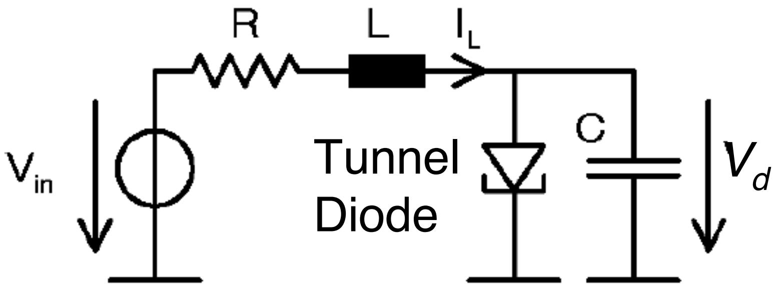
# Verification of Hybrid Systems and Validation of Controllers


**Goran Frehse**
Université Grenoble 1 Joseph Fourier
Verimag, France


EMACS Summer School, Bourges, 2015

# Example: Tunnel Diode Oscillator




$$\dot{V}_C = \tfrac{1}{C}\left(-I_d(V_C) + I_L\right)$$
$$\dot{I}_L = \tfrac{1}{L}\left(-V_C - RI_L + V_{in}\right)$$

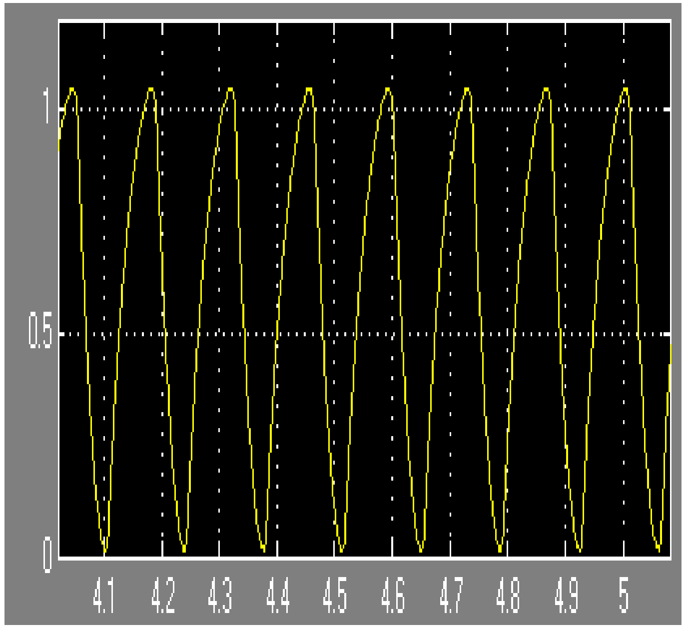
Dang, Donze, Maler, FMCAD' 04

- **What are good parameters?**
  - startup conditions
  - parameter variations
  - disturbances

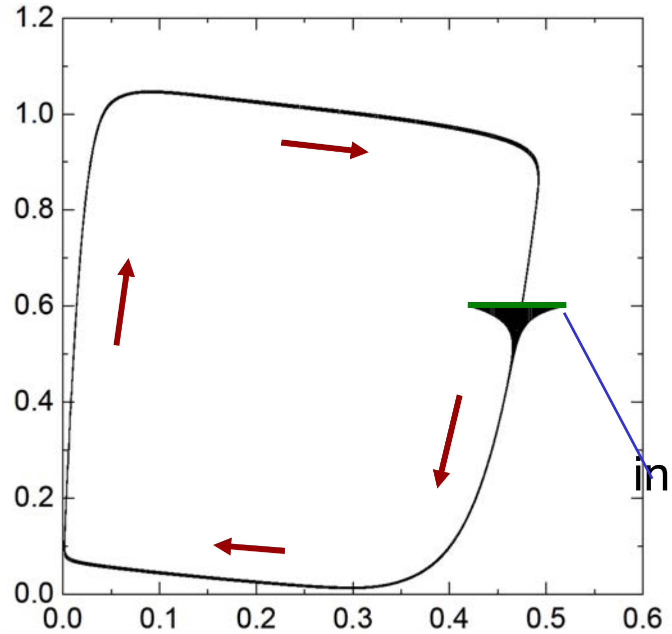
# Example: Tunnel Diode Oscillator

**R=0.20Ω ⇒ Oscillation**

$I_L$ [mA]




Time [μs]

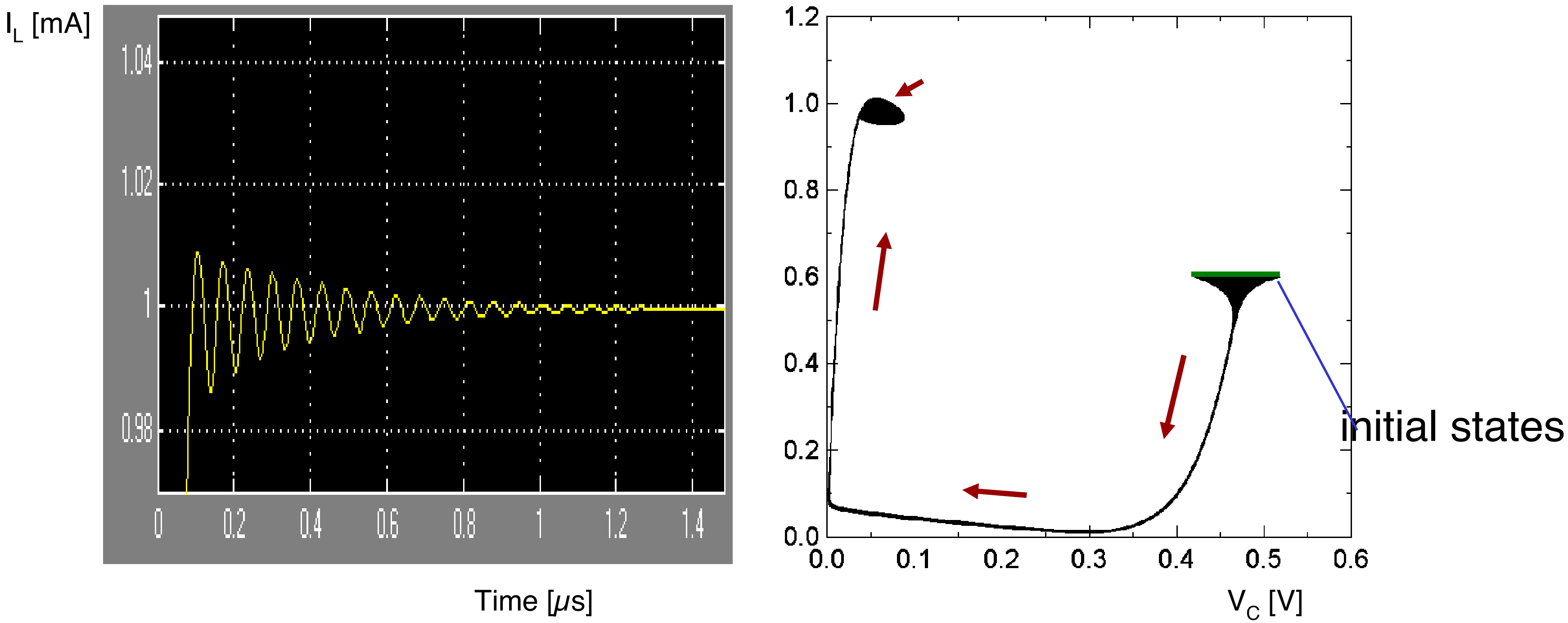



initial states

$V_C$ [V]

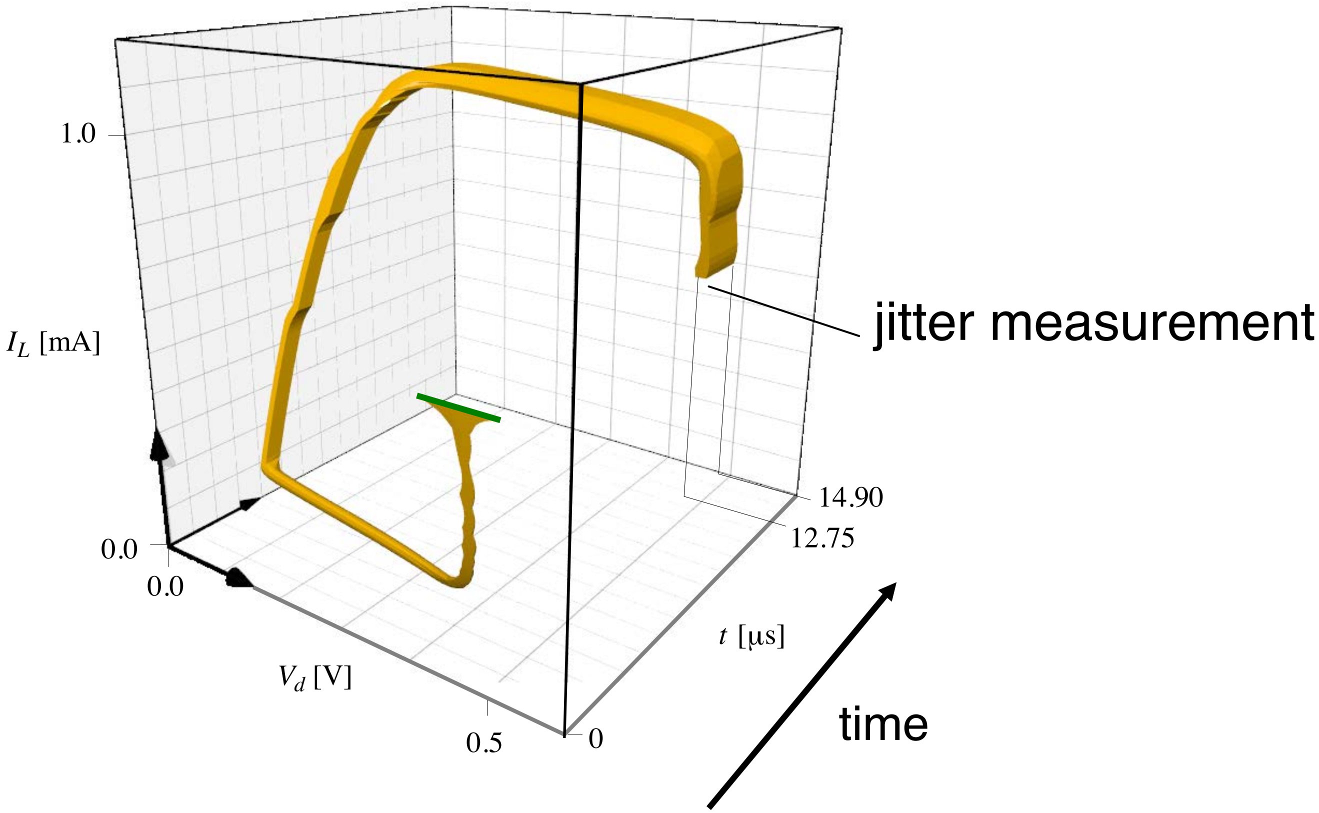
# Example: Tunnel Diode Oscillator

**R=0.24Ω ⇒ Stable equilibrium**

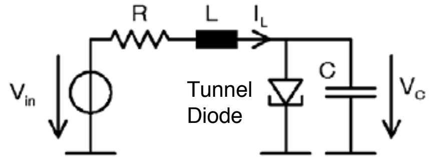


# Example: Tunnel Diode Oscillator

- **Jitter measurement**
  - add clock that is reset at zero crossing



# Example: Tunnel Diode Oscillator



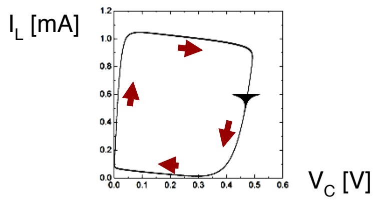

**Analog/Mixed Signal Circuit**

↓

$$\dot{V}_C = \tfrac{1}{C}\left(-I_d(V_C) + I_L\right)$$
$$\dot{I}_L = \tfrac{1}{L}\left(-V_C - RI_L + V_{in}\right)$$

**Formal Model**

↓




**Reachability Analysis**

↓

- Oscillation
- Jitter
- ...

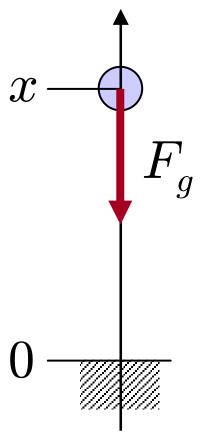
**Guaranteed Safety Property**

# Outline

- **Modeling with Hybrid Automata**
- **Reachability versus Simulation**
- **Reachability Algorithms**
  - piecewise constant dynamics
  - piecewise affine dynamics
- **Case Study: Controller Implementation**
- **SpaceEx Tool Platform**
- **Bibliography**

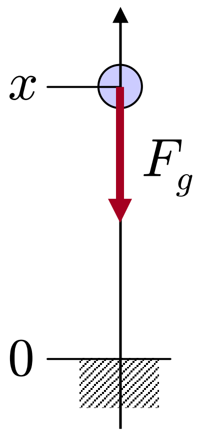
# Modeling with Hybrid Automata

- **Example: Bouncing Ball**
  - ball with mass $m$ and position $x$ in free fall
  - bounces when it hits the ground at $x = 0$
  - initially at position $x_o$ and at rest



# Part I – Free Fall

- **Condition for Free Fall**
  - ball above ground: $x \geq 0$




- **First Principles (physical laws)**
  - gravitational force :

$$F_g = -mg$$

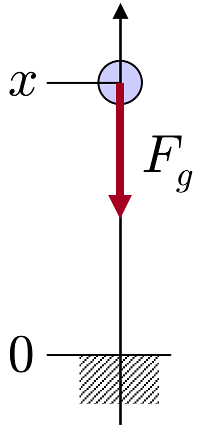
$$g = 9.81\mathrm{m/s}^2$$

  - Newton's law of motion :

$$m\ddot{x} = F_g$$

# Part I – Free Fall

$$\begin{aligned} F_g &= -mg \\ m\ddot{x} &= F_g \end{aligned}$$




- **Obtaining 1st Order ODE System**
  - ordinary differential equation $\dot{x} = f(x)$
  - transform to 1st order by introducing variables for higher derivatives
  - here: $v = \dot{x}$:

$$\begin{aligned} \dot{x} &= v \\ \dot{v} &= -g \end{aligned}$$

# Part II – Bouncing

- **Conditions for "Bouncing"**
  - ball at ground position: $x = 0$
  - downward motion: $v < 0$
- **Action for "Bouncing"**
  - velocity changes direction
  - loss of velocity (deformation, friction)
  - $v := -cv$, $0 \leq c \leq 1$

# Combining Part I and II

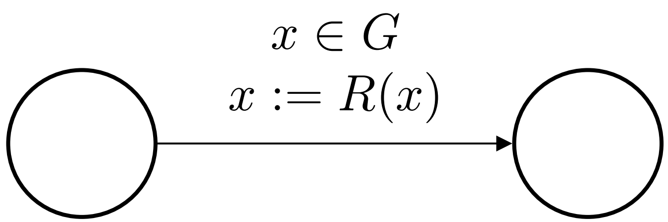
- **Free Fall**
  - while $x \geq 0$,
    $$\begin{aligned}\dot{x} &= v\\ \dot{v} &= -g\end{aligned}$$

