Optimization-Based Methods for Controller Synthesis and Verification

Alberto Bemporad

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

"Computation and Control"

IST-2001-33520

The European Commission
Community Research
ist information society technologies

Review Meeting

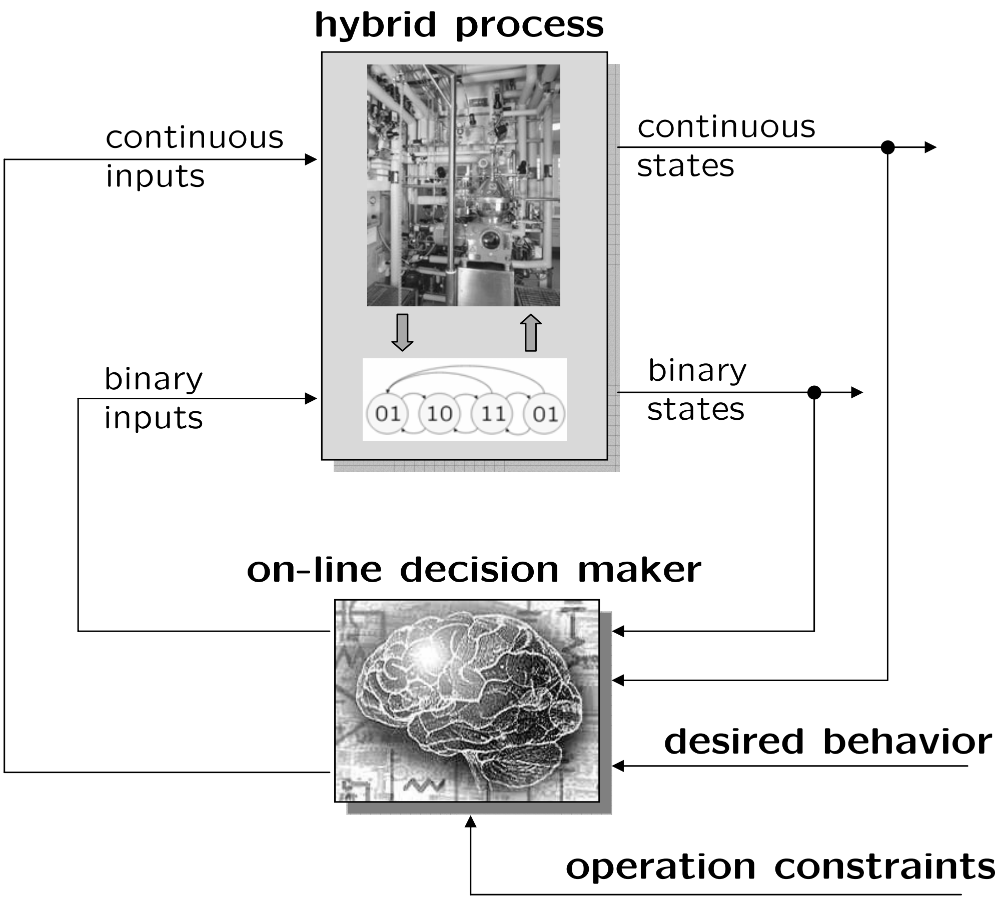
Zurich, March 12, 2005

COHES Group
Control and Optimization of Hybrid and Embedded Systems

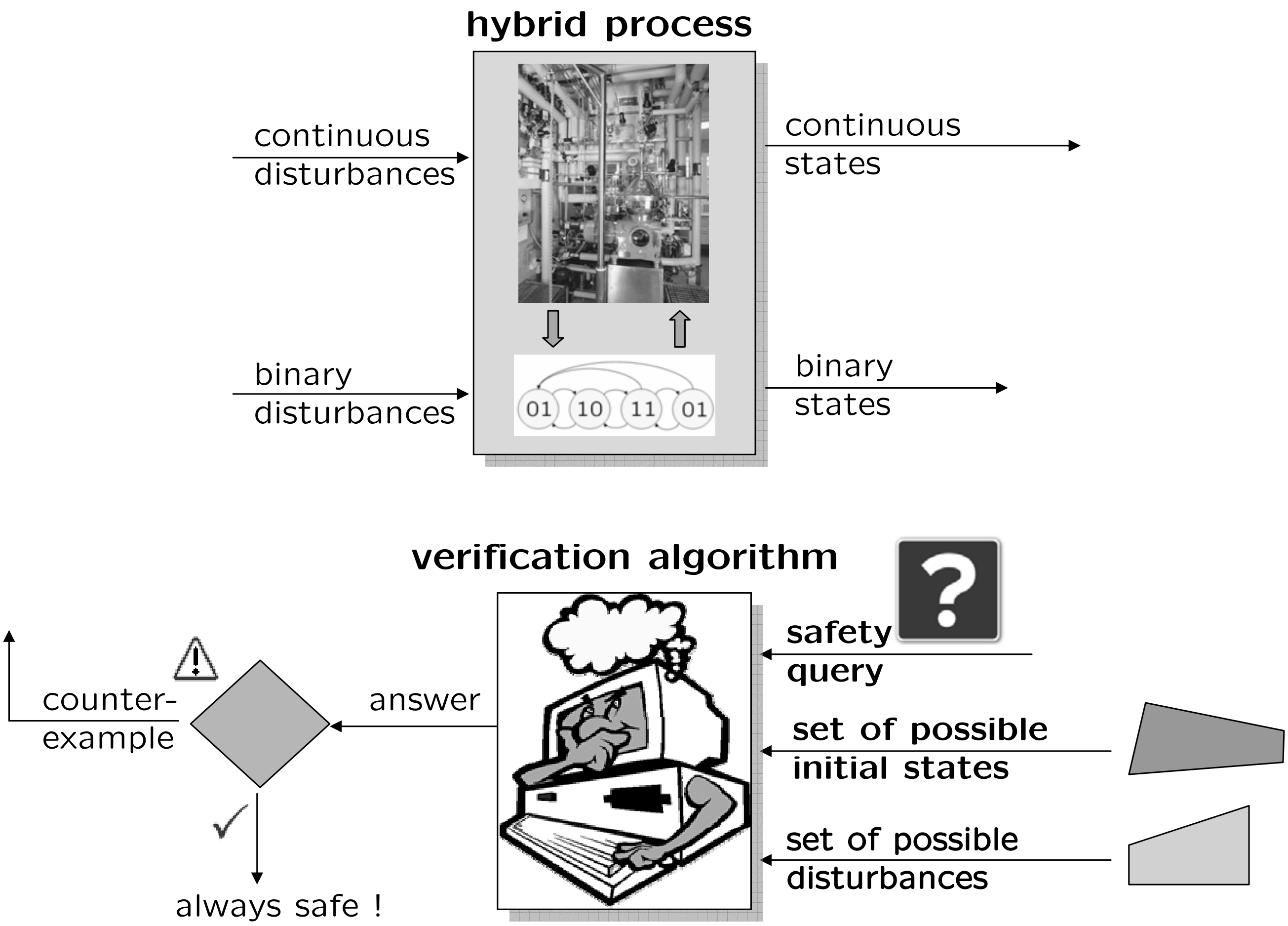
Dept. of Information Engineering

University of Siena, Italy
(founded in 1240)

