Tutorial on

Model Predictive Control of Hybrid Systems

Alberto Bemporad

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

UNIVERSITY OF SIENA, ITALY

Dept. Information Engineering

Advanced Process Control Applications for Industry Workshop (APC 2007)

Vancouver, British Columbia, May 14, 2007

Structure of the Tutorial

- Models of hybrid systems
- Model predictive control (MPC) of hybrid systems
- Computational aspects of hybrid MPC
- Explicit MPC (multiparametric programming)
- Stochastic and event-based hybrid MPC
- Application examples

Slides download:

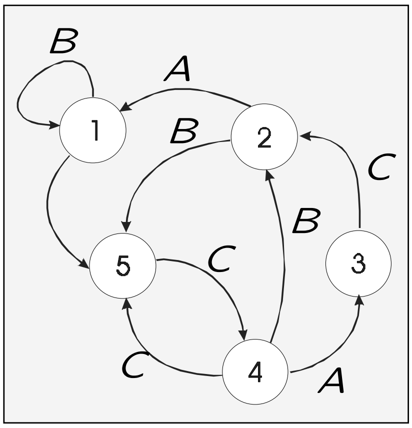
http://www.dii.unisi.it/hybrid/workshop/apc07.pdf

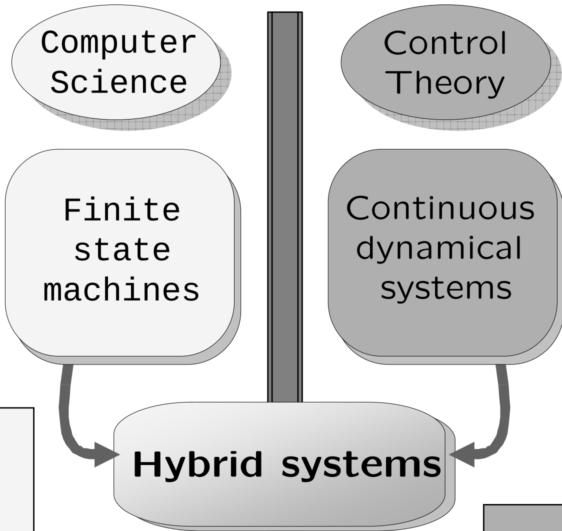
Models of Hybrid Systems

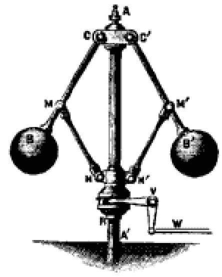
Hybrid Systems

$\mathtt{x} \in \{1, 2, 3, 4, 5\}$

$\mathtt{u} \in \{A, B, C\}$



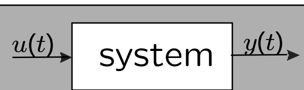




$x \in \mathbb{R}^n$

$u \in \mathbb{R}^m$

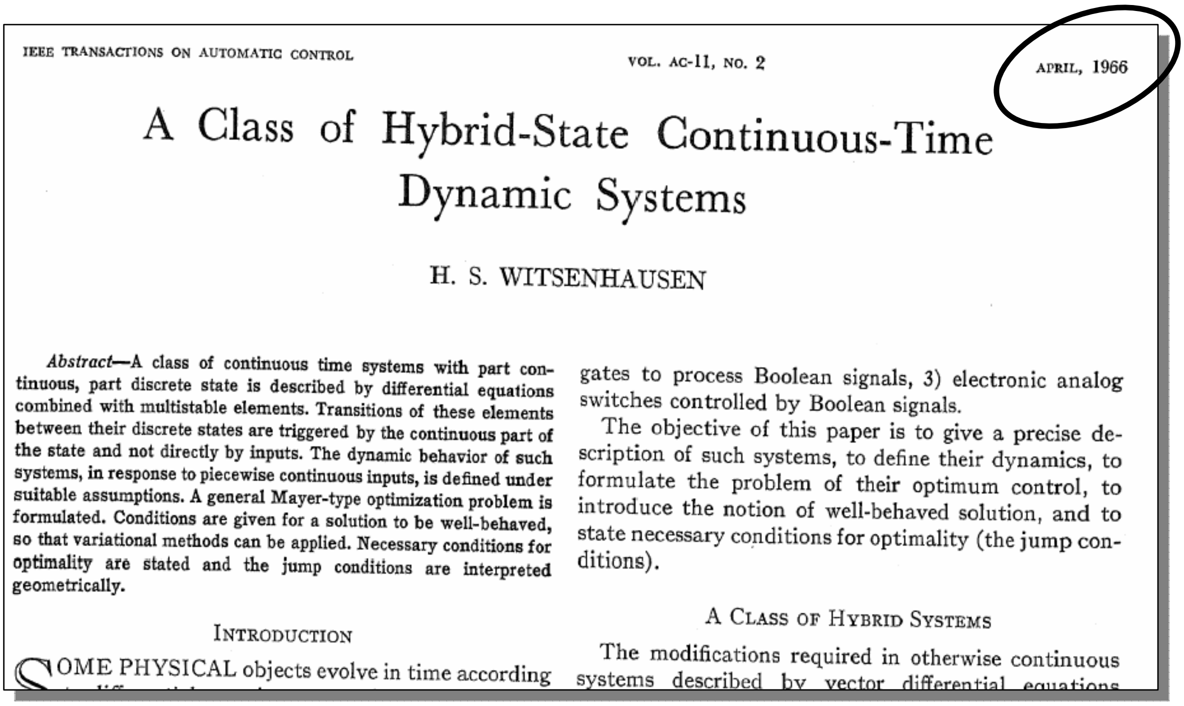
$y \in \mathbb{R}^p$



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

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

Hybrid Systems


IEEE TRANSACTIONS ON AUTOMATIC CONTROL VOL. AC-11, NO. 2 APRIL, 1966

A Class of Hybrid-State Continuous-Time Dynamic Systems

H. S. WITSENHAUSEN

Abstract—A class of continuous time systems with part continuous, part discrete state is described by differential equations combined with multistable elements. Transitions of these elements between their discrete states are triggered by the continuous part of the state and not directly by inputs. The dynamic behavior of such systems, in response to piecewise continuous inputs, is defined under suitable assumptions. A general Mayer-type optimization problem is formulated. Conditions are given for a solution to be well-behaved, so that variational methods can be applied. Necessary conditions for optimality are stated and the jump conditions are interpreted geometrically.

INTRODUCTION

SOME PHYSICAL objects evolve in time according

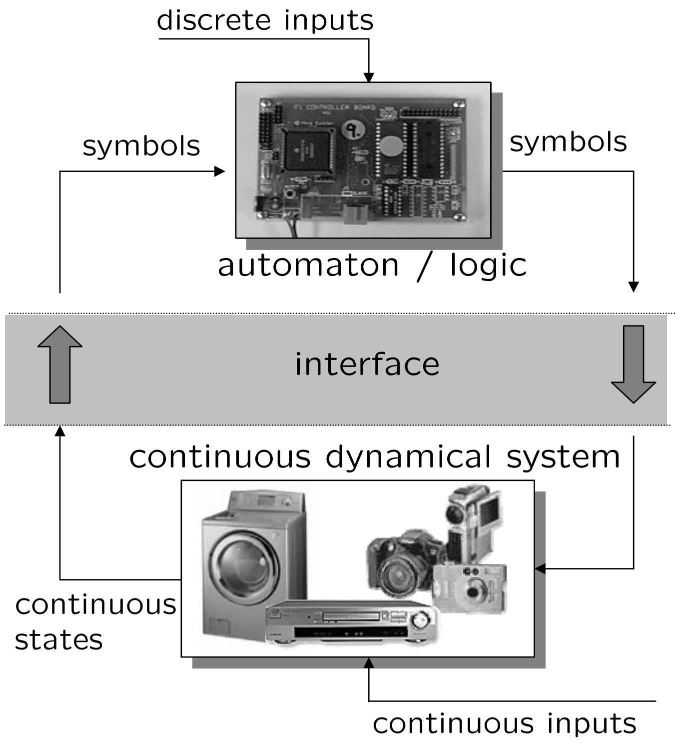
gates to process Boolean signals, 3) electronic analog switches controlled by Boolean signals.

The objective of this paper is to give a precise description of such systems, to define their dynamics, to formulate the problem of their optimum control, to introduce the notion of well-behaved solution, and to state necessary conditions for optimality (the jump conditions).

A CLASS OF HYBRID SYSTEMS

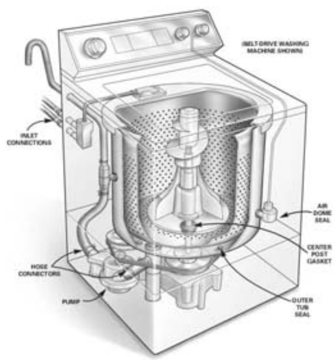
The modifications required in otherwise continuous systems described by vector differential equations

Embedded Systems



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

Example:



Key Requirements for Hybrid Models

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

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

?

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

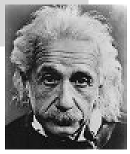
linear systems

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

nonlinear systems

linear hybrid systems

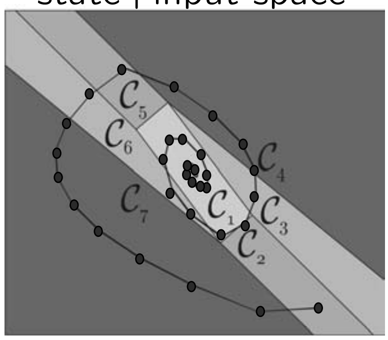
“Make everything as simple as possible, but not simpler.”
— Albert Einstein



Piecewise Affine Systems

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

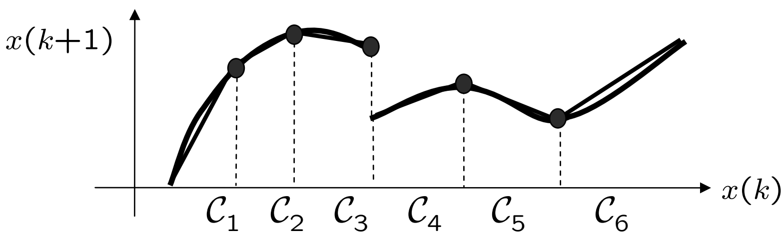
state+input space



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

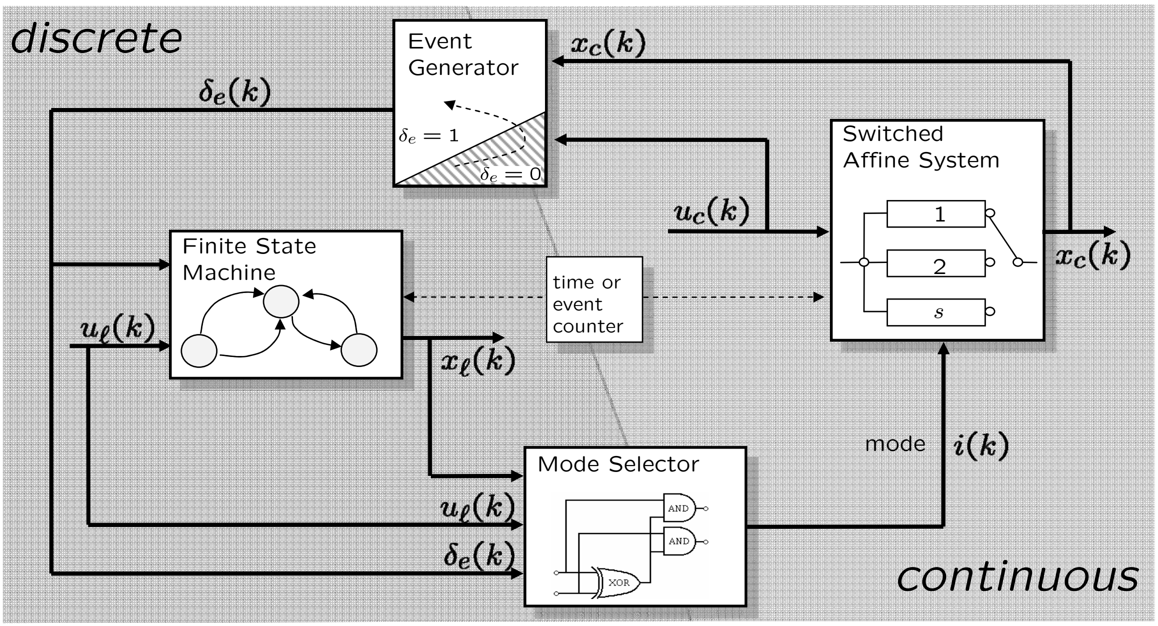
(Sontag 1981)

Can approximate nonlinear and/or discontinuous dynamics arbitrarily well



Discrete Hybrid Automaton

(Torrisi, Bemporad, 2004)



$x_\ell \in \{0,1\}^{n_b}$ = binary states

$u_\ell \in \{0,1\}^{m_b}$ = binary inputs

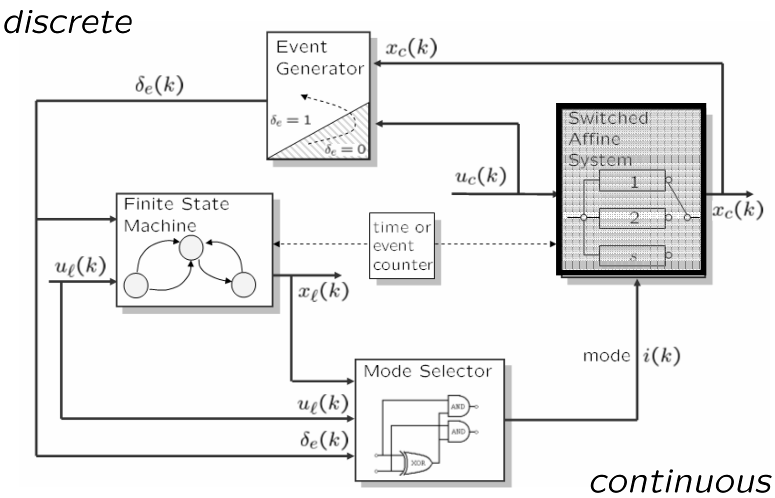
$\delta_e \in \{0,1\}^{n_e}$ = event variables

$x_c \in \mathbb{R}^{n_c}$ = continuous states

$u_c \in \mathbb{R}^{m_c}$ = continuous inputs

$i \in \{1, 2, \ldots, s\}$ = current mode

Switched Affine System

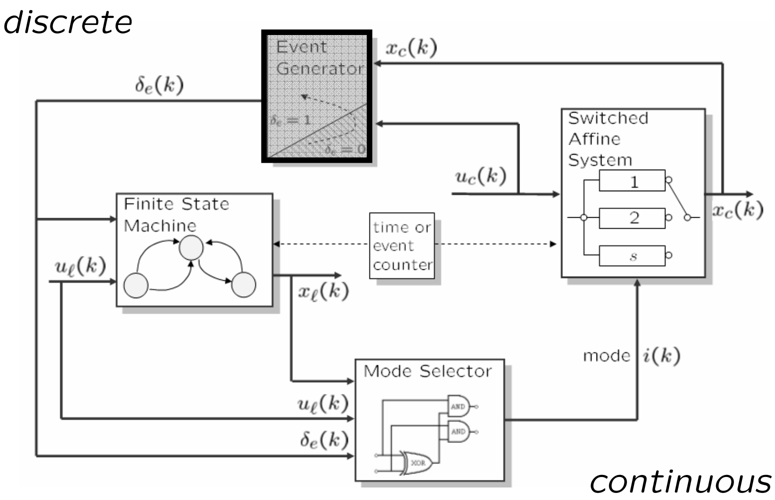


The affine dynamics depend on the current mode $i(k)$:

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

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

Event Generator



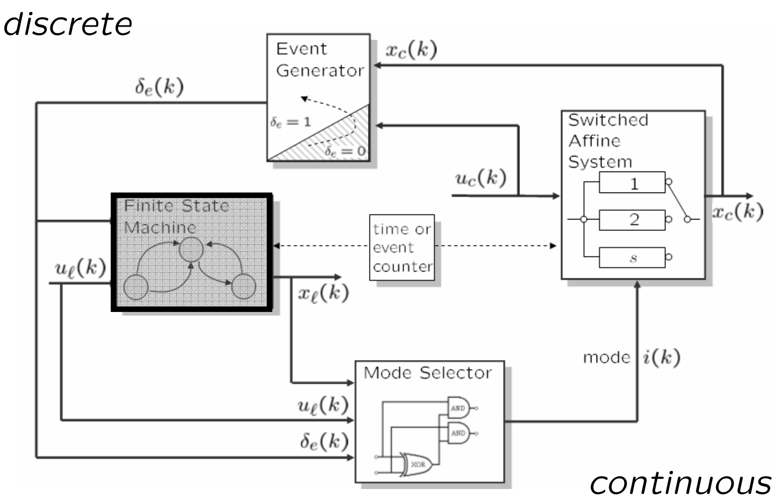
Event variables are generated by linear threshold conditions over continuous states, continuous inputs, and time:

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

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

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

Finite State Machine



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

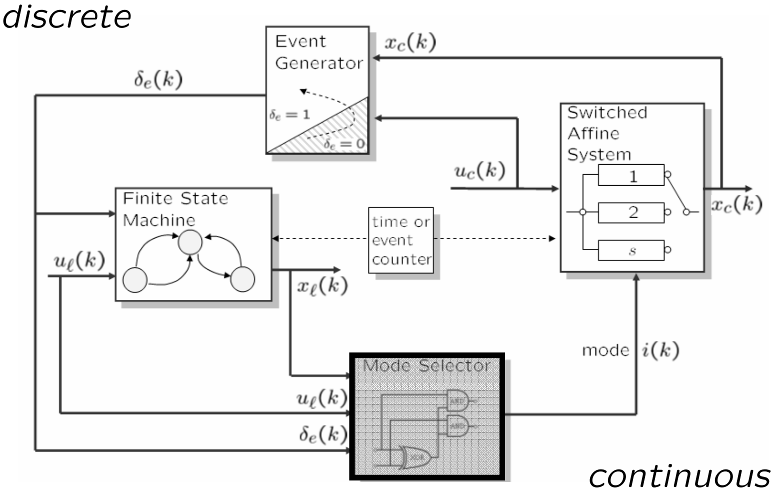
$$x_\ell(k+1) = f_{\mathsf{B}}(x_\ell(k), u_\ell(k), \delta_e(k))$$

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

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

Mode Selector

discrete



continuous

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

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

$$i(k) = f_{\mathsf{M}}(x_\ell(k), u_\ell(k), \delta_e(k))$$

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

Example:

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

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

the system has 3 modes

Logic and Inequalities

(Glover 1975, Williams 1977, Hooker 2000)

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

Any logic statement

$f(X) =$TRUE

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

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

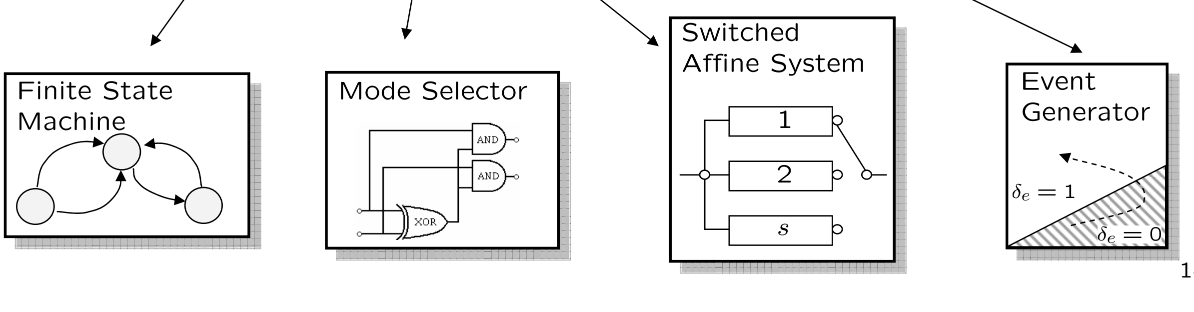
$$\begin{cases} 1 \leq \sum_{i \in P_1} \delta_i + \sum_{i \in N_1} (1 - \delta_i) \\ \vdots \\ 1 \leq \sum_{i \in P_m} \delta_i + \sum_{i \in N_m} (1 - \delta_i) \end{cases}$$

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

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

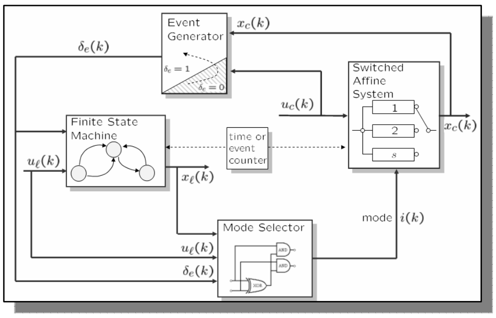
IF $[\delta = 1]$ THEN $z = a_1^T x + b_1^T u + f_1$
ELSE $z = a_2^T x + b_2^T u + f_2$

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



Mixed Logical Dynamical Systems

Discrete Hybrid Automaton



HYSDEL

(Torrisi, Bemporad, 2004)

Mixed Logical Dynamical (MLD) Systems (Bemporad, Morari 1999)

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

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 model (mixed-integer programming)

HYSDEL

(HYbrid Systems DEscription Language)

- Describe *hybrid systems:*

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

(Torrisi, Bemporad, 2004)

- Automatically generate MLD models in Matlab

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

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

Hybrid Toolbox for Matlab

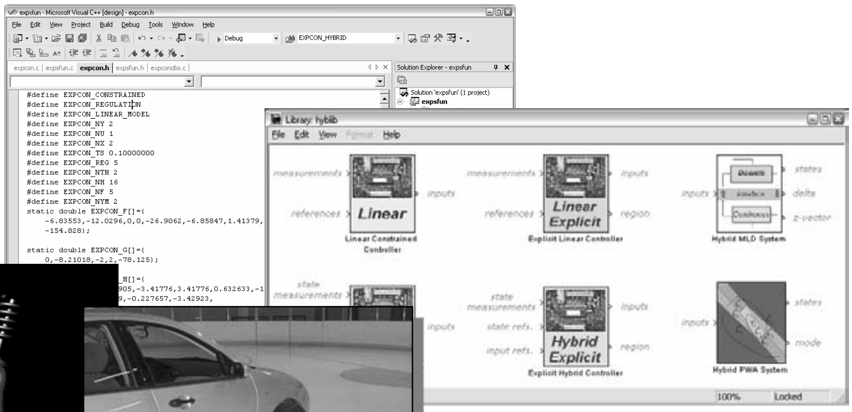
(Bemporad, 2003-2007)

Features:

- Hybrid model (MLD and PWA) design, simulation, verification
- Control design for linear systems w/ constraints and hybrid systems (on-line optimization via QP/MILP/MIQP)
- Explicit control (via multiparametric programming)
- C-code generation
- Simulink

Support:

FORD RESEARCH LABORATORY



>1300 downloads in 2 years

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

Mixed-Integer Models in OR

Translation of logical relations into linear inequalities is heavily used in operations research (OR) for solving complex decision problems by using mixed-integer programming (MIP)

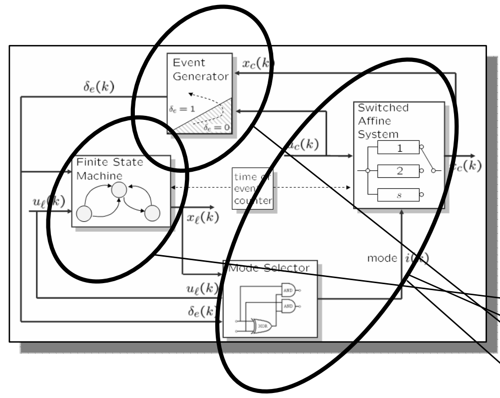
Example: Optimal investments for quality of supply improvement in electrical energy *distribution networks* (Bemporad, Muñoz, Piazzesi, 2006)

Example: Timetable generation (for demanding professors ...)