**continuous dynamics**

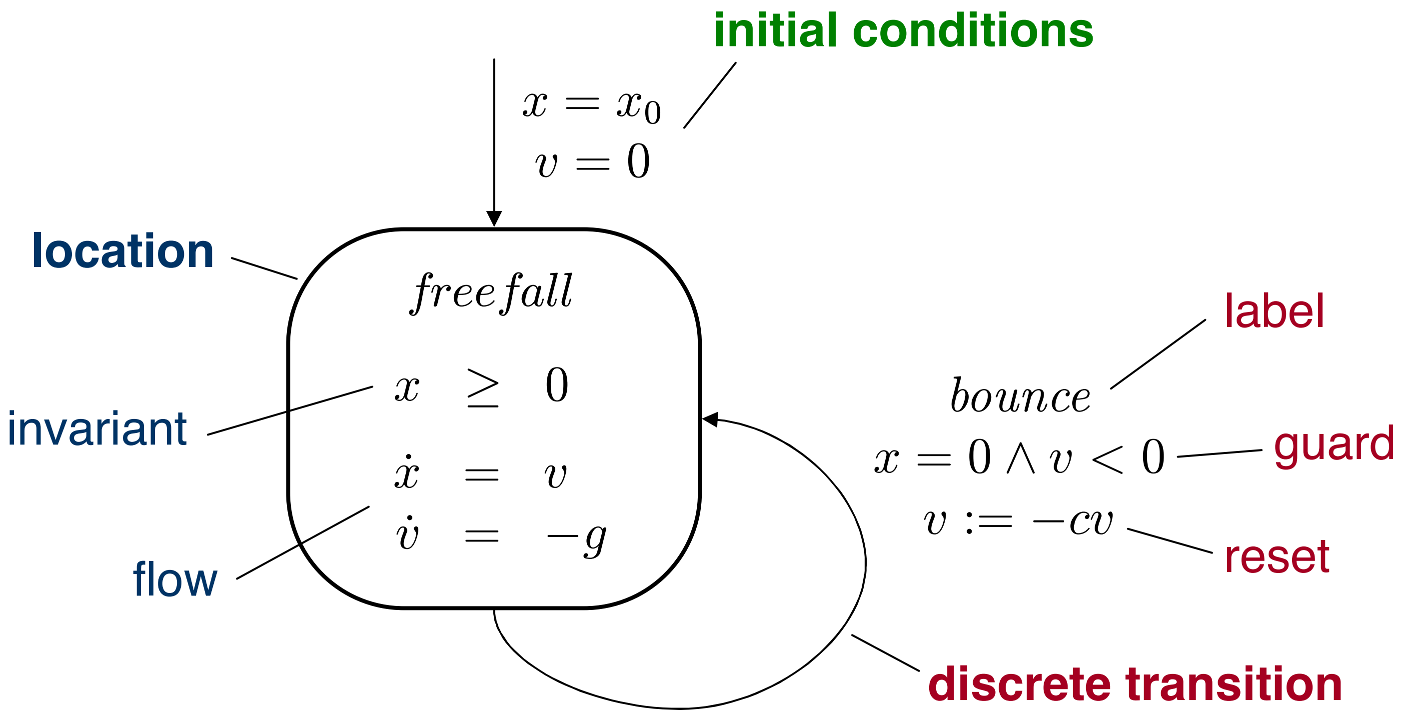
$$\dot{x} = f(x)$$

- **Bouncing**
  - if $x = 0$ and $v < 0$
    $$v := -cv$$

**discrete dynamics**

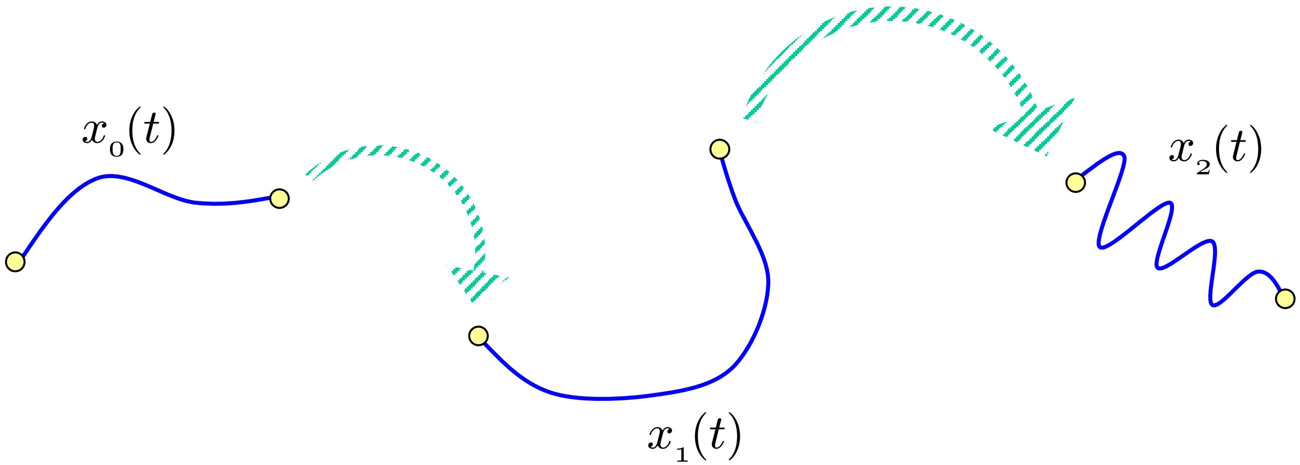


# Hybrid Automaton Model



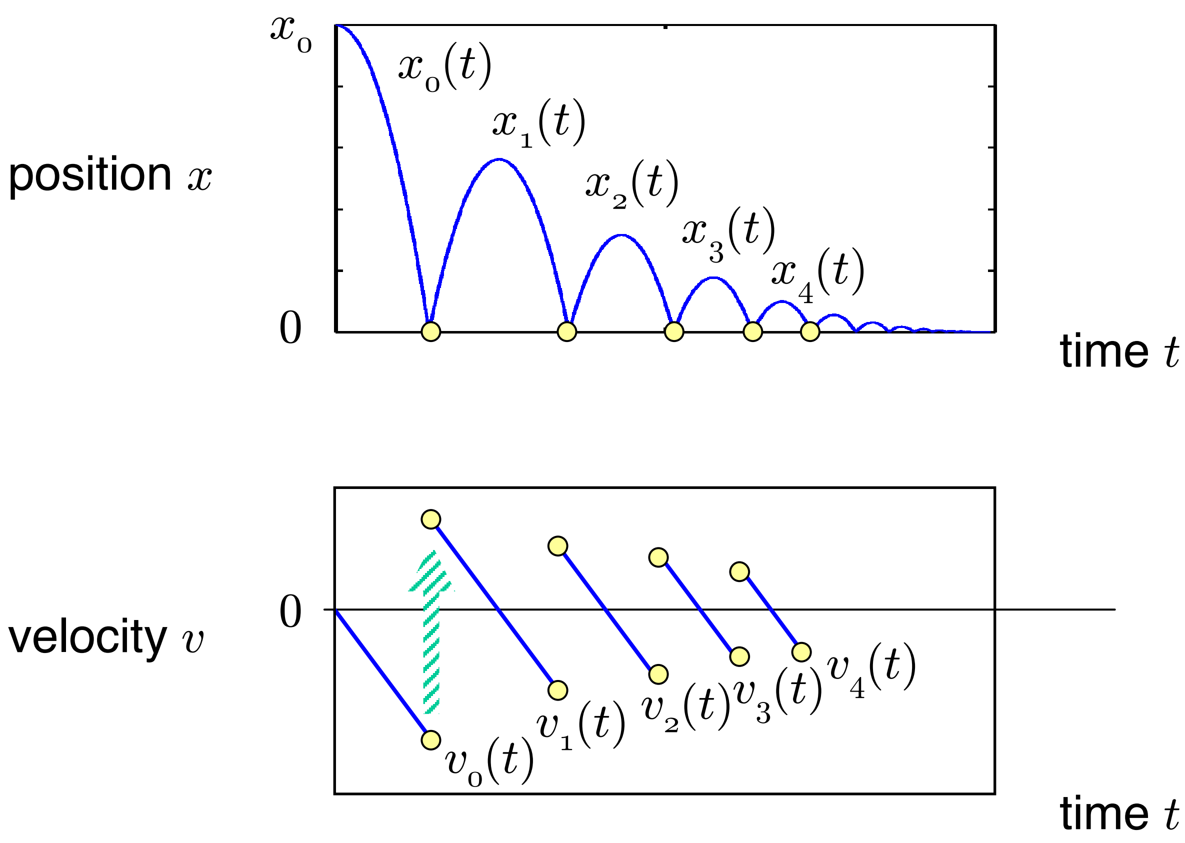
# ODEs with Switching

- **Continous/Discrete Behaviour**
  - evolution with time according to ODE dynamics
  - dynamics can switch (instantaneous)
  - state can jump (instantaneous)



