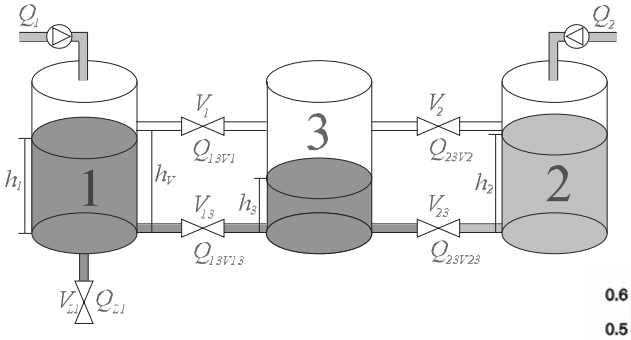
At each time t solve the optimization: $\min \sum_{k=1}^{T} \|\hat{y}(t-k|t) - y(t-k)\|^2 + \dots$ and get estimates $\hat{x}(t)$

➡ MHE optimization = MIQP (Bemporad, Mignone, Morari, ACC 1999)

➡ Convergence can be guaranteed (Ferrari-T., Mignone, Morari, 2002)

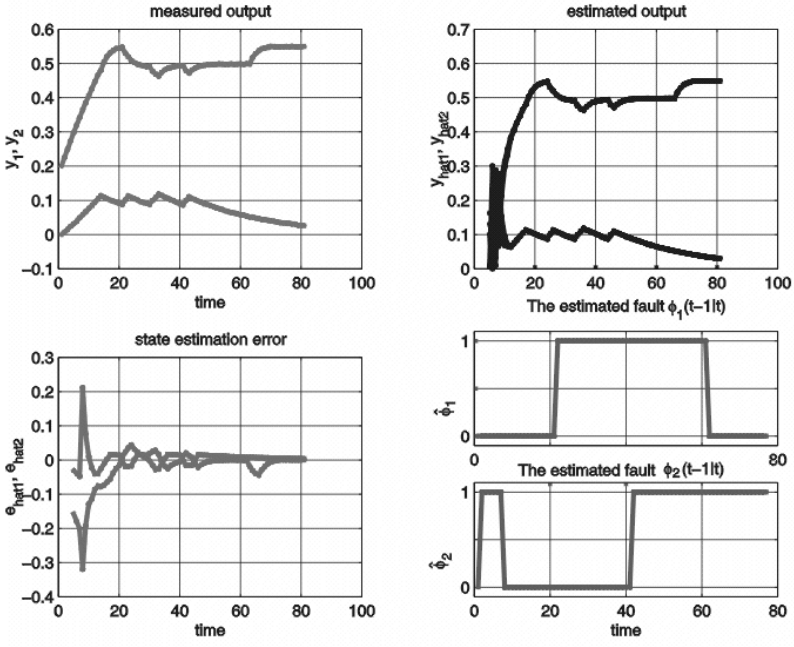
Fault detection: augment MLD with unknown **binary** distubances $\phi \in \{0,1\}^p$

Example: Three Tank System

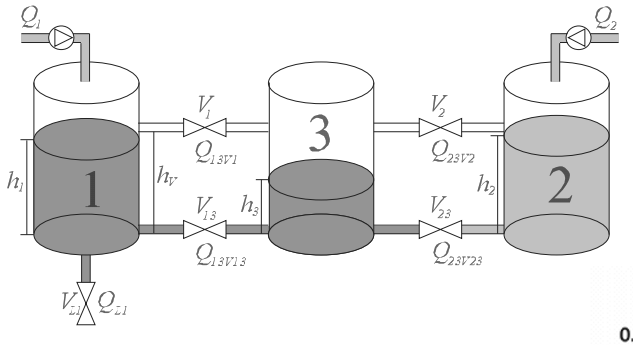


COSY Benchmark problem, ESF

- ϕ_1 : leak in tank 1 for $20s \le t \le 60s$
- ϕ_2 : valve V_1 blocked for $t \ge 40s$



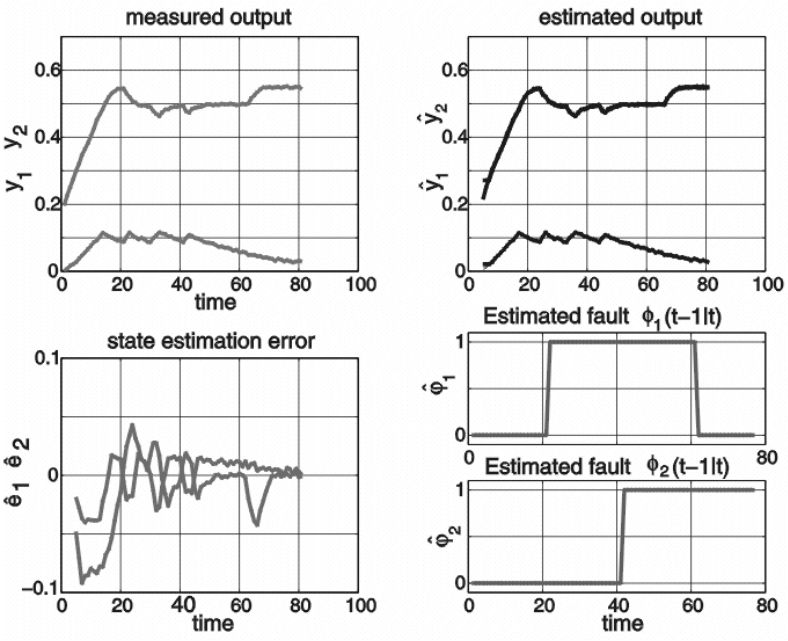
Example: Three Tank System



COSY Benchmark problem, ESF

- ϕ_1 : leak in tank 1 for $20s \leq t \leq 60s$
- ϕ_2 : valve V_1 blocked for $t \geq 40s$
- Add logic constraint

$$[h_1 \leq h_v] \Rightarrow \phi_2 = 0$$



Optimal Control of Hybrid Systems: Computational Aspects

MIQP Formulation of MPC

(Bemporad, Morari, 1999)

$$\min \sum_{k=0}^{T-1} \left\{ y'(t+k+1|t) Q y(t+k+1|t) + u'(t+k) R u(t+k) \right\}$$
$$\text{s.t.} \; \text{MLD dynamics}$$

- Set $\xi \triangleq [u(t), \ldots, u(t+T-1), \delta(t|t), \ldots, \delta(t+T-1|t), z(t|t), \ldots, z(t+T-1|t)]$

$$\min_{\xi} J(\xi, x(t)) = \frac{1}{2}\xi' H \xi + x'(t) F \xi + \frac{1}{2} x'(t) Y x(t)$$
$$\text{s.t.} \; G\xi \leq W + Sx(t)$$

Mixed Integer Quadratic Program (MIQP)

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

MILP Formulation of MPC

(Bemporad, Borrelli, Morari, 2000)

$$\min \sum_{k=0}^{T-1} \|Q y(t+k+1|t)\|_\infty + \|R u(t+k)\|_\infty$$
$$\text{s.t.} \; \text{MLD dynamics}$$

- Introduce slack variables: $\min |x| \Rightarrow \begin{array}{ll} \min & \epsilon \\ \text{s.t.} & \epsilon \geq x \\ & \epsilon \geq -x \end{array}$

$$\begin{array}{rcl} \epsilon_k^x & \geq & \|Qy(t+k|t)\|_\infty \\ \epsilon_k^u & \geq & \|Ru(t+k)\|_\infty \end{array} \Rightarrow \begin{array}{rcll} \epsilon_k^x & \geq & [Qy(t+k|t)]_i & i=1,\ldots,p, \; k=1,\ldots,T-1 \\ \epsilon_k^x & \geq & -[Qy(t+k|t)]_i & i=1,\ldots,p, \; k=1,\ldots,T-1 \\ \epsilon_k^u & \geq & [Ru(t+k))]_i & i=1,\ldots,m, \; k=0,\ldots,T-1 \\ \epsilon_k^u & \geq & -[Ru(t+k))]_i & i=1,\ldots,m, \; k=0,\ldots,T-1 \end{array}$$

- Set $\xi \triangleq [\epsilon_1^x, \ldots, \epsilon_{N_y}^x, \epsilon_1^u, \ldots, \epsilon_{T-1}, U, \delta, z]$

Mixed Integer Linear Program (MILP)

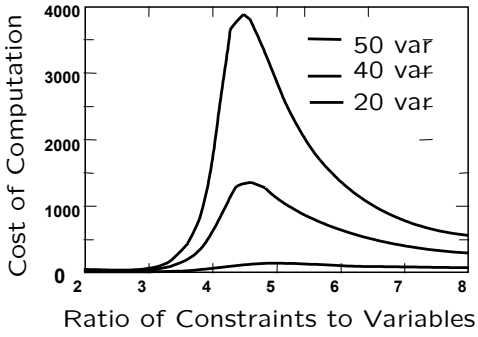
$$\min_{\xi} J(\xi, x(t)) = \sum_{k=0}^{T-1} \epsilon_i^x + \epsilon_i^u$$
$$\text{s.t.} \; G\xi \leq W + Sx(t)$$

Mixed-Integer Program Solvers

- Mixed-Integer Programming is *NP*-hard

  Phase transitions have been found in computationally hard problems.

  BUT



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

- General purpose Branch & Bound/Branch & Cut solvers available for MILP and MIQP (Ilog CPLEX, XPRESS-MP, Sahinidis, ...)
  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

Good for large sampling times (e.g., 1 h) / expensive hardware ...

... but not for fast sampling (e.g. 10 ms) / cheap hardware !

Hybrid Control Example: Heat Exchange

Hybrid Control - An Example

(Hedlund and Rantzer, CDC1999)

$$\begin{bmatrix}\dot{T}_1\\ \dot{T}_2\end{bmatrix} = \begin{bmatrix}-1 & 0\\ 0 & -2\end{bmatrix}\begin{bmatrix}T_1\\ T_2\end{bmatrix} + B_i u_0$$

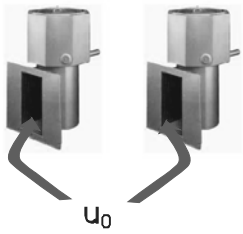
$$B_i = \begin{cases}\begin{bmatrix}1\\0\end{bmatrix} & \text{if first furnace heated}\\ \begin{bmatrix}0\\1\end{bmatrix} & \text{if second furnace heated}\\ \begin{bmatrix}0\\0\end{bmatrix} & \text{if no heating}\end{cases}$$

- Objective:
  - Control the temperature to a given set-point $T_1,\ T_2$
- Constraints:
  - Only three operation modes:
    1- Heat only the first furnace
    2- Heat only the second furnace
    3- Do not heat any furnaces

Amount of heating power u_o is constant

Alternate Heating of Two Furnaces

- HYSDEL model:



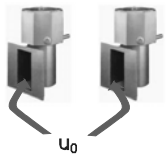

```
/* HEAT EXCHANGE model  - (C) 2001 by A. Bemporad, Zurich, March 13, 2001 */

SYSTEM furnaces {

INTERFACE {
        STATE {
                REAL t1,t2,u0; }
        INPUT {
                BOOL heat1,heat2; }
        PARAMETER {
                REAL Ts=0.08; /* sampling time, seconds */
                REAL Bd1=0.07688365361336; /* discretization of B matrix, heat 1 active */
                REAL Bd2=0.07392810551689; /* discretization of B matrix, heat 2 active */
                REAL A11=.9231; /* discretization of A matrix */
                REAL A22=.8521; /* discretization of A matrix */

                REAL u0max=10; /* upperbound on u0 */
                REAL e = 1e-6; /* precision for strict inequalities */  }
}

IMPLEMENTATION {
                AUX {REAL z1,z2; }

                DA {    z1 = {IF heat1 THEN Bd1*u0 [Bd1*u0max,0,e]};
                        z2 = {IF heat2 THEN Bd2*u0 [Bd2*u0max,0,e]}; }

                CONTINUOUS { t1 = A11*t1+z1;
                             t2 = A22*t2+z2;
                             u0 = u0; }

                MUST   {  ~heat1 | ~heat2;  /* heat1 and heat2 cannot be both active */ }
}
}
```

- MLD model:

$$x(t+1) = \begin{bmatrix}0.9231 & 0 & 0\\ 0 & 0.8521 & 0\\ 0 & 0 & 1.0000\end{bmatrix} x(t) + \begin{bmatrix}1 & 0\\ 0 & 1\\ 0 & 0\end{bmatrix} z(t)$$

$$\begin{bmatrix}-1 & 0\\ 1 & 0\\ -1 & 0\\ 1 & 0\\ 0 & -1\\ 0 & 1\\ 0 & -1\\ 0 & 1\\ 0 & 0\end{bmatrix} z(t) \square \begin{bmatrix}-0.7688 & 0\\ 0 & 0\\ 0 & 0\\ 0.7688 & 0\\ 0 & -0.7393\\ 0 & 0\\ 0 & 0\\ 0 & 0.7393\\ -1.0000 & -1.0000\end{bmatrix} u(t) + \begin{bmatrix}-0.0769\\ 0.0769\\ 0\\ 0\\ -0.0739\\ 0.0739\\ 0\\ 0\\ 0\end{bmatrix} x_3(t) + \begin{bmatrix}0.7688\\ 0\\ 0\\ 0\\ 0.7393\\ 0\\ 0\\ 0\\ 1.0000\end{bmatrix}$$

Alternate Heating of Two Furnaces



- Performance index

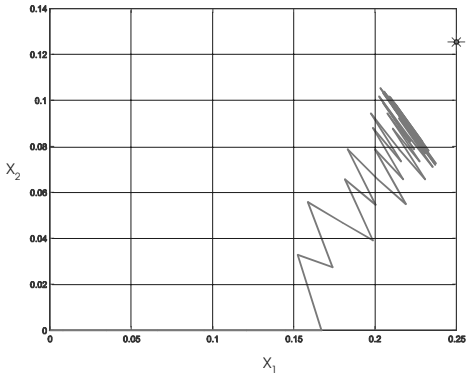
$$\min_{\{u(t),u(t+1),u(t+2)\}} \sum_{k=0}^{2} \|R(u(t+k+1)-u(t+k))\|_\infty + \|Q(x(t+k|t)-x_e)\|_\infty$$

- Constraints

$$\begin{array}{rcl} -1 & \leq x_1 \leq & 1 \\ -1 & \leq x_2 \leq & 1 \end{array}$$

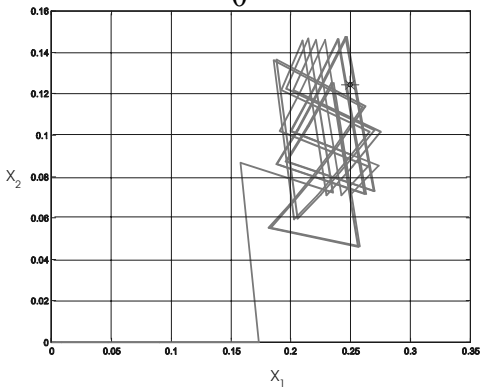
Closed-Loop Behavior

u_0=0.4



Set point cannot be reached

u_0=0.8



Set point is reached

- Computational complexity of on-line MILP

  (Bemporad, Borrelli, Morari, 2000)

Linear constraints	168
Continuous variables	33
Binary variables	6
Parameters	3
Time to solve MILP (av.)	1,09 s

On-Line vs. Off-Line Optimization

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

$$\text{subj. to} \begin{cases} \text{MLD model} \\ x(t|t) = x(t) \\ x(t+T|t) = 0 \end{cases}$$

- On-line optimization: given $x(t)$, solve the problem at each time step t

Mixed-Integer Linear Program (MILP)