	8	9	10	11	12	13	14	15	16	17	18	19
lun		Sistemi Operativi (°18)					Misure per la Automazione (°7)		Ingegneria del Software (°18)			
							Basi di Dati (°18)					
mar		Basi di Dati (°3)		Sistemi Operativi (°3)			Robotica ed Automazione di Processo (°18)					
mer		Robotica ed Automazione di Processo (°8)		Misure per la Automazione (°7)				Laboratorio di Robotica e Realtà Virtuale (°15)				
		Ingegneria del Software (°18)										
gio		Basi di Dati (°3)					Sistemi Operativi (°5)					
		Laboratorio di Robotica e Realtà Virtuale (°15)										
ven		Robotica ed Automazione di Processo (°8)					Misure per la Automazione (°7)					
		Ingegneria del Software (°18)										
sab												

CPU time: 0.2 s

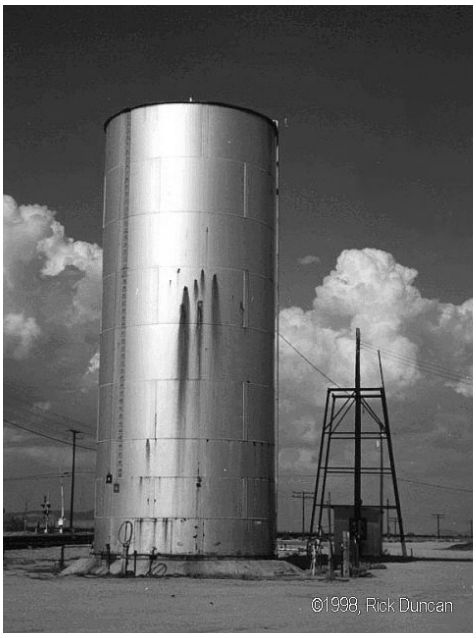
DHA and HYSDEL Models



Additional relations constraining system's variables

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

   IMPLEMENTATION {
      AUX { BOOL d;
            REAL z; }

      AUTOMATA { xl = xl & ~ul; }

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

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

      CONTINUOUS {
         xc = z; }

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


Example 1: Definition of Event Vars.



$$[s = 1] \leftrightarrow [h \geq h_{\text{max}}]$$

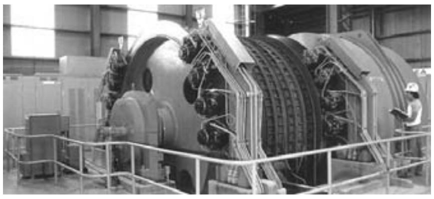
$s \in \{0, 1\}$
$h \in \mathbb{R}$

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

Example 2: Nonlinear (PWA) Functions

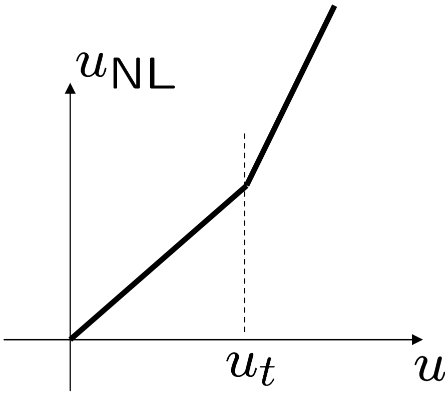
Nonlinear amplification unit

$$u_{\text{NL}}(k+1) = \begin{cases} u(k) & \text{if } u(k) < u_t \\ 2.3u(k) - 1.3u_t & \text{if } u(k) \geq u_t \end{cases}$$



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


AD { th = ut - u <= 0; }
```

$[t_h = 1] \leftrightarrow [u \geq u_t]$



Example 3: Logical relations

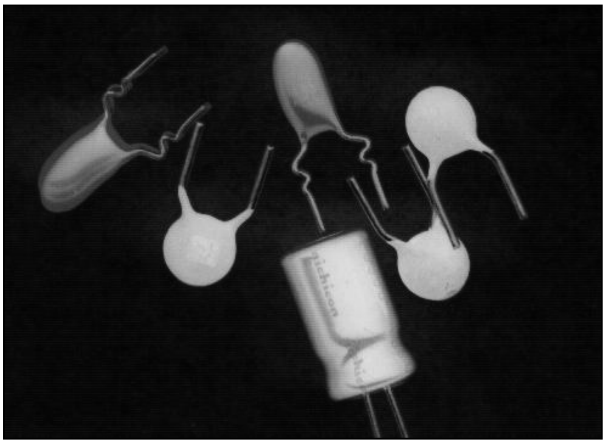
<u>Rule</u>: brake if there is an alarm signal, but only if the train is not on fire in a tunnel

$\delta_{\text{brake}}, \delta_{\text{alarm}}, \delta_{\text{tunnel}}, \delta_{\text{fire}} \in \{0, 1\}$

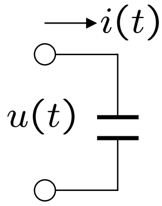
$$\delta_{\text{brake}} = \delta_{\text{alarm}} \wedge (\neg\delta_{\text{tunnel}} \vee \neg\delta_{\text{fire}})$$

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

Example 4: Continuous dynamics



$$i(t) = C\frac{du(t)}{dt}$$

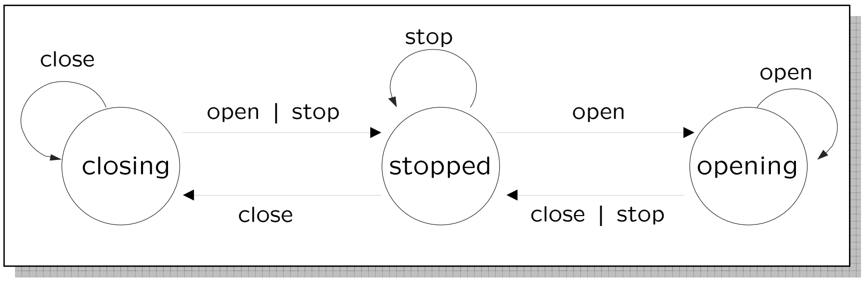


Apply forward difference rule:

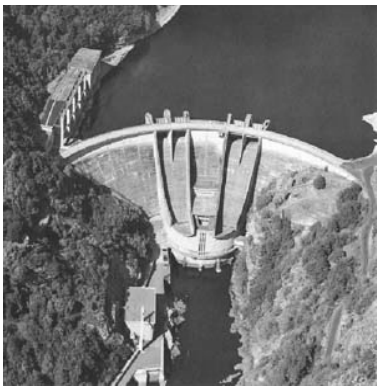
$$u((k+1)T) = u(kT) + \frac{T}{C}i(kT)$$

```
CONTINUOUS {
      u = u + T/C*i;
}
```


Example 5: Automaton



Flow control through a dam



binary inputs: $u_{\text{open}}, u_{\text{close}}, u_{\text{stop}} \in \{0,1\}$

binary states: $x_{\text{opening}}, x_{\text{closing}}, x_{\text{stopped}} \in \{0,1\}$

```
AUTOMATA {
   xclosing = (uclose & xclosing) | (uclose & xstopped);
   xstopped = ustop | (uopen & xclosing) | (uclose & xopening);
   xopening = (uopen & xstopped) | (uopen & xopening);
}
```

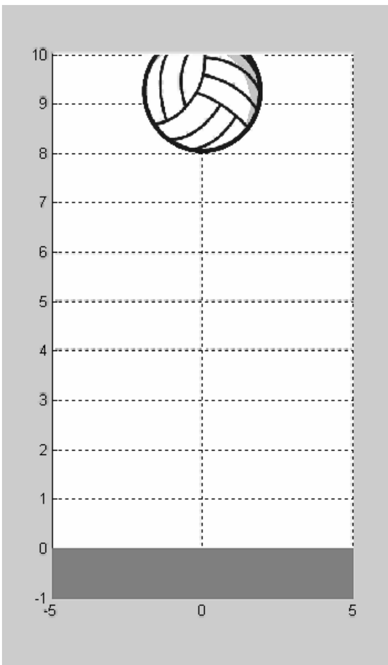
Example 6: Impose a constraint

$$0 \le h(k) \le h_{\text{max}}$$

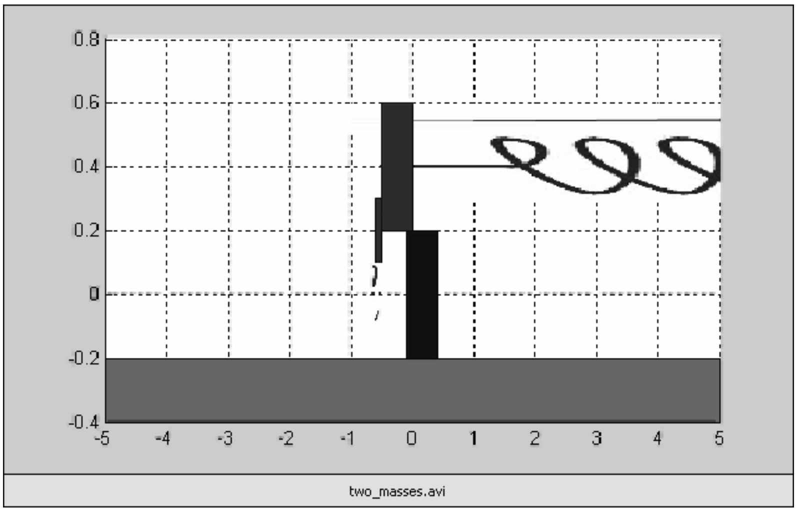
```
MUST {
     h - hmax <= 0;
     -h       <= 0; }
}
```


Examples: Systems with Impacts

bouncing ball



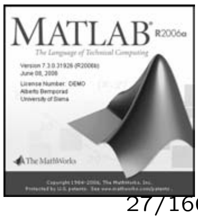
electromagnetically actuated fuel injector



Systems easily modeled in discrete-time linear hybrid form

HYSDEL - Bouncing Ball

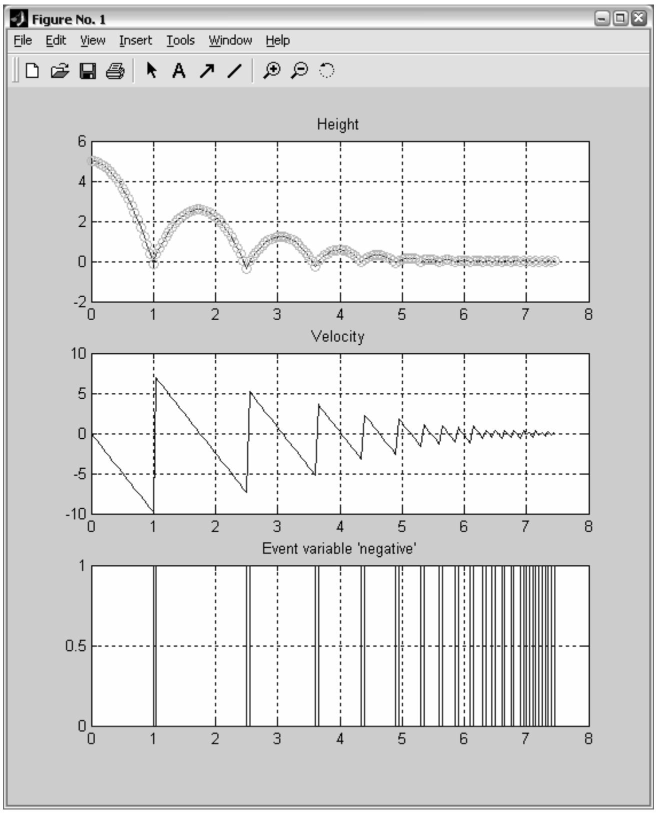
```
SYSTEM bouncing_ball {
INTERFACE {
/* Description of variables and constants */
        STATE { REAL height [-10,10];
                REAL velocity [-100,100];   }

        PARAMETER {
                REAL g;
                REAL alpha;  /* 0=elastic, 1=completely anelastic */
                REAL Ts; }
}
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-alpha)*velocity
                        ELSE velocity-Ts*g};  }
        CONTINUOUS {
                height = z1;
                velocity=z2;}
}}
```



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

Bouncing Ball

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

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

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

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



<u>Note</u>: no Zeno effects in discrete time !

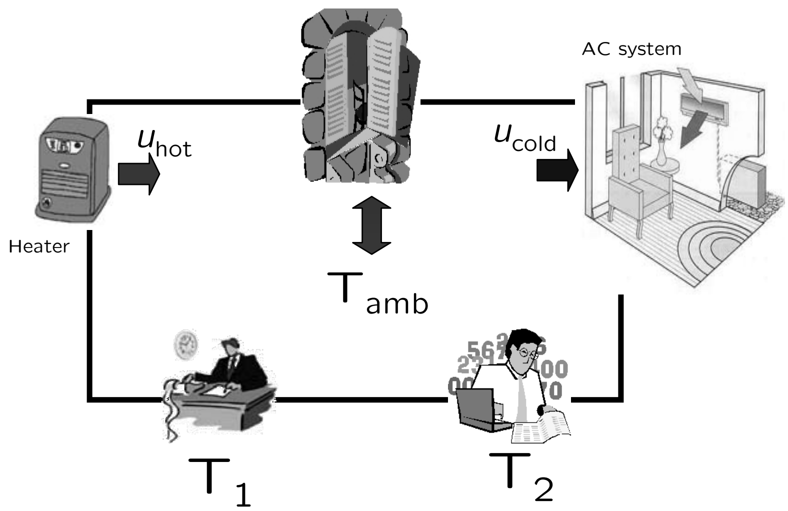
MLD and PWA Systems

Theorem MLD systems and PWA systems are equivalent

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

- 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
- Efficient algorithms are available for converting MLD models into PWA models avoiding such an enumeration:
  - A. Bemporad, "*Efficient Algorithms for Converting Mixed Logical Dynamical Systems into an Equivalent Piecewise Affine Form*", IEEE Trans. Autom. Contr., 2004.
  - T. Geyer, F.D. Torrisi and M. Morari, "*Efficient Mode Enumeration of Compositional Hybrid Models*", HSCC'03

Example: Room Temperature



Hybrid dynamics

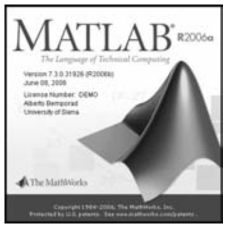
- #1 turns the heater (air conditioning) on whenever he is cold (hot)
- If #2 is cold he turns the heater on, unless #1 is hot
- If #2 is hot he turns the air conditioning on, unless #2 is cold
- Otherwise, heater and air conditioning are off

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

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

HYSDEL Model

```
SYSTEM heatcool {

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



Hybrid Toolbox for Matlab

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

```
>>S=mld('heatcoolmodel',Ts)
```
get the MLD model in Matlab

```
>>[XX,TT]=sim(S,x0,U);
```
simulate the MLD model

Hybrid MLD Model

- MLD model

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

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

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

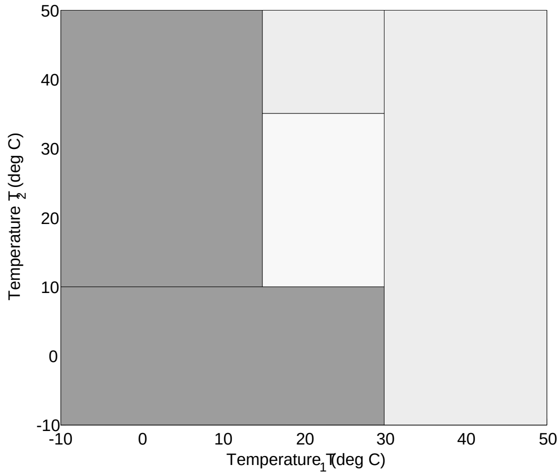
Hybrid PWA Model

- PWA model

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

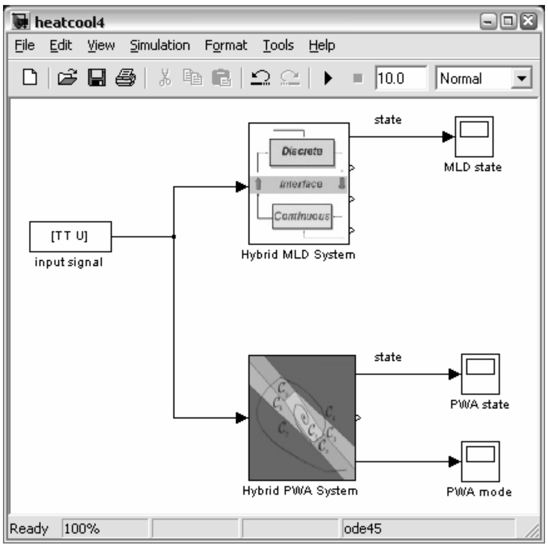
- 2 continuous states:
  (temperatures T_1,T_2)
- 1 continuous input:
  (room temperature T_{amb})
- 5 polyhedral regions
  (partition does not depend on input)

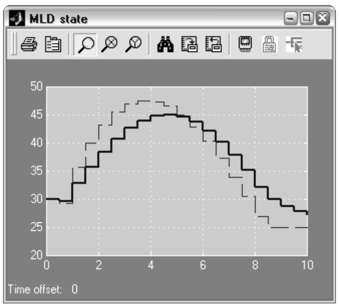
```
>>P=pwa(S);
```

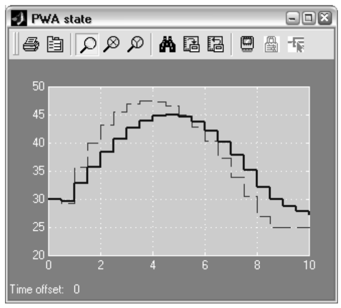


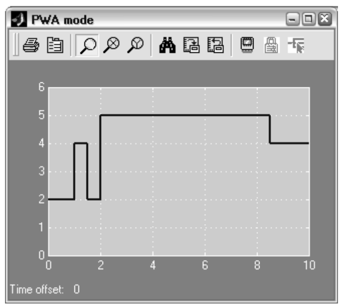
$u_{\text{hot}} = 0$, $u_{\text{cold}} = 0$ | $u_{\text{hot}} = 0$, $u_{\text{cold}} = \bar{U}_C$ | $u_{\text{hot}} = \bar{U}_H$, $u_{\text{cold}} = 0$

Simulation in Simulink









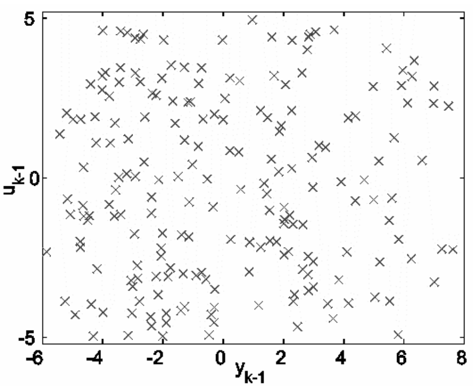
MLD and PWA models are equivalent

Using MLD→PWA for Model Checking

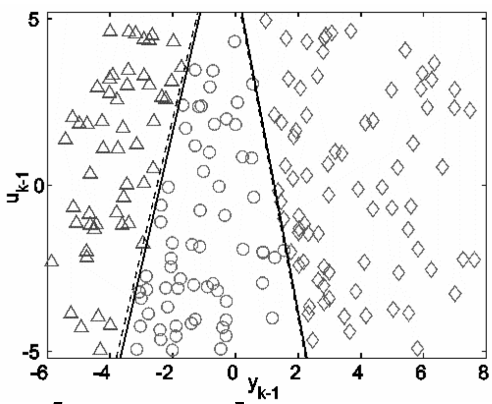
- Assume plant and controller can be modeled as DHA:
  - Plant = PWA approximation (e.g.: NL switched model)
  - Controller = switched linear controller (e.g: a combinations of threshold conditions, logics, linear feedback laws, ...)
- Write HYSDEL model, convert to MLD, then to PWA
- The resulting PWA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

Hybrid Systems Identification

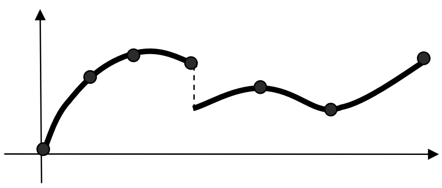
Given I/O data, estimate the parameters of the affine submodels **and** the partition of the PWA map



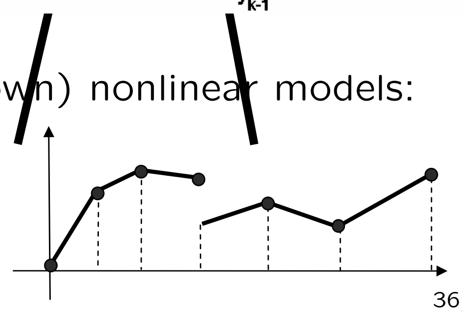
hybrid ID algorithm



Other scenario: “hybridization” of (known) nonlinear models:



hybrid ID algorithm



PWA Identification Problem

A. Known Guardlines (partition known, parameters unknown):
least-squares problem **EASY PROBLEM**
(Ljung's ID TBX)

B. Unknown Guardlines (partition *and* parameters unknown):
Generally non-convex, local minima **HARD PROBLEM!**

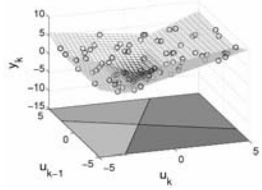
<u>Some recent approaches to Hybrid ID:</u>

- K-means clustering in a feature space (Ferrari-Trecate, Muselli, Liberati, Morari, 2003)
- Bayesian approach (Juloski, Heemels, Weiland, 2004)
- Mixed-integer programming (Roll, Bemporad, Ljung, 2004)
- Bounded error (partition of infeasible set of inequalities) (Bemporad, Garulli, Paoletti, Vicino, 2003)
- Algebraic geometric approach (Vidal, Soatto, Sastry, 2003)
- Hyperplane clustering in data space (Münz, Krebs, 2002)

Hybrid Identification Toolboxes

- PWAID – Piecewise Affine Identification Toolbox
  (Paoletti, Roll, 2007)

  `http://www.control.isy.liu.se/~roll/PWAID/`



  - Mixed-integer approach (hinging-hyperplane models)
  - Bounded error approach

- HIT – Hybrid Identification Toolbox
  (Ferrari-Trecate, 2006)

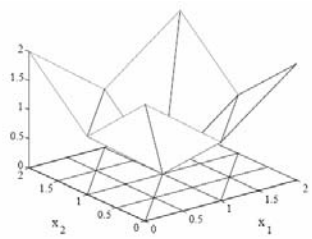
  `http://www-rocq.inria.fr/who/Giancarlo.Ferrari-Trecate/HIT_toolbox.html`

  - K-means clustering in a feature space

- PWL Toolbox
  (Julian, 2000)

  `http://www.pedrojulian.com`

  - High level canonical PWL representations



Why interested in MLD/PWA models ?

Many problems of analysis:

- Stability (Johansson, Rantzer, 1998)
- Safety / Reachability (Torrisi, Bemporad, 2001)
- Observability (Bemporad, Ferrari, Morari, 2000)
- Passivity (Bemporad, Bianchini, Brogi, 2006)
- Well-posedness (Heemels, 1999)

Many problems of synthesis:

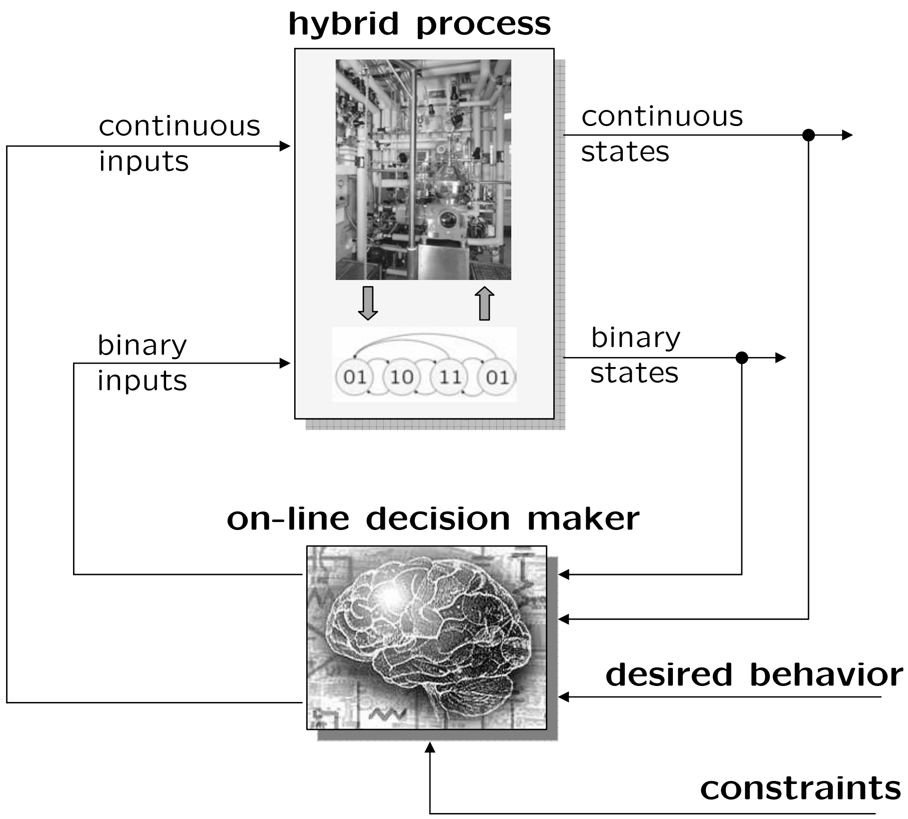
- Controller design (Bemporad, Morari, 1999)
- Filter design (state estimation/fault detection) (Bemporad, Mignone, Morari, 1999) (Ferrari, Mignone, Morari, 2002)

can be solved through mathematical programming

(However, all these problems are NP-hard !)

Model Predictive Control of Hybrid Systems

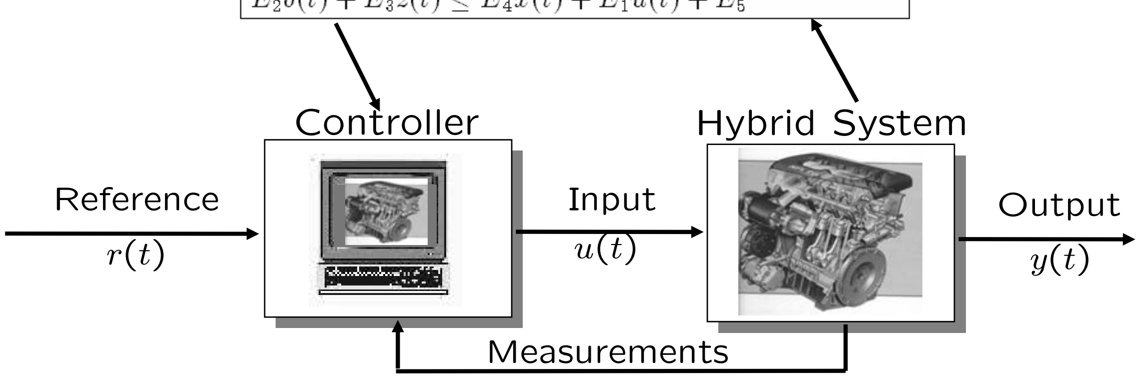
Hybrid Control Problem



Model Predictive Control of Hybrid Systems

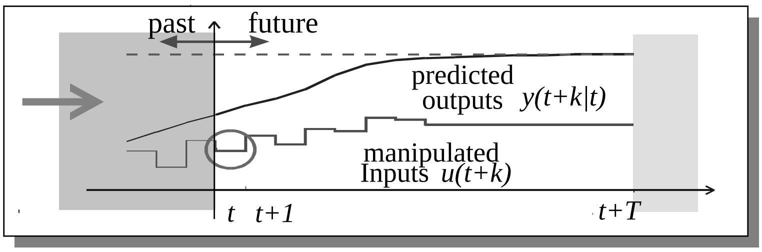
MLD model

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



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

MPC of Hybrid Systems



Model Predictive (MPC) Control

- At time t solve the open-loop optimal control problem

$$\min_{u_t,\ldots,u_{t+T-1}} \quad \sum_{k=0}^{T-1} \|y_{t+k} - r(t)\|^2 + \rho\|u_{t+k} - u_r(t)\|^2$$

subject to mixed logical dynamical (MLD) prediction model

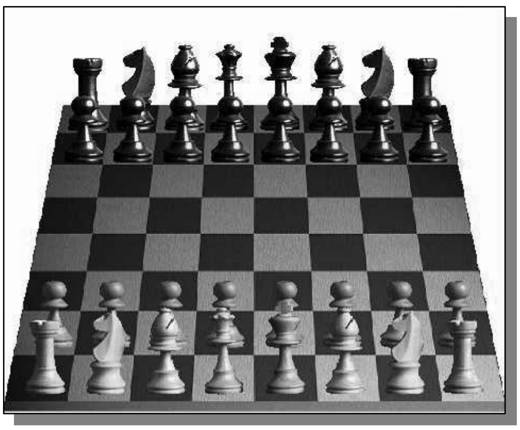
$$x_t = x(t)$$

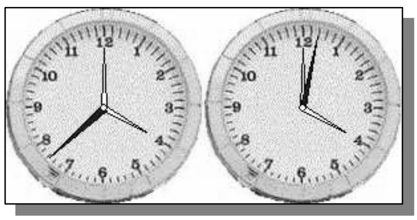
- Apply only $u(t)=u_t^*$ (discard the remaining optimal inputs)
- At time t+1: get new measurements, repeat optimization

Advantage of on-line optimization: **FEEDBACK!**

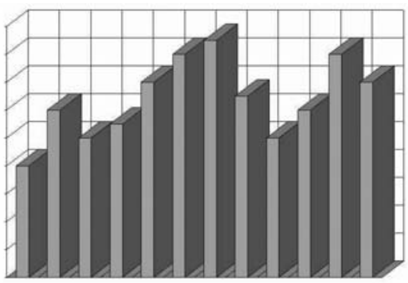
Receding Horizon - Examples

- MPC is like **playing chess** !