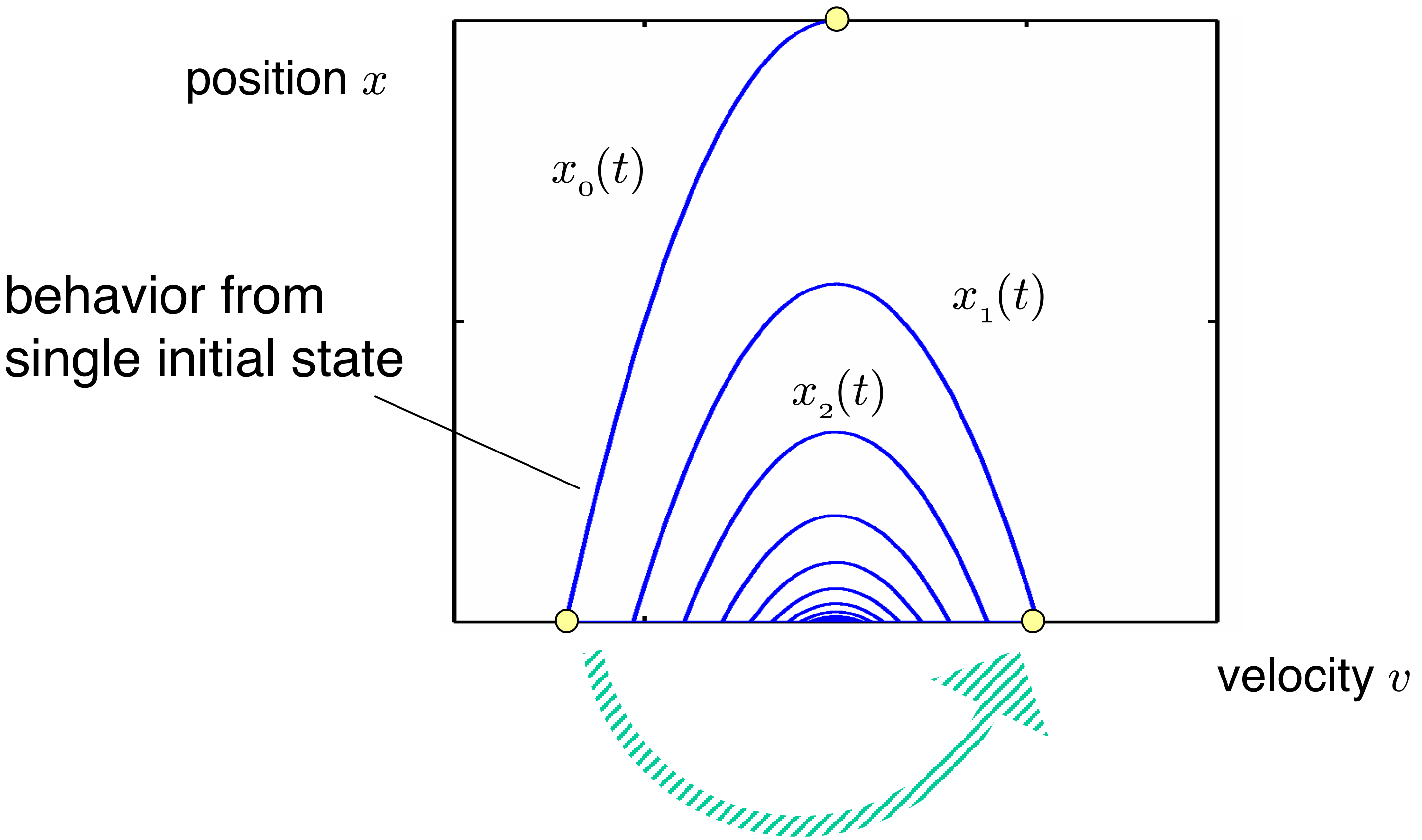
# Example: Bouncing Ball

- **States over Time**



# Example: Bouncing Ball

- **States over States = State-Space View**

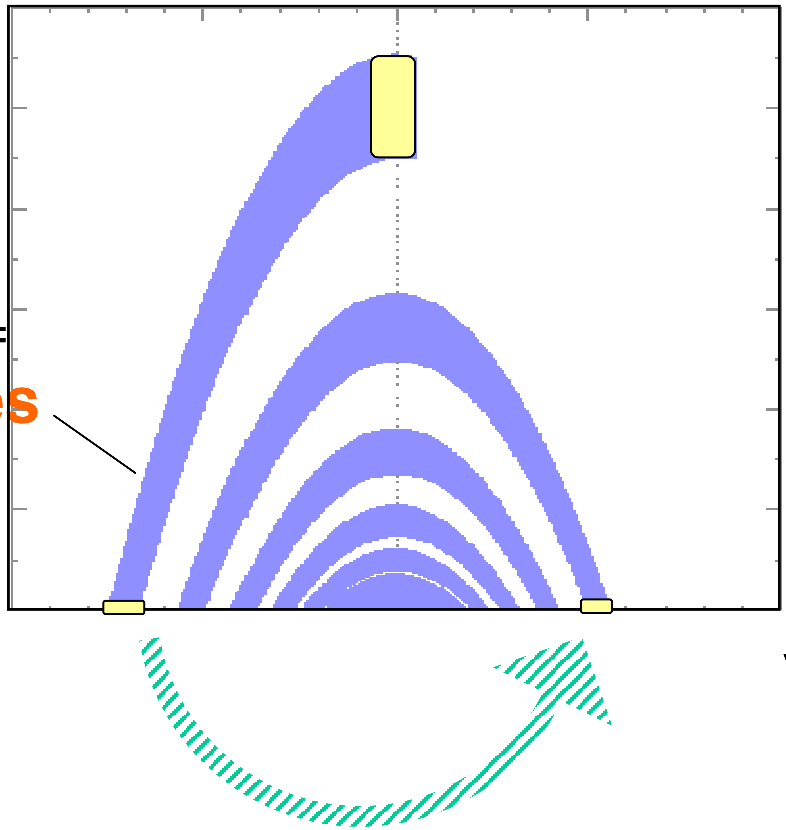


# Example: Bouncing Ball

- **Reachability in State-Space**

position $x$

behaviors from
set of initial states =
**reachable states**

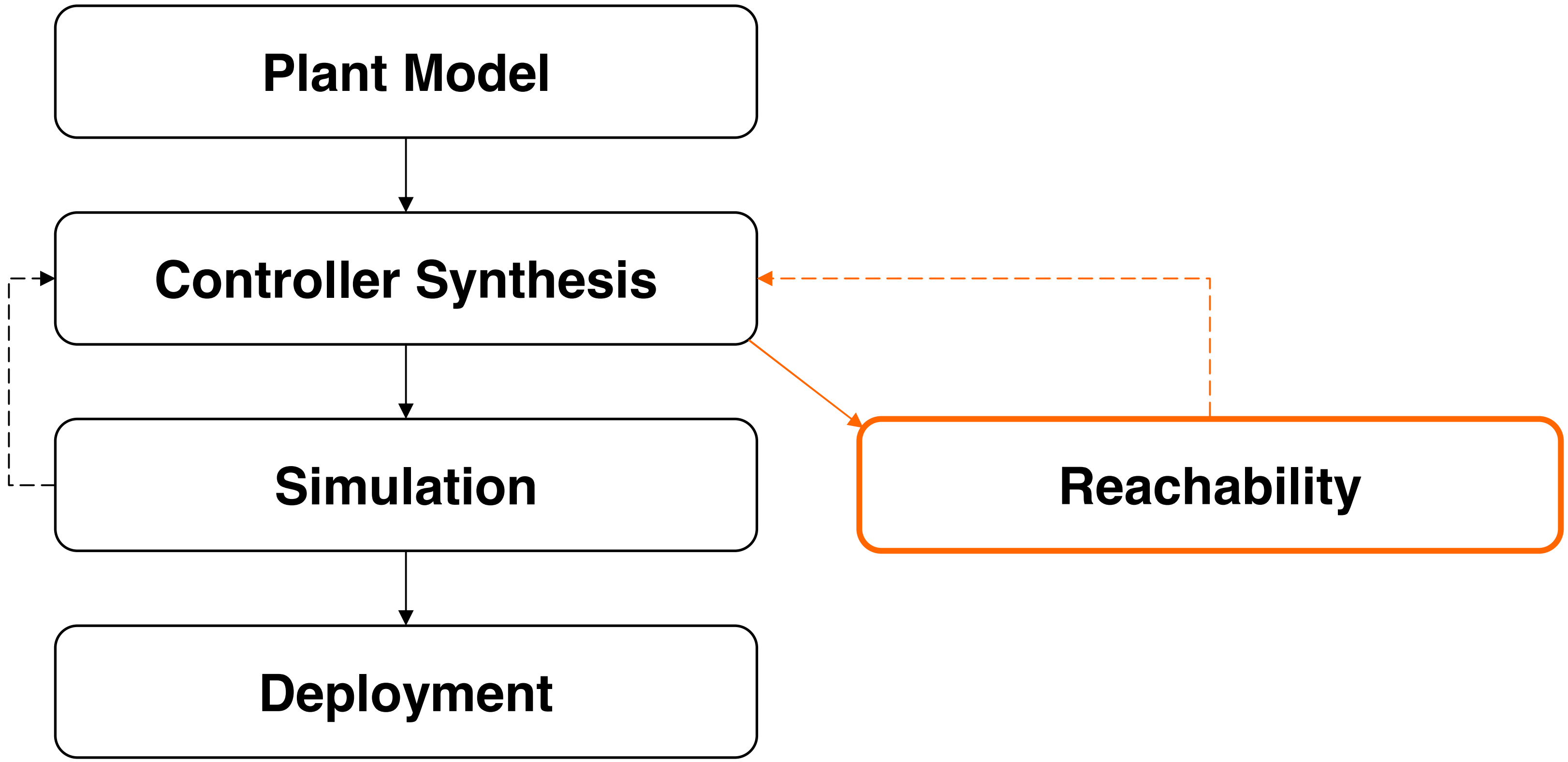


velocity $v$

# Outline

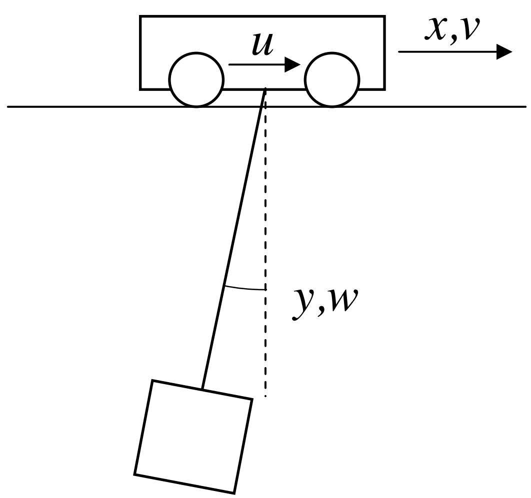
- **Modeling with Hybrid Automata**
- **Reachability versus Simulation**
- **Reachability Algorithms**
  - piecewise constant dynamics
  - piecewise affine dynamics
- **Case Study: Controller Implementation**
- **SpaceEx Tool Platform**
- **Bibliography**

# Reachability in Model Based Design



# Example: Overhead Crane

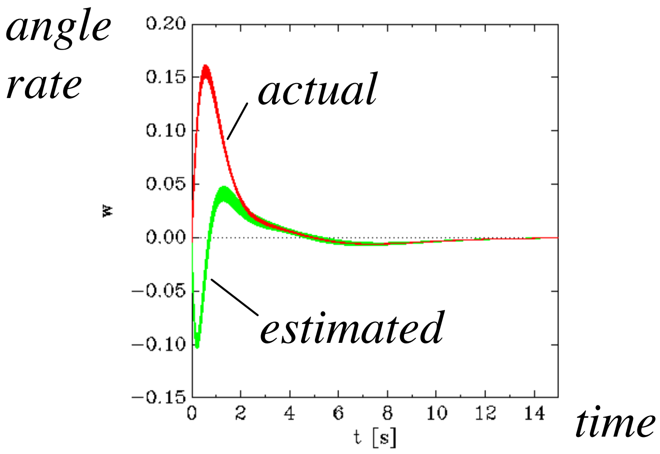
- **State variables**
  - position $x$, speed $v$
  - line angle $y$, angle rate $w$
- **Feedback controller**
  - state estimated by observer
- **Goals**
  - validate observer for $y,w$
  - validate swing



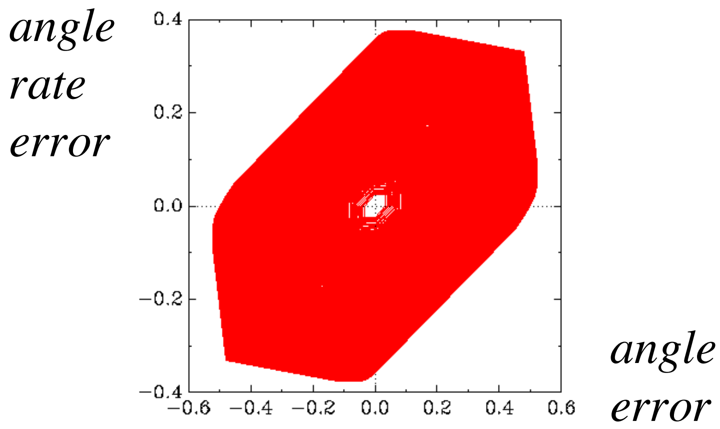

$$
\begin{aligned}
\dot{x} &= v \\
\dot{v} &= b_{21}u + b_{22}g \\
\dot{y} &= w \\
\dot{w} &= -a_{43}y - b_{41}u
\end{aligned}
$$

# Overhead Crane – Observer

- **Validation of observer quality**
- **Standard:**
  – Simulation of “representative trajectories”

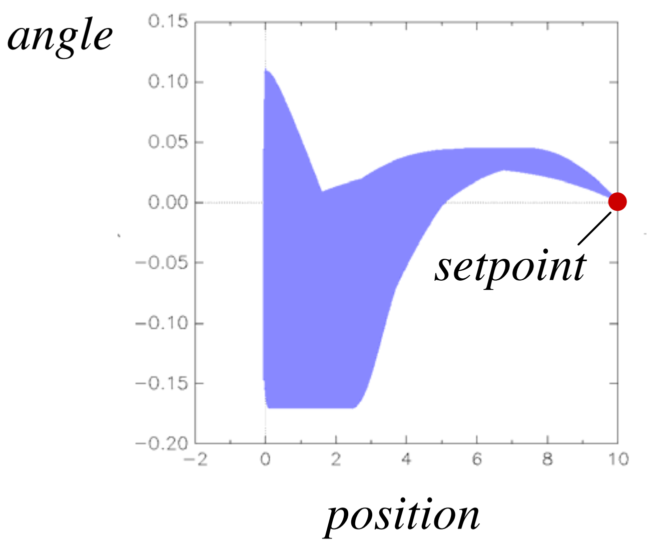



- **Reachability:**
  – Error bounds over **range** of initial states & inputs

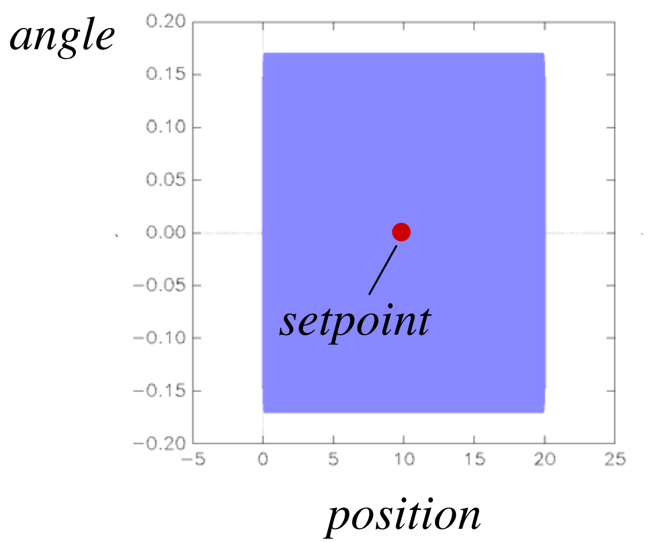


# Overhead Crane - Controller

- **Evaluation of swing (angle range)**




over small initial range
[-0.17,0.12]




over full operating range
[-0.17,0.17]

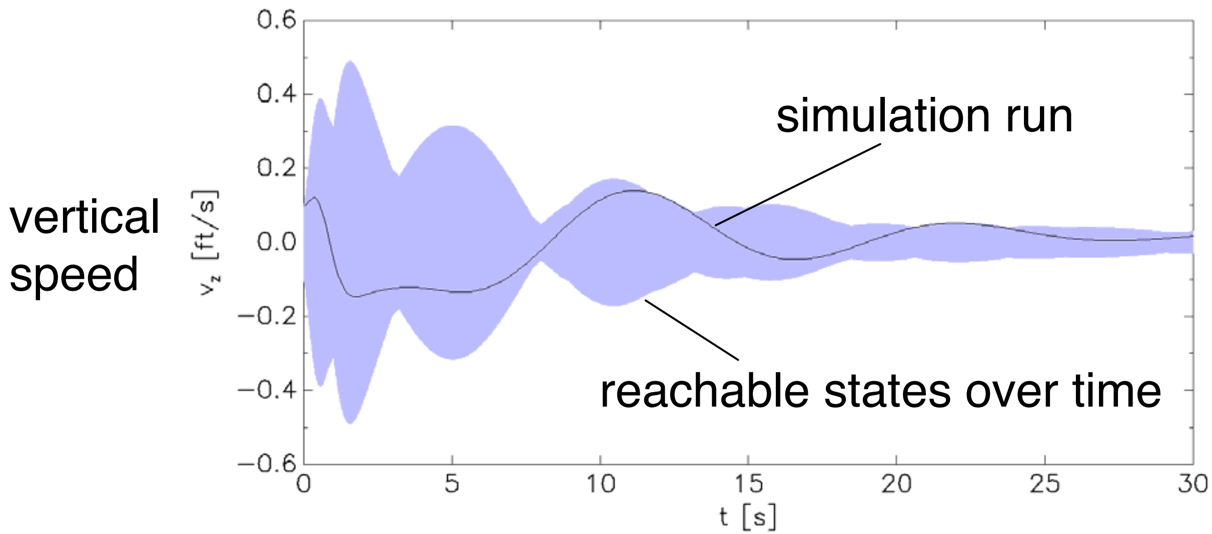
# Example: Controlled Helicopter


- **28-dim model of a Westland Lynx helicopter**
  - 8-dim model of flight dynamics
  - 20-dim continuous H∞ controller for disturbance rejection
  - stiff, highly coupled dynamics

S. Skogestad and I. Postlethwaite, Multivariable Feedback Control: Analysis and Design. John Wiley & Sons, 2005.

# Simulation vs Reachability

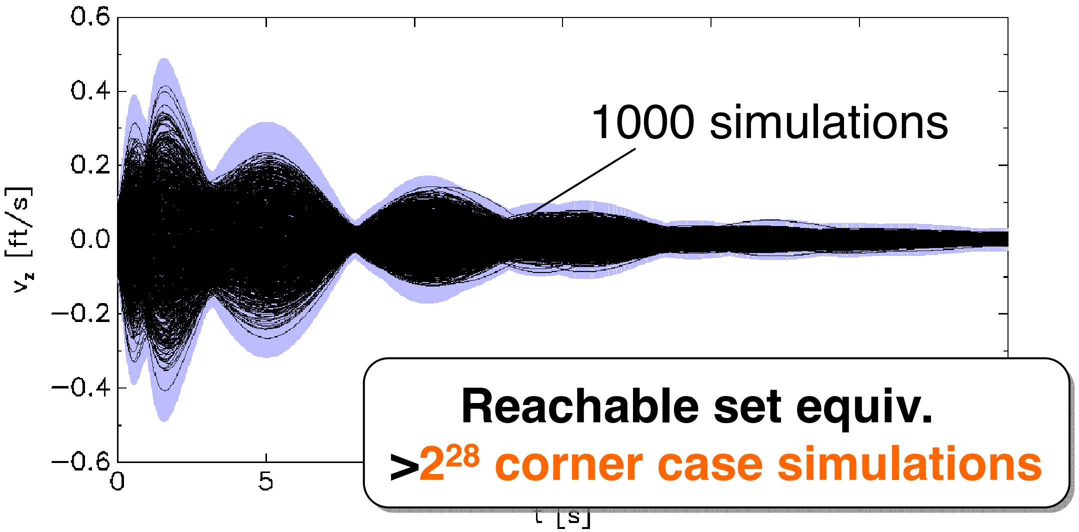
- **Simulation**
  - approximative sample of **single** behavior
  - over finite time
- **Reachability**
  - over-approximative set-valued cover of **all** behaviors
  - over finite or infinite time



# Simulation vs Reachability

- **Simulation**
  - deterministic
    - resolve nondet. using Monte Carlo etc.
  - scalable for nonlinear dyn.
- **Reachability**
  - **nondeterministic**
    - continuous disturbances...
    - implementation tolerances...
  - scalable for linear dynamics

vertical speed

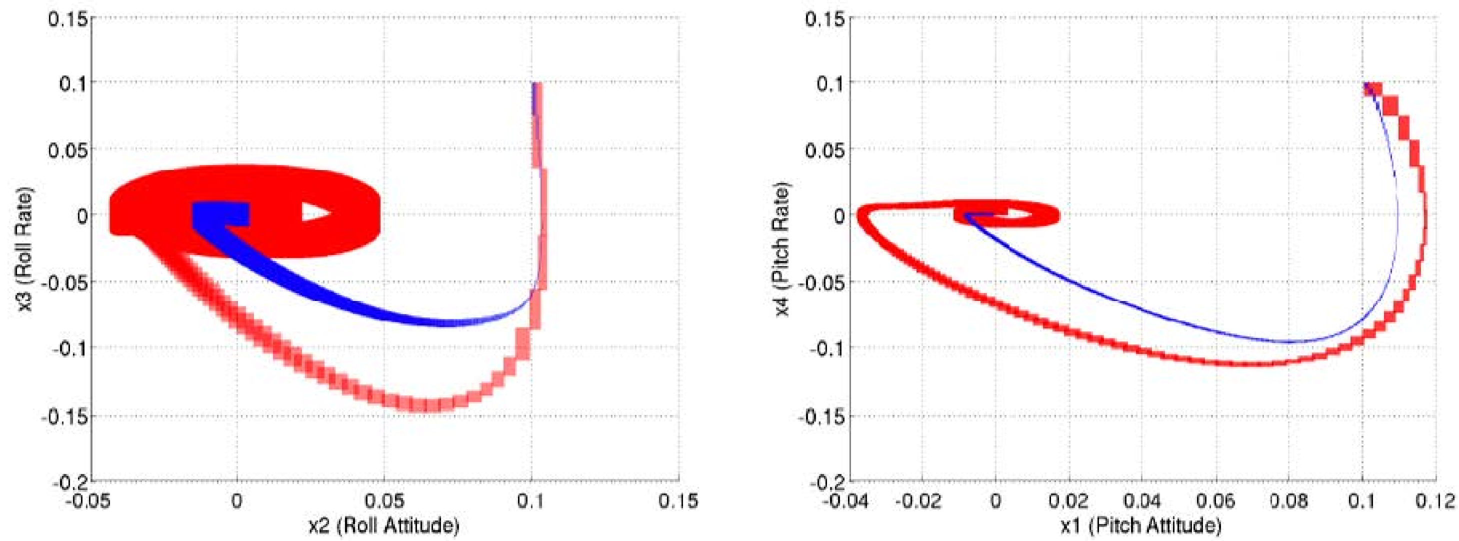



**Reachable set equiv.**
**>$2^{28}$ corner case simulations**

Frehse et al. "SpaceEx: Scalable verification of hybrid systems." Computer Aided Verification. Springer, 2011.

