Reachability Analysis of Piecewise Affine Systems Using Polytopes

F. D. Torrisi

torrisi@aut.ee.ethz.ch, http://control.ethz.ch/~torrisi

Currenty with esmertec AG http://www.esmertec.com

Joint Work with

A.Bemporad

A Motivating Example

Renault Clio 1.9 DTI RXE

Hybrid: Continuous (accelerator pedal, and brakes) and discrete (gear ratio) inputs

Cruise Control System

Gear selector:

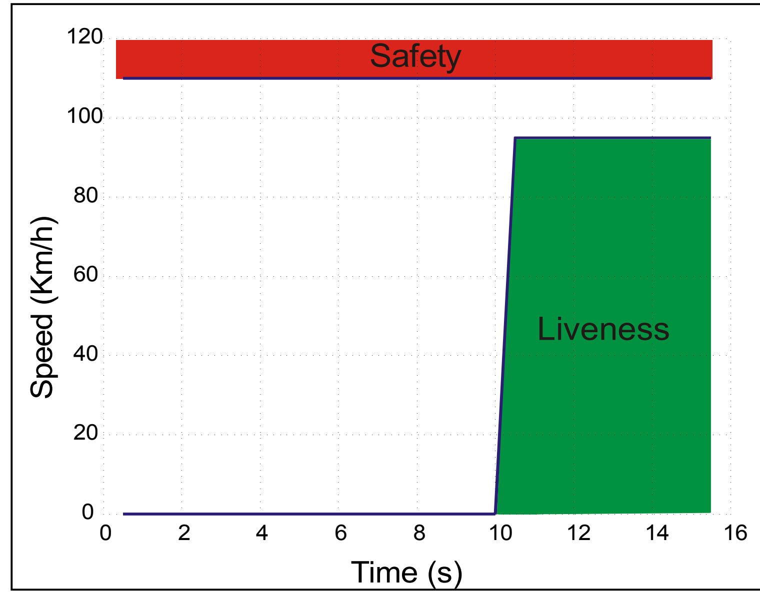
$$1 \;\underset{w < w_l}{\overset{w \geq w_u}{\rightleftarrows}}\; 2 \;\underset{w < w_l}{\overset{w \geq w_u}{\rightleftarrows}}\; 3 \;\underset{w < w_l}{\overset{w \geq w_u}{\rightleftarrows}}\; 4 \;\underset{w < w_l}{\overset{w \geq w_u}{\rightleftarrows}}\; 5$$

Speed controller:

$$\begin{aligned}
e(t+1) &= e(t) + T_s(v_r(t) - v(t)) \\
u_e(t) &= \begin{cases} k_e(v_e(t) - v(t)) + i_e e(t) & \text{if } v(t) < v_r + 1 \\ 0 & \text{otherwise} \end{cases} \\
u_b(t) &= \begin{cases} k_b(v_e(t) - v(t)) & \text{if } v(t) \geq v_r + 1 \\ 0 & \text{otherwise} \end{cases}
\end{aligned}$$

Problem: Verification

Question: Will the cruise control reach the desired speed reference within 10 s without exceeding the speed limit?



Safety $\mathcal{Z}_1 = \{v : v > v_r + r_{\text{toll}}\}$

Liveness $\mathcal{Z}_2 = \{v, t : v < v_r - 2r_{\text{toll}}, t > 10/T_s\}$

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

Modeling Requirements

Model:

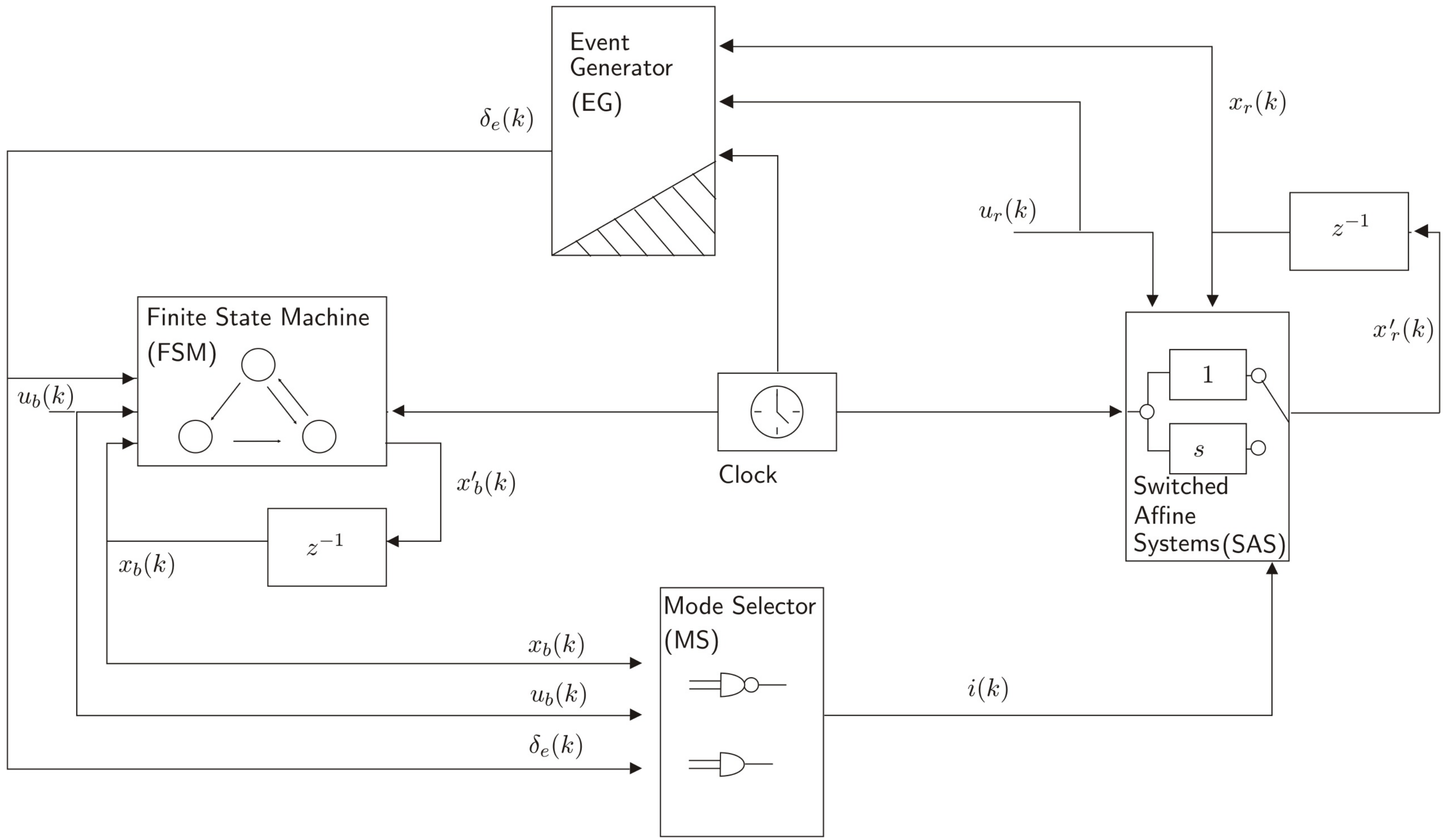
Detailed to capture the system behavior

Simple to efficiently solve problems

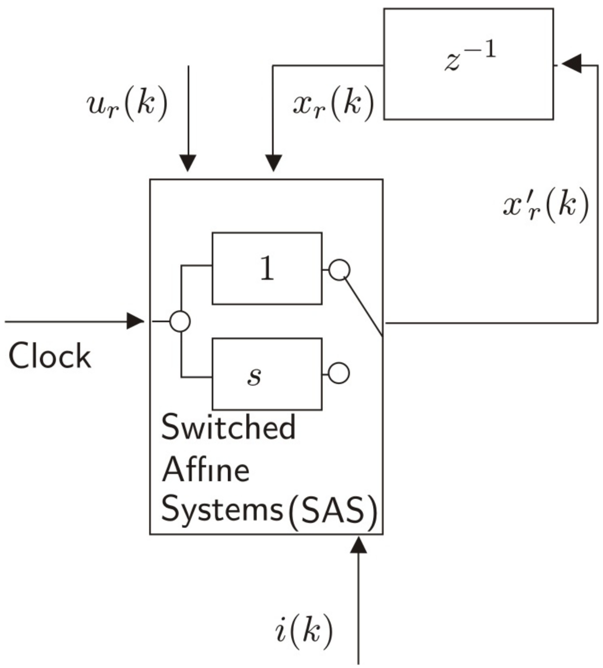
Discrete-time linear dynamics selected by

- Logic state
- Exogenous logic inputs
- Threshold conditions
- Time
- Any logic combination of the former