- “Rolling horizon” policies are also used frequently **in finance**



Receding Horizon – GPS Navigation

- **prediction model**
  how vehicle moves on the map

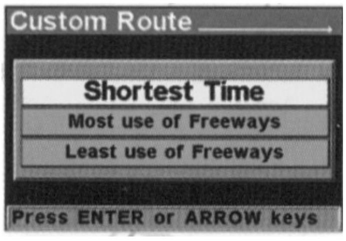
- **constraints**
  drive on roads, respect one-way roads, etc.

- **disturbances**
  road works, driver's inattention, etc.

- **set point**
  desired location

- **cost function:**
  Ex: minimum time
  Ex: penalty on highways

- **receding horizon mechanism**
  event-based (optimal route re-planned when path is lost)



It's a **feedback** strategy !

Closed-Loop Convergence

Theorem 1 *Let $(x_r, u_r, \delta_r, z_r)$ be the equilibrium values corresponding to the set point r. Assume all variables are weighted with positive definite weights and assume the terminal constraint $x_{t+T} = x_r$ is enforced. If $x(0)$ is such that the MPC problem is feasible at time $t = 0$, then*

$$\begin{aligned} \lim_{t\to\infty} y(t) &= r \\ \lim_{t\to\infty} u(t) &= u_r \end{aligned}$$

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

(Bemporad, Morari 1999)

Proof: Easily follows from standard Lyapunov arguments

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

Convergence Proof

- Assume we set the terminal constraint $x(t+T|t) = x_r$ 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 ⟹ Lyapunov function
- By construction, $\mathcal{U}_1 = \{u_t^*(1), \ldots, u_t^*(T-1), u_r\}$ is feasible @$t+1$
- Hence,

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

- 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, $\|y(t) - r\|_Q \to 0$,$\|u(t) - u_r\|_R \to 0$, . . . , $\|x(t) - x_r\| \to 0$

Note: Global optimum not needed for convergence !

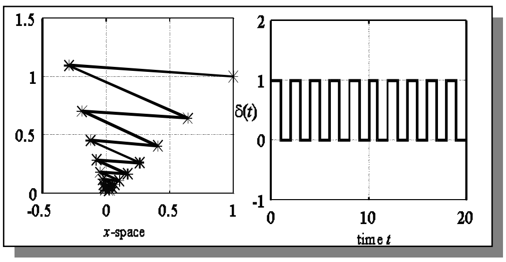
Hybrid MPC - Example

PWA system:

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

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

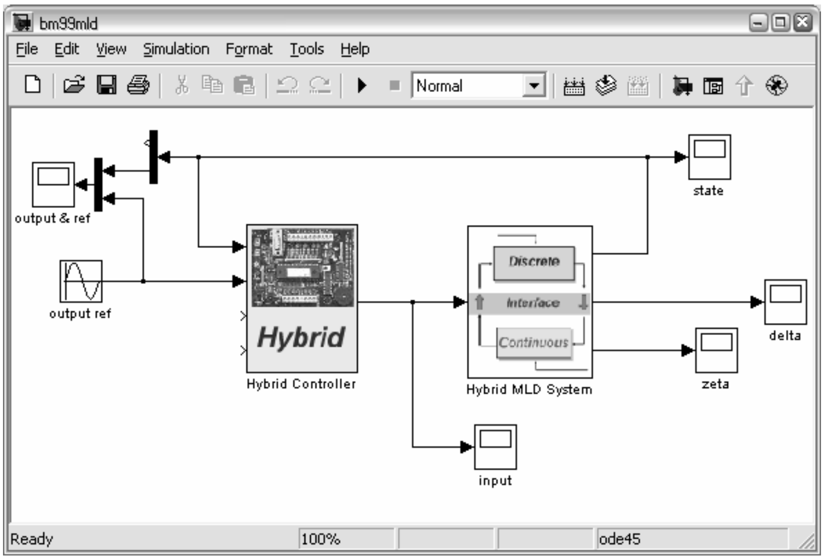
Open loop behavior

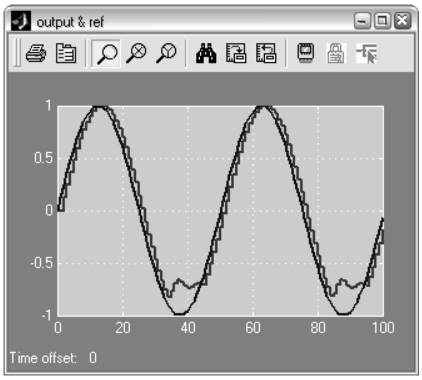


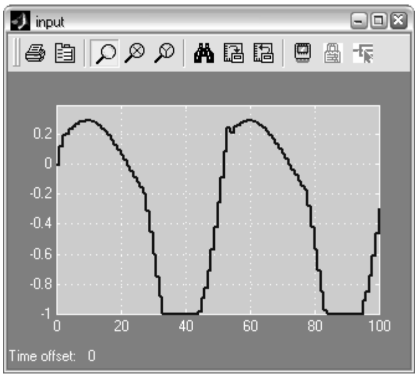
go to demo **/demos/hybrid/bm99sim.m**

Hybrid MPC - Example

Closed loop:







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

Hybrid MPC – Temperature Control

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

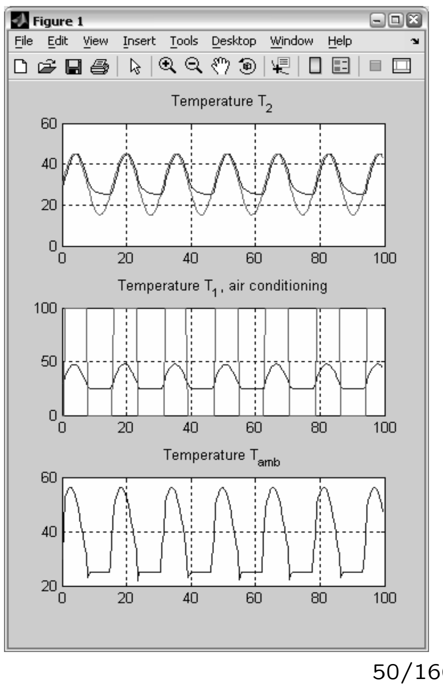
$$\min \sum_{k=1}^{2} (x_2(k) - r)^2$$

s.t. $x_1(k) \geq 25 \;\; k = 1, 2$

MLD model

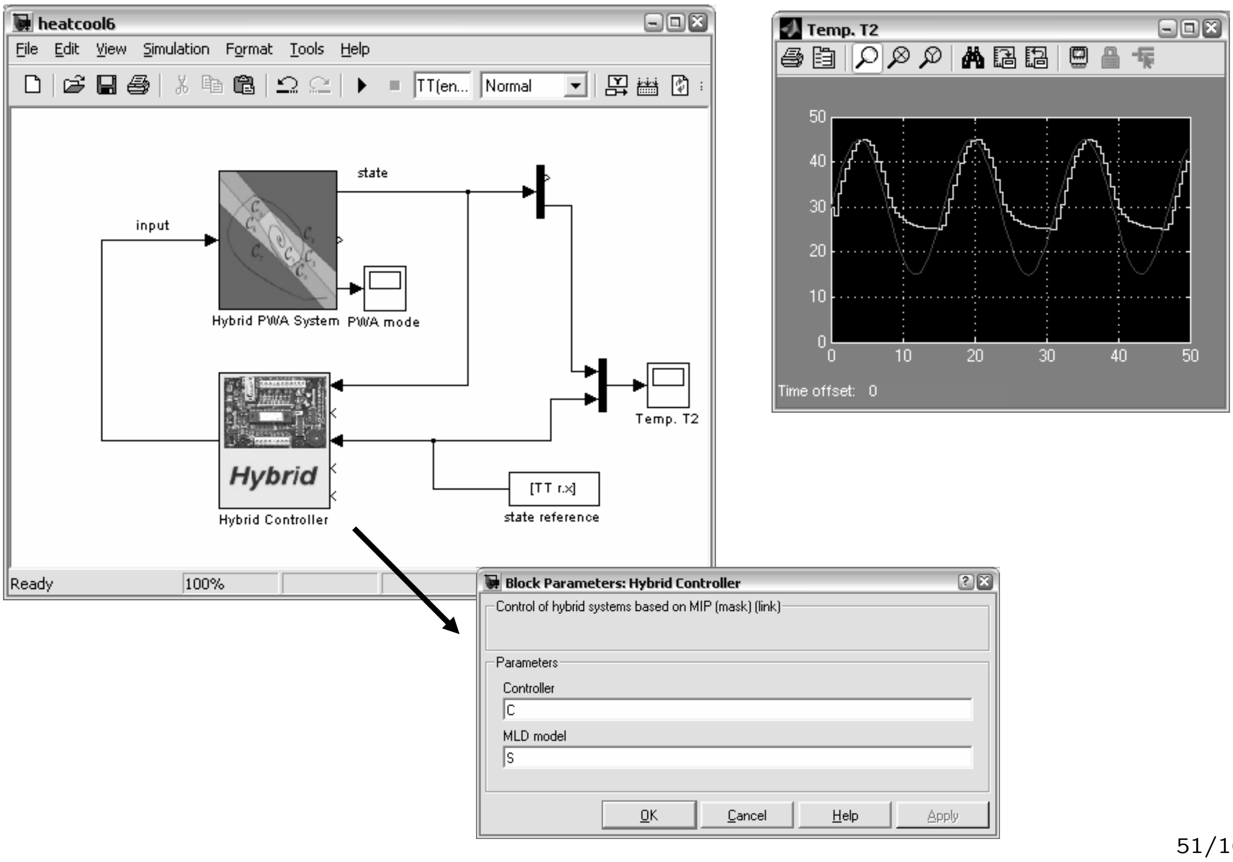
```
>>C=hybcon(S,Q,N,limits,refs);
```

```
>> C

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

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

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

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




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

Hybrid MPC – Temperature Control



Optimal Control of Hybrid Systems: Computational Aspects

MPC of Linear Systems

Linear model $\begin{cases} x_{k+1} = Ax_k + Bu_k \\ y_k = Cx_k \end{cases}$ $\quad x_0 = x(t)$

Quadratic performance index $\min_U J(x(t), U) = \sum_{k=0}^{N-1} \left[x_k' Q x_k + u_k' R u_k \right] + x_N' P x_N$

$$\begin{array}{ll} \min_U & \frac{1}{2} U' H U + x'(t) F' U + \frac{1}{2} x(t) Y x(t) \\ \text{subj. to} & GU \le W + Sx(t), \end{array} \qquad U = \begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ u_{N-1} \end{bmatrix}$$

Constraints $\begin{cases} u_{\min} \le u_k \le u_{\max} \\ y_{\min} \le y_k \le y_{\max} \end{cases}$

This is a **(convex) Quadratic Program (QP)** for all states $x(t)$

MIQP Formulation of Hybrid MPC

(Bemporad, Morari, 1999)

$$\begin{array}{rl} \min_\xi J(\xi, x(0)) & = \sum_{t=0}^{T-1} y'(t) Q y(t) + u'(t) R u(t) \\ \text{subject to} & \begin{cases} x(t+1) = Ax(t) + B_1 u(t) + B_2 \delta(t) + B_3 z(t) + B_5 \\ y(t) = Cx(t) + D_1 u(t) + D_2 \delta(t) + D_3 z(t) + D_5 \\ E_2 \delta(t) + E_3 z(t) \le E_4 x(t) + E_1 u(t) + E_5 \end{cases} \end{array}$$

- Optimization vector:

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

$$\begin{array}{ll} \min_\xi & \frac{1}{2} \xi' H \xi + x(0)' F \xi + \frac{1}{2} x'(0) Y x(0) \\ \text{subj. to} & G\xi \le W + Sx(t) \end{array}$$

Mixed Integer Quadratic Program (MIQP)

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

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

MILP Formulation of Hybrid MPC

(Bemporad, Borrelli, Morari, 2000)

$$\min_{\xi} \quad J(\xi, x(0)) = \sum_{t=0}^{T-1} \|Qy(t)\|_\infty + \|Ru(t)\|_\infty$$

subject to MLD model

- Basic trick: introduce slack variables:

$$\min_x |x| \quad \Rightarrow \quad \begin{array}{ll} \min\limits_{x,\epsilon} & \epsilon \\ \text{s.t.} & \epsilon \geq x \\ & \epsilon \geq -x \end{array}$$

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

- Optimization vector:

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

$$\Rightarrow \quad \begin{array}{ll} \min\limits_{\xi} & J(\xi, x(0)) = \sum\limits_{k=0}^{T-1} \epsilon_i^x + \epsilon_i^u \\ \text{s.t.} & G\xi \leq W + Sx(0) \end{array}$$

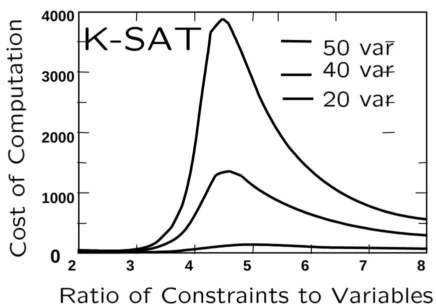
Mixed Integer Linear Program (MILP)

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

Mixed-Integer Program Solvers

- Mixed-Integer Programming is *NP*-complete

  Phase transitions have been found in computationally hard problems.



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

BUT

- Extremely rich literature in operations research (still very active)

MILP/MIQP is nowadays a technology (CPLEX, Xpress-MP, BARON, GLPK, see e.g. http://plato.la.asu.edu/bench.html for a comparison)

Mixed-Integer Program Solvers

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

  Example: restrict the number of possible modes

(Ingimundarson, Ocampo-Martinez, Bemporad, 2007)

$$\sum_{k=0}^{N-1} |\delta_k^i - \bar{\delta}_k^i| \leq M$$

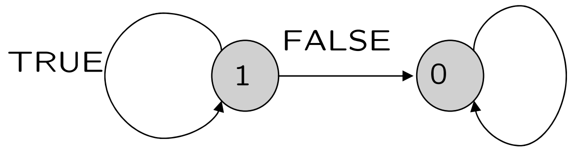
$\forall i = 1, \ldots, n_\delta$

$\bar{\delta}_k^i$ = reference mode sequence

(e.g.: obtained through simulation of MLD simulation from $x(t)$ using previous optimal input sequence)

- Possibility of exploiting logic structures of the problem

$$\begin{cases} x_\ell(k+1) + (1 - x_\ell(k)) + (1 - u_\ell(k)) \geq 1 \\ x_\ell(k) + (1 - x_\ell(k+1)) \geq 1 \\ u_\ell(k) + (1 - x_\ell(k+1)) \geq 1 \end{cases} \Longleftarrow$$



$$x_\ell(k+1) = x_\ell(k) \wedge u_\ell(k)$$

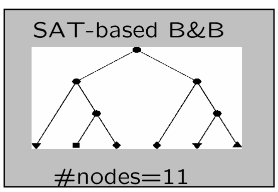
Exploiting Logic Structures

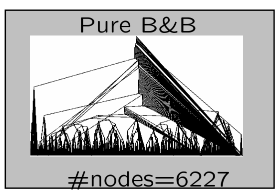
- Loss of the original Boolean structure is a drawback of Mixed-Integer Programming for solving hybrid MPC problems

- Efficiency of MIP solver usually not good when continuous LP/QP relaxations are not tight

		Sat instances		Unsat instances	
N. Vars	N. Cons	zCHAFF	CPLEX	zCHAFF	CPLEX
20	91	0	0.036	-	-
50	218	0	0.343	0	0.453
75	325	0	0.203	0	3.671
100	430	0	23.328	0	33.921
125	538	0.016	15.171	0.031	209.766
150	645	0.031	20.625	0.281	4949.58
175	753	0.031	> 1500	0.891	> 5000

satisfiability of logic formulas





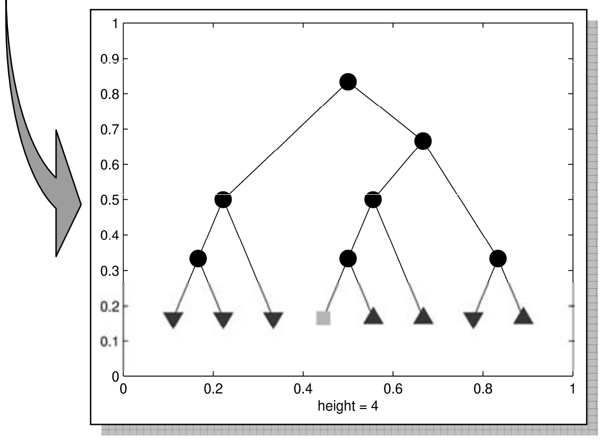
- Possibility of combining symbolic + numerical solvers
  Example: **SAT + LP** or **SAT + QP**

Exploiting Logic Structures

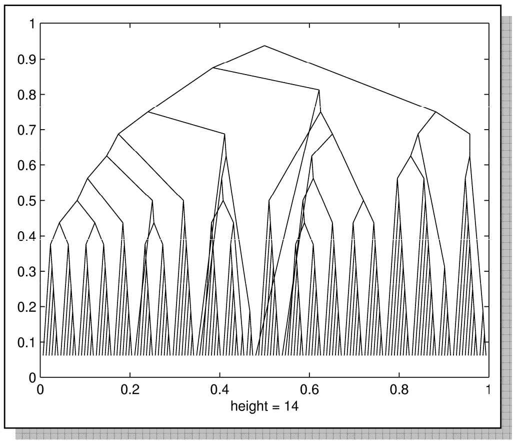
(Bemporad, Giorgetti, IEEE TAC 2006)

T	Bool. Vars	SATbB&B (s)	SATbB&B LPs	SATbB&B SATs	CPbB&B (s)	CPbB&B LPs	CPLEX (s)	CPLEX LPs	Naive MILP (s)	Naive MILP LPs
5	210	0.401	8	10	0.201	5	0.61	74	3.45	193
10	420	1.430	10	16	1.349	8	1.802	216	8.91	312
20	840	7.825	33	40	7.612	17	10.301	632	18.160	748
30	1260	11.510	78	98	13.67	104	23.081	692	114.53	1021
40	1680	79.318	264	304	104.67	579	125.930	934	813.23	1404
100	4200	207.840	2306	2409	253.273	3201	403.020	3657	$>$ 1200	–

hybrid MPC problem



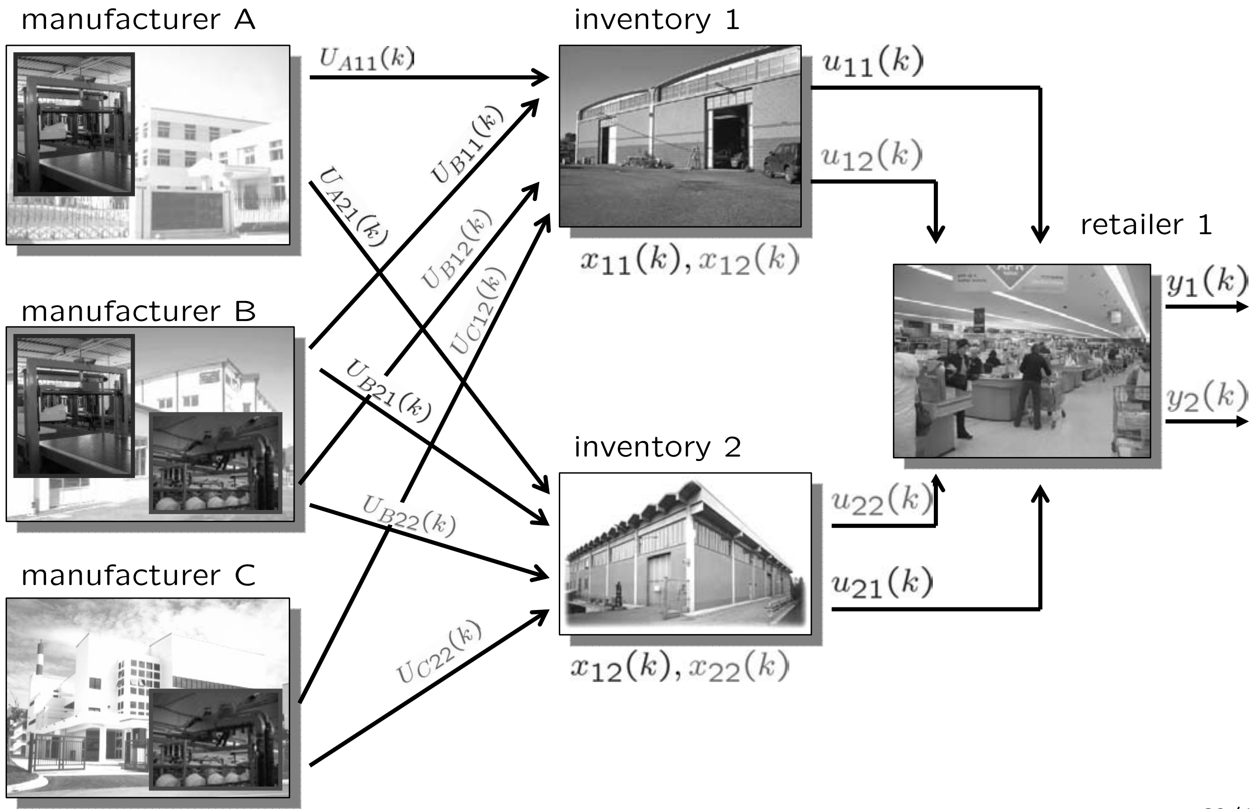
SAT-based algorithm



Naive MILP algorithm

Pentium IV 1.8GHz SAT solver: zCHAFF 2003.07.22

A Simple Example in Supply Chain Management



System Variables

- <u>continuous states:</u>

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

- <u>continuous outputs:</u>

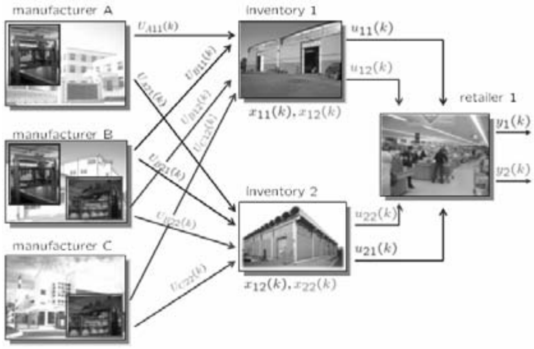
$y_j(k)$ = amount of j sold at time k
(i=1,2)

- <u>continuous inputs:</u>

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

- <u>binary inputs:</u>

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



Constraints

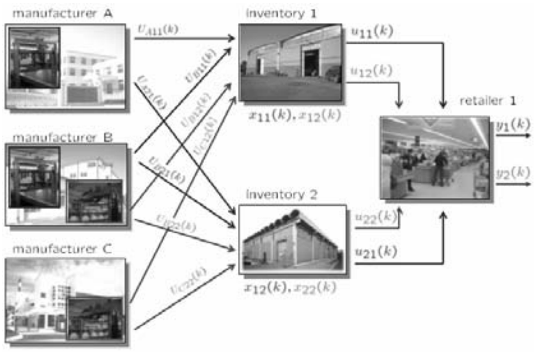
- Max capacity of inventory i:

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

<u>Numerical values</u>:
x_{M1}=10, x_{M2}=10

- Max transportation from inventories:

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



- A product can only be sent to one inventory:

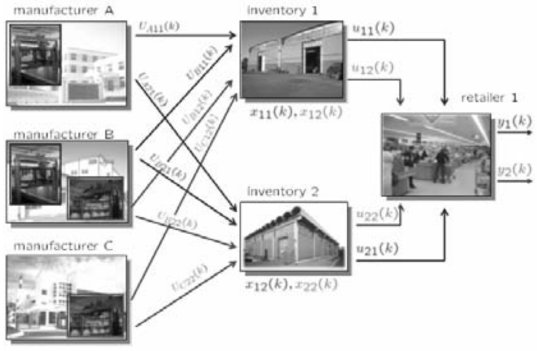
UA11(k) and UA21(k) cannot be =1 at the same time
UB11(k) and UB21(k) cannot be =1 at the same time
UB12(k) and UB22(k) cannot be =1 at the same time
UC12(k) and UC22(k) cannot be =1 at the same time

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

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

Dynamics

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



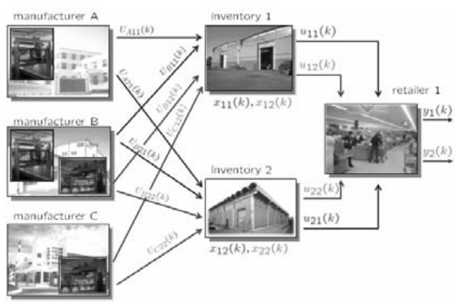
Numerical values:
P_{A1}=4, P_{B1}=6, P_{B2}=7, P_{C2}=3

- Level of inventories:

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

Hybrid Dynamical Model

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

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

        OUTPUT {REAL y1,y2;}

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

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


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



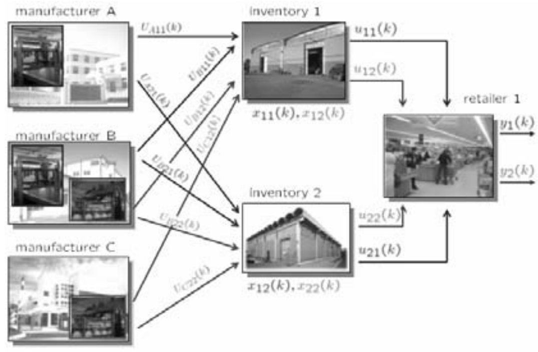