# Example: Controlled Helicopter

- **Comparing two controllers subject to continuous disturbance**




(a) Roll stabilization (b) Pitch stabilization

Frehse, G., et al. "SpaceEx: Scalable verification of hybrid systems." Computer Aided Verification. Springer, 2011.

# Outline

- **Modeling with Hybrid Automata**
- **Reachability versus Simulation**
- **Reachability Algorithms**
  - piecewise constant dynamics
  - piecewise affine dynamics
- **Case Study: Controller Implementation**
- **SpaceEx Tool Platform**
- **Bibliography**

# Computing Reachable States

- **Computing One-Step Successors**
  - time elapse: $Y = Post_c(X)$
  - jumps: $S = Post_d(S)$
- **Fixpoint computation**
  - Initialization: $R_0 = Ini$
  - Recurrence: $R_{k+1} = R_k \cup Post_d(R_k) \cup Post_c(R_k)$
  - Termination: $R_{k+1} = R_k \Rightarrow Reach = R_k$.

# Computing Reachable States

- **Set-based integration can answer many interesting questions about a system**
  - safety, bounded liveness,…
- **Problems**
  - in general termination not guaranteed
  - set-based integration of ODEs is hard
- **Solution**
  - piecewise constant approximations
  - piecewise linear approximations
  - math tricks (implicit set representations,...)

# Piecewise Constant Dynamics

- **A very simple class of hybrid systems: Linear Hybrid Automata**
  - trajectories are straight lines
- **Exact computation of successor states possible**
  - reachability is nonetheless **undecidable**.

# Linear Hybrid Automata

- **Continuous Dynamics**
  - piecewise constant: $\dot{x} = 1$
  - intervals: $\dot{x} \in [1, 2]$
  - conservation laws: $\dot{x}_1 + \dot{x}_2 = 0$
  - general form: conjunctions of linear constraints

$$a \cdot \dot{x} \bowtie b, \qquad a \in \mathbb{Z}^n, b \in \mathbb{Z}, \bowtie \in \{<, \leq\}.$$

**= convex polyhedron over derivatives**

# Linear Hybrid Automata

- **Discrete Dynamics**
  - affine transform: $x := ax + b$
  - with intervals: $x_2 := x_1 \pm 0.5$
  - general form: conjunctions of linear constraints (new value $x'$)

$$a \cdot x + a' \cdot x' \bowtie b, \qquad a, a' \in \mathbb{Z}^n, b \in \mathbb{Z}, \bowtie \in \{<, \leq\}$$

**= convex polyhedron over $x$ and $x$'**

# Linear Hybrid Automata

- **Invariants, Initial States**
  - general form: conjunctions of linear constraints

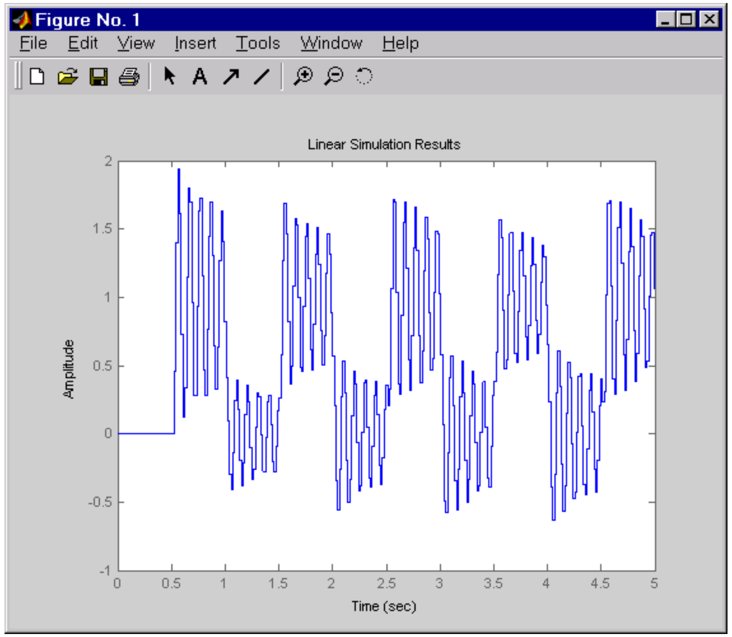
$$a \cdot x \bowtie b, \qquad a \in \mathbb{Z}^n, b \in \mathbb{Z}, \bowtie \in \{<, \leq\},$$

**= convex polyhedron over** $x$

# Linear Hybrid Automata

(source: wikipedia)

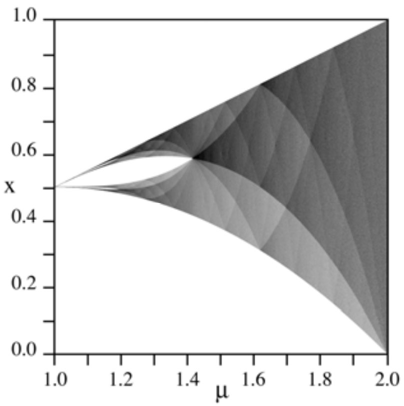
- **model complex behavior**
  - discrete jump maps can model discrete-time linear control systems (widely used in industry)



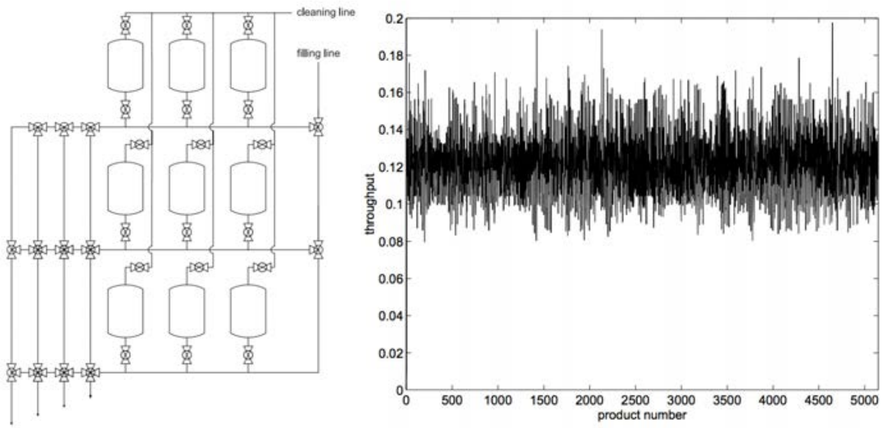

source: mathworks.com

# Linear Hybrid Automata

- **chaos**
  - even with 1 variable, 1 location, 1 transition (tent map)
  - observed in actual production systems [Schmitz,2002]




states of the Tent map
source: wikipedia

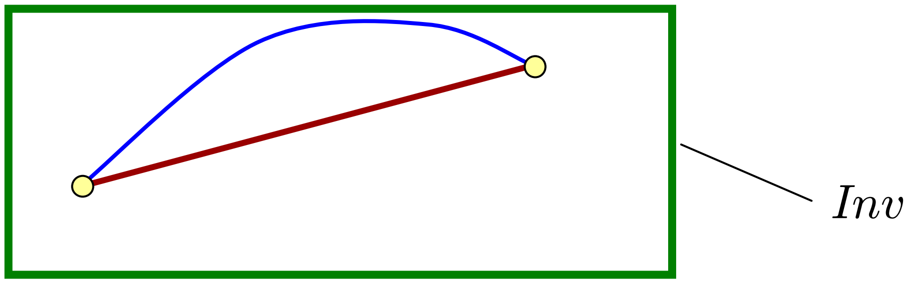



brewery and chaotic throughput [Schmitz,2002]

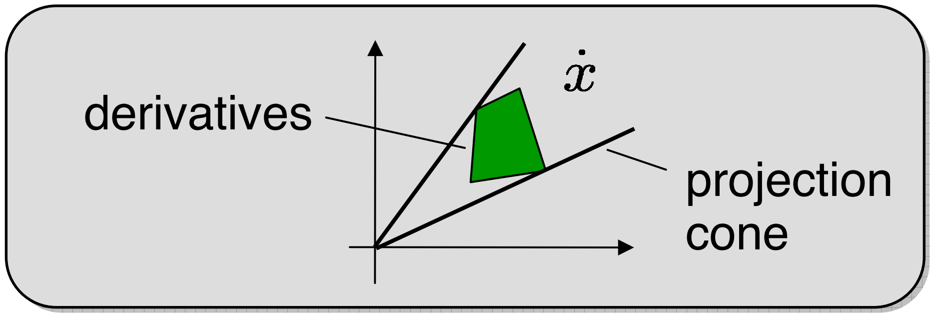

Schmitz, J. P. M., D. A. Van Beek, and J. E. Rooda. "Chaos in discrete production systems?." Journal of Manufacturing Systems 21.3 (2002): 236-246.c

# Compute time elapse states $Post_c(S)$

- **arbitrary trajectory iff straight line exists (convex invariant)** [Alur et al.]




- **time elapse along straight line can be computed as projection along cone** **[Halbwachs et al.]**