Discrete Hybrid Automata



Switched Affine Systems

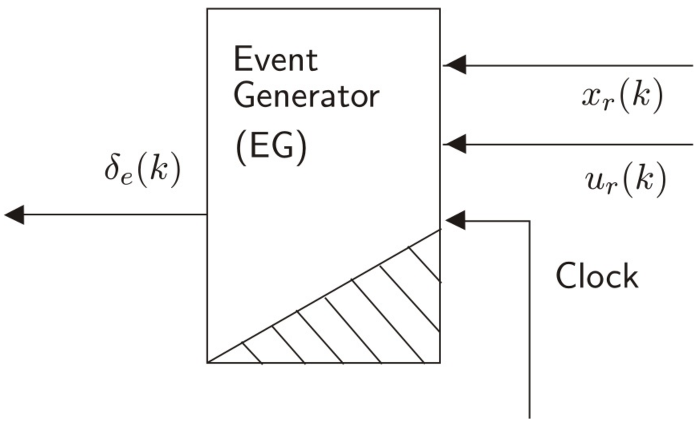


Linear affine dynamics depends upon the mode selector $i(t)$

$$x'_r(k) = A_{i(k)}x_r(k) + B_{i(k)}u_r(k) + f_{i(k)}$$

$$y_r(k) = C_{i(k)}x_r(k) + D_{i(k)}u_r(k) + g_{i(k)}$$

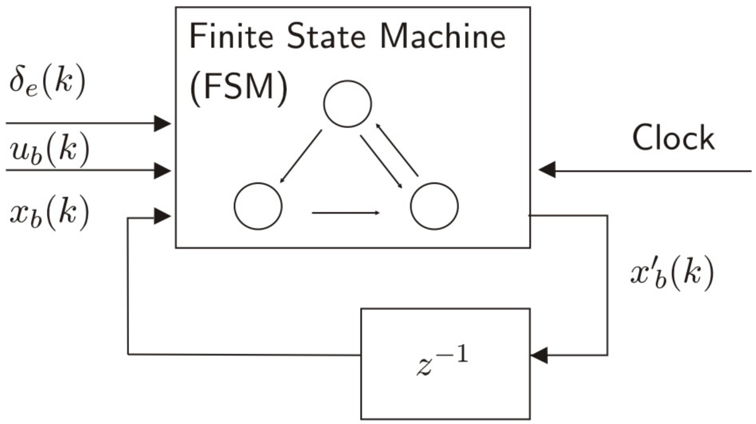
Event Generator



Generates a logic signal according to the satisfaction of a linear affine constraint

$$\delta_e(k) = f_{\mathsf{H}}(x_r(k), u_r(k), k)$$

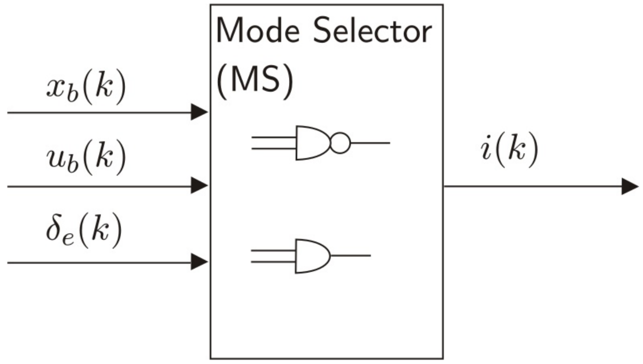
Finite State Machine



Discrete dynamic process
Evolves according to a logic state update function

$$
\begin{aligned}
x'_b(k) &= f_{\mathsf{B}}(x_b(k), u_b(k), \delta_e(k)) \\
y_b(k) &= g_{\mathsf{B}}(x_b(k), u_b(k), \delta_e(k))
\end{aligned}
$$