Hybrid Control Problem

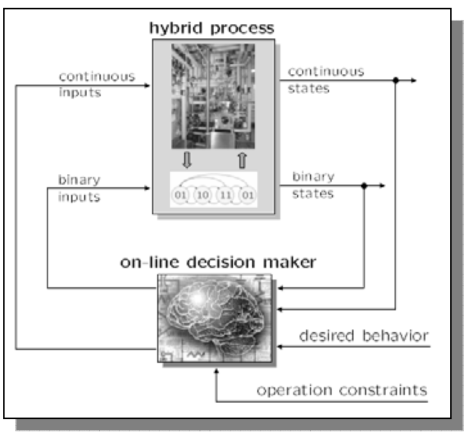


Hybrid Verification Problem

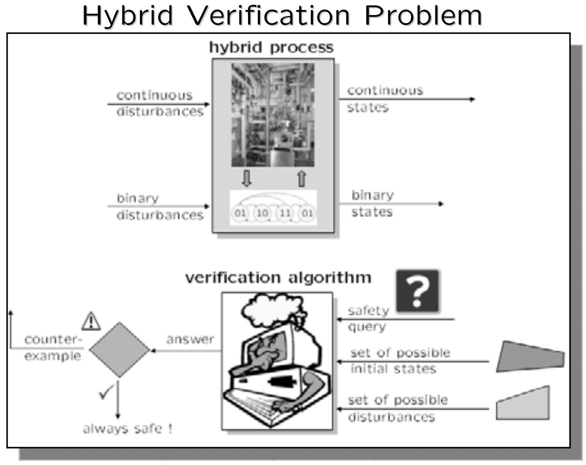


Model-based Optimization Approach

Hybrid Control Problem

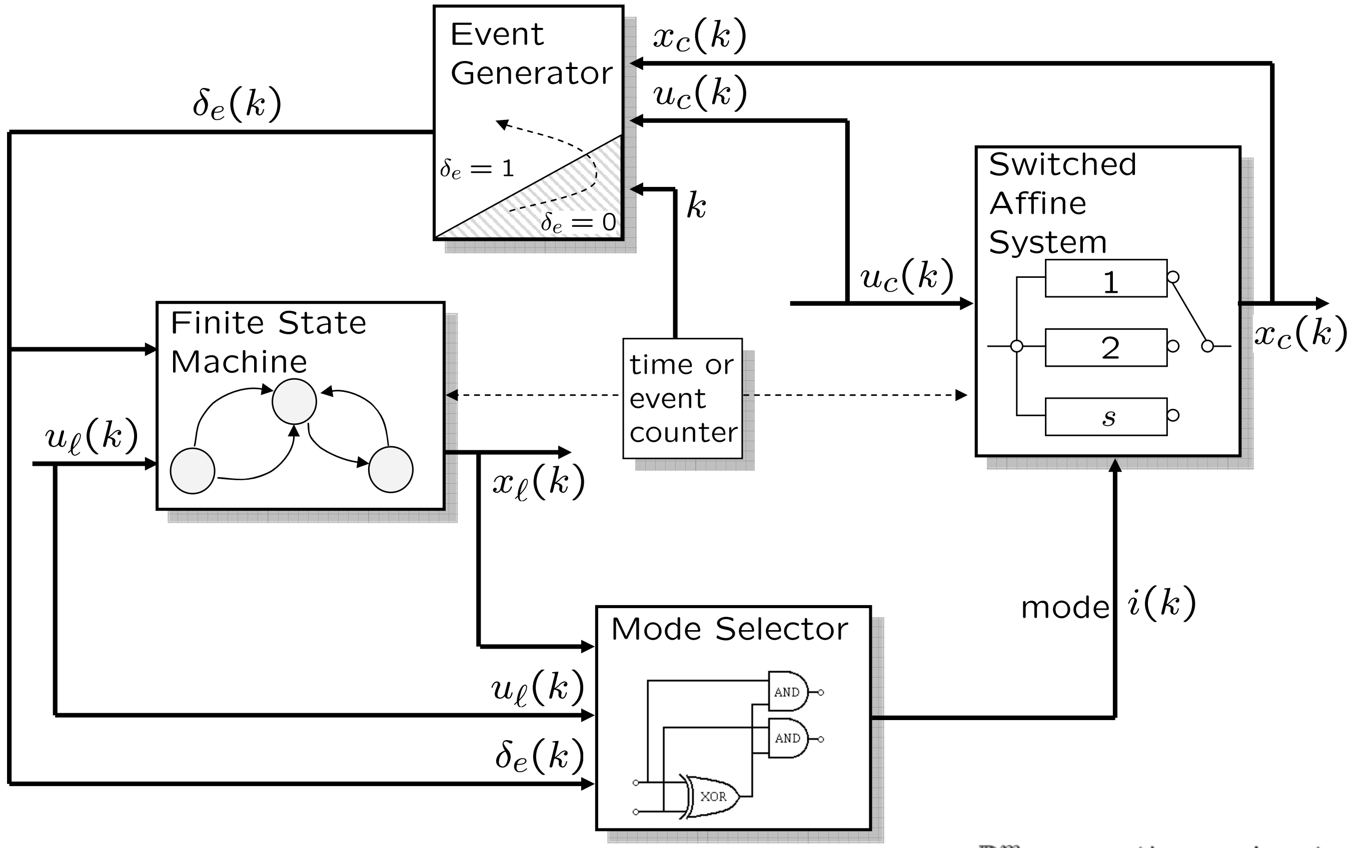


Hybrid Verification Problem



- Need for a <u>hybrid model</u> of the process reproducing the behavior of the process (simulation)
- A model suitable for controller synthesis and verification
- A model for which <u>computational tools</u> can be applied

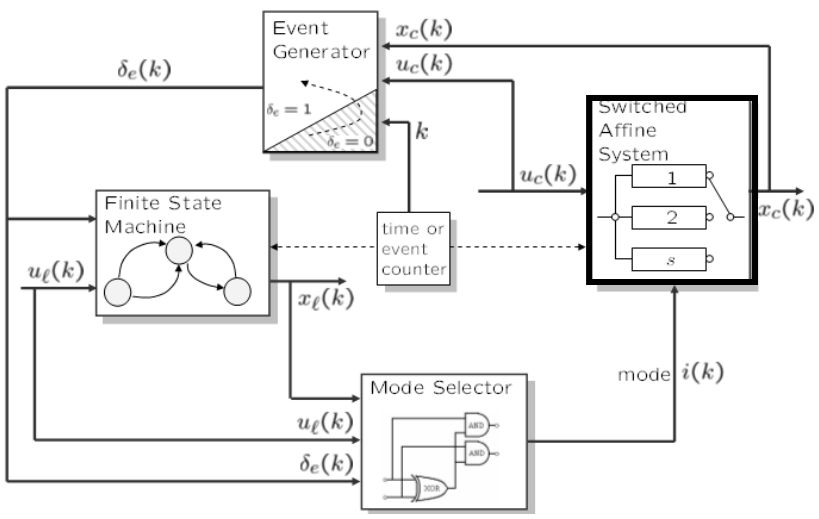
Hybrid Model: Discrete Hybrid Automaton



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

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

Switched Affine System

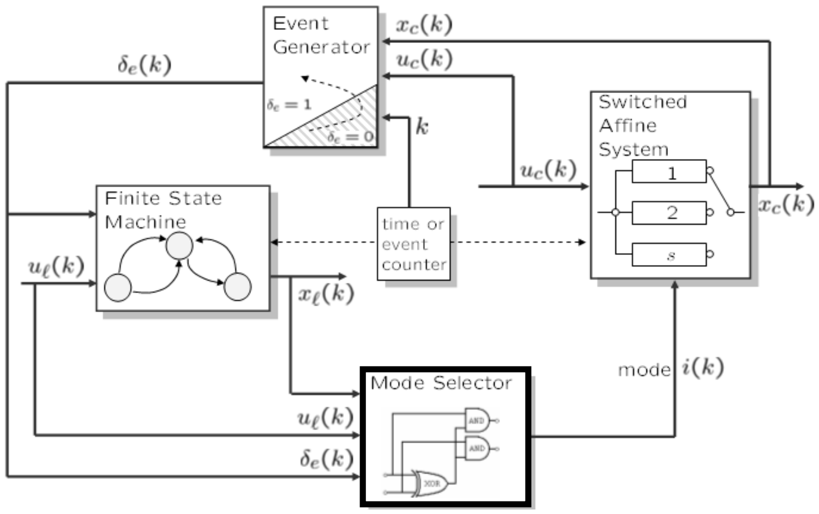


The affine dynamics depend on the current mode $i(k)$:

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

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

Mode Selector



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

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

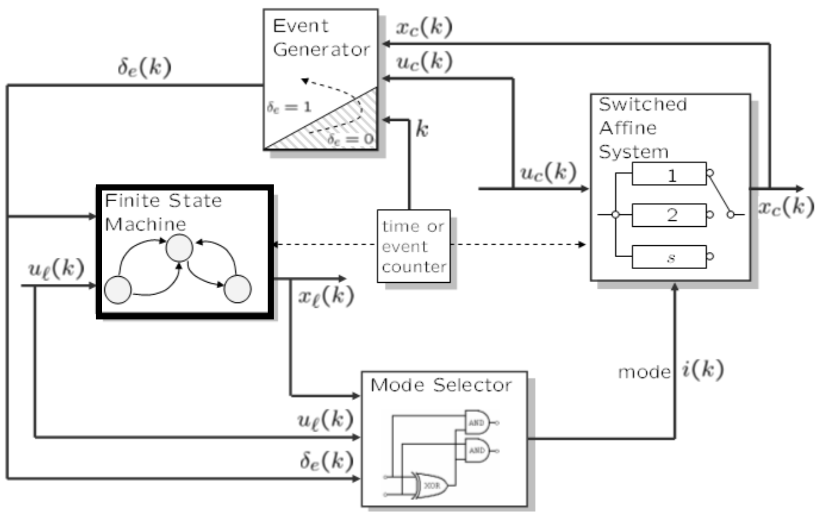
Example:

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

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

Finite State Machine

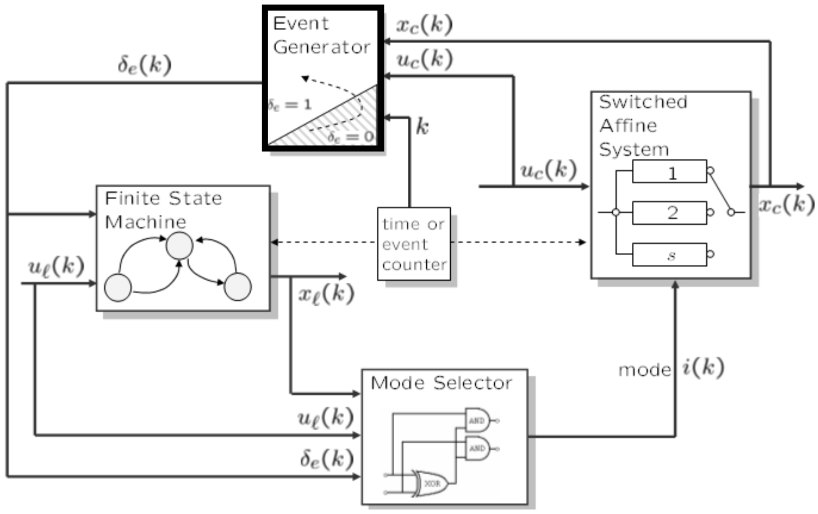


The binary state of the finite state machine evolves according to a Boolean state update function:

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

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

Event Generator



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

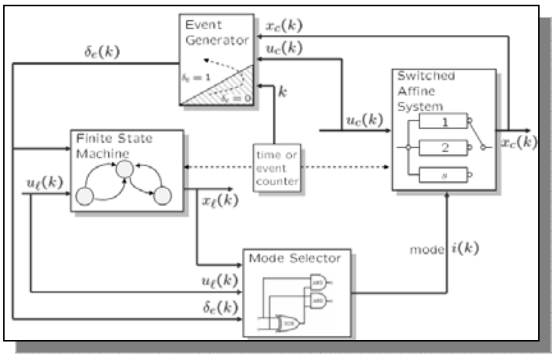
$$[\delta_e^i(k) = 1] \leftrightarrow [H^i x_c(k) + K^i u_c(k) \leq W^i]$$

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

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

Computational Hybrid Models

Discrete Hybrid Automaton



Piecewise Affine (PWA) Systems

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

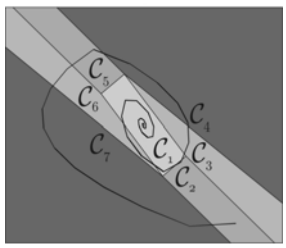
HYSDEL

(Torrisi, Bemporad, 2004)

MLD2PWA

(Bemporad, 2004)

state+input space



(Sontag 1981)

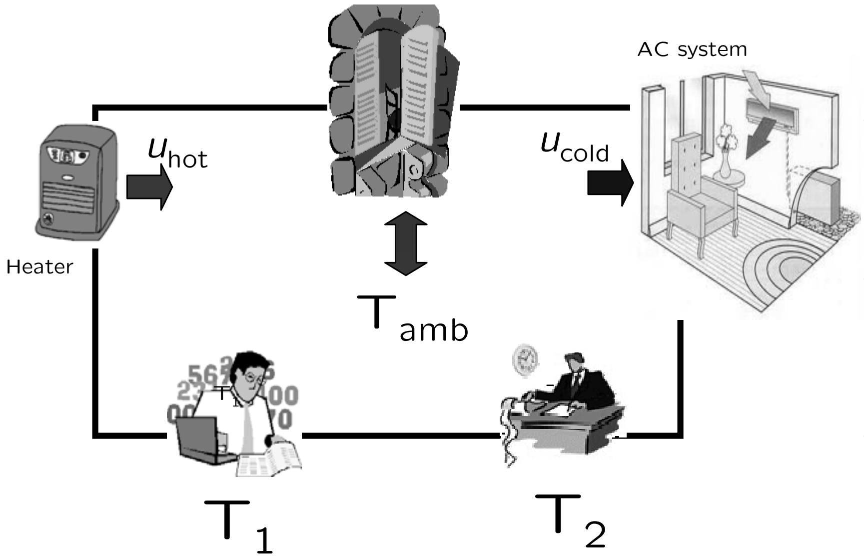
Mixed Logical Dynamical (MLD) Systems

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

(Bemporad, Morari 1999)

The translation from DHA to MLD/PWA is done automatically
(using symbolic/mathematical programming tools)

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

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

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**
(Bemporad, 2003-2005)