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

$$\min_\xi J(\xi, x(t)) \triangleq f'\xi$$
$$\text{s.t.} G\xi \leq W + Fx(t)$$

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

Explicit Form of Model Predictive Control

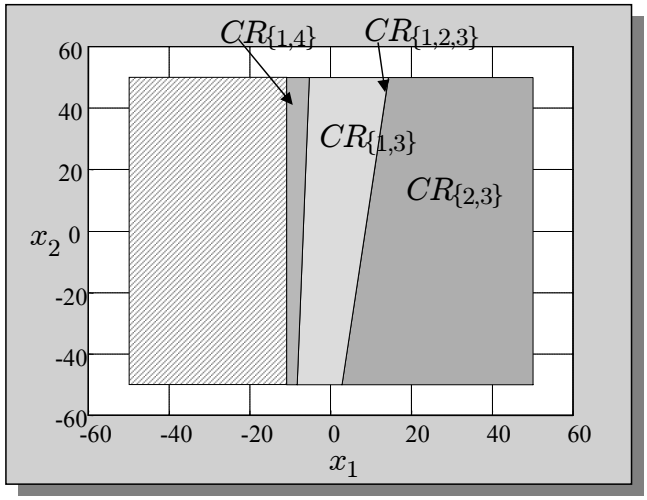
via Multiparametric Programming

Example of Multiparametric Solution

Multiparametric LP ($\xi \in \mathbf{R}^2$)

$$\min_{\xi} \quad -3\xi_1 - 8\xi_2$$

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



$$\xi(x) = \begin{cases} \begin{bmatrix} 1.00 & 0 \\ -33.00 & 55.00 \end{bmatrix} x + \begin{bmatrix} -13.00 \\ 13.00 \end{bmatrix} & \text{if} \quad \begin{bmatrix} 1.00 & 0.00 \\ -1.00 & 0.00 \\ 1.00 & -50.00 \\ 0.00 & 50.00 \end{bmatrix} x \leq \begin{bmatrix} 1.00 \\ 1.00 \\ 60.00 \\ 0.00 \end{bmatrix} & \text{CR}^{\{1,4\}} \\ \begin{bmatrix} 1.00 & 0.00 \\ -1.00 & 21.00 \\ 1.00 & -50.00 \\ 0.00 & 50.00 \end{bmatrix} x \leq \begin{bmatrix} 1.00 \\ 1.00 \\ 1.00 \\ 1.00 \end{bmatrix} & \text{CR}^{\{1,3\}} \\ \begin{bmatrix} 1.00 & 0.00 \\ -1.00 & 10.00 \\ 0.00 & -50.00 \\ 0.00 & 50.00 \end{bmatrix} x \leq \begin{bmatrix} -1.00 \\ 1.00 \\ 1.00 \\ 1.00 \end{bmatrix} & \text{CR}^{\{2,3\}} \end{cases}$$

Linear MPC

- Linear Model:

$$\begin{aligned} x(t+1) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) \end{aligned}$$

$$\Delta u(t) = u(t) - u(t-1)$$

input increments
(for integral action)

- Prediction:

$$y(t+k|t) = CA^k x(t) + \sum_{i=0}^{k-1} CA^i B u(t+k-1-i)$$

$$u(t+k) = u(t-1) + \sum_{i=0}^{k-1} \Delta u(t+i)$$

Linear MPC

Quadratic Performance Index $\qquad U \triangleq \{\Delta u(t), \ldots, \Delta u(t+m-1)\}$

$$\min_U \sum_{k=0}^{p-1} \| W^y \left[y(t+k+1|t) - r(t) \right] \|^2 + \| W^{\Delta u} \Delta u(t+k) \|^2 + W^u \left[u(t+k) - u_{\text{target}}(t) \right] \|^2$$

$$\min_U J(U, x(t)) = \tfrac{1}{2} U' H U + [x'(t) \;\; r'(t)] F U + \tfrac{1}{2} [x'(t) \;\; r'(t)] Y \begin{bmatrix} x(t) \\ r(t) \end{bmatrix}$$

$$\text{subject to } GU \leq W + K \begin{bmatrix} x(t) \\ r(t) \end{bmatrix}$$

(convex) QUADRATIC PROGRAM (QP)

$$\begin{aligned} y_{min} \leq y(t+k|t) \leq y_{max}, \quad & k = 1, \ldots, p \\ u_{min} \leq u(t+k) \leq u_{max}, \quad & k = 0, 1, \ldots, m-1 \\ \Delta u_{min} \leq \Delta u(t+k) \leq \Delta u_{max}, \quad & k = 0, 1, \ldots, m-1 \end{aligned}$$

Constraints

Multiparametric Quadratic Programming

(Bemporad et al., 2002)

$$\begin{array}{ll} \min\limits_U & \frac{1}{2} U' H U + x F' U + \frac{1}{2} x Y x \\ \text{subj. to} & GU \leq W + Sx, \end{array}$$

$U \triangleq [u_0' \;\; \ldots \;\; u_{T-1}']'$

$U \in \mathbb{R}^r$, $r \triangleq n_u T$

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

- Objective: solve the QP **for all** $x \in X \subseteq \mathbb{R}^n$

Linearity of the Solution

$x_0 \in X$ ⇨ solve QP to find $U^*(x_0)$, $\lambda^*(x_0)$

⇨ identify active constraints at $U^*(x_0)$

⇨ form matrices $\tilde{G}$, $\tilde{W}$, $\tilde{S}$ by collecting active constraints $\tilde{G}U^*(x_0) - \tilde{W} - \tilde{S}x_0 = 0$

KKT optimality conditions:

(1) $HU + Fx + G'\lambda = 0$, (2) $\tilde{G}U - \tilde{W} - \tilde{S}x = 0$
(3) $\lambda_i(G^iU - W^i - s^ix) = 0$, (4) $\hat{G}U \leq \hat{W} + \hat{S}x = 0$
(5) $\tilde{\lambda}_i \geq 0$, $\hat{\lambda}_i = 0$

From (1) : $U = -H^{-1}(Fx + \tilde{G}'\tilde{\lambda})$

From (2) :

$$\tilde{\lambda}(x) = -(\tilde{G}H^{-1}\tilde{G}')^{-1}(\tilde{W} + (\tilde{S} + \tilde{G}H^{-1}F)x).$$
$$U(x) = H^{-1}[\tilde{G}'(\tilde{G}H^{-1}\tilde{G}')^{-1}(\tilde{W}+(\tilde{S}+\tilde{G}H^{-1}F)x)-Fx]$$

⇨ In some neighborhood of x_0, λ and U are explicit affine functions of x !

Determining a Critical Region

- Impose primal and dual feasibility:

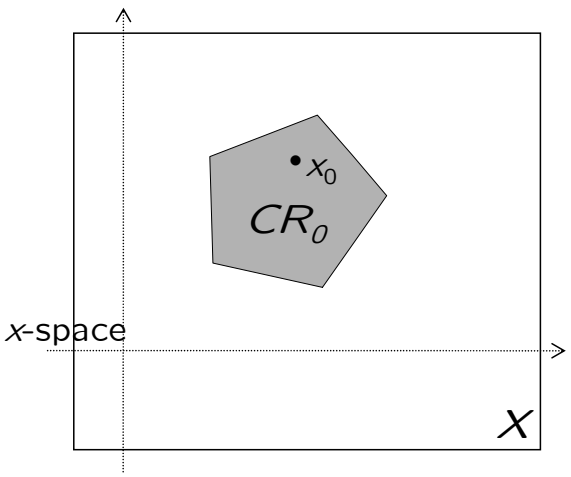
$$\hat{G}U(x) \leq \hat{W} + \hat{S}x$$
$$\tilde{\lambda}(x) \geq 0$$

⇨ linear inequalities in x !

- Remove redundant constraints

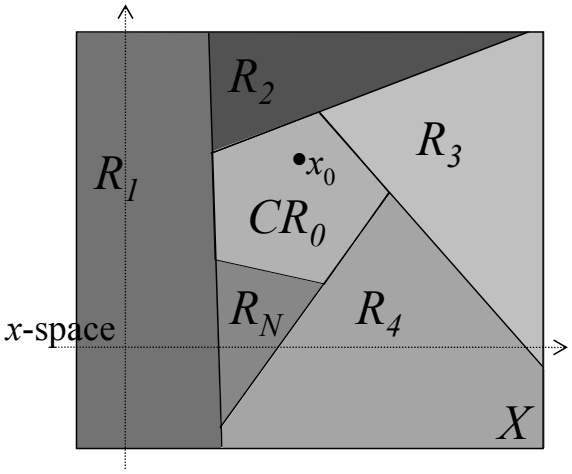
⇨ critical region CR_0

$$CR_0 = \{\mathcal{A}x \leq \mathcal{B}\}$$



- CR_0 is the set of all and only parameters x for which $\tilde{G}, \tilde{W}, \tilde{S}$ is the optimal combination of active constraints at the optimizer

Multiparametric QP



$$CR_0 = \{\mathcal{A}x \leq \mathcal{B}\}$$

$$R_i = \{x \in X : \ \mathcal{A}^i x > \mathcal{B}^i, \ \mathcal{A}^j z \leq \mathcal{B}^j, \ \forall j < i\}$$

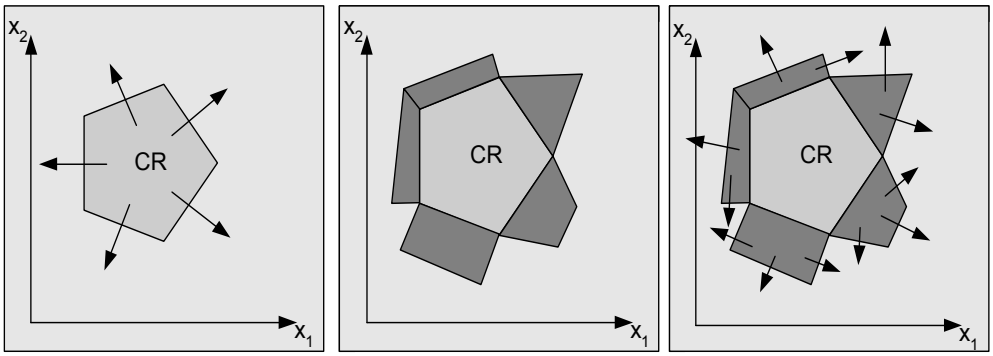
Theorem: $\{CR_0, R_1, \ldots, R_N\}$ is a partition of $X \subseteq \mathbf{R}^n$

Proceed iteratively: for each region R_i repeat the whole procedure with $X \leftarrow R_i$

The recursive algorithm terminates after a finite number of steps, because the number of combinations of active constraints is finite

Mp-QP – More efficient method

(Tøndel, Johansen, Bemporad, 2003)



The active set of a neighboring region is found by using the active set of the current region + knowledge of the type of hyperplane we are crossing:

$\widehat{G}^i U(x) \leq \widehat{W}^i + \widehat{S}^i x \Rightarrow$ The corresponding constraint is **added** to the active set

$\tilde{\lambda}_j(x) \geq 0 \quad \Rightarrow$ The corresponding constraint is **withdrawn** from the active set

Multiparametric QP

Theorem 1 *Consider a multi-parametric quadratic program with $H \succ 0$. The set X^* of parameters x for which the problem is feasible is a polyhedral set, the value function $J^* : X^* \mapsto \mathbb{R}$ is piecewise quadratic, convex and continuous and the optimizer $U^* : X^* \mapsto \mathbb{R}^r$ is piecewise affine and continuous.*

$$U^*(x) = \arg\min_U \ \tfrac{1}{2}U'HU + x'F'U \quad \text{subj. to} \quad Gz \leq W + Sx$$

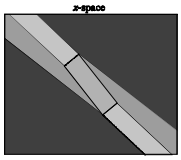
continuous, piecewise affine

$$V^*(x) = \tfrac{1}{2}x'Yx + \min_U \ \tfrac{1}{2}U'HU + x'F'U \quad \text{subj. to} \quad Gz \leq W + Sx$$

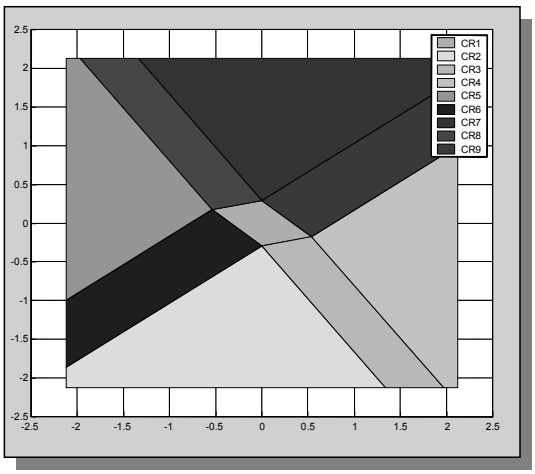
convex, continuous, piecewise quadratic, C^1 (if no degeneracy)

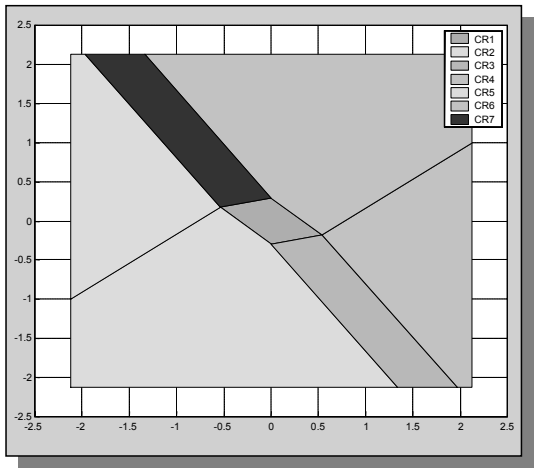
Corollary: The linear MPC controller is a continuous piecewise affine function of the state

$$u(x) = \begin{cases} F_1x + G_1 & \text{if} \quad H_1x \leq K_1 \\ \vdots & \vdots \\ F_Nx + G_N & \text{if} \quad H_Nx \leq K_N \end{cases}$$



Complexity Reduction





$$U(x(0)) \triangleq [u_0(x(0)), \ldots, u_{T-1}(x(0))]$$

Regions where the first component of the solution is the same can be joined (when their union is convex). (Bemporad, Fukuda, Torrisi, *Computational Geometry*, 2001)

Double Integrator Example

- System: $y(t) = \frac{1}{s^2}\ u(t)$ ⟹ (sampling + ZOH, *T=1* s)

$$x(t+1) = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(t)$$

$$y(t) = \begin{bmatrix} 1 & 0 \end{bmatrix} x(t)$$

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

- Control objective: minimize $\sum_{t=0}^{\infty} y'(t)y(t) + \frac{1}{100}u^2(t)$

$$u_{t+k} = K_{LQ}\ x(t+k|t)\ \ \forall k \ge N_u$$

- Optimization problem: for N_u=2

$$H = \begin{bmatrix} 0.8365 & 0.3603 \\ 0.3603 & 0.2059 \end{bmatrix},\ F = \begin{bmatrix} 0.4624 & 1.2852 \\ 0.1682 & 0.5285 \end{bmatrix}$$

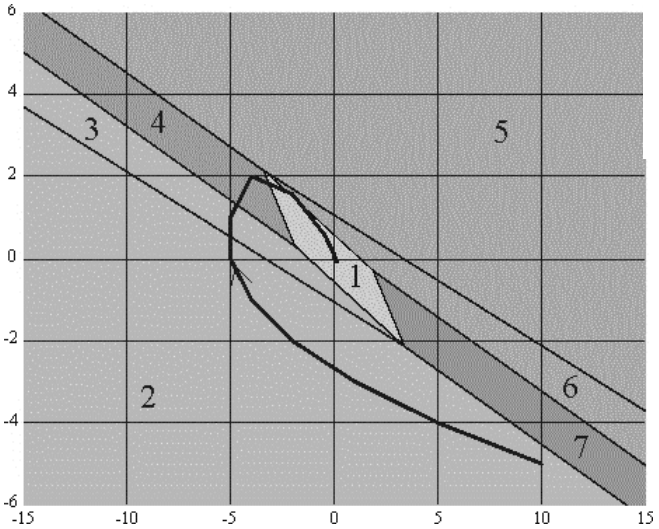
(cost function is normalized by max λ(H))

$$G = \begin{bmatrix} 1 & 0 \\ -1 & 0 \\ 0 & 1 \\ 0 & -1 \end{bmatrix},\ W = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix},\ S = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$$

