Optimization-based Control of Hybrid Dynamical Systems

Alberto Bemporad

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

• Models of hybrid systems
• Model predictive control of hybrid systems
• Explicit reformulation
• Automotive applications
Hybrid systems

\[ x \in \{1, 2, 3, 4, 5\} \]
\[ u \in \{A, B, C\} \]

**Computer Science**  
Finite state machines

**Control Theory**  
Continuous dynamical systems

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

\[
\begin{align*}
\frac{dx(t)}{dt} &= f(x(t), u(t)) \\
y(t) &= g(x(t), u(t))
\end{align*}
\]

\[
\begin{align*}
x(k+1) &= f(x(k), u(k)) \\
y(k) &= g(x(k), u(k))
\end{align*}
\]
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 to differential equations. The modifications required in otherwise continuous systems described by vector differential equations...
Embedded Systems

- Consumer electronics
- Home appliances
- Oce automation
- Automobiles
- Industrial plants
- ...
Motivation: “Intrinsically Hybrid”

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

- **Four-stroke engines**
  - Automaton, dependent on power train motion
Key Requirements for Hybrid Models

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

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

```
linear hybrid systems
```

“Make everything as simple as possible, but not simpler.”
— Albert Einstein
**Piecewise Ane Systems**

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

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

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

(Sontag 1981)

Can approximate nonlinear and/or discontinuous dynamics arbitrarily well.
Discrete Hybrid Automaton

(Torrisi, Bemporad, 2004)

\[ x_\ell \in \{0, 1\}^{m_b} = \text{binary states} \]
\[ u_\ell \in \{0, 1\}^{m_b} = \text{binary inputs} \]
\[ \delta_e \in \{0, 1\}^{m_e} = \text{event variables} \]

\[ x_c \in \mathbb{R}^{n_c} = \text{continuous states} \]
\[ u_c \in \mathbb{R}^{m_c} = \text{continuous inputs} \]
\[ i \in \{1, 2, \ldots, s\} = \text{current mode} \]
Switched Ane System

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

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

$x_c \in \mathbb{R}^{nc}$, $u_c \in \mathbb{R}^{mc}$
Event variables are generated by linear threshold conditions over continuous states, continuous inputs, and time:

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

\[ x_c \in \mathbb{R}^{nc}, \ u_c \in \mathbb{R}^{mc}, \ \delta_e \in \{0, 1\}^{ne} \]

Example: \([\delta(k) = 1] \iff [x_c(k) \geq 0]\)
The binary state of the finite state machine evolves according to a Boolean state update function:

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

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

Example:

\[ x_\ell(k + 1) = \neg \delta_e(k) \lor (x_\ell(k) \land u_\ell(k)) \]
The active mode $i(k)$ is selected by a Boolean function of the current binary states, binary inputs, and event variables:

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

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

Example:

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

<table>
<thead>
<tr>
<th>$u_\ell/x_\ell$</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
<td>$i = [\frac{1}{2}]$</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>$i = [\frac{1}{8}]$</td>
</tr>
</tbody>
</table>

The mode selector can be seen as the output function of the discrete dynamics. The system has 3 modes.
Logic and Inequalities

\[ X_1 \lor X_2 = \text{TRUE} \quad \Rightarrow \quad \delta_1 + \delta_2 \geq 1, \quad \delta_1, \delta_2 \in \{0, 1\} \]

Any logic statement
\[ f(X) = \text{TRUE} \]

\[
\begin{align*}
\bigwedge_{j=1}^{m} \left( \bigvee_{i \in P_j} X_i \lor \bigvee_{i \in N_j} \neg X_i \right) \quad (\text{CNF})
\end{align*}
\]

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

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

\[
\begin{align*}
H^i x_c(k) - W^i & \leq M^i (1 - \delta_e^i) \\
H^i x_c(k) - W^i & > m^i \delta_e^i
\end{align*}
\]

