Recent Advances in Applied Model Predictive Control

Alberto Bemporad

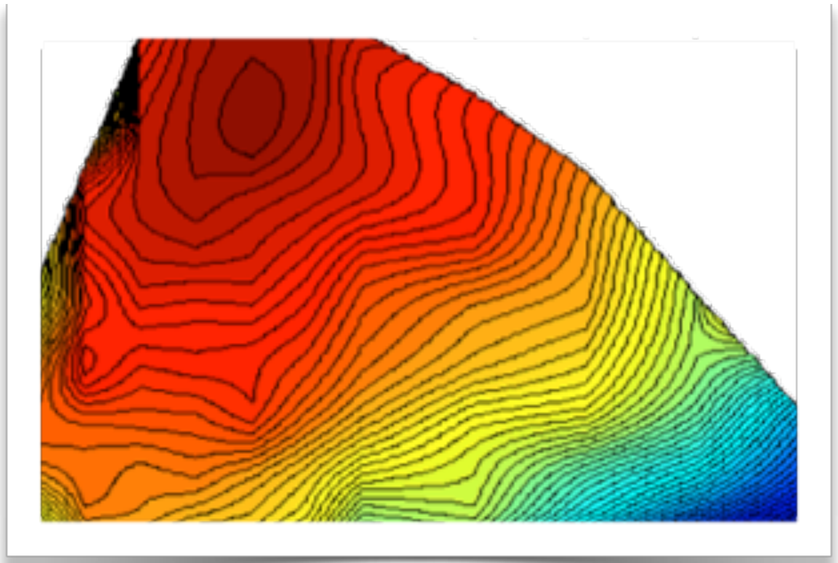
http://cse.lab.imtlucca.it/~bemporad

Institutions Markets Technologies
IMT
INSTITUTE FOR ADVANCED STUDIES LUCCA

DYSCO Research Unit

Dynamical Systems, Control, and Optimization

Typical control problems in industry

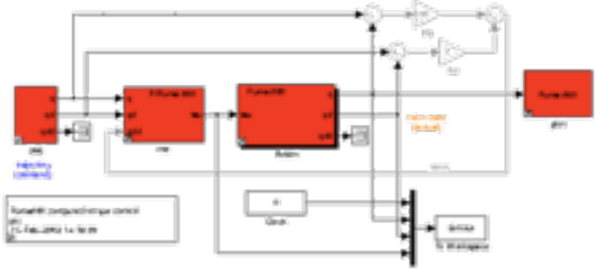


consumption map

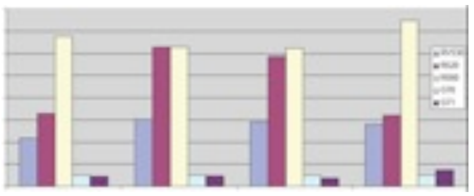
homogeneous stratified

- Dynamics often **multi-variable, nonlinear, switching, time-varying, …**
- Several **constraints** on manipulated variables and other variables (torque, voltage, speed, position, …)
- **Optimal performance** sought (x% less energy consumption, for example)
- Often characterized by **fast dynamics** (sample time of milliseconds)

A “wish list” for control design



- **Model-based** (linear, nonlinear, switching dynamics)



- Easy & direct way of defining **performance** objectives and **constraints** on variables

- Easy to design, tune, and **maintain** (avoid “spaghetti” designs !)

Motorola MPC 555

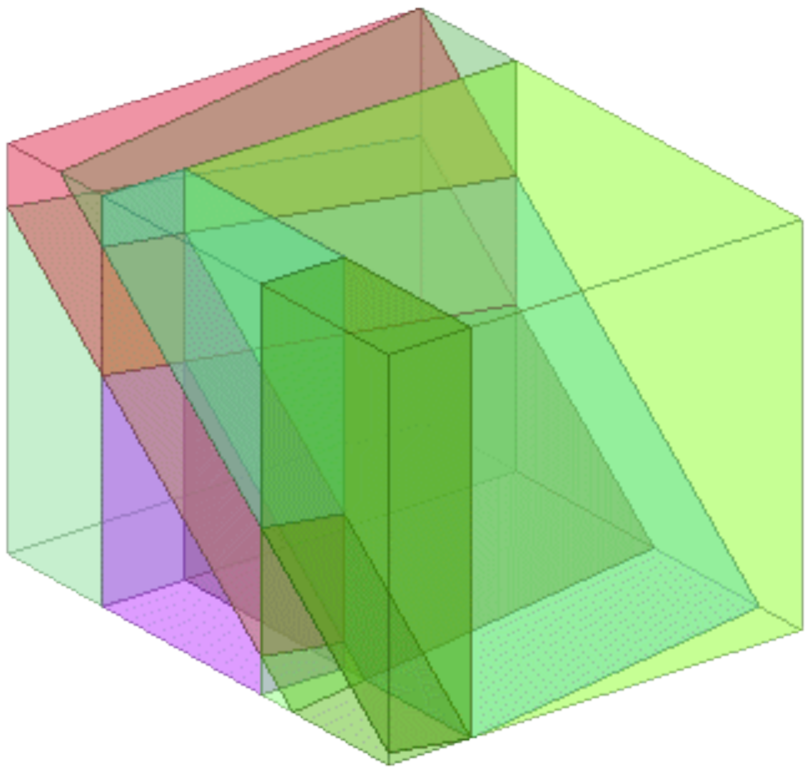
- Suitable for **fast sampling** implementation (1-10 ms) on **simple μ-controllers** with **limited memory** (kb)

- Solid theoretical certification (stability guarantees)

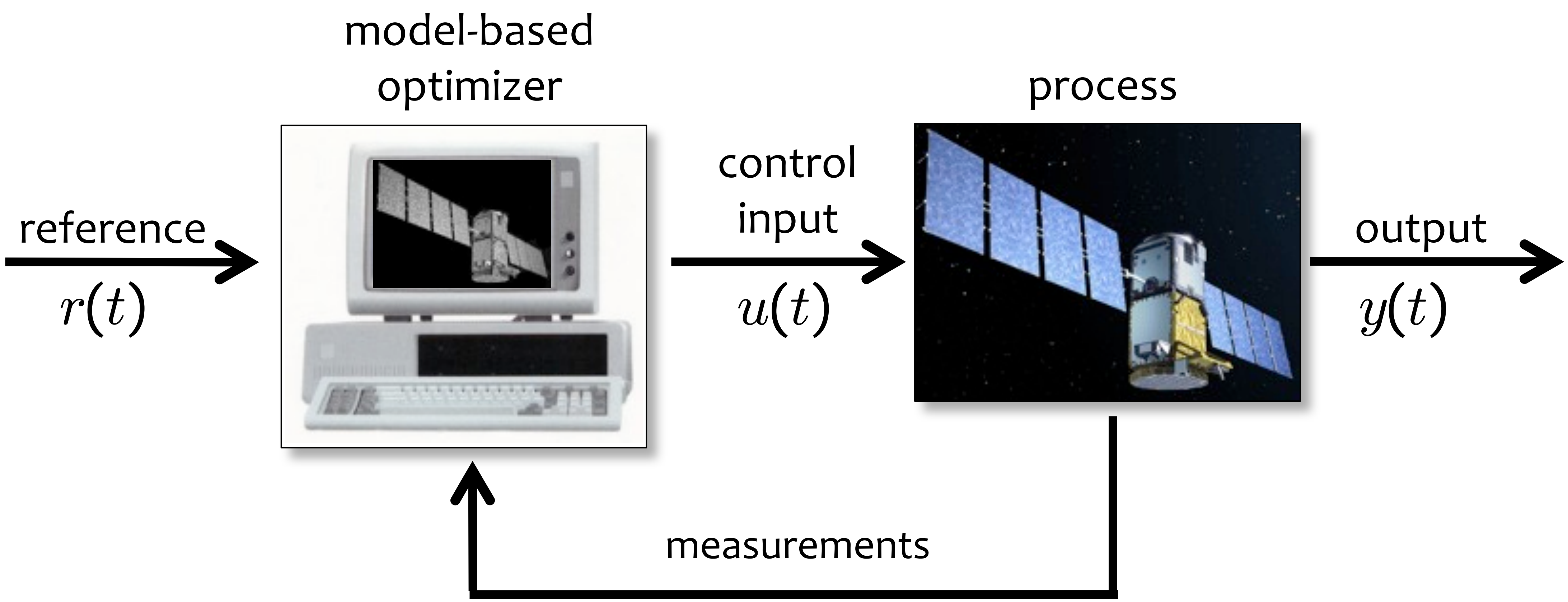
Model Predictive Control

Outline

- Basic ideas of Model Predictive Control (MPC)
- MPC applications (mainly automotive & aerospace)
- Conclusions



Model Predictive Control (MPC)

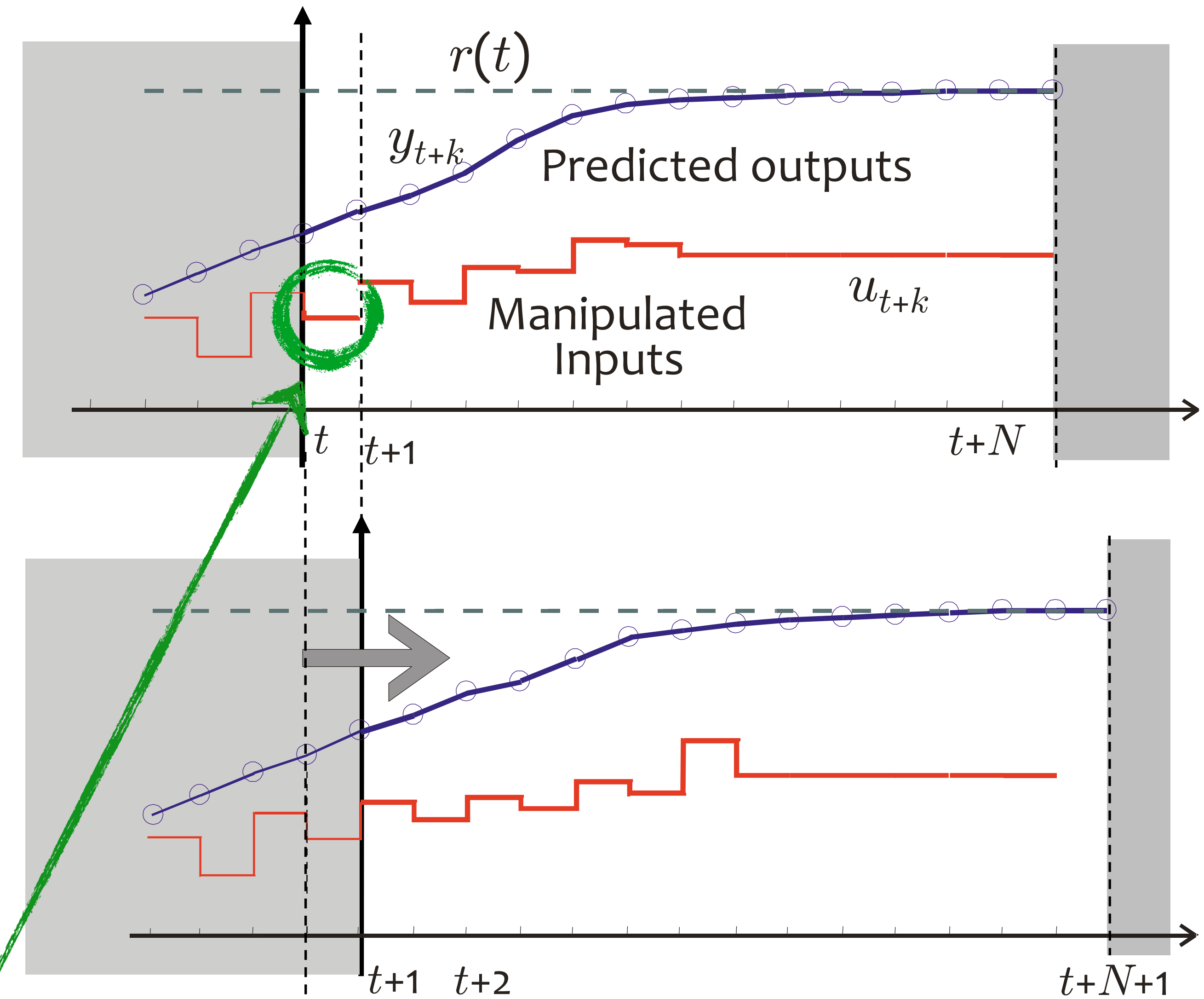


*Use a dynamical **model** of the process to **predict** its future evolution and choose the "best" **control** action*

MPC algorithm

- **At time** t, solve an **optimal control** problem over a future horizon of N steps

$$
\begin{aligned}
\min \quad & \sum_{k=0}^{N-1} \ell(y_{t+k} - r_{t+k}, u_{t+k}) \\
\text{s.t.} \quad & x_{t+k+1} = f(x_{t+k}, u_{t+k}) \\
& y_{t+k} = g(x_{t+k}, u_{t+k}) \\
& \text{constraints on } u_{t+k}, x_{t+k}, y_{t+k} \\
& x_t = x(t)
\end{aligned}
$$

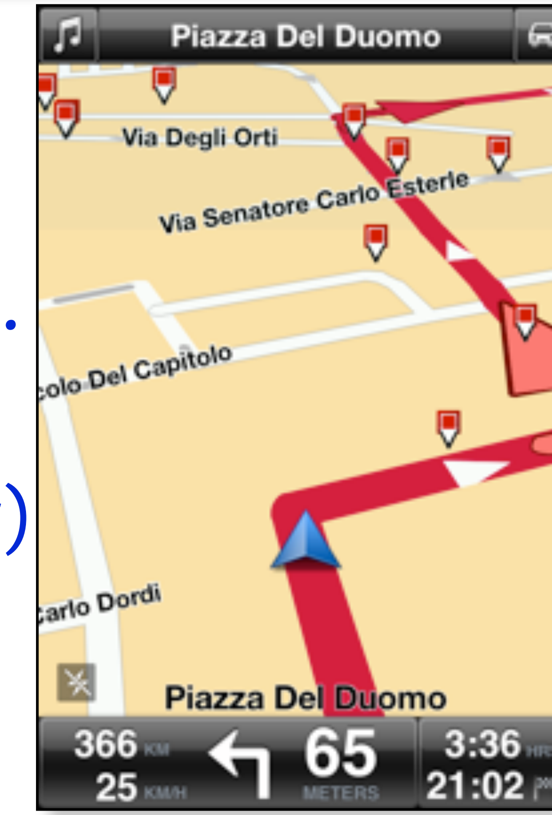


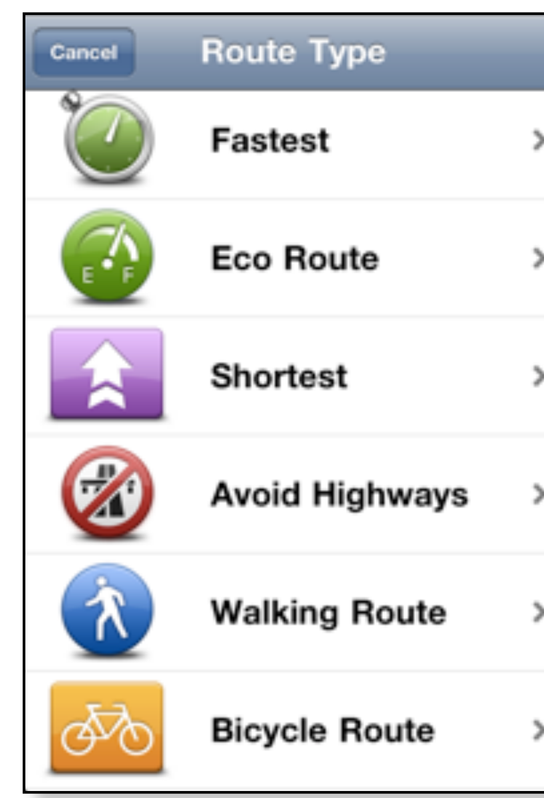
- Apply only the first optimal move $u^*(t)$, trash the rest of the optimal sequence
- **At time** t**+1:** Get new measurements, repeat the optimization. And so on ...

MPC transforms open-loop optimal control into a **feedback** control law

Receding horizon planning: GPS example

- **prediction model** how vehicle moves on map
- **constraints** drive on roads, respect one-way roads, etc.
- **disturbances** (driver does not follow directions properly)
- **set point** desired location
- **cost function** minimum time, minimum distance, etc.
- **receding horizon mechanism**
  (event-based: optimal route re-planned when path is lost)





x = GPS position

u = navigation commands

MPC of linear systems

linear model
$$\begin{cases} x_{k+1} = Ax_k + Bu_k \\ y_k = Cx_k \end{cases}$$

$R = R' \succ 0$
$Q = Q' \succeq 0$
$P = P' \succeq 0$

performance index
$$\min_U x_N' P x_N + \sum_{k=0}^{N-1} x_k' Q x_k + u_k' R u_k$$

$x_0 = x(t)$

$$\begin{array}{ll} \min_U & \frac{1}{2}U'HU + x'(t)F'U + \cancel{\frac{1}{2}x'(t)Yx(t)} \\ \text{s.t.} & GU \leq W + Sx(t) \end{array}$$

$$U = \begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ u_{N-1} \end{bmatrix}$$

constraints
$$\begin{cases} u_{\min} \leq u_k \leq u_{\max} \\ y_{\min} \leq Cx_k \leq y_{\max} \end{cases}$$

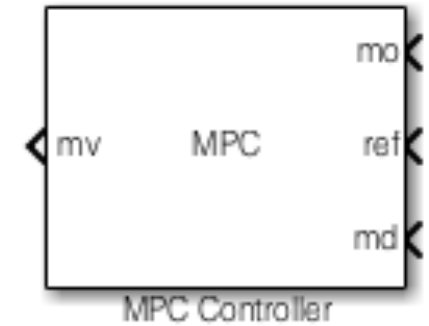
MPC implemented by solving a **(convex) Quadratic Program (QP)**

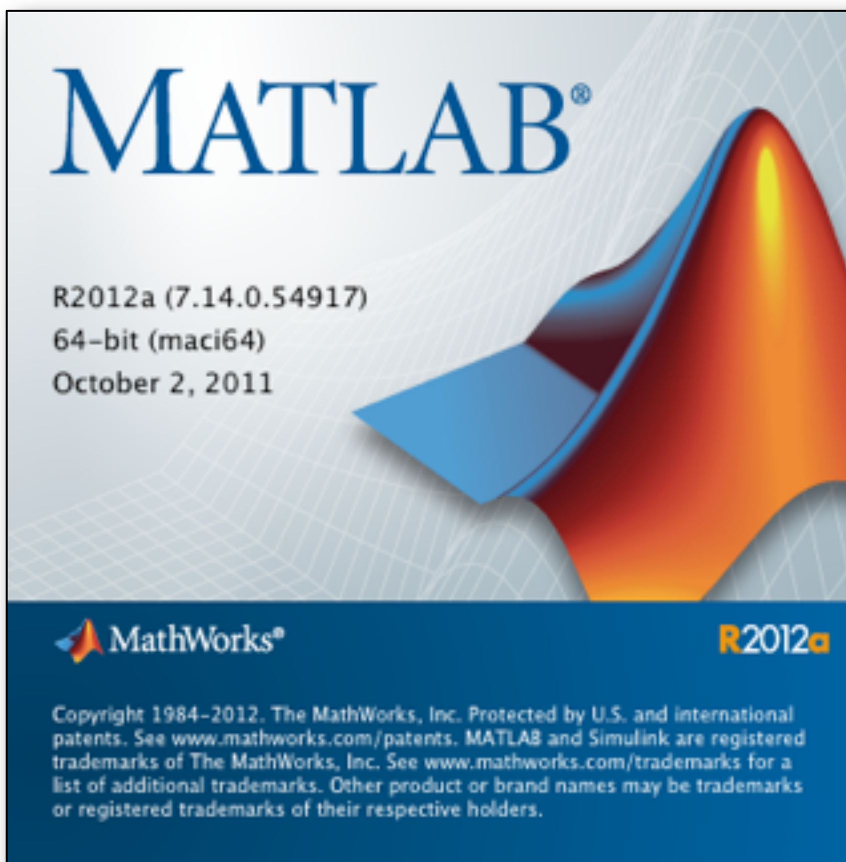
Routinely used in the process industries

Model Predictive Control Toolbox

(Bemporad, Ricker, Morari, 1998-2012)

- **MPC Toolbox 4.0** (The Mathworks, Inc.)
  - MPC Simulink Library
  - Easy design (MATLAB objects, MPC GUI, Tuning Advisor)
  - Code generation [RTW, xPC Target, dSpace, etc.]
  - Linked to OPC Toolbox, System ID Toolbox, …
  - New QP solver (QPKWIK algorithm)





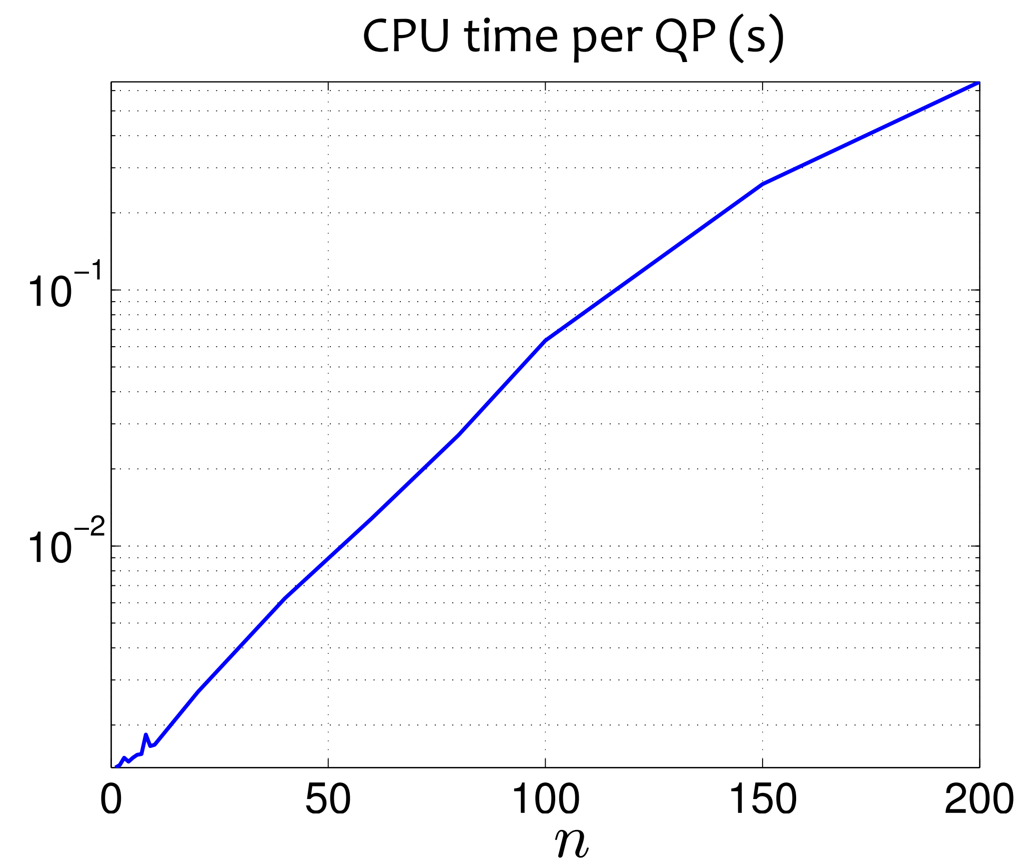
Complete solution for linear MPC design

http://www.mathworks.com/products/mpc/

Note: MPC Toolbox 3.0 **most successful webinar** in 2009 !