Mode Selector

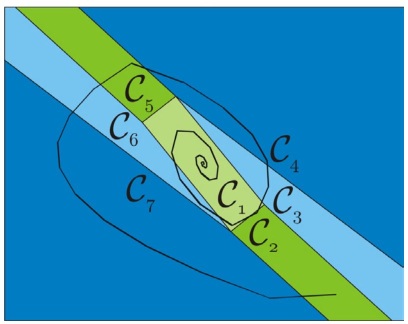


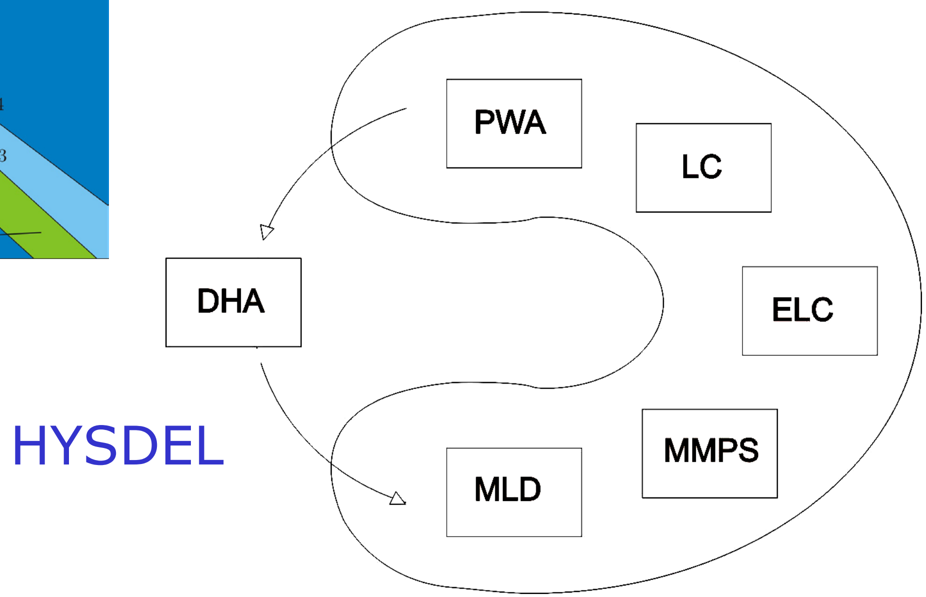
A Boolean function selects the active mode $i(k)$ of the SAS

$$i(k) = f_{\mathsf{M}}(x_b(k), u_b(k), \delta_e(k))$$

DHA and Other Modeling Frameworks

Piecewise Affine Models (PWA) define an affine dynamics on each cell of a polyhedral partition





Mixed Logic Dynamical Models (MLD) are

linear systems
plus mixed integ
inequalities

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

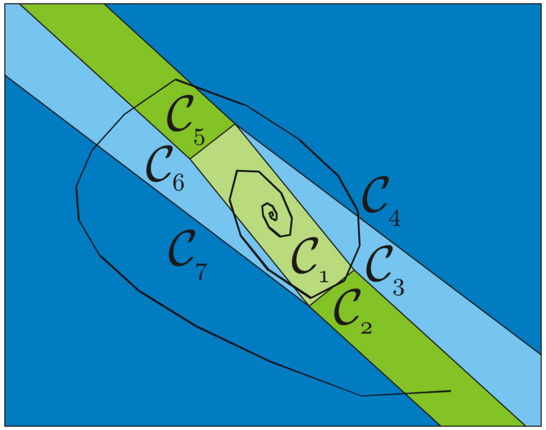
Piecewise Affine Systems

Sontag 1981

Approximates nonlinear dynamics arbitrarily well

Automatic conversion from MLD (Bemporad-Ferrari 2000)

Is Equivalent to DHA



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

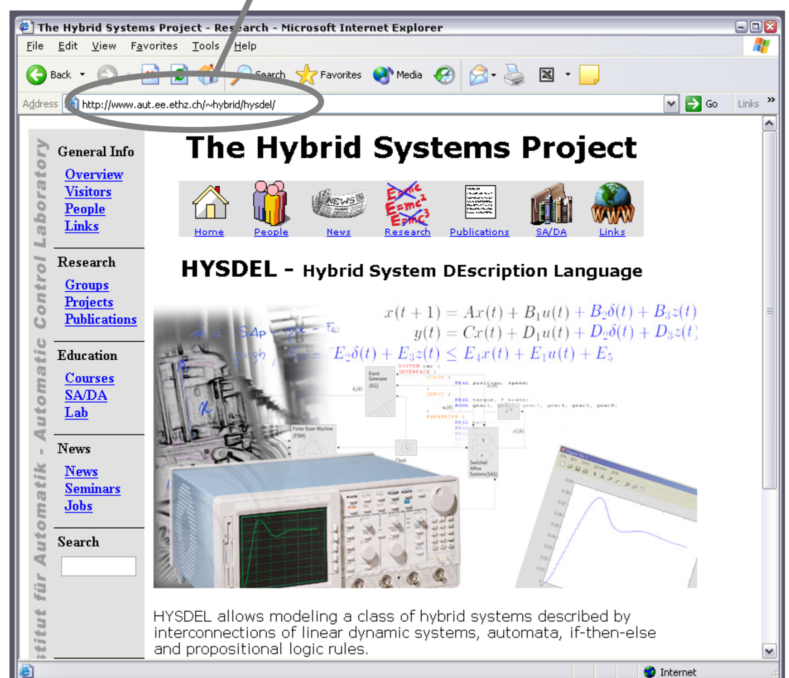
$$x \in \mathcal{X} \subseteq \mathbb{R}^n, \; u \in \mathcal{U} \subseteq \mathbb{R}^m, \; y \in \mathcal{Y} \subseteq \mathbb{R}^p$$