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

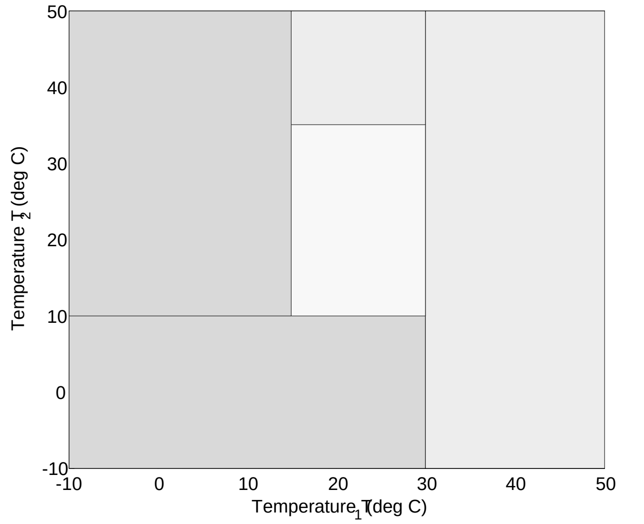
Hybrid PWA Model

- PWA model

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

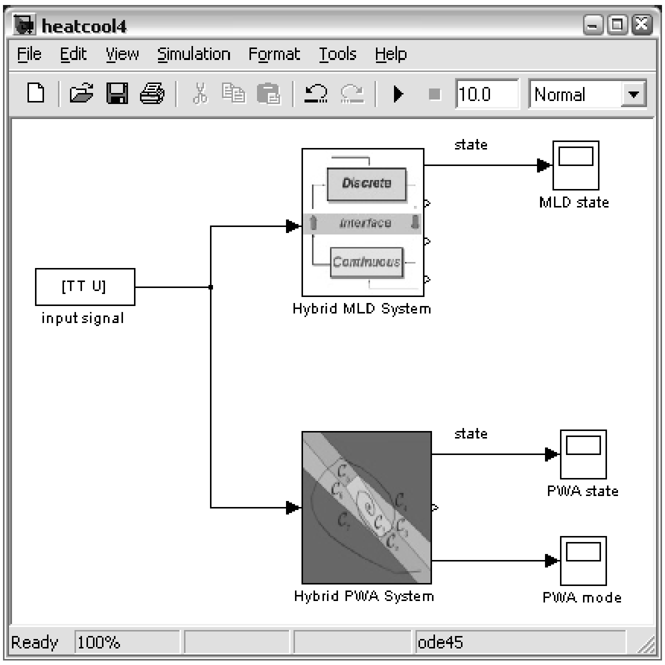
- 2 continuous states:
  (temperatures T_1,T_2)
- 1 continuous input:
  (room temperature T_{amb})
- 5 polyhedral regions
  (partition does not depend on input)

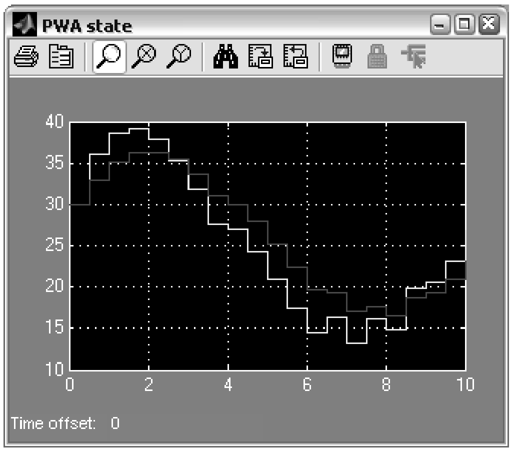
```
>>P=pwa(S);
```

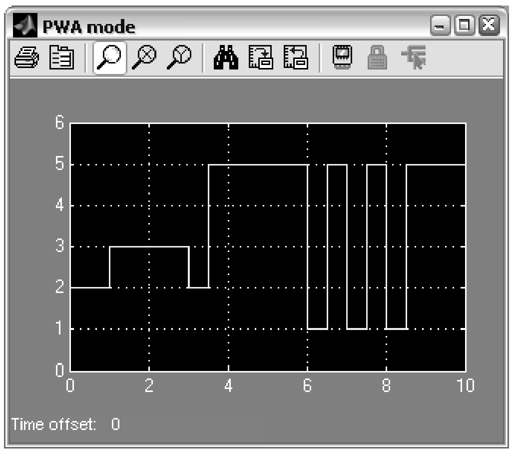


$\begin{aligned} u_{\text{hot}} &= 0 \\ u_{\text{cold}} &= 0 \end{aligned}$ $\quad$ $\begin{aligned} u_{\text{hot}} &= 0 \\ u_{\text{cold}} &= \bar{U}_C \end{aligned}$ $\quad$ $\begin{aligned} u_{\text{hot}} &= \bar{U}_H \\ u_{\text{cold}} &= 0 \end{aligned}$

Simulation in Simulink







Verification of DHA/MLD/PWA

Verification Algorithm

- **QUERY**: Is the target set X_f reachable after N steps from some initial state $x_0 \in X_0$ for some input profile $u \in U$?

- Computation: Solve the mixed-integer linear program (MILP)

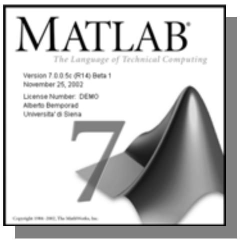
$$
\begin{array}{ll}
\min & 0 \\
\text{s.t.} & \left\{ \begin{array}{rcl}
x(k+1) & = & Ax(k) + B_1 u(k) + B_2 \delta(k) + B_3 z(k) \\
E_2 \delta(k) & + & E_3 z(k) \leq E_1 u(k) + E_4 x(k) + E_5 \\
S_u u(k) & \leq & T_u \quad (u(k) \in U) \\
 & & k = 0, 1, \ldots, N-1 \\
 & & \\
S_0 x(0) & \leq & T_0 \quad (x(0) \in X_0) \\
S_f x(N) & \leq & T_f \quad (x(N) \in X_f)
\end{array} \right.
\end{array}
$$

with respect to u(0),δ(0),z(0),..., u(N-1),δ(N-1),z(N-1),x(0)

- **Alternative solutions**:
  - Exploit the special structure of the problem and use polyhedral computation. (Torrisi, 2003)
  - Use abstractions (LPs) + SAT solvers (Giorgetti, Pappas, 2005)

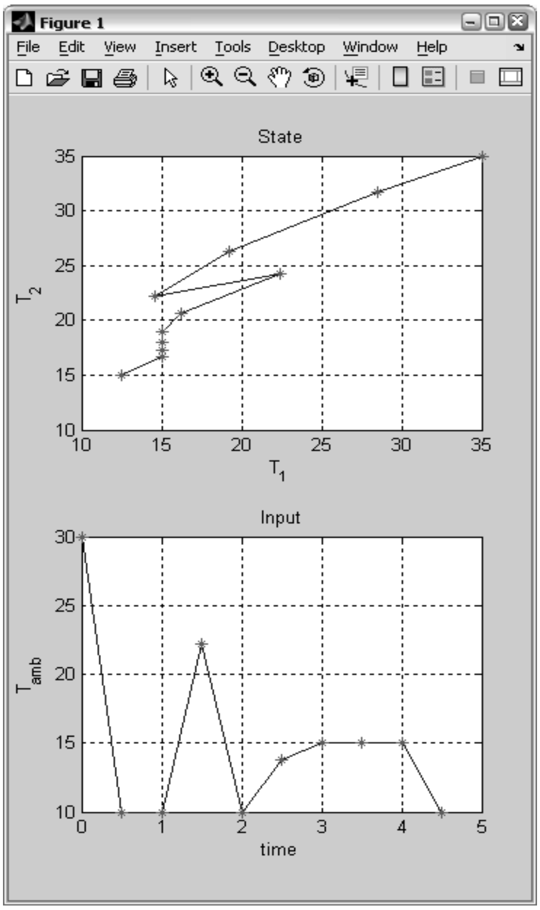
Verification Example

- MLD model: room temperature system
- $X_f = \{\begin{bmatrix} T_1 \\ T_2 \end{bmatrix} : 10 \leq T_1, T_2 \leq 15\}$ (set of unsafe states)
- $X_0 = \{\begin{bmatrix} T_1 \\ T_2 \end{bmatrix} : 35 \leq T_1, T_2 \leq 40\}$ (set of initial states)
- $U = \{T_{\text{amb}} : 10 \leq T_{\text{amb}} \leq 30\}$ (set of possible inputs)
- N=10 (time horizon)

