Acknowledgments

- Francesco Borrelli
- Mike Fodor
- Davor Hrovat
- Mitch McConnell

Ford Motor Company

Hybrid Control Example: Cruise Control System

Hybrid Control Problem

Renault Clio 1.9 DTI RXE

GOAL:

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

Hybrid Model

- Vehicle dynamics

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

$\dot{x}$ = vehicle speed

F_e = traction force

F_b = brake force

⟹ discretized with sampling time $T_s = 0.5 \text{ s}$

- Transmission kinematics

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

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

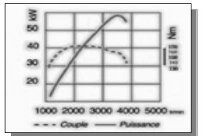
ω = engine speed

M = engine torque

i = gear

Hybrid Model

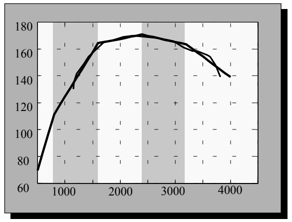
- Engine torque $\boxed{-C_e^-(\omega) \leq M \leq C_e^+(\omega)}$
- Max engine torque $\boxed{C_e^+(\omega)}$



http://www.renault.fr

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

requires: 4 binary aux variables
4 continuous aux variables



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

Hybrid Model

- Gear selection: for each gear #i, define a binary input $\boxed{g_i \in \{0, 1\}}$

- Gear selection (traction force):

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

define auxiliary continuous variables:

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

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

- Gear selection (engine/vehicle speed):