HYSDEL

HYSDEL (HYbrid Systems Description Language) compactly describes Discrete Hybrid Automata:

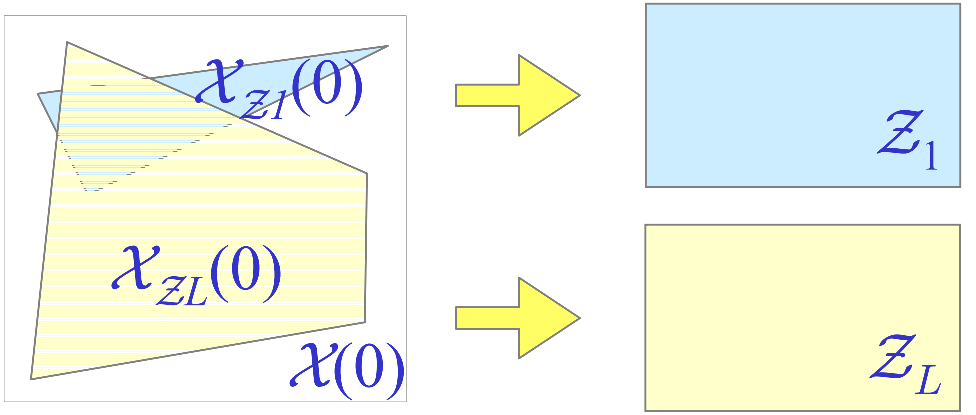
- Automata and Propositional Logic
- Continuous Dynamics
- A/D and D/A converters
- Constraints

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



HYSDEL compiler generates MLD and PWA models

Verification



Given:

- PWA system Σ
- Set of initial conditions $\mathcal{X}(0)$
- Target sets $\mathcal{Z}_1$, …, $\mathcal{Z}_L$ (disjoint)
- Time horizon $t < T_{\max}$

Problem:

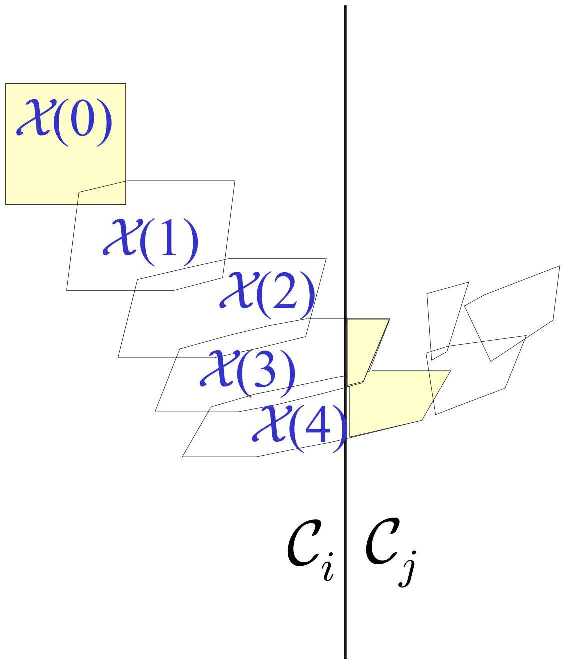
- Is $\mathcal{Z}_i$ reachable from $\mathcal{X}(0)$ in t steps?
- If yes, from which subset $\mathcal{X}_{\mathcal{Z}i}(0)$ of $\mathcal{X}(0)$?
- Disturbance/inputs driving $\mathcal{X}_{\mathcal{Z}i}(0)$ to $\mathcal{Z}_i$

Reach-Set Computation

$T_{\max} < \infty$, discrete-time $\Rightarrow$ Decidable but $\mathcal{NP}$-hard

Algorithm:

- Compute the polyhedral reach set $\mathcal{X}(t)$
- Detect switching
- Describe new intersections $\mathcal{X}(t) \cap \mathcal{C}_j$
- Stopping criteria for a single exploration



Reach Set Computation

Reach set implicitly defined by linear inequalities

$$\begin{cases} x(t) = A_i^t x(0) + \sum_{k=0}^{t-1} A_i^k [B_i u(t-1-k) + f_i], \\ x(0) \in \mathcal{X}(0), \\ u(k) \in \mathcal{U}(0), \quad k = 0, \ldots, t-1. \end{cases}$$