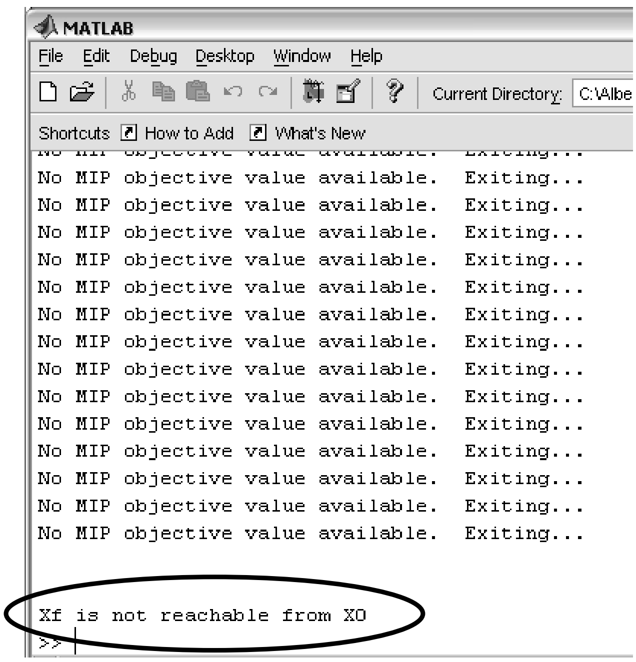



```
>>[flag,x0,U]=reach(S,N,Xf,X0,umin,umax);
```

Verification Example



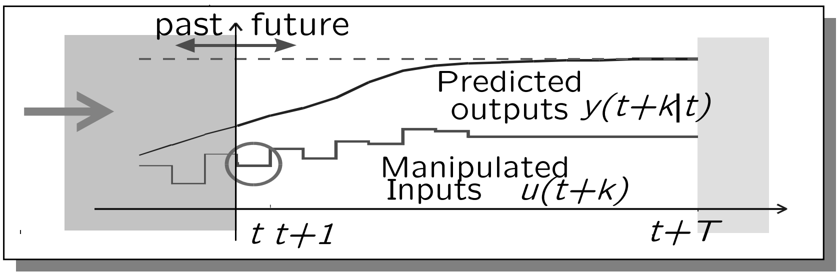
$U = \{T_{\text{amb}} : 10 \leq T_{\text{amb}} \leq 30\}$



$U = \{T_{\text{amb}} : 20 \leq T_{\text{amb}} \leq 30\}$

Controller Synthesis

Control Strategy: MPC



Model Predictive (MPC) Control

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

$$\min_{u(t),\ldots,u(t+T-1)} \sum_{k=0}^{T-1} \|R(y(t+k|t) - r(t+k))\|_p + \|Qu(t+k)\|_p$$

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

$p = 1, 2, \infty \qquad \|v\|_2 = v'v \quad \|v\|_\infty = \max |v_i| \quad \|v\|_1 = \sum v_i$

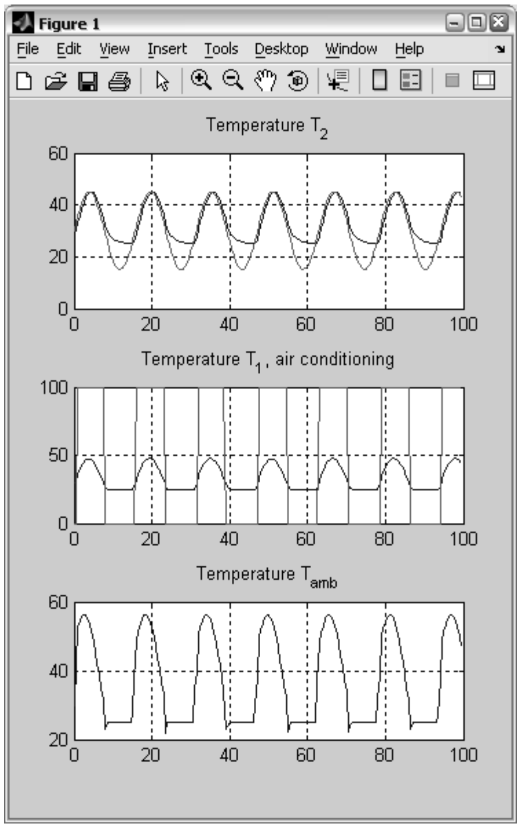
- Apply only $u(t) = u^*(t)$ (discard the remaining optimal inputs);
- Repeat the whole optimization at time $t+1$

Hybrid MPC - Example

```
>>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];
```