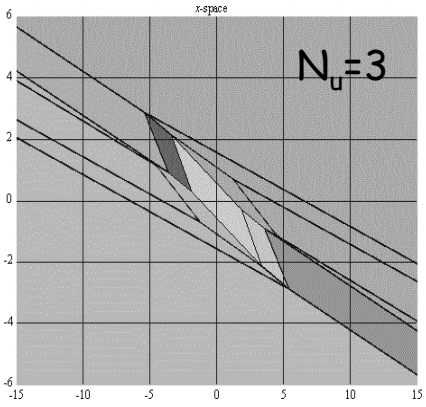
mp-QP solution

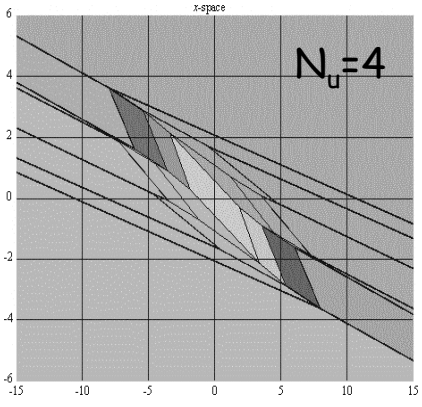
N_u=2

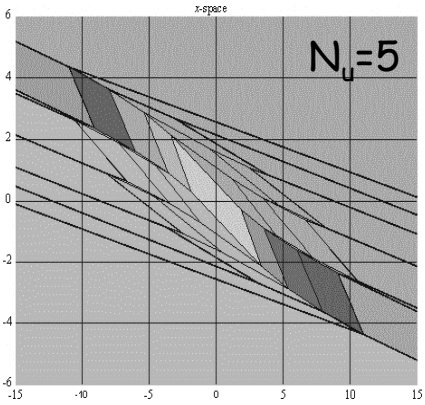
$$u(x) = \begin{cases} \begin{bmatrix} -0.8166 & -1.7499 \end{bmatrix} x & \text{if } \begin{bmatrix} -0.8166 & -1.7499 \\ 0.8166 & 1.7499 \\ 0.6124 & 0.4957 \\ -0.6124 & -0.4957 \end{bmatrix} x \le \begin{bmatrix} 1.0000 \\ 1.0000 \\ 1.0000 \\ 1.0000 \end{bmatrix} & \text{(Region \#1)} \\ 1.0000 & \text{if } \begin{bmatrix} 0.3864 & 1.0738 \\ 0.2970 & 0.9333 \end{bmatrix} x \le \begin{bmatrix} -1.0000 \\ -1.0000 \end{bmatrix} & \text{(Region \#2)} \\ 1.0000 & \text{if } \begin{bmatrix} 0.9712 & 2.6991 \\ -0.2970 & -0.9333 \\ 0.8166 & 1.7499 \end{bmatrix} x \le \begin{bmatrix} -1.0000 \\ 1.0000 \\ -1.0000 \end{bmatrix} & \text{(Region \#3)} \\ \begin{bmatrix} -0.5528 & -1.5364 \end{bmatrix} x + 0.4308 & \text{if } \begin{bmatrix} -0.9712 & -2.6991 \\ 0.3864 & 1.0738 \\ 0.6124 & 0.4957 \end{bmatrix} x \le \begin{bmatrix} 1.0000 \\ 1.0000 \\ -1.0000 \end{bmatrix} & \text{(Region \#4)} \\ -1.0000 & \text{if } \begin{bmatrix} -0.3864 & -1.0738 \\ -0.2970 & -0.9333 \end{bmatrix} x \le \begin{bmatrix} -1.0000 \\ -1.0000 \end{bmatrix} & \text{(Region \#5)} \\ -1.0000 & \text{if } \begin{bmatrix} -0.9712 & -2.6991 \\ 0.2970 & 0.9333 \\ -0.8166 & -1.7499 \end{bmatrix} x \le \begin{bmatrix} -1.0000 \\ 1.0000 \\ -1.0000 \end{bmatrix} & \text{(Region \#6)} \\ \begin{bmatrix} -0.5528 & -1.5364 \end{bmatrix} x - 0.4308 & \text{if } \begin{bmatrix} -0.3864 & -1.0738 \\ 0.9712 & 2.6991 \\ -0.6124 & -0.4957 \end{bmatrix} x \le \begin{bmatrix} 1.0000 \\ 1.0000 \\ -1.0000 \end{bmatrix} & \text{(Region \#7)} \end{cases}$$

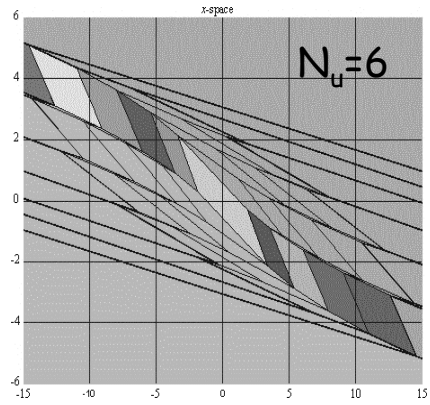


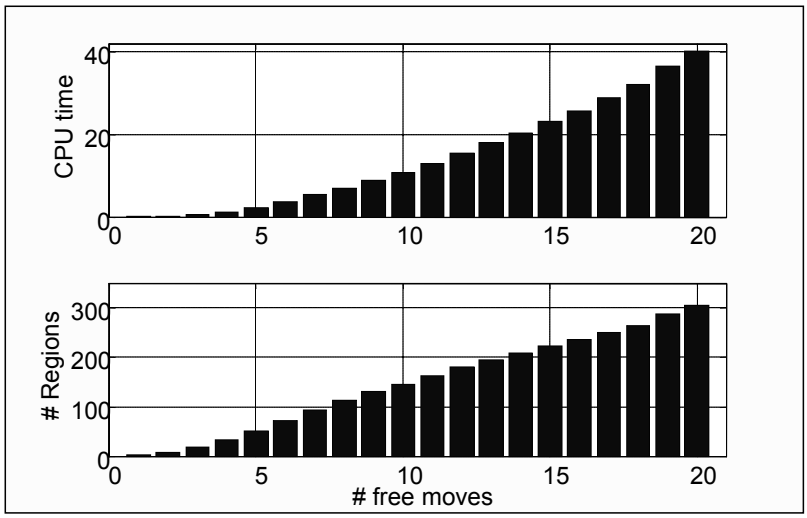
Complexity











Extensions

- Tracking of reference $r(t)$: $\delta u(t) = F(x(t), u(t-1), r(t))$

- Rejection of measured disturbance $v(t)$: $\delta u(t) = F(x(t), u(t-1), v(t))$

- Soft constraints: $u(t) = F(x(t))$

  $y_{\min} - \epsilon \leq y(t+k|t) \leq y_{\max} + \epsilon$

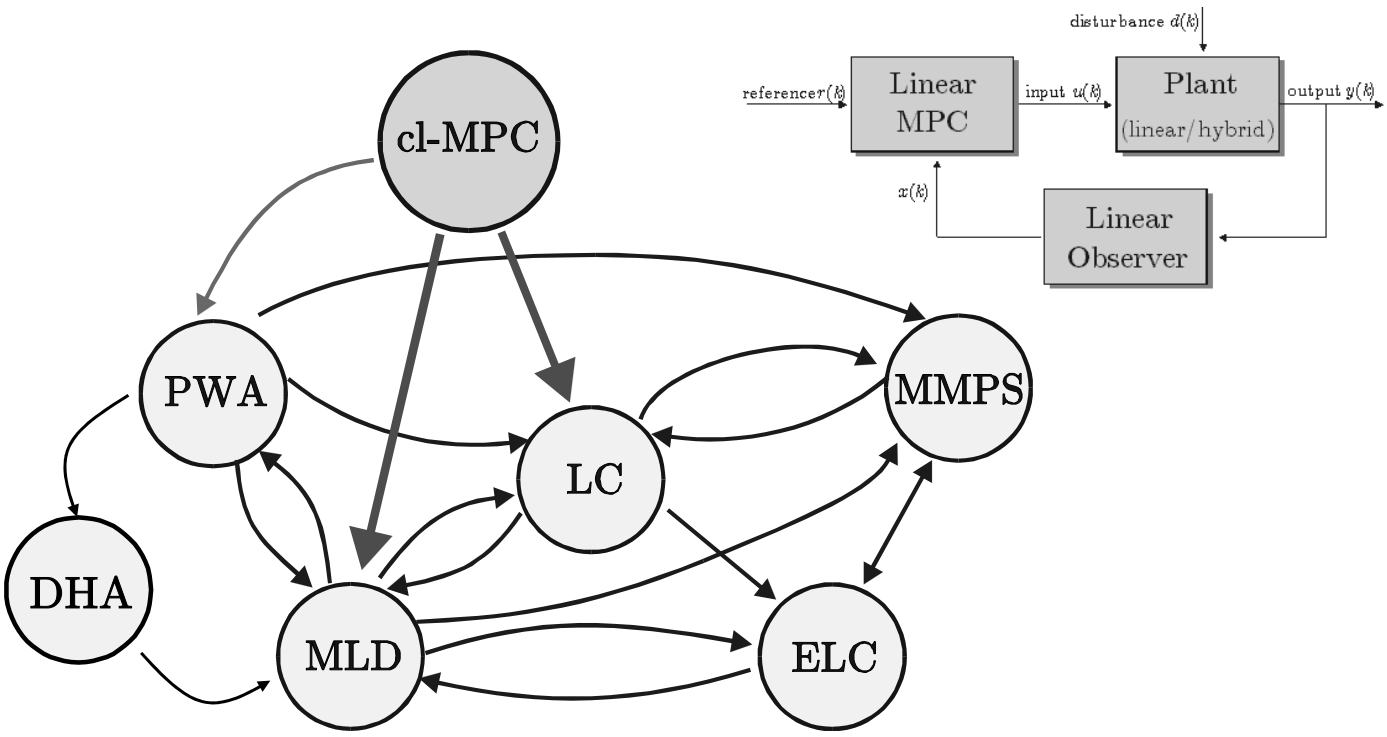
- Variable constraints: $u(t) = F(x(t), u_{\min}(t), \ldots, y_{\max}(t))$

  $u_{\min}(t) \leq u(t+k) \leq u_{\max}(t)$
  $y_{\min}(t) \leq y(t+k|t) \leq y_{\max}(t)$

- Linear norms: $\min_U J(U, x(t)) \triangleq \sum_{k=0}^{p} \|Qy(t+k|t)\|_\infty + \|Ru(t+k)\|_\infty$

  (Bemporad, Borrelli, Morari, IEEE TAC, 2002)

Closed-Loop MPC and Hybrid Systems



Motivation:

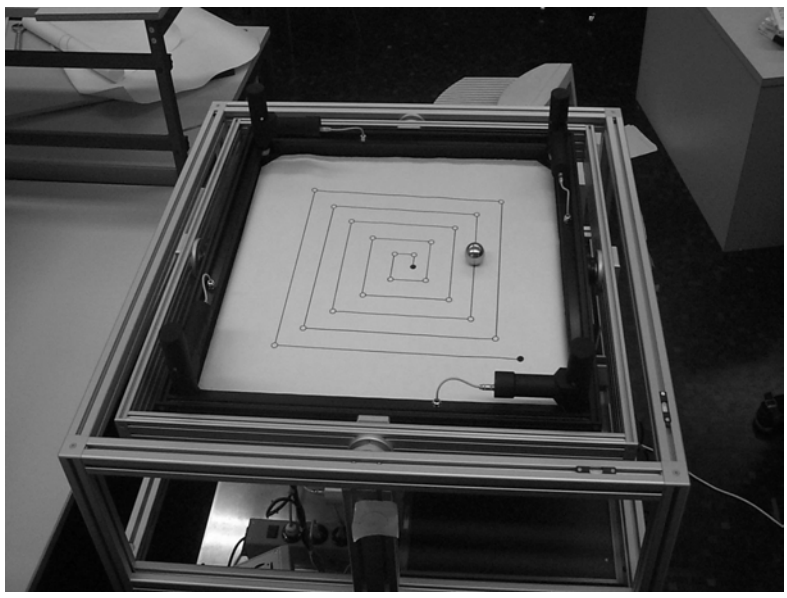
Use hybrid techniques to analyze closed-loop MPC systems !

(Bemporad, Heemels, De Schutter IEEE TAC, 2002)

MPC Regulation of a Ball on a Plate

Task:

- Tune an MPC controller by simulation, using the *MPC Simulink Toolbox*
- Get the *explicit solution* of the MPC controller.
- Validate the controller on *experiments*.



B&P Experimental Setup

Host

Ball & Plate System

TCP/IP Code Data

Ball & Plate

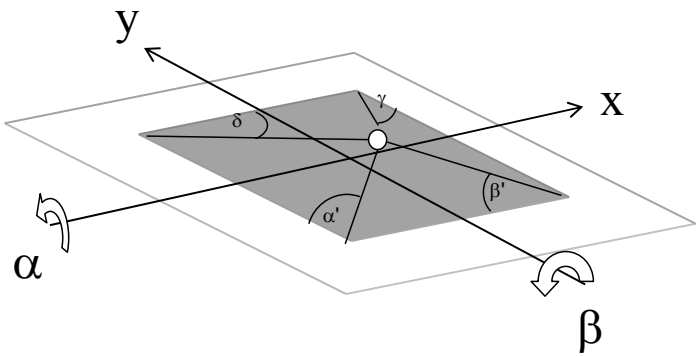
u_1, u_2

a,b

x, y

Target

Ball&Plate Experiment



- Specifications:

  Angle: -17 deg ... +17deg

  Plate: -30 cm ...+30 cm

  Input Voltage: -10 V... +10 V

  Computer: PENTIUM166

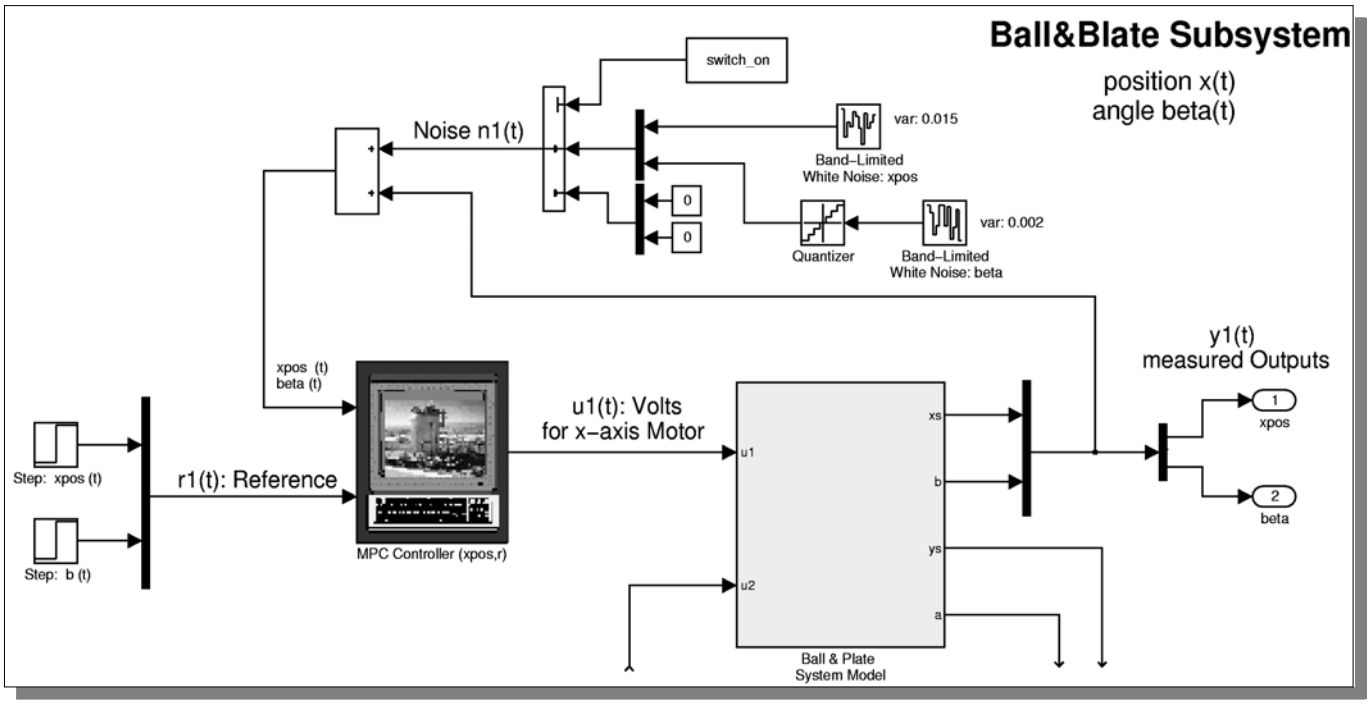
  Sampling Time:30 ms

- Model: LTI 14 states

  Constraints on inputs and states

General Philosophy: (1) MPC Design

- Step 1: Tune the MPC controller (in simulation)