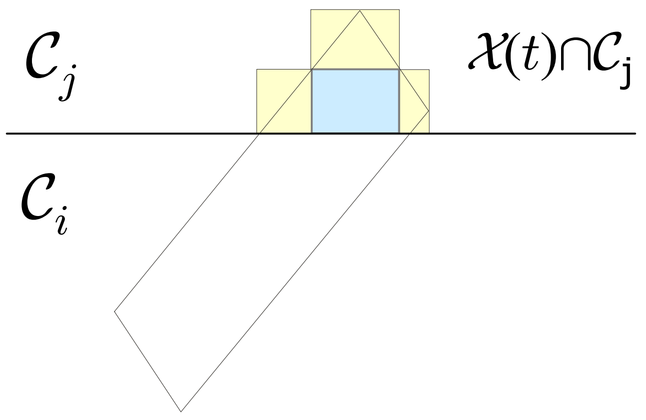
Simple to compute

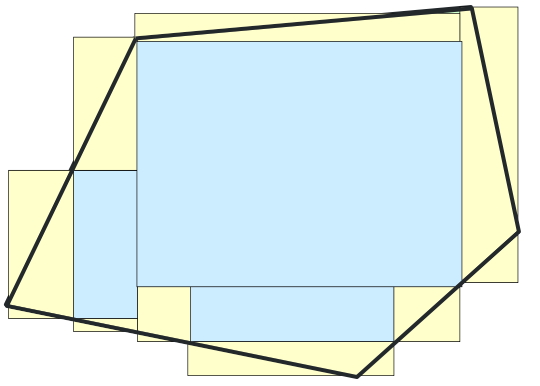
Number of constraints grows linearly with time

Explicit form also possible via projection methods

(e.g. CDD, Fukuda 1997)

Approximation

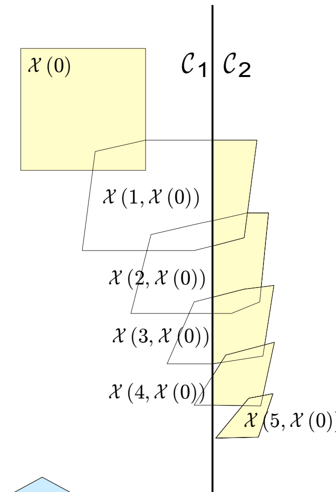




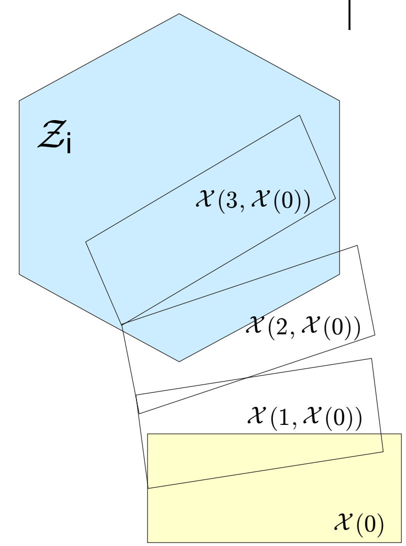
- Simple to compute via Linear Programming (LP)
- Can approximate with arbitrary precision
- Trade-off between quality and complexity of the approximation
- Both inner and outer approximations in one shot
- Approximate computation of projections