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

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

Finite State Machine
Mode Selector
Switched Ane System
Event Generator

(Glover 1975, Williams 1977, Hooker 2000)
Mixed Logical Dynamical Systems

Discrete Hybrid Automaton

Mixed Logical Dynamical (MLD) Systems

\[
\begin{align*}
    x(t+1) &= Ax(t) + B_1 u(t) + B_2 \delta(t) + B_3 z(t) + B_5 \\
    y(t) &= Cx(t) + D_1 u(t) + D_2 \delta(t) + D_3 z(t) + D_5 \\
    E_2 \delta(t) + E_3 z(t) &\leq E_4 x(t) + E_1 u(t) + E_5
\end{align*}
\]

- Continuous and binary variables

- Computationally oriented (mixed-integer programming)
- Suitable for *controller synthesis, verification*, ...
Hybrid Toolbox for Matlab

Features:

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

Support:

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 (linear) programming (MIP)

Example: Optimal multi-period investments for maintenance and upgrade of electrical energy distribution networks

(Bemporad, Muñoz, Piazzesi, 2006)

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

CPU time: 0.2 s
Major Advantages of Linear Hybrid Models

Many problems of analysis:
- Stability
- Safety / Reachability
- Observability
- Passivity

(Johansson, Rantzer, 1998)
(Torrisi, Bemporad, 2001)
(Bemporad, Ferrari-Trecate, Morari, 2000)
(Bemporad, Bianchini, Brogi, 2006)

Many problems of synthesis:
- Controller design
- Robust control design
- Filter design (state estimation/fault detection)

(Bemporad, Morari, 1999)
(Silva, Bemporad, Botto, Sá da Costa, 2003)
(Bemporad, Mignone, Morari, 1999)
(Ferrari-Trecate, Mignone, Morari, 2002)
(Pina, Botto, 2006)

can be solved through mathematical programming

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

✓ Models of hybrid systems

- Model predictive control of hybrid systems
- Explicit reformulation
- Automotive applications
Hybrid Control Problem

hybrid process

binary inputs
continuous inputs

continuous states
binary states

on-line decision maker

desired behavior
constraints

hybrid process

01 10 11 01

20
MPC for Hybrid Systems

- 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:

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

$$+ \sigma(\|\delta(t+k) - \delta_r\| + \|z(t+k) - z_r\| + \|x(t+k|t) - x_r\|)$$

subject to MLD model

$$x(t|t) = x(t)$$

$$x(t+T|t) = x_r$$

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

**Theorem 1** Let $(x_r, u_r, \delta_r, z_r)$ be the equilibrium values corresponding to the set point $r$, and assume $x(0)$ is such that the MPC problem is feasible at time $t = 0$. Then $\forall Q, R > 0, \forall \sigma > 0$

\[
\lim_{t \to \infty} y(t) = r \\
\lim_{t \to \infty} u(t) = u_r \\
\lim_{t \to \infty} x(t) = x_r, \lim_{t \to \infty} \delta(t) = \delta_r, \lim_{t \to \infty} z(t) = z_r,
\]
and all constraints are fulfilled.

(Bemporad, Morari 1999)

Proof: Easily follows from standard Lyapunov arguments

More stability results: see (Lazar, Heemels, Weiland, Bemporad, 2006)
Hybrid MPC - Example

PWA system:

\[
x(t + 1) = 0.8 \begin{bmatrix} \cos \alpha(t) & -\sin \alpha(t) \\ \sin \alpha(t) & \cos \alpha(t) \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(t) \\
y(t) = x_2(t)
\]

\[
\alpha(t) = \begin{cases} 
\frac{\pi}{6} & \text{if } x_1(t) > 0 \\
-\frac{\pi}{3} & \text{if } x_1(t) \leq 0 
\end{cases}
\]

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

Open loop behavior

/demos/hybrid/bm99sim.m
Hybrid MPC - Example

Closed loop:

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

Discrete-time Hybrid Stochastic Automaton (DHSA)

Stochastic Finite State Machine (sFSM)

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

\( k = \text{discrete-time counter} \)

All MPC techniques described earlier can be applied!