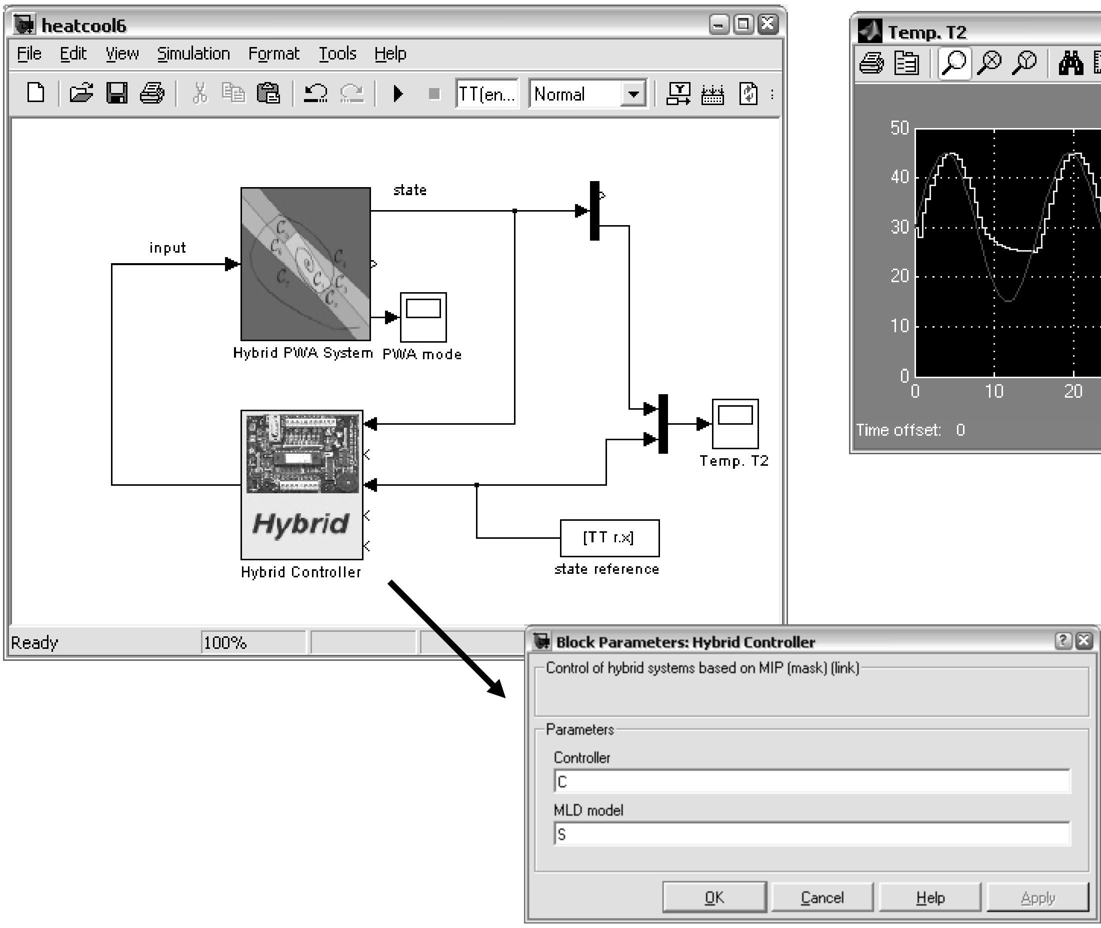
```
>>C=hybcon(S,Q,N,limits,refs);
```

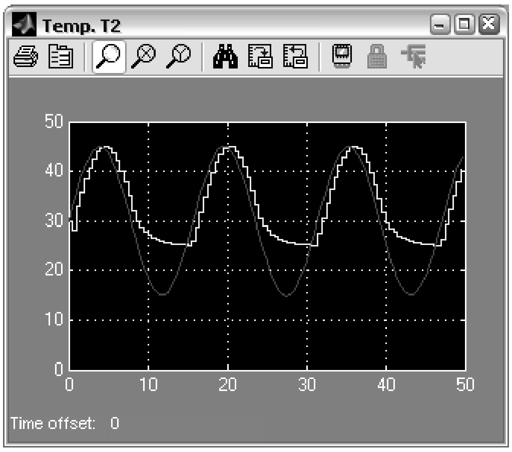
```
>> C

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 - Example





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> $x(t)$

$$\min_\zeta J(\zeta, x(t)) = \begin{cases} f'\zeta & \infty/1\text{-norms} \\ \zeta' H\zeta + f'\zeta & \text{quadratic forms} \end{cases}$$

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

multi-parametric programming

Explicit Hybrid MPC

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

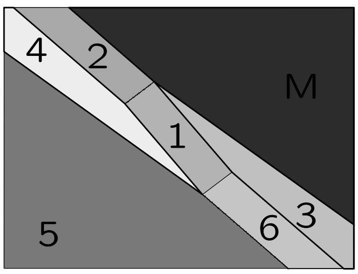
$$\text{subject to} \begin{cases} \text{PWA model} \\ x(0) = x \end{cases}$$

- Solution $u(x,r)$ found via a combination of
  - Dynamic programming or enumeration of feasible mode sequences, multiparametric linear or quadratic programming, and polyhedral computation. (Borrelli, Baotic, Bemporad, Morari, 2003) (Mayne, ECC 2001) (Alessio, Bemporad, 2005)

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

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

(x,r)-space

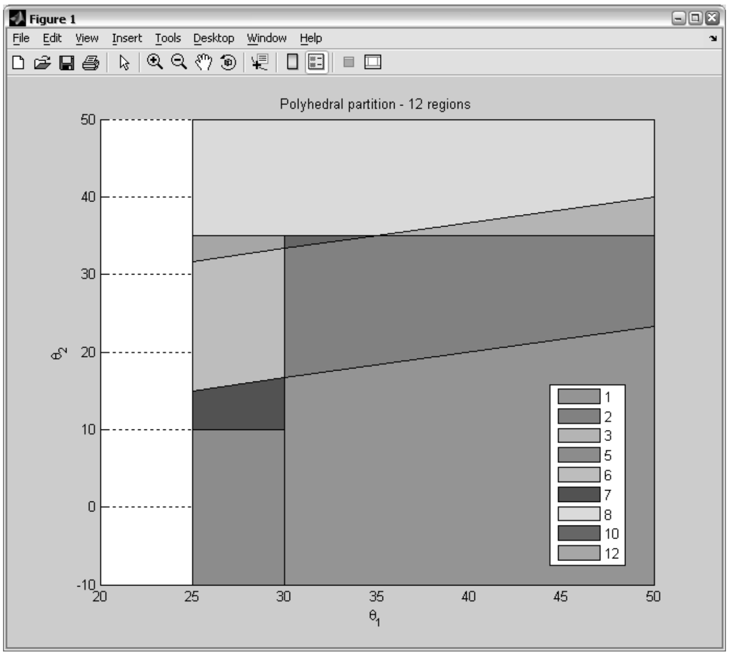


<u>Note</u>: in the quadratic case the partition may not be fully polyhedral

Explicit Hybrid MPC - Example

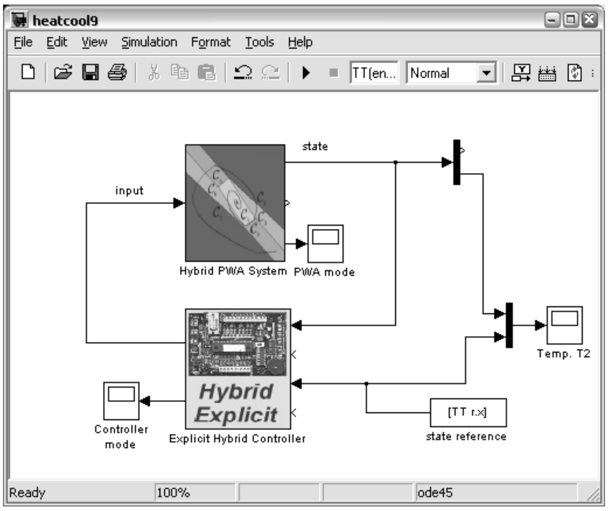
```
>>E=expcon(C,range,options);
```

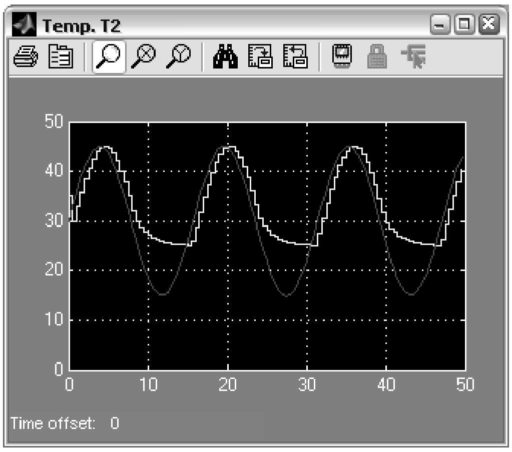
```
>> E