Stopping Criteria

Reach set has left the current region

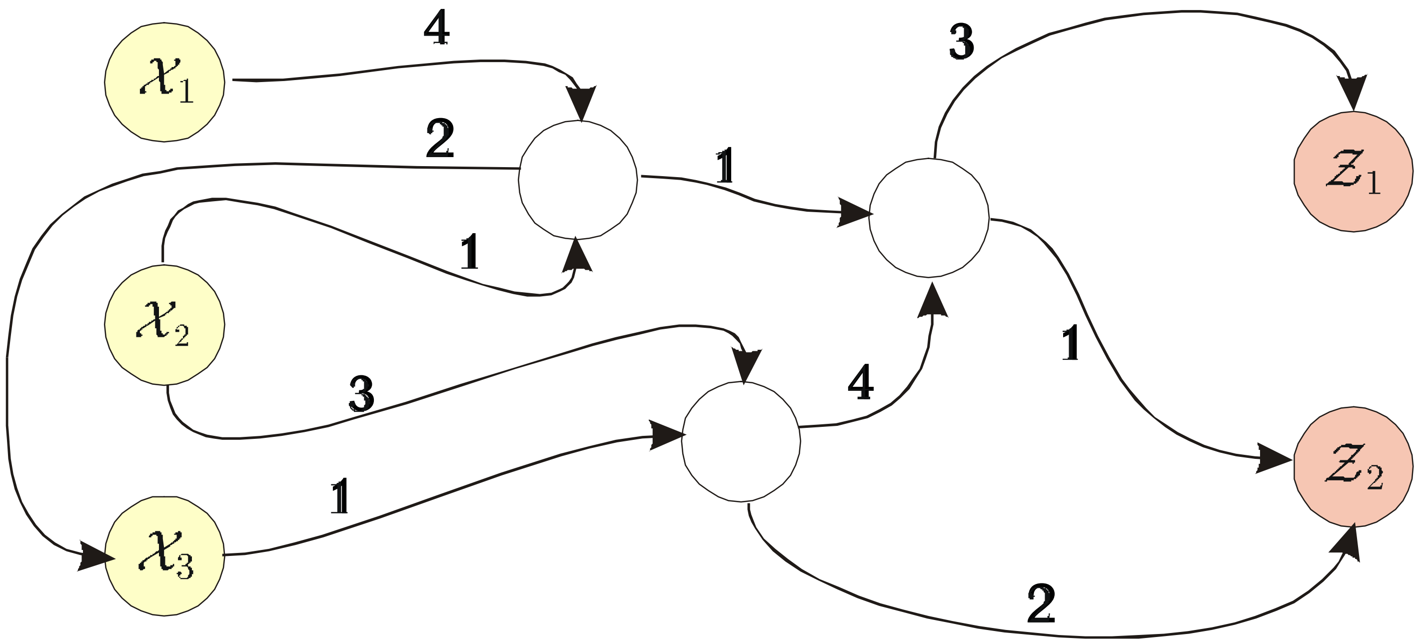


Reach set is all inside a target $\mathcal{Z}_i$



$t > T_{\max}$ (to guarantee termination)

Switching Sequences



All switching sequences of the system are paths in the graph

The converse is not true in general

Car Model

Vehicle dynamics

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

F_e = traction force F_b = brake force

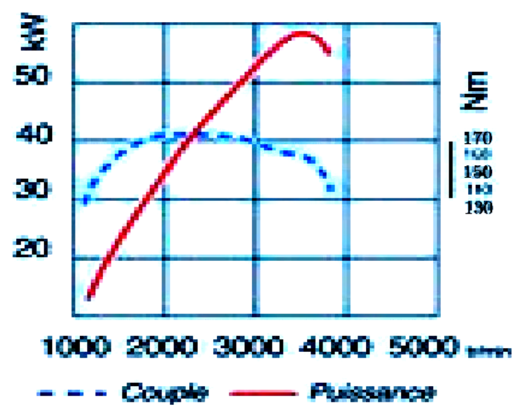
Transmission kinematics

$$\omega = \frac{k_g}{R_g(i)}\dot{x} \qquad F_e = \frac{k_g}{R_g(i)}M$$

ω = engine speed M = engine torque i = gear

Constraints

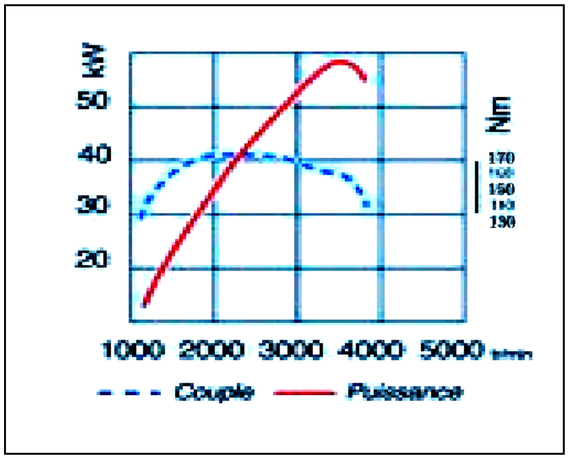
$$M_{\mathrm{min}}(\omega) \leq M \leq M_{\mathrm{max}}(\omega)$$

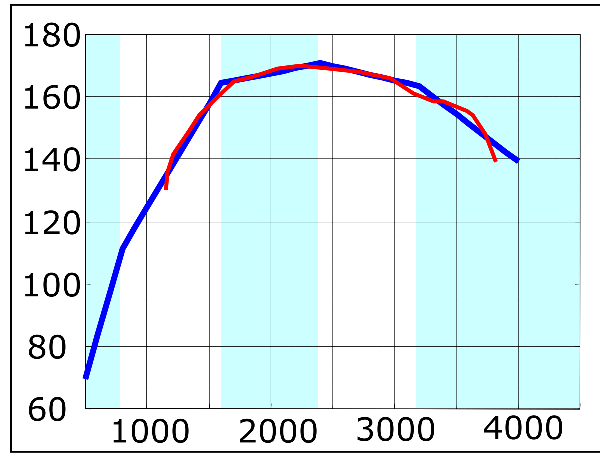


http://www.renault.fr

Car Model

Engine torque $M_{\mathrm{min}}(\omega) \leq M \leq M_{\mathrm{max}}(\omega)$





Piecewise-linearization: (PWL Toolbox, Julián 1999)