Event-based Continuous-time Hybrid Automaton (icHA)

Switched integral dynamics

\[ \frac{dx_c(t)}{dt} = B_i(t)u_c(t) + f_i(t) \]

\( k = \text{event counter} \)

(Bemporad, Di Cairano, HSCC'05)

(Bemporad, Di Cairano, Julvez, CDC05 & HSCC-06)
Optimal Control of Hybrid Systems: Computational Aspects
MPC for Linear Systems

Linear model

\[
\begin{align*}
    x_{k+1} &= Ax_k + Bu_k \\
y_k &= Cx_k
\end{align*}
\]

\[x_0 = x(t)\]

Quadratic performance index

\[
\min_U J(x(t), U) = \sum_{k=0}^{N-1} \left[ x_k'Qx_k + u_k'Ru_k \right] + x_N'Px_N
\]

Constraints

\[
\begin{align*}
    \min_U & \quad \frac{1}{2}U'HU + x'(t)F'U + \frac{1}{2}x(t)Yx(t) \\
    \text{subj. to} & \quad GU \leq W + Sx(t),
\end{align*}
\]

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

This is a \textbf{(convex) Quadratic Program (QP)}
MIQP Formulation of MPC

(Bemporad, Morari, 1999)

\[
\begin{align*}
\text{min}_{\xi} J(\xi, x(0)) &= \sum_{t=0}^{T-1} y'(t)Qy(t) + u'(t)Ru(t) \\
\text{subject to} & \\
& \begin{cases} 
  x(t+1) = Ax(t) + B_1 u(t) + B_2 \delta(t) + B_3 z(t) + B_5 \\
  y(t) = Cx(t) + D_1 u(t) + D_2 \delta(t) + D_3 z(t) + D_5 \\
  E_2 \delta(t) + E_3 z(t) \leq E_4 x(t) + E_1 u(t) + E_5
\end{cases}
\end{align*}
\]

- Optimization vector:

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

\[
\begin{align*}
\text{min} & \quad \frac{1}{2} \xi' H \xi + x(0)' F \xi + \frac{1}{2} x'(0) Y x(0) \\
\text{subj. to} & \quad G \xi \leq W + S x(t)
\end{align*}
\]

\[u \in \mathbb{R}^{nu}, \delta \in \{0, 1\}^{n\delta}, z \in \mathbb{R}^{nz}\] 

\[\xi \in \mathbb{R}^{(nu+nz)T} \times \{0, 1\}^{n\delta T}\]

\(\xi\) has both real and \{0, 1\} components
MILP Formulation of MPC

(Bemporad, Borrelli, Morari, 2000)

\[
\min_{\xi} J(\xi, x(0)) = \sum_{t=0}^{T-1} \|Qy(t)\|_\infty + \|Ru(t)\|_\infty \\
\text{subject to } \text{MLD model}
\]

- **Basic trick:** introduce slack variables:

\[
\min_x |x| \quad \Rightarrow \quad \min_{x, \epsilon} \epsilon \\
\text{s.t. } \epsilon \geq x \\
\epsilon \geq -x
\]

- **Generalization:**

\[
\left\{ \begin{array}{l}
\epsilon_k^x \geq \|Qy(t + k|t)\|_\infty \\
\epsilon_k^u \geq \|Ru(t + k)\|_\infty
\end{array} \right.
\]

- **Optimization vector:**

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

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

\(\xi\) has both real and \{0, 1\} components.
Mixed-Integer Program Solvers

- Mixed-Integer Programming is NP-hard

**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](http://plato.la.asu.edu/bench.html) for a comparison)

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

- Possibility of combining symbolic + numerical solvers
  Example: SAT + linear programming

(Bemporad, Giorgetti, IEEE TAC 2006)
Contents

✓ Models of hybrid systems
✓ Model predictive control of hybrid systems

- Explicit reformulation
- Automotive applications
- Hybrid MPC over wireless sensor networks
On-Line vs O-Line Optimization

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!