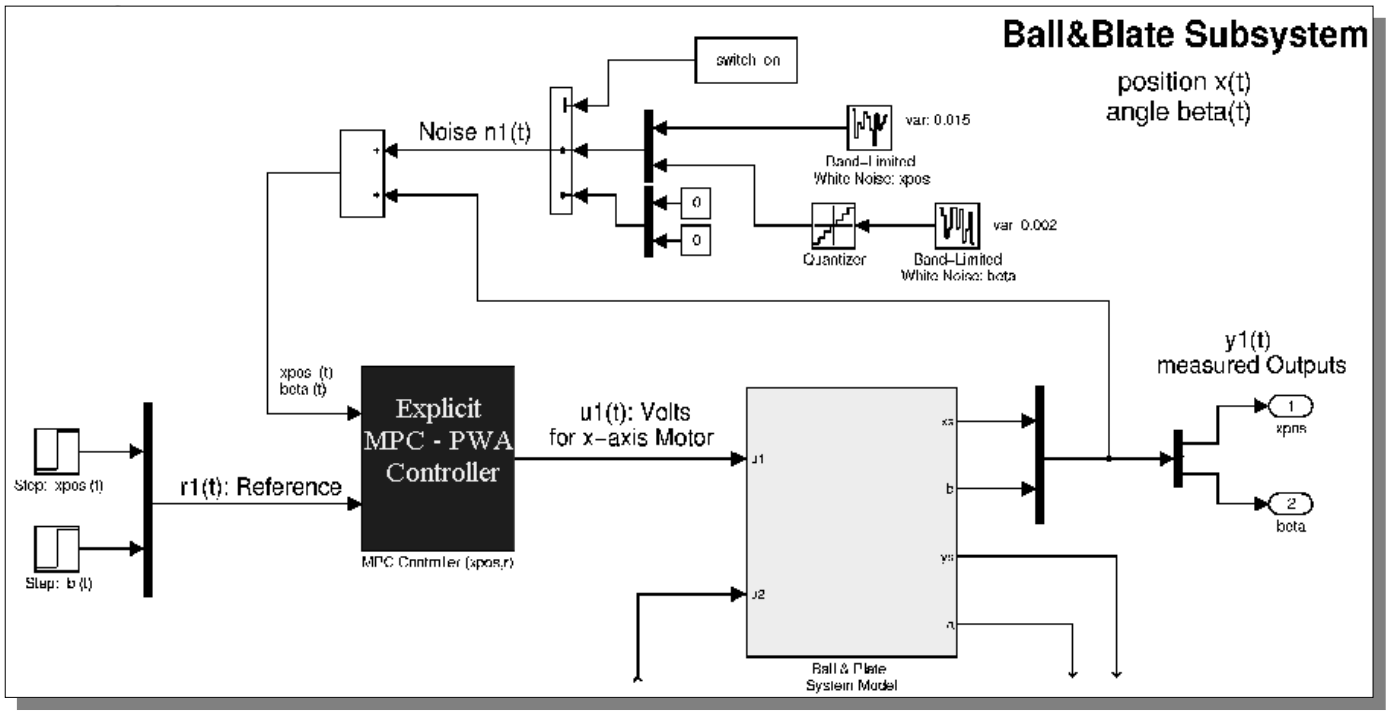


E.g: MPC Toolbox for Matlab

(Bemporad, Morari, Ricker, 2003)

General Philosophy: (2) Implementation

- Step 2: Solve mp-QP and implement Explicit

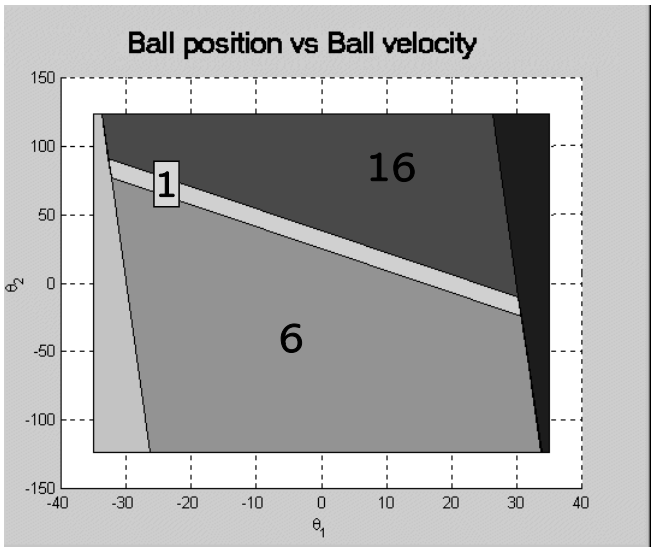


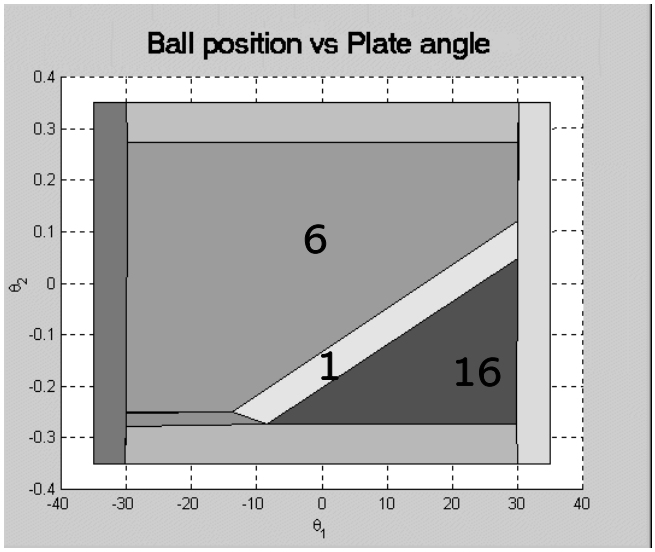
E.g: Real-Time Workshop + xPC Toolbox

Explicit MPC Solution

Controller: x: 22 Regions, y: 23 Regions

x-MPC: sections at $\alpha_x^{\circ}=0$, $\alpha_x=0$, $u_x=0$, $r_x=18$, $r_\alpha=0$





Region 1: LQR Controller (near Equilibrium)

Region 6: Saturation at -10

Region 16: Saturation at +10

MPC Regulation of a Ball on a Plate

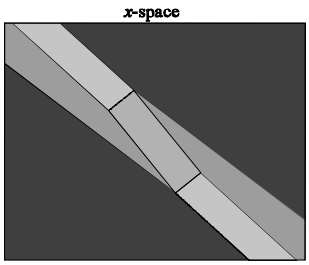
<u>Design Steps:</u>

- Tune an MPC controller by simulation, using the *MPC Simulink Toolbox* .
- Get the *explicit solution* of the MPC controller.
- ✓ Validate the controller on *experiments*.

Comments on Explicit MPC

- Multiparametric Quadratic Programs (mp-QP) can be solved efficiently
- Model Predictive Control (MPC) can be solved off-line via mp-QP
- Explicit solution of MPC controller $u = f(x)$ is Piecewise Affine

$$u(x) = \begin{cases} F_1 x + G_1 & \text{if} \quad H_1 x \leq K_1 \\ \vdots & \vdots \\ F_N x + G_N & \text{if} \quad H_N x \leq K_N \end{cases}$$



⇒ Eliminate heavy on-line computation for MPC

⇒ Make MPC suitable for fast/small/cheap processes

MILP Formulation of MPC

(Bemporad, Borrelli, Morari, 2000)

$$\min \sum_{k=0}^{T-1} \|Qy(t+k+1|t)\|_\infty + \|Ru(t+k)\|_\infty$$
$$\text{s.t.} \text{MLD dynamics}$$

- Introduce slack variables: $\min |x|$ ⇒

$$\begin{array}{ll} \min & \epsilon \\ \text{s.t.} & \epsilon \geq x \\ & \epsilon \geq -x \end{array}$$

$$\begin{array}{rcl} \epsilon_k^x & \geq & \|Qy(t+k|t)\|_\infty \\ \epsilon_k^u & \geq & \|Ru(t+k)\|_\infty \end{array} \Rightarrow$$

$$\begin{array}{rcll} \epsilon_k^x & \geq & [Qy(t+k|t)]_i & i=1,\ldots,p, \quad k=1,\ldots,T-1 \\ \epsilon_k^x & \geq & -[Qy(t+k|t)]_i & i=1,\ldots,p, \quad k=1,\ldots,T-1 \\ \epsilon_k^u & \geq & [Ru(t+k))]_i & i=1,\ldots,m, \quad k=0,\ldots,T-1 \\ \epsilon_k^u & \geq & -[Ru(t+k))]_i & i=1,\ldots,m, \quad k=0,\ldots,T-1 \end{array}$$

- Set $\xi \triangleq [\epsilon_1^x, \ldots, \epsilon_{N_y}^x, \epsilon_1^u, \ldots, \epsilon_{T-1}, U, \delta, z]$

Mixed Integer Linear Program (MILP)

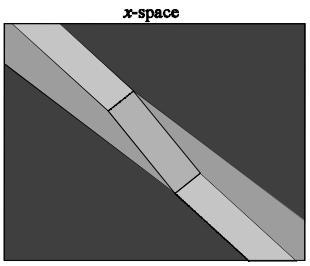
$$\min_\xi J(\xi, x(t)) = \sum_{k=0}^{T-1} \epsilon_i^x + \epsilon_i^u$$
$$\text{s.t.} G\xi \leq W + Sx(t)$$

Multiparametric MILP

$$\begin{aligned} \min_{\xi=\{\xi_c,\xi_d\}} \quad & f'\xi_c + d'\xi_d \\ \text{s.t.} \quad & G\xi_c + E\xi_d \leq W + Fx \end{aligned} \qquad \begin{aligned} \xi_c &\in \mathbf{R}^n \\ \xi_d &\in \{0,1\}^m \end{aligned}$$

- mp-MILP can be solved (by alternating MILPs and mp-LPs)
  (Dua, Pistikopoulos, 1999)
- **Theorem:** The multiparametric solution $\xi^*(x)$ is piecewise affine
- **Corollary: The MPC controller is piecewise affine in** x

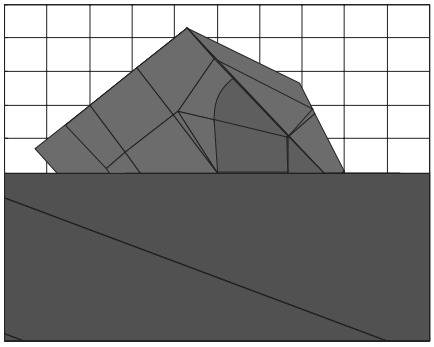
$$u(x) = \begin{cases} F_1 x + G_1 & \text{if} \quad H_1 x \leq K_1 \\ \vdots & \vdots \\ F_N x + G_N & \text{if} \quad H_N x \leq K_N \end{cases}$$



Solutions via Dynamic Programming

(Borrelli, Bemporad, Baotic, Morari, 2003)
(Mayne, ECC 2001)

- Explicit solutions to finite-time optimal control problems for PWA systems can be obtained using a combination of
  - Dynamic Programming
  - Multiparametric Linear (1-norm, ∞-norm), or or Quadratic (squared 2-norm) programming



Note: in the 2-norm case,
the partition may not be polyhedral

Hybrid Control Examples (Revisited)

Hybrid Control - Example

Switching System:

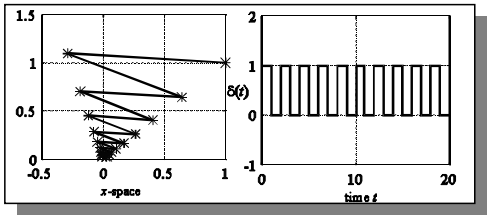
$$x(t+1) = 0.8 \begin{bmatrix} \cos\alpha(t) & -\sin\alpha(t) \\ \sin\alpha(t) & \cos\alpha(t) \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(t)$$

$$y(t) = [\,0 \quad 1\,]\, x(t)$$

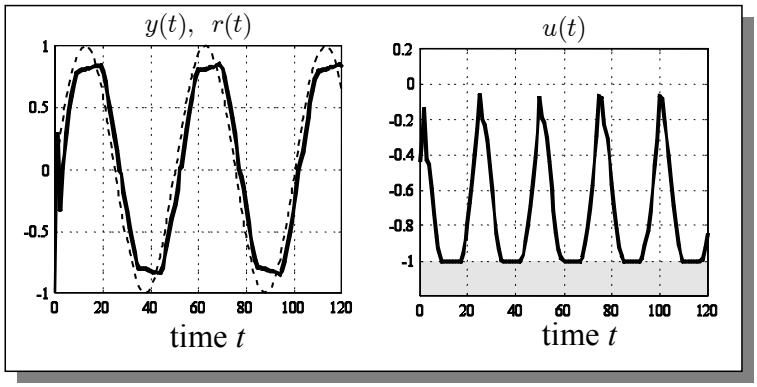
$$\alpha(t) = \begin{cases} \frac{\pi}{3} \text{ if } [1\ \ 0]x(t) \geq 0 \\ -\frac{\pi}{3} \text{ if } [1\ \ 0]x(t) < 0 \end{cases}$$

Constraint: $\boxed{-1 \leq u(t) \leq 1}$

Open loop:



Closed loop:



Hybrid MPC - Example

- MLD system

State x(t)	2 variables
Input u(t)	1 variables
Aux. binary vector δ(t)	1 variables
Aux. continuous vector z(t)	4 variables

- mp-MILP optimization problem

$$\min_{\left\{v_0^1\right\}} J(v_0^1, x(t)) \triangleq \sum_{k=0}^{1} \|Q_1(v(k)-u_e)\|_\infty + \|Q_2(\delta(k|t)-\delta_e)\|_\infty + \|Q_3(z(k|t)-z_e)\|_\infty + \|Q_4(x(k|t)-x_e)\|_\infty$$

subject to constraints

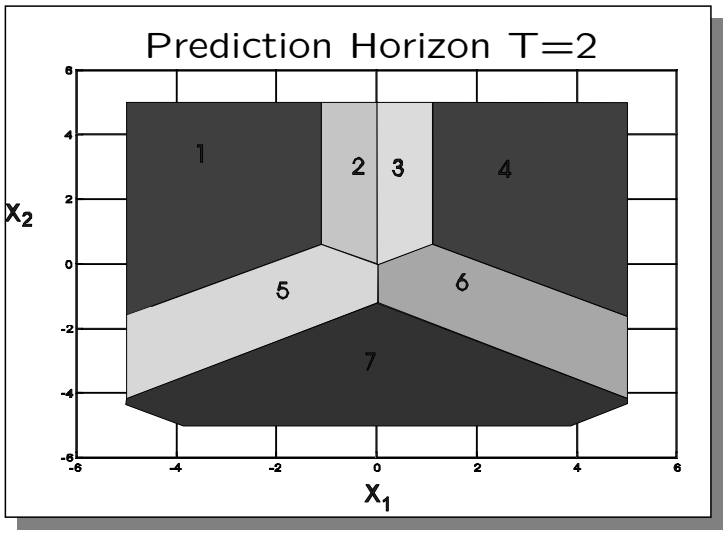
to be solved in the region $\begin{array}{rcl} -5 & \leq x_1 \leq & 5 \\ -5 & \leq x_2 \leq & 5 \end{array}$

- Computational complexity of mp-MILP

Linear constraints	84
Continuous variables	20
Binary variables	2
Parameters	2
Time to solve mp-MILP	3 min
Number of regions	7

mp-MILP Solution

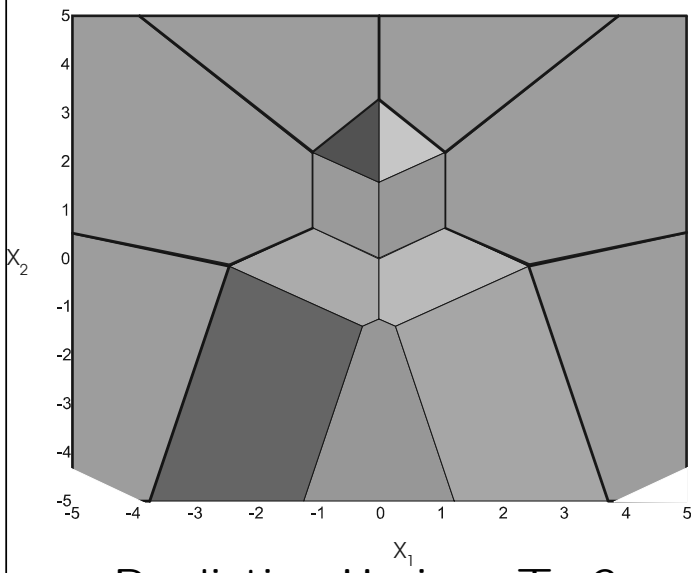
$$u = \begin{cases} -1.0000 & \text{if } \begin{bmatrix} 1.0000 & -1.7296 \\ -1.0000 & 0.0000 \\ 0.0000 & 1.0000 \\ 94.5067 & -1.0000 \end{bmatrix} x \leq \begin{bmatrix} -2.1680 \\ 5.0000 \\ 5.0000 \\ -105.1385 \end{bmatrix} \\ \text{(Region \#1)} \\ \begin{bmatrix} 0.9238 & 0.0000 \end{bmatrix} x & \text{if } \begin{bmatrix} -188.1504 & 1.0000 \\ -20.3158 & -34.1997 \\ 1.0000 & 0.0000 \\ 0.0000 & 1.0000 \end{bmatrix} x \leq \begin{bmatrix} 204.3621 \\ 1.0000 \\ 0.0000 \\ 5.0000 \end{bmatrix} \\ \text{(Region \#2)} \\ \begin{bmatrix} -0.9238 & -0.0000 \end{bmatrix} x & \text{if } \begin{bmatrix} -217.00 & 1.0000 \\ 38.1919 & -64.29 \\ 1.0000 & 0.0000 \\ 0.0000 & 1.0000 \end{bmatrix} x \leq \begin{bmatrix} 4.9708 \\ 1.0000 \\ 1.0000 \\ 5.0000 \end{bmatrix} \\ \text{(Region \#3)} \\ -1.0000 & \text{if } \begin{bmatrix} -188.15 & -1.0000 \\ -1.0000 & -1.6858 \\ 1.0000 & 0.0000 \\ 1.0000 & 133.68 \end{bmatrix} x \leq \begin{bmatrix} -204.39 \\ -2.1673 \\ 5.0000 \\ 665.60 \end{bmatrix} \\ \text{(Region \#4)} \\ \begin{bmatrix} 0.4619 & -0.8000 \end{bmatrix} x & \text{if } \begin{bmatrix} 1.0000 & -1.7193 \\ -1.0000 & 1.7295 \\ -1.0000 & 0.0000 \\ 13.4029 & 23.6383 \\ 43.4009 & 0.0000 \end{bmatrix} x \leq \begin{bmatrix} 2.1387 \\ 2.1174 \\ 5.0000 \\ -1.0000 \\ -1.0000 \end{bmatrix} \\ \text{(Region \#5)} \\ \begin{bmatrix} -0.4619 & -0.8000 \end{bmatrix} x & \text{if } \begin{bmatrix} 1.0000 & 1.7175 \\ -20.2527 & 34.1997 \\ -1.0000 & 0.0000 \\ -1.0000 & -1.7369 \\ 106.3247 & 1.0000 \end{bmatrix} x \leq \begin{bmatrix} 2.0413 \\ -1.0000 \\ 0.0000 \\ 2.2345 \\ 525.0259 \end{bmatrix} \\ \text{(Region \#6)} \\ 1.0000 & \text{if } \begin{bmatrix} 1.0000 & -1.6446 \\ -1.0000 & 1.7449 \\ -1.0000 & 0.0000 \\ -1.0000 & -1.6788 \\ 1.0000 & -265.137 \\ 1.0000 & 1.7369 \\ 1.0000 & 0.0000 \end{bmatrix} x \leq \begin{bmatrix} 12.0940 \\ -2.2448 \\ 5.0000 \\ 12.3141 \\ 1329.55 \\ -2.2345 \\ 5.0000 \end{bmatrix} \\ \text{(Region \#7)} \end{cases}$$