Gear selection:

$F_e = k_g M/R_g(i)$ depends on gear i:

IF $g_i=1$ THEN $F_{ei} = k_g M/R_g(i)$ ELSE 0

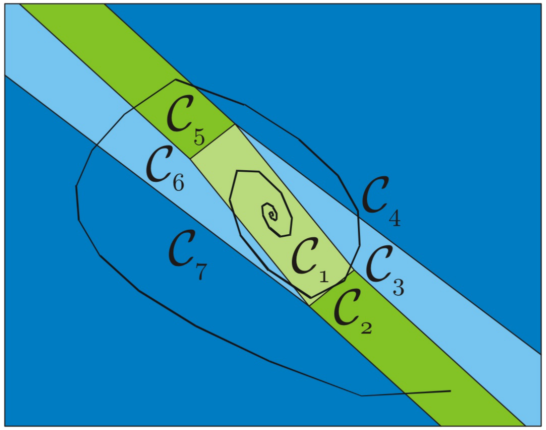
and $F_e = F_{eR} + F_{e1} + F_{e2} + F_{e3} + F_{e4} + F_{e5}$

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+
             DCe2 = {IF dPWL2 THEN (aPWL3-aPWL2)+(bPWL3-bPWL2)*(w1+w2+w3+
             DCe3 = {IF dPWL3 THEN (aPWL4-aPWL3)+(bPWL4-bPWL3)*(w1+w2+w3+
             DCe4 = {IF dPWL4 THEN (aPWL5-aPWL4)+(bPWL5-bPWL4)*(w1+w2+w3+
             }


        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+I

                  -(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;  }
```

Hybrid Model



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

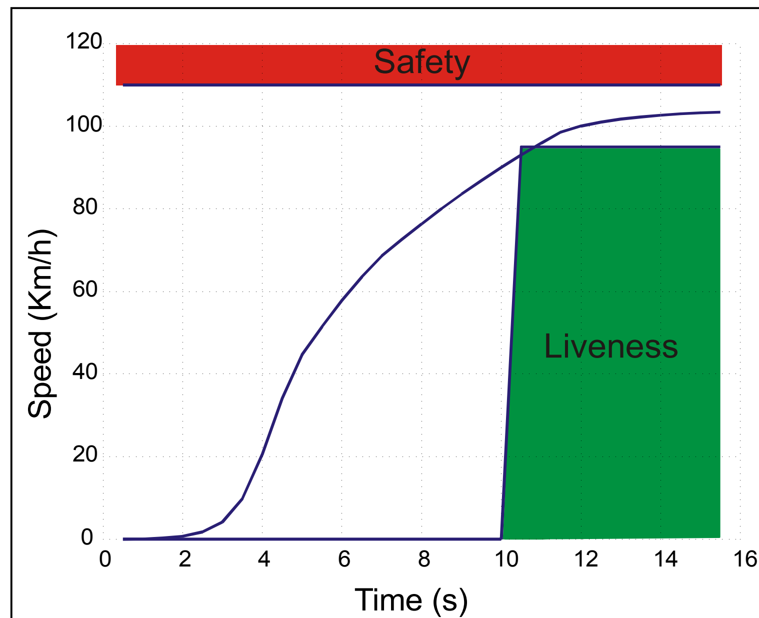
$$x \in \mathcal{X} \subseteq \mathbb{R}^n,\ u \in \mathcal{U} \subseteq \mathbb{R}^m,\ y \in \mathcal{Y} \subseteq \mathbb{R}^p$$

$x \in \mathcal{X} \subseteq \mathbb{R}^8$, 150 regions.

Cruise Control: Verification Results

For all $v_r \in [30,70]$ km/h, the controller satisfies both liveness & safety properties (CPU time: 9109 s on Matlab5.3, PC 650 MHz)

For $v_r \in [30,120]$ km/h the verification algorithm finds the first counterexample after 415 s.



Conclusions

Reachability Analisis of hybrid systems answers important safety and liveness questions

PWA models capture well the behavior of real systems

Polyhedral computation is a key tool for reachability analysis of hybrid systems