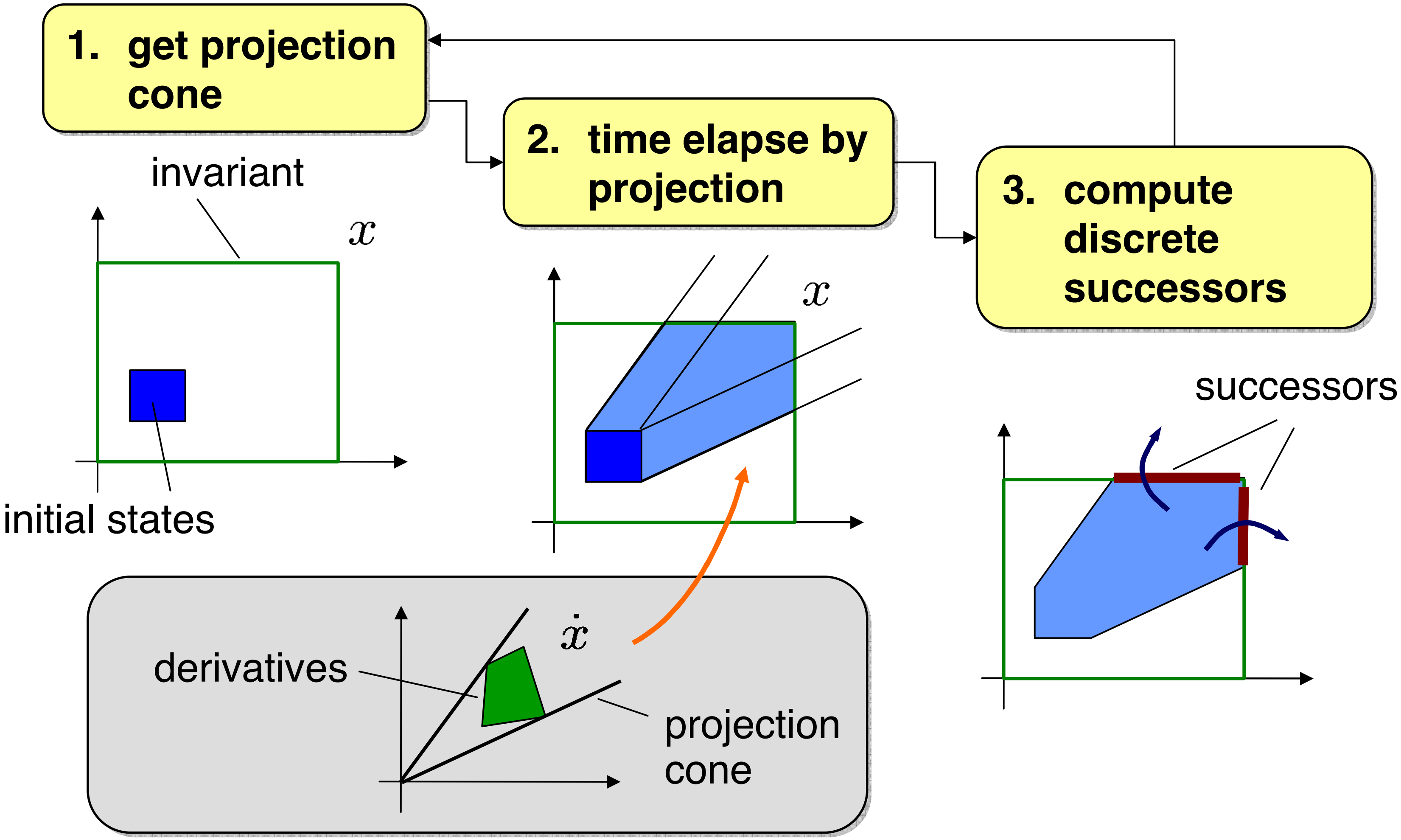


# Compute discrete successors $Post_d(S)$

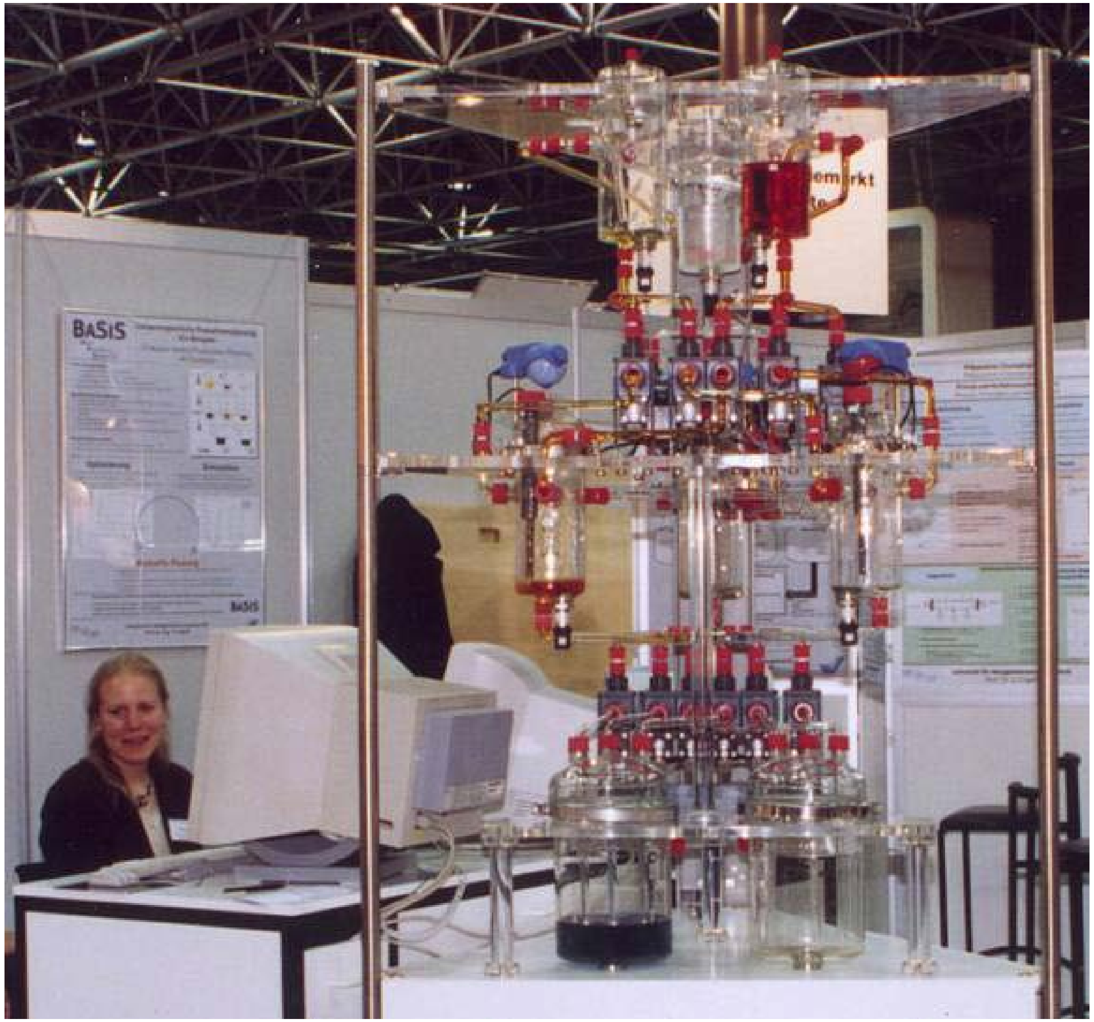
- $Post_d(S) =$ **all $x$' for which exists $x \in S$ s.t.**
  - guard: $x \in G$
  - reset and target invariant: $x$' $\in R(x) \cap Inv$

- **Operations:**
  - existential quantification
  - intersection
  - standard operations on convex polyhedra, but O(exp(n))

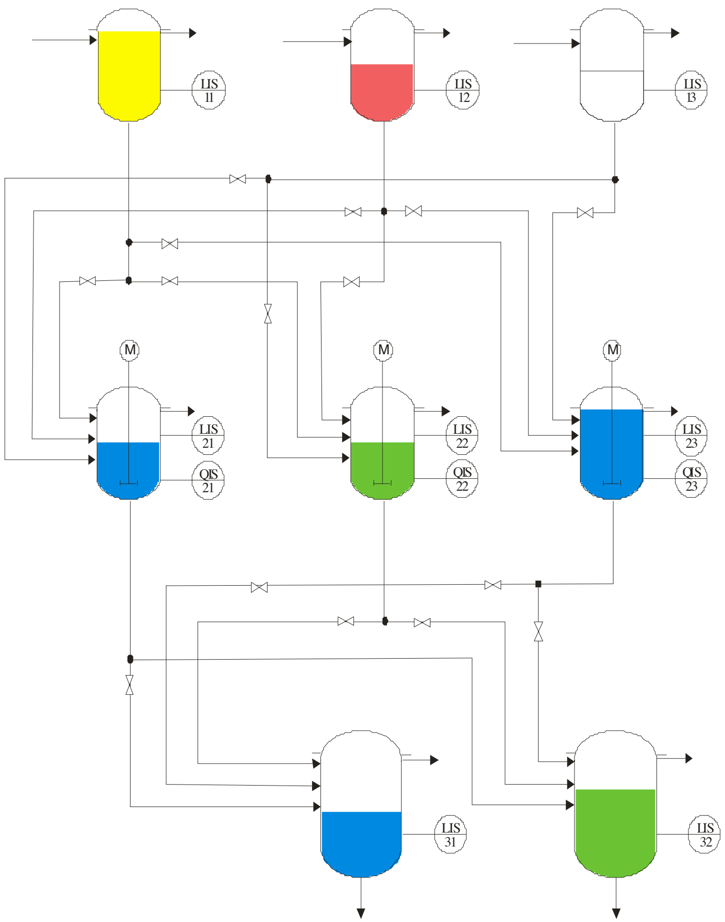
# Reachability with LHA [Halbwachs, Henzinger, 93-97]



# Multi-Product Batch Plant

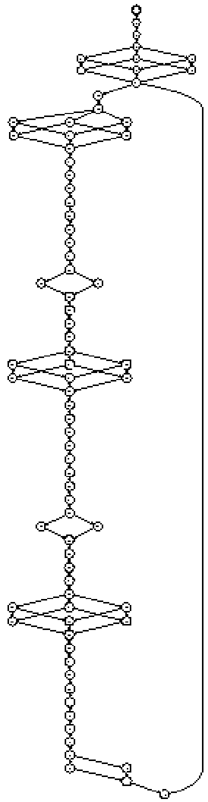


# Multi-Product Batch Plant

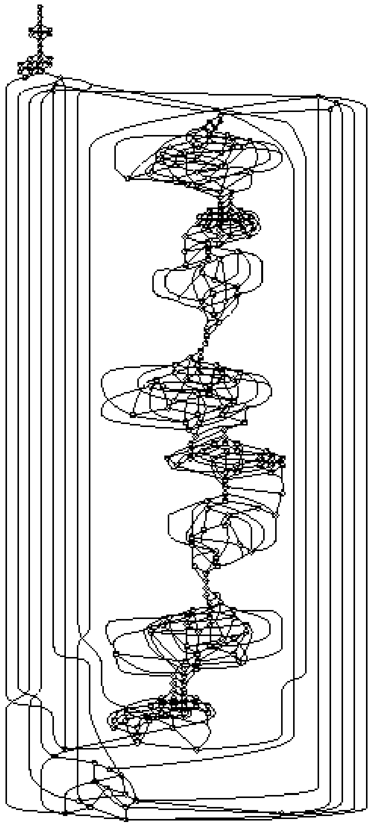



- **Cascade mixing process**
  - 3 educts via 3 reactors
    $\Rightarrow$ 2 products
- **Verification Goals**
  - Invariants
    - overflow
    - product tanks never empty
  - Filling sequence
- **Design of verified controller**

# Verification with PHAVer



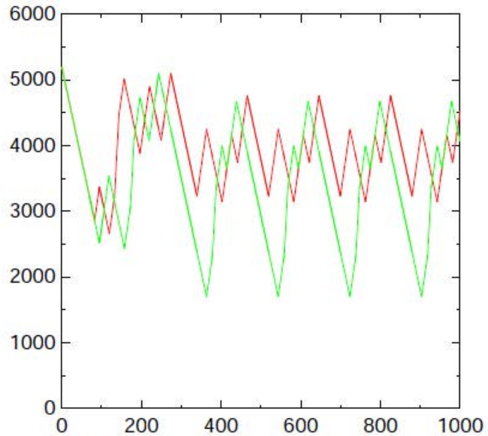
Controller



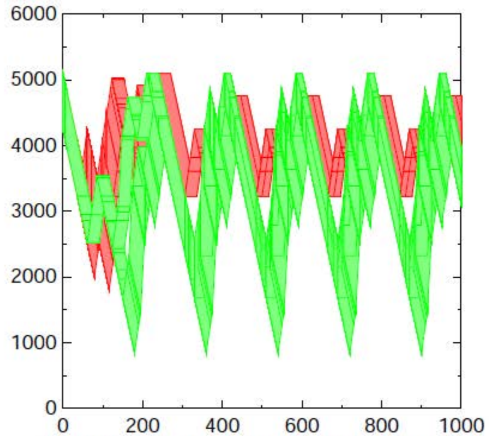
Controlled Plant

- **Controller + Plant**
  - 266 locations, 823 transitions (~150 reachable)
  - 8 continuous variables
- **Reachability over infinite time**
  - 120s—1243s, 260—600MB
  - computation cost increases with nondeterminism (intervals for throughputs, initial states)

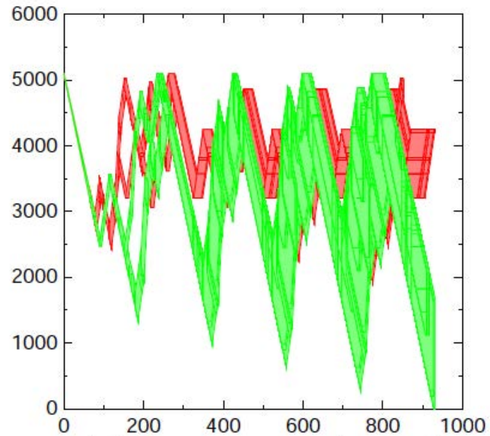
# Verification with PHAVer



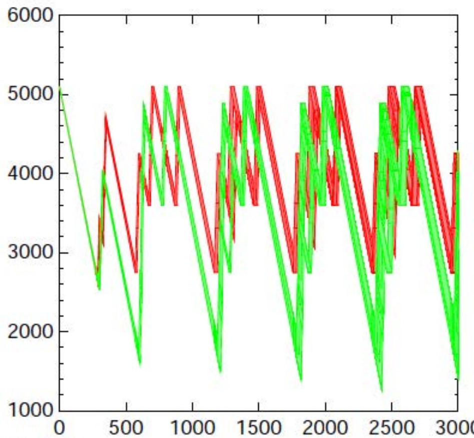
(a) BP8.1: nominal case



(b) BP8.2: varying initial cond.



(c) BP8.3: varying demand



(d) BP8.4: varying but slow demand

| | | | | | Automaton | | Reachable Set | |
|---|---|---|---|---|---|---|---|---|
| Instance | Time [s] | Mem. [MB] | Depth[a] | Checks[b] | Loc. | Trans. | Loc. | Poly. |
| BP8.1 | 120 | 267 | 173 | 279 | 266 | 823 | 130 | 279 |
| BP8.2 | 139 | 267 | 173 | 422 | 266 | 823 | 131 | 450 |
| BP8.3 | 845 | 622 | 302 | 2669 | 266 | 823 | 143 | 2737 |
| BP8.4 | 1243 | 622 | 1071 | 4727 | 266 | 823 | 147 | 4772 |

* on Xeon 3.20 GHz, 4GB RAM running Linux; [a] lower bound on depth in breadth-first search; [b] number of applications of post-operator

# Outline

- **Modeling with Hybrid Automata**
- **Reachability versus Simulation**
- **Reachability Algorithms**
  - piecewise constant dynamics
  - piecewise affine dynamics
- **Case Study: Controller Implementation**
- **SpaceEx Tool Platform**
- **Bibliography**

# Piecewise Affine Dynamics

- **Not quite so simple dynamics**
  - trajectories = exponential functions
- **Exact computation at discrete points in time**
  - used to overapproximate continuous time
- **Efficient data structures**

# Time Elapse Computation

- **Continuous time elapse for affine dynamics**
  - efficient, scalable
  - approximation without accumulation of approximation error (wrapping effect)
- **It took a long time to do it well...**
  - Chutinan, Krogh. HSCC'99
  - Asarin, Bournez, Dang, Maler. HSCC'00
  - Girard. HSCC'05
  - Le Guernic, Girard. HSCC'06, CAV'09
  - Frehse, Kateja, Le Guernic. HSCC'13

# Affine Dynamics

- **linear terms plus inputs $U$:**

$$\dot{x} = Ax + u, \; u \in U$$

- **solution:**

$$x(t) = e^{At}x(0) + \int_0^t e^{A(t-\tau)}u(\tau)d\tau$$

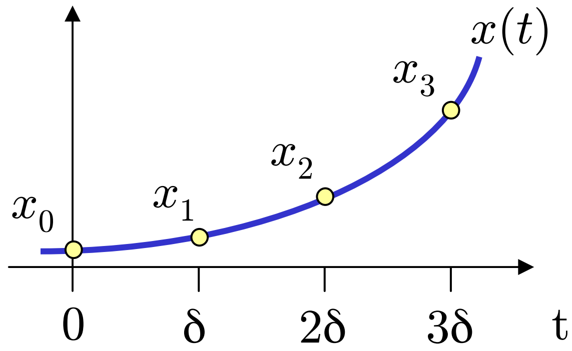
matrix exponential

factors influence of inputs
(stable system forgets the past)

# Time-Discretization (no inputs)

- **Analytic solution:** $x(t) \quad = \quad e^{At} x_{Ini}$
  - with $t = \delta k$ :

$$x(\delta(k+1)) \quad = \quad e^{A\delta} x(\delta k)$$




- **Explicit solution in discretized time (recursive):**

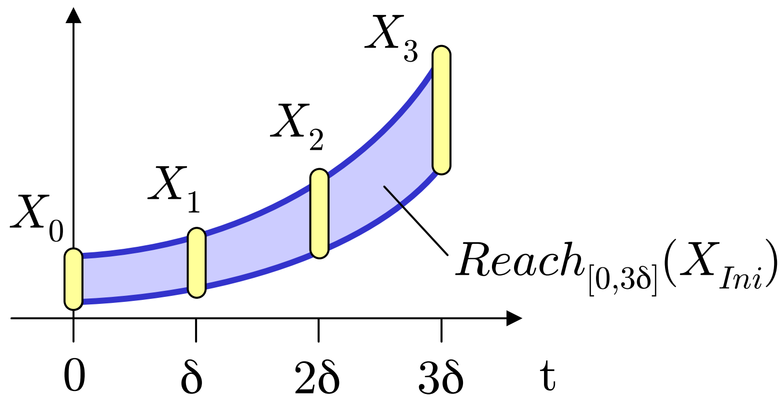
$$\begin{aligned} x_0 &= x_{Ini} \\ x_{k+1} &= e^{A\delta} x_k \end{aligned}$$

multiplication with const. matrix $e^{A\pm}$
= linear transform

# Time-Discretization for an Initial Set

- **Explicit solution in discretized time**

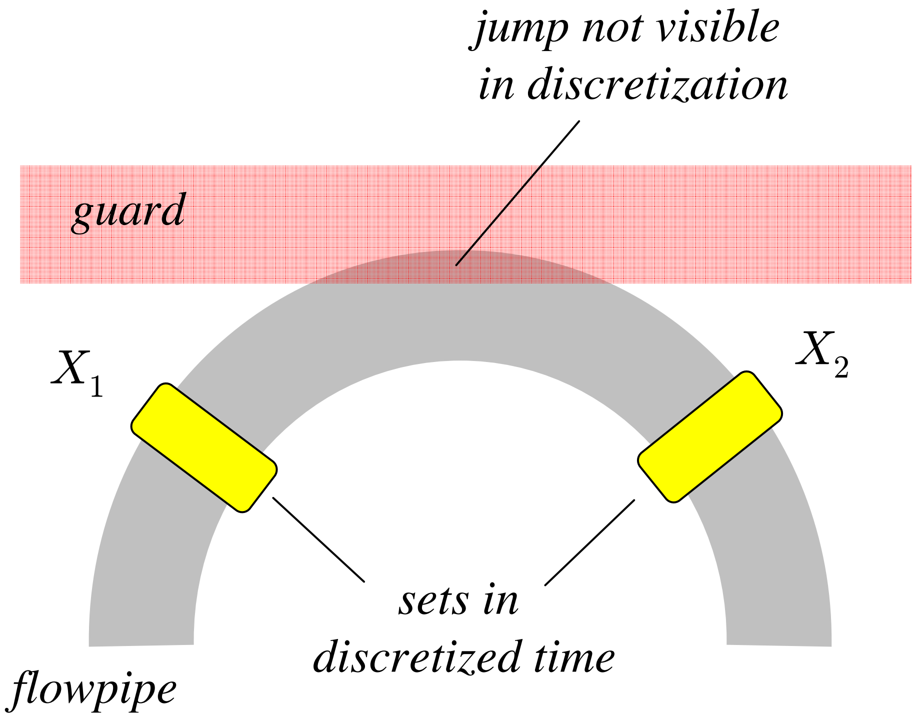
$$
\begin{aligned}
X_0 &= X_{Ini} \\
X_{k+1} &= e^{A\delta} X_k
\end{aligned}
$$