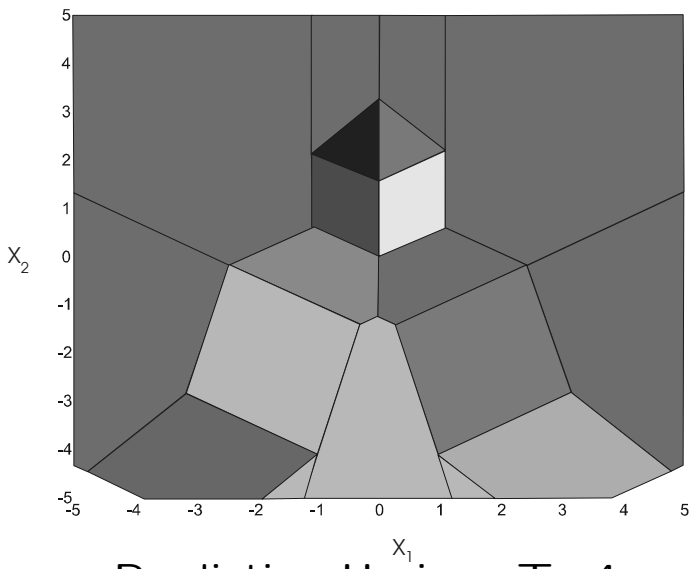
PWA law $\equiv$ MPC law

Linear constraints	84
Continuous variables	20
Binary variables	2
Parameters	2
Time to solve mp-MILP	3 min
Number of regions	7

mp-MILP Solution



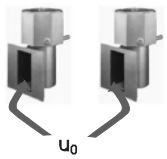
Prediction Horizon T=3



Prediction Horizon T=4

Alternate Heating of Two Furnaces

(Bemporad, Borrelli, Morari, 2000)



- mp-MILP optimization problem

$$\min_{\{u(t),u(t+1),u(t+2)\}} \sum_{k=0}^{2} \|R(u(t+k+1)-u(t+k))\|_\infty + \|Q(x(t+k|t)-x_e)\|_\infty$$

- Parameter set

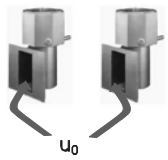
$$\begin{array}{rcl} -1 & \leq x_1 \leq & 1 \\ -1 & \leq x_2 \leq & 1 \\ 0 & \leq u_0 \leq & 1 \end{array}$$

parameterized !

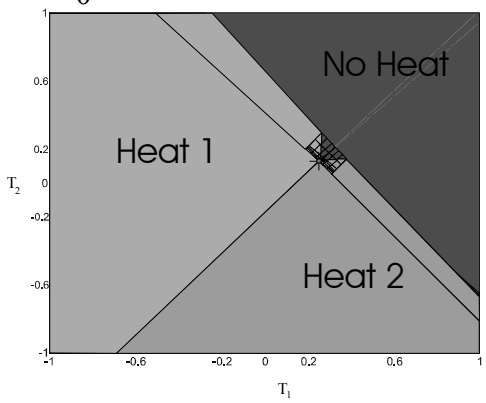
- Computational complexity of mp-MILP

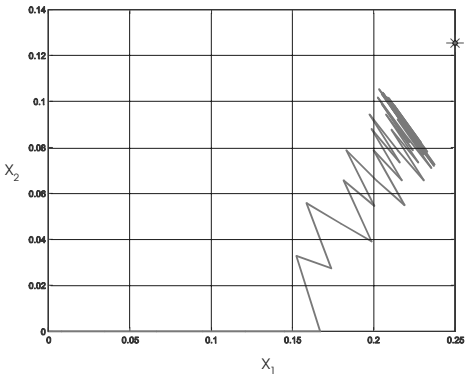
Linear constraints	168
Continuous variables	33
Binary variables	6
Parameters	3
Time to solve mp-MILP	5 min
Number of regions	105

mp-MILP Solution



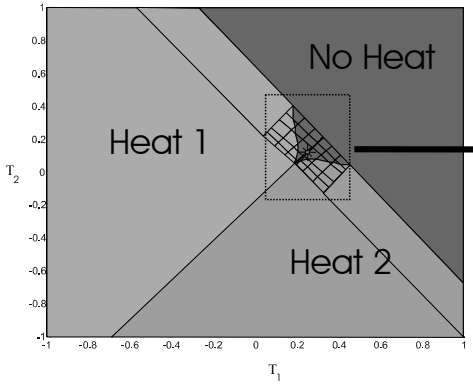
u_0=0.4

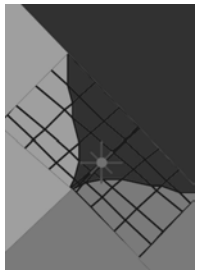


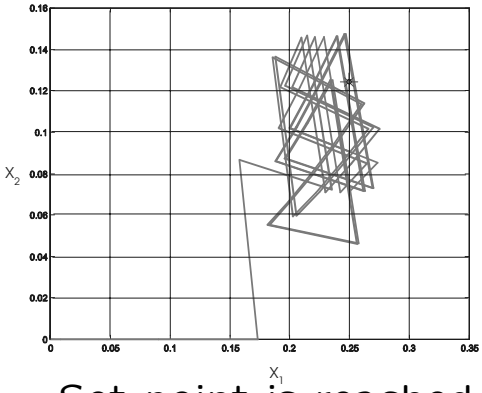


Set point cannot be reached

u_0=0.8







Set point is reached

NOTE: The control action of explicit and of implicit MPC are totally equal !

Hybrid Control Example: Traction Control System

Vehicle Traction Control

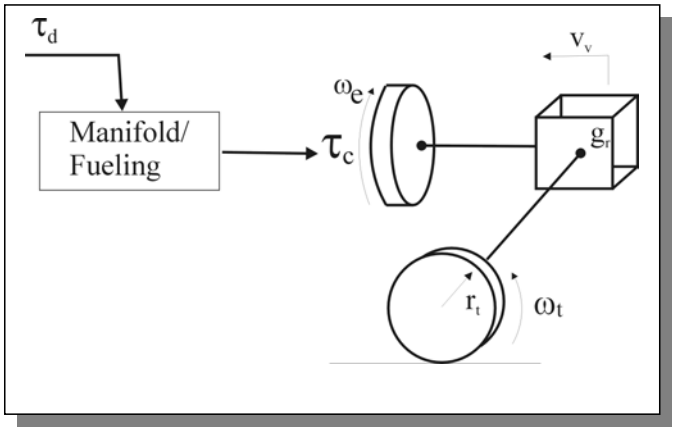
Improve driver's ability to control a vehicle under adverse external conditions (wet or icy roads)

Model
nonlinear, uncertain, constraints

Controller
suitable for real-time implementation

MLD hybrid framework + optimization-based control strategy

Simple Traction Model

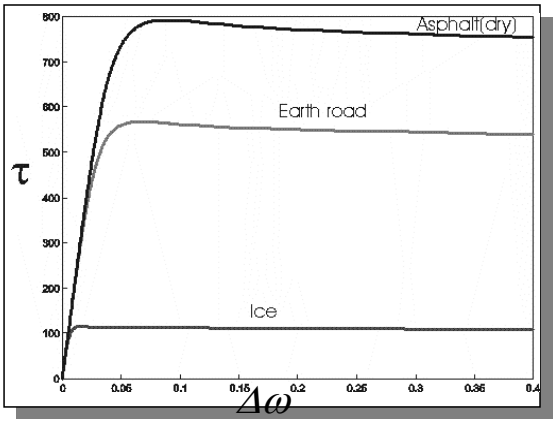


- Mechanical system

$$\dot{\omega}_e = \frac{1}{J_e}\left(\tau_c - b_e\omega_e - \frac{\tau_t}{g_r}\right)$$
$$\dot{v}_v = \frac{\tau_t}{m_v r_t}$$

- Manifold/fueling dynamics

$$\tau_c = b_i\tau_d(t - \tau_f)$$

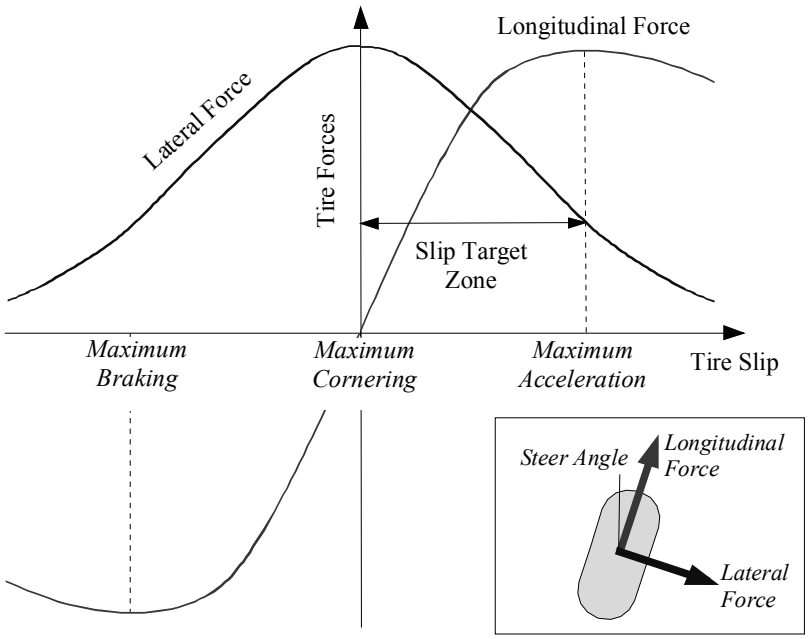


- Tire torque τ_t is a function of slip $\Delta\omega$ and road surface adhesion coefficient μ

$$\Delta\omega = \frac{\omega_e}{g_r} - \frac{v_v}{r_t}$$

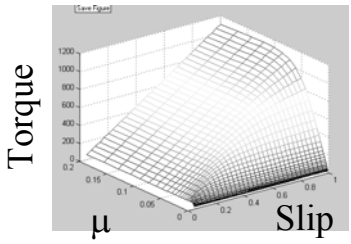
Wheel slip

Tire Force Characteristics



Ford Motor Company

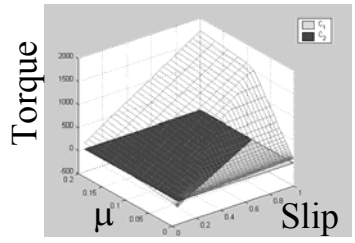
Hybrid Model

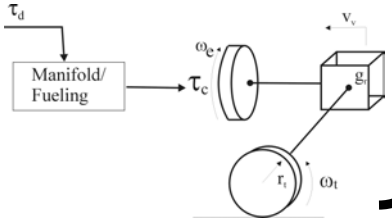


Nonlinear tire torque $\tau_t = f(\Delta\omega, \mu)$

PWA Approximation

(PWL Toolbox, Julian, 1999)





HYSDEL
(Hybrid Systems Description Language)

Mixed-Logical Dynamical (MLD) Hybrid Model (discrete time)

MLD Model

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

State x(t)	9 variables
Input u(t)	1 variable
Aux. Binary vars δ(t)	3 variables
Aux. Continuous vars z(t)	4 variables

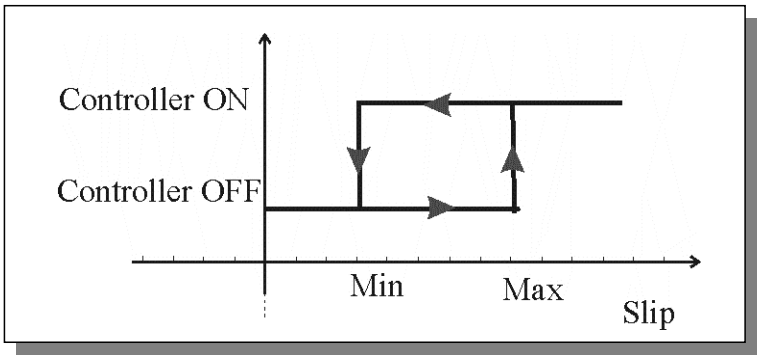
➡ The MLD matrices are automatically generated in Matlab format by HYSDEL

Performance and Constraints

- Control objective:

$$\begin{aligned} &\min \sum_{k=0}^{N} \quad |\Delta\omega(k|t) - \Delta\omega_{des}| \\ &\text{subj. to.} \quad \text{MLD Dynamics} \end{aligned}$$

- Constraints:
  - Limits on the engine torque: $-20Nm \leq \tau_d \leq 176Nm$
- Logic Constraint:
  - Hysteresis

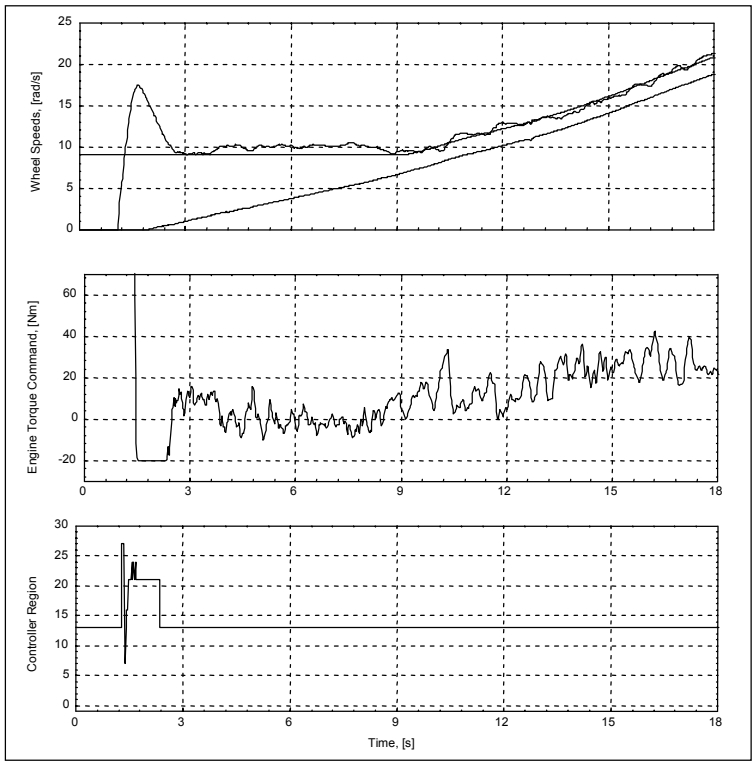


Experimental Apparatus

Experimental Apparatus

Ford Motor Company

Experimental Results



Ford Motor Company

Experiment

- >500 regions
- 20ms sampling time
- Pentium 266Mhz + Labview

Ford Motor Company

Hybrid Control Example: Cruise Control System

Hybrid Control Problem

Renault Clio 1.9 DTI RXE

GOAL:

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

Hybrid Model

- Vehicle dynamics

$$m\ddot{x} = F_e - F_b - \beta\dot{x}$$

$\dot{x}$ = vehicle speed

F_e = traction force

F_b = brake force

$\Longrightarrow$ discretized with sampling time $T_s = 0.5$ s

- Transmission kinematics

$$\omega = \frac{R_g(i)}{k_s}\dot{x}$$

$$F_e = \frac{R_g(i)}{k_s}M$$

ω = engine speed

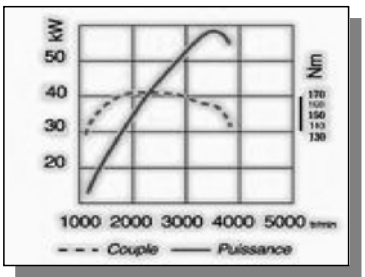
M = engine torque

i = gear

Hybrid Model

- Engine torque $-C_e^-(\omega) \leq M \leq C_e^+(\omega)$

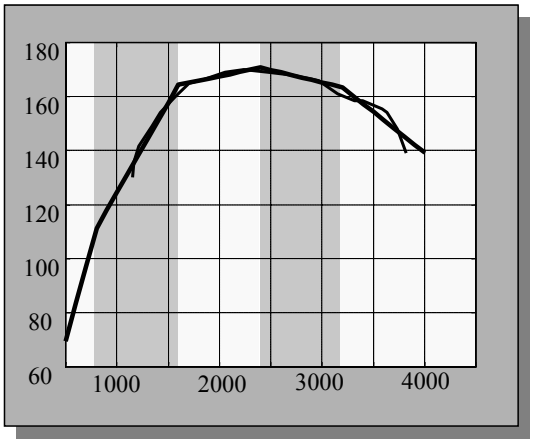
- Max engine torque $C_e^+(\omega)$



http://www.renault.fr

Piecewise-linearization:
(PWL Toolbox, Julián, 1999)

$\Longrightarrow$



requires: 4 binary aux variables
4 continuous aux variables

- Min engine torque $C_e^-(\omega) = \alpha_1 + \beta_1\omega$

Hybrid Model

- Gear selection: for each gear #i,

  define a binary input $\boxed{g_i \in \{0,1\}}$

- Gear selection (traction force):

  $\boxed{F_e = \frac{R_g(i)}{k_s} M}$ depends on gear #i

  define auxiliary continuous variables:

  $$\boxed{\text{IF } \; g_i = 1 \; \text{ THEN } \; F_{ei} = \frac{R_g(i)}{k_s} M \; \text{ ELSE } \; 0}$$

  $$\Longrightarrow \boxed{F_e = F_{eR} + F_{e1} + F_{e2} + F_{e3} + F_{e4} + F_{e5}}$$

- Gear selection (engine/vehicle speed):

  $\boxed{\omega = \frac{R_g(i)}{k_s} \dot{x}}$ similarly, also requires 6 auxiliary continuous variables

Hybrid Model

- MLD model

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

- 2 continuous states: x, v (vehicle position and speed)
- 2 continuous inputs: M, F_b (engine torque, brake force)
- 6 binary inputs: $g_R, g_1, g_2, g_3, g_4, g_5$ (gears)
- 1 continuous output: v (vehicle speed)
- 16 auxiliary continuous vars: (6 traction force, 6 engine speed, 4 PWL max engine torque)
- 4 auxiliary binary vars: (PWL max engine torque breakpoints)
- 96 mixed-integer inequalities

Hysdel Model

```
SYSTEM car {

INTERFACE {
        STATE { REAL position, speed; }
        INPUT { REAL torque, F_brake;
                BOOL gear1, gear2, gear3, gear4, gear5, gearR; }

        PARAMETER{
                REAL mass = 1020; /* kg */
                REAL beta_friction = 25; /* N/m*s */
                REAL Rgear1 = 3.7271; REAL Rgear2 = 2.048;
                REAL Rgear3 = 1.321; REAL Rgear4 = 0.971;
                REAL Rgear5 = 0.756; REAL RgearR = -3.545;
                REAL wheel_rim = 14;        /* in */
                ...
                } }
IMPLEMENTATION {
        AUX {REAL Fe1, Fe2, Fe3, Fe4, Fe5, FeR;
             REAL w1, w2, w3, w4, w5, wR;
             BOOL dPWL1,dPWL2,dPWL3,dPWL4;
             REAL DCe1,DCe2,DCe3,DCe4; }

        AD { dPWL1 = wPWL1-(w1+w2+w3+w4+w5+wR)<=0;
             dPWL2 = wPWL2-(w1+w2+w3+w4+w5+wR)<=0;
             dPWL3 = wPWL3-(w1+w2+w3+w4+w5+wR)<=0;
             dPWL4 = wPWL4-(w1+w2+w3+w4+w5+wR)<=0; }

        DA { Fe1 = {IF gear1  THEN torque/speed_factor*Rgear1;
             Fe2 = {IF gear2  THEN torque/speed_factor*Rgear2;
             Fe3 = {IF gear3  THEN torque/speed_factor*Rgear3;
             Fe4 = {IF gear4  THEN torque/speed_factor*Rgear4;
             Fe5 = {IF gear5  THEN torque/speed_factor*Rgear5;
             FeR = {IF gearR  THEN torque/speed_factor*RgearR;

             w1 = {IF gear1  THEN speed/speed_factor*Rgear1;
             w2 = {IF gear2  THEN speed/speed_factor*Rgear2;
             w3 = {IF gear3  THEN speed/speed_factor*Rgear3;
             w4 = {IF gear4  THEN speed/speed_factor*Rgear4;
             w5 = {IF gear5  THEN speed/speed_factor*Rgear5;
             wR = {IF gearR  THEN speed/speed_factor*RgearR;

             DCe1 = {IF dPWL1 THEN (aPWL2-aPWL1)+(bPWL2-bPWL1)*(w1+w2+w3+w4+w5+wR);
             DCe2 = {IF dPWL2 THEN (aPWL3-aPWL2)+(bPWL3-bPWL2)*(w1+w2+w3+w4+w5+wR);
             DCe3 = {IF dPWL3 THEN (aPWL4-aPWL3)+(bPWL4-bPWL3)*(w1+w2+w3+w4+w5+wR);
             DCe4 = {IF dPWL4 THEN (aPWL5-aPWL4)+(bPWL5-bPWL4)*(w1+w2+w3+w4+w5+wR);
             }

        CONTINUOUS { position = position+Ts*speed;
                     speed = speed+Ts/mass*(Fe1+Fe2+Fe3+Fe4+Fe5+FeR-
                                            F_brake-beta_friction*speed); }

        MUST    {  wemin <= w1+w2+w3+w4+w5+wR;
                   w1+w2+w3+w4+w5+wR <= wemax;
                   -F_brake <=0; /* brakes cannot accelerate ! */
                   F_brake <= max_brake_force;

                   -torque-(alpha1+beta1*(w1+w2+w3+w4+w5+wR)) <=0;
                   torque-(aPWL1+bPWL1*(w1+w2+w3+w4+w5+wR)+DCe1+DCe2+DCe3+DCe4)-1<=0;

                   -(gear1+gear2+gear3+gear4+gear5+gearR)<=-1;
                   (gear1+gear2+gear3+gear4+gear5+gearR)<=1;
                   Fe1+Fe2+Fe3+Fe4+Fe5+FeR <= max_force;
                   -Fe1-Fe2-Fe3-Fe4-Fe5-FeR <= -max_force;

                   dPWL4 -> dPWL3; dPWL4 -> dPWL2;
                   dPWL4 -> dPWL1; dPWL3 -> dPWL2;
                   dPWL3 -> dPWL1; dPWL2 -> dPWL1;  }
```

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

Hybrid Controller

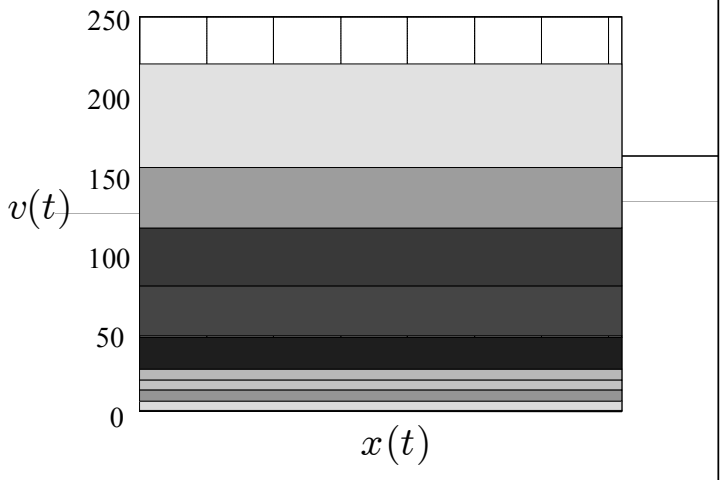
- Max-speed controller

$$\max_{u_t} J(u_t, x(t)) \triangleq v(t+1|t)$$

$$\text{subj. to} \begin{cases} \text{MLD model} \\ x(t|t) = x(t) \end{cases}$$

MILP optimization problem

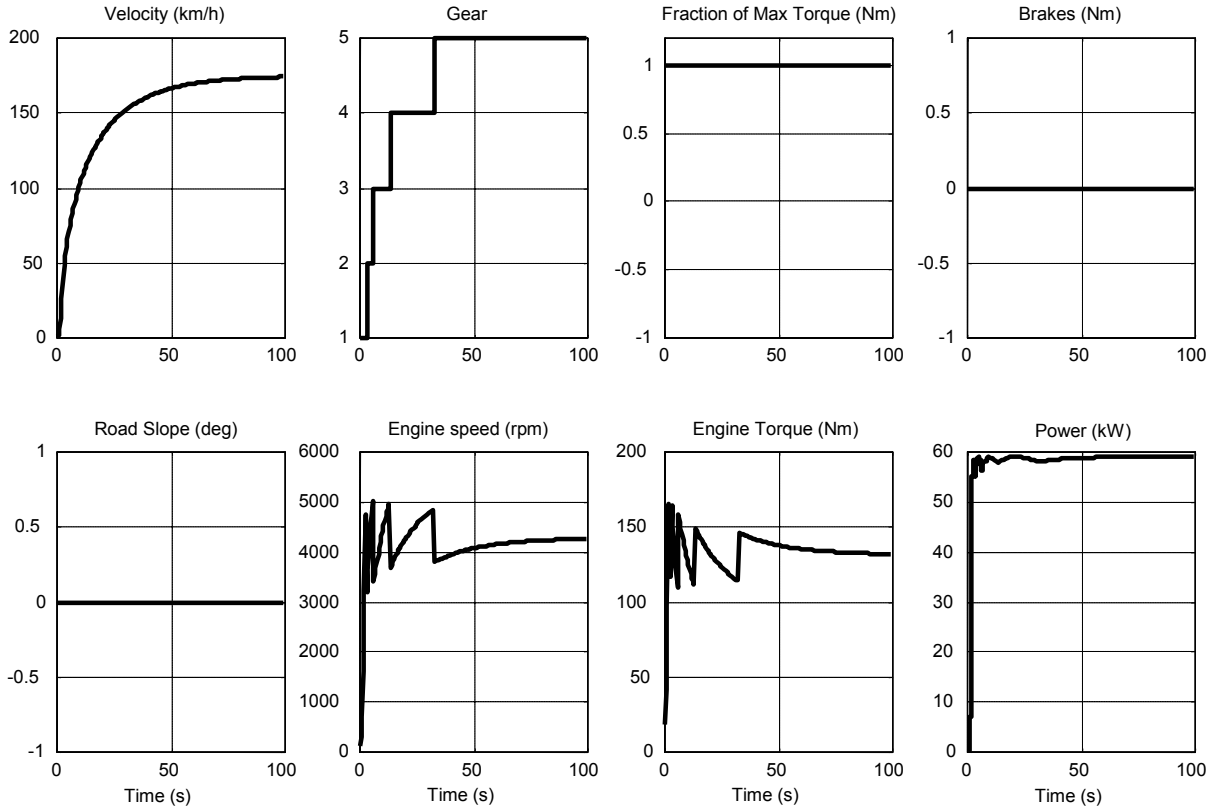
Linear constraints	96
Continuous variables	18
Binary variables	10
Parameters	1
Time to solve mp-MILP (Sun Ultra 10)	45 s
Number of regions	**11**



($x(t)$ is irrelevant)

Hybrid Controller

- Max-speed controller



Hybrid Controller

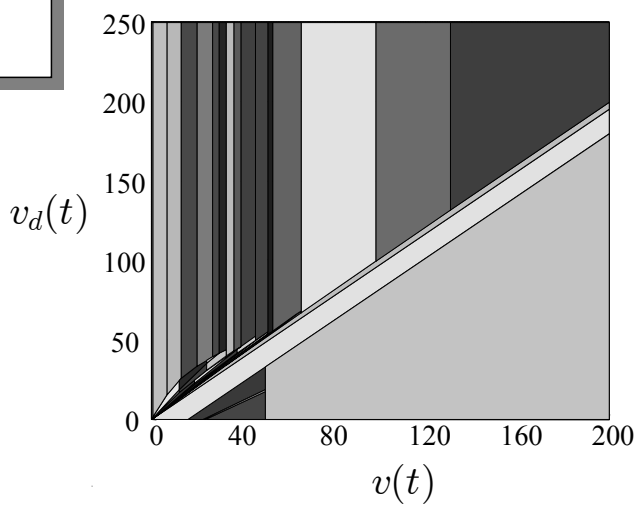
- Tracking controller

$$\min_{u_t} J(u_t, x(t)) \triangleq |v(t+1|t) - v_d(t)| + \rho|\omega|$$

$$\text{subj. to} \begin{cases} \text{MLD model} \\ x(t|t) = x(t) \end{cases}$$

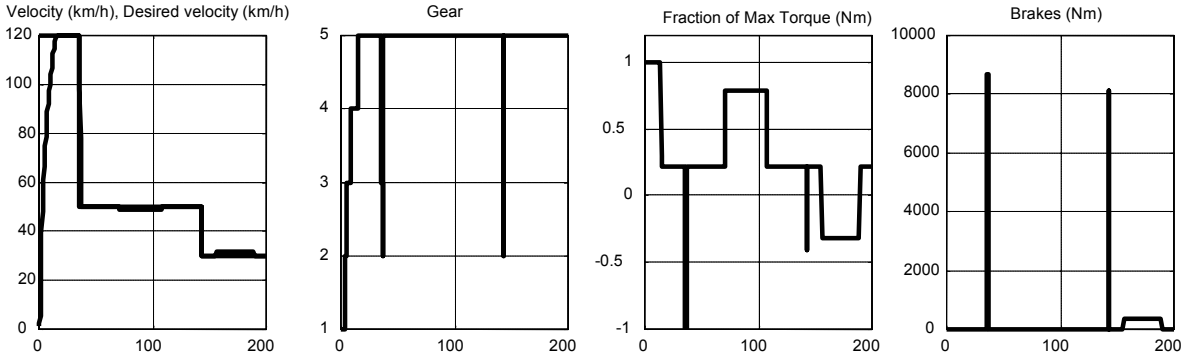
MILP optimization problem

Linear constraints	98
Continuous variables	19
Binary variables	10
Parameters	2
Time to solve mp-MILP (Sun Ultra 10)	27 m
Number of regions	**49**

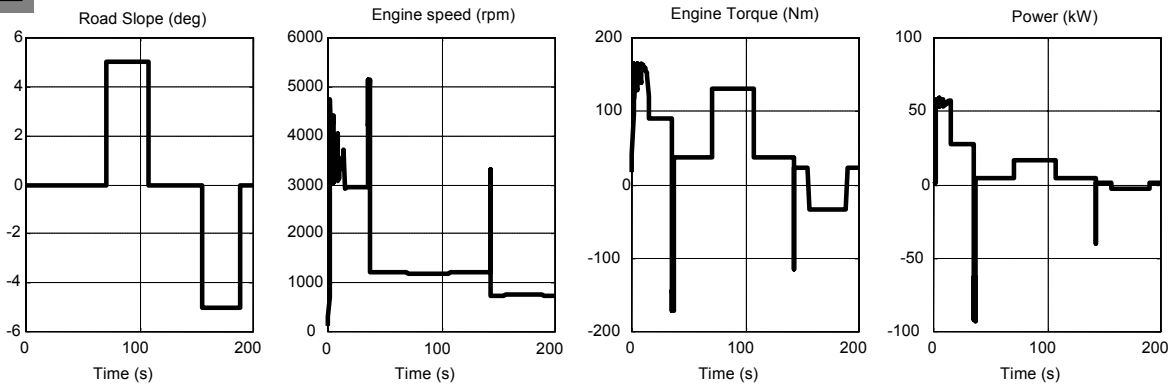


Hybrid Controller

- Tracking controller $\min_{u_t} J(u_t, x(t)) \triangleq |v(t+1|t) - v_d(t)| + \rho|\omega|$