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

Hybrid Model

- MLD model

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

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

Hysdel Model

```
SYSTEM car {

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

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

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

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

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

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

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

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

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

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

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

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

Hybrid Controller

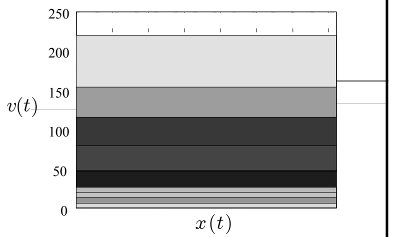
- Max-speed controller

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

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

MILP optimization problem

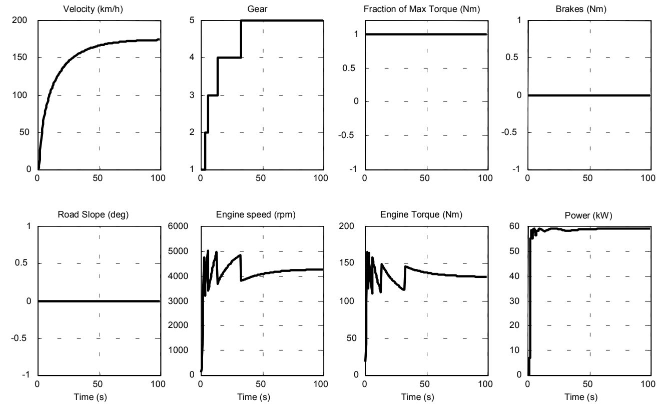
Linear constraints	96
Continuous variables	18
Binary variables	10
Parameters	1
Time to solve mp-MILP (Sun Ultra 10)	45 s
Number of regions	**11**



($x(t)$ is irrelevant)

Hybrid Controller

- Max-speed controller



Hybrid Controller

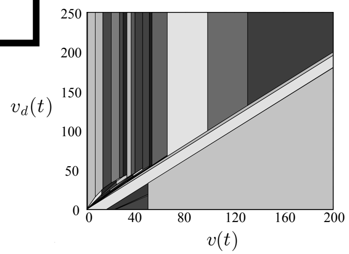
- Tracking controller

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

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

MILP optimization problem

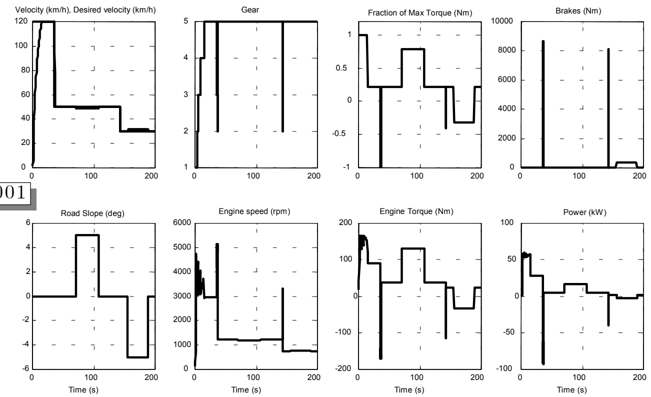
Linear constraints	98
Continuous variables	19
Binary variables	10
Parameters	2
Time to solve mp-MILP (Sun Ultra 10)	27 m
Number of regions	**49**



Hybrid Controller

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

$\rho = 0.001$



Hybrid Controller

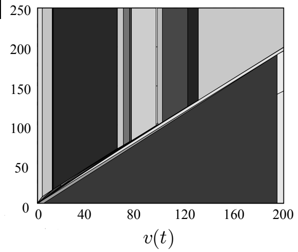
- Smoother tracking controller

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

$$\text{subj. to} \begin{cases} |v(t+1|t) - v(t)| < T_s a_{\max} \\ \text{MLD model} \\ x(t|t) = x(t) \end{cases}$$

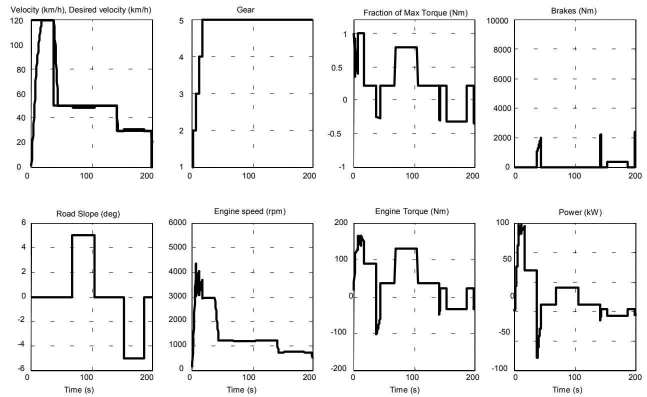
MILP optimization problem

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



Hybrid Controller

- Smoother tracking controller



Reachability Analysis (Verification)

Verification

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

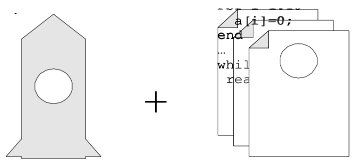
- **CERTIFY** that such combination behaves as desired
  - for ALL initial conditions within a given set
  - for ALL disturbances within a given class

- or **PROVIDE** a counterexample.

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

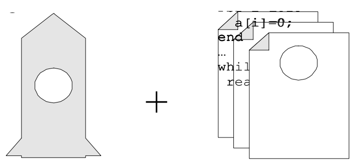
Reachability Analysis: provides the answer.

Ariane 5 - 1996



Physical beaviour　　Computer code

•Success

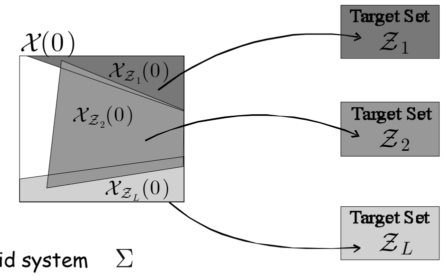


•Failure

Need for verification

Reachability Analysis/Verification

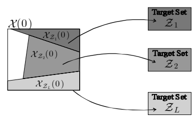
(Bemporad, Torrisi, Morari, 2000)



• Given:
- A hybrid system Σ
- A set of initial conditions $\mathcal{X}(0)$
- Target sets $\mathcal{Z}_1, \mathcal{Z}_2, \ldots, \mathcal{Z}_L$ (disjoint)
- Time horizon $t \le T_{\max}$

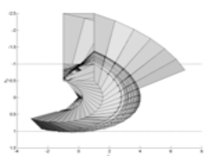
• Problem:
- Is $\mathcal{Z}_i$ reachable from $\mathcal{X}(0)$ in t steps ?
- If yes, from which subset $\mathcal{X}_{\mathcal{Z}_i}(0)$ of $\mathcal{X}(0)$
- Disturbance/input sequences driving $\mathcal{X}_{\mathcal{Z}_i}(0)$ to $\mathcal{Z}_i$

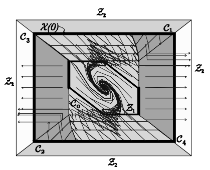
Applications

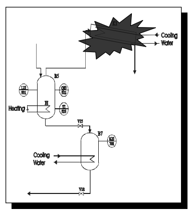