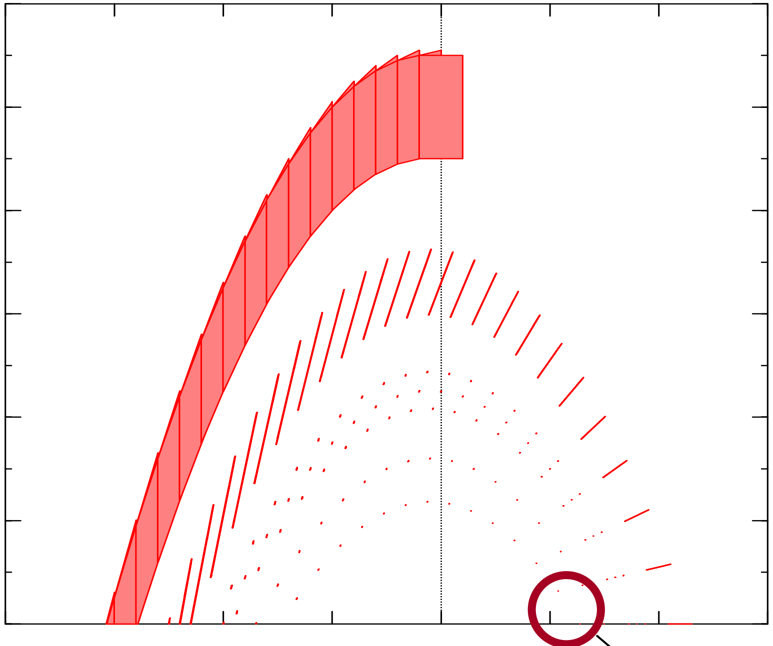

- **Acceptable solution for purely continuous systems**
  - $x(t)$ is in $\epsilon(\delta)$-neighborhood of some $X_k$
- **Unacceptable for hybrid systems**
  - discrete transitions might “fire” between sampling times
  - if transitions are “missed,” $x(t)$ not in $\epsilon(\delta)$-neighborhood

# Time Discretization for Hybrid Systems

- **One can miss jumps (guard)**



# Bouncing Ball



$X_{90} = \emptyset$

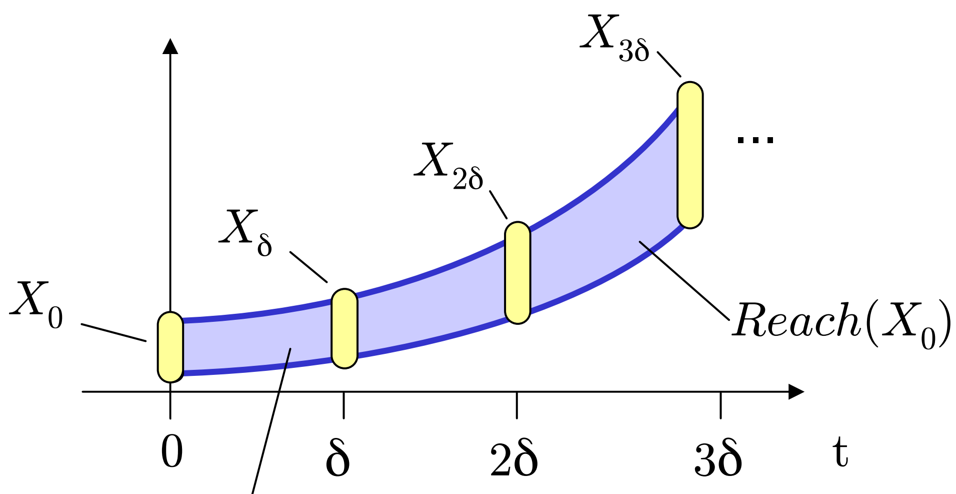
- Note: Computed in exact arithmetic, no numerical errors
- In other examples this error might not be as obvious…

# From Time-Discretization to Reach

- **States in discrete time:**

$$X_{k\delta} \quad = \quad (e^{A\delta})^k X_0 \oplus S_{k\delta}$$

integral over inputs

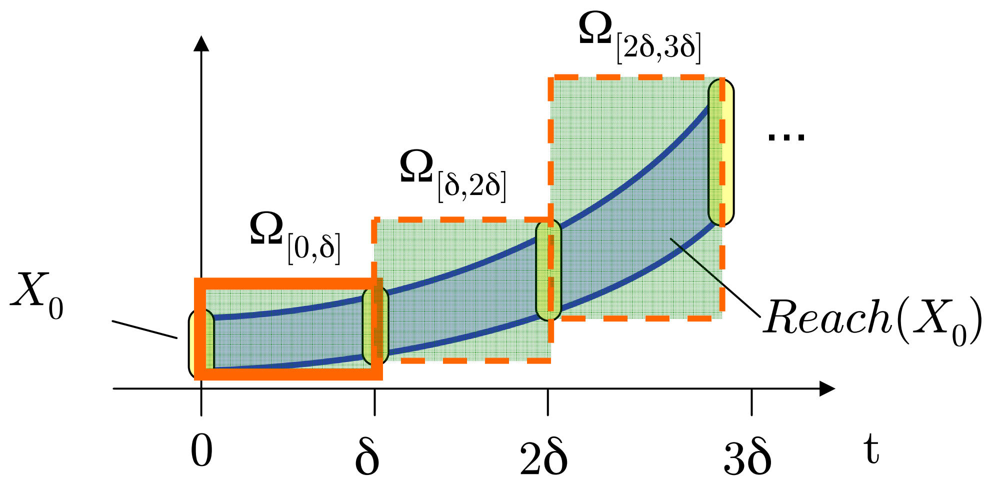



need to cover also states in between!

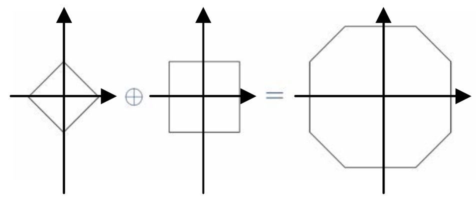
# From Time-Discretization to Reach

- **Cover in discrete time:**

$$\Omega_{[k\delta,(k+1)\delta]} \quad = \quad (e^{A\delta})^k \Omega_{[0,\delta]} \oplus \Psi_{k\delta}$$

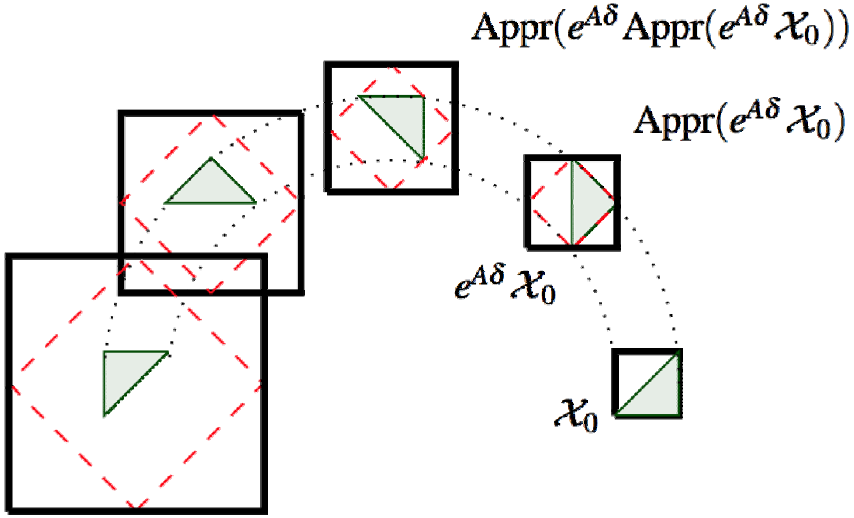



$\oplus$ Minkowski sum = pointwise sum of sets

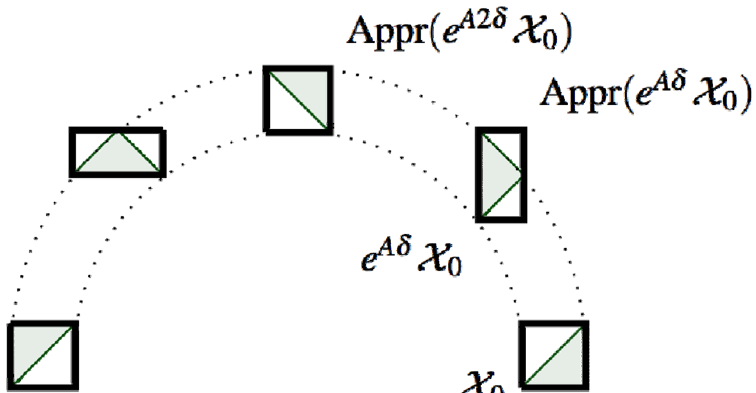


# Wrapping Effect

- accumulation of approximation error
- avoidable using the right approximation




wrapping effect




wrapping-free algorithm

Antoine Girard, Colas Le Guernic, and Oded Maler. Efficient computation of reachable sets of linear time-invariant systems with inputs. HSCC 2006

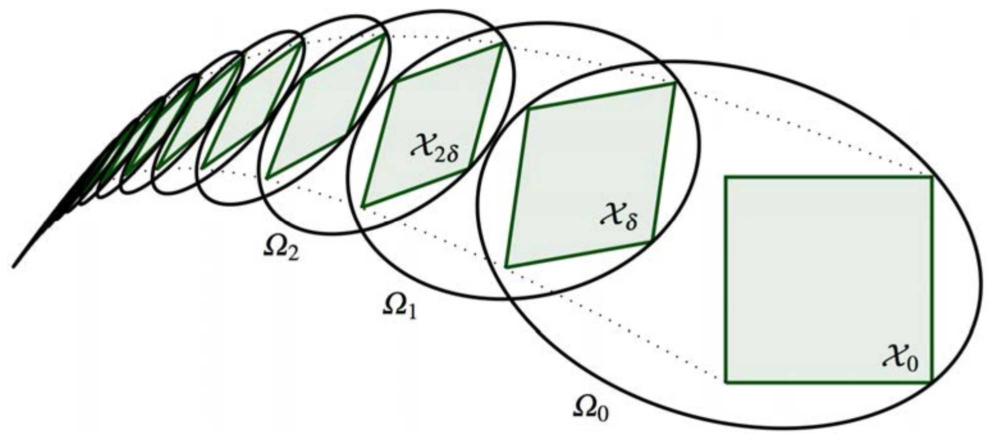
# Reachability in High Dimensions

- **Scalability Trick 1:**

  **Use data structures adapted to operations**

# Scalable Set Representations

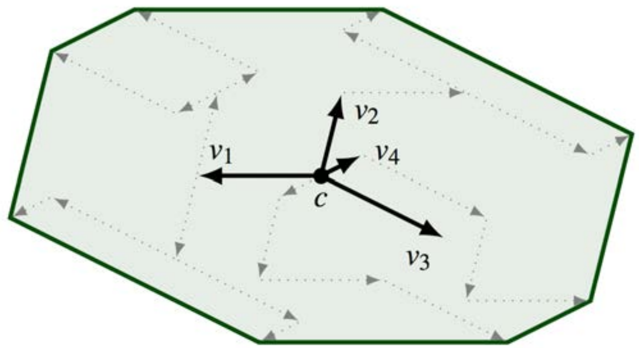
- **Ellipsoids** **[Kurzhansky, Varaiya 2006]**
  - bad representation of intersection, convex hull, flat sets

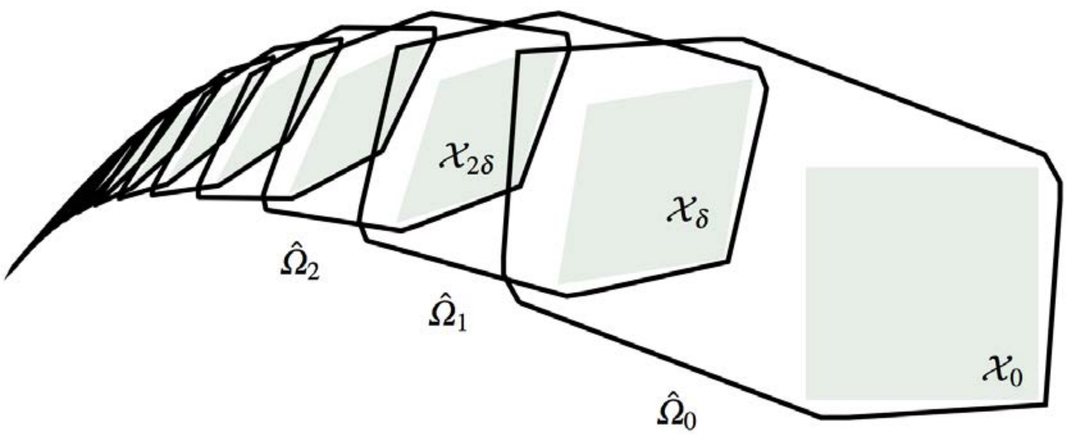



(this is an illustration, not actual computation)

# Scalable Set Representations

- **Zonotopes** [Girard 2005]
  - symmetric polytope spanned by set of generator vectors
  - bad representation of intersection, convex hull, asymmetric sets

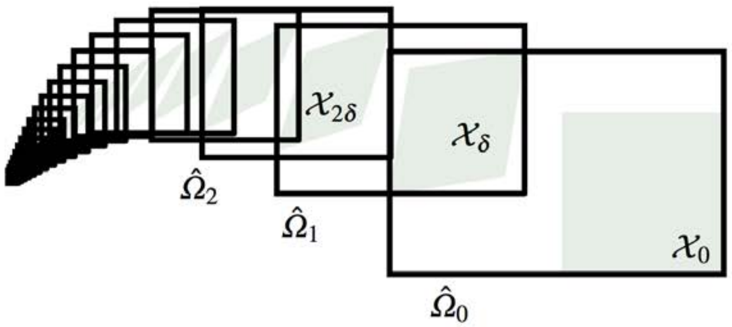


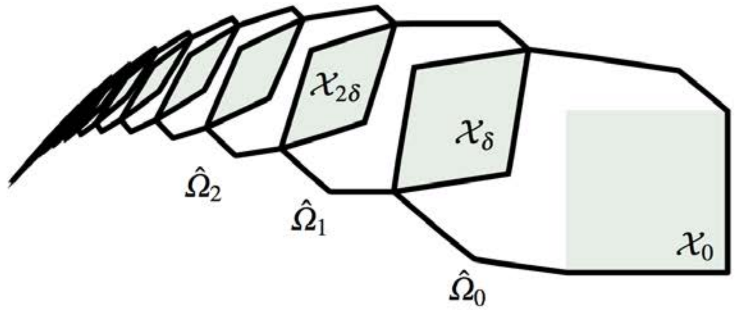

(computed with Zonotope toolbox of M. Althoff)