Explicit controller (based on hybrid controller C)
  3 parameter(s)
  1 input(s)
 11 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.
>>
```



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

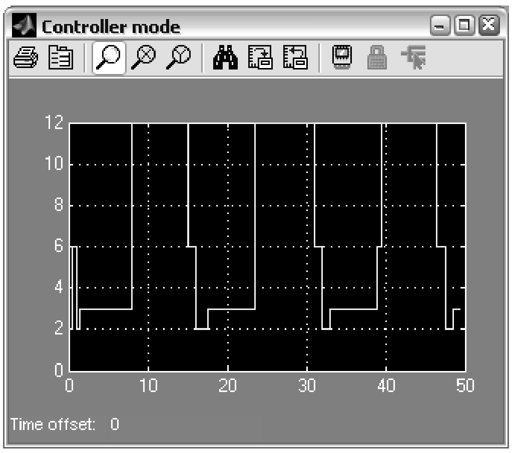
Explicit Hybrid MPC - Example



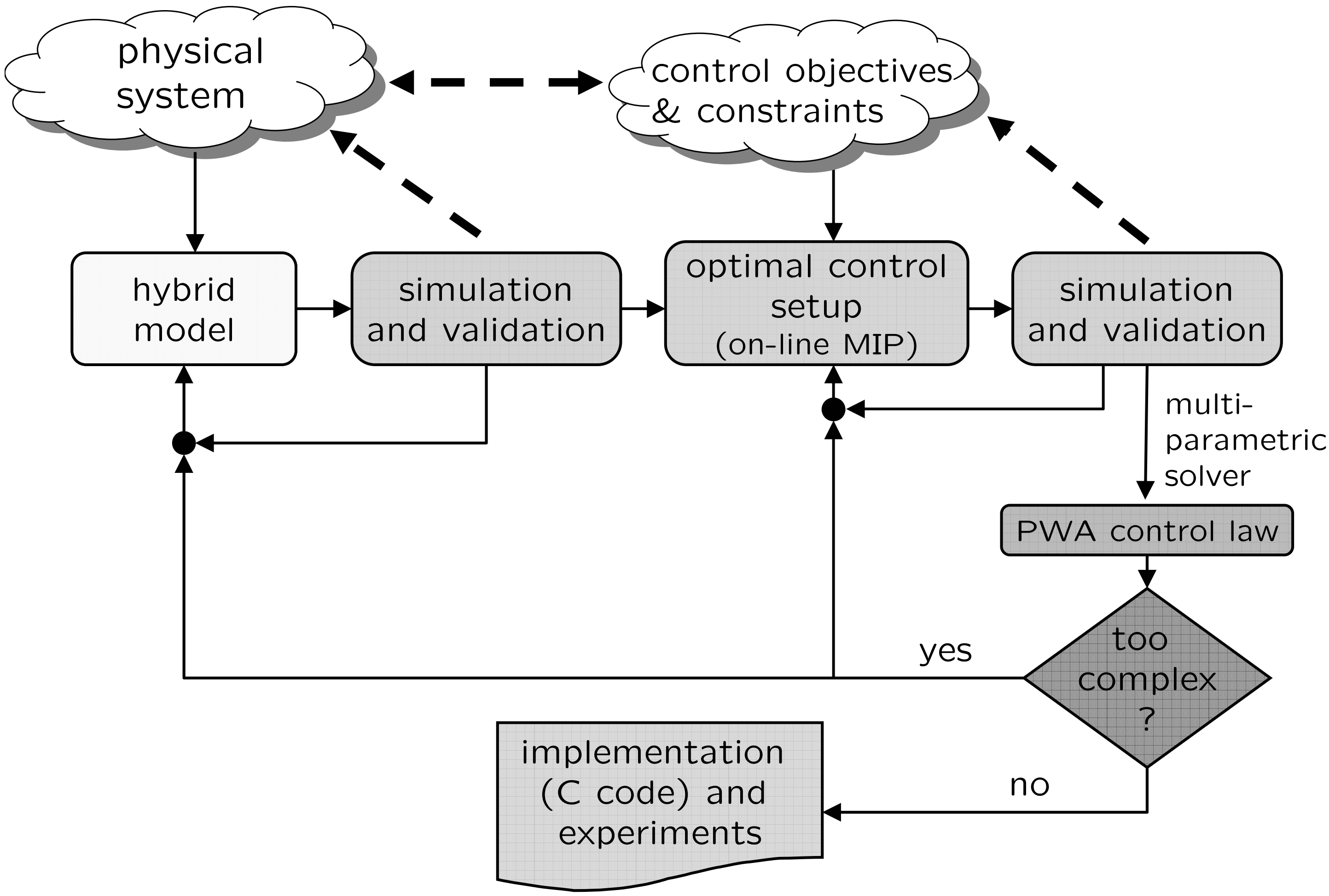


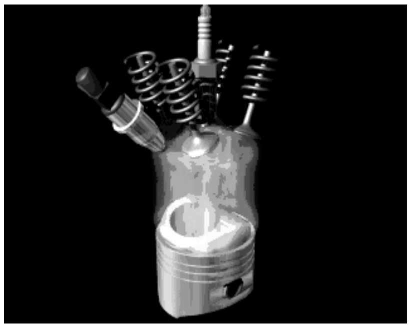
Generated C-code