```
        CONTINUOUS {x11 = x11 + zA11 + zB11 - u11;
                    x12 = x12 + zB12 + zC12 - u12;
                    x21 = x21 + zA21 + zB21 - u21;
                    x22 = x22 + zB22 + zC22 - u22;  }

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

        MUST {  ~(UA11 & UA21);
            ~(UC12 & UC22);
            ~((UB11 | UB21) & (UB12 | UB22));

            ~(UB11 & UB21);
            ~(UB12 & UB22);
            x11+x12 <= xM1;
            x11+x12 >=0;
            x21+x22 <= xM2;
            x21+x22 >=0; }
} }
```

/demos/hybrid/supply_chain.m

Objectives

- Meet customer demand as much as possible:

  $y_1 \approx r_1,\ y_2 \approx r_2$



- Minimize transportation costs

- Fulfill all constraints

Performance Specs

penalty on demand tracking error

$$\min \sum_{k=0}^{N-1} 10\,(|y_1(k) - r_1(k)| + |y_2(k) - r_2(k)|) +$$
$$4\,(|u_{11}(k)| + |u_{12}(k)|) +$$
$$2\,(|u_{21}(k)| + |u_{22}(k)|) +$$
$$1\,(|U_{A11}(k)| + |U_{A21}(k)|) +$$
$$4\,(|U_{B11}(k)| + |U_{B12}(k)| + |U_{B21}(k)| + |U_{B22}(k)|) +$$
$$10\,(|U_{C12}(k)| + |U_{C22}(k)|)$$

cost for shipping from inv.#1 to market

cost for shipping from inv.#2 to market

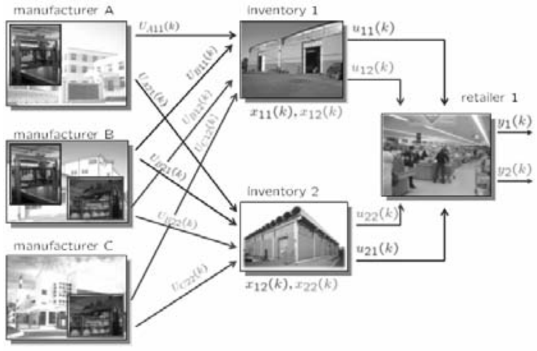
cost from A to inventories

cost from B to inventories

cost from C to inventories

Hybrid MPC - Example

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



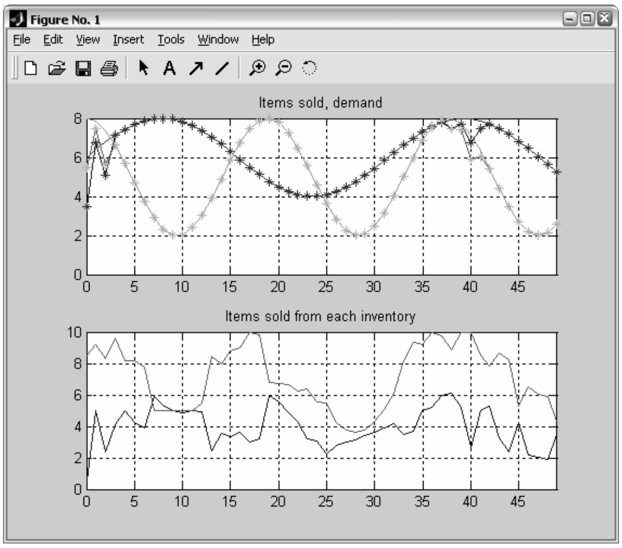

```
>>C=hybcon(S,Q,N,limits,refs);
```

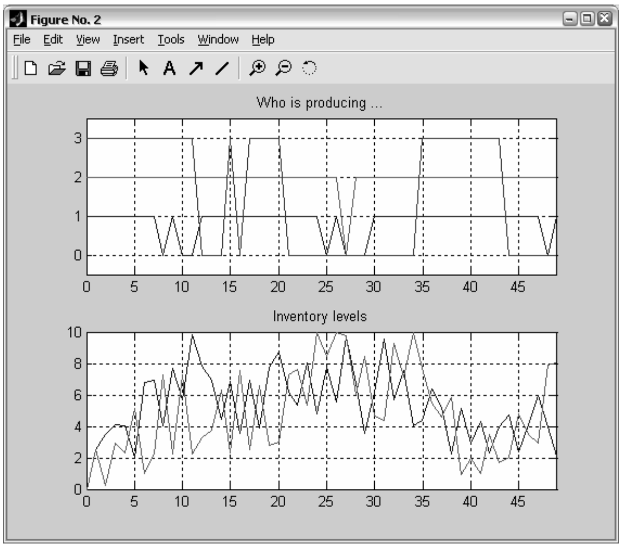
```
>> C

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

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

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

Type "struct(C)" for more details.

>>
```


Hybrid MPC - Example

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

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





CPU time: ≈ 30ms per time step (using GLPK on this machine)

A Few Hybrid MPC Tricks ...

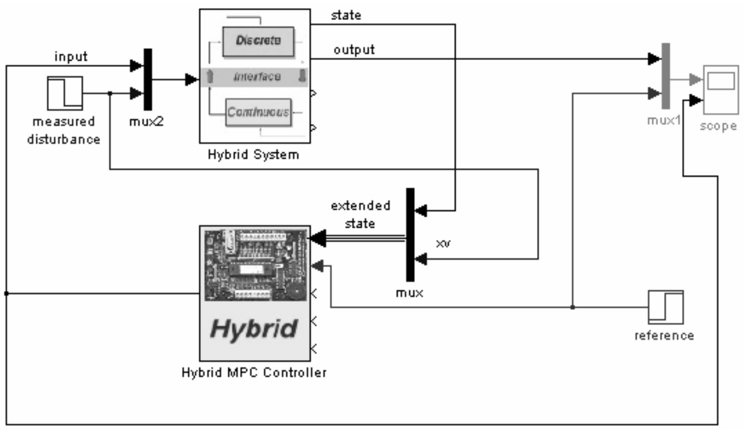
Measured Disturbances

- Disturbance $v(k)$ can be measured at time k
- Augment the hybrid prediction model with a constant state

$$x_v(k+1) = x_v(k)$$

- In Hysdel:

```
INTERFACE{
      STATE{
         REAL x     [-1e3, 1e3];
         REAL xv    [-1e3, 1e3];
      }
      ...
   }
IMPLEMENTATION{
      CONTINUOUS{
         x = A*x + B*u + Bv*xv
         xv= xv;
         ...
      }
}
```



/demos/hybrid/hyb_meas_dist.m

Note: same trick applies to linear MPC

Hybrid MPC - Tracking

- Optimal control problem (quadratic performance index):

$$\min_{\Delta U} \sum_{k=0}^{N-1} \|W^y(y(k+1)-r(t))\|^2 + \|W^{\Delta u}\Delta u(k)\|^2$$

$$[\Delta u(k) \triangleq u(k) - u(k-1)]$$

$$\text{subj. to } u_{\min} \le u(k) \le u_{\max}, \ k = 0,\dots,N-1$$

$$\Delta u_{\min} \le \Delta u(k) \le \Delta u_{\max}, \ k = 0,\dots,N-1$$

$$y_{\min} \le y(k) \le y_{\max}, \ k = 1,\dots,N$$

- Optimization problem: (MIQP)

$$\min_{\Delta U} J(\Delta U, x(t)) = \frac{1}{2}\Delta U' H \Delta U + [x'(t)\ r'(t)\ u'(t-1)]F\Delta U$$

$$\text{s.t. } G\Delta U \le W + K \begin{bmatrix} x(t) \\ r(t) \\ u(t-1) \end{bmatrix}$$

Note: same trick as in linear MPC

Integral Action in Hybrid MPC

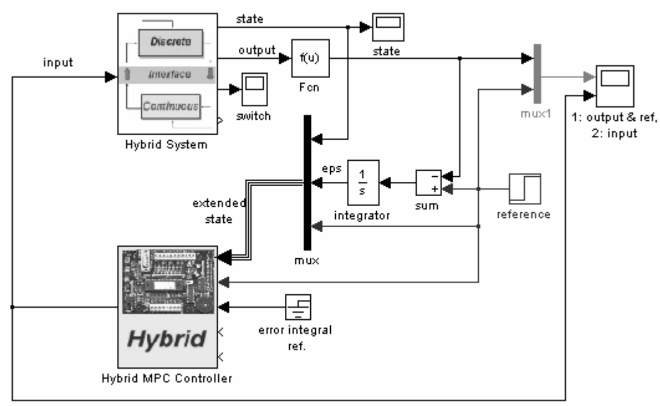
- Augment the hybrid prediction model with integrators of output errors as additional states:

$$\epsilon(k+1) = \epsilon(k) + T_s \cdot (r(k) - y(k))$$

$T_s =$ sampling time

- Treat $r(k)$ as a measured disturbance (=additional constant state)
- Add weight on $\varepsilon(k)$ in cost function to make $\varepsilon(k) \to 0$
- In Hysdel:

```
INTERFACE{
      STATE{
         REAL x          [-100,100];
         ...
         REAL epsilon    [-1e3, 1e3];
         REAL r          [0,    100]; }
      OUTPUT {
         REAL y; }
      ... }
IMPLEMENTATION{
      CONTINUOUS{
         epsilon=epsilon+Ts*(r-(c*x));
         r=r;
         ...   }
      OUTPUT{
         y=c*x; }   }
```



/demos/hybrid/hyb_integral_action.m

Note: same trick applies to linear MPC

Reference/Disturbance Preview

Measured disturbance $v(t)$ is known M steps in advance

$$\begin{cases} x_{v,M-1}(k+1) &= x_{v,M-2}(k) \\ x_{v,M-2}(k+1) &= x_{v,M-3}(k) \\ &\vdots \\ x_{v,1}(k+1) &= x_{v,0}(k) \\ x_{v,0}(k+1) &= x_{v,0}(k) \\ v(k) &= x_{v,M-1}(k),\ k=0,\dots,N-1 \end{cases}$$

initial condition

Note: same trick applies to linear MPC

$$\begin{cases} x_{v,M-1}(0) &= v(t) \\ x_{v,M-2}(0) &= v(t+1) \\ \vdots &= \vdots \\ x_{v,1}(0) &= v(t+M-2) \\ x_{v,0}(0) &= v(t+M-1) \end{cases}$$

produces v={$v(t)$,$v(t$+1),...,$v(t$+M-1), $v(t$+M-1),...}

Preview of reference $r(t)$: similar.

Delays – Method 1

- Hybrid model w/ delays:

$$\begin{aligned} x(t+1) &= Ax(t) + B_1u(t-\tau) + B_2\delta(t) + B_3z(t) \\ E_2\delta(t) &+ E_3z(t) \le E_1u(t-\tau) + E_4x(t) + E_5 \end{aligned}$$

- Map delays to poles in z=0:

$$x_k(t) \triangleq u(t-k) \quad \Rightarrow \quad x_k(t+1) = x_{k-1}(t) \qquad k=1,\dots,\tau$$

- Extend the state space of the MLD model:

$$\begin{bmatrix} x \\ x_\tau \\ x_{\tau-1} \\ \vdots \\ x_1 \end{bmatrix}(t+1) = \begin{bmatrix} A & B_1 & 0 & 0 & \dots & 0 \\ 0 & 0 & I_m & 0 & \dots & 0 \\ 0 & 0 & 0 & I_m & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \dots & 0 \end{bmatrix} \begin{bmatrix} x \\ x_\tau \\ x_{\tau-1} \\ \vdots \\ x_1 \end{bmatrix}(t) + \begin{bmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ I_m \end{bmatrix} u(t) + \begin{bmatrix} B_2 \\ 0 \\ 0 \\ \vdots \\ I_m \end{bmatrix} \delta(t) + \begin{bmatrix} B_3 \\ 0 \\ 0 \\ \vdots \\ I_m \end{bmatrix} z(t)$$

- Apply MPC to the extended MLD system

Note: same trick as in linear MPC

Delays – Method 2

- Delay-free MLD model:
$$\begin{aligned}\bar{x}(t+1) &= A\bar{x}(t) + B_1u(t) + B_2\bar{\delta}(t) + B_3\bar{z}(t)\\ E_2\bar{\delta}(t) &+ E_3\bar{z}(t) \leq E_1u(t) + E_4\bar{x}(t) + E_5\end{aligned}$$

  $\bar{x}(t) \triangleq x(t+\tau)$, $\bar{\delta}(t) \triangleq \delta(t+\tau)$, $\bar{z}(t) \triangleq z(t+\tau)$

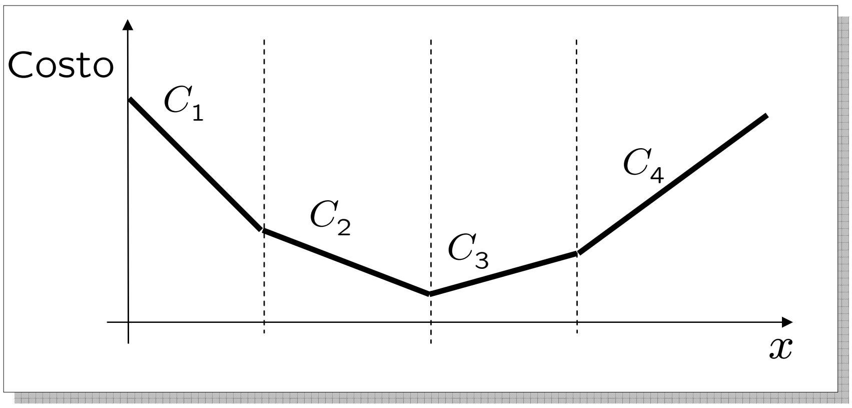
- Design MPC for delay-free model: $u(t) = f_{\text{MPC}}(\bar{x}(t))$

- Compute the predicted state:
$$\bar{x}(t) = A^\tau x(t) + \sum_{j=0}^{\tau-1} A^{\tau-1-j}\Big(B_1u(t-1-j) + B_2\bar{\delta}(t+j) + B_3\bar{z}(t+j)\Big)$$
  where $\bar{\delta}(t+j), \bar{z}(t+j)$ are obtained from MLD ineq. (or HYSDEL model)

- Compute MPC action:
$$u(t) = f_{\text{MPC}}(\bar{x}(t))$$

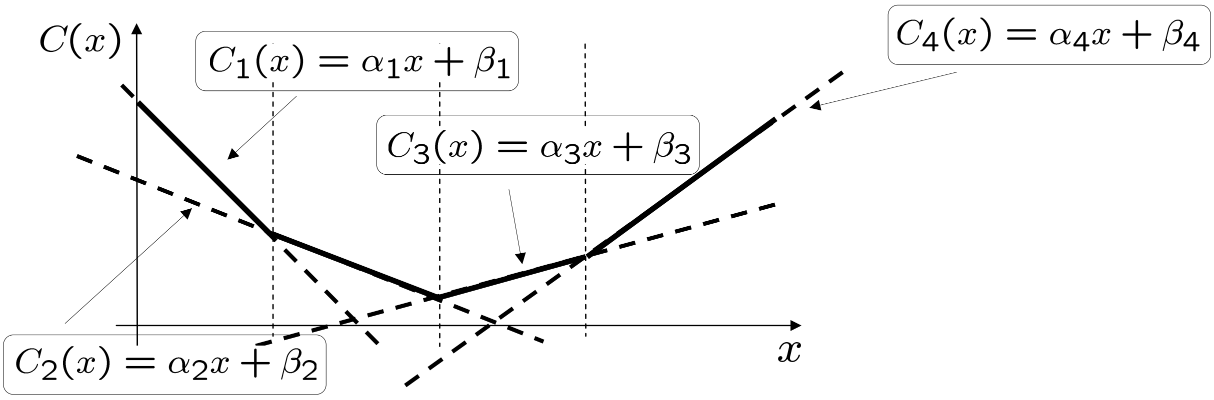
For better closed-loop performance the model used for predicting the future hybrid state $x(t+\tau)$ may be more accurate than MLD model !

Piecewise Affine Cost Functions



Convex piecewise affine cost functions can be represented **without** introducing binary variables

Piecewise Affine Cost Functions

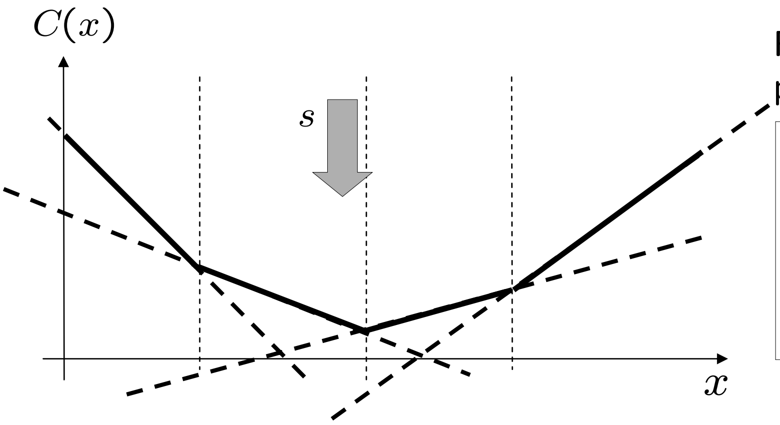


It is easy to see that:

$$C(x) = \max\{\alpha_1 x + \beta_1, \alpha_2 x + \beta_2, \alpha_3 x + \beta_3, \alpha_4 x + \beta_4\}$$

<u>In general</u>: Every convex piecewise affine function can be represented as the max of affine functions, and vice versa (Schechter, 1987)

Piecewise Affine Cost Functions



Reformulation as a linear program:

$$\begin{array}{ll} \min & s \\ \text{subj.to} & \left\{ \begin{array}{lcl} s & \geq & \alpha_1 x + \beta_1 \\ s & \geq & \alpha_2 x + \beta_2 \\ s & \geq & \alpha_3 x + \beta_3 \\ s & \geq & \alpha_4 x + \beta_4 \end{array} \right. \end{array}$$

$$s \geq \max\{\alpha_1 x + \beta_1, \alpha_2 x + \beta_2, \alpha_3 x + \beta_3, \alpha_4 x + \beta_4\}$$

Variabile s is an upper-bound on the max

It is easy to show (by contradiction) that at optimality we have:

$$s = \max\{\alpha_1 x + \beta_1, \alpha_2 x + \beta_2, \alpha_3 x + \beta_3, \alpha_4 x + \beta_4\}$$

Piecewise affine convex constraints can be dealt with similarly

General Remarks About MIP Modeling

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

Henceforth, when creating a hybrid model one has to

Be thrifty with integer variables !

Adding logical constraints usually helps ...

Generally speaking:

Modeling is art

(a unifying general theory does not exist)

Explicit Hybrid MPC

On-Line vs Off-Line Optimization

$$\min_U J(U, x(t)) = \sum_{k=0}^{T-1} \|Rx(t+k|t)\|_p + \|Qu(t+k)\|_p$$

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

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

Mixed-Integer Linear/Quadratic Program (MILP/MIQP)

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

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

- Off-line optimization: solve the MILP/MIQP <u>for all</u> $x(t)$

$$\min_\zeta J(\zeta, x(t)) = \begin{cases} f'\zeta & \infty\text{-norm} \\ \zeta' H \zeta + f'\zeta & \text{2-norm} \end{cases}$$

$$\text{s.t. } G\zeta \leq W + Fx(t)$$

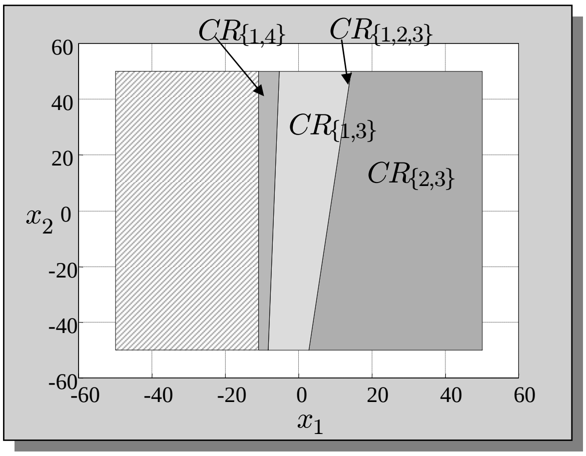
multi-parametric programming

Example of Multiparametric Solution

Multiparametric LP

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

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



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

MPC of Linear Systems

Linear model $\begin{cases} x_{k+1} = Ax_k + Bu_k \\ y_k = Cx_k \end{cases}$ $\quad x_0 = x(t)$

Quadratic performance index $\min_U J(x(t), U) = \sum_{k=0}^{N-1} \left[x_k' Q x_k + u_k' R u_k\right] + x_N' P x_N$

$$\begin{array}{ll} \min_U & \frac{1}{2}U'HU + x'(t)F'U + \frac{1}{2}x(t)Yx(t) \\ \text{subj. to} & GU \leq W + Sx(t), \end{array} \qquad U = \begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ u_{N-1} \end{bmatrix}$$

Constraints $\begin{cases} u_{\min} \leq u_k \leq u_{\max} \\ y_{\min} \leq y_k \leq y_{\max} \end{cases}$

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

(Bemporad et al., 2002)

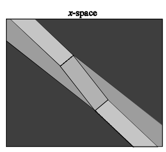
Properties of multiparametric-QP

(Bemporad et al., 2002)

optimizer	$U^*(x) = \arg\min_U \frac{1}{2}U'HU + x'F'U$ subj. to $GU \leq W + Sx$	continuous, piecewise affine
value function	$V^*(x) = \frac{1}{2}x'Yx + \min_U \frac{1}{2}U'HU + x'F'U$ subj. to $GU \leq W + Sx$	convex continuous, piecewise quadratic, C^1 (if no degeneracy)
feasible state set	$X^* = \{x : \exists U \text{ such that } Gx \leq W + SU\}$	convex polyhedral

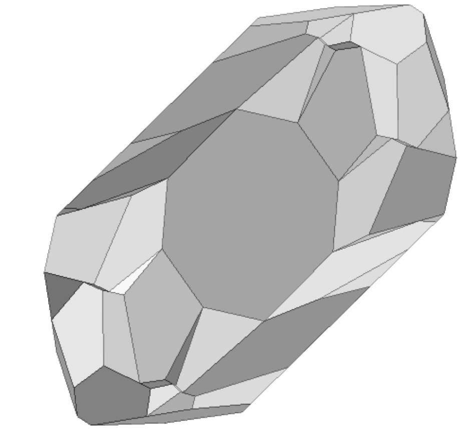
Corollary: The linear MPC controller is a continuous piecewise affine function of the state

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



Set of Feasible Parameters X^*

Why is the set X^* of parameters for which the QP problem is solvable a **convex polyhedral** set ?



$$X^* = \{x : \exists U \text{ such that } GU \leq W + Sx\}$$

X^* is the projection of a polyhedron onto the parameter space. Therefore X^* is a polyhedron.

$$X^* = \text{Proj}_x \left\{ \begin{bmatrix} U \\ x \end{bmatrix} : \begin{bmatrix} G & -S \end{bmatrix} \begin{bmatrix} U \\ x \end{bmatrix} \leq W \right\}$$

Double Integrator Example

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

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

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

- Control objective: minimize $\sum_{t=0}^{\infty} y^2(t) + \frac{1}{100}u^2(t)$

$$u(t+k) = K_{LQ}x(t+k|t), \ \forall k \geq N_u$$

- Optimization problem: for N_u=2

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

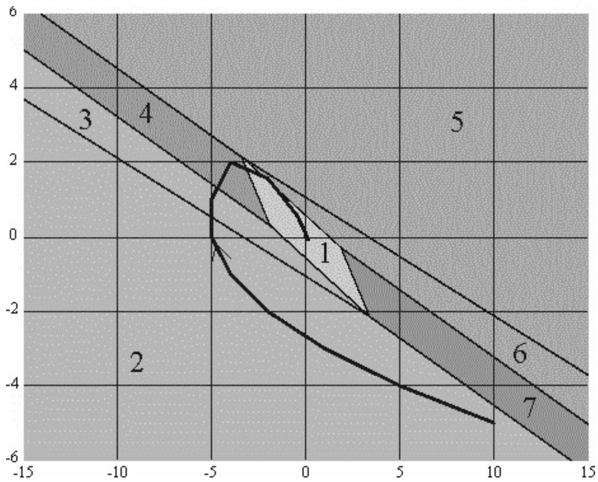
←costfunctionis normalizedby maxsvd←H←←

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

mp-QP solution

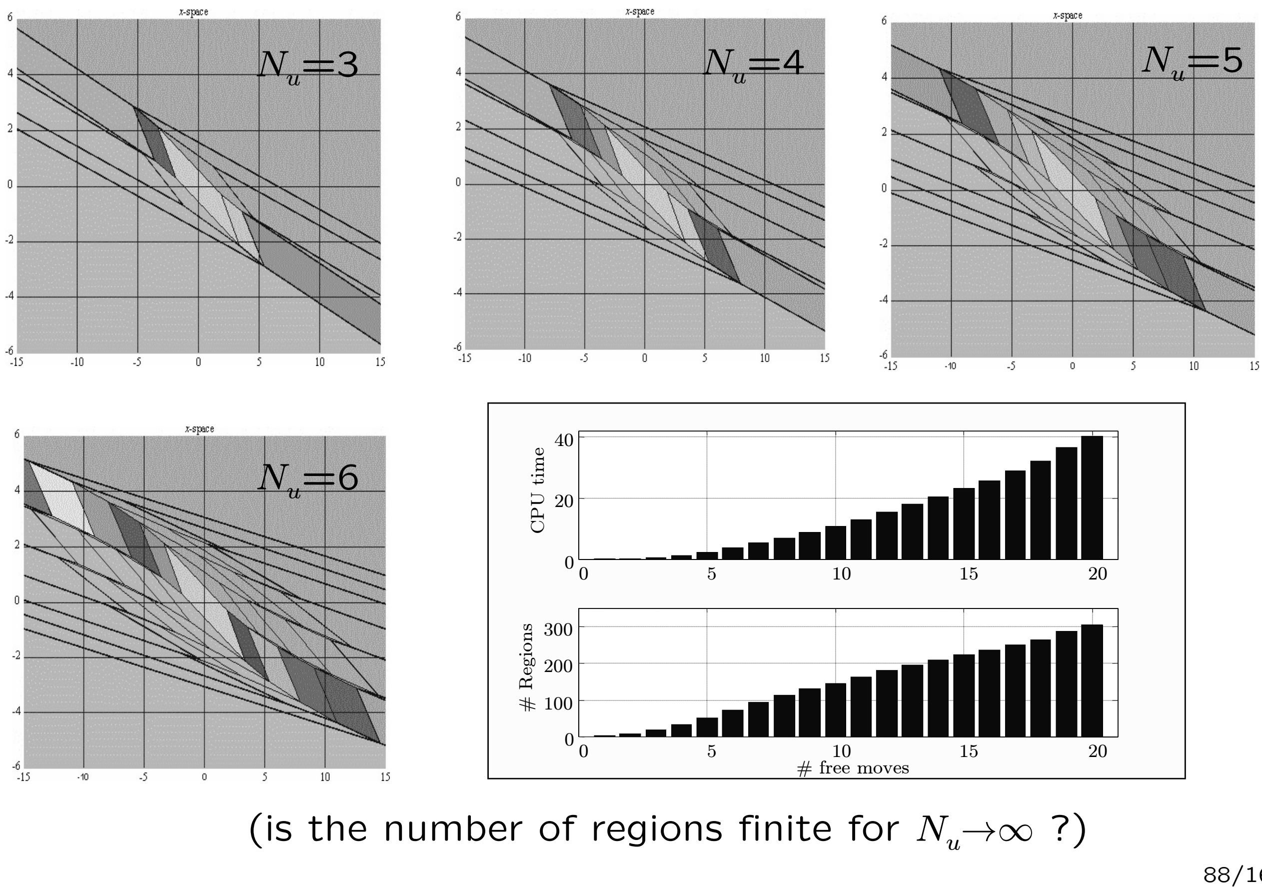
N_u=2

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



go to demo **/demos/linear/doubleintexp.m** (Hyb-Tbx)

Complexity



(is the number of regions finite for $N_u \to \infty$?)

Complexity

- Worst-case complexity analysis:

$$M \triangleq \sum_{\ell=0}^{q} \binom{q}{\ell} = 2^q \quad \text{combinations of active constraints}$$

- Usually the number of regions is much smaller, as many combinations of active constraints are never feasible and optimal at any parameter vector x
- Strongest dependence on the number q of constraints
- Strong dependence on the number N_u of free moves
- Weak dependence on the number n of parameters x
- Example:

states\horizon	$N=1$	$N=2$	$N=3$	$N=4$	$N=5$
n=2	3	6.7	13.5	21.4	19.3
n=3	3	6.9	17	37.3	77
n=4	3	7	21.65	56	114.2
n=5	3	7	22	61.5	132.7
n=6	3	7	23.1	71.2	196.3
n=7	3	6.95	23.2	71.4	182.3
n=8	3	7	23	70.2	207.9

Data averaged over 20 randomly generated single-input single-output systems subject to input saturation (q=2N)

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: $u(t) = f(x(t))$

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

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

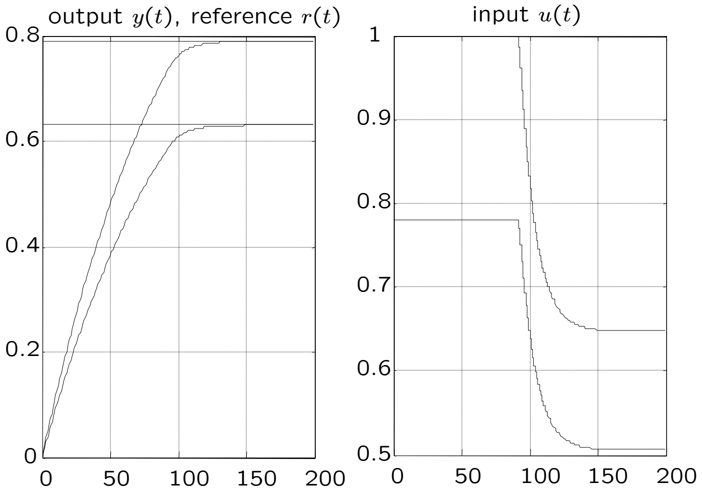
  $$\begin{aligned} u_{\min}(t) &\leq u(k) \leq u_{\max}(t) \\ y_{\min}(t) &\leq y(k) \leq y_{\max}(t) \end{aligned}$$

- Other models (hybrid, uncertain) and other norms ($\|\cdot\|_1$, $\|\cdot\|_\infty$)

Reference Tracking, MIMO System

- System: $y(t) = \frac{10}{100s+1}\begin{bmatrix} 4 & -5 \\ -3 & 4 \end{bmatrix} u(t)$ sampling + ZOH (T_s=1 s)
- Constraints: $-1 \leq u_1, u_2 \leq 1$
- Control objective: $\min \sum_{k=0}^{19} \|y(t+k|t) - r(t)\|^2 + \frac{1}{10}\|\delta u(t+k)\|^2$

$$u(t+k) = u(t),\ \forall k \geq 1$$



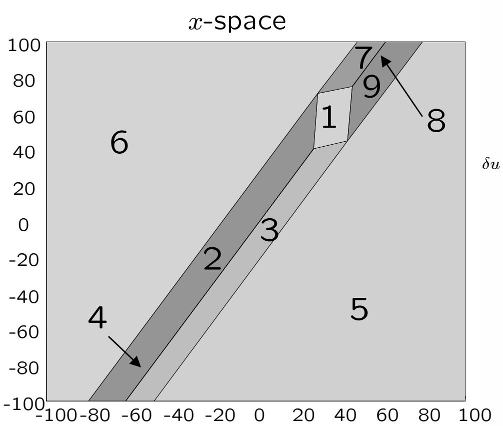
N=20
N_u=1

go to demo
linear/mimo.m
(Hyb-Tbx)

Reference Tracking, MIMO System

MPC law

State-space partition



Polyhedral partition of the state-space for u=[0 0]' and r=[0.63 0.79]'

$$\delta u = \begin{cases}
\begin{bmatrix} -0.1251 & 0.0084 \\ 0.0168 & -0.0668 \end{bmatrix} x + \begin{bmatrix} -0.8535 & 0.1143 \\ 0.1143 & -0.9107 \end{bmatrix} u + \begin{bmatrix} 2.5583 & 3.1741 \\ 1.8949 & 2.5583 \end{bmatrix} r & \text{if } \begin{bmatrix} -0.1251 & 0.0084 \\ 0.0168 & -0.0668 \\ 0.1251 & -0.0084 \\ -0.0168 & 0.0668 \end{bmatrix} x + \begin{bmatrix} 0.1465 & 0.1143 \\ 0.1143 & 0.0893 \\ -0.1465 & -0.1143 \\ -0.1143 & -0.0893 \end{bmatrix} u + \begin{bmatrix} 2.5583 & 3.1741 \\ 1.8949 & 2.5583 \\ -2.5583 & -3.1741 \\ -1.8949 & -2.5583 \end{bmatrix} r \leq \begin{bmatrix} 1.0000 \\ 1.0000 \\ 1.0000 \\ 1.0000 \end{bmatrix} \quad \text{(Region \#1)} \\
\begin{bmatrix} 0.0000 & 0.0000 \\ 0.1144 & -0.0733 \end{bmatrix} x + \begin{bmatrix} -1.0000 & 0.0000 \\ -0.0000 & -0.9999 \end{bmatrix} u + \begin{bmatrix} 0.0000 & 0.0000 \\ -0.1016 & 0.0813 \end{bmatrix} r + \begin{bmatrix} 1.0000 \\ 0.7804 \end{bmatrix} & \text{if } \begin{bmatrix} 0.5210 & -0.3337 \\ -0.0643 & 0.0412 \\ 0.1251 & -0.0084 \end{bmatrix} x + \begin{bmatrix} -0.0000 & 0.0005 \\ 0.0000 & -0.0001 \\ -0.1465 & -0.1143 \end{bmatrix} u + \begin{bmatrix} -0.4625 & 0.3700 \\ 0.0570 & -0.0456 \\ -2.5583 & -3.1741 \end{bmatrix} r \leq \begin{bmatrix} 1.0000 \\ 1.0000 \\ -1.0000 \end{bmatrix} \quad \text{(Region \#2)} \\
\begin{bmatrix} -0.1466 & 0.0938 \\ 0.0000 & 0.0000 \end{bmatrix} x + \begin{bmatrix} -0.9998 & -0.0000 \\ 0.0000 & -1.0000 \end{bmatrix} u + \begin{bmatrix} 0.1333 & -0.0999 \\ -0.0000 & 0.0000 \end{bmatrix} r + \begin{bmatrix} 1.2798 \\ 1.0000 \end{bmatrix} & \text{if } \begin{bmatrix} -0.5239 & 0.3353 \\ 0.0643 & -0.0411 \\ -0.0168 & 0.0668 \end{bmatrix} x + \begin{bmatrix} 0.0007 & -0.0000 \\ -0.0001 & 0.0000 \\ -0.1143 & -0.0893 \end{bmatrix} u + \begin{bmatrix} 0.4763 & -0.3572 \\ -0.0584 & 0.0438 \\ -1.8949 & -2.5583 \end{bmatrix} r \leq \begin{bmatrix} -1.0000 \\ 1.0000 \\ -1.0000 \end{bmatrix} \quad \text{(Region \#3)} \\
\begin{bmatrix} -1.0000 & 0.0000 \\ 0.0000 & -1.0000 \end{bmatrix} u + \begin{bmatrix} 1.0000 \\ 1.0000 \end{bmatrix} & \text{if } \begin{bmatrix} 0.5239 & -0.3353 \\ -0.5210 & 0.3337 \end{bmatrix} x + \begin{bmatrix} -0.0007 & 0.0000 \\ 0.0000 & -0.0005 \end{bmatrix} u + \begin{bmatrix} -0.4763 & 0.3572 \\ 0.4625 & -0.3700 \end{bmatrix} r \leq \begin{bmatrix} 1.0000 \\ -1.0000 \end{bmatrix} \quad \text{(Region \#4)} \\
\begin{bmatrix} -1.0000 & 0.0000 \\ -0.0000 & -1.0000 \end{bmatrix} u + \begin{bmatrix} -1.0000 \\ 1.0000 \end{bmatrix} & \text{if } \begin{bmatrix} -0.0643 & 0.0412 \\ -0.0643 & 0.0411 \end{bmatrix} x + \begin{bmatrix} 0.0000 & -0.0001 \\ 0.0001 & -0.0000 \end{bmatrix} u + \begin{bmatrix} 0.0570 & -0.0456 \\ 0.0584 & -0.0438 \end{bmatrix} r \leq \begin{bmatrix} -1.0000 \\ -1.0000 \end{bmatrix} \quad \text{(Region \#5)} \\
\begin{bmatrix} -1.0000 & 0.0000 \\ 0.0000 & -1.0000 \end{bmatrix} u + \begin{bmatrix} 1.0000 \\ -1.0000 \end{bmatrix} & \text{if } \begin{bmatrix} 0.0643 & -0.0411 \\ 0.0643 & -0.0412 \end{bmatrix} x + \begin{bmatrix} -0.0001 & 0.0000 \\ -0.0000 & 0.0001 \end{bmatrix} u + \begin{bmatrix} -0.0584 & 0.0438 \\ -0.0570 & 0.0456 \end{bmatrix} r \leq \begin{bmatrix} -1.0000 \\ -1.0000 \end{bmatrix} \quad \text{(Region \#6)} \\
\begin{bmatrix} 0.0000 & 0.0000 \\ 0.1144 & -0.0733 \end{bmatrix} x + \begin{bmatrix} -1.0000 & 0.0000 \\ -0.0000 & -0.9999 \end{bmatrix} u + \begin{bmatrix} 0.0000 & 0.0000 \\ -0.1016 & 0.0813 \end{bmatrix} r + \begin{bmatrix} -1.0000 \\ -0.7804 \end{bmatrix} & \text{if } \begin{bmatrix} 0.0643 & -0.0412 \\ -0.5210 & 0.3337 \\ -0.1251 & 0.0084 \end{bmatrix} x + \begin{bmatrix} -0.0000 & 0.0001 \\ 0.0000 & -0.0005 \\ 0.1465 & 0.1143 \end{bmatrix} u + \begin{bmatrix} -0.0570 & 0.0456 \\ 0.4625 & -0.3700 \\ 2.5583 & 3.1741 \end{bmatrix} r \leq \begin{bmatrix} 1.0000 \\ 1.0000 \\ -1.0000 \end{bmatrix} \quad \text{(Region \#7)} \\
\begin{bmatrix} -0.1466 & 0.0938 \\ 0.0000 & 0.0000 \end{bmatrix} x + \begin{bmatrix} -0.9998 & -0.0000 \\ 0.0000 & -1.0000 \end{bmatrix} u + \begin{bmatrix} 0.1333 & -0.0999 \\ -0.0000 & 0.0000 \end{bmatrix} r + \begin{bmatrix} -1.2798 \\ -1.0000 \end{bmatrix} & \text{if } \begin{bmatrix} -0.0643 & 0.0411 \\ 0.5239 & -0.3353 \\ 0.0168 & -0.0668 \end{bmatrix} x + \begin{bmatrix} 0.0001 & -0.0000 \\ -0.0007 & 0.0000 \\ 0.1143 & 0.0893 \end{bmatrix} u + \begin{bmatrix} 0.0584 & -0.0438 \\ -0.4763 & 0.3572 \\ 1.8949 & 2.5583 \end{bmatrix} r \leq \begin{bmatrix} 1.0000 \\ -1.0000 \\ -1.0000 \end{bmatrix} \quad \text{(Region \#8)} \\
\begin{bmatrix} -1.0000 & 0.0000 \\ 0.0000 & -1.0000 \end{bmatrix} u + \begin{bmatrix} -1.0000 \\ -1.0000 \end{bmatrix} & \text{if } \begin{bmatrix} -0.5239 & 0.3353 \\ 0.5210 & -0.3337 \end{bmatrix} x + \begin{bmatrix} 0.0007 & -0.0000 \\ -0.0000 & 0.0005 \end{bmatrix} u + \begin{bmatrix} 0.4763 & -0.3572 \\ -0.4625 & 0.3700 \end{bmatrix} r \leq \begin{bmatrix} 1.0000 \\ -1.0000 \end{bmatrix} \quad \text{(Region \#9)}
\end{cases}$$

MPC Regulation of a Ball on a Plate

Explicit Hybrid MPC (MLD)

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

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

- On-line optimization: solve the problem *for each* given $x(t)$

  Mixed-Integer Linear Program (MILP)

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

$$\begin{aligned} \min_{\xi} \quad & \sum_{k=0}^{T-1} \epsilon_i^x + \epsilon_i^u \\ \text{s.t.} \quad & G\xi \leq W + S \begin{bmatrix} x(t) \\ r(t) \end{bmatrix} \end{aligned}$$

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

Multiparametric MILP

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

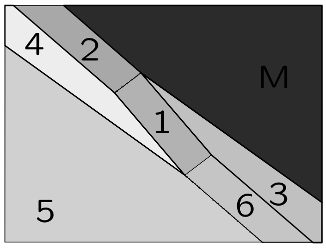
- mp-MILP can be solved (by alternating MILPs and mp-LPs)

(Dua, Pistikopoulos, 1999)

- **Theorem:** The multiparametric solution $\xi^*(x)$ is piecewise affine

- **The MPC controller is piecewise affine in x,r**

$$u(x,r) = \begin{cases} F_1 x + E_1 r + g_1 & \text{if} \quad H_1 \left[\begin{smallmatrix} x \\ r \end{smallmatrix}\right] \leq K_1 \\ \vdots & \vdots \\ F_M x + E_M r + g_M & \text{if} \quad H_M \left[\begin{smallmatrix} x \\ r \end{smallmatrix}\right] \leq K_M \end{cases}$$



Explicit Hybrid MPC (PWA)

$$\begin{array}{l} \min_U J(U,x,r) = \sum_{k=0}^{T-1} \|R(y(k)-r)\|_p + \|Qu(k)\|_p \\ \text{subject to} \begin{cases} \text{PWA model} \\ x(0) = x \end{cases} \end{array}$$

$p = 1, 2, \infty$

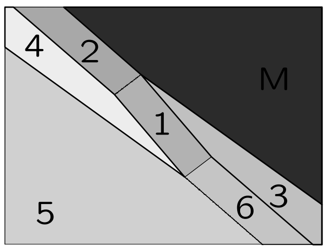
$\|v\|_2 = v'v$

$\|v\|_\infty = \max |v_i|$

$\|v\|_1 = \sum v_i$

- **The MPC controller is piecewise affine in x,r**

$$u(x,r) = \begin{cases} F_1 x + E_1 r + g_1 & \text{if} \quad H_1 \left[\begin{smallmatrix} x \\ r \end{smallmatrix}\right] \leq K_1 \\ \vdots & \vdots \\ F_M x + E_M r + g_M & \text{if} \quad H_M \left[\begin{smallmatrix} x \\ r \end{smallmatrix}\right] \leq K_M \end{cases}$$



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

Computation of Explicit Hybrid MPC (PWA)

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

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

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

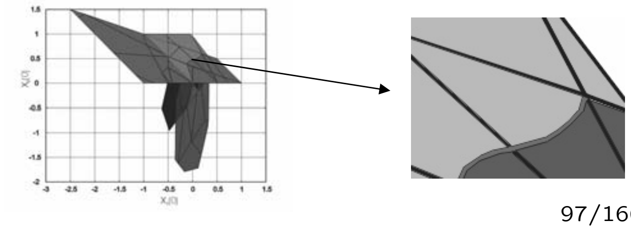
1 - Use backwards (=DP) reachability analysis for enumerating all feasible mode sequences $I=\{i(0),i(1),\ldots,i(T\text{-}1)\}$;

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

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

Quadratic case: keep overlapping regions (possibly eliminate overlaps that are never optimal) and compare on-line (if needed).

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



Hybrid Control Examples (Revisited)

Hybrid Control Example

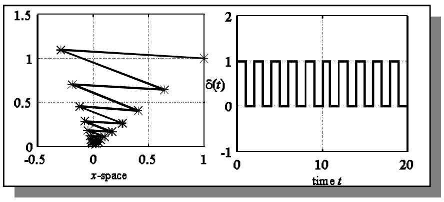
PWA system:

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

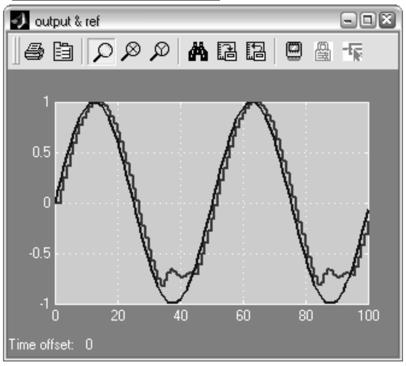
Constraints: $-1 \leq u(t) \leq 1$

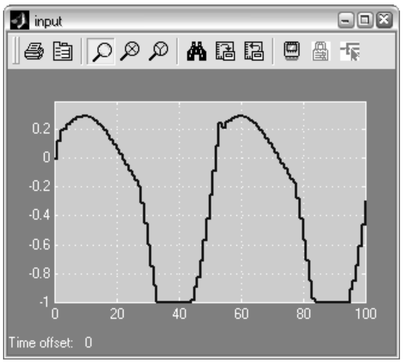
Objective: $\min \sum_{k=1}^{2} |y(t+k|t)-r(t)|$

Open loop behavior:



Closed loop:

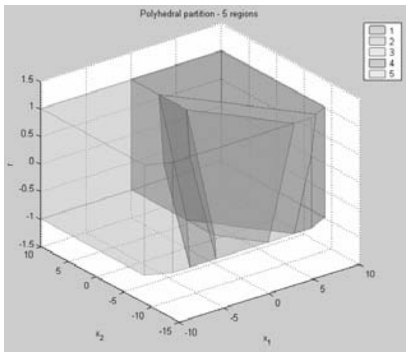




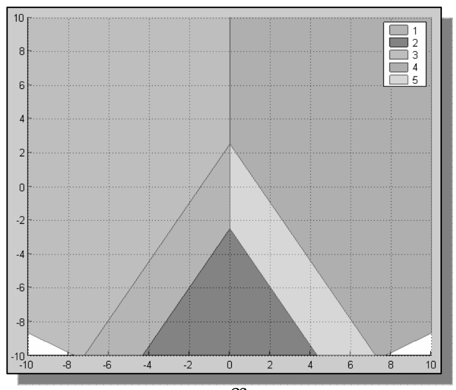
HybTbx: **/demos/hybrid/bm99sim.m**

Explicit PWA Controller

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



Section with r=0



x_2

x_1

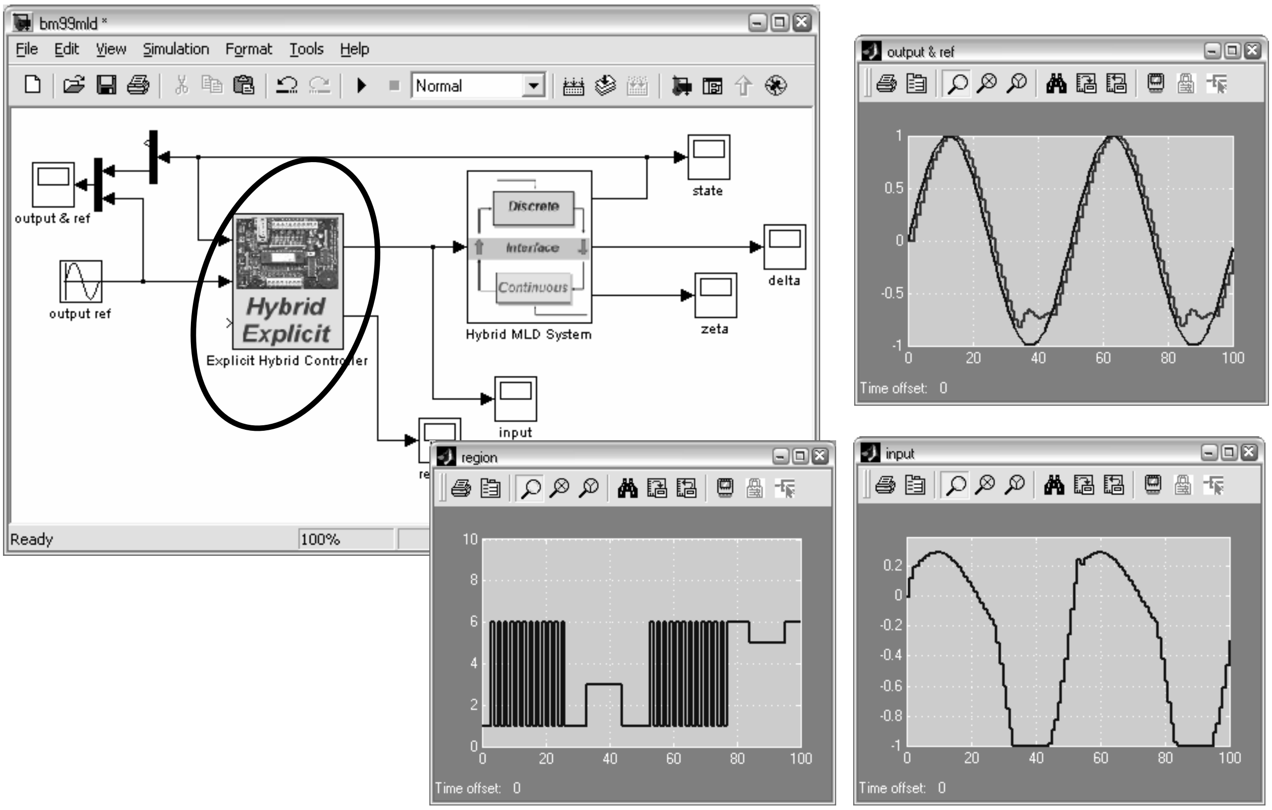
HybTbx: **/demos/hybrid/bm99sim.m**

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

PWA law ≡ MPC law !

Hybrid Control Example

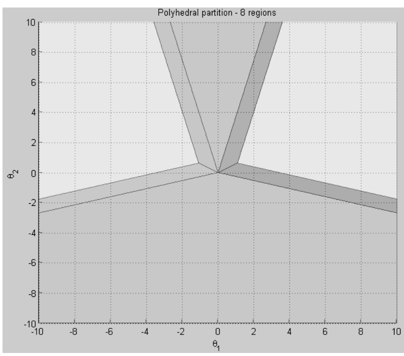
Closed loop:



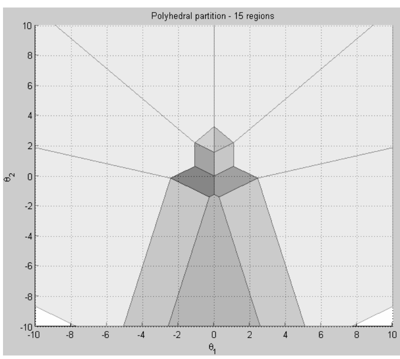
Explicit PWA Regulator

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

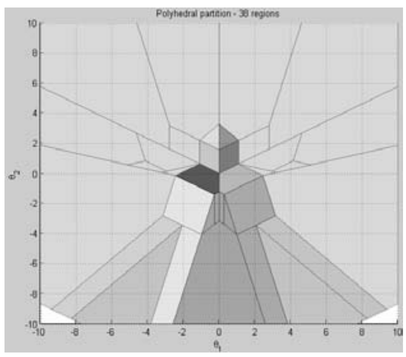
Prediction horizon N=1



Prediction horizon N=2



Prediction horizon N=3

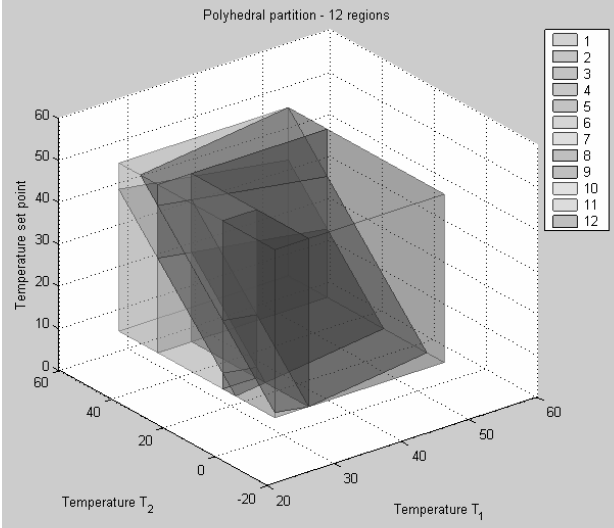


HybTbx: **/demos/hybrid/bm99benchmark.m**

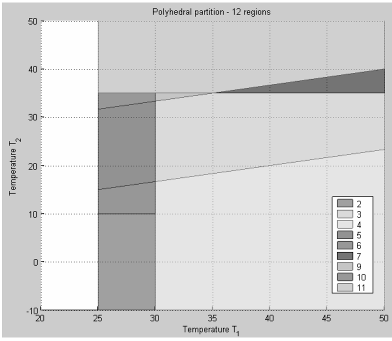
Explicit MPC – Temperature Control

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