O-line optimization: solve the MILP/MIQP for all $x(t)$

multi-parametric programming
Example of Multiparametric Solution

Multiparametric LP \( \emptyset \mathbb{R}^2 \)

\[
\begin{align*}
\min_{\xi} & \quad -3\xi_1 - 8\xi_2 \\
\text{s.t.} & \quad \xi_1 + \xi_2 \leq 13 + x_1 \\
& \quad 5\xi_1 - 4\xi_2 \leq 20 \\
& \quad -8\xi_1 + 22\xi_2 \leq 121 + x_2 \\
& \quad -4\xi_1 - \xi_2 \leq -8 \\
& \quad -\xi_1 \leq 0 \\
& \quad -\xi_2 \leq 0
\end{align*}
\]

\[
\xi(x) = \begin{cases} 
[0.00\ 0.05] x + [11.85] & \text{if } \begin{bmatrix} 0.02 & 0.00 \\ 0.00 & 0.02 \\ -0.12 & -0.02 \\ -0.15 & 0.01 \end{bmatrix} x \leq \begin{bmatrix} 1.00 \\ 1.00 \\ 1.00 \\ -1.00 \end{bmatrix} \quad \text{CR}_{1,4} \\
[0.73\ -0.03] x + [5.50] & \text{if } \begin{bmatrix} 0.00 & 0.02 \\ 0.00 & -0.02 \\ 0.12 & -0.01 \\ -0.15 & 0.00 \end{bmatrix} x \leq \begin{bmatrix} 1.00 \\ 1.00 \\ 1.00 \\ 1.00 \end{bmatrix} \quad \text{CR}_{1,3} \\
[-0.33\ 0.00] x + [-1.67] & \text{if } \begin{bmatrix} 0.00 & 0.02 \\ 0.00 & -0.02 \\ 0.15 & -0.00 \\ -0.09 & 0.00 \end{bmatrix} x \leq \begin{bmatrix} 1.00 \\ 1.00 \\ -1.00 \\ 1.00 \end{bmatrix} \quad \text{CR}_{1,4}
\end{cases}
\]
MPC for Linear Systems

Linear model

\[
\begin{align*}
\begin{cases}
x_{k+1} &= Ax_k + Bu_k \\
y_k &= Cx_k
\end{cases}
\end{align*}
\]

\[x_0 = x(t)\]

Quadratic performance index

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

\[
\begin{array}{c}
\min_U \\
\text{subj. to}
\end{array}
\begin{align*}
\frac{1}{2} U' H U + x'(t) F' U + \frac{1}{2} x(t) Y x(t) \\
G U &\leq W + S x(t),
\end{align*}
\]

Constraints

\[
\begin{cases}
\underline{u} \leq u_k \leq \overline{u} \\
\underline{y} \leq y_k \leq \overline{y}
\end{cases}
\]

Objective: solve the QP for all \( x(t) \in X \subseteq \mathbb{R}^n \) (o-line)
## Properties of multiparametric-QP

(Bemporad et al., 2002)

<table>
<thead>
<tr>
<th>Optimizer</th>
<th>[ U^*(x) = \arg \min_U \frac{1}{2} U' H U + x' F' U ] subj. to [ G U \leq W + S x ]</th>
<th>continuous, piecewise affine</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value function</td>
<td>[ V^*(x) = \frac{1}{2} x' Y x + \min_U \frac{1}{2} U' H U + x' F' U ] subj. to [ G U \leq W + S x ]</td>
<td>convex continuous, piecewise quadratic, ( C^1 ) (if no degeneracy)</td>
</tr>
<tr>
<td>Feasible state set</td>
<td>( X^* = { x : \exists U \text{ such that } G x \leq W + S U } )</td>
<td>convex polyhedral</td>
</tr>
</tbody>
</table>

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

\[
 u(x) = \begin{cases} 
 F_1 x + g_1 & \text{if } H_1 x \leq K_1 \\
 \vdots & \vdots \\
 F_M x + g_M & \text{if } H_M x \leq K_M 
\end{cases}
\]
Polyhedral Invariant Sets for Closed-loop Linear MPC Systems

- Convexity of value function implies convexity of the piecewise ellipsoidal sets \( \{ x : V^*(x) \leq \gamma \} \) and \( \{ x : V^*(x) - x'Qx \leq \gamma \} \).

- Any polyhedron \( P \) contained in between is a positively invariant set for the closed-loop MPC system.

- By changing \( \gamma \) invariant polyhedra of arbitrary size can be constructed for the closed-loop MPC system.