# Scalable Set Representations

- **Support Functions** **[Le Guernic, Girard 2009]**
  - lazy representation of any convex set
  - gives outer polyhedral approximation that can be refined
  - scalable except for intersection




low accuracy




high accuracy

(computed with SpaceEx)

# Operations on Convex Sets

| | Polyhedra | | | |
|---|---|---|---|---|
| **Operators** | Constraints | Vertices | **Zonotopes** | **Support F.** |
| Convex hull | -- | + | -- | ++ |
| Affine transform | +/- | ++ | ++ | ++ |
| Minkowski sum | -- | -- | ++ | ++ |
| Intersection | + | -- | -- | - |

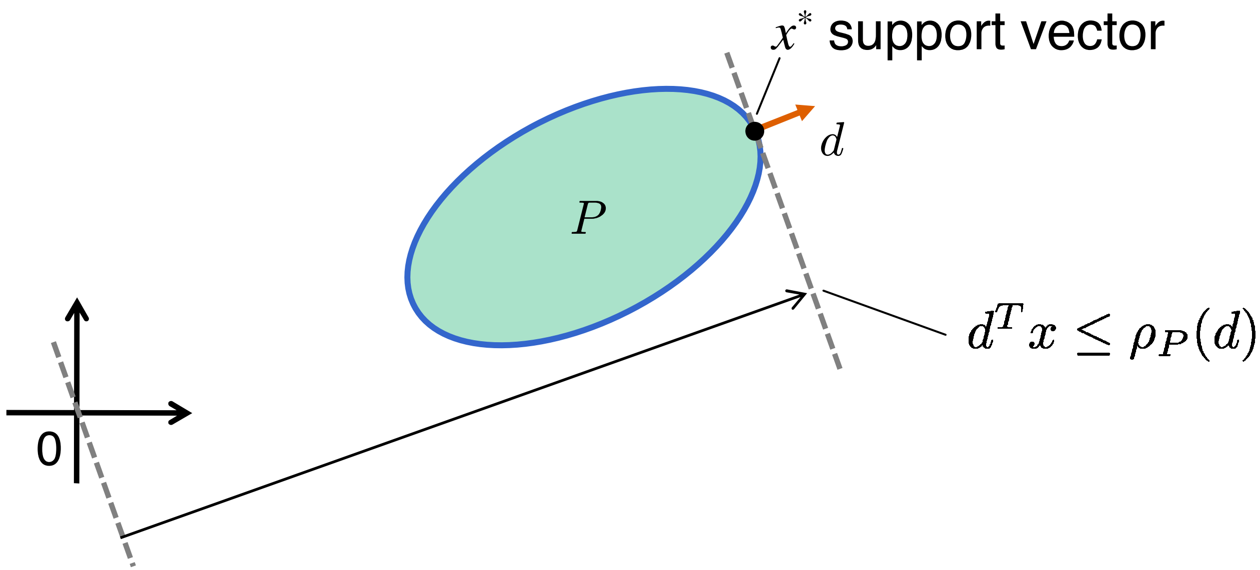
Le Guernic, Girard. CAV'09

# Support Functions

- **Support Function** $R^n \rightarrow \; R$
  - direction $d \rightarrow$ position of supporting halfspace

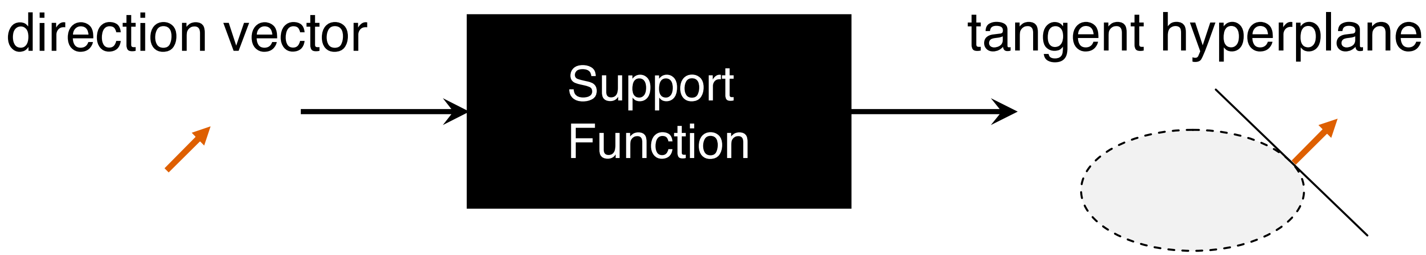
$$\rho_P(d) \quad = \quad \max_{x \in P} d^T x$$

  - exact set representation



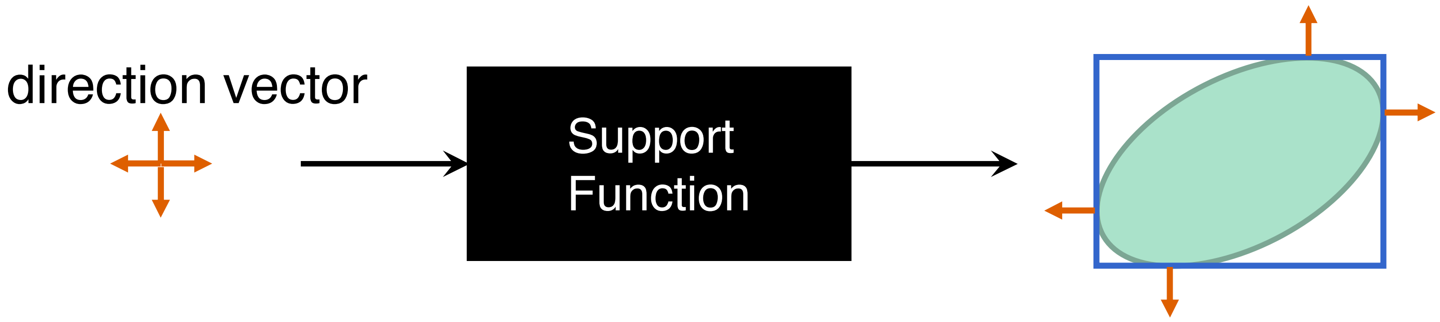
# Support Functions

- **black box representation of a convex set**
- **implementation: function objects**



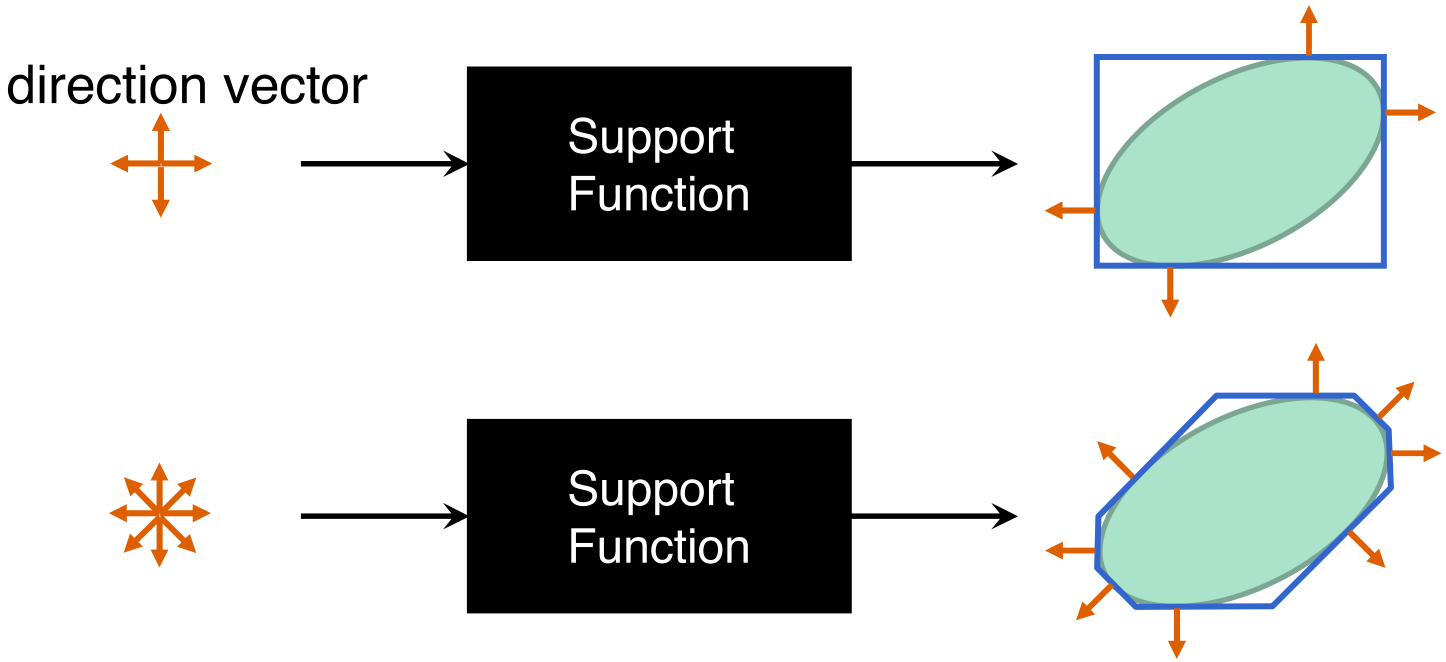
# Support Functions

- **black box representation of a convex set**
- **implementation: function objects**



# Support Functions

- **black box representation of a convex set**
- **implementation: function objects**

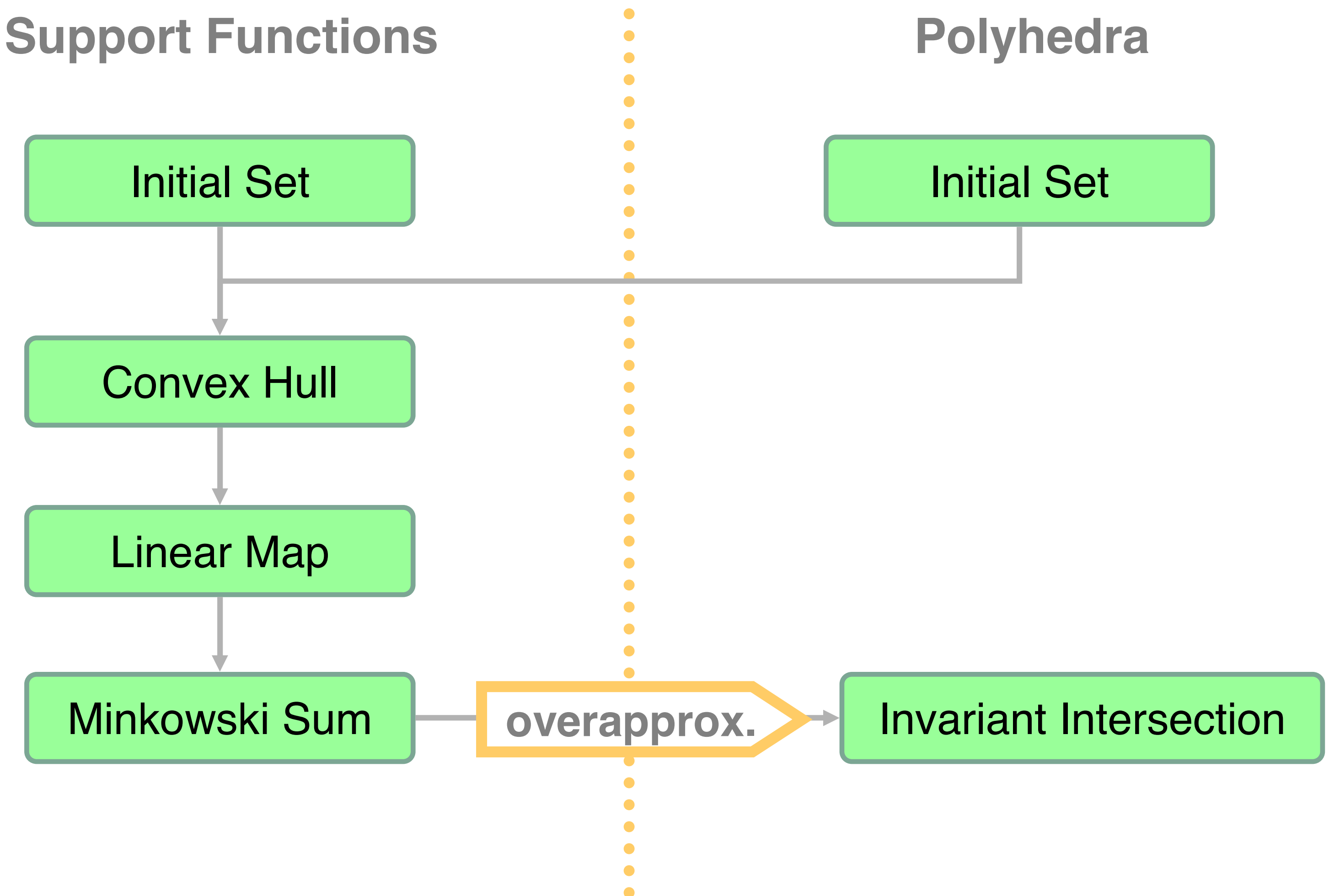


# Reachability in High Dimensions

- **Scalability Trick 2:**

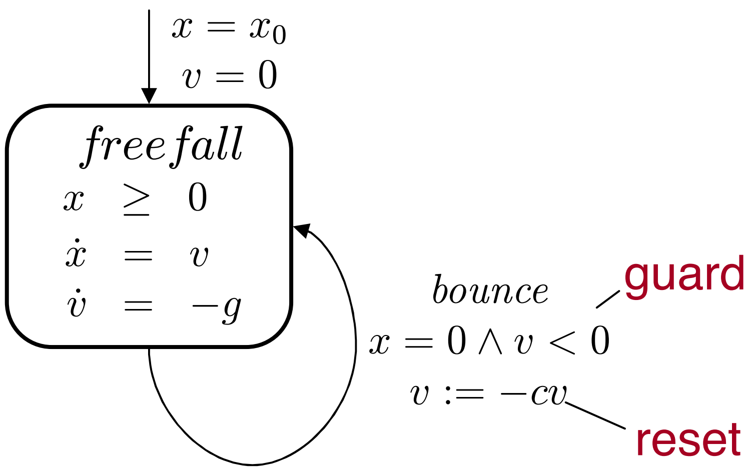
  **Change data structures (data-dependent)**