```
>> E

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

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

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

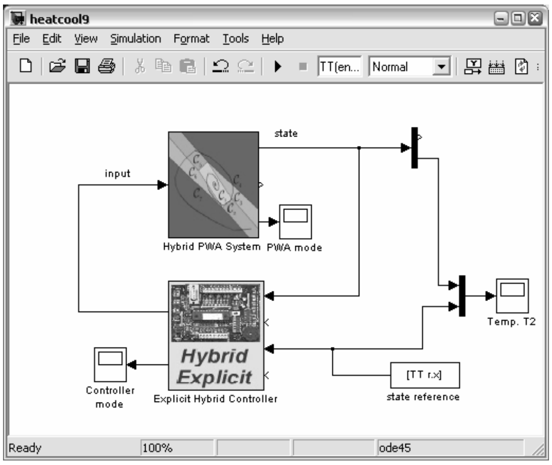


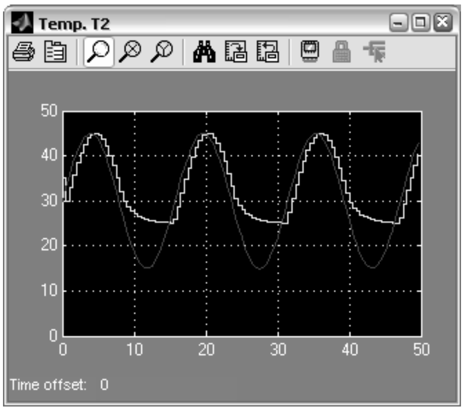
$$
\begin{aligned}
\min \quad & \sum_{k=1}^{2} x_2^2(k) \\
\text{s.t.} \quad & x_1(k) \geq 25 \;\; k = 1,2 \\
& \text{PWA model}
\end{aligned}
$$



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

Explicit MPC – Temperature Control

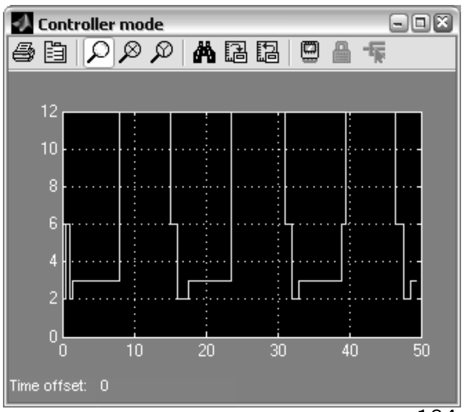




Generated C-code

utils/expcon.h

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

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

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



Implementation Aspects of Hybrid MPC

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

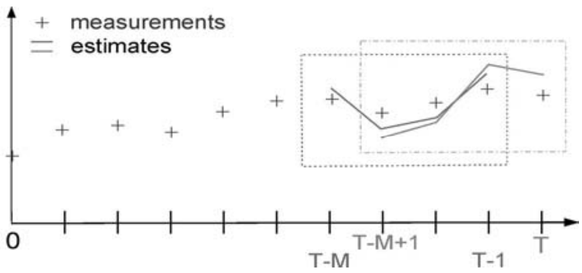
- Small problems (short horizon N=1,2, one or two inputs): explicit PWA control law preferable
  - time to evaluate the control law is shorter than MIP
  - control code is simpler (no complex solver must be included in the control software !)
  - more insight in controller's behavior

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

Moving Horizon Estimation
Fault Detection & Isolation

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

At each time t solve the problem:

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

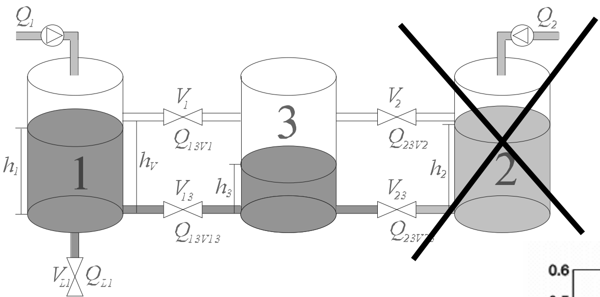
and get estimate $\hat{x}(t)$

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

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

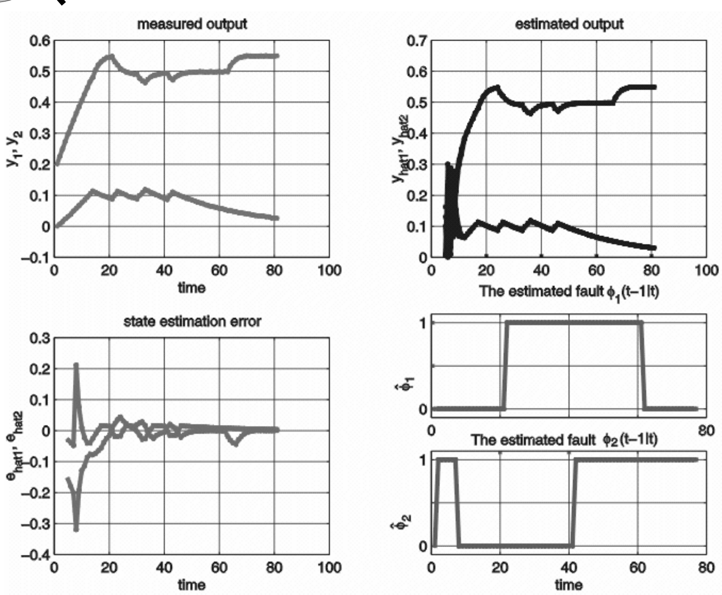
Fault detection: augment MLD with unknown **binary** distubances $\phi \in \{0,1\}^{n_f}$

Example: Three Tank System

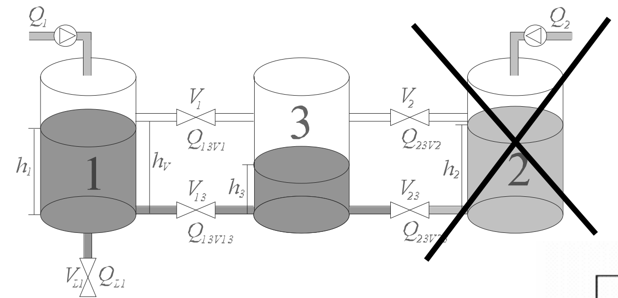


COSY Benchmark problem, ESF

- ϕ_1 : leak in tank 1 for $20\text{s} \leq t \leq 60\text{s}$
- ϕ_2 : valve V_1 blocked for $t \geq 40\text{s}$



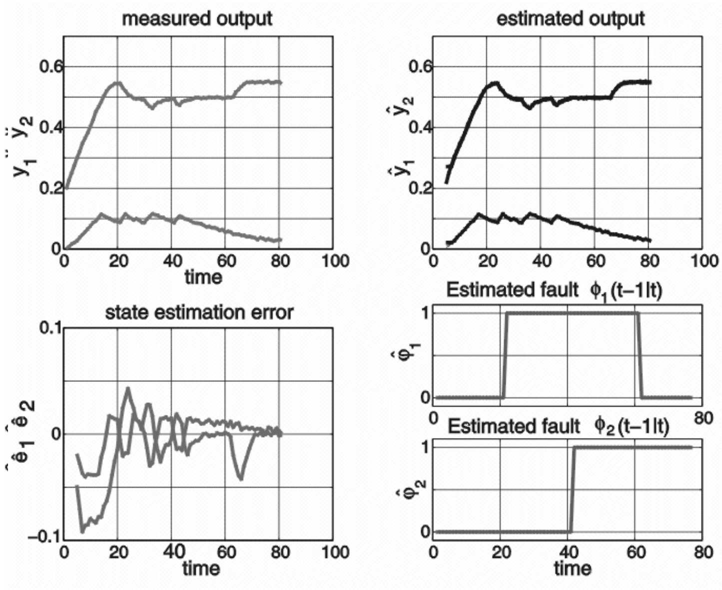
Example: Three Tank System



COSY Benchmark problem, ESF

- ϕ_1 : leak in tank 1 for $20\text{s} \le t \le 60\text{s}$
- ϕ_2 : valve V_1 blocked for $t \ge 40\text{s}$
- Add logic constraint

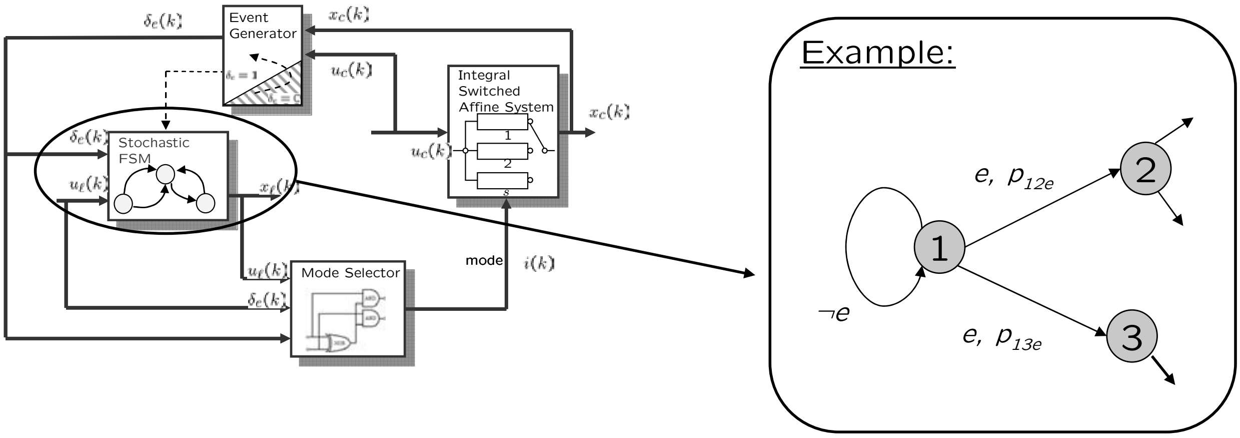
$$[h_1 \le h_v] \Rightarrow \phi_2 = 0$$



Stochastic and event-based hybrid MPC

Discrete Hybrid Stochastic Automaton

(Bemporad, Di Cairano, HSCC-05)

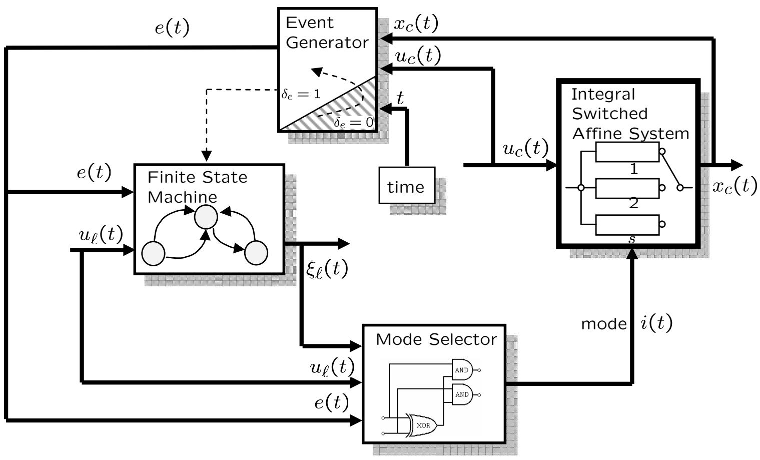


$$P[x_b(k+1)=1] = f_{\text{sFSM}}(x_b(k), u_b(k), \delta_e(k))$$

Stochastic Finite State Machine

- Maximize **performance** (best tracking scenario) and **robustness** (=likelihood of scenario to happen)
- Possibly include **chance constraints** (e.g. constraint must be fulfilled with certain probability)

Event-based Hybrid Systems (Continuous-time)



Switched integral dynamics

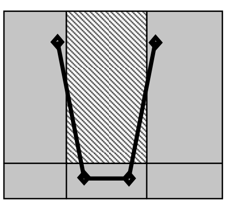
$$\frac{dx_c(t)}{dt} = B_{i(t)}u_c(t) + f_{i(t)}$$

$$i(t) \in \{1, \dots, s\}$$

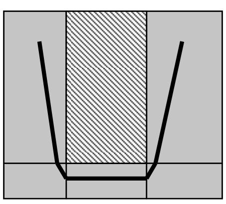
$$x_c(t) \in \mathbb{R}^{n_c},\ u_c(t) \in \mathbb{R}^{m_c}$$

Simple, but widely used !

Constraint fulfillment:



discrete-time



discrete-event

Restriction: input $u_c(t)$ is constant between two consecutive events

DHA - Extensions

- Discrete stochastic hybrid automata (DHSA) can be transformed into an equivalent MLD form

- Event-based continuous-time integral hybrid automata (icHA) can be transformed into an equivalent MLD form

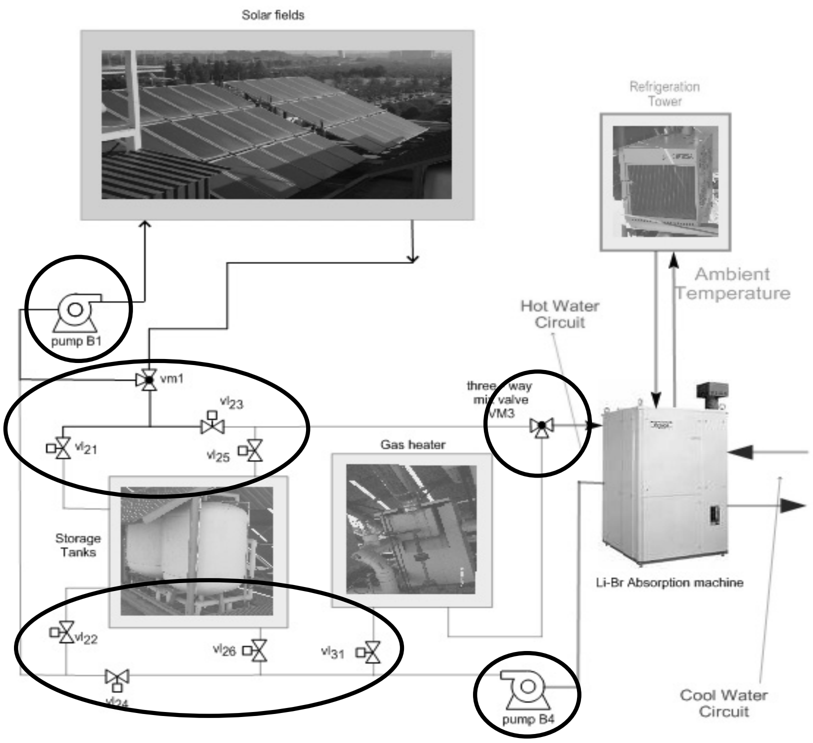
All MPC techniques developed for DHA can be extended to DHSA and icHA !

Application Examples

Hybrid Control of a Solar Air Conditioning Plant

(Menchinelli, Bemporad, 2007)

Solar Air Conditioning Plant



10 binary inputs:

- 7 electro-valves
- 1 on/off pump
- 2 (valve position vm1, 0 or 100%)

2 continuous inputs:

- pump velocity
- three-way mix valve position

Plant located at the Automatic Control Dept., University of Sevilla, Spain

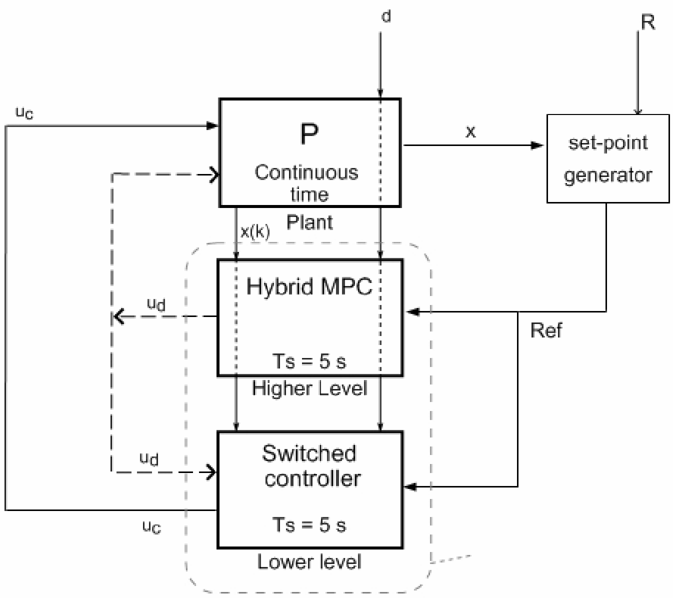
Control Objectives

1. Track set-point on temperature of cooled water
2. Minimize gas consumption (=use of gas heater)
3. Maximize heat stored in tanks

Complexity: the system has 9 possible operating modes, depending on choice of binary valve/pump values

Control architecture

Two-level control strategy:



Lower Level:

- continuous inputs
- different control laws for different operating modes

Higher Level:

- decide operating modes
- use hybrid MPC

(hybrid control of full dynamical model possible, but too complex)

Hybrid MPC Supervisor

- Must choose the best mode at each time interval, based on measurements of process states and of disturbances

- Hybrid MPC solution: use static model

  $\Rightarrow$ set prediction horizon N=1

  - piecewise-affine cost function
  - mode-dependent linear constraints
  - penalties on mode transitions (to avoid chattering)

Cost Function

$$C = \sum_{i \in \mathcal{I}} C_i \quad \mathcal{I} \text{ is the set of operative modes}$$

$$C_i = \begin{cases} W_i & \text{if } L_i = 1 \\ 0 & \text{otherwise} \end{cases}$$

$$W_i = \begin{cases} P_i - \Delta E_s & \text{if} \quad i \in \{M_1, M_2, M_3, M_4, M_8, M_9\} \\ P_i & \text{if} \quad i \in \{M6, M7\} \end{cases}$$

- L_i = logic variable associated to the operative mode M_i
- W_i penalizes the choice of mode M_i
- ΔE_s is a function of the current solar irradiation
- P_i = fixed cost paid for using mode M_i, depends on incentive for using solar irradiation

HYSDEL Formulation

The cost function is easily formulated in HYSDEL

overall cost

```
LINEAR { C = c1 + c2 + ... + c9; }

DA {  c1 = {IF l1 THEN p1 - a1*Ig ELSE 0}; }

LOGIC { l1 = !vl21 & !vl23 & !vl25 & !vl31 & !B4
             & B1 & !vm1; }
```

cost of mode M_1

operating mode l_1 is
a Boolean function
of logic inputs

Constraints

- Mode selection is unambiguous: $\sum_i l_i = 1$

- Exclude operating modes that are currently inadmissible.

Example: avoid using tanks to feed the chiller when their temperature is below the minimum value

```
AD { BOOL usetanks = Ttam >= 75 }

MUST { ! ( l7 & !usetanks) }
```

Hybrid MPC Formulation

Hysdel model has been converted into MLD form:

$$\begin{array}{l} x(t+1) = x(t) \\ E_2\delta(t) + E_3 z(t) \leq E_4 x(t) + E_1 u(t) + E_5 \end{array}$$

(static model)

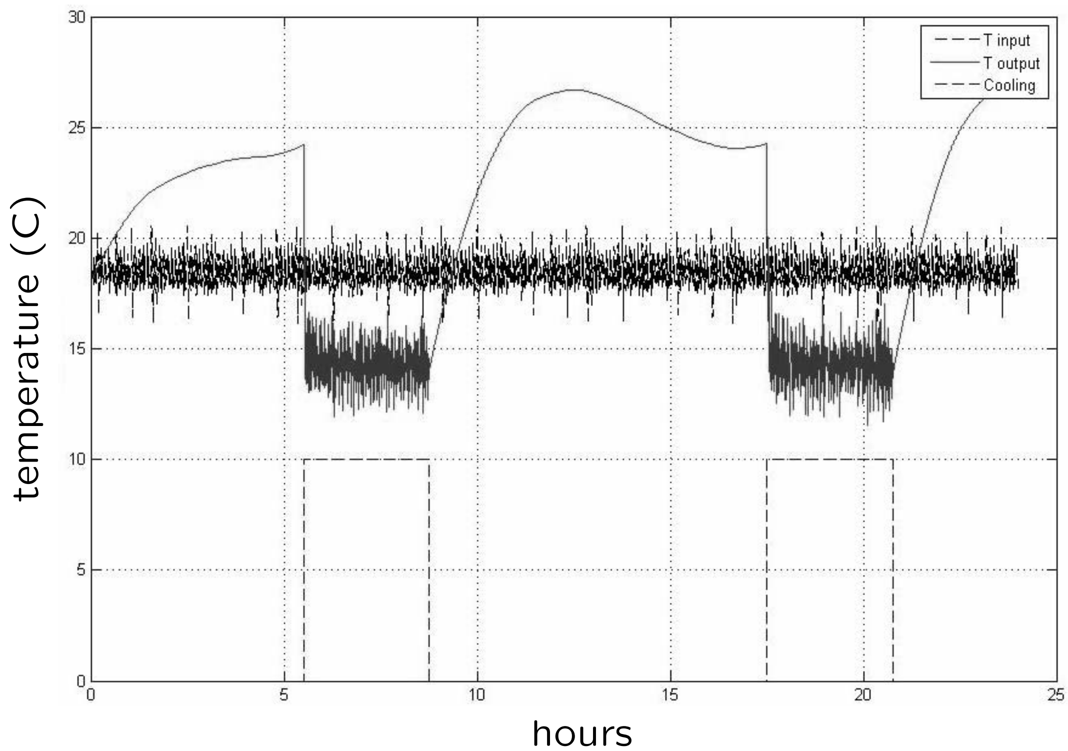
- Using the Hybrid Toolbox for Matlab, the MLD model is used to compute the hybrid MPC controller

- Controller complexity:

  (MIQP optimization)

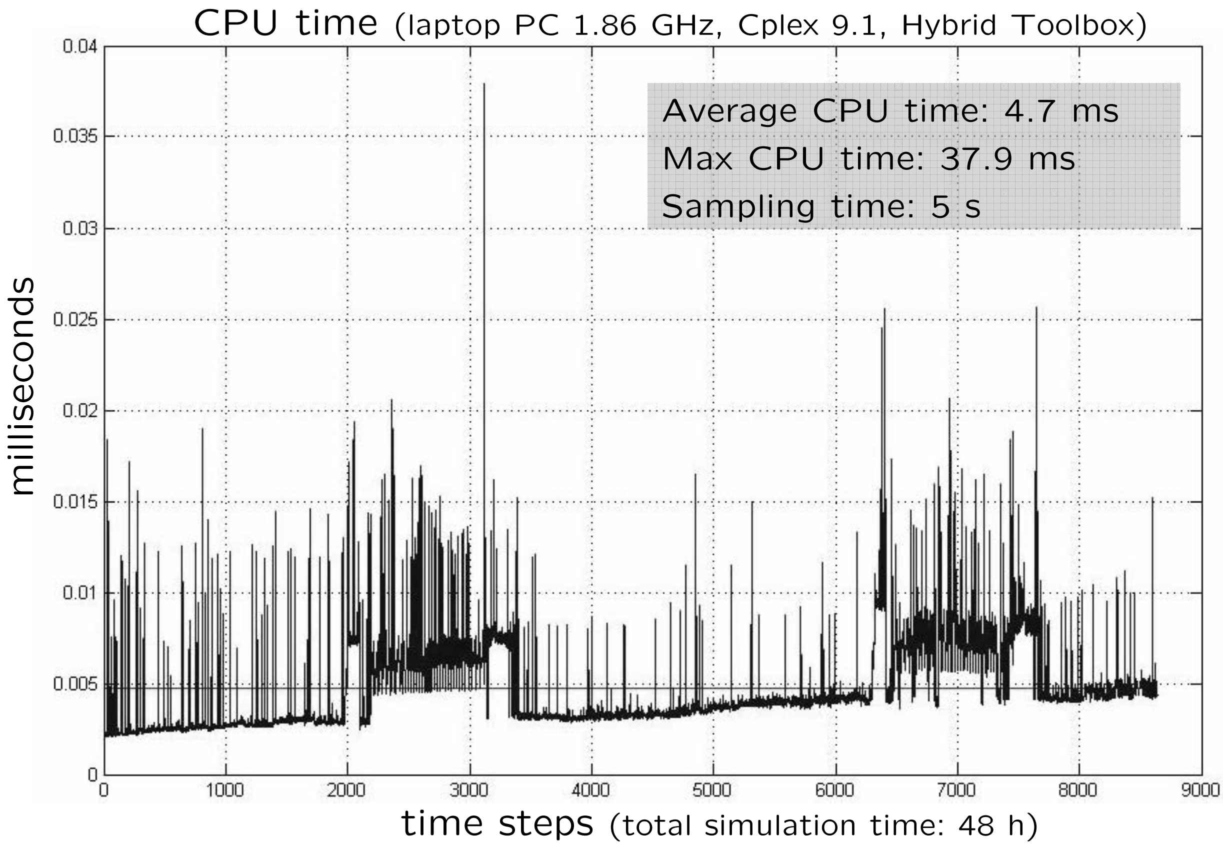
# continuous variables	18
# binary variables	32
# linear constraints	184

Simulation Results



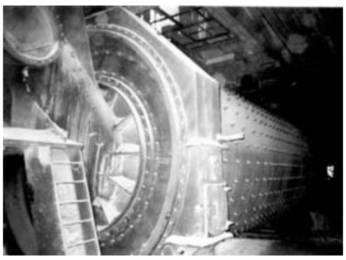
(next experimental campaign scheduled end of May, 2007)

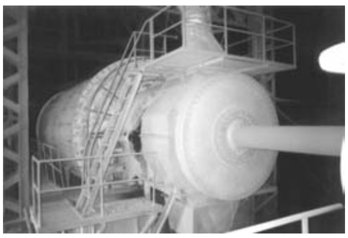
Computational burden

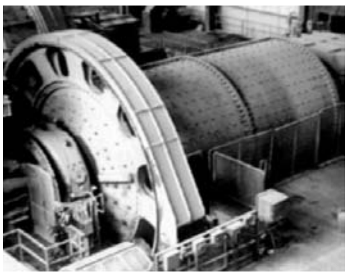


Cement Mills Scheduling

(a joint work with D. Castagnoli, E. Gallestey)







Goal: Decide when to produce a certain grade and on which mill

- **Complexity**
  - Several mills, several cement grades, several silos and conveyor belts
  - High power consumption, variable tariffs, production and transportation constraints
- **Customer needs**
  - Lowest possible energy bill
  - Less low grade cement
  - Strict contractual constraints satisfaction

Problem Description

Problem data:

- A single hybrid model for the milling and dispatching parts;
- 2 Mills, 3 Grades, 2 Belts, 3 Silos
- Sampling time: 1 hour
- Prediction horizon: 3 days (=24x3 time slots)

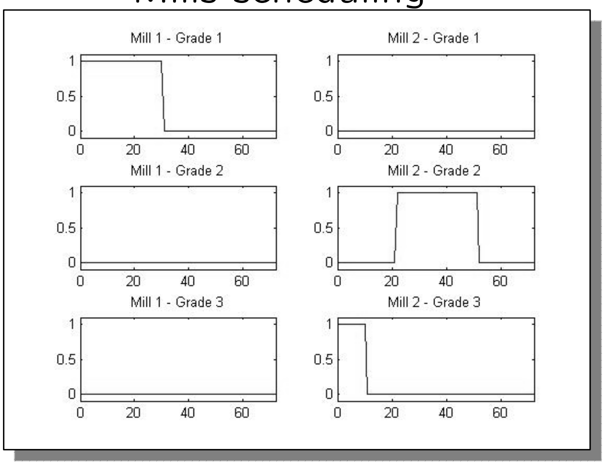
Resulting optimization problem:

- 3075 constraints
- 1161 variables (648 binary)

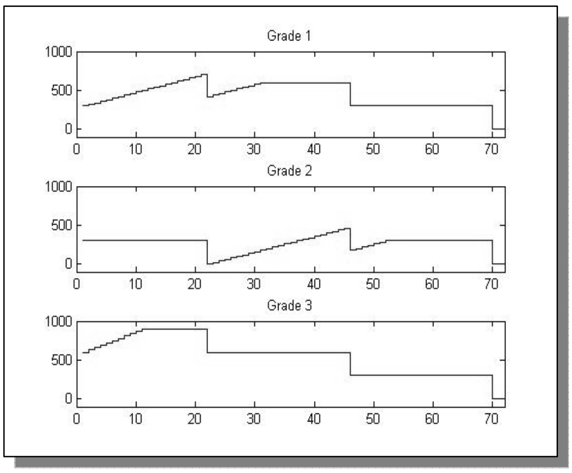
MILP solved in 15.71 s

Simulation Results

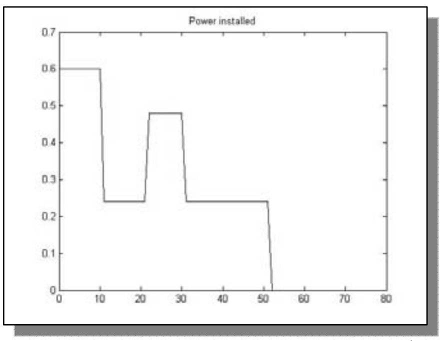
Mills scheduling



Silo levels



Power installed



Data: courtesy of ABB Ltd.