(Alessio, Bemporad, Lazar, Heemels, CDC’06)

(Note: explicit form of MPC not required)
Explicit Hybrid MPC (PWA)

\[
\min_U J(U, x, r) = \sum_{k=0}^{T-1} \|R(y(k) - r)\|_p + \|Qu(k)\|_p
\]
subject to \(\text{PWA model}\)
\[
x(0) = x
\]

- The MPC controller is \textit{piecewise ane} in \(x, r\)

\[
u(x, r) = \begin{cases} 
F_1x + E_1r + g_1 & \text{if } H_1 \left[ \frac{x}{r} \right] \leq K_1 \\
\vdots & \vdots \\
F_Mx + E_Mr + g_M & \text{if } H_M \left[ \frac{x}{r} \right] \leq K_M
\end{cases}
\]

(Borrelli, Baotic, Bemporad, Morari, \textit{Automatica}, 2005)

(Mayne, ECC 2001) (Alessio, Bemporad, ADHS 2006)

\textbf{Note:} in the 2-norm case the partition may not be fully polyhedral
Hybrid Control - Example

PWA system:

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

\[
y(t) = x_2(t)
\]

\[
\alpha(t) = \begin{cases} -\frac{\pi}{3} & \text{if } x_1(t) \geq 0 \\ \frac{\pi}{3} & \text{if } x_1(t) < 0 \end{cases}
\]

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} 
0.6928 -0.4 & 1 \begin{bmatrix} x \\ r \end{bmatrix} & \text{if} \\
\begin{bmatrix} 0.6928 & -0.4 & 1 \\
-0.4 & -0.6928 & 0 \\
0 & 0 & -1 \\
-0.6928 & 0.4 & -1 \\
0 & 0 & 1 \\
0 & 1 & 0 \\
\end{bmatrix} \begin{bmatrix} x \\ r \end{bmatrix} \leq 1 \\
0 & 1 & 1 & 0 & 1 & 0 \\
1 & 0 & 0 & 1 & 0 & 1 \\
1 & 1 & 0 & 0 & 1 & 0 \\
1 & 0 & 0 & 1 & 0 & 1 \\
1 & 1 & 0 & 0 & 1 & 0 \\
1 & 1 & 0 & 0 & 1 & 0 \\
\end{bmatrix} \\
(\text{Region } \#1) \\
1 & \begin{bmatrix} 0.6928 & 0.4 & -1 \\
0.6928 & 0.4 & -1 \\
0 & 0 & 1 \\
0 & 0 & -1 \\
-1 & 0 & 0 \\
-1 & 0 & 1 \\
\end{bmatrix} \begin{bmatrix} x \\ r \end{bmatrix} \leq 1 \\
0 & 1 & 1 & 0 & 1 & 0 \\
1 & 0 & 0 & 1 & 0 & 1 \\
1 & 1 & 0 & 0 & 1 & 0 \\
1 & 1 & 0 & 0 & 1 & 0 \\
1 & 1 & 0 & 0 & 1 & 0 \\
1 & 1 & 0 & 0 & 1 & 0 \\
\end{bmatrix} \\
(\text{Region } \#2) \\
-1 & \begin{bmatrix} -0.4 & -0.6928 & 0 \\
0 & 0 & 1 \\
0 & 0 & -1 \\
-0.6928 & -0.4 & 1 \\
0 & 0 & 1 \\
0 & 0 & -1 \\
\end{bmatrix} \begin{bmatrix} x \\ r \end{bmatrix} \leq 1 \\
0 & 1 & 1 & 0 & 1 & 0 \\
1 & 0 & 0 & 1 & 0 & 1 \\
1 & 1 & 0 & 0 & 1 & 0 \\
1 & 1 & 0 & 0 & 1 & 0 \\
1 & 1 & 0 & 0 & 1 & 0 \\
1 & 1 & 0 & 0 & 1 & 0 \\
\end{bmatrix} \\
(\text{Region } \#3) \\
-1 & \begin{bmatrix} -1 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & 1 \\
-1 & 0 & 0 \\
0 & 0 & 1 \\
0 & 0 & 1 \\
\end{bmatrix} \begin{bmatrix} x \\ r \end{bmatrix} \leq 1 \\
0 & 1 & 1 & 0 & 1 & 0 \\
1 & 0 & 0 & 1 & 0 & 1 \\
1 & 1 & 0 & 0 & 1 & 0 \\
1 & 1 & 0 & 0 & 1 & 0 \\
1 & 1 & 0 & 0 & 1 & 0 \\
1 & 1 & 0 & 0 & 1 & 0 \\
\end{bmatrix} \\
(\text{Region } \#4) \\
-1 & \begin{bmatrix} 0.6928 & -0.4 & 1 \\
-0.6928 & 0.4 & -1 \\
0 & 0 & 1 \\
0 & 0 & -1 \\
-1 & 0 & 0 \\
-1 & 0 & 1 \\
\end{bmatrix} \begin{bmatrix} x \\ r \end{bmatrix} \leq 1 \\
0 & 1 & 1 & 0 & 1 & 0 \\
1 & 0 & 0 & 1 & 0 & 1 \\
1 & 1 & 0 & 0 & 1 & 0 \\
1 & 1 & 0 & 0 & 1 & 0 \\
1 & 1 & 0 & 0 & 1 & 0 \\
1 & 1 & 0 & 0 & 1 & 0 \\
\end{bmatrix} \\
(\text{Region } \#5) \\
0.6928 -0.4 & 1 \begin{bmatrix} x \\ r \end{bmatrix} & \text{if} \\
\end{cases}
\]