Numerical complexity of linear MPC - An example

- Linear MPC of random square MIMO systems
  - n outputs, n inputs, $3n$ states
  - prediction horizon $N=10$, control horizon $m=2$
  - constraints: $-1 \leq u_k \leq 1, \ -1 \leq y_k \leq 1$
  - QP size: $(mn+1)$ variables, $(2Nn+2mn)$ constraints



n	#vars	# constraints	CPU time (s)
1	3	24	0.00136
5	11	120	0.00149
20	41	480	0.00270
100	201	2400	0.06432
150	301	3600	0.25873
200	401	4800	0.64981

Macbook Air 2.13 GHz (this mac !)

Inter Core 2 Duo 4GB RAM

MPC Toolbox 4.0, MATLAB R2011b

New **active set QP** in **EML** (dense matrices)

Pros and cons of on-line optimization

PROS

✓ Continuously update the best decision, reacting to unexpected events (disturbances, faults, obstacles,...)

✓ Excellent LP/QP/MIP/NLP solvers exist today
(“LP is a technology” – S. Boyd)

CONS

✖ Computation time may be too long for large-problems/fast sampling

✖ Requires relatively expensive hardware (such as a microprocessor)

✖ Software complexity: solver code must be embedded in the control code

✖ Real-time: worst-case CPU time often hard to estimate

Explicit model predictive control

$$
\begin{array}{ll}
\min\limits_{U} & \frac{1}{2}U'HU + \mathbf{x}'(t)F'U + \cancel{\frac{1}{2}x'(t)Yx(t)} \\
\text{subj. to} & GU \leq W + S\mathbf{x}(t)
\end{array}
$$

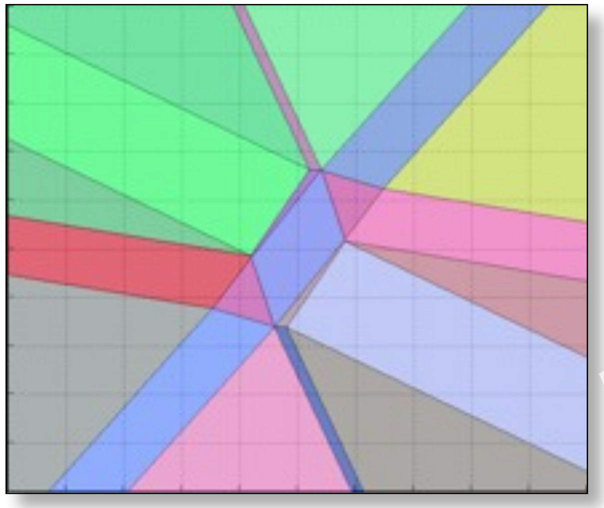
Idea: solve the QP **for all** $x(t)$ within a given range of $\mathbb{R}^n$ **off-line**

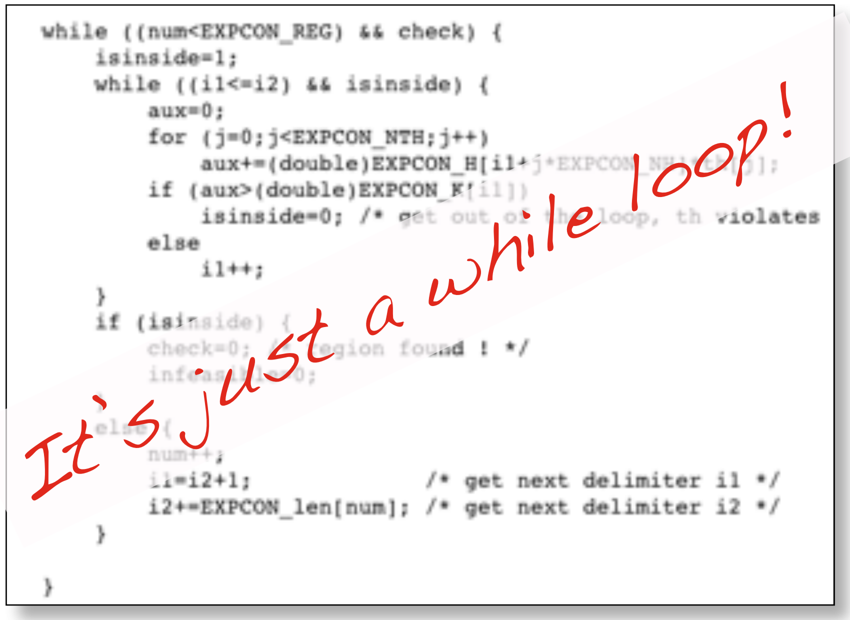
→ multi-parametric programming problem

Linear MPC is a **continuous and piecewise affine** control law !

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

(Bemporad, Morari, *et al.*, 2002)





```
while ((num<EXPCON_REG) && check) {
    isinside=1;
    while ((i1<=i2) && isinside) {
        aux=0;
        for (j=0;j<EXPCON_NTH;j++)
            aux+=(double)EXPCON_H[i1+j*EXPCON_NH]*th[j]];
        if (aux>(double)EXPCON_K[i1])
            isinside=0; /* get out of the loop, th violates
        else
            i1++;
    }
    if (isinside) {
        check=0; /* region found ! */
        infeasible=0;
    }
    else{
        num++;
        i1=i2+1;              /* get next delimiter i1 */
        i2+=EXPCON_len[num]; /* get next delimiter i2 */
    }
}
```

It's just a while loop!

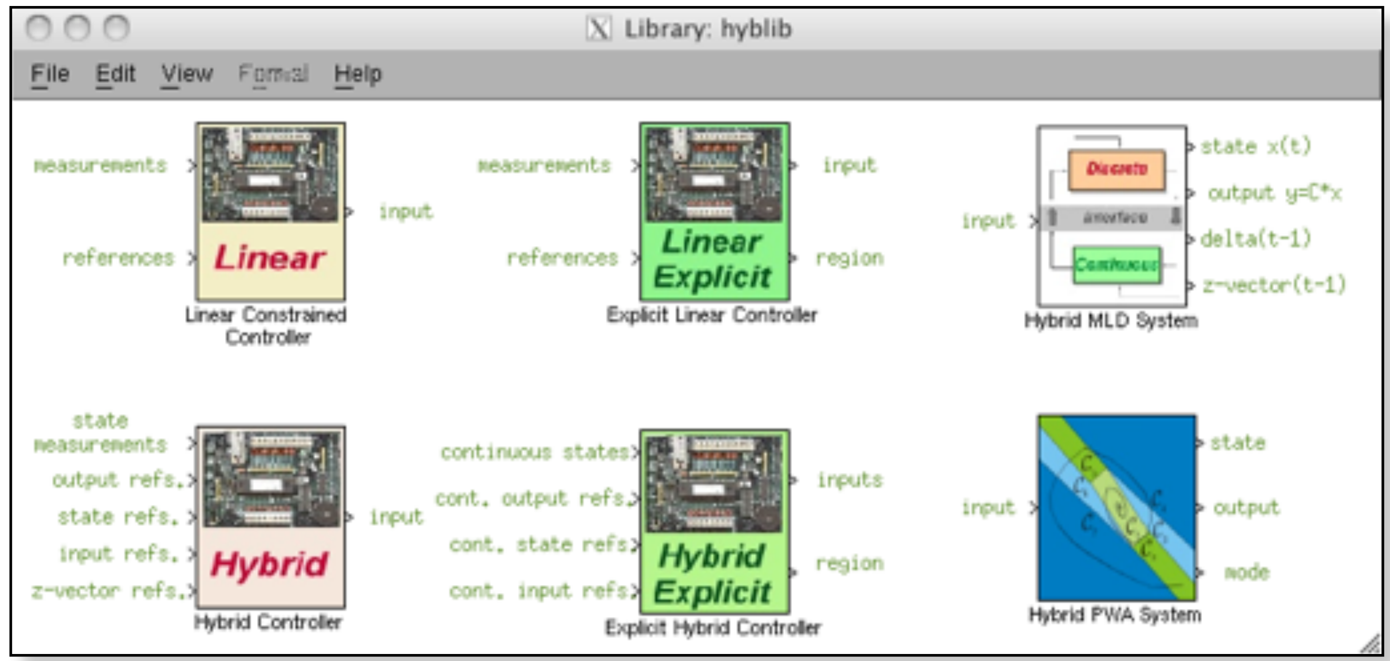
Hybrid Toolbox for MATLAB

(Bemporad, 2003-2012)

Supported by

Features:

- **Explicit** MPC control (via multi-parametric programming)
- Simulink library
- C-code generation

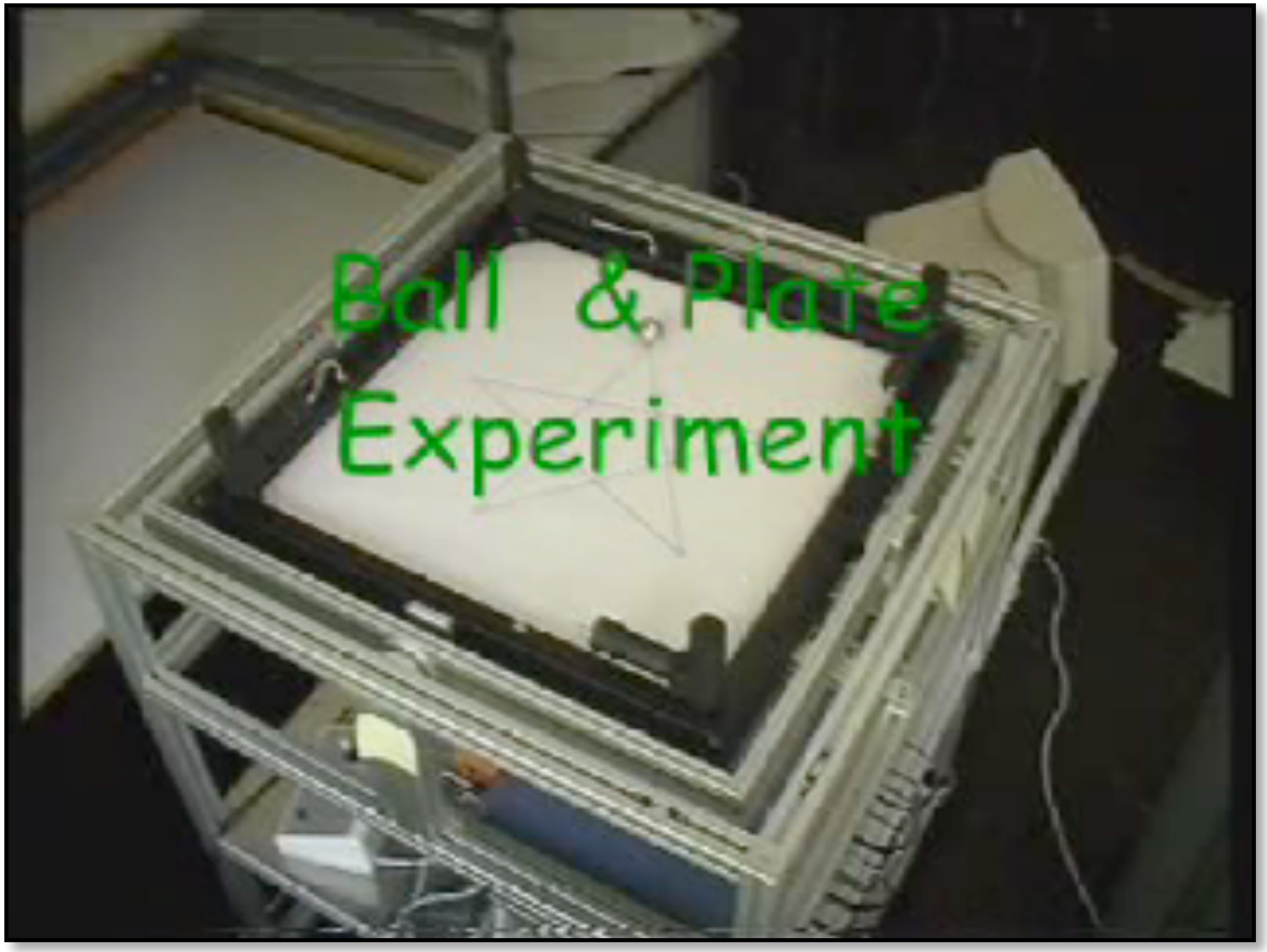


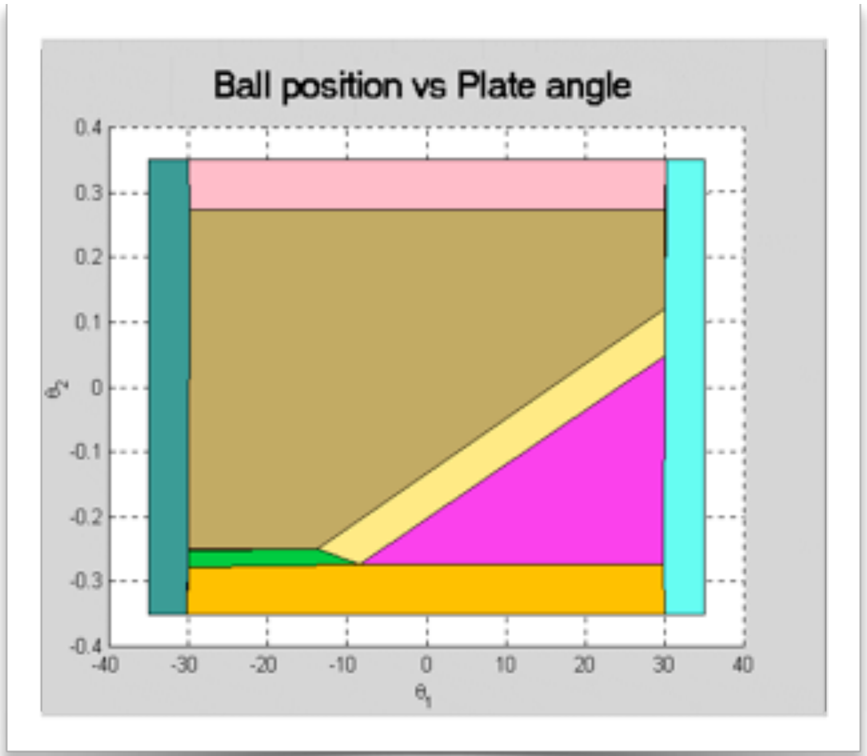
- **Hybrid** models: design, simulation, verification
- Control design for linear systems w/ constraints and hybrid systems
- Interfaces to several QP/LP and Mixed-Integer Programming solvers

4000 download requests since October 2004

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

Explicit MPC of a ball & plate system





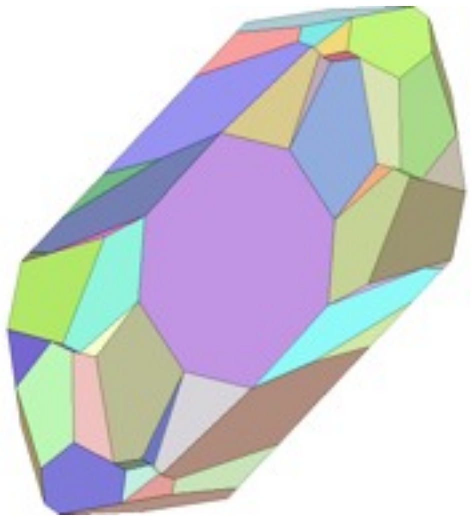
Two explicit MPC controllers

x-axis = 22 regions,
y-axis = 23 regions

sample time = 20ms

Complexity of explicit MPC

- Number of regions depends on number of possible combinations of active constraints

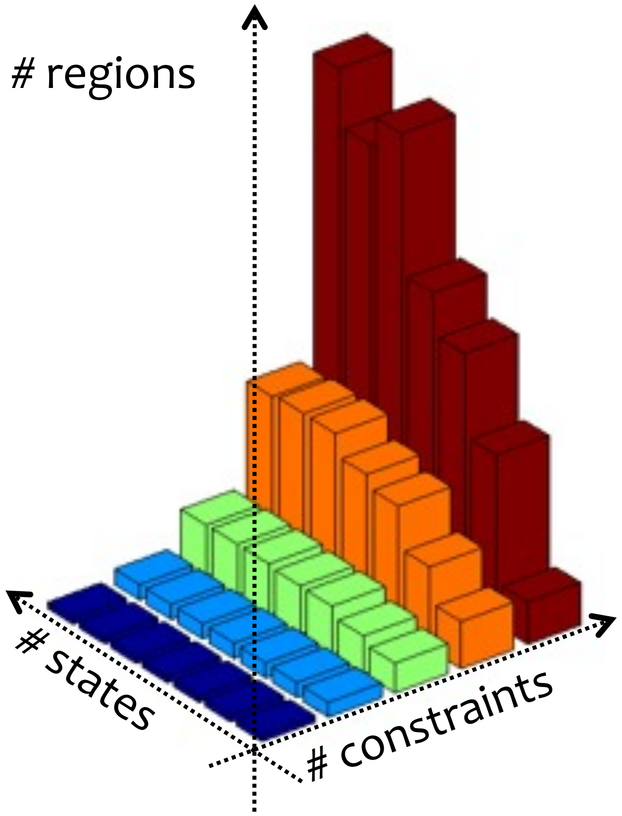


- Weak dependence on number of states and references

- On-line QP vs explicit MPC comparison:

$2N$	QP (ms) average	worst	explicit (ms) average	worst	regions	[storage kb]
4	1.1	1.5	0.005	0.1	25	16
8	1.3	1.9	0.023	1.1	175	78
20	2.5	2.6	0.038	3.3	1767	811
30	5.3	7.2	0.069	4.4	5162	2465
40	10.9	13.0	0.239	15.6	11519	5598

(Intel Centrino 1.4 GHz)