- Safety ($\mathcal{Z}_1, \mathcal{Z}_2, \ldots, \mathcal{Z}_L$ = unsafe sets)
- Stability ($\mathcal{Z}_1$ = invariant set around the origin)
- Scheduling ($u(0), \ldots, u(T_{\max})$ = optimal strategy,

  $\mathcal{Z}_1$ = reference set) (Bemporad, Giovanardi, Torrisi, CDC 2000)
- Liveness ($\mathcal{Z}_1$ = set to be reached within a finite time)
- Robust Simulation







Simple Verification Algorithm

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

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

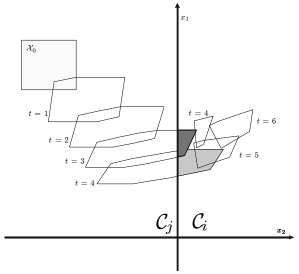
with respect to $x(0), \{u(t), \delta(t), z(t)\}_{t=0}^{T}$

• Only practical for small problems ! because number of free integer variables $\delta(0), \delta(1), \ldots, \delta(T)$ grows with T

• **Efficient Solution**: Exploit the special structure of the problem.
(Bemporad, Torrisi, Morari, HSCC 2000)

Reach-Set Evolution

- $T_{\max} < \infty$, discrete-time model
  - → **Decidable !**
- NP-hard !
  because of integer variables



Reachability analysis algorithm:

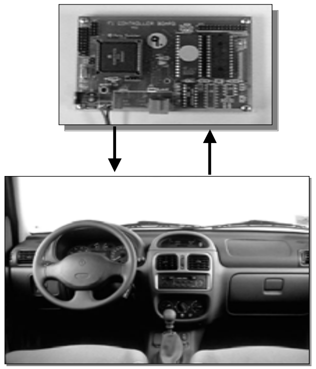
- Compute the reach set $\mathcal{X}(t)$ (linear dynamics) (polyhedral sets)
- Switching detection using Mixed Integer Programming
- Describe new intersections $\mathcal{X}(t) \bigcap \mathcal{C}_j$ as unions of hyperboxes
- Stopping criteria for a single exploration

(Bemporad, Torrisi, Morari, 2000)

Verification Example: Cruise Control System

Cruise Control System

Renault Clio 1.9 DTI RXE



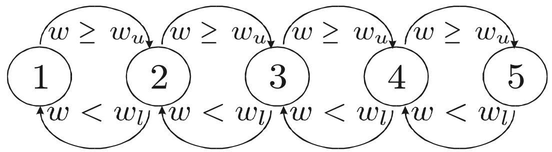
GOAL:

Verify if a given switching controller satisfies certain specifications

(Torrisi, Bemporad, 2001)

Cruise Control System

Gear selector:



Speed controller:

$$e(t+1) = e(t) + T_s(v_r(t) - v(t))$$ + saturation

$$u_t(t) = \begin{cases} k_t(v_r(t) - v(t)) + i_t e(t) & \text{if } v(t) < v_r(t) + 1 \\ 0 & \text{otherwise} \end{cases}$$

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

Hysdel Model (HYbrid Systems DEscription Language)

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

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

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

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

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

http://control.ethz.ch/~hybrid/hysdel (Torrisi, Bemporad, Mignone, 2000)

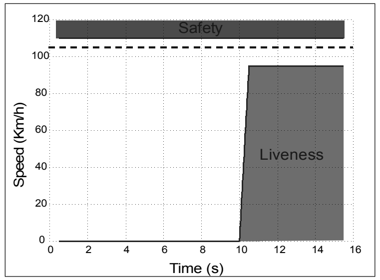
Hybrid Model

- MLD model

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

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

Verification

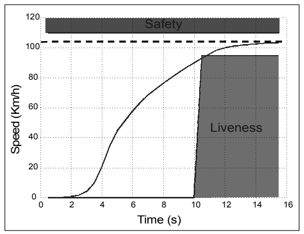


Verify that the cruise control reaches the desired speed reference ($\mathcal{Z}_2 = \{v, t : v < v_r - 2r_{\mathrm{toll}}, t > 10/T_s\}$) and without driving above the limit ($\mathcal{Z}_1 = \{v : v > v_r + r_{\mathrm{toll}}\}$)

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

Verification Results

- For all $v_r \in [30, 70]\ km/h$ the controller satisfies both liveness & safety properties
- CPU time: ~2.5h on Matlab5.3, PC650MHz.



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

Conclusions

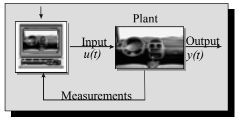
- **Hybrid systems** as a framework for new applications, where both logic and continuous dynamics are relevant
- **Mixed Logical Dynamical (MLD)** systems as discrete-time, *computation-oriented* models for hybrid systems

HYSDEL

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

Conclusions

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



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

Conclusions

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

References

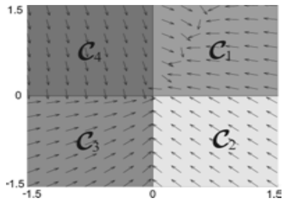