$\rho = 0.001$



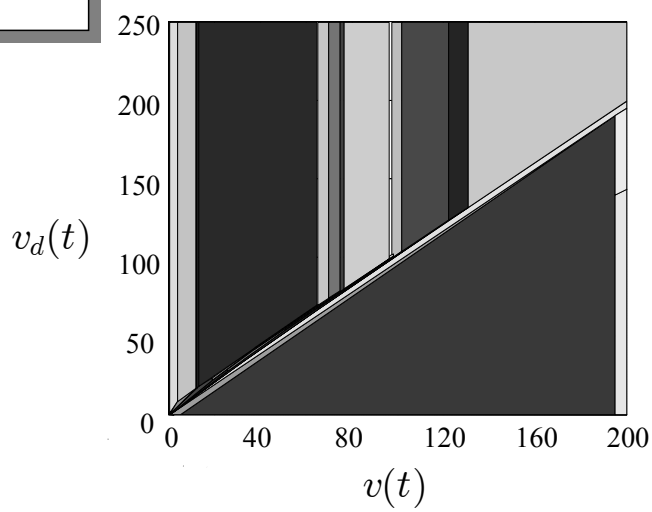
Hybrid Controller

- Smoother tracking controller

$$\min_{u_t} J(u_t, x(t)) \triangleq |v(t+1|t) - v_d(t)| + \rho|\omega|$$
$$\text{subj. to} \begin{cases} |v(t+1|t) - v(t)| < T_s a_{\max} \\ \text{MLD model} \\ x(t|t) = x(t) \end{cases}$$

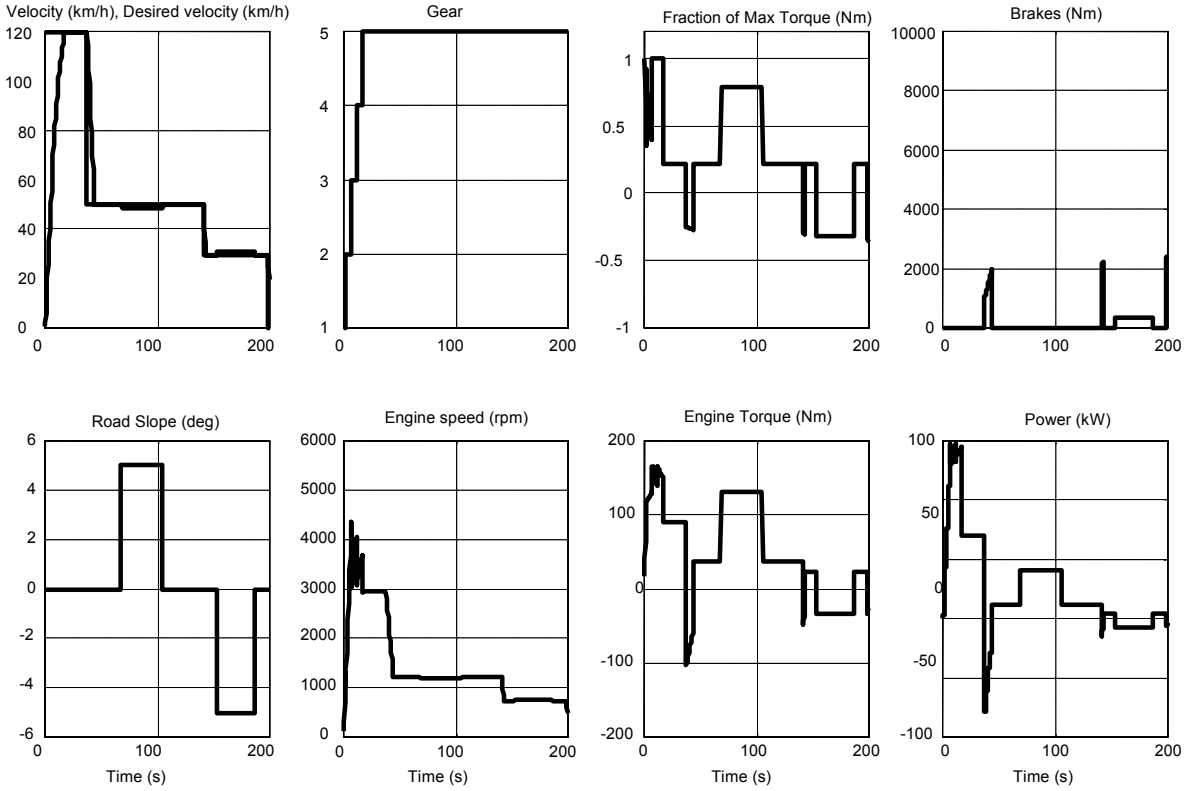
MILP optimization problem

Linear constraints	100
Continuous variables	19
Binary variables	10
Parameters	2
Time to solve mp-MILP (Sun Ultra 10)	28 m
Number of regions	**54**



Hybrid Controller

- Smoother tracking controller



Reachability Analysis (Verification)

Verification

- **GIVEN**: an embedded system (continuous dynamical system + logic controller)

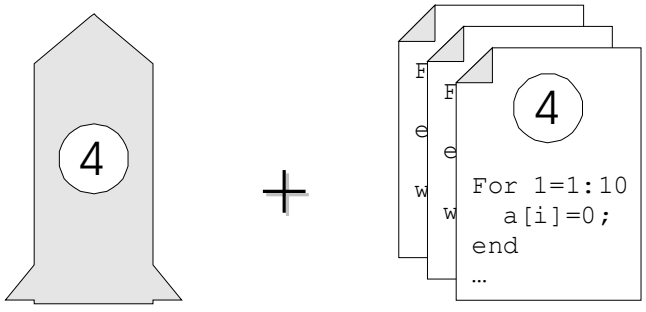
- **CERTIFY** that such combination behaves as desired
  - for ALL initial conditions within a given set
  - for ALL disturbances within a given class

- or **PROVIDE** a counterexample.

Simulation: provides a partial answer (not all possibilities can be tested!)

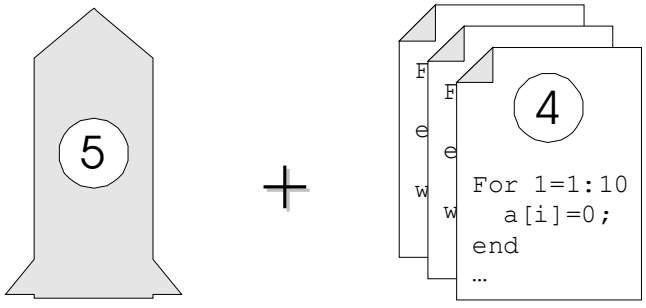
Reachability Analysis: provides the answer.

Ariane 5 - 1996



•Success

Physical beaviour Computer code

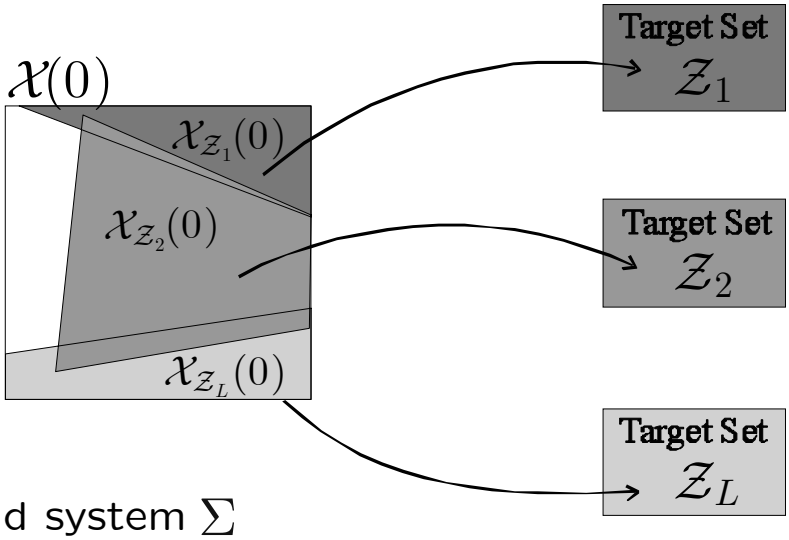


•Failure

Need for verification

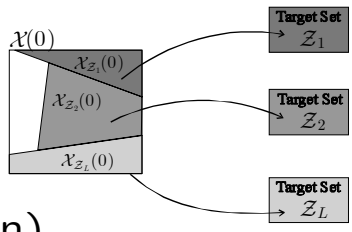
Reachability Analysis/Verification

(Bemporad, Torrisi, Morari, 2000)

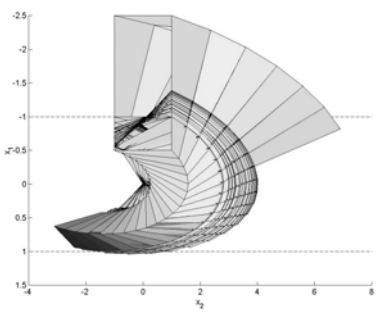


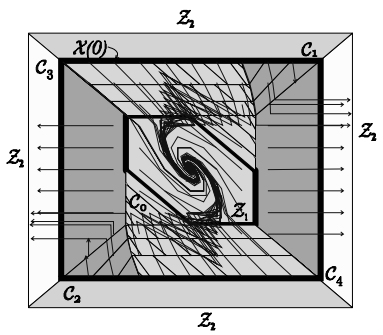
- Given:
  - A hybrid system Σ
  - A set of initial conditions $\mathcal{X}(0)$
  - Target sets $\mathcal{Z}_1,\ \mathcal{Z}_2,\ \ldots,\ \mathcal{Z}_L$ (disjoint)
  - Time horizon $t \leq T_{\max}$
- Problem:
  - Is $\mathcal{Z}_i$ reachable from $\mathcal{X}(0)$ in t steps ?
  - If yes, from which subset $\mathcal{X}_{\mathcal{Z}_i}(0)$ of $\mathcal{X}(0)$?
  - Disturbance/input sequences driving $\mathcal{X}_{\mathcal{Z}_i}(0)$ to $\mathcal{Z}_i$.

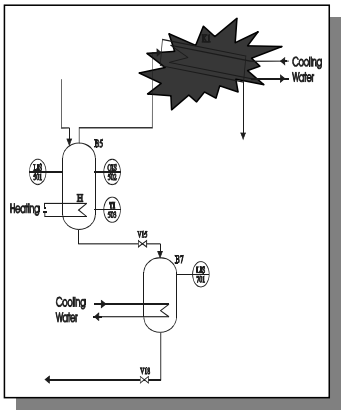
Applications



- Safety ($\mathcal{Z}_1,\ \mathcal{Z}_2,\ \ldots,\ \mathcal{Z}_L$ = unsafe sets)
- Stability ($\mathcal{Z}_1$ = invariant set around the origin)
- Optimal control ($u(0), \ldots, u(T_{\max})$ = optimal strategy, $\mathcal{Z}_1$ = reference set) (Bemporad, Giovanardi, Torrisi, CDC 2000)
- (practical) Liveness ($\mathcal{Z}_1$ = set to be reached within a finite time)







Verification Algorithm via MILP

- Simple solution: Solve $\forall T>0$ the mixed-integer linear feasibility test

$$\begin{cases} x(0) \in \mathcal{X}(0) \\ x(T) \in \mathcal{Z}_i \\ u(t) \in \mathcal{U}, 0 \le t \le T \\ x(t+1) = Ax(t) + B_1 u(t) + B_2 \delta(t) + B_3 z(t) \\ E_2 \delta(t) + E_3 z(t) \le E_1 u(t) + E_4 x(t) + E_5 \end{cases}$$

with respect to $x(0), \{u(t), \delta(t), z(t)\}_{t=0}^{T}$

- Only practical for small problems ! because number of free integer variables $\delta(0), \delta(1), \ldots, \delta(T)$ grows with T
- **Efficient Solution**: Exploit the special structure of the problem.

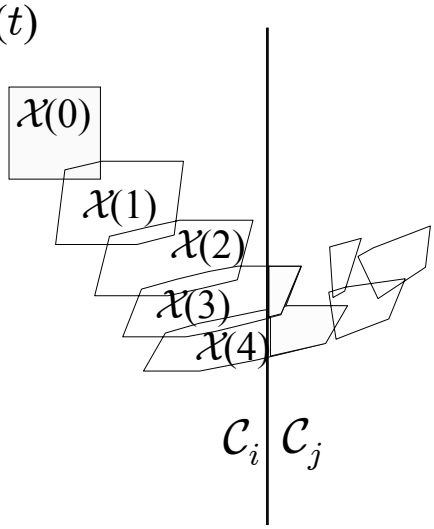
(Bemporad, Torrisi, Morari, 2000)
(Torrisi, Bemporad, Giovanardi, 2003)

Reach-Set Evolution

$T_{\max} < \infty$, discrete-time $\Rightarrow$ Decidable but $\mathcal{NP}$-hard

Reachability analysis algorithm:

- Compute the polyhedral reach set $\mathcal{X}(t)$
- Detect switching
- Describe new intersections $\mathcal{X}(t) \cap \mathcal{C}_j$
- Stopping criteria for a single exploration
- Organize the search



(Bemporad, Torrisi, Morari, 2000)

Reach Set Computation

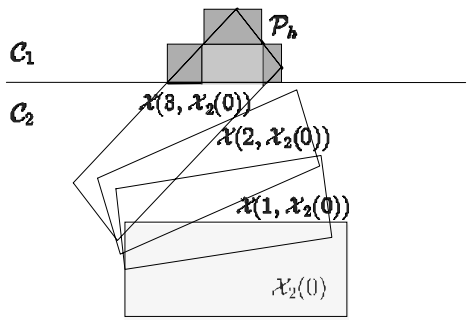
Reach set implicitly defined by linear inequalities

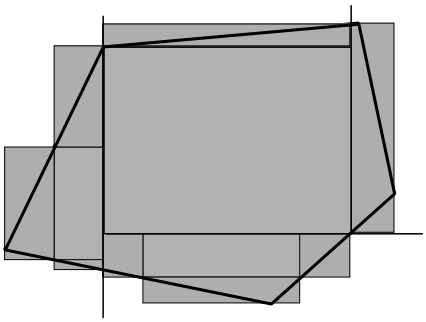
$$\begin{cases} x \in \mathcal{X}(0) \\ K_i^x \left(A_i^k x + \sum_{k=0}^{t-1} A_i^j [B_i u(t-1-k) + f_i] \right) + \\ \quad + K_i^u u(t) \leq H, \;\; k = 1, \ldots, t \\ u_{\min} \leq u(k) \leq u_{\max}, \;\; k = 0, \ldots, t-1 \end{cases}$$

where $\mathcal{X}_i = \{(x,u) : \;\; K_i^x x + K_i^u u \leq H\}$ is the current region

- Simple to compute
- Number of constraints grows linearly with time
- Explicit form also possible via projection methods (e.g. CDD, Fukuda 1997)

Approximation of Intersections

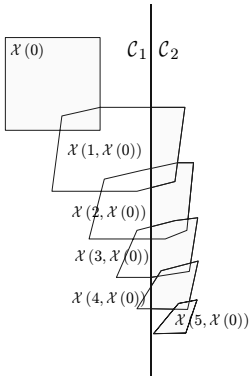




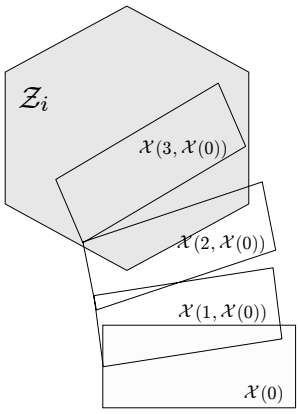
- ✓ Simple to compute via Linear Programming (LP)
- ✓ Can approximate with arbitrary precision
- ✓ Trade off between conservativeness and complexity
- ✓ Both inner and outer approximations in one shot
- ✓ Approximate computation of projections

Stopping Criteria

- Reach set has left the current region

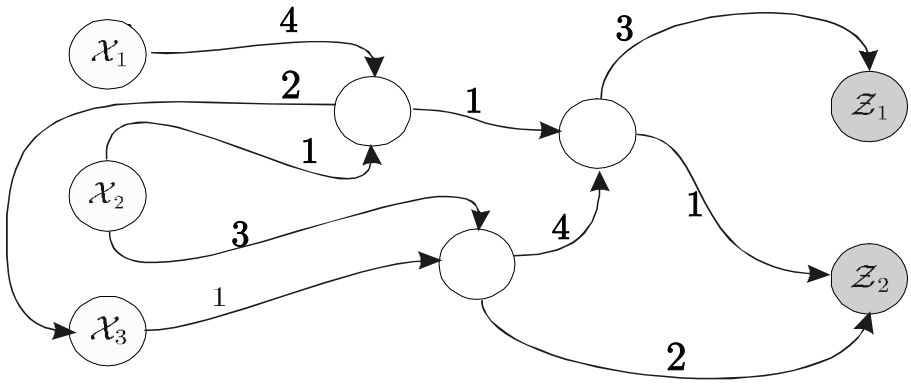


- Reach set is all inside a target $\mathcal{Z}_i$



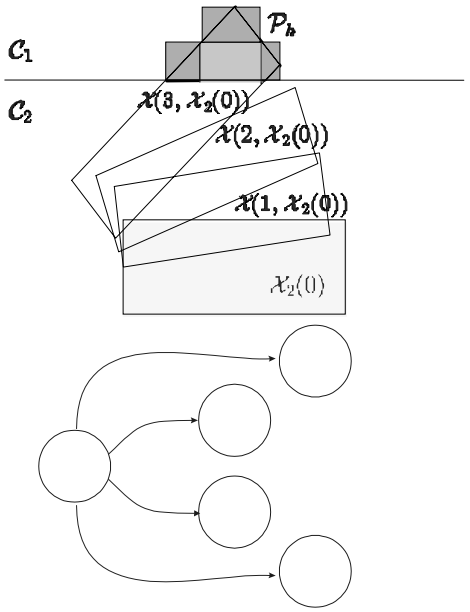
- $t > T_{\max}$ (to guarantee termination)