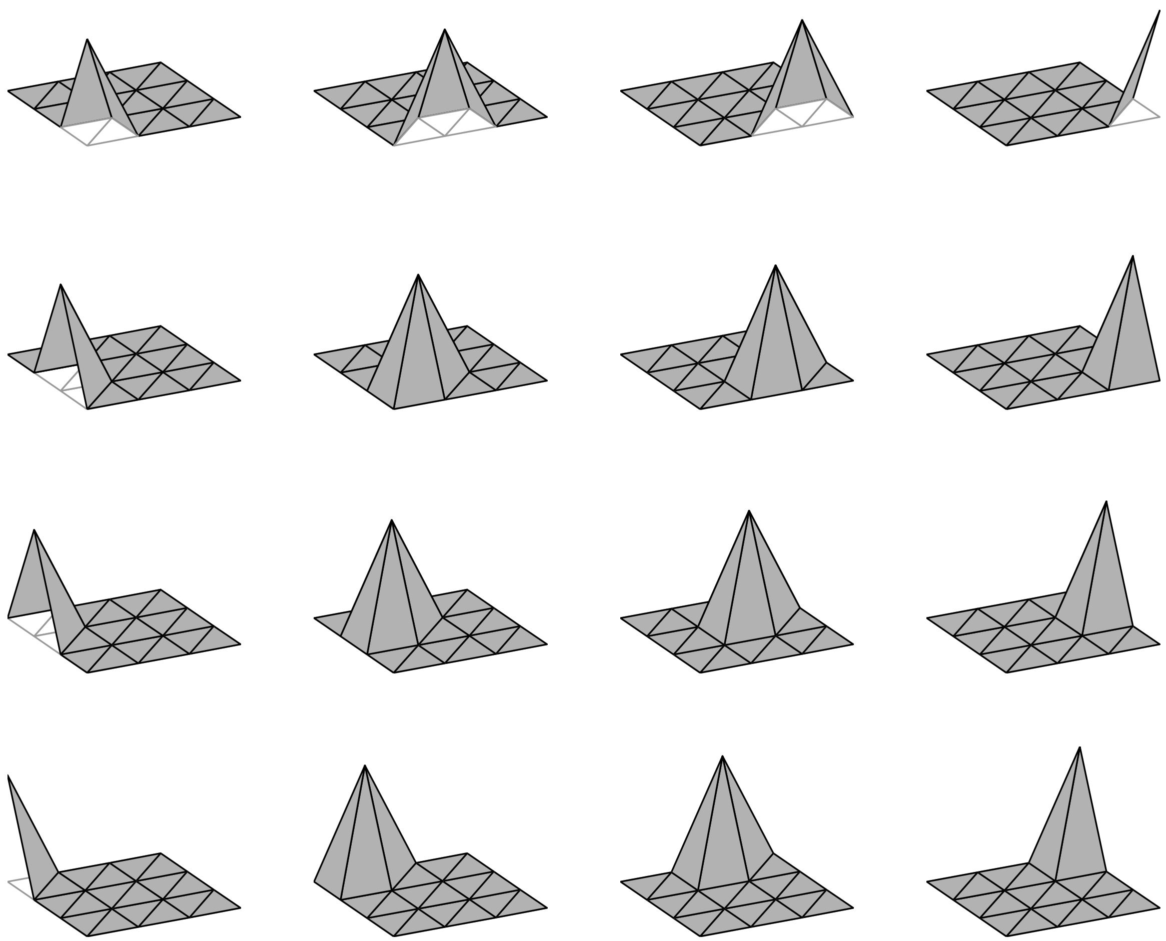


Explicit MPC typically limited to 6-8 free control moves and 8-12 states+references

Ultra fast & cheap MPC approximations

- ***Approximate*** a given linear MPC controller by using **canonical PWA** functions over **simplicial partitions** (**PWAS**)

(Bemporad, Oliveri, Poggi, Storace , IEEE TAC, 2011)



(Julian, Desages, Agamennoni, 1999)

$$\hat{u}(x) = \sum_{k=1}^{N_v} w_k \, \phi_k(x) = w' \phi(x)$$

approximate MPC law

Weights w_k optimized **off-line**
to best approximate a given MPC law

European FP7-ICT project MOBY-DIC
"Model-based synthesis of digital electronic circuits for embedded control"

PWA approximation of MPC over simplices

- **Extremely cheap**: **PWAS functions can be directly implemented on FPGA**, or even **ASIC** (Application Specific Integrated Circuits)

- **Extremely fast** computations (**10-100 nanoseconds**)

fit criterion

Control	p	Latency (A - B) [ns]
L^2	7	170 - 31
	31	238 - 45
	63	272 - 46
L^∞	7	170 - 31
	31	238 - 45
	63	272 - 46

partitions per dimension

online time to compute MPC (Xilinx Spartan 3 FPGA)

Architecture A: mainly serial

Architecture B: fully parallel

Exact explicit MPC: 52 regions
383 ns (avg) - 486 ns (max)

MIMO system dynamics

$$x_{k+1} = \begin{bmatrix} 1.2 & 1 \\ 0 & 1.1 \end{bmatrix} x_k + \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix} u_k$$

$$y_k = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} x_k$$

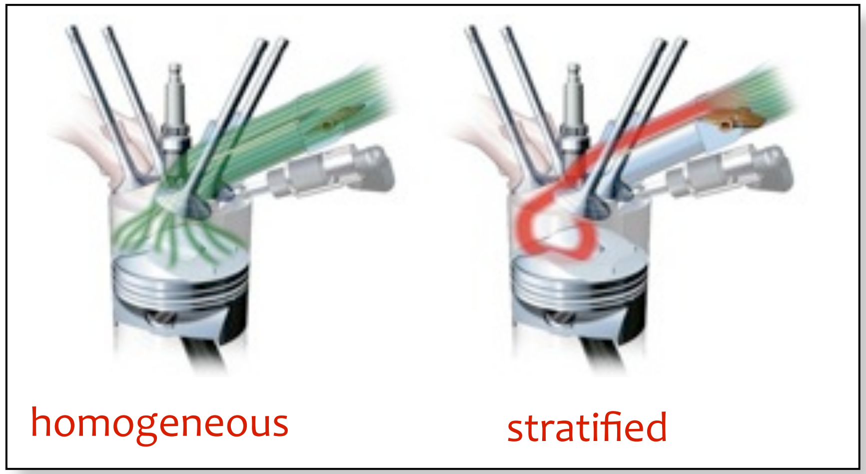
subject to saturation on u and y

- Certified closed-loop stability by constructing a **PWA Lyapunov function**

- Fulfillment of constraints on inputs (soft constraints on states)

Automotive applications of MPC

traction control



engine control

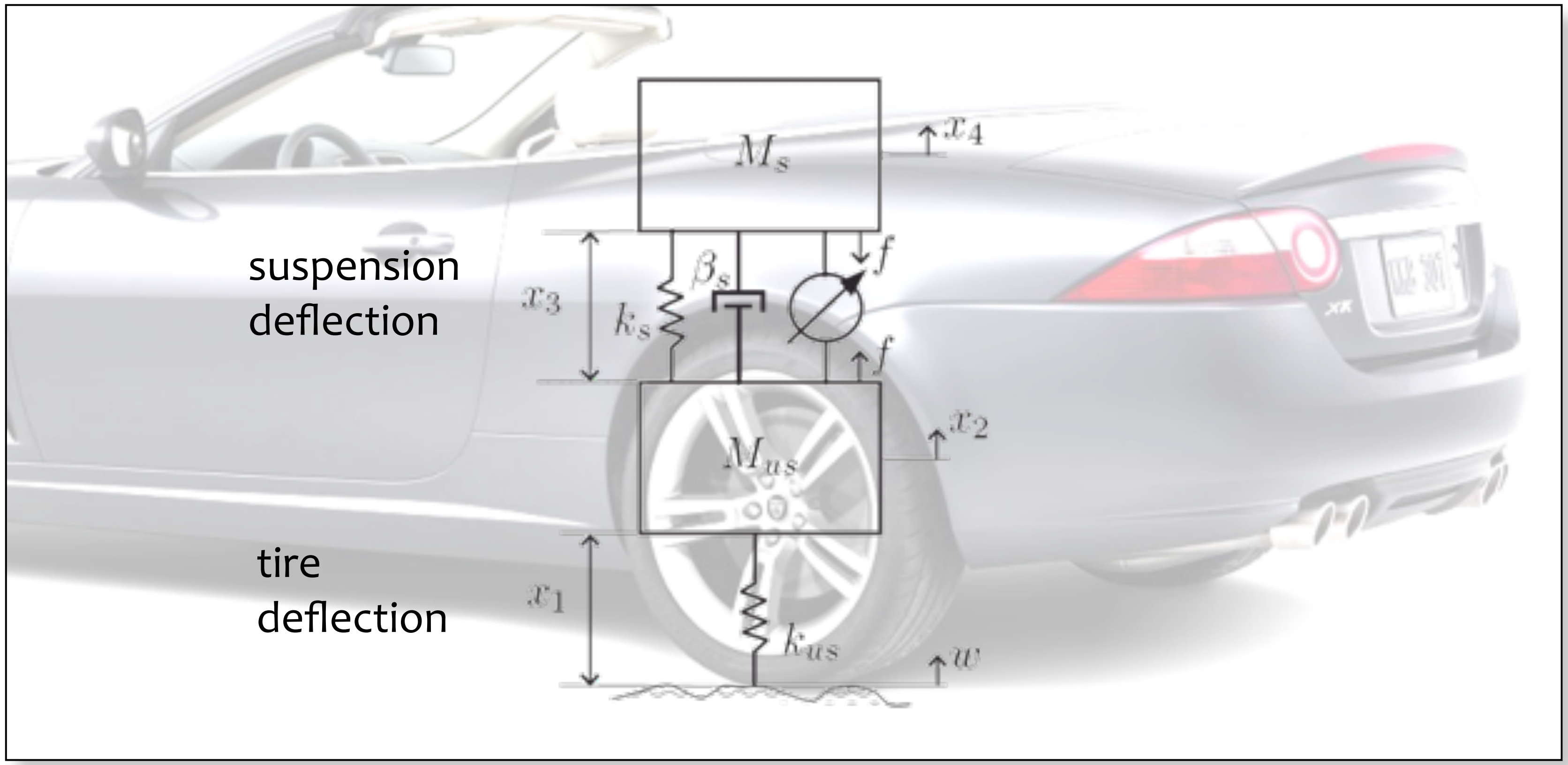
Ford

DENSO

CENTRO RICERCHE FIAT

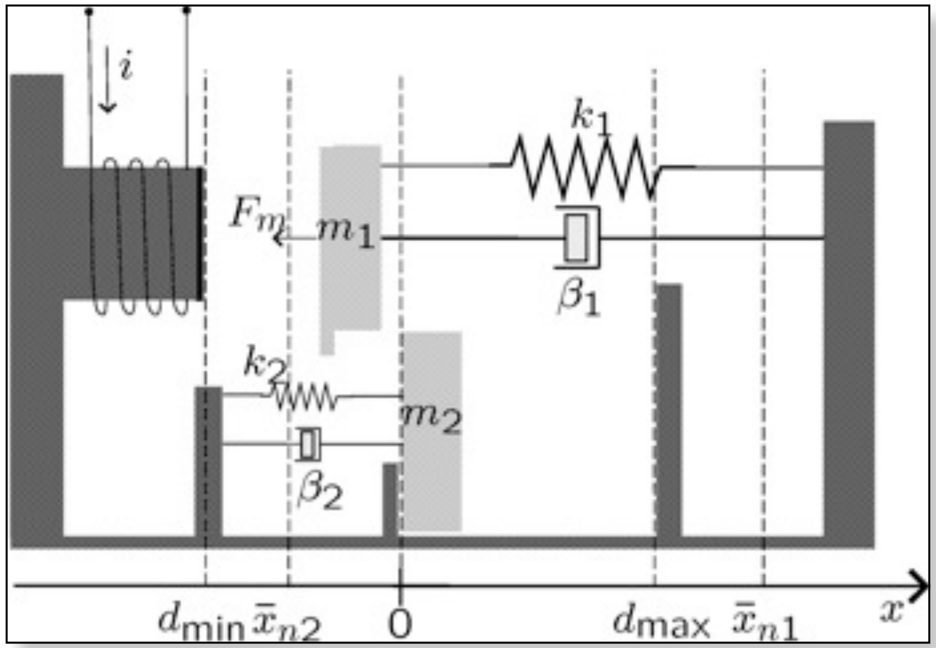
Institutions Markets Technologies IMT INSTITUTE FOR ADVANCED STUDIES LUCCA

idle speed control



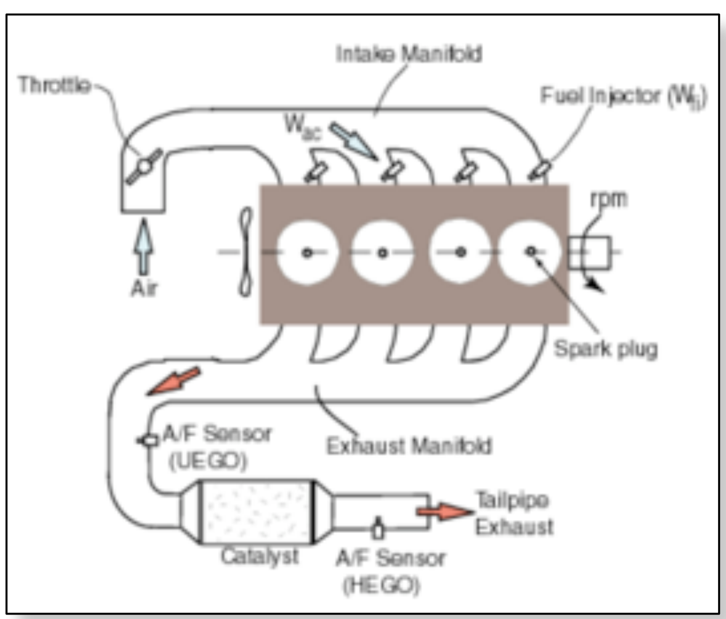
semiactive suspensions

Automotive applications of MPC



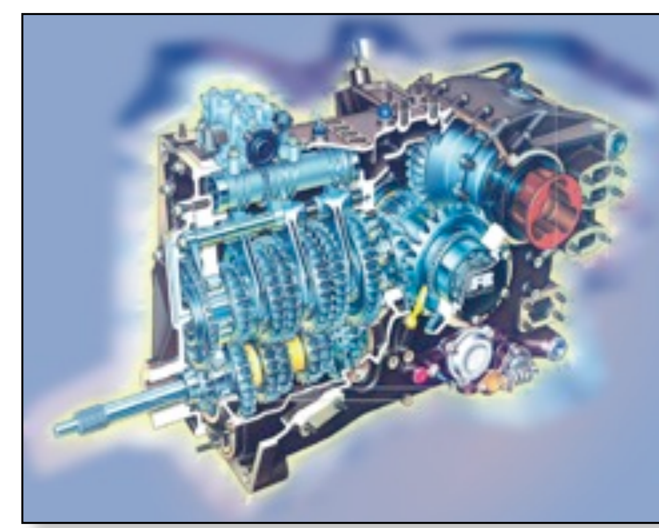
magnetic actuators

hybrid electric vehicles



air-to-fuel ratio

active steering



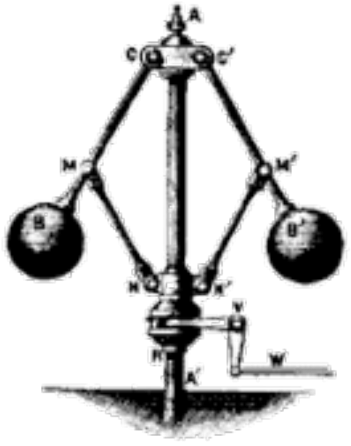
robotized gearbox

Explicit MPC for idle speed control

(Di Cairano, Yanakiev, Bemporad, Kolmanovsky, Hrovat, 2011)

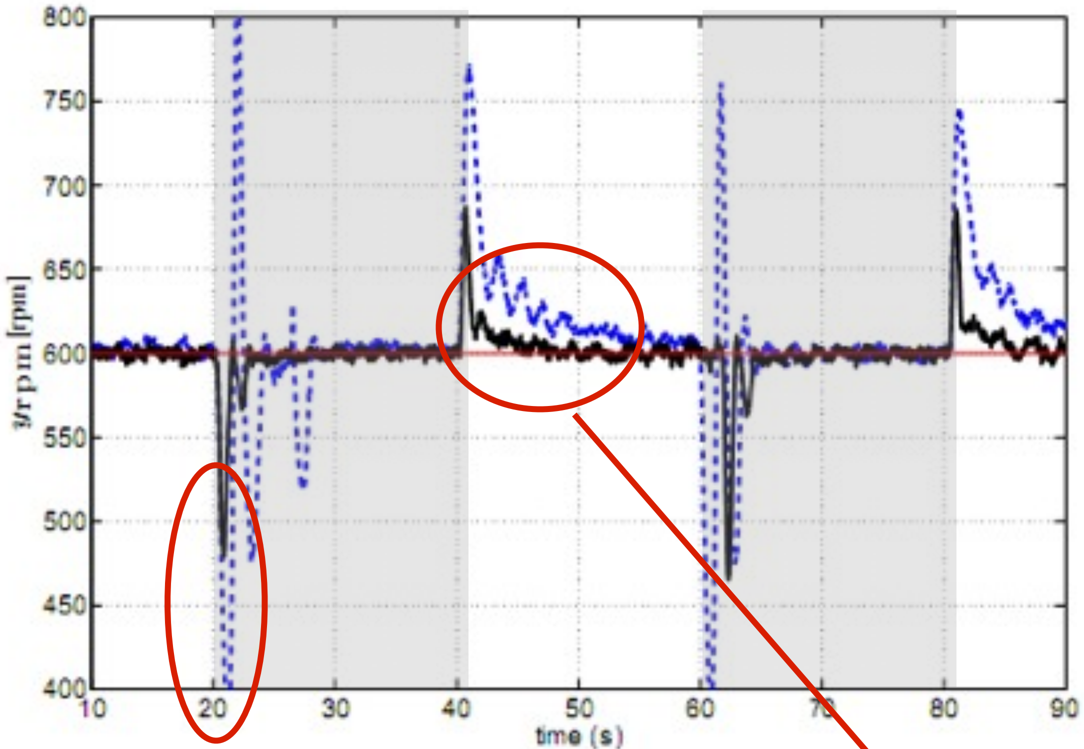
- Ford pickup truck, V8 4.6L gasoline engine
- Process:
  - *1 output* (engine speed) to regulate
  - *2 inputs* (airflow, spark advance)
  - input *delays*

- Objectives and specs:
  - **regulate engine speed** at constant rpm
  - **saturation limits** on airflow and spark
  - **lower bound** on engine speed ($\geq$450 rpm)

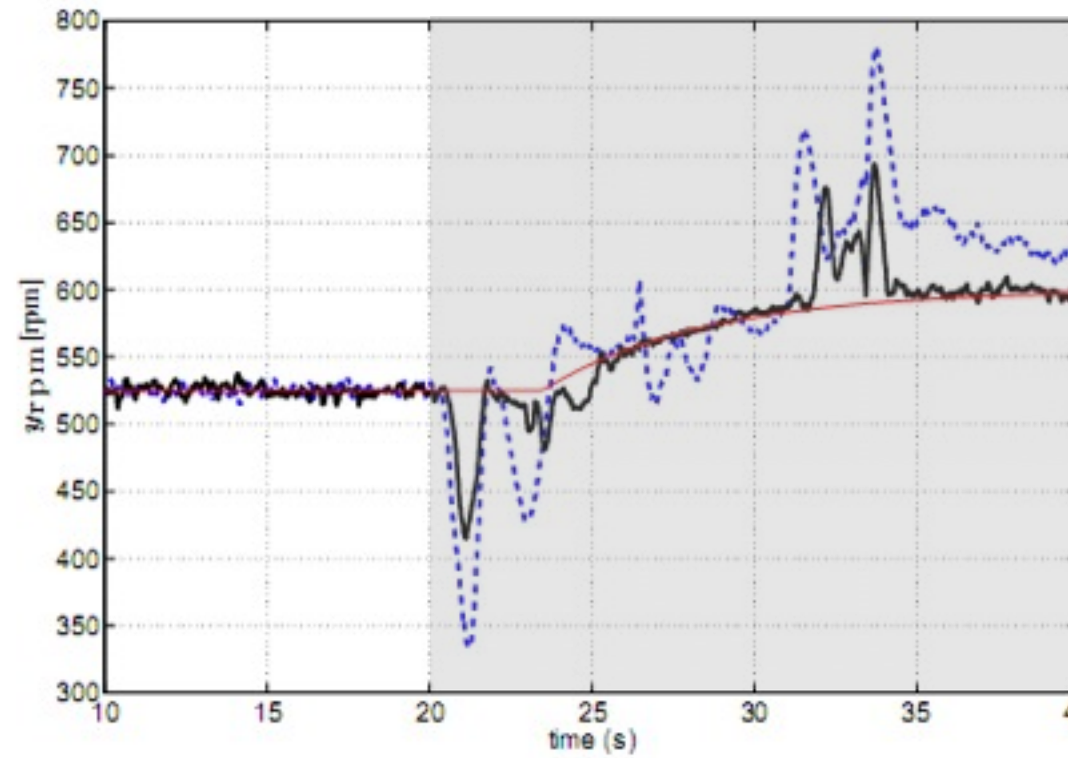


- Related to most classical problem in control: Watt's governor (1787)
- Problem suitable for MPC design (Hrovat, 1996)

Explicit MPC for idle speed - Experiments



Load torque (power steering)



Power steering + air conditioning

peak reduced by 50%

convergence 10s faster

```
#define EXPCON_NTH 2
#define EXPCON_NH 16
#define EXPCON_NF 5
#define EXPCON_NYM 2
static double EXPCON_F[]={
    -6.83553,-12.0296,0,0,-26.9062,-6
    -154.828};
```

PWA control law
(C-code)