```
#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,
```



Hybrid Control Design Flow





(Photo: Courtesy Mitsubishi)

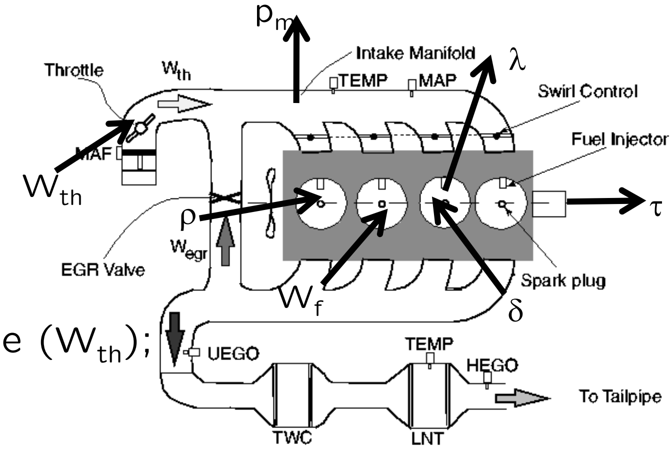
Application of Explicit Hybrid MPC to DISC Engine Control

(joint work with N. Giorgetti, I. Kolmanovsky, D. Hrovat)

FORD RESEARCH LABORATORY

DISC Engine

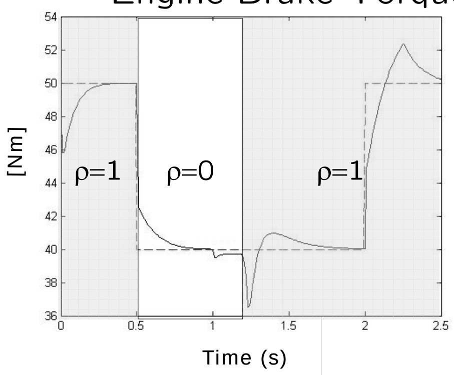
- States/Controlled outputs:
  - Intake manifold pressure (p_m);
  - Air-to-fuel ratio (λ);
  - Engine brake torque (τ);
- Inputs (continuous):
  - Air Mass flow rate through throttle (W_{th});
  - Mass flow rate of fuel (W_f);
  - Spark timing (δ);
- Inputs (binary):
  - ρ = regime of combustion (homogeneous/stratified);
- Constraints on:
  - Air-to-Fuel ratio (due to engine roughness, misfiring, smoke emiss.)
  - Spark timing (to avoid excessive engine roughness)
  - Mass flow rate on intake manifold (constraints on throttle)



→ Dynamic equations are **nonlinear**
Dynamics and constraints **depend on regime** ρ !

Explicit MPC Controller (quadratic costs)

Engine Brake Torque



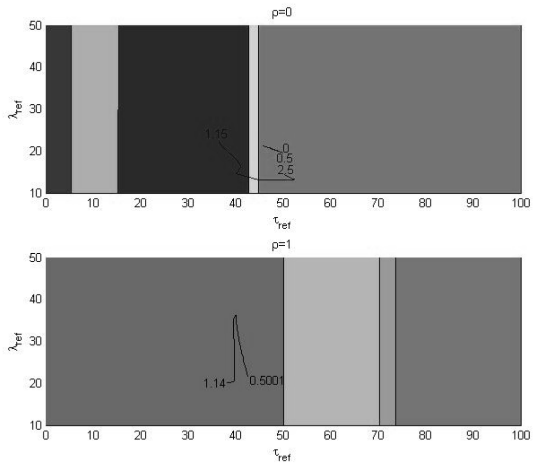
On-line simulations:

- Control horizon N=1;
- Simulation time T=2.5 s;
- Sampling time Ts=10 ms;
- CPU time ≃ 8 s;
  (on an Intel Centrino 1.2 GHz, 640 Mb RAM with Cplex 9.01)

→ ≃ 32 ms per time step

Not directly implementable !

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



Explicit simulations:

- Control horizon N=1;
- 9 Parameters;
- CPU time ≃ 0.52 s

→ **75** Partitions

≃ 2 ms per time step

Implementable !

Conclusions

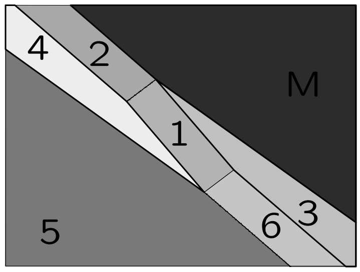
- **Discrete hybrid automata** are simple yet versatile models of hybrid systems, and lead immediately computationally-useful models

- **Optimization-based control** handles performance specs and constraints in a natural and direct way. Quite complex hybrid systems can be controlled using on-line mixed-integer programming

- **Piecewise affine MPC controllers** can be synthesized, off-line, and implemented as look-up tables of linear gains

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

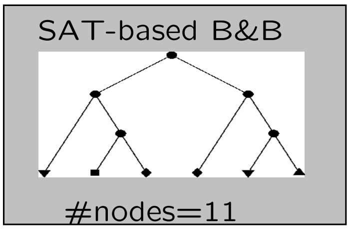
(x,r)-space

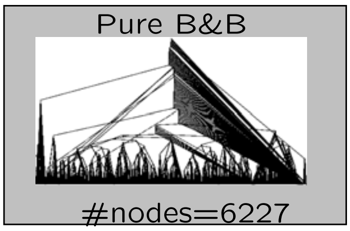


- **Hybrid Toolbox for Matlab** available to assist controller design: modeling, simulation, verification, MPC, code generation

Ongoing Research

- **Logic-based methods:** use combined SAT+LP to largely improve computation performance of optimal control/verification





(N. Giorgetti)

- **Stochastic hybrid models:** (robust) optimal control can be solved using mixed-integer programming (Di Cairano, HSCC05)

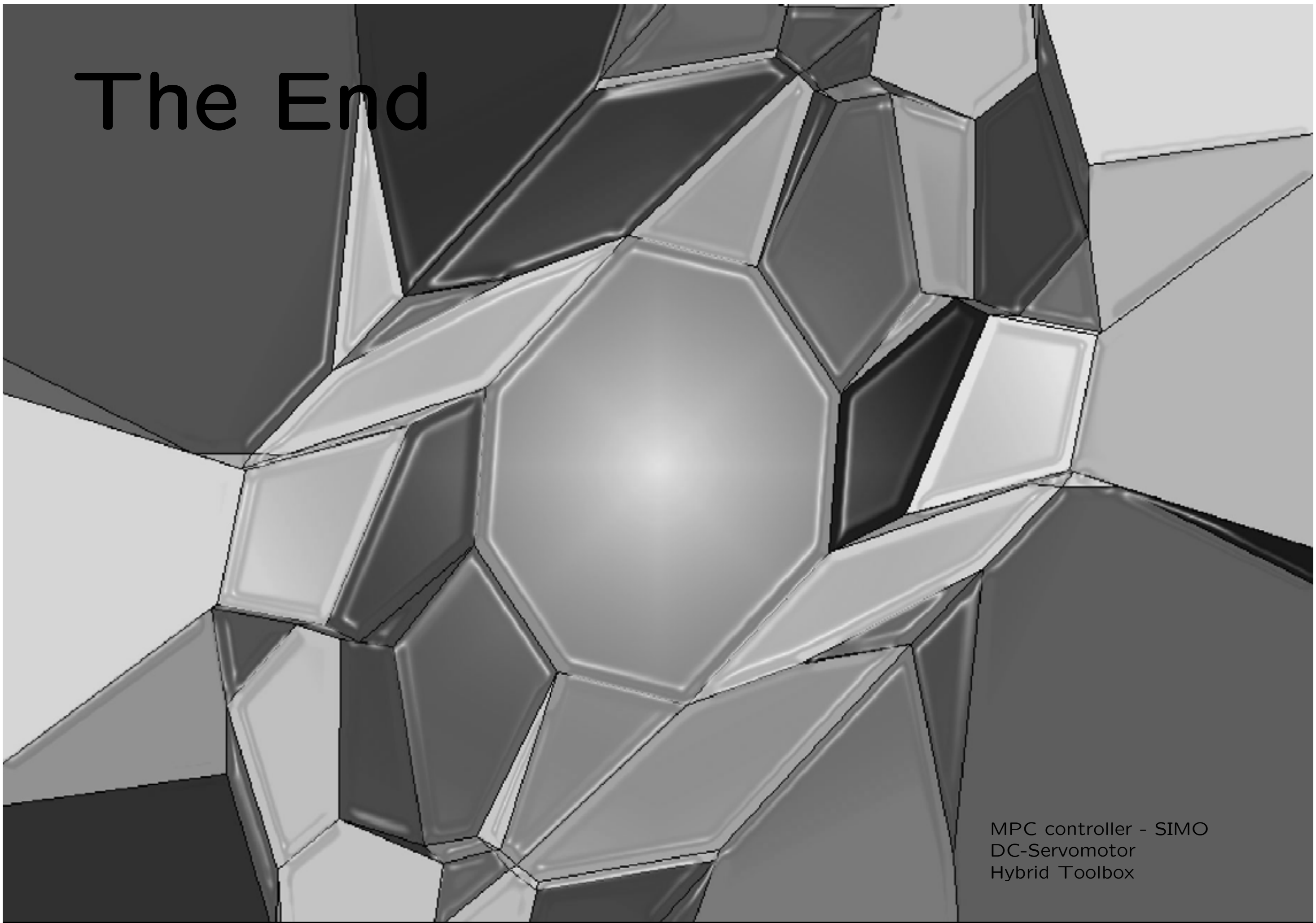
- **Event-based optimal control of hybrid systems:** index k denotes events occurring in piecewise-affine continuous-time systems, rather than discrete time (Di Cairano, Julvez)

optimal control train-gate system

N	t_s	t_{cpu} (sec.)	t_{viol} (time units)	$t_{viol}^{(sup)}$ (time units)
5	20.00	0.45	3.82	40.00
6	16.66	1.46	21.83	33.32
10	10.00	8.58	9.82	20.00
15	6.66	66.77	6.55	13.32
20	5.00	747.14	4.82	10.00
	cont.time	1.09	—	—

The European Commission Community Research ist information society technologies

The End



MPC controller - SIMO
DC-Servomotor
Hybrid Toolbox