HybTbx: /demos/hybrid/bm99sim.m

(CPU time: 1.51 s, Pentium M 1.4GHz)
Hybrid MPC - Example

Closed loop:
Explicit PWA Regulator

Objective: \[
\min \sum_{k=1}^{N} \|x(t + k|t)\|_{\infty}
\]
Comments on Explicit Solutions

- Alternatives: either (1) solve an MIP on-line or (2) evaluate a PWA function

- For problems with many variables and/or long horizons: MIP may be preferable

- For simple problems (short horizon/few constraints):
  - 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
Contents

✓ Models of hybrid systems
✓ Model predictive control of hybrid systems
✓ Explicit reformulation

● Automotive applications
Hybrid Control of a DISC Engine

(joint work with N. Giorgetti, G. Ripaccioli, I. Kolmanovsky, and D. Hrovat)
DISC Engine

Two distinct regimes:

<table>
<thead>
<tr>
<th>Regime</th>
<th>fuel injection</th>
<th>air-to-fuel ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>Homogeneous combustion</td>
<td>intake stroke</td>
<td>$\lambda=14.64$</td>
</tr>
<tr>
<td>Stratified combustion</td>
<td>compression stroke</td>
<td>$\lambda &gt; 14.64$</td>
</tr>
</tbody>
</table>

**Objective:** Design a controller for the engine that

- Automatically choose operating **mode** (homogeneous/stratified)
- Can cope with **nonlinear** dynamics
- Handles **constraints** (on A/F ratio, air-flow, spark)
- Achieves **optimal** performance
Hybridization of DISC Dynamics

- Proprietary **nonlinear** model of the DISC engine developed and validated at Ford Research Labs (Dearborn) \((Kolmanovsky, Sun, ...)\)
- Model good for simulation, not good for control design!

**MODEL HYBRIDIZATION**

**DYNAMICS** (intake pressure, air-to-fuel ratio, torque):

- Definition of two operating points;
- Numerical linearization of nonlinear dynamics;
- Time discretization of the linear models.

**CONSTRAINTS** on:

- Air-to-Fuel Ratio: \(\lambda_{\text{min}}(\rho) \leq \lambda(t) \leq \lambda_{\text{max}}(\rho)\);
- Mass of air through the throttle: \(0 \leq W_{\text{th}} \leq K\);
- Spark timing: \(0 \leq \delta(t) \leq \delta_{\text{mbt}}(\lambda, \rho)\)

Hybrid system with 2 modes (switched and system)
Integral Action

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

\[
\begin{align*}
\epsilon_\tau(t+1) &= \epsilon_\tau(t) + T \cdot (\tau_{ref} - \tau) \\
\epsilon_\lambda(t+1) &= \epsilon_\lambda(t) + T \cdot (\lambda_{ref} - \lambda)
\end{align*}
\]

\( T = \text{sampling time} \)