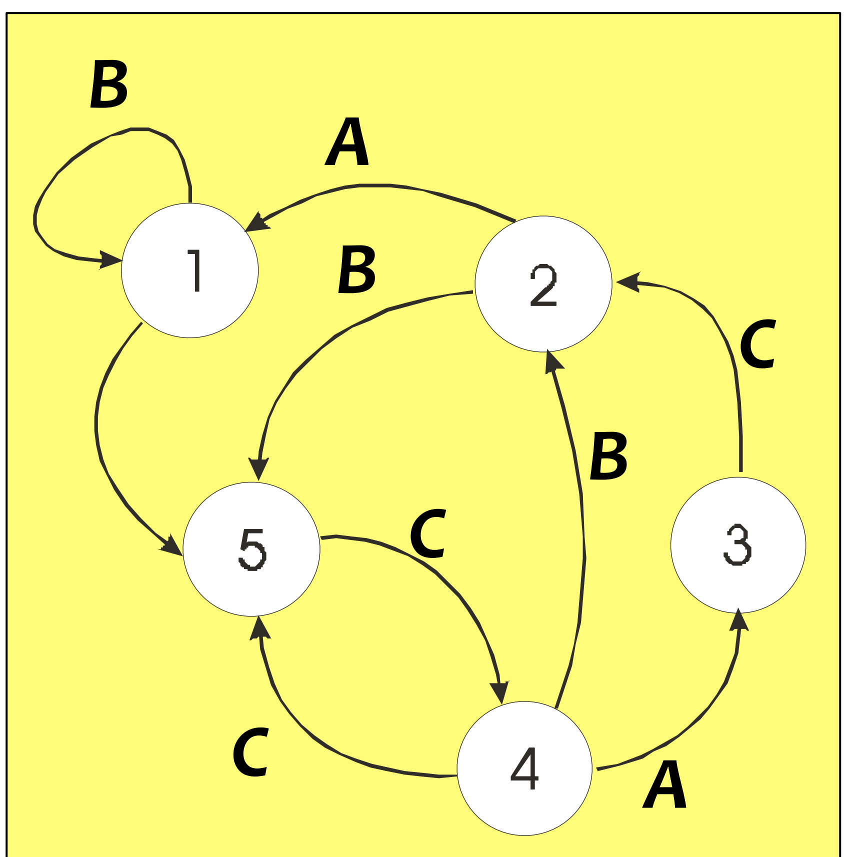
explicit MPC

baseline controller (linear)

set-point

Hybrid MPC

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



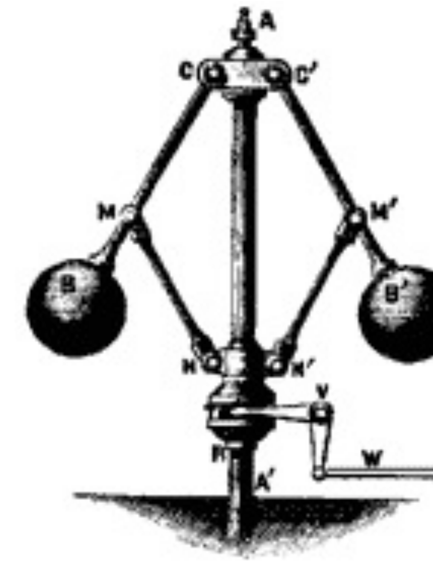
Computer Science

Finite state machines

Hybrid systems

Control Theory

Continuous dynamical systems



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

$u(t) \rightarrow$ system $\rightarrow y(t)$

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

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

Hybrid MPC

- Introduce "indicator variables" (or "events") in dynamical model / constraints

$$[\delta_k = 1] \quad \longleftrightarrow \quad [c(x_k, u_k) \leq 0]$$

$$\delta_k \in \{0, 1\}, \ x \in \mathbb{R}^n, \ u \in \mathbb{R}^m$$

(Glover 1975, Williams 1977, Hooker 2000)

"big-M" technique

$$\begin{cases} c(x_k, u_k) & \leq & M(1 - \delta_k) \\ c(x_k, u_k) & > & m\delta_k \end{cases}$$

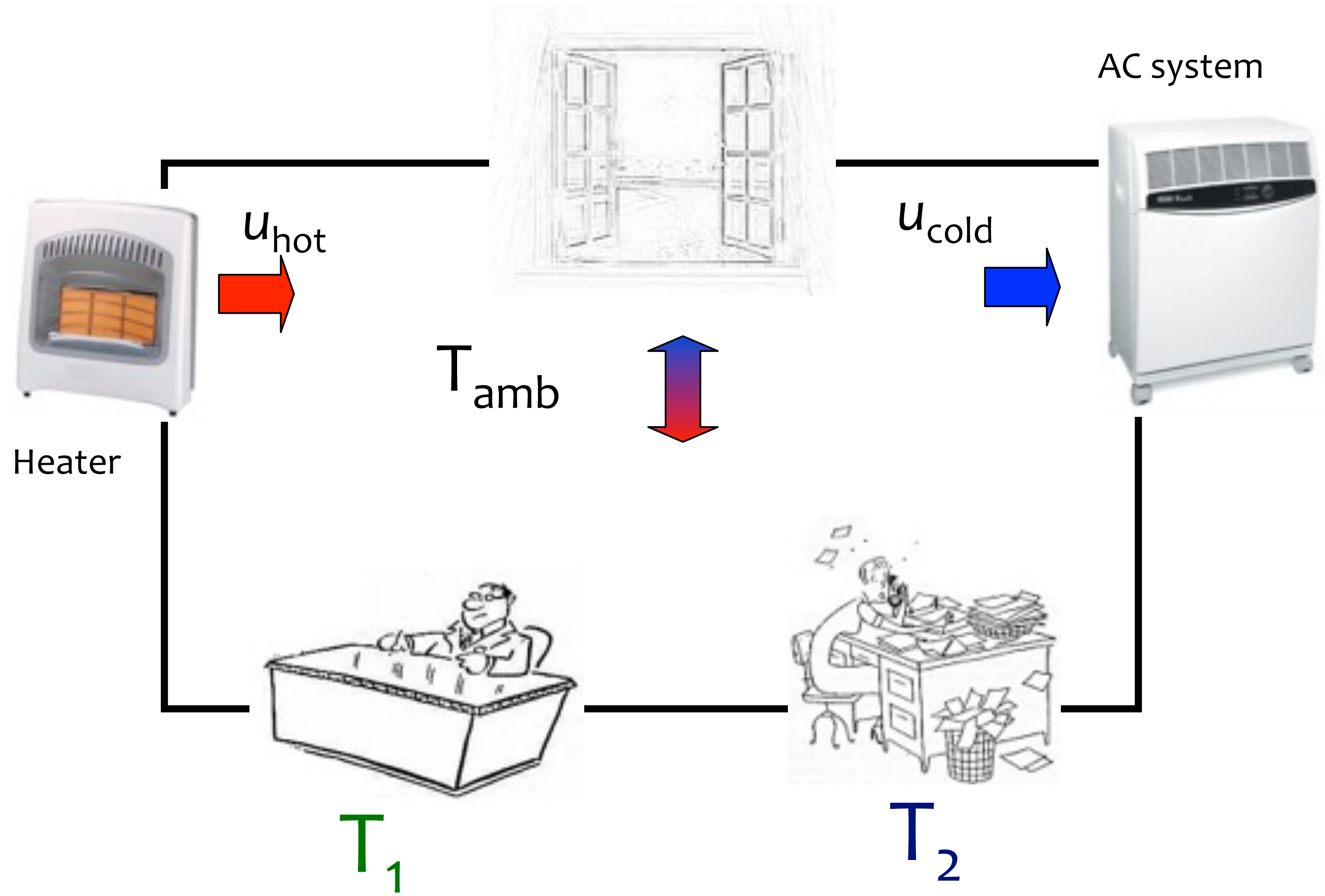
$$m \leq c(x_k, u_k) \leq M$$
$$\forall \text{ feasible } x_k, u_k$$

mixed-integer linear inequalities when c is a linear function

- Any **logic formula** involving Boolean variables can be translated into a set of **integer linear (in)equalities**
  - Example: $\delta_1 \text{ OR } \delta_2 = \text{TRUE}$ $\Longrightarrow$ $\delta_1 + \delta_2 \geq 1$ (Raman, Grossmann, 1991)
  - Translation of logic to inequalities can be automatic (see modeling language **HYSDEL**) (Torrisi, Bemporad, 2004)

- MPC optimization problem becomes a Mixed Integer Program (Bemporad, Morari, 1999) (Bemporad, Borrelli, Morari, 2000)

Room temperature control problem



Hybrid dynamics

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

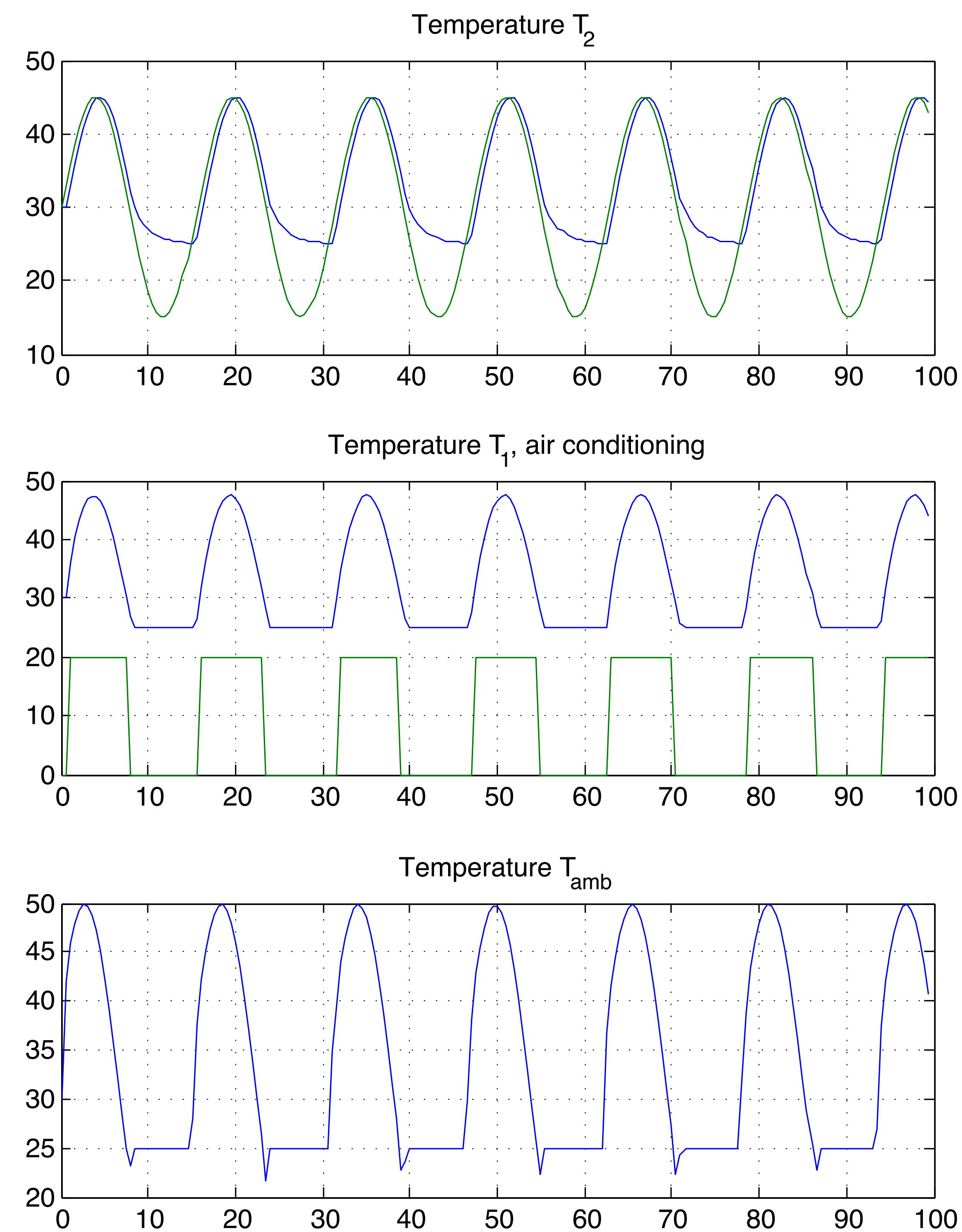
- $\dot{T}_1 = -\alpha_1(T_1 - T_{amb}) + k_1(u_{hot} - u_{cold})$ (body temperature dynamics of #1)
- $\dot{T}_2 = -\alpha_2(T_2 - T_{amb}) + k_2(u_{hot} - u_{cold})$ (body temperature dynamics of #2)

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

Hybrid MPC design (Hybrid Toolbox)

```
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));
          }
    }
}
```

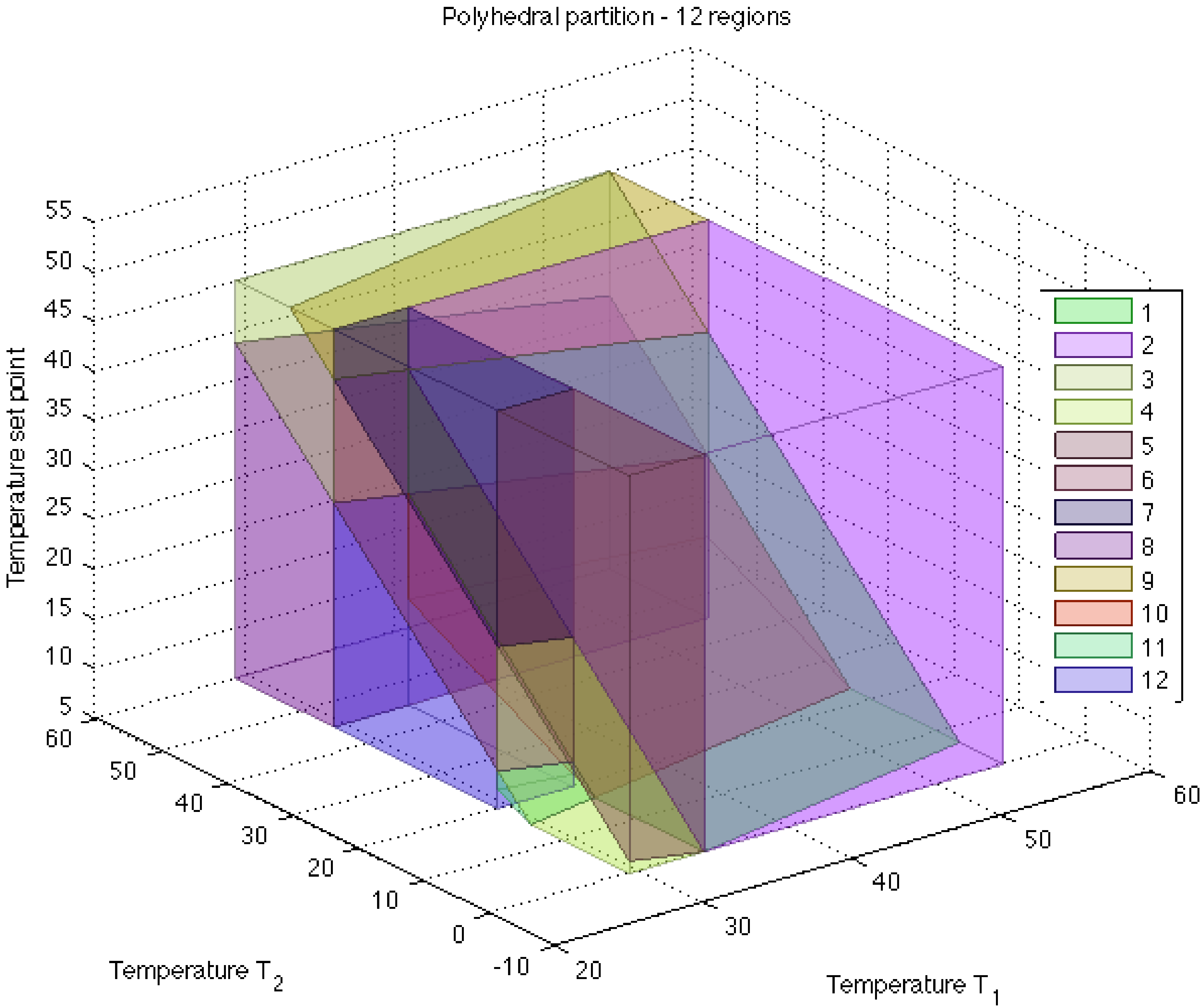


$$\min \sum_{k=1}^{2} |x_2(k) - r|$$
$$\text{s.t.} \quad x_1(k) \geq 25, \; k = 1,2$$
hybrid dynamics

```
20 optimization variables (8 continuous, 12 binary)
46 mixed-integer linear inequalities
```

CPU time = **1.3 ms** (IBM CPLEX 11.2, this Mac)

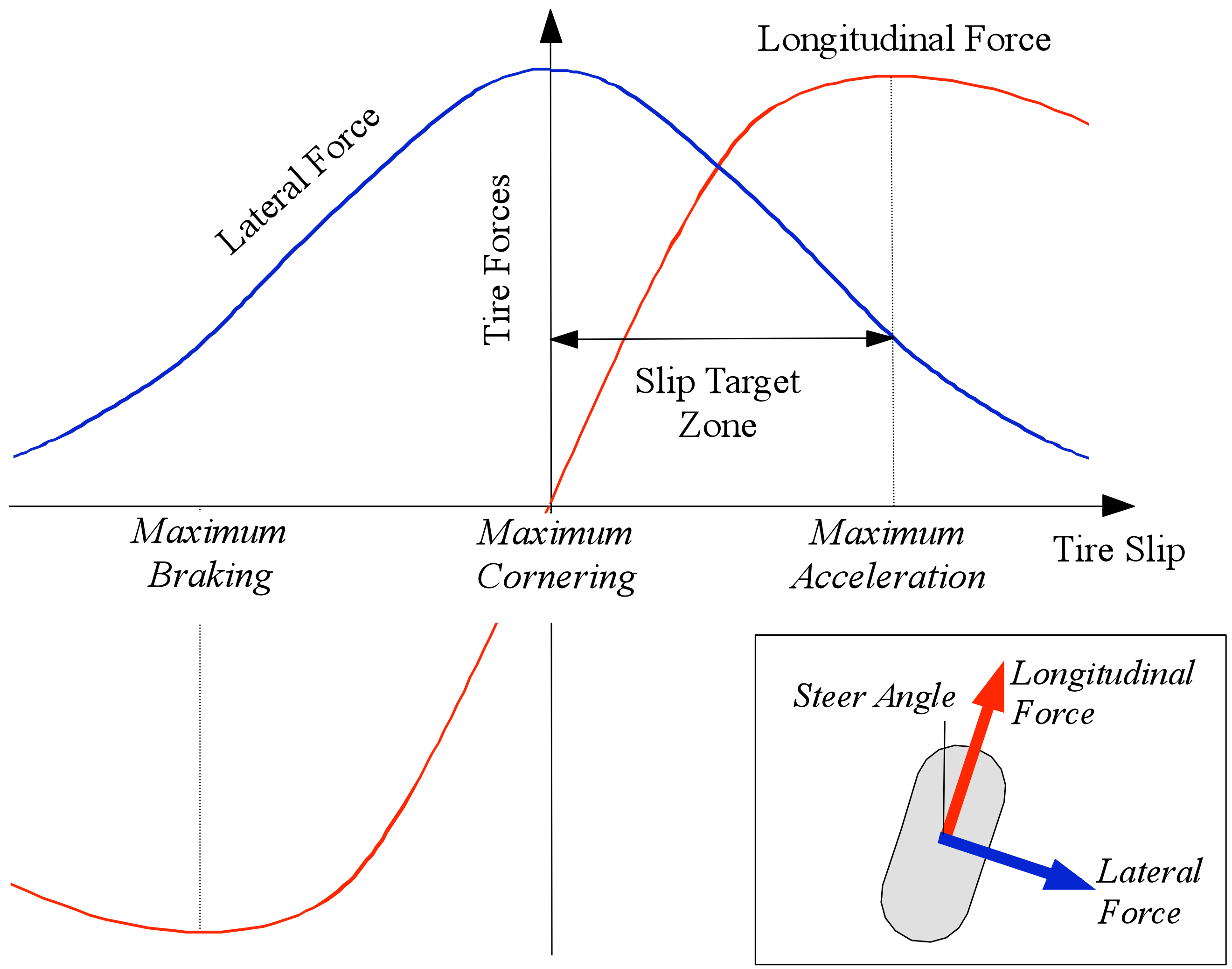
Explicit hybrid MPC equivalent (Hybrid Toolbox)



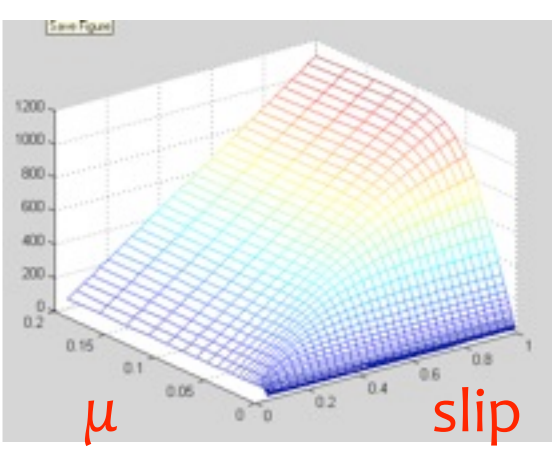
Note: explicit form does not change the control law at all !