Hybrid MPC of Barcelona's Sewer Network

(Ocampo-Martinez, Bemporad, Ingimundarson, Puig, 2007)

Heavy Rain Consequences in Barcelona

Barcelona's Sewer Network

Problem Features

- Nonlinear dynamics
- Delays between subsystems
- Continuous and discrete dynamics
- Operating ranges and/or operational constraints
- Stochastic disturbances (rain, leaks)

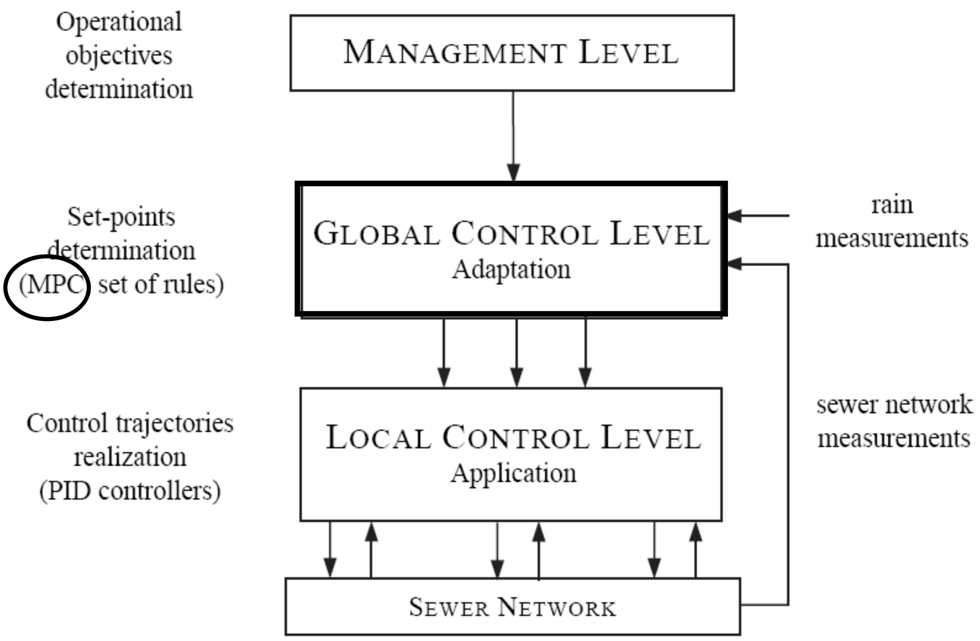





Main objectives:

- **Minimize flooding**
- **Minimize pollution**
- **Minimize sewage accumulation**

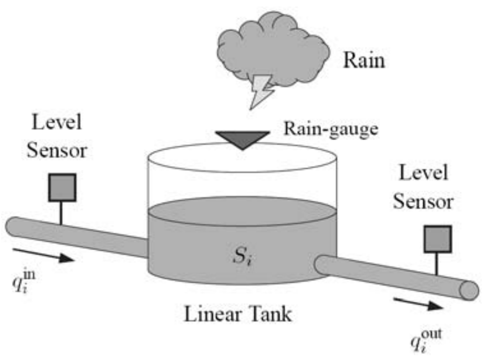
Hierarchical Control Structure

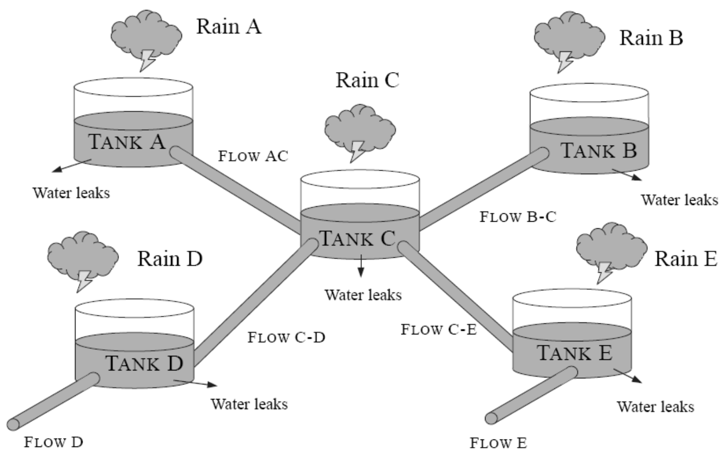


Modeling Principles

(Ocampo-Martinez, 2007)

The *Virtual Tank* approach:

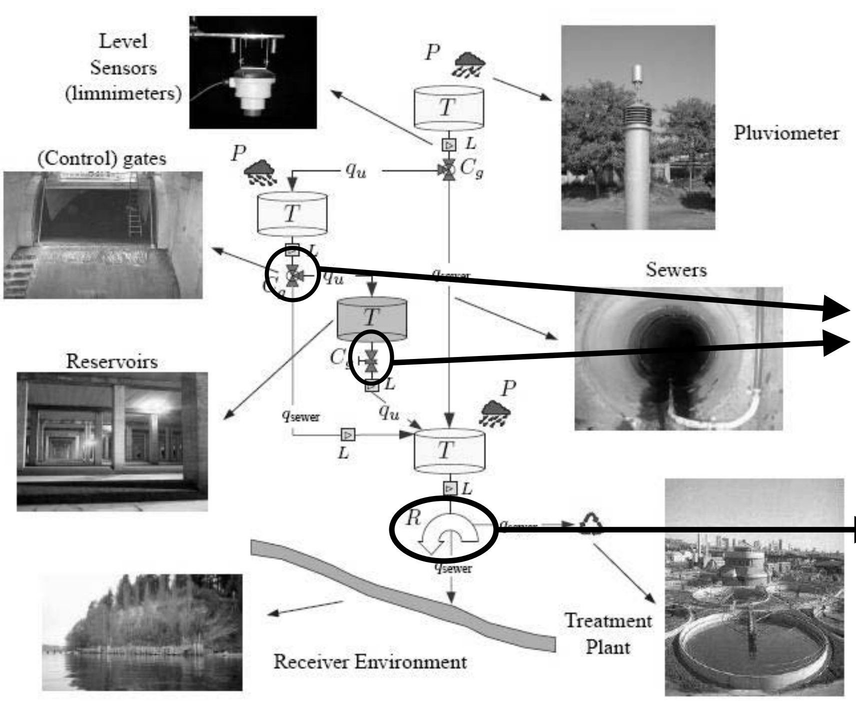


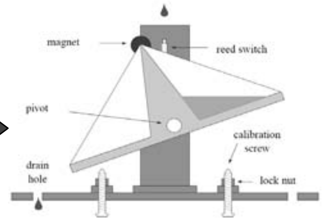


$$v_{ik+1} = v_{ik} + \Delta t \varphi_i S_i P_{ik} + \Delta t \left(q_{ik}^{\text{in}} - q_{ik}^{\text{out}}\right)$$
$$q_{ik}^{\text{out}} = \beta_i v_{ik}$$

$\phi =$ ground absorption coefficient
$S =$ tank area
$P =$ precipitation intensity
$\Delta t =$ sampling time
$\beta =$ volume-flow conversion factor

Constitutive Elements





Gates { Redirection, Retention }

Weir (type of node)

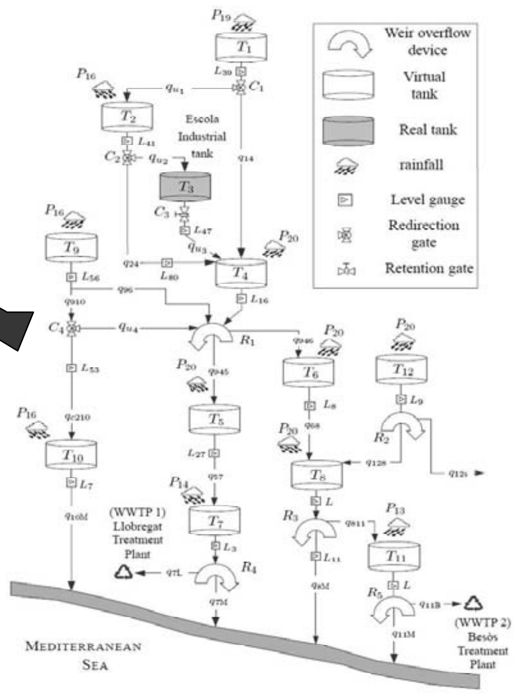
$$q_{in} = \sum_{i=0}^{n} q_i$$

$$q_{out} = \min\{q_{in}, \overline{q}_{out}\}$$

$$q_{over} = \max\{0, q_{in} - \overline{q}_{out}\}$$

Case Study

Barcelona Test Catchment (BTC)



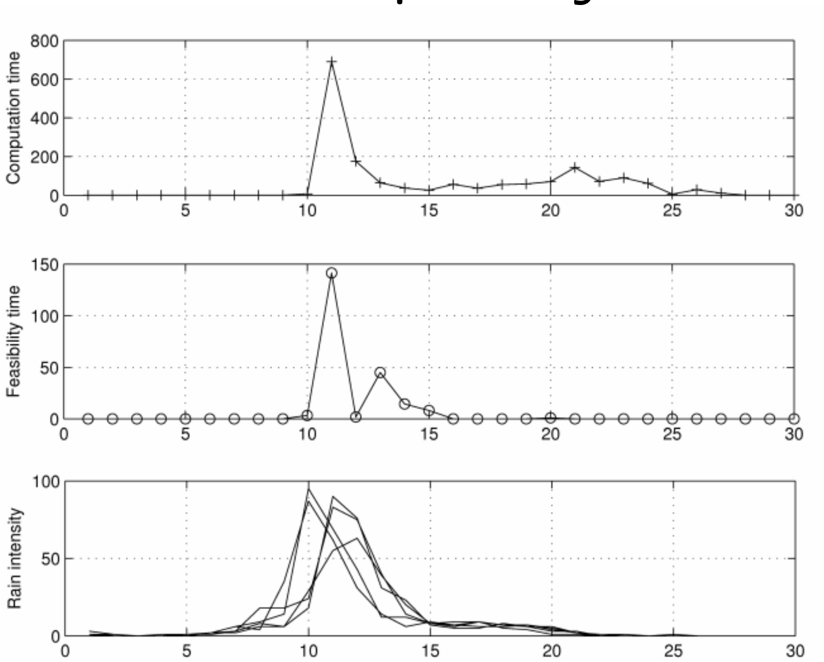
- ✓ 22.6 Km^2
- ✓ 11 sub-catchments
- ✓ 4 Control gates
- ✓ 1 real detention tank
- ✓ 5 rain gauges
- ✓ 2 WWTP

Simulation Results

Rain Episodes	Open Loop Flooding ×10^3 (m^3)	Open Loop Pollution ×10^3 (m^3)	Open Loop Treated W. ×10^3 (m^3)	Closed Loop Flooding ×10^3 (m^3)	Closed Loop Pollution ×10^3 (m^3)	Closed Loop Treated W. ×10^3 (m^3)
14-09-1999	108	225.8	278.4	92.9 (14%)	223.5	280.7
09-10-2002	116.1	409.8	533.8	97.1 (16%)	398.8	544.9
03-09-1999	1	42.3	234.3	0 (100%)	44.3	232.3
31-07-2002	160.3	378	324.4	139.7 (13%)	374.6	327.8
17-10-1999	0	65.1	288.4	0	58.1 (11%)	295.3
28-09-2000	1	104.5	285.3	1	98 (6%)	291.9
25-09-1998	0	4.8	399.3	0	4.8	398.8
22-09-2001	0	25.5	192.3	0	25	192.4
01-08-2002	0	1.2	285.8	0	1.2	285.8
20-04-2001	0	35.4	239.5	0	32.3 (9%)	242.5

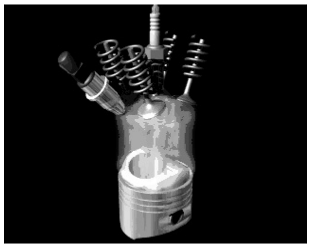
Hybrid MPC setup: Sampling time=5min, prediction horizon=30min

Complexity



time steps (Δt=5 min)

- Worst-case CPU time on standard PC ≈ 700 s > sampling time = 300 s (CPLEX 9.1 + Hybrid Toolbox for Matlab)
- Suboptimal approach possible (or just use a better computer !)



Hybrid Control of a DISC Engine

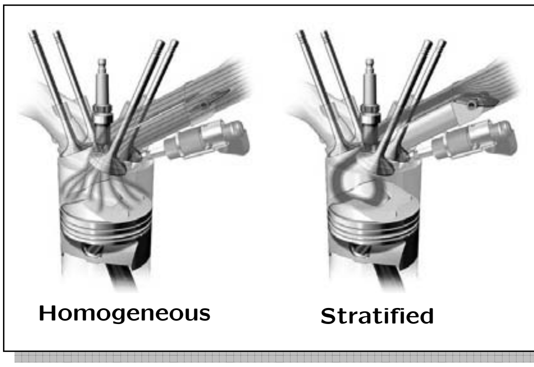
(Giorgetti, Ripaccioli, Bemporad, Kolmanovsky, Hrovat, IEEE Tr. Mechatronics, 2006)

DISC Engine

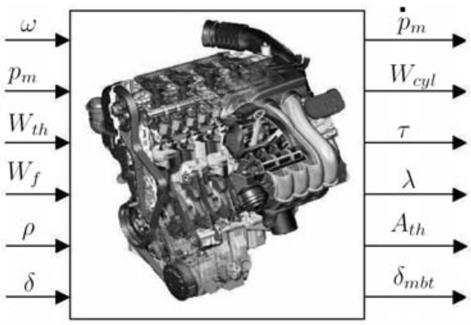
Two distinct regimes:

Regime	fuel injection	air-to-fuel ratio
Homogeneous combustion	intake stroke	λ =14.64
Stratified combustion	compression stroke	λ >14.64

Pro: reduce consumption up to 15%;
Con: complex treatment of exhaust gas



- **States**: intake manifold pressure (p_m)
- **Outputs:** Air-to-fuel ratio (λ), torque (τ), max-brake-torque spark timing (δ_{mbt})
- **Inputs:** spark advance (δ), air flow (W_{th}), fuel flow (W_f), combustion regime (ρ);
- **Disturbance:** engine speed (ω) [measured]



DISC Engine – Control Objective

Objective: Design a controller for the engine that

- Automatically chooses operating **mode** (homogeneous/stratified)
- Can cope with **nonlinear** dynamics
- Handles **constraints** (on A/F ratio, air-flow, spark)
- Achieves **optimal** performance (tracking of desired torque and A/F ratio)

DISC Engine - HYSDEL List

```
SYSTEM hysdisc{
   INTERFACE{
   STATE{
     REAL pm      [1, 101.325];
     REAL xtau    [-1e3, 1e3];
     REAL xlam    [-1e3, 1e3];
     REAL taud    [0,    100];
     REAL lamd    [10,    60];
      }
   OUTPUT{
     REAL lambda, tau, ddelta;
         }
   INPUT{
     REAL Wth     [0,38.5218];
     REAL Wf      [0,      2];
     REAL delta   [0,     40];
     BOOL rho;
      }
   PARAMETER{
     REAL Ts, pm1, pm2;
     …
      }
   }

   IMPLEMENTATION{
   AUX{
     REAL lam,taul,dmbtl,lmin,lmax;
      }
   DA{
   lam={IF rho THEN l11*pm+l12*Wth...
               +l13*Wf+l14*delta+l1c
        ELSE   l01*pm+l02*Wth+l03*Wf...
               +l04*delta+l0c      };

      taul={IF rho THEN tau11*pm+...
        tau12*Wth+tau13*Wf+tau14*delta+tau1c
            ELSE   tau01*pm+tau02*Wth...
             +tau03*Wf+tau04*delta+tau0c };

      dmbtl ={IF rho THEN dmbt11*pm+dmbt12*Wth...
              +dmbt13*Wf+dmbt14*delta+dmbt1c+7
              ELSE dmbt01*pm+dmbt02*Wth...
              +dmbt03*Wf+dmbt04*delta+dmbt0c-1};

      lmin ={IF rho THEN 13 ELSE 19};
      lmax ={IF rho THEN 21 ELSE 38};
           }
      CONTINUOUS{
          pm=pm1*pm+pm2*Wth;
          xtau=xtau+Ts*(taud-taul);
          xlam=xlam+Ts*(lamd-lam);
          taud=taud; lamd=lamd;
               }
      OUTPUT{
         lambda=lam-lamd;
         tau=taul-taud;
         ddelta=dmbtl-delta;
         }
      MUST{
         lmin-lam     <=0;
         lam-lmax     <=0;
         delta-dmbtl  <=0;
         }
      }
   }
```

MPC of DISC Engine

$$\min \sum_{k=0}^{N-1} (u_k - u_r)' R (u_k - u_r) + (y_k - y_r)' Q (y_k - y_r) + (x_{k+1} - x_r)' S (x_{k+1} - x_r)$$

subj. to $\begin{cases} x_0 = x(t), \\ \text{hybrid model} \end{cases}$

Solve **MIQP problem** (mixed-integer quadratic program) to compute $u(t)$

$u(t) = [W_{th}(t),\ W_f(t),\ \delta(t),\ \rho(t)]$

Weights:

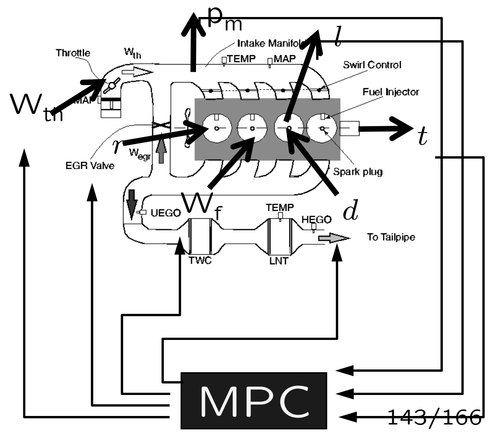
$$R = \begin{pmatrix} 0.01 & 0 & 0 & 0 \\ 0 & 0.001 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

r_ρ (prevents unneeded chattering)

$$Q = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0.001 & 0 \\ 0 & 0 & 0.01 \end{pmatrix} \quad S = \begin{pmatrix} 0.04 & 0 & 0 \\ 0 & 1500 & 0 \\ 0 & 0 & 0.01 \end{pmatrix}$$

q_τ q_λ s_{ε_τ} s_{ε_λ}

main emphasis on torque



Integral Action

Integrators on torque error and air-to-fuel ratio error are added to obtain zero offsets in steady-state:

$$\epsilon_\tau(t+1) = \epsilon_\tau(t) + T \cdot (\tau_{ref} - \tau)$$
$$\epsilon_\lambda(t+1) = \epsilon_\lambda(t) + T \cdot (\lambda_{ref} - \lambda)$$

T = sampling time

τ_{ref}, λ_{ref} brake torque and air-to-fuel references

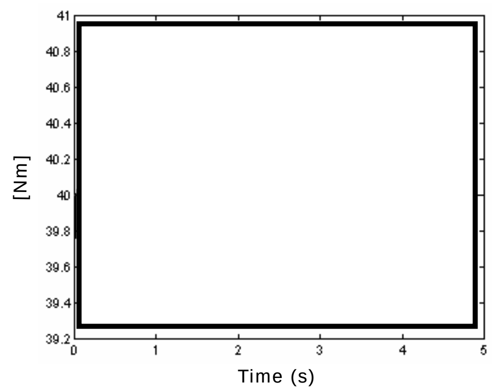
Simulation based on nonlinear model confirms zero offsets in steady-state

(despite the model mismatch)

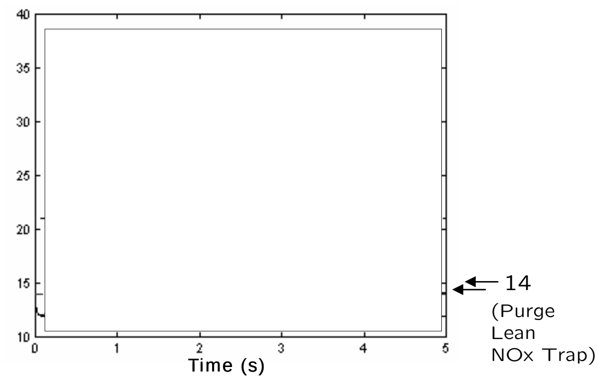
Simulation Results (nominal engine speed)

$\omega = 2000$ rpm

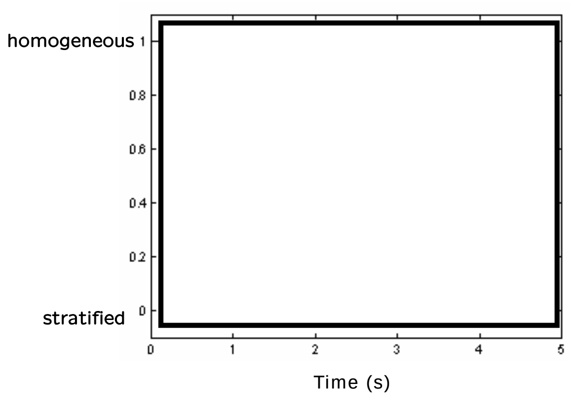
Engine Brake Torque



Air-to-Fuel Ratio



Combustion mode

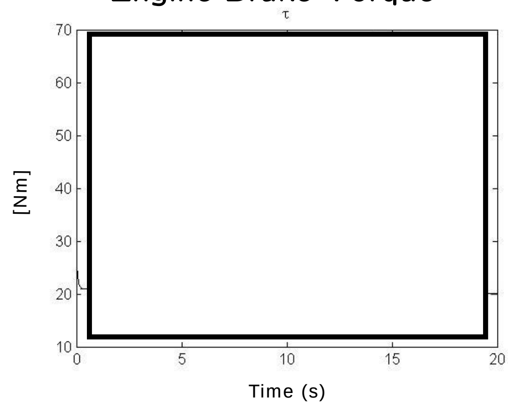


- Control horizon **N=1**;
- Sampling time **Ts=10 ms**;
- PC Xeon 2.8 GHz + Cplex 9.1

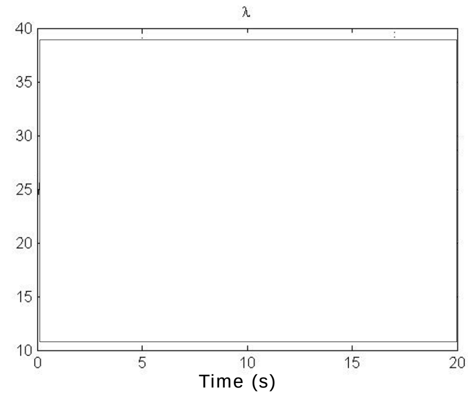
$\simeq$3 ms per time step

Simulation Results (varying engine speed)

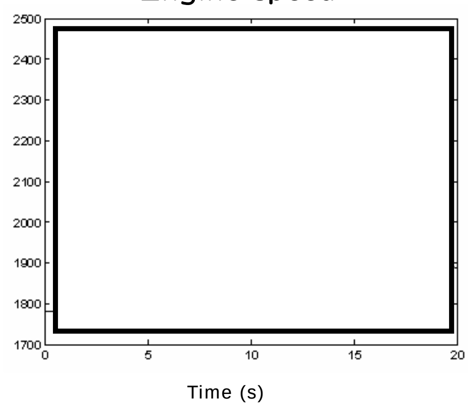
Engine Brake Torque



Air-to-Fuel Ratio



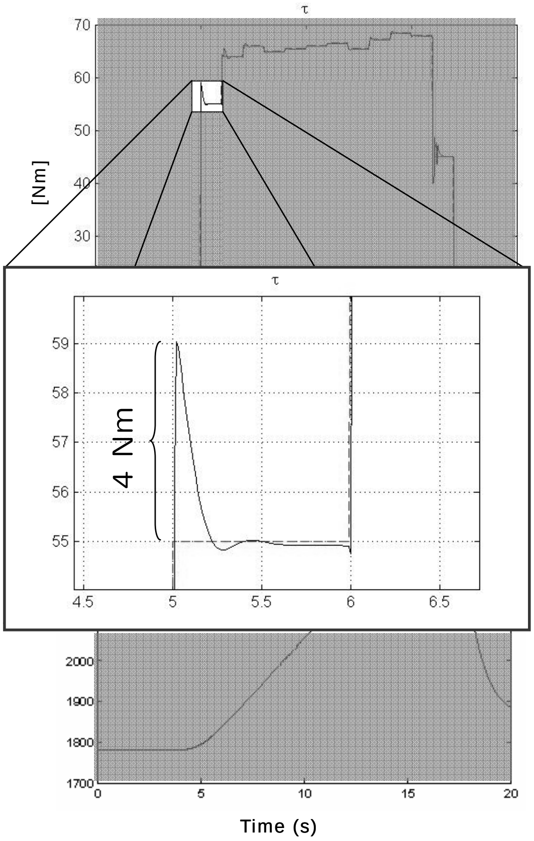
Engine speed

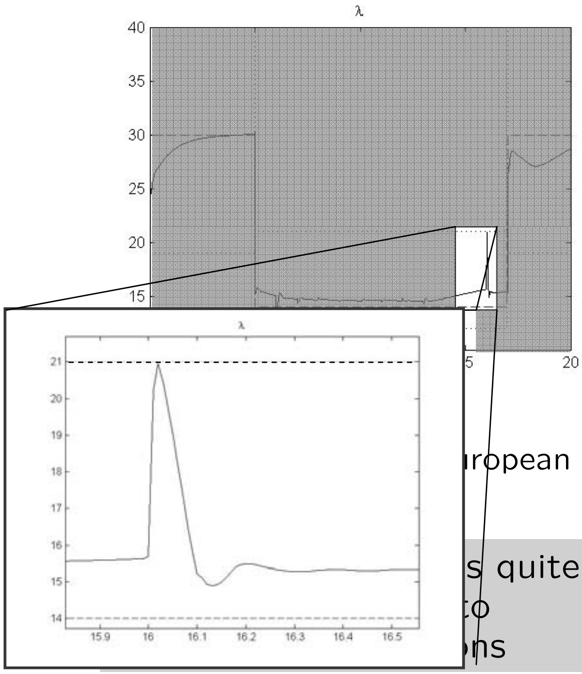


20 s segment of the European drive cycle (NEDC)

Hybrid MPC design is quite robust with respect to engine speed variations

Simulation Results (varying engine speed)





Control code too complex (MILP) ⇒ not implementable !