\( \tau_{ref}, \lambda_{ref} \) brake torque and air-to-fuel references

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

( despite the model mismatch )

Reference values are automatically generated from \( \tau_{ref} \) and \( \lambda_{ref} \)

by numerical computation based on the nonlinear model
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 111*pm+112*Wth... +113*Wf+114*delta+11c ELSE 101*pm+102*Wth+103*Wf... +104*delta+10c };  
            taul={IF rho THEN taul1*pm+... taul2*Wth+taul3*Wf+taul4*delta+taulc ELSE taul01*pm+taul02*Wth... +taul03*Wf+taul04*delta+taul0c };  
            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

\[
\min_{\xi} J(\xi, x(t)) = \sum_{k=0}^{N-1} u'_k R u_k + y'_k Q y_k + x'_{k+1} S x_{k+1}
\]

subj. to \[
\begin{align*}
    x_0 &= x(t), \\
    \text{hybrid model}
\end{align*}
\]

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

Weights:

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

\[
Q = \begin{pmatrix}
    1 & 0 & 0 & 0 \\
    0 & 0.001 & 0 & 0 \\
    0 & 0 & 0.01 & 0 \\
    0 & 0 & 0 & 0
\end{pmatrix}
\]

\[
S = \begin{pmatrix}
    0.04 & 0 & 0 & 0 \\
    0 & 0 & 0 & 0 \\
    1500 & 0 & 0 & 0 \\
    0.01 & 0 & 0 & 0
\end{pmatrix}
\]

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

(prevents unneeded chattering)

main emphasis on torque
Simulation Results (nominal engine speed)

Engine Brake Torque

\[ \omega = 2000 \text{ rpm} \]

Air-to-Fuel Ratio

Combustion mode

- Control horizon \( N = 1 \);
- Sampling time \( T_s = 10 \text{ ms} \);
- PC Xeon 2.8 GHz + Cplex 9.1

\[ \simeq 3 \text{ ms per time step} \]
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

Control code too complex (MILP) ⇒ not implementable!
Explicit MPC Controller

Explicit control law:

\[ u(t) = f(\theta(t)) \]

where:

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

\[ \theta = [p_m \varepsilon_\tau \varepsilon_\lambda \tau_{ref} \lambda_{ref} \rho_{m,ref} W_{th,ref} W_{f,ref} \delta_{ref}]' \]

N=1 (control horizon)

75 partitions

Cross-section by the \( \tau_{ref}-\lambda_{ref} \) plane

- Time to compute the explicit MPC: 3.4750 s;
- Sampling time \( T_s=10 \) ms;
- PC Xeon 2.8 GHz + Cplex 9.1

\[ \approx 8 \mu s \text{ per time step} \]
Microcontroller Implementation

- C-code automatically generated by the Hybrid Toolbox
- Microcontroller Motorola MPC 555 (custom made for Ford)
- 43 Kb memory available
- Floating point arithmetic

≈ 3ms execution time
sampling period = 10ms

Implementable!

- Further reduction of number of partitions possible
- C-code can be further optimized

(Alessio, Bemporad, 2005)
(Tøndel, Johansen, Bemporad, 2003)
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 o-line via multiparametric programming for fast-sampling applications.

\[ u(x,r) = \begin{cases} 
    F_1x + E_1r + g_1 & \text{if } H_1[r] \leq K_1 \\
    \vdots & \vdots \\
    F_Mx + E_Mr + g_M & \text{if } H_M[r] \leq K_M 
\end{cases} \]
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)
Challenges in Wireless MPC

- Synchronization schemes must ensure correct prediction in spite of packet loss

- MPC algorithm must be robust w.r.t. packet loss
  ⇒ stochastic hybrid MPC, robust hybrid MPC

- Wireless sensors must be interfaced to optimization tools
Demo Application in Wireless Automation

(Automatic Control Lab, Univ. Siena)

- Telos motes provide wireless temperature feedback in Matlab

- Hybrid MPC algorithm adjust belt speed and coordinate linear motors (via Simulink/xPC-Target link)
Acknowledgements


Announcements


● 2nd PhD School on Hybrid Systems, Siena, 2007