CPU time = **0.8 ms** (compiled C-code, this Mac)

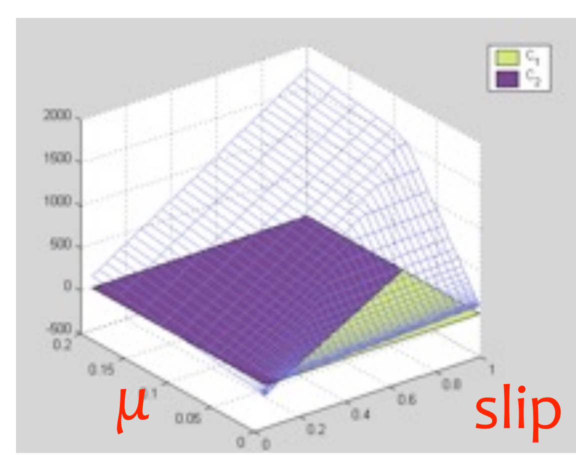
Explicit hybrid MPC for vehicle traction control



torque



piecewise affine approximation



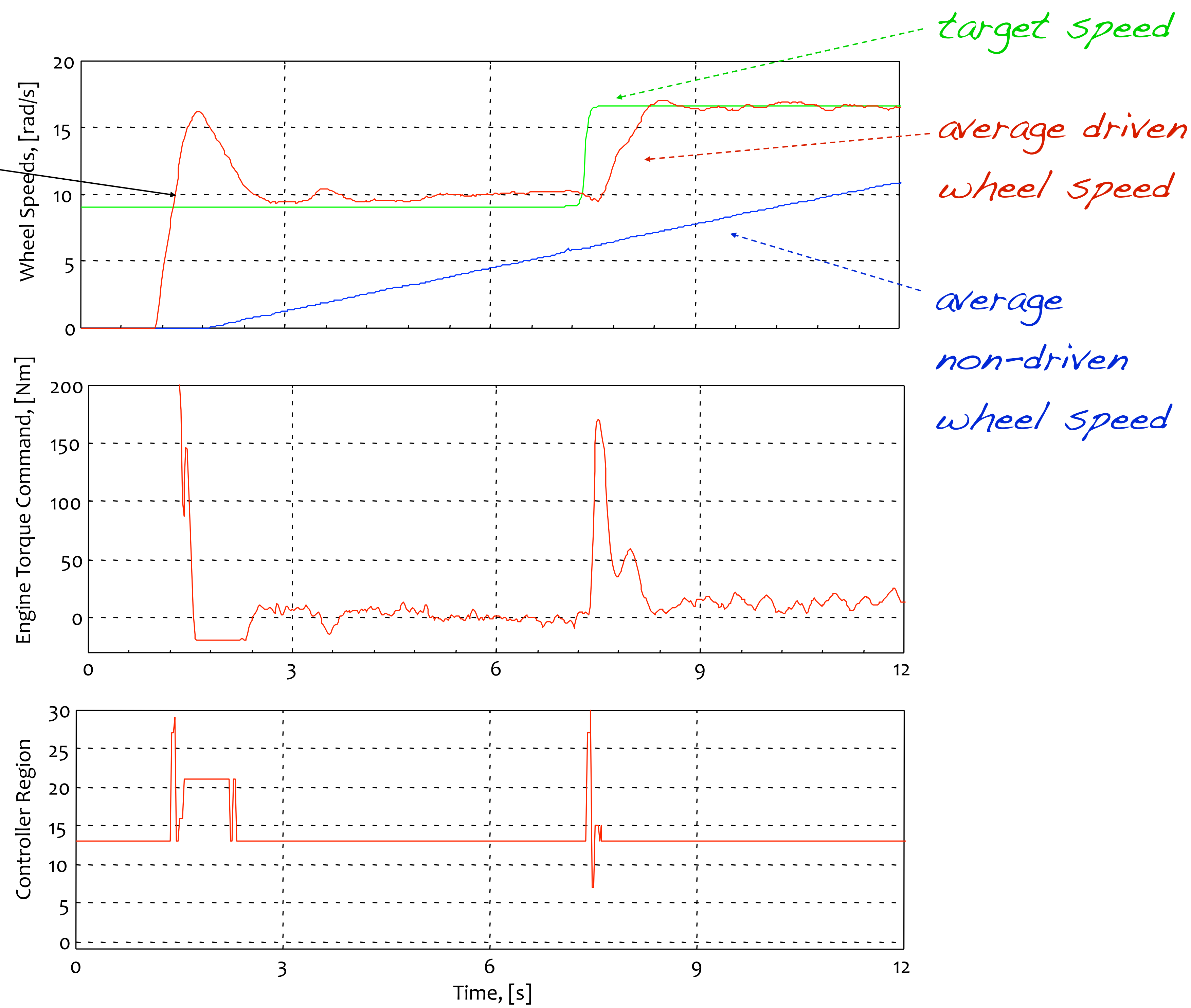
hybrid dynamical system (w/ delay)

input = torque
output = tire slip

Experimental Results

controller is triggered ON

(250 ms delay from commanded to actual engine torque → initial overspin)



Experimental results

indoor ice arena
(μ≈ 0.2)

2000 Ford Focus
2.0l 4-cyl engine
5-speed manual
transmission

Ford Motor Company

$$\begin{array}{ll}\min & \sum_{k=0}^{N} |\Delta\omega_{t+k} - \Delta\omega_{\text{des}}| \\ \text{s.t.} & -20 \text{ Nm} \leq \tau_d \leq 176 \text{ Nm}\end{array}$$

- 504 regions
- 20ms sampling time
- Pentium 266Mhz + Labview

Linear time-varying MPC

- MPC can easily handle **linear time-varying (LTV)** problems

$$\begin{cases} x_{k+1} = A_k(t,x(t))x_k + B_k(t,x(t))u_k + f_k(t,x(t)) \\ y_k = C_k(t,x(t))x_k + D_k(t,x(t))u_k + g_k(t,x(t)) \end{cases}$$

$$\underline{E_k(t,x(t))x_k + F_k(t,x(t))u_k \leq h_k(t,x(t))} \qquad k = 0,1,\ldots,N-1$$

$$\min \sum_{k=0}^{N} \ell_k(y_k, u_k, r(t+k), t, x(t)) \qquad \underline{\ell_k = \text{quadratic function of } y_k, u_k}$$

$$\begin{array}{ll} \min\limits_{U} & \frac{1}{2}U'H(t)U + F(t)'U + \cancel{\alpha(t)} \\ \text{s.t.} & G(t)U \leq W(t) \end{array}$$

LTV-MPC still leads to a **Quadratic Program (QP)** !

- Applications: time-varying systems (e.g.: aerospace), NL systems

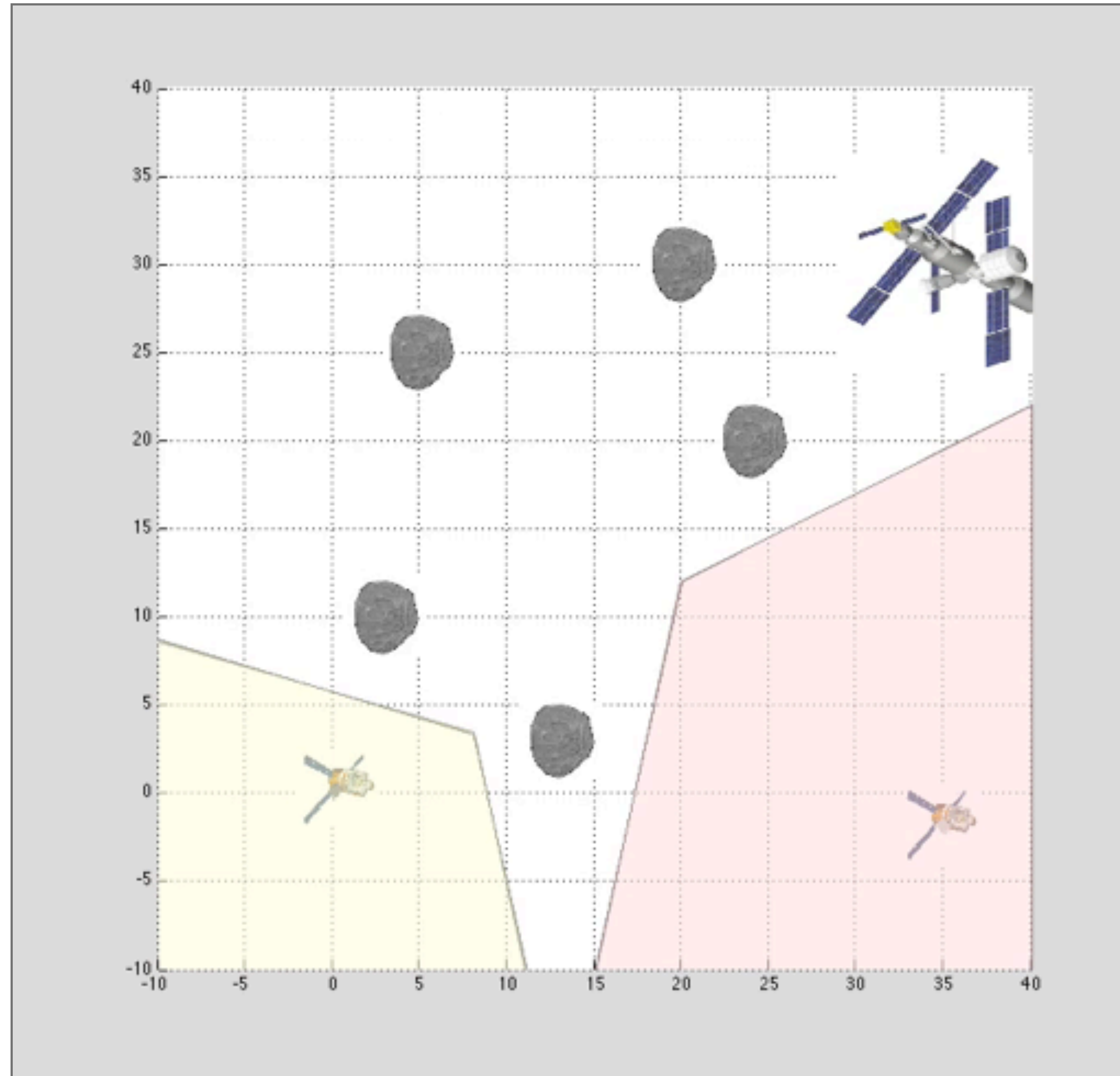
Example: LTV-MPC of UAVs

- Two unmanned aerial vehicles (UAVs) avoiding each other and obstacles

- Feasible space approximated as a (time-varying) polyhedron

- Each UAV solves its own MPC problem

- Previous optimal sequences exchanged to improve accuracy in predicting future locations of the other UAV



ROBMPC project
Robust Model Predictive Control (MPC) for Space Constrained Systems

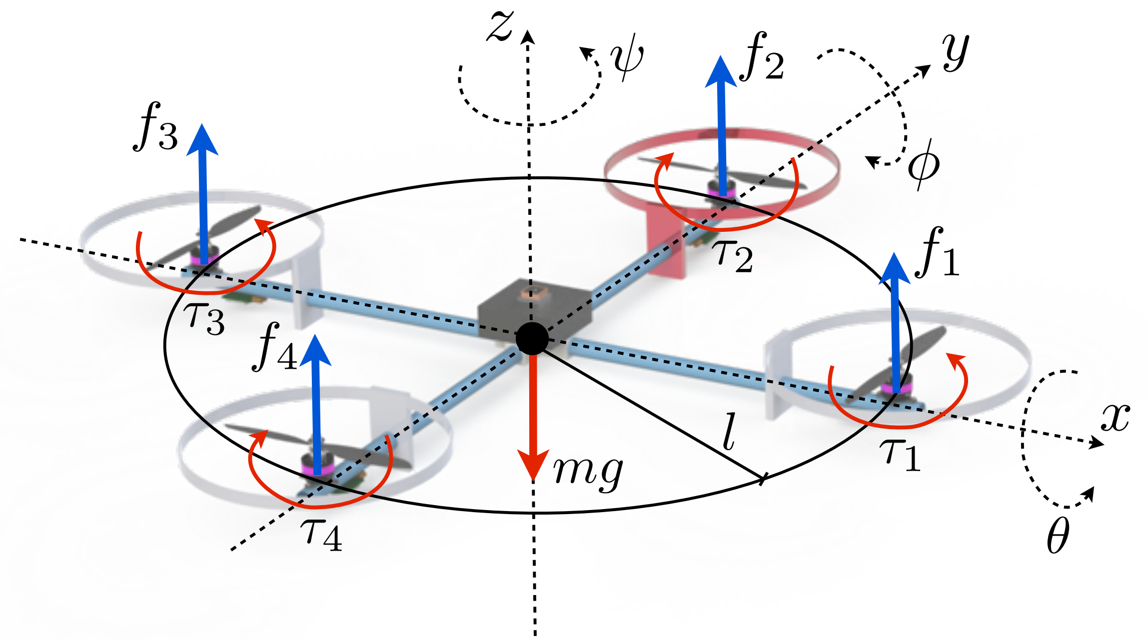
ORCSAT project
On-line Reconfiguration Control System and Avionics Technologies

MPCSofT Toolbox for MATLAB

(Bemporad, 2010-2011)

LTV-MPC for formation flying of quadcopters

- 4 command inputs: motor voltages $V_{Mi},\ \ i = 1, \ldots, 4$
- 12 states $\theta, \phi, \psi, x, y, z, \dot{\theta}, \dot{\phi}, \dot{\psi}, \dot{x}, \dot{y}, \dot{z}$



$$
\begin{aligned}
m\ddot{x} &= -f \sin\theta - \beta\dot{x} \\
m\ddot{y} &= f \cos\theta \sin\phi - \beta\dot{y} \\
m\ddot{z} &= f \cos\theta \cos\phi - mg - \beta\dot{z} \\
\ddot{\theta} &= \frac{\tau_\theta}{I_{xx}} \\
\ddot{\phi} &= \frac{\tau_\phi}{I_{yy}} \\
\ddot{\psi} &= \frac{l}{I_{zz}}(-f_1 + f_2 - f_3 + f_4)
\end{aligned}
$$

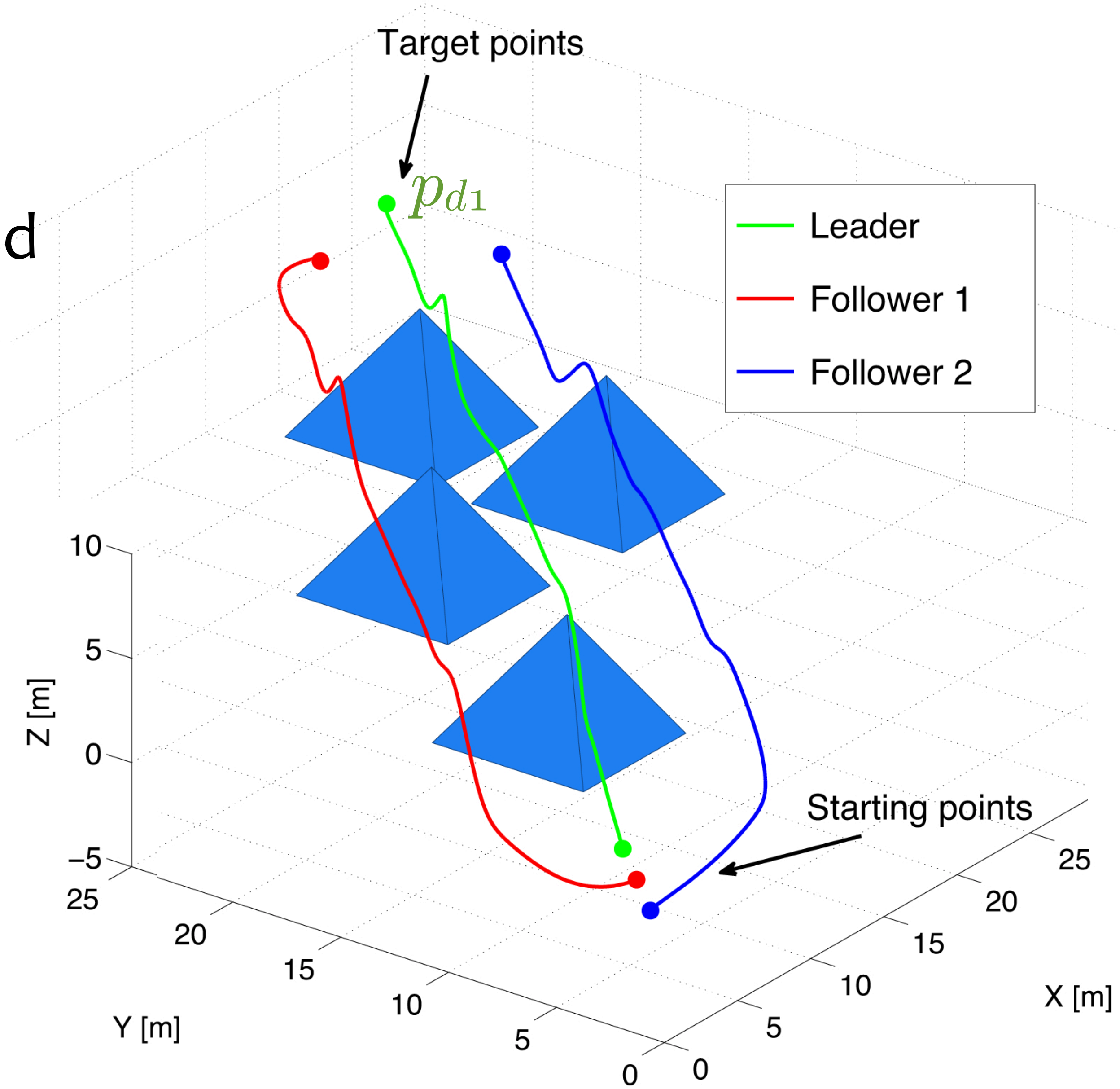
$$
\begin{aligned}
f &= f_1 + f_2 + f_3 + f_4 \\
\tau_\theta &= (f_2 - f_4)l \\
\tau_\phi &= (f_3 - f_1)l
\end{aligned}
$$

$$
f_i = \frac{9.81(22.5V_{Mi} - 9.7)}{1000},\ \ i = 1, \ldots, 4
$$

- Highly nonlinear and coupled dynamics

Example: LTV-MPC for formation flying

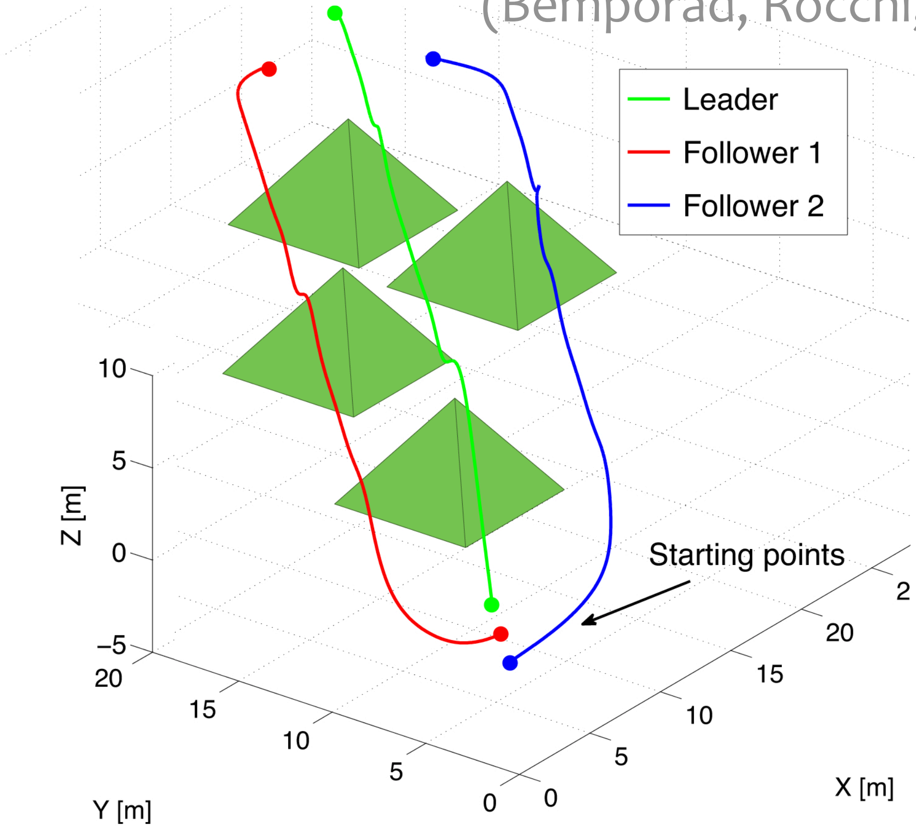
- 4 tetrahedral obstacles, $W = \mathrm{conv}\left(\begin{bmatrix}-1/3\\-1/3\\-1/2\end{bmatrix}, \begin{bmatrix}2/3\\-1/3\\-1/2\end{bmatrix}, \begin{bmatrix}-1/3\\2/3\\-1/2\end{bmatrix}, \begin{bmatrix}0\\0\\1/2\end{bmatrix}\right)$
- UAVs modeled as small parallelepipeds
- *MPCSofT Toolbox* used for LTV-MPC design, simulation, and code generation
- QP problem builder and solver implemented in Embedded MATLAB
- CPU time = ~**75 ms** (per time-step) (MATLAB R2009b, Macbook Air)



Comparison with other guidance methods

decentralized **hybrid** MPC

(Bemporad, Rocchi, IFAC 2011)