Explicit MPC Controller

Explicit control law: $u(t) = f(\theta(t))$

where: $u = [W_{th}\ W_f\ \delta\ \rho]'$

$\theta = [p_m\ \epsilon_\tau\ \epsilon_\lambda\ \tau_{\text{ref}}\ \lambda_{\text{ref}}\ p_{m,\text{ref}}\ W_{th,\text{ref}}\ W_{f,\text{ref}}\ \delta_{\text{ref}}]'$

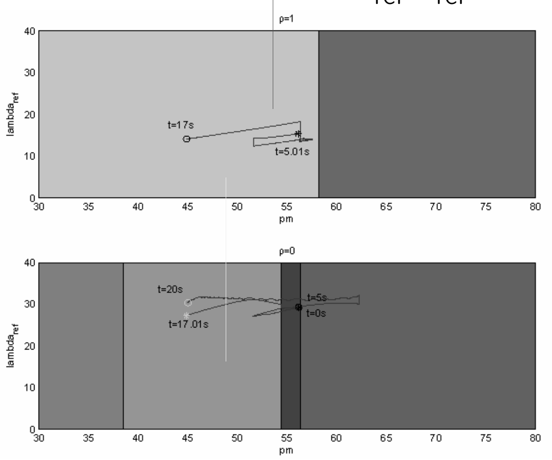
N=1 (control horizon)

42 partitions

- Time to compute explicit MPC: ≈ 3s;
- Sampling time Ts=10 ms;
- PC Xeon 2.8 GHz + Cplex 9.1

≃**8 μs** per time step

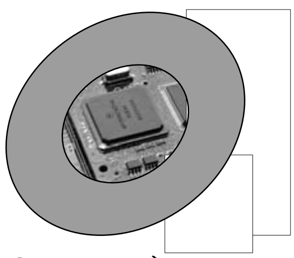
Cross-section by the τ_{ref}-λ_{ref} plane



ρ=0

ρ=1

≈ 3ms on μcontroller Motorola MPC 555 43kb RAM (custom made for Ford)



Traction Control System

(Borrelli, Bemporad, Fodor, Hrovat, IEEE Tr. Mechatronics, 2006)

Vehicle Traction Control

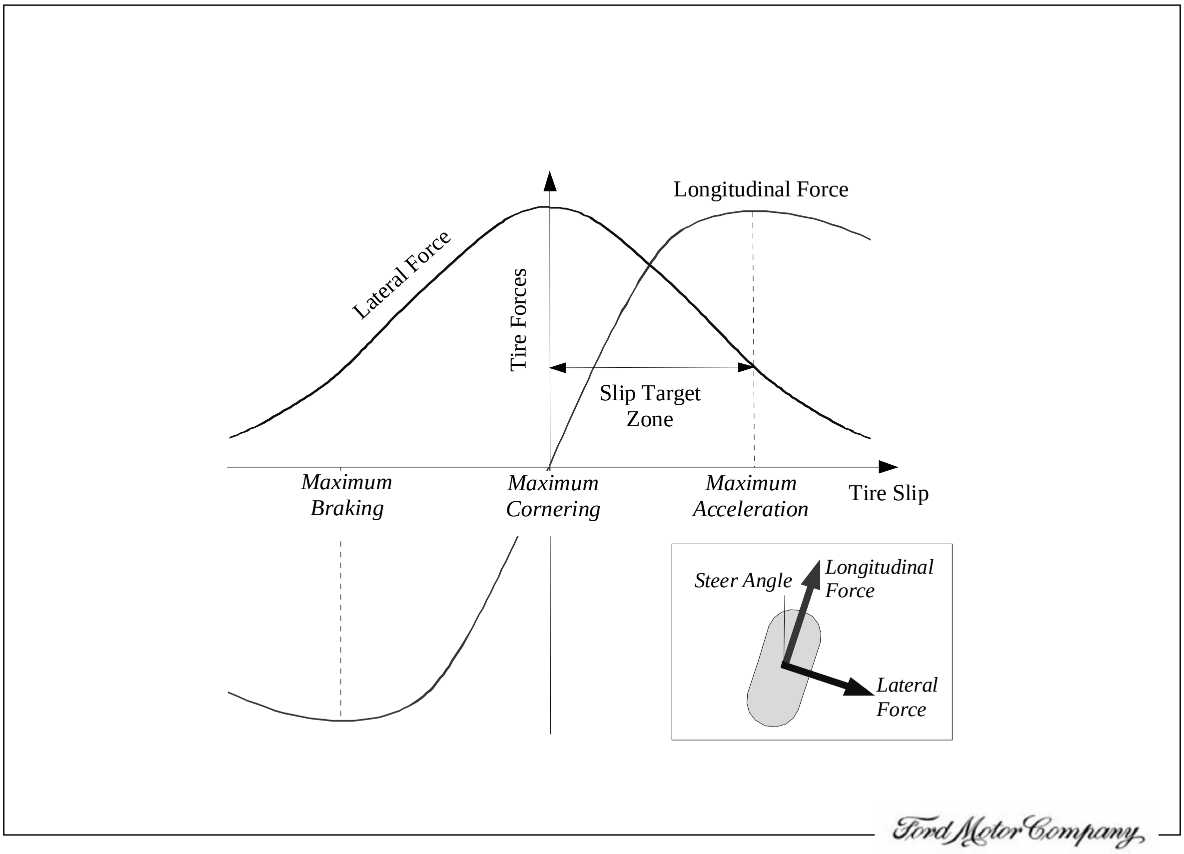
Improve driver's ability to control a vehicle
under adverse external conditions (wet or icy roads)

Model
nonlinear, uncertain,
constraints

Controller
suitable for real-time
implementation

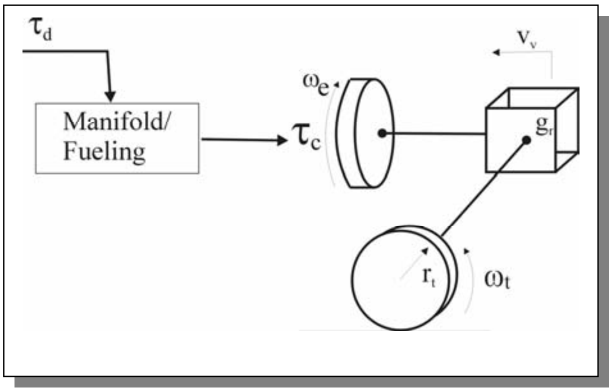
MLD hybrid framework + optimization-based control strategy

Tire Force Characteristics



Simple Traction Model

(Borrelli, Bemporad, Fodor, Hrovat, 2006)



- Mechanical system

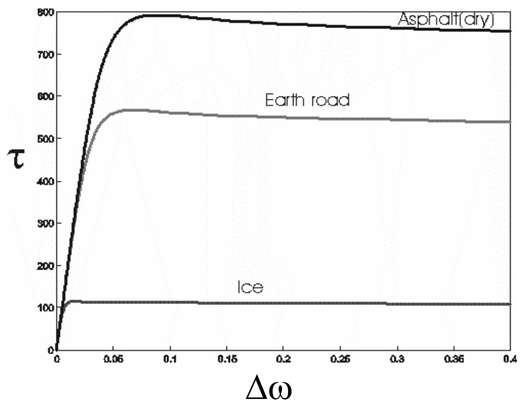
$$\dot{\omega}_e = \frac{1}{J_e}\left(\tau_c - b_e\omega_e - \frac{\tau_t}{g_r}\right)$$

$$\dot{v}_v = \frac{\tau_t}{m_v r_t}$$

- Manifold/fueling dynamics

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

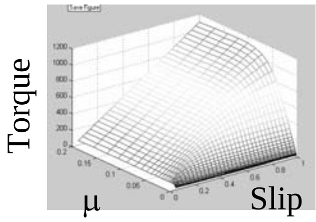
- Tire torque τ_t is a function of slip $\Delta\omega$ and road surface adhesion coefficient μ



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

wheel slip

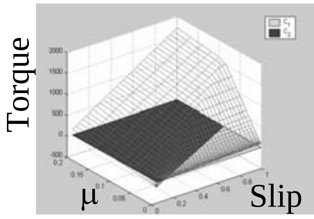
Hybrid Model

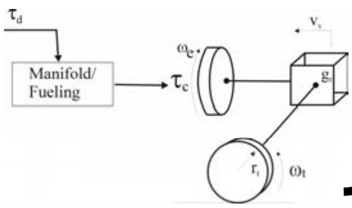


Nonlinear tire torque $\tau_t = f(\Delta\omega\,,\ \mu)$

PWA Approximation

(PWL Toolbox, Julian, 2000)





HYSDEL → 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)	4 variables
Input u(t)	1 variable
Aux. Binary vars δ(t)	1 variable
Aux. Continuous vars z(t)	3 variables
Mixed-integer inequalities	14

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

go to demo **/demos/traction/init.m**

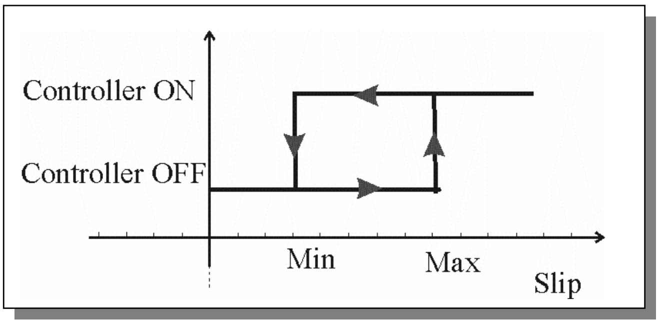
Performance and Constraints

- Control objective:

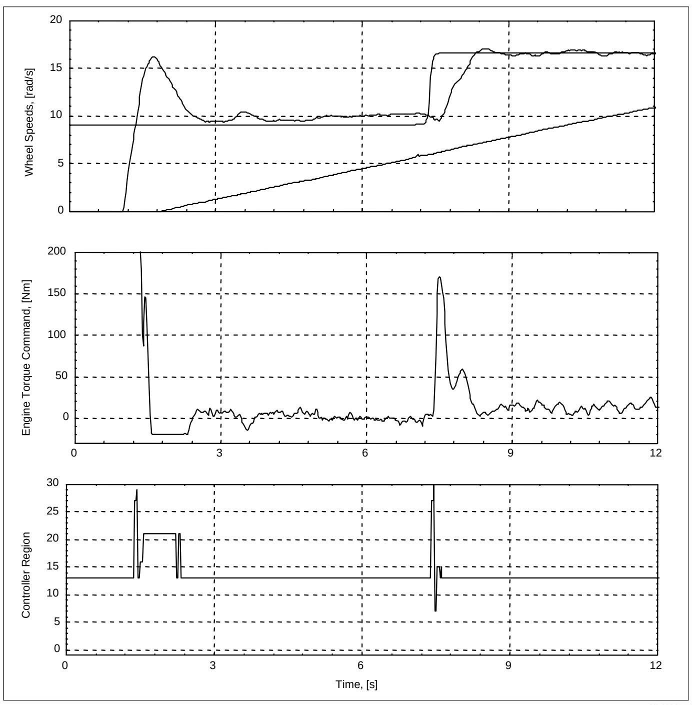
$$\begin{array}{ll} \min & \sum_{k=0}^{N} |\Delta\omega(t+k|t) - \Delta\omega_{\text{des}}| \\ \text{s.t.} & \text{MLD dynamics} \end{array}$$

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

- Note: a logic constraint (hysteresis) may be also taken into account



Experimental Results



Ford Motor Company

Experiment

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

Ford Motor Company

Current Research Activities

Decentralized MPC

(Alessio, Bemporad, ECC'07)

Process model: coupled dynamics

$x_{k+1} = Ax_k + Bu_k,\ x \in \mathbb{R}^n,\ u \in \mathbb{R}^m$ (Hp: system open-loop as. stable)

Cost function: full matrices Q,R,P

$$\sum_{k=0}^{N-1} \left[x'_k Q x_k + u'_k R u_k\right] + x'_N P x_N$$

Constraints: $u_{\min} \le u_k \le u_{\max}$

Main idea: replace centralized MPC algorithm with m simpler decentralized MPC algorithms, one for each actuator

- Limited communication among subsystems
- Drop optimality in favor of computation simplicity ...
- ... but still guarantee closed-loop stability !

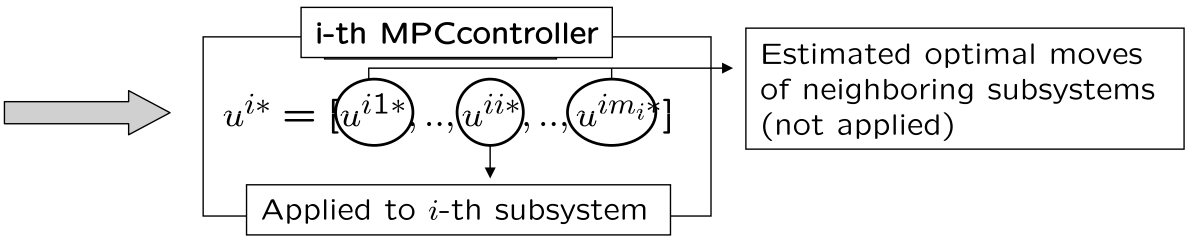
Decentralized MPC

Prediction models: partially decoupled dynamics

$x^i(k+1) = A_i x^i(k) + B_i u^i(k),\ x^i \in \mathbb{R}^{n_i},\ u^i \in \mathbb{R}^{m_i}$

(Hp: A_i strictly Hurwitz)

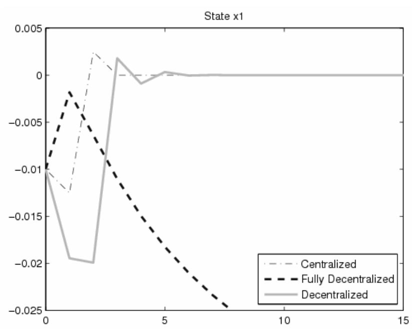
x^i = subset of of x, u^i = subset of of u



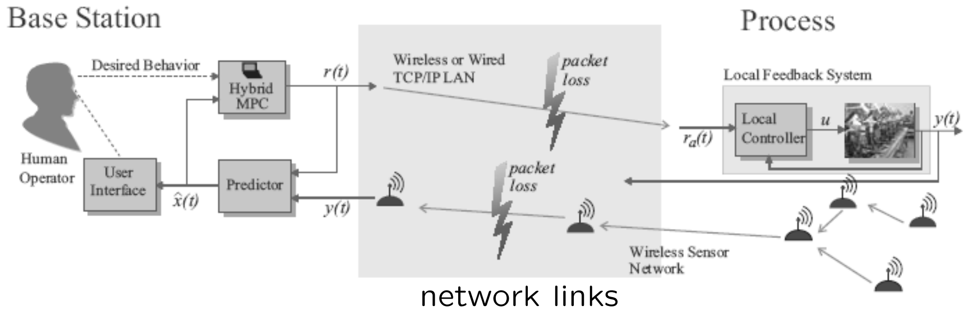
Stability result: local stability can be checked by eigenvalue computation, global stability via LMIs

$$u(t) = \begin{bmatrix} -93.7776 & 58.3261 & 17.5837 \\ -23.5555 & -95.7687 & -4.8290 \\ -91.0992 & -6.3035 & -169.0643 \end{bmatrix} x(t)$$ centralized (local gain)

$$u(t) = \begin{bmatrix} -107.7638 & 0 & 14.7213 \\ -19.2757 & -95.4731 & 0 \\ -89.7068 & 0 & -168.7838 \end{bmatrix} x(t)$$ decentralized (local gain)



Hybrid MPC & Wireless Sensor Networks



- Measurements acquired and sent to base station (MPC) by wireless sensors

- MPC computes the optimal plan when new measurements arrive

- Optimal plan implemented by local controller if received in time, otherwise previous plan still kept

Packet loss possible along both network links,
delayed packets must be discarded (out-of-date data)

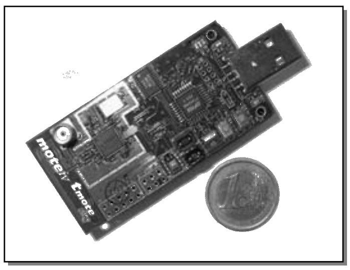
stochastic / robust hybrid MPC needed

Demo Application in Wireless Automation

(Bemporad, Di Cairano, Henriksson)

(Automatic Control Lab, Univ. Siena)

Telos mote



TinyOS→ Java→ Matlab
interface developed at UNISI

- Telos motes provide wireless temperature feedback in Matlab/Simulink
- xPC-Target link

- MPC algorithm adjusts belt speed and turns lamps on/off (HybTBX+CPLEX)

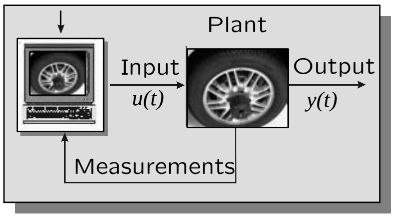
<u>Hybrid MPC problem:</u>
- 2 binary inputs (lamps)
- 1 continuous input (speed)
- PWL state function heating=f(position)
- Outputs: temp, position
- Sampling = 4Hz

Conclusions

- **Hybrid systems** as a framework for new applications, where both logic and continuous dynamics are relevant

- **Supervisory MPC controllers** schemes can be synthesized via on-line mixed-integer programming (MILP/MIQP)

- **Piecewise Linear MPC Controllers** can be synthesized off-line via multiparametric programming for fast-sampling applications



Hybrid modeling and MPC design

Multiparametric programming

$$u(x,r) = \begin{cases} F_1 x + E_1 r + g_1 & \text{if } H_1 \left[\begin{smallmatrix} x \\ r \end{smallmatrix}\right] \leq K_1 \\ \vdots & \vdots \\ F_M x + E_M r + g_M & \text{if } H_M \left[\begin{smallmatrix} x \\ r \end{smallmatrix}\right] \leq K_M \end{cases}$$

C-code download & testing

Conclusions

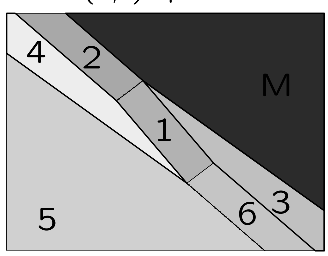
- **Hybrid systems** as a framework for new applications, where both logic and continuous dynamics are relevant

- **Supervisory MPC controllers** schemes can be synthesized via on-line mixed-integer programming (MILP/MIQP)

- **Piecewise Linear MPC Controllers** can be synthesized off-line via multiparametric programming for fast-sampling applications

$$u(x,r) = \begin{cases} F_1 x + E_1 r + g_1 & \text{if } H_1 \left[\begin{smallmatrix} x \\ r \end{smallmatrix}\right] \leq K_1 \\ \vdots & \vdots \\ F_M x + E_M r + g_M & \text{if } H_M \left[\begin{smallmatrix} x \\ r \end{smallmatrix}\right] \leq K_M \end{cases}$$

(x,r)-space



- **Matlab tools** are available to assist the whole design process

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

Acknowledgements

A. Alessio, M. Baotic, F. Borrelli, B. De Schutter,
S. Di Cairano, G. Ferrari-Trecate, K. Fukuda, N. Giorgetti,
M. Heemels, D. Hrovat, J. Julvez, I. Kolmanovski, M. Lazar,
L. Ljung, D. Mignone, M. Morari, D. Munoz de la Pena,
S. Paoletti, G. Ripaccioli, J. Roll, F.D. Torrisi

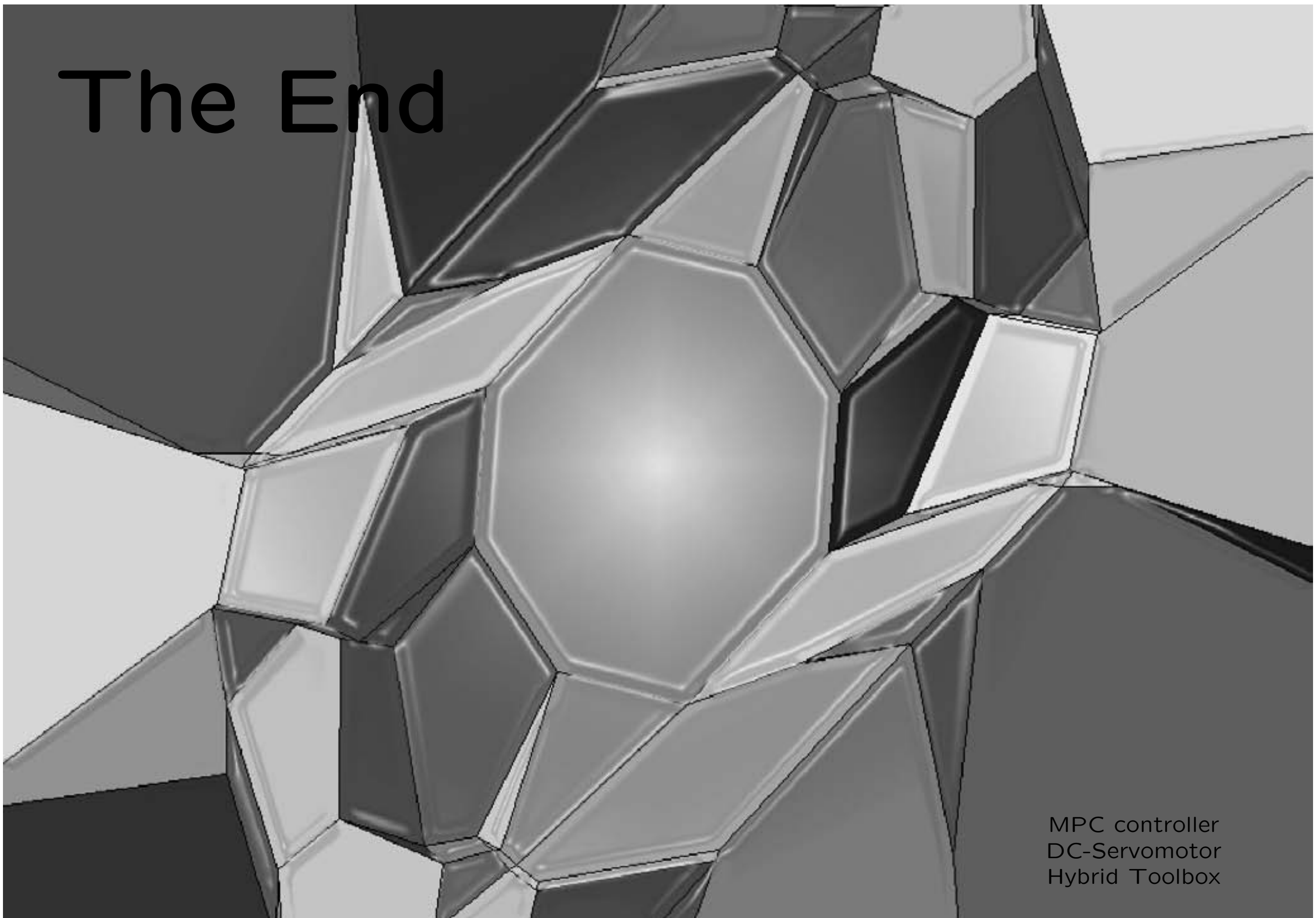
Announcement

2nd PhD School on Hybrid Systems
(Siena, July 19-22, 2007)

http://www.dii.unisi.it/hybrid/school07

HYSCOM
IEEE CSS Technical Committee on Hybrid Systems

IEEE CSS HYCON



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