[1] A. Bemporad and M. Morari.***Control of systems integrating logic, dynamics, and constraints***. Automatica, 35(3):407 –427, March 1999.
[2] A. Bemporad, G.Ferrari-Trecate, and M.Morari. ***Observability and controllability of piecewise affine and hybrid systems***. IEEE Transactions Automatic Control, vol. 45, n.10, pp. 1864-1876, October 2000.
[3] A. Bemporad, F.D. Torrisi, M. Morari. ***Optimization-based verification and stability characterization of piecewise affine and hybrid systems***. In Hybrid Systems:Computation and Control, Lecture Notes in Computer Science 1790, pages 45–58. Springer Verlag, 1999.
[4] A. Bemporad, F. Borrelli, M. Morari. ***Piecewise linear optimal controllers for hybrid systems.*** American Control Conference, Chicago, pp. 1190-1194, 2000.
[5] A. Bemporad, D. Mignone, and M. Morari. ***Moving horizon estimation for hybrid systems an fault detection***. In Proc. American Contr. Conf., 1999.
[6] M. Heemels, B. De Schutter, A. Bemporad. ***Equivalence of hybrid dynamical models***. Automatica, vol. 37, n.7, pp. 1085-1091, 2001
[7] A Bemporad, M. Morari, V. Dua, E.N. Pistikopoulos. ***The explicit linear quadratic regulator for constrained systems***. Automatica, to appear.
[8] A. Bemporad, M. Morari. ***Optimization-based hybrid control tools***. In Proc. American Contr. Conf., 2001 (***survey paper***).

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

Stability Analysis

Stability of PWA systems

$$x_{k+1} = \begin{cases} \begin{pmatrix} -0.04 & -0.46 \\ -0.14 & 0.34 \end{pmatrix} x_k & \text{if} \quad x_k \in \mathcal{C}_1 \\ \begin{pmatrix} 0.94 & 0.32 \\ 0.79 & -0.05 \end{pmatrix} x_k & \text{if} \quad x_k \in \mathcal{C}_2 \\ \begin{pmatrix} -0.86 & 0.81 \\ 0.49 & 0.62 \end{pmatrix} x_k & \text{if} \quad x_k \in \mathcal{C}_3 \\ \begin{pmatrix} -0.02 & 0.64 \\ 0.76 & 0.27 \end{pmatrix} x_k & \text{if} \quad x_k \in \mathcal{C}_4 \end{cases}$$



Problem: check if the origin of a PWA system is asymptotically stable and determine a region of attraction.

Difficulties:

- *Not possible to deduce stability from stability of subsystems* (Branicky, 1995)
- *In general stability is either undecidable or NP-complete* (Blondel & Tsitsiklis, 1999)

Look for sufficient conditions for stability

Quadratic Lyapunov Functions

$$V(x) = x'Px \quad x \in \mathcal{C}_i \quad P = P' > 0$$

Theorem. If there exists P such that

$$V(x) = x'Px \quad A_j'PA_j - P < 0 \qquad (1)$$

the origin of the PWA system is exponentially stable.

- Solvable as a *Linear Matrix Inequalities problem* via Semidefinite Programming (interior-point methods) (Boyd,Vandenberghe, 1996)
- Sometimes too conservative

Piecewise Quadratic Lyapunov Functions

(Johannson, Rantzer, 1998)

$$V(x) = x'P_ix \quad x \in \mathcal{C}_i \quad P_i = P_i' > 0 \quad i = 1, \ldots, s$$

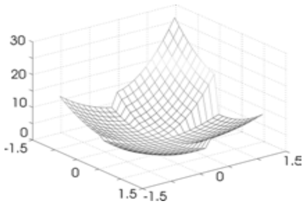
Theorem. If there exists P_i such that (Mignone, Ferrari-Trecate, Morari, 2000)

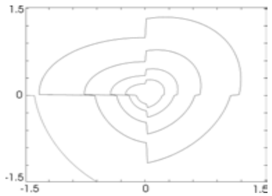
$$A_j'P_iA_j - P_j < 0 \qquad \forall (i,j) \in \mathcal{S} \qquad (1)$$

the origin of the PWA system is exponentially stable.

- $\mathcal{S}$: set of one-step switches between different regions (easily computed via *reachability analysis*)
- Solvable as a *Linear Matrix Inequalities problem* via Semidefinite Programming (interior-point methods) (Boyd,Vandenberghe, 1996)
- Explicit computation of a *discontinuos Lyapunov function* and characterization of the region of attraction.
- The LMIs for analysis can be adapted to synthesize a stabilizing *piecewise linear state-feedback* $u(k) = K_i x(k)$

Piecewise Quadratic Lyapunov Function





The End