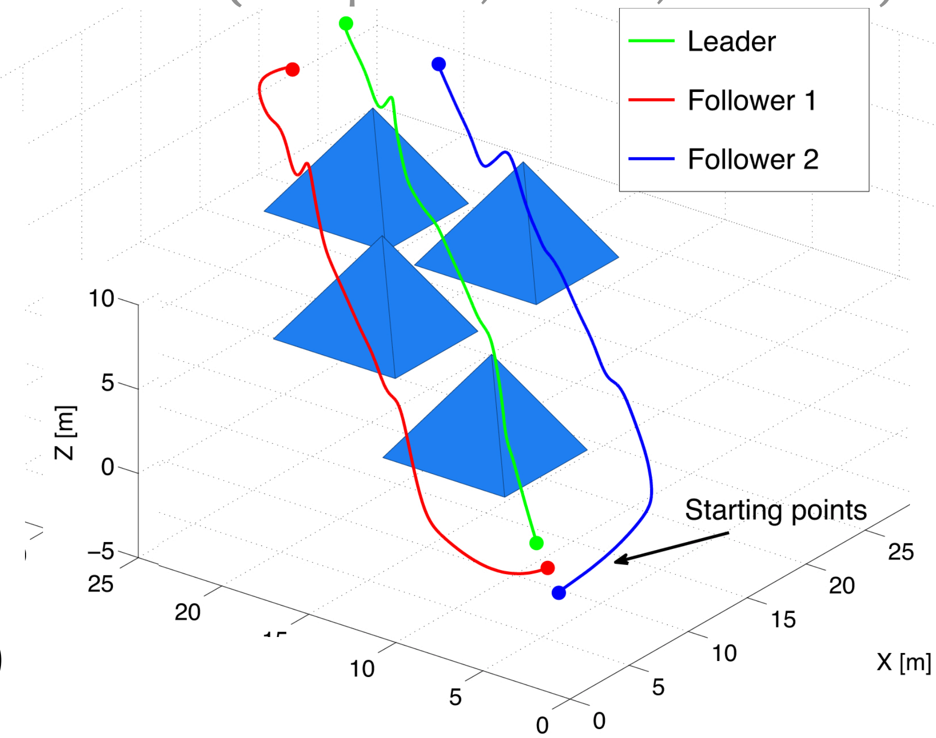


CPU time = ~**45 ms**

(IBM CPLEX)

decentralized **LTV** MPC

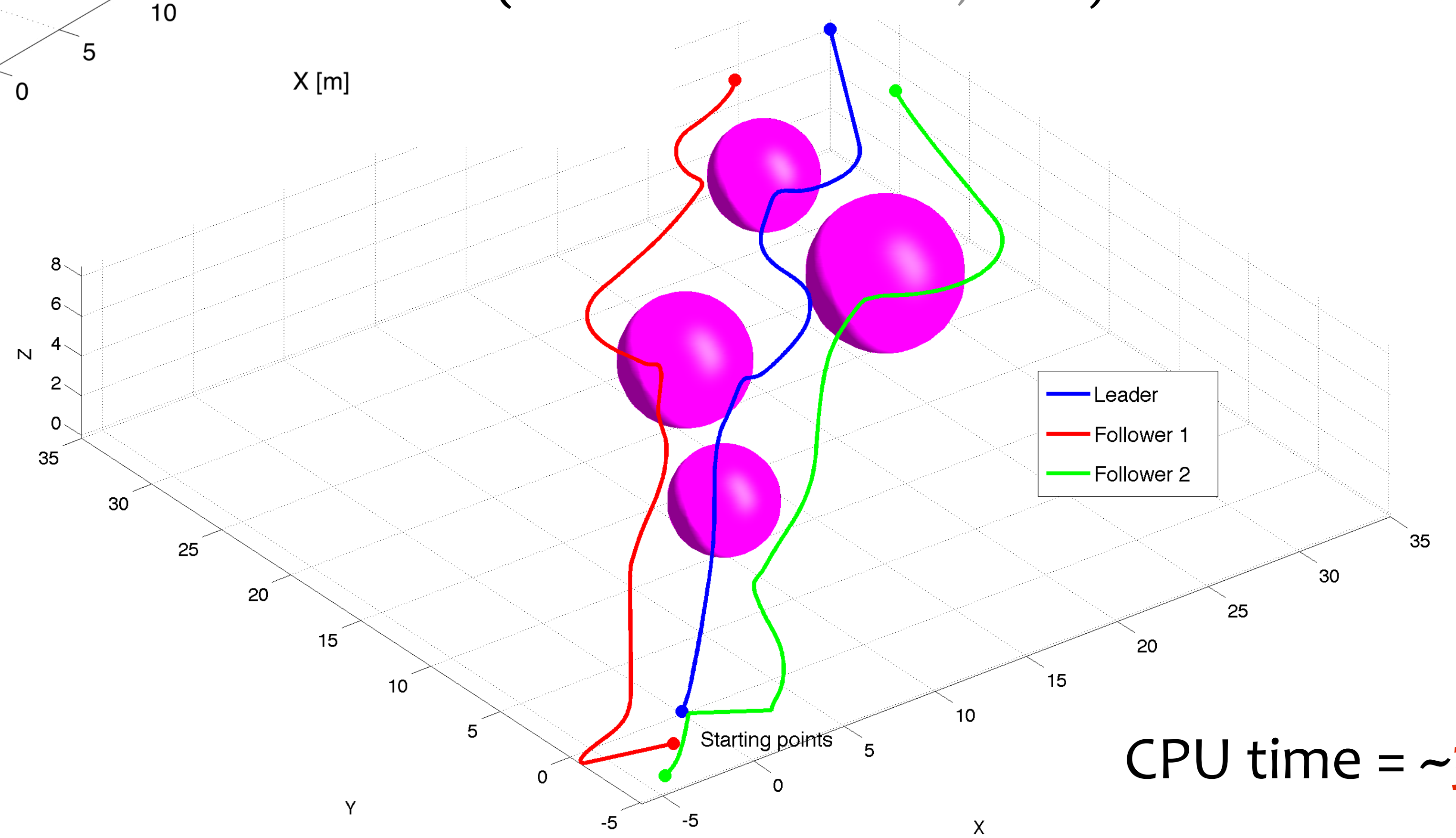
(Bemporad, Rocchi, CDC 2011)



CPU time = ~**75 ms**

potential fields

(method of Paul *et al.*, 2008)



CPU time = ~**3 ms**

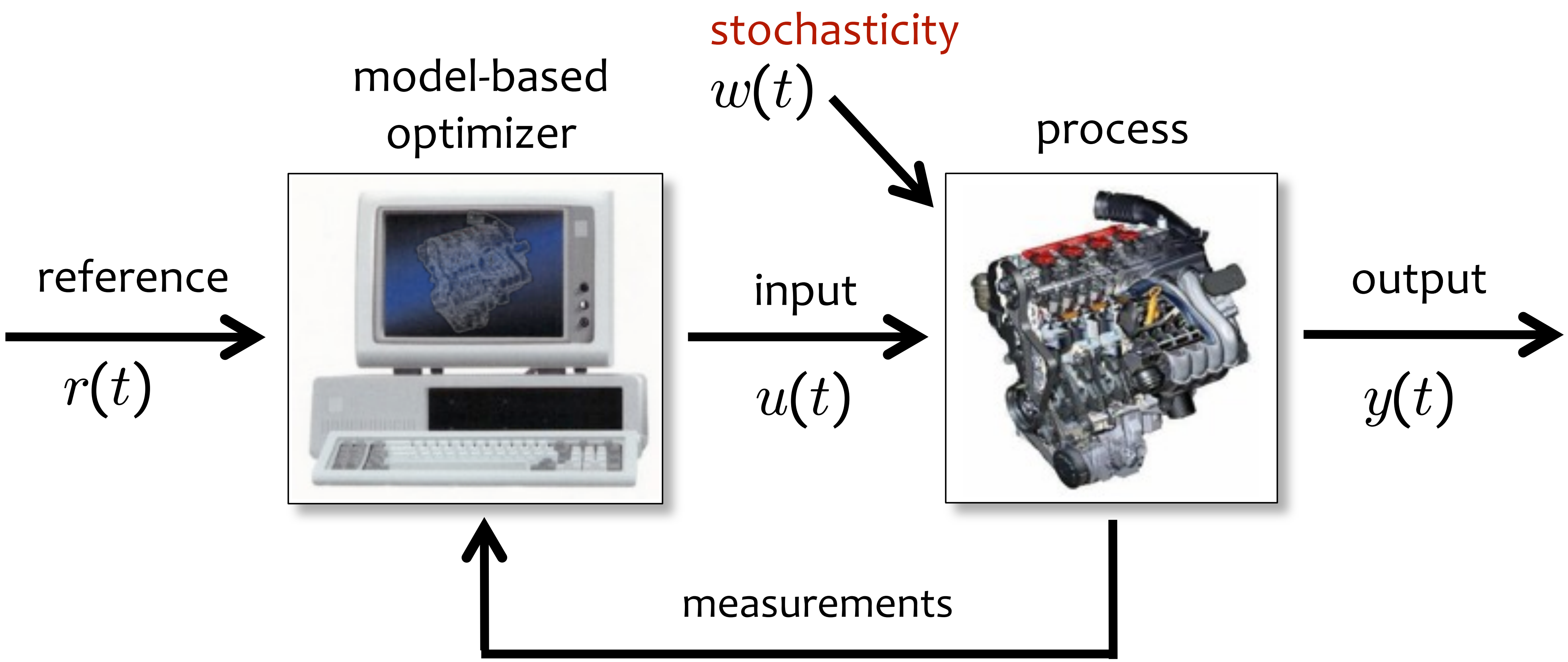
Comparison with other guidance methods

$J_{\mathrm{tt}} = \sum_{k=350}^{3080} \|p_L(k) - p_t\|_2^2$	*target tracking* Integral Square Error (ISE)
$J_{\mathrm{fpt}} = \sum_{k=350}^{3080} \|p_L(k) - p_{F1}(k) - p_{d1}\|_2^2 + \|p_L(k) - p_{F2}(k) - p_{d2}\|_2^2$	*formation pattern tracking* ISE
$J_{\mathrm{u}} = \sum_{k=350}^{3080} \|u(k) - u(k-1)\|_1$	*absolute derivative of input signals* (IADU)

	J_{tt}	J_{fpt}	J_{u}
centralized hybrid MPC	1	1	1
decentralized hybrid MPC	-0.09%	-0.51%	-0.03%
decentralized LTV MPC	-2.46%	-15.47%	+9.20%
potential fields	+210.32%	+294.30%	+212.75%

(indices normalized with respect to the centralized hybrid MPC performance)

Stochastic Model Predictive Control (SMPC)



Use a ***stochastic model*** *of the process to* ***predict*** *the possible future evolutions of the process in order to optimize the* ***control*** *signal*

Receding horizon philosophy

- At time t: solve a **stochastic optimal control** problem over a finite future horizon of N steps:

$$
\begin{aligned}
\min_{u} \quad & E_w\left[\sum_{k=0}^{N-1} \|y_{t+k} - r(t)\|^2 + \rho\|u_{t+k}\|^2\right] \\
\text{s.t.} \quad & x_{t+k+1} = f(x_{t+k}, u_{t+k}, w_{t+k}) \\
& y_{t+k} = g(x_{t+k}, u_{t+k}, w_{t+k}) \\
& u_{\min} \leq u_{t+k} \leq u_{\max} \\
& Prob(y_{\min} \leq y_{t+k} \leq y_{\max}) \geq p \\
& x_t = x(t), \;\; k = 0, \ldots, N-1
\end{aligned}
$$

$x(t)$ = process state

$u(t)$ = manipulated vars

$y(t)$ = controlled output

$w(t)$ = stochastic dist.

- Only apply the first optimal move $u^*(t)$, discard $u^*(t+1)$, $u^*(t+2)$, …

- At time $t+1$: Get new measurement $x(t+1)$, repeat the optimization, and so on

Scenario-based stochastic MPC

Existing literature

(Schwarme & Nikolaou, 1999)
(Wendt & Wozny, 2000)
(Batina, Stoorvogel, Weiland, 2002)
(van Hessem & Bosgra 2002)
(Munoz de la Pena, Bemporad, Alamo, 2005)
(Couchman, Cannon, Kouvaritakis, 2006)
(Primbs, 2007)
(Bemporad, Di Cairano, 2005)
(Oldewurtel, Jones, Morari, 2008)
(Ono, Williams, 2008)
(Bernardini & Bemporad, 2009)

Stochastic prediction model

$$x(k+1) = A(\underline{w(k)})x(k) + B(\underline{w(k)})u(k) + H\underline{w(k)}$$

$$w(k) \in \{w_1, w_2, \ldots, w_s\}$$

$$P[w(k) = w_i] = p_i(k)$$

Control goals

- Less conservative control action w.r.t. *robust* MPC
- No restrictive assumptions on the disturbance distribution
- Guarantee stochastic convergence $\lim_{k\to\infty} E[x'(k)x(k)] = 0$ (for H=0) and recursive feasibility
- Decouple performance optimization and stability issues

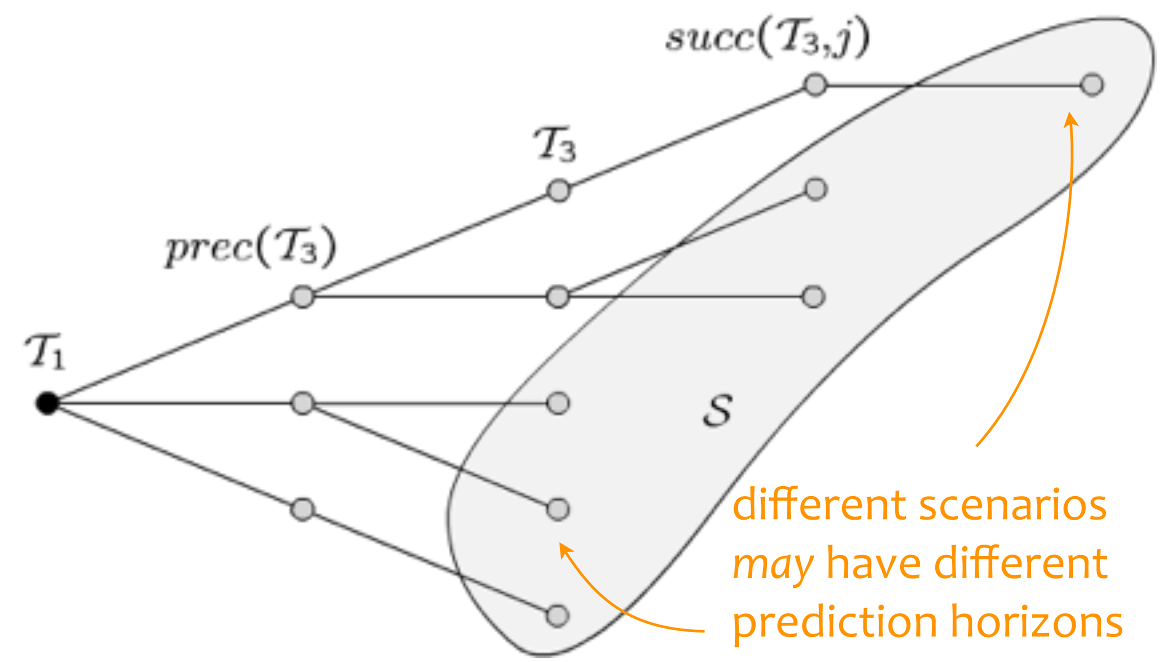
Some theoretical results for SMPC

(Bernardini, Bemporad, IEEE TAC, 2012)

- **Convergence** and **feasibility** guaranteed by solving **offline** an LMI problem (synthesis of a stochastic Lyapunov function)

- **Performance** optimized ***online*** via flexible **stochastic performance trees**

- A finite number of **scenarios** is generated only considering the **"most probable"** nodes



- Optimize on-line a stochastic performance index *(less probable nodes are less weighted)*

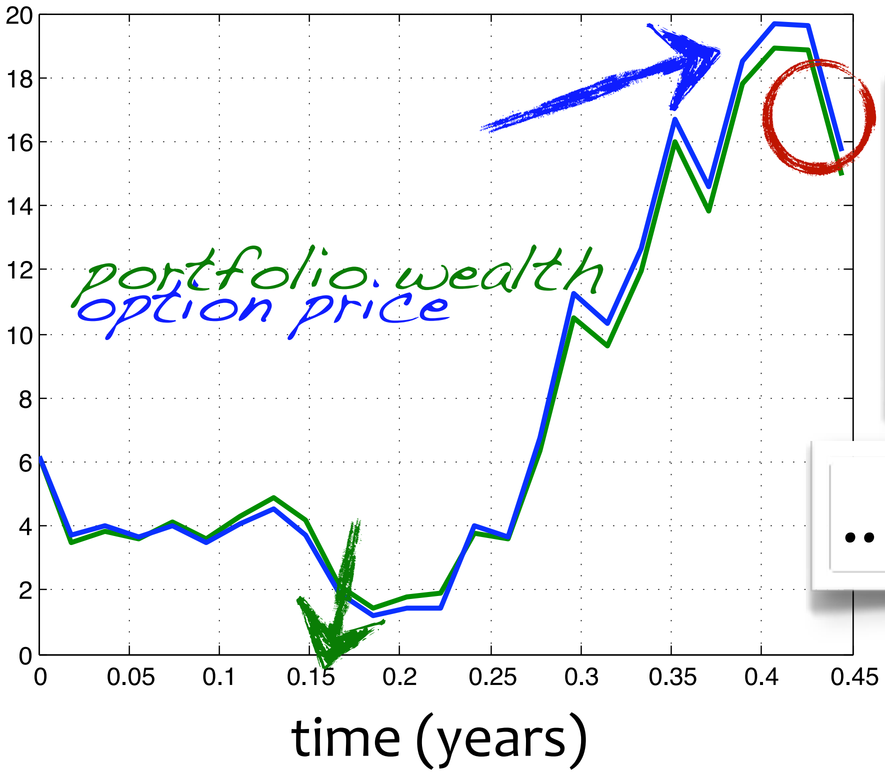
$$\min\left\{\sum_{i\in\mathcal{T}\setminus\{\mathcal{T}_1\}}\pi_i x_i'Q_x x_i+\sum_{j\in\mathcal{T}\setminus\mathcal{S}}\pi_j u_j'Q_u u_j\right\}$$

π_i is the realization probability of node i

Stochastic MPC for option hedging

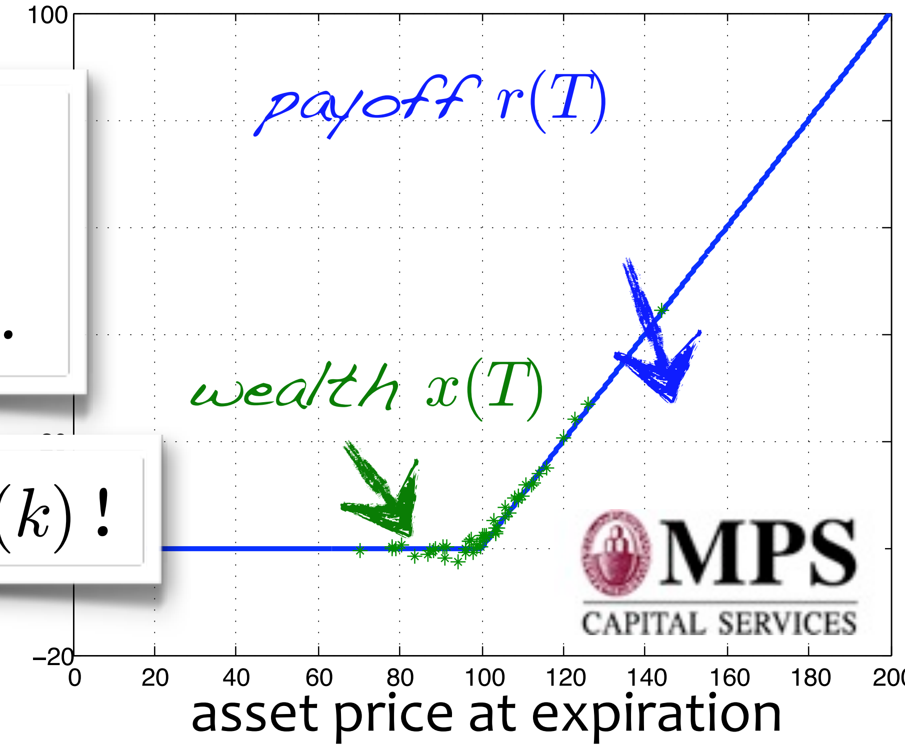
- Dynamic hedging of **financial options** (Bemporad, Bellucci, Gabbriellini, 2009)

(Bemporad, Gabbriellini, Puglia, Bellucci, CDC'10)



SMPC **adjusts dynamically** the portfolio so that **wealth** $x(T) \cong$ **payoff** $r(T)$ ...

.. for any price realization $w_i(k)$!



SMPC for real-time market-based power dispatch

- A Balance Responsible Party (BRP) is the only legal entity trading on the *energy (PX)* and *ancillary service (AS)* markets
- **Objective:** Minimize (expected) costs via efficient use of intermittent resources, and maximize (expected) profits by trading on PX and AS markets
- **Constraints:** Grid capacity constraints, rate limits, load balancing, AS balancing

p_i = power injection (decision)

a_i = reserve capacity (decision)

r = intermittent generation (stochastic)

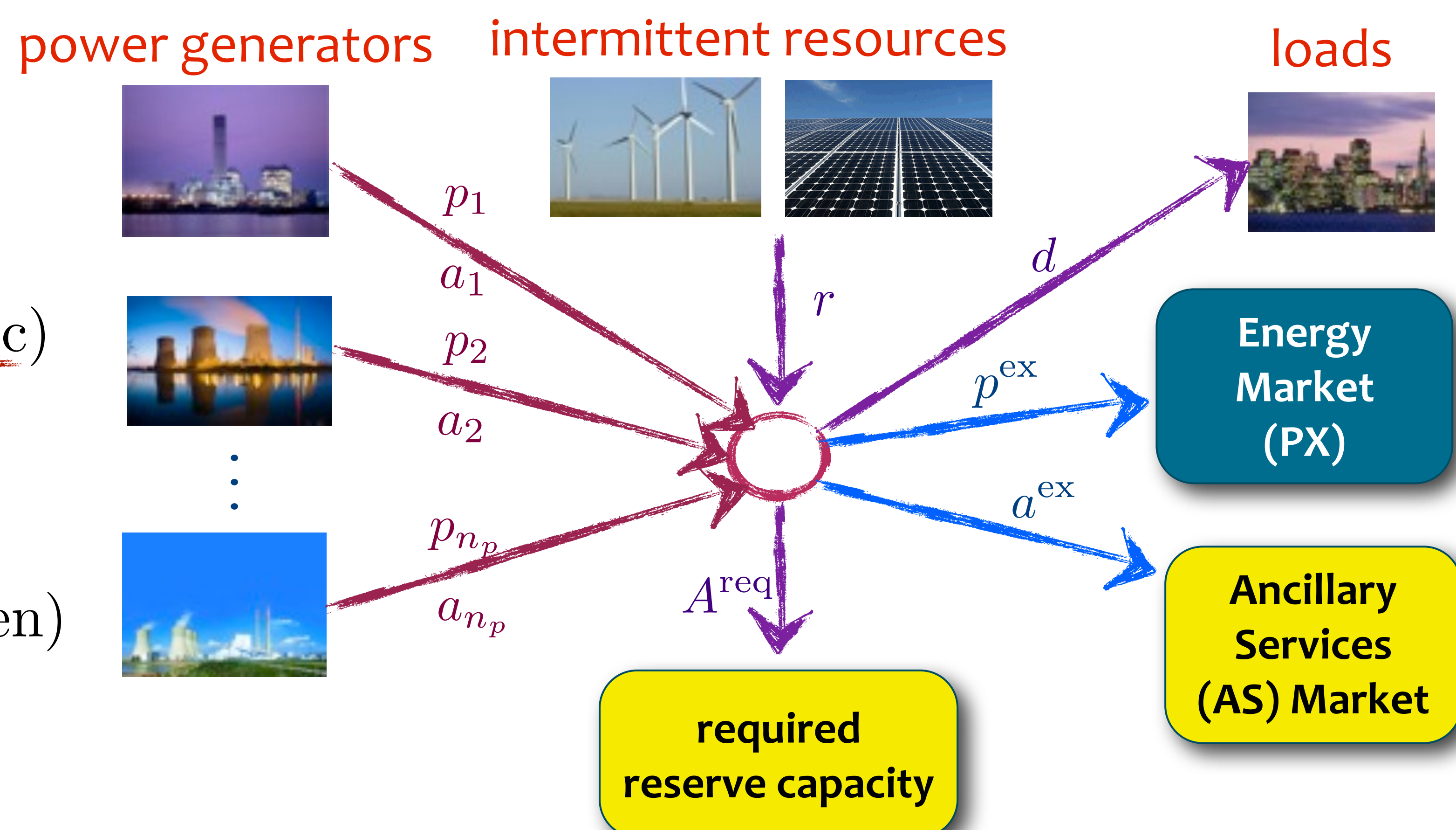
d = demand (stochastic)

p^{ex} = power exchanged (decision)

p^{as} = power exchanged (decision)

A^{req} = required reserve for BRP (given)

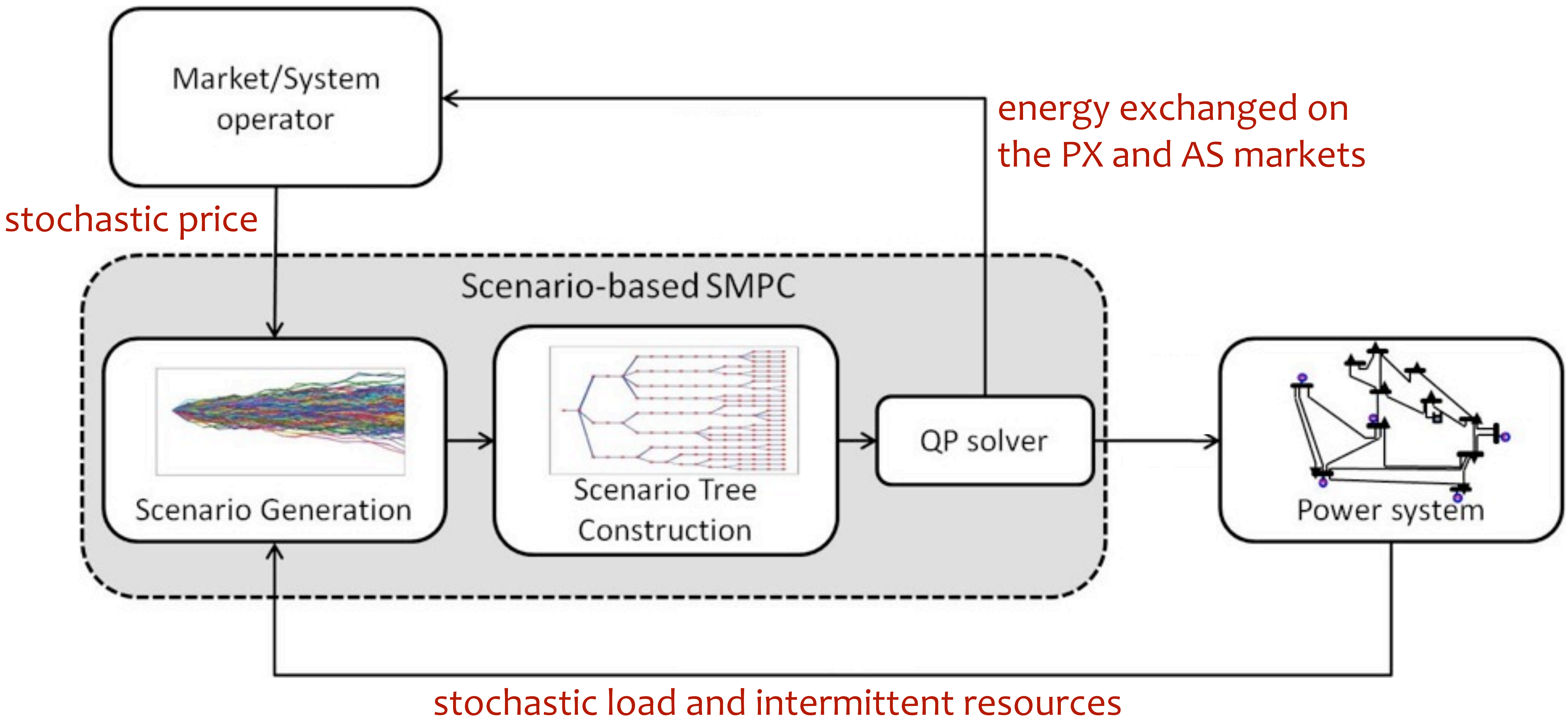
energy prices are also stochastic



SMPC for real-time market-based power dispatch

(Patrinos, Trimboli, Bemporad 2011)

- Stochastic MPC architecture



SMPC for market-based optimal power dispatch

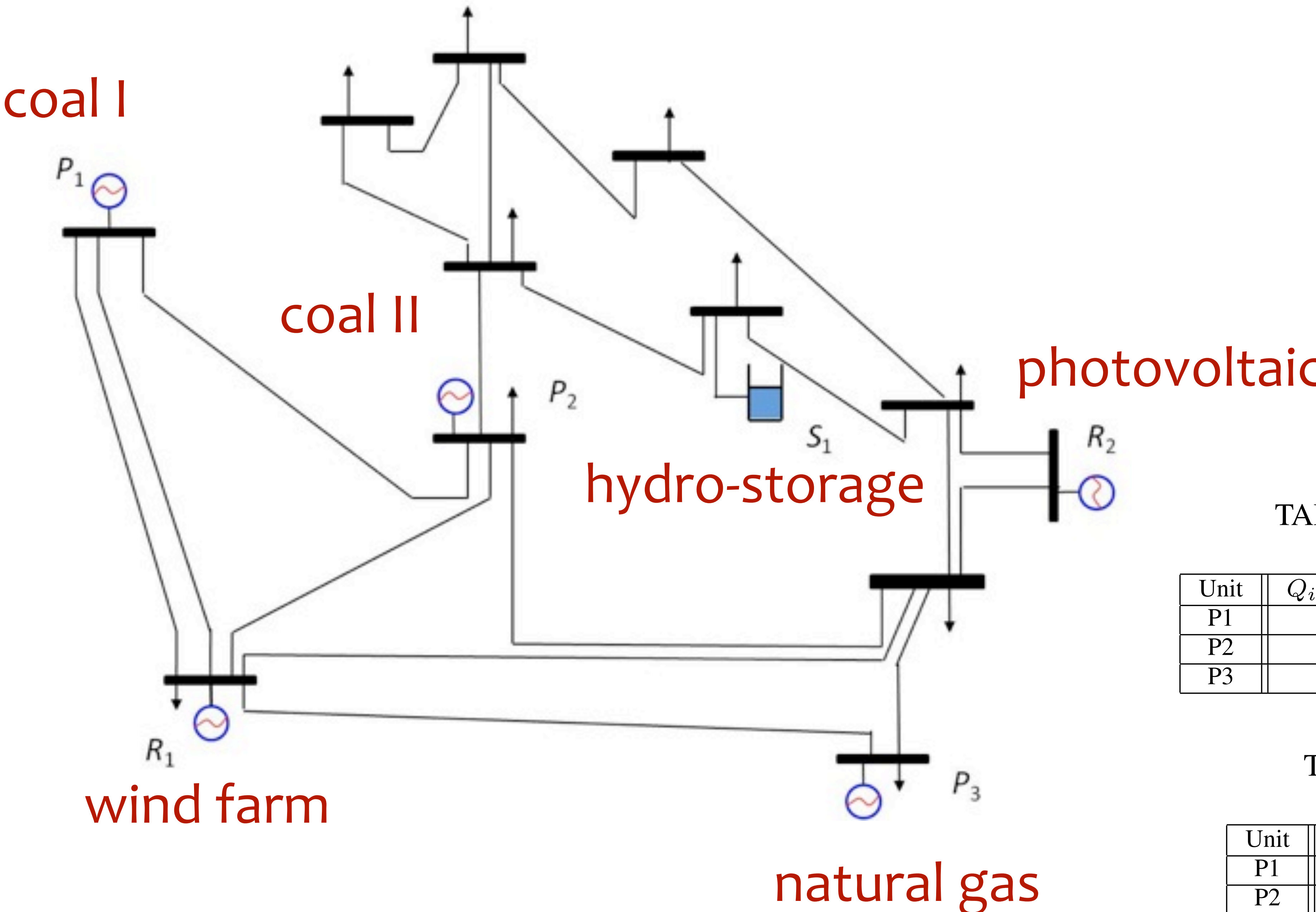


TABLE I: Generator Cost Data

Unit	Q_i (\$/MWh^2)	q_i (\$/MWh)	c_i (\$)
P1	0.009	30.375	398.025
P2	0.0225	73.35	292.275
P3	0.0488	61.488	489.952

TABLE II: Generator Data

Unit	$p_i^{\min}$	$p_i^{\max}$	$\Delta p_i^{\min}$	$\Delta p_i^{\min}$
P1	450	1100	-250	250
P2	50	500	-200	200
P3	50	100	-75	75

TABLE III: Storage Data

Unit	$x_i^{\min}$	$x_i^{\max}$	$\Delta x_i^{\min}$	$\Delta x_i^{\min}$	α_i	α_i^{c}	α_i^{d}
S1	15	300	-120	120	0.95	0.85	0.90
	$u_i^{\mathrm{c,min}} = u_i^{\mathrm{d,min}}$			$u_i^{\mathrm{c,max}} = u_i^{\mathrm{d,max}}$			
	0			300			

SMPC for market-based optimal power dispatch

- Numerical results

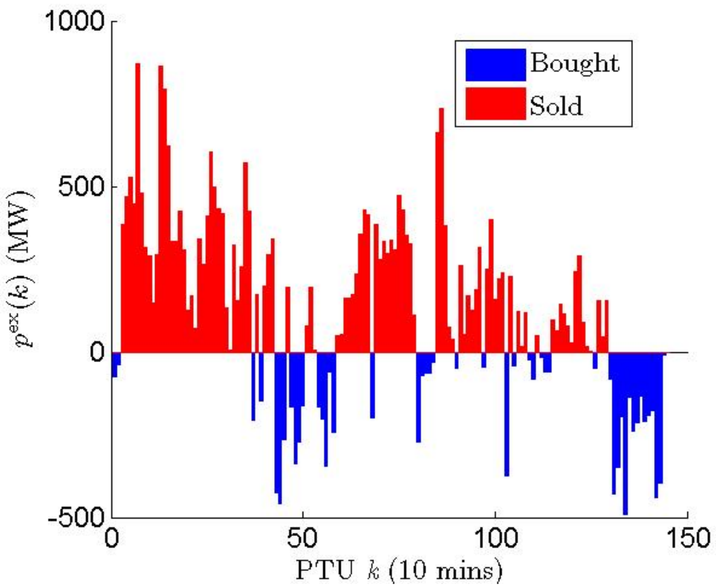
Exact knowledge future uncertainty

Time-varying expectations used for future uncertainty

SMPC: as tree density increases, performance gets better, although numerical complexity gets larger

Algorithm	Storage Cost	No Storage Cost	Avg # of nodes
Prescient-OC	6427979	6879741	
CE-MPC	9778750	9819518	
SSMPC ($e_{\text{rel}} = 0.1$)	7134582	7245962	350
SSMPC ($e_{\text{rel}} = 0.2$)	7144011	7249401	335
SSMPC ($e_{\text{rel}} = 0.3$)	7148494	7250207	172
SSMPC ($e_{\text{rel}} = 0.4$)	7179848	7264505	87
SSMPC ($e_{\text{rel}} = 0.5$)	7224912	7267497	50
SSMPC ($e_{\text{rel}} = 0.6$)	7239985	7277410	38
SSMPC ($e_{\text{rel}} = 0.7$)	7259491	7298023	31
SSMPC ($e_{\text{rel}} = 0.8$)	7255246	7312092	26
SSMPC ($e_{\text{rel}} = 0.9$)	7260424	7318643	22
SSMPC ($e_{\text{rel}} = 1.0$)	7260424	7318642	20

power exchanged with grid

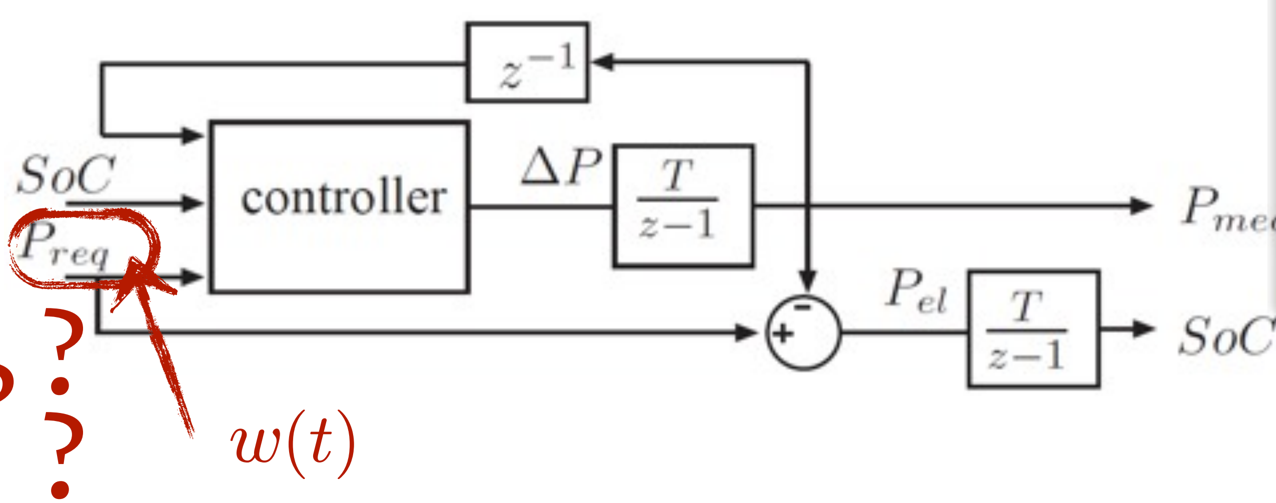


SMPC for hybrid electric vehicles (HEVs)

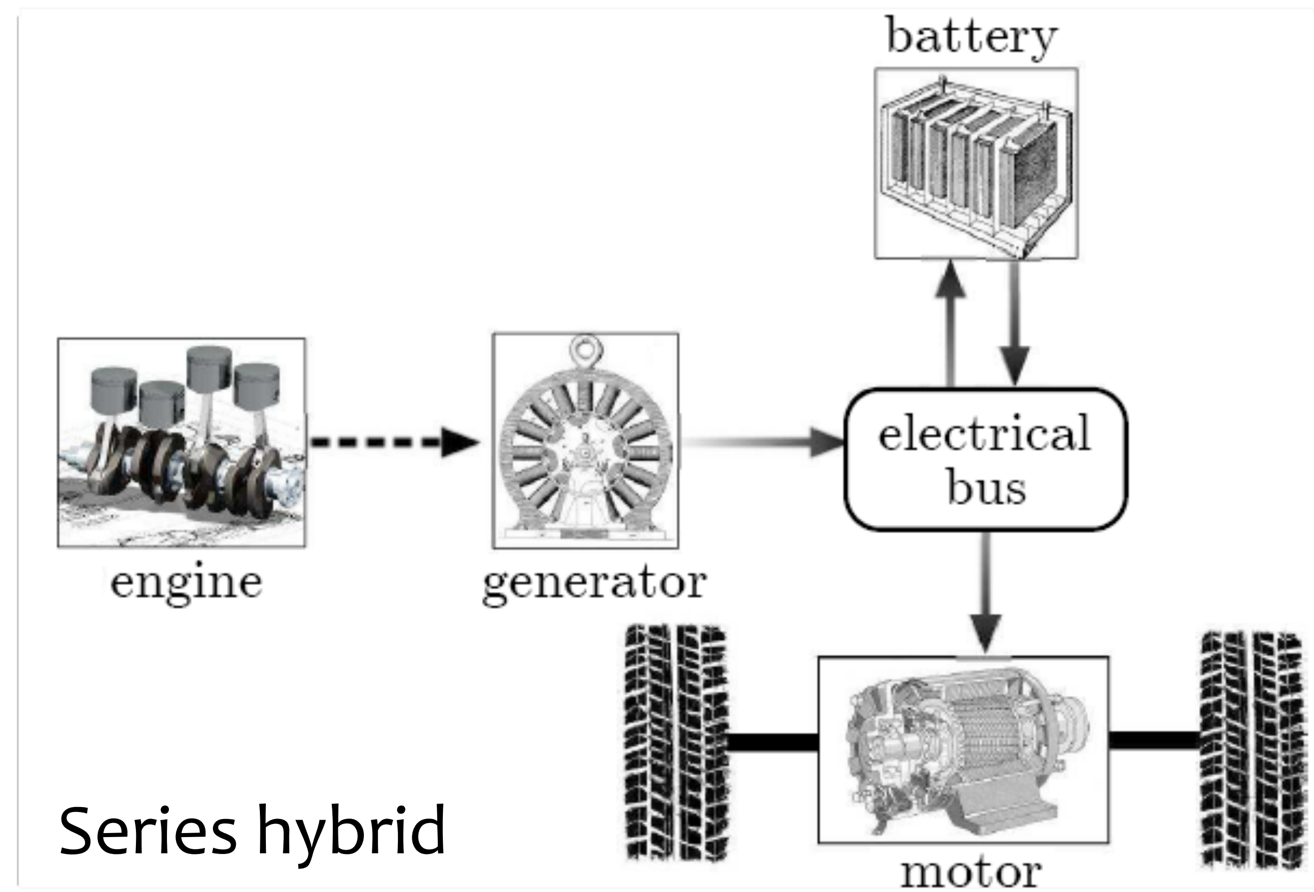
Control problem: Split power request among the different power sources in HEVs to optimize a given performance metrics

Existing approaches

- Deterministic Dynamic Programming (DDP)
- Stochastic Dynamic Programming (SDP)
- Rule-based
- Game Theory
- Deterministic (hybrid) MPC



$P_{req}(w(t))$ = driver's power request



$$P_{req}(k) = P_{el}(k) + P_{mec}(k) - P_{br}(k)$$

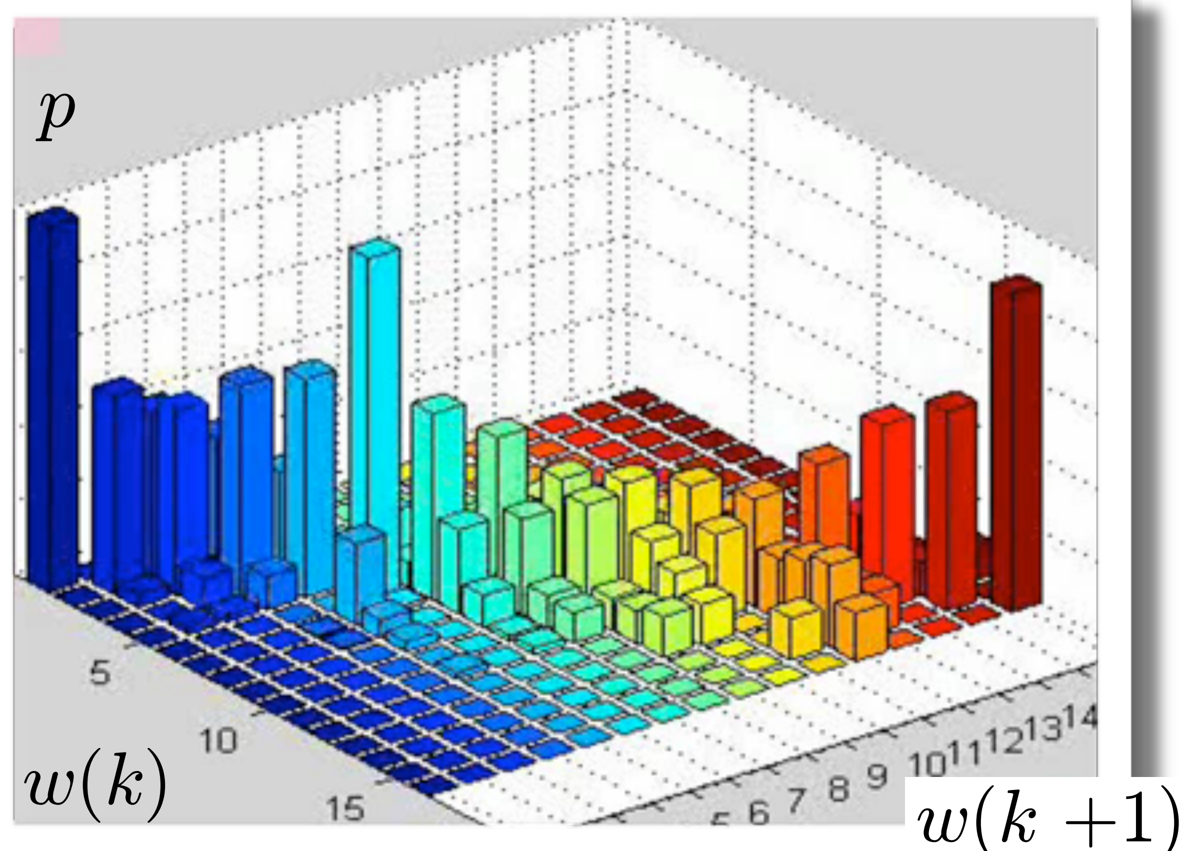
Learning a stochastic model of the driver

- The driver action on the vehicle is modeled by the **stochastic** process $w(k)$
- Assume that the realization $w(k)$ can be **measured** at every time step k
- Depending on the **application**, $w(k)$ may represent different quantities (e.g., power request in an HEV, acceleration, velocity, steering wheel angle, ...)

Good model for control purposes: $w(k)$ = **Markov chain**

$$[T]_{ij} = \mathbf{P}[w(k+1) = w_j | w(k) = w_i]$$

Number of states in Markov chain determines the **trade-off** between complexity *and* accuracy



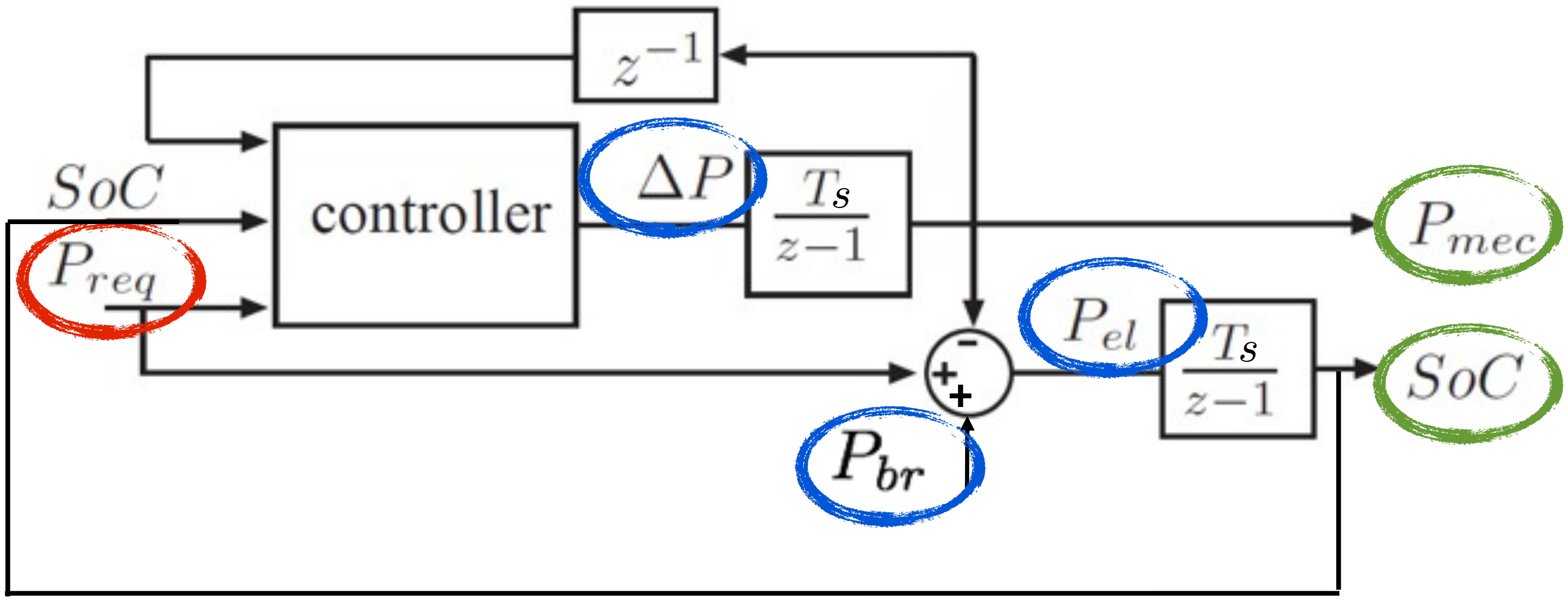
Transition probability matrix T is easily estimated from driver's data

Several model improvements are possible (e.g., multiple Markov chains)

SMPC problem for HEV power management

Manipulates inputs

$$\Delta P(k),\ P_{el}(k),\ P_{br}(k)$$



Uncertainty

$$P_{req}(w(k))$$

sample time T_s=1 s

Controlled output

$$P_{req}(k) = P_{el}(k) + P_{mec}(k) - P_{br}(k)$$

Constraints

$$\begin{array}{rcccl}
SoC_{min} & \leq & SoC(k) & \leq & SoC_{max} \\
0 & \leq & P_{mec}(k-1) & \leq & P_{mec,max} \\
P_{el,min} & \leq & P_{el}(k) & \leq & P_{el,max} \\
\Delta P_{min} & \leq & \Delta P & \leq & \Delta P_{max} \\
0 & \leq & P_{br}(k) & &
\end{array}$$

State-space equations

$$SoC(k+1) = SoC(k) - KT_s P_{el}(k)$$

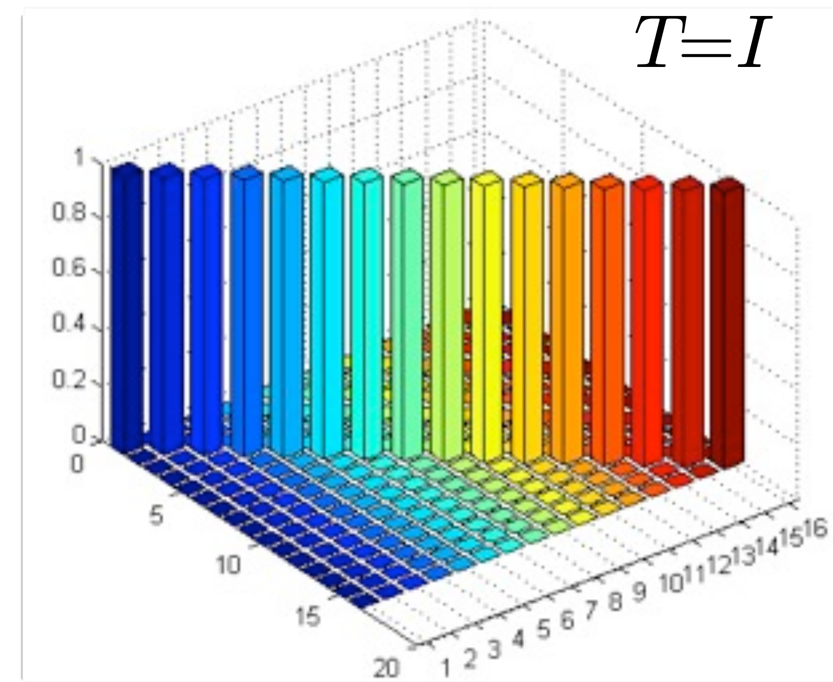
$$P_{mec}(k+1) = P_{mec}(k) + \Delta P(k)$$

Comparison with deterministic MPC

“Frozen-time” MPC (FTMPC)

No stochastic disturbance model,
simply ZOH along prediction horizon

$$P_{req}(w(t+k|k)) = P_{req}(w(k))$$



“Prescient” MPC (PMPC)

Future disturbance sequence $P_{req}(w(t+k|k))$
known in advance

Simulation results: controller comparison

Comparison on the NEDC cycle

	Fuel cons. [kg]	% Fuel improv.
FTMPC	0.281	–
SMPC (static)	0.243	13.5%
SMPC (adaptive)	0.199	29.2%
PMPC	0.197	29.9%

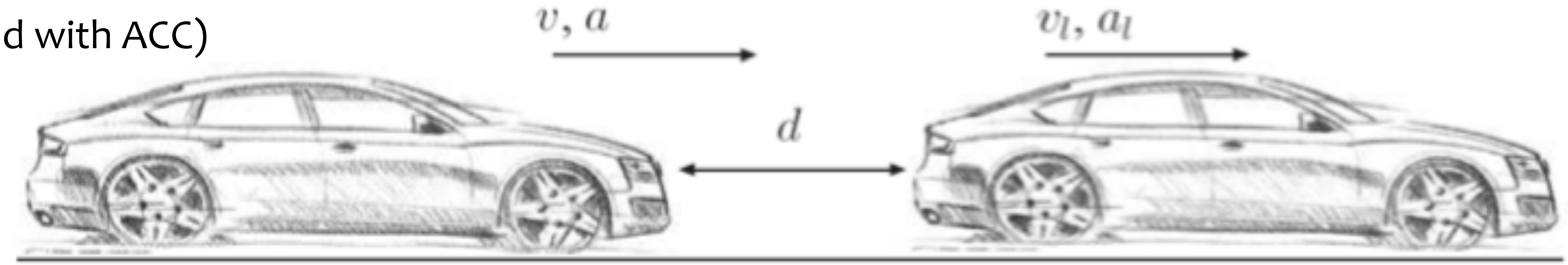
Comparison on different driving cycles
Fuel consumption (kg)

	NEDC	FTP	10-15 mode
FTMPC	0.281	0.533	0.125
1 MC	0.243	0.323	0.091
2 MC	0.224	0.323	0.089
2 MC adaptive	0.199	0.325	0.088
PMPC	0.197	0.320	0.071

pretty close to having the crystal ball.
But we don't, we just model uncertainty carefully

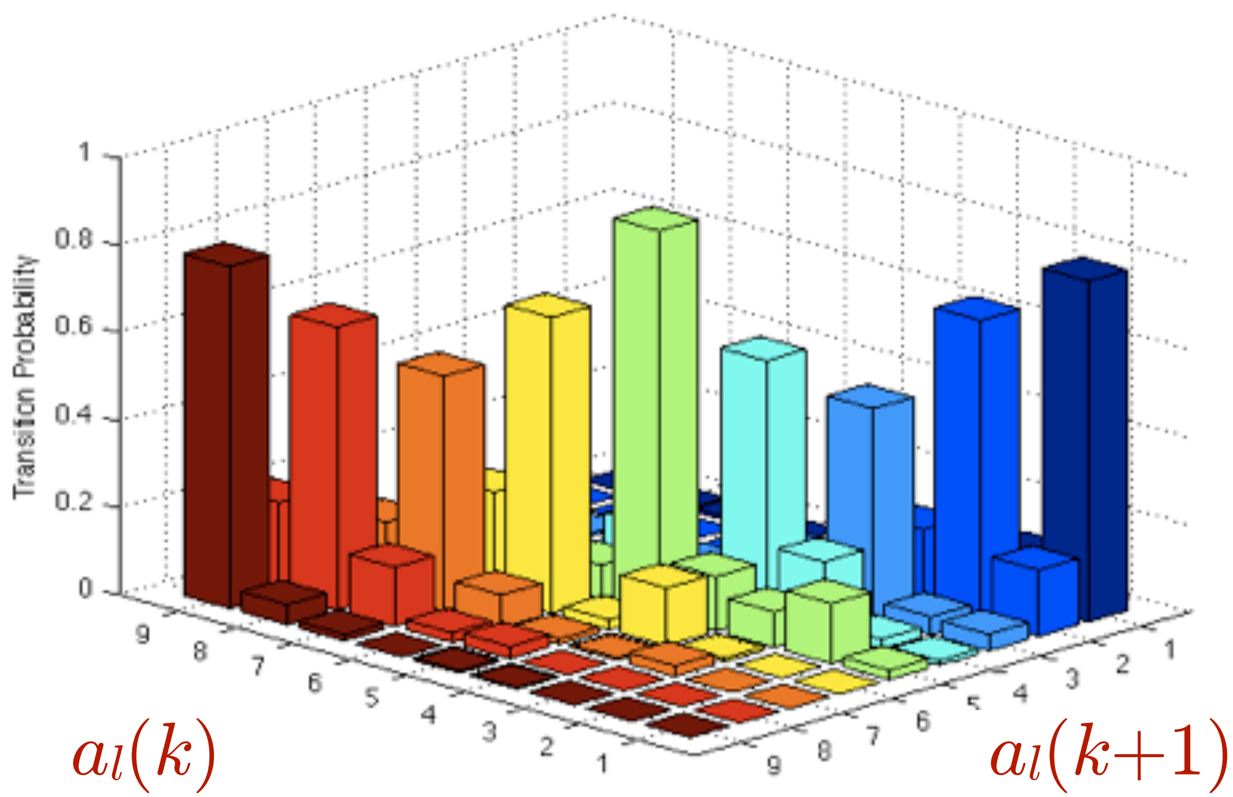
SMPC for ACC: stochastic leader model

Follower vehicle
(equipped with ACC)



Leader
vehicle

Leader acceleration a_l modeled by a Markov Chain (quantized in 9 states)

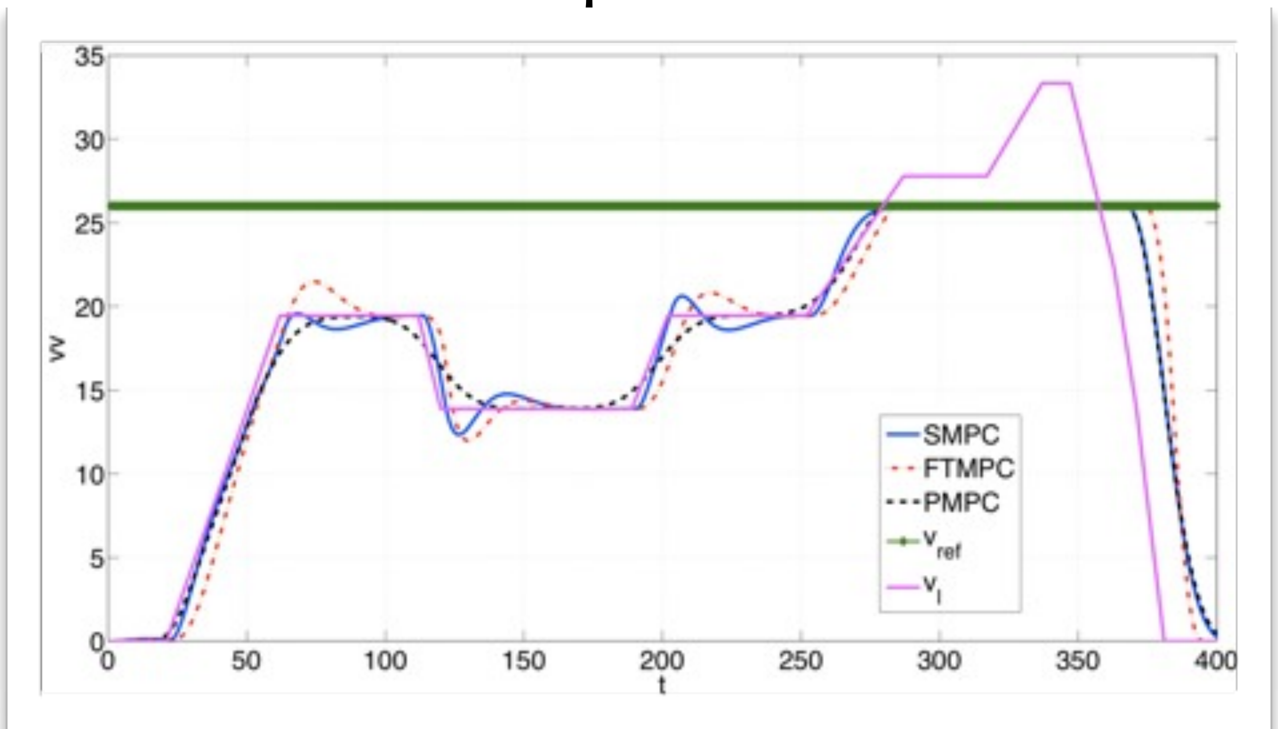


The Markov Chain is:

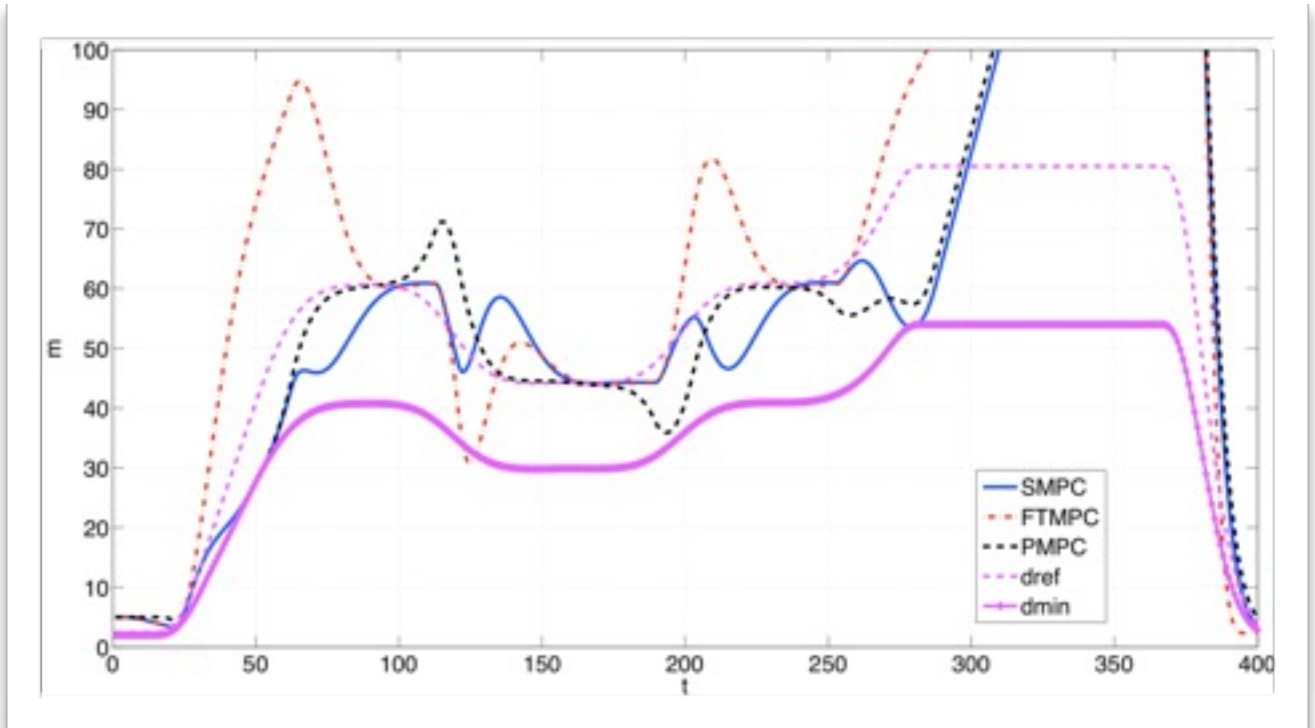
- Trained off-line on a collection of driving cycles (FTP, NEDC, 10-15 Mode)
- Adapted on-line by means of the learning algorithm

SMPC for ACC: simulation results

Speed



Distance



Stochastic MPC (blue solid line)

Frozen Time MPC (red dashed line)

Prescient MPC (black dashed line)

Simulation results on European Urban Driving Cycle (EUDC)

Conclusions

- **Linear and explicit MPC** can be implemented extremely efficiently (either in C on a processor, or on FPGA/ASIC circuits)
- **Linear time-varying MPC** can very effectively deal with nonlinearities (by on-line linearization of dynamics) and time varying systems
- **Stochastic MPC** provides a very good robustness vs. performance tradeoff and is easy implementable via QP

MPC is becoming an **increasingly mature technology** for a variety of industrial applications (not only for process control)

ODYS
OPTIMIZATION AND CONTROL ENGINEERING

A spinoff company of IMT Lucca
http://www.odys.it

Institutions Markets Technologies IMT INSTITUTE FOR ADVANCED STUDIES LUCCA