# Computing Time Elapse

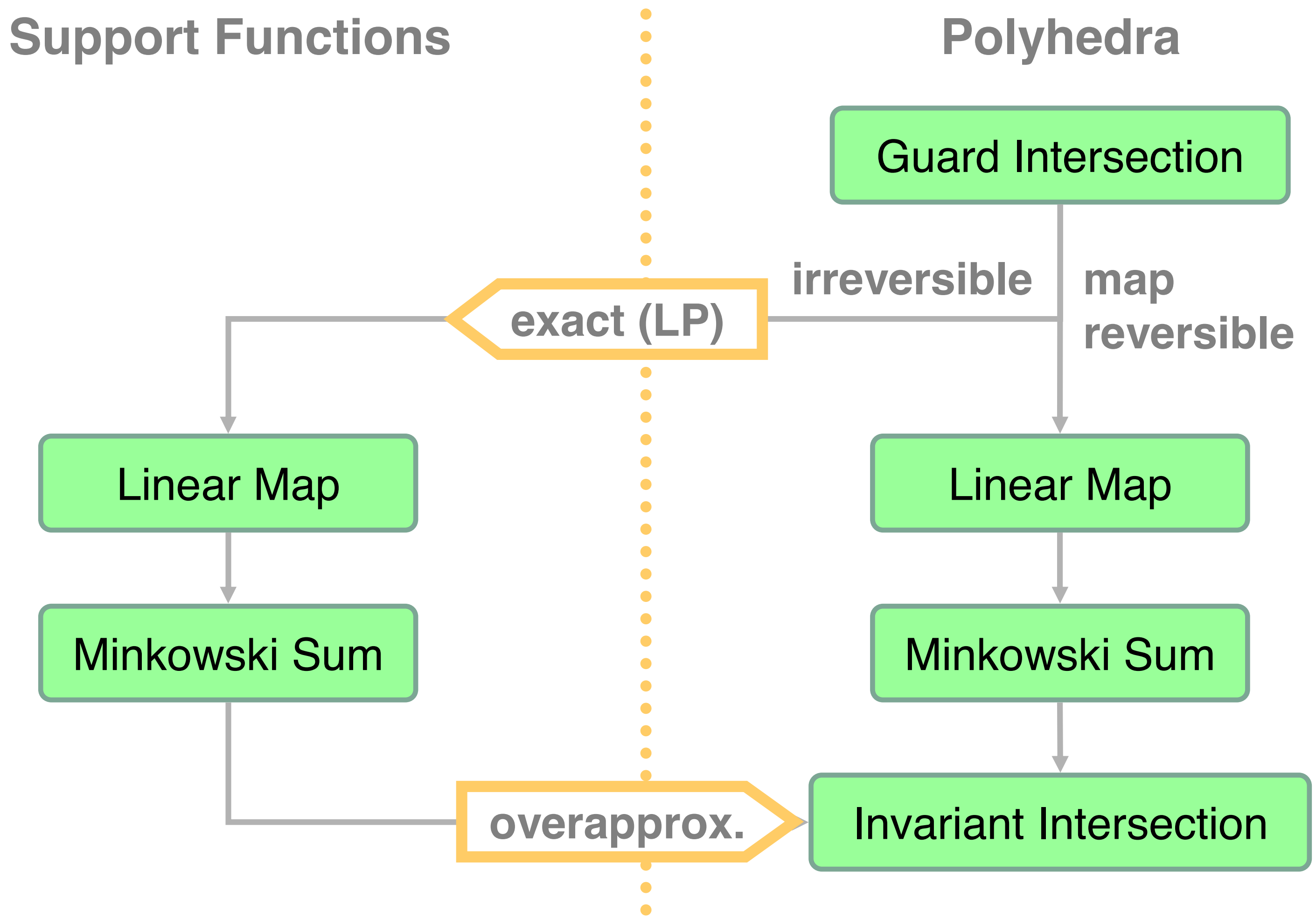


# Computing Transition Successors

- **Intersection with guard**
  - use outer poly approximation
- **Linear map & Minkowski sum**
  - with polyhedra if invertible (map regular, input set a point)
  - otherwise use support functions
- **Intersection with target invariant**
  - use outer poly approximation

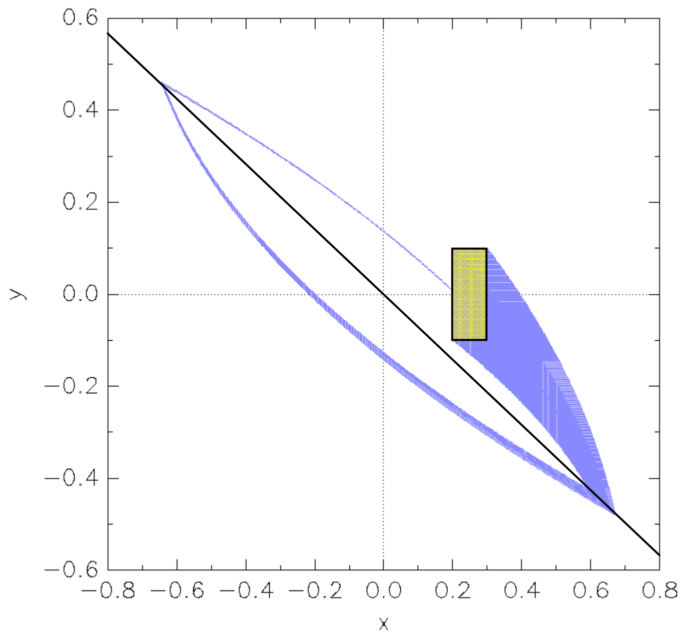


# Computing Transition Successors



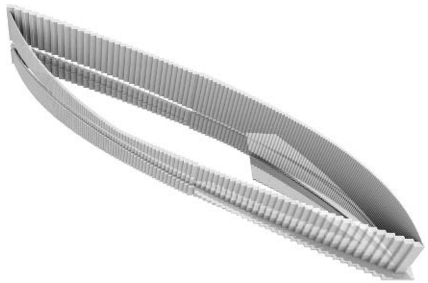
# Example: Switched Oscillator

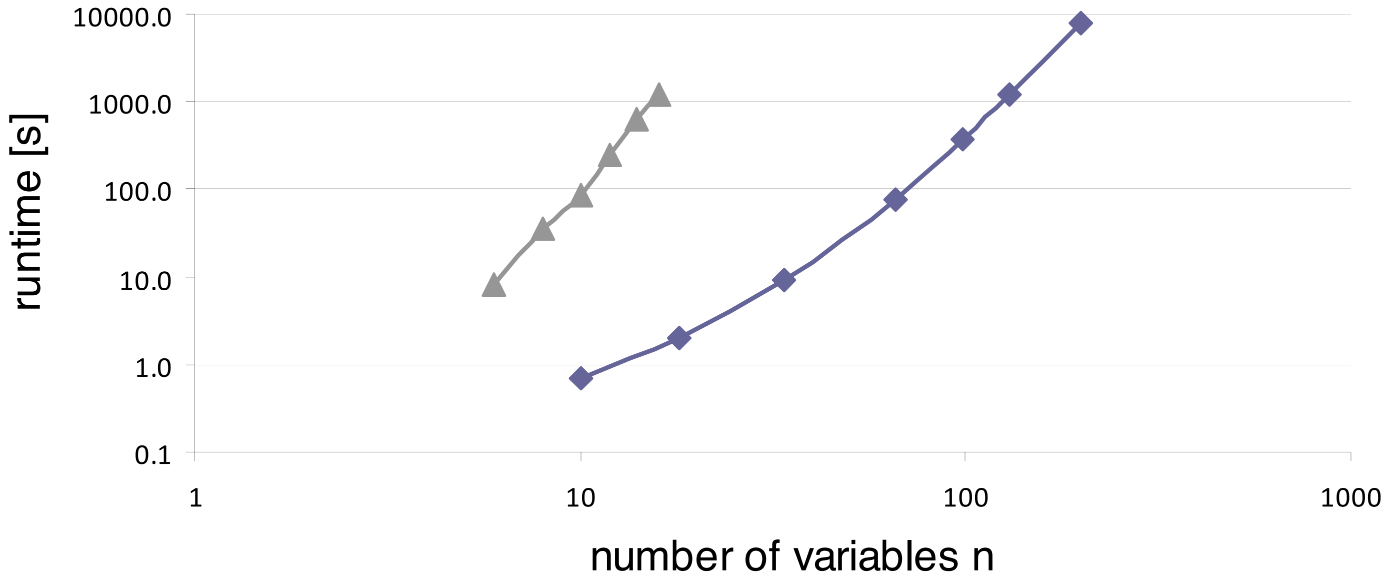
- **Switched oscillator**
  - 2 continuous variables
  - 4 discrete states
  - similar to many circuits (Buck converters,…)
- **plus linear filter**
  - $m$ continuous variables
  - dampens output signal
- **affine dynamics**
  - total $2 + m$ continuous variables



# Example: Switched Oscillator

- **Scalability Measurements:**
  - fixpoint reached in $\mathrm{O}(nm^2)$ time
  - box constraints: $\mathrm{O}(n^3)$
  - octagonal constraints: $\mathrm{O}(n^5)$



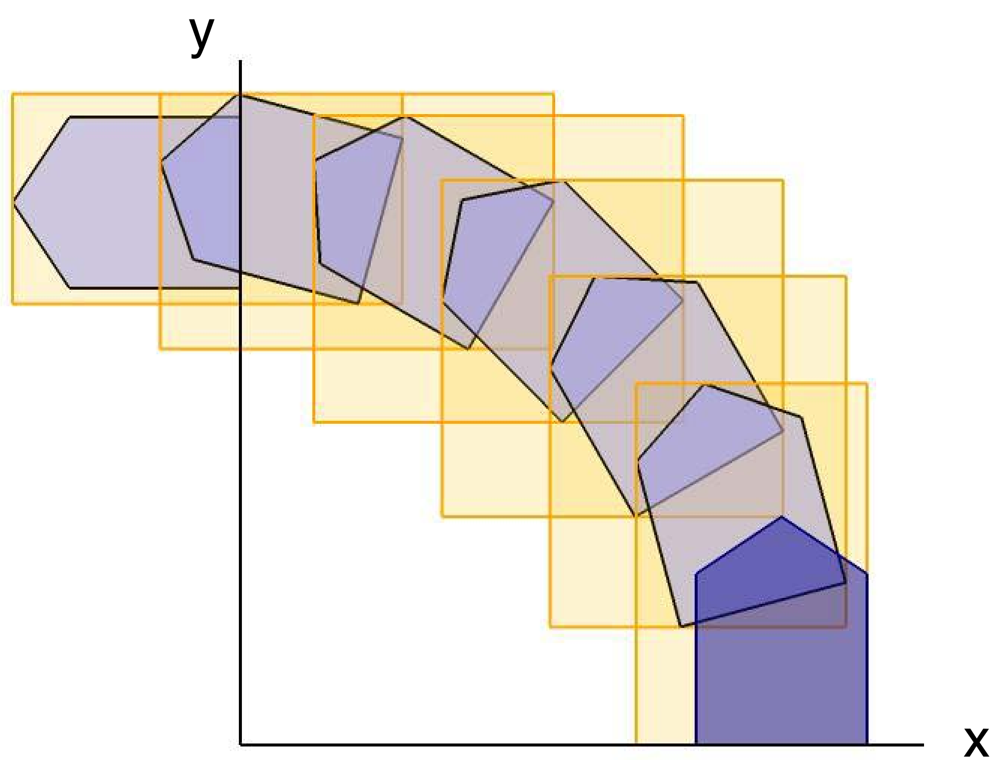


# Reachability in High Dimensions

- **Scalability Trick 3:**

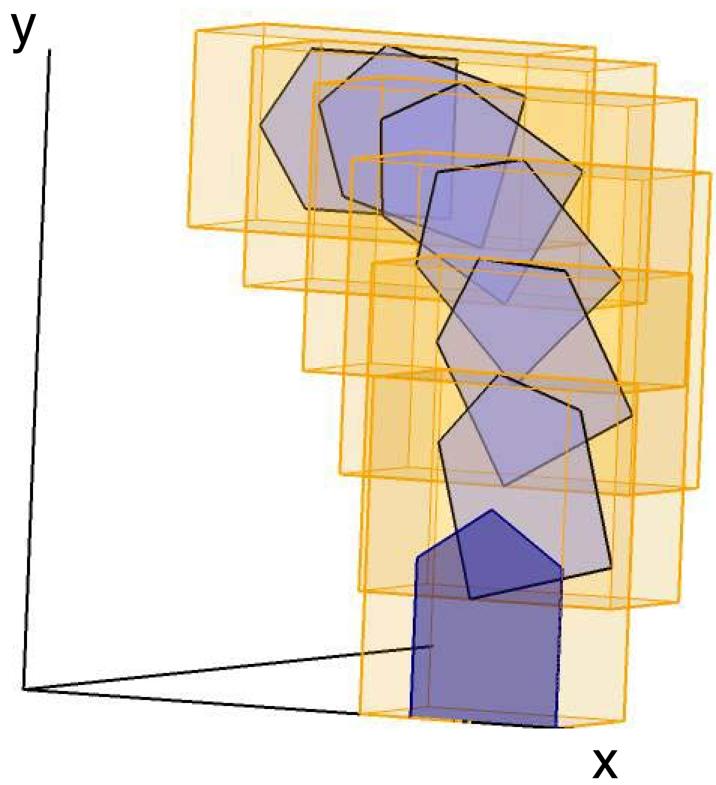
  **Work in Space-Time (exploit pointwise convexity)**

# Approximation in Space-Time

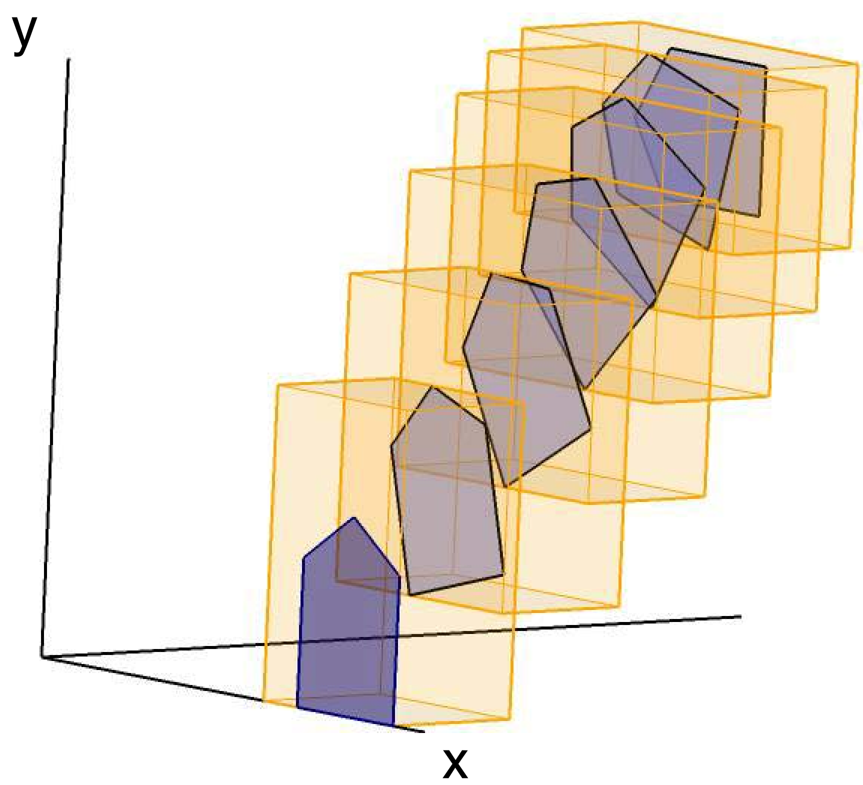



Improve the approximation by adding time...