Graph of Evolution

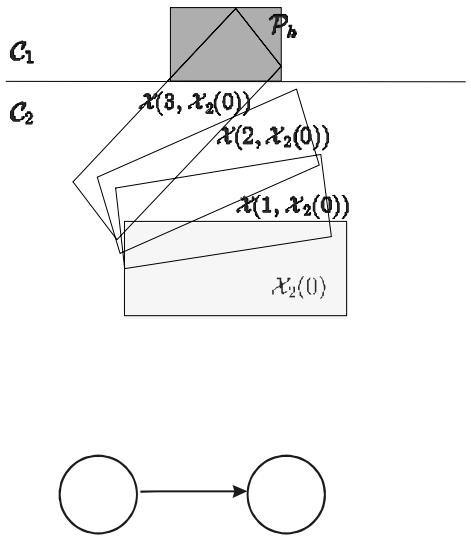


- The graph is initialized with all the initial subsets $\mathcal{X}_i=\mathcal{X}(0)\cap\mathcal{C}_i$ and all the target sets
- A node is added for each initial region of a new exploration
- If a set can reach another set, an oriented arc is drawn
- The time needed to reach the set is a weight associated with the arc

Conservativeness vs Graph Complexity



- Less conservative
- Graph more complicate

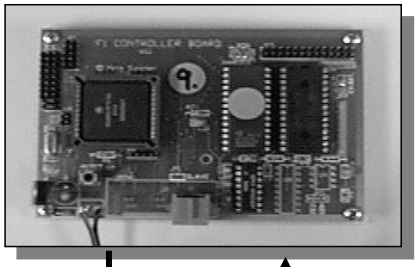


- Graph less complicate
- More conservative

LP determines if a switching sequence is feasible

Verification Example: Cruise Control System

Cruise Control System



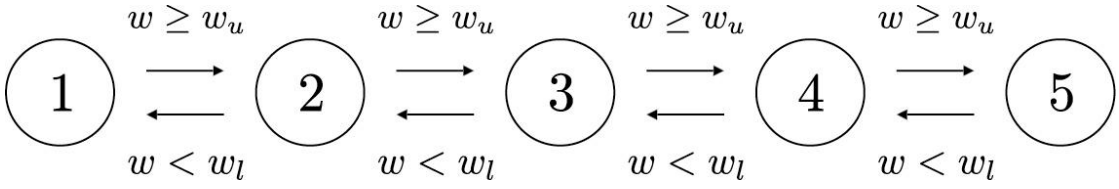
GOAL:

Verify if a given switching controller satisfies certain specifications

(Torrisi, Bemporad, 2001)

Cruise Control System

Gear selector:



Speed controller:

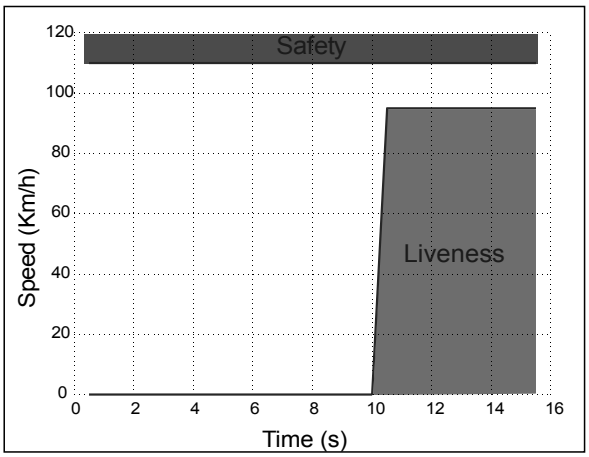
$$e(t+1) = e(t) + T_s(v_r(t) - v(t)) \quad (+ \text{ saturation})$$

$$u_e(t) = \begin{cases} k_e(v_e(t) - v(t)) + i_e e(t) & \text{if } v(t) < v_r + 1 \\ 0 & \text{otherwise} \end{cases}$$

$$u_b(t) = \begin{cases} k_b(v_e(t) - v(t)) & \text{if } v(t) \geq v_r + 1 \\ 0 & \text{otherwise} \end{cases}$$

Verification Problem

Question: Will the cruise control reach the desired speed reference within 10 s without exceeding the speed limit?



Safety $\mathcal{Z}_1 = \{v : v > v_r + r_{\text{toll}}\}$

Liveness $\mathcal{Z}_2 = \{v, t : v < v_r - 2r_{\text{toll}}, t > 10/T_s\}$

$r_{\text{toll}} = 5$ km/h

Hysdel Model (HYbrid Systems DEscription Language)

```
SYSTEM car {
INTERFACE {
    STATE { REAL speed, err, vr; BOOL gear1, gear2, gear3, gear4, gear5; }
    PARAMETER {…}}
IMPLEMENTATION {
    AUX {
        REAL Fe1, Fe2, Fe3, Fe4, Fe5, w1, w2, w3, w4, w5, DCe1, DCe2, DCe3, DCe4,zut, zub, ierr, torque, F_brake;
        BOOL dPWL1, dPWL2, dPWL3, dPWL4, sd, su, verr, sat_torque, sat_F_brake, no_sat;}
    LOGIC { no_sat = ~(sat_torque | sat_F_brake) & verr; }
    AD {dPWL1 = wPWL1 - (w1 + w2 + w3 + w4 + w5) <= 0; dPWL2 = wPWL2 - (w1 + w2 + w3 + w4 + w5) <= 0;
        dPWL3 = wPWL3 - (w1 + w2 + w3 + w4 + w5) <= 0; dPWL4 = wPWL4 - (w1 + w2 + w3 + w4 + w5) <= 0;

        sd = (w1 + w2 + w3 + w4 + w5) - wl <= 0; su = wu - (w1 + w2 + w3 + w4 + w5) <= 0; verr = speed - vr - 2 <= 0;
        sat_torque = - zut + (DCe1 + DCe2 + DCe3 + DCe4) + 1 <= 0; sat_F_brake = - zub + max_brake_force <= 0;          }
    DA {Fe1 = {IF gear1 THEN torque / speed_factor * Rgear1}; Fe2 = {IF gear2 THEN torque / speed_factor * Rgear2};
        Fe3 = {IF gear3 THEN torque / speed_factor * Rgear3}; Fe4 = {IF gear4 THEN torque / speed_factor * Rgear4};
        Fe5 = {IF gear5 THEN torque / speed_factor * Rgear5};

        w1 = {IF gear1 THEN speed / speed_factor * Rgear1}; w2 = {IF gear2 THEN speed / speed_factor * Rgear2};
        w3 = {IF gear3 THEN speed / speed_factor * Rgear3}; w4 = {IF gear4 THEN speed / speed_factor * Rgear4};
        w5 = {IF gear5 THEN speed / speed_factor * Rgear5};

        DCe1 = {IF dPWL1 THEN (aPWL2) + (bPWL2) * (w1 + w2 + w3 + w4 + w5) ELSE (aPWL1) + (bPWL1) * (w1 + w2 + w3 + w4 + w5)};
        DCe2 = {IF dPWL2 THEN (aPWL3 - aPWL2) + (bPWL3 - bPWL2) * (w1 + w2 + w3 + w4 + w5)};
        DCe3 = {IF dPWL3 THEN (aPWL4 - aPWL3) + (bPWL4 - bPWL3) * (w1 + w2 + w3 + w4 + w5)};
        DCe4 = {IF dPWL4 THEN (aPWL5 - aPWL4) + (bPWL5 - bPWL4) * (w1 + w2 + w3 + w4 + w5)};

        zut = {IF verr THEN kt * (vr - speed) + it * err}; zub = {IF ~verr THEN - kb * (vr - speed) - ib * err};
        torque ={IF sat_torque THEN (DCe1 + DCe2 + DCe3 + DCe4) + 1 ELSE zut}; F_brake = {IF sat_F_brake THEN max_brake_force ELSE zub};
        ierr = {IF no_sat THEN err + Ts * (vr - speed)};}
    CONTINUOUS {
        speed = speed + Ts / mass * (Fe1 + Fe2 + Fe3 + Fe4 + Fe5 - F_brake - beta_friction * speed); err = ierr; vr = vr;}
    AUTOMATA {
        gear1 =                (gear2 & sd) | (gear1 &       ~su); gear2 = (gear1 & su) | (gear3 & sd) | (gear2 & ~sd & ~su);
        gear3 = (gear2 & su) | (gear4 & sd) | (gear3 & ~sd & ~su); gear4 = (gear3 & su) | (gear5 & sd) | (gear4 & ~sd & ~su);
        gear5 = (gear4 & su)                | (gear5 & ~sd );}
    MUST {
        -w1 <= -wemin; w1 <= wemax; -w2 <= -wemin; w2 <= wemax; -w3 <= -wemin; w3 <= wemax; -w4 <= -wemin; w4 <= wemax; -w5 <= -wemin;
        w5 <= wemax; -F_brake <= 0; F_brake <= max_brake_force; torque - (DCe1 + DCe2 + DCe3 + DCe4) - 1 <= 0;
        -((REAL gear1) + (REAL gear2) + (REAL gear3) + (REAL gear4) + (REAL gear5)) <= -0.9999;
        (REAL gear1) + (REAL gear2) + (REAL gear3) + (REAL gear4) + (REAL gear5)  <= 1.0001;
        dPWL4 -> dPWL3; dPWL4 -> dPWL2; dPWL4 -> dPWL1; dPWL3 -> dPWL2; dPWL3 -> dPWL1; dPWL2 -> dPWL1;}}
```

Hybrid Model

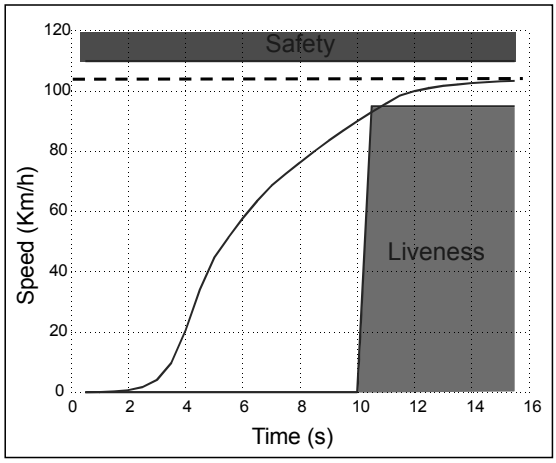
- MLD model

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

- 3 continuous states: v, v_r, e (speed, reference and tracking error)
- 5 binary states: g_1, g_2, g_3, g_4, g_5 (gears)
- 19 auxiliary continuous vars: (5 traction force, 5 engine speed, 5 reset/saturation, 4 PWL max engine torque)
- 15 auxiliary binary vars: (4 PWL max torque breakpoints, 4 saturations 5 logic updates, 2 gear switching conditions)
- 173 mixed-integer inequalities

Verification Results

- For all $v_r \in [30, 70]\ km/h$ the controller satisfies both liveness & safety properties
- CPU time: ~2.5h (Matlab 5.3, PC650MHz)



- For $v_r \in [30, 120]\ km/h$ the verification algorithm finds the first counterexample after ~7m

Conclusions

- **Hybrid systems** as a framework for new applications, where both logic and continuous dynamics are relevant
- **Mixed Logical Dynamical (MLD)** systems as discrete-time, *computation-oriented* models for hybrid systems

```
/* HEAT EXCHANGE model  - (C) 2001 by A. Bemporad, Zurich, March 13, 2001 */

SYSTEM furnaces {

INTERFACE {
        STATE {
                REAL t1,t2,u0; }
        INPUT {
                BOOL heat1,heat2; }
        PARAMETER {
                REAL Ts=0.08; /* sampling time, seconds */
                REAL Bd1=0.07688365361336; /* discretization of B matrix, heat 1 active */
                REAL Bd2=0.07392810551689; /* discretization of B matrix, heat 2 active */
                REAL A11=.9231; /* discretization of A matrix */
                REAL A22=.8521; /* discretization of A matrix */

                REAL u0max=10; /* upperbound on u0 */
                REAL e = 1e-6; /* precision for strict inequalities */  }
}

IMPLEMENTATION {
                AUX {REAL z1,z2; }

                DA {    z1 = (IF heat1 THEN Bd1*u0 [Bd1*u0max,0,e]);
                        z2 = (IF heat2 THEN Bd2*u0 [Bd2*u0max,0,e]); }

                CONTINUOUS { t1 = A11*t1+z1;
                             t2 = A22*t2+z2;
                             u0 = u0; }

                MUST   {  -heat1 | -heat2;  /* heat1 and heat2 cannot be both active */ }
}
}
```

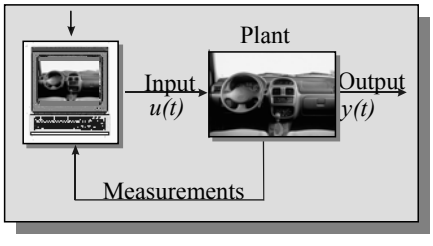
HYSDEL

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

Conclusions

- **Hybrid systems** as a framework for new applications, where both logic and continuous dynamics are relevant
- **Mixed Logical Dynamical (MLD)** systems as discrete-time, *computation-oriented* models for hybrid systems

- **Supervisory MPC controllers** and **State Estimation/Fault Detection** schemes can be synthesized via on-line mixed-integer programming (MILP/MIQP)
- **Piecewise Linear Optimal Controllers** can be synthesized via off-line multiparametric programming for fast-sampling applications



$$
u(x) = \begin{cases} F_1x + G_1 & \text{if} \quad H_1x \leq K_1 \\ \vdots & \vdots \\ F_Nx + G_N & \text{if} \quad H_Nx \leq K_N \end{cases}
$$

Conclusions

- **Hybrid systems** as a framework for new applications, where both logic and continuous dynamics are relevant
- **Mixed Logical Dynamical (MLD)** systems as discrete-time, *computation-oriented* models for hybrid systems
- **Supervisory MPC controllers** and **State Estimation/Fault Detection** schemes can be synthesized via on-line mixed-integer programming (MILP/MIQP)
- **Piecewise Linear Optimal Controllers** can be synthesized via off-line multiparametric programming for fast-sampling applications
- **Safety Analysis** properties can be formally verified

References

[1] A. Bemporad and M. Morari. Control of systems integrating logic, dynamics, and constraints. *Automatica*, 35(3):407–427, March 1999.

[2] A. Bemporad, F. Borrelli, and M. Morari. Piecewise linear optimal controllers for hybrid systems. In *American Control Conference*, pages 1190–1194, Chicago, IL, June 2000.

[3] A. Bemporad, M. Morari, V. Dua, and E.N. Pistikopoulos. The explicit linear quadratic regulator for constrained systems. *Automatica*, 38(1):3–20, 2002.

[4] F.D. Torrisi and A. Bemporad. HYSDEL — A tool for generating computational hybrid models. *IEEE Transactions on Control Systems Technology*, 2002. Accepted for publication. `http://control.ethz.ch/~hybrid/hysdel`.

[5] A. Bemporad, W.P.M.H. Heemels, and B. De Schutter. On hybrid systems and closed-loop MPC systems. *IEEE Trans. Automatic Control*, 47(5):863–869, May 2002.

[6] F.D. Torrisi and A. Bemporad. Discrete-time hybrid modeling and verification. In *Proc. 40th IEEE Conf. on Decision and Control*, pages 2899–2904, Orlando, Florida, 2001.

[7] A. Bemporad, G. Ferrari-Trecate, and M. Morari. Observability and controllability of piecewise affine and hybrid systems. *IEEE Trans. Automatic Control*, 45(10):1864–1876, 2000.

[8] W.P.M.H. Heemels, B. De Schutter, and A. Bemporad. Equivalence of hybrid dynamical models. *Automatica*, 37(7):1085–1091, July 2001.

Paper download: **`http://www.dii.unisi.it/~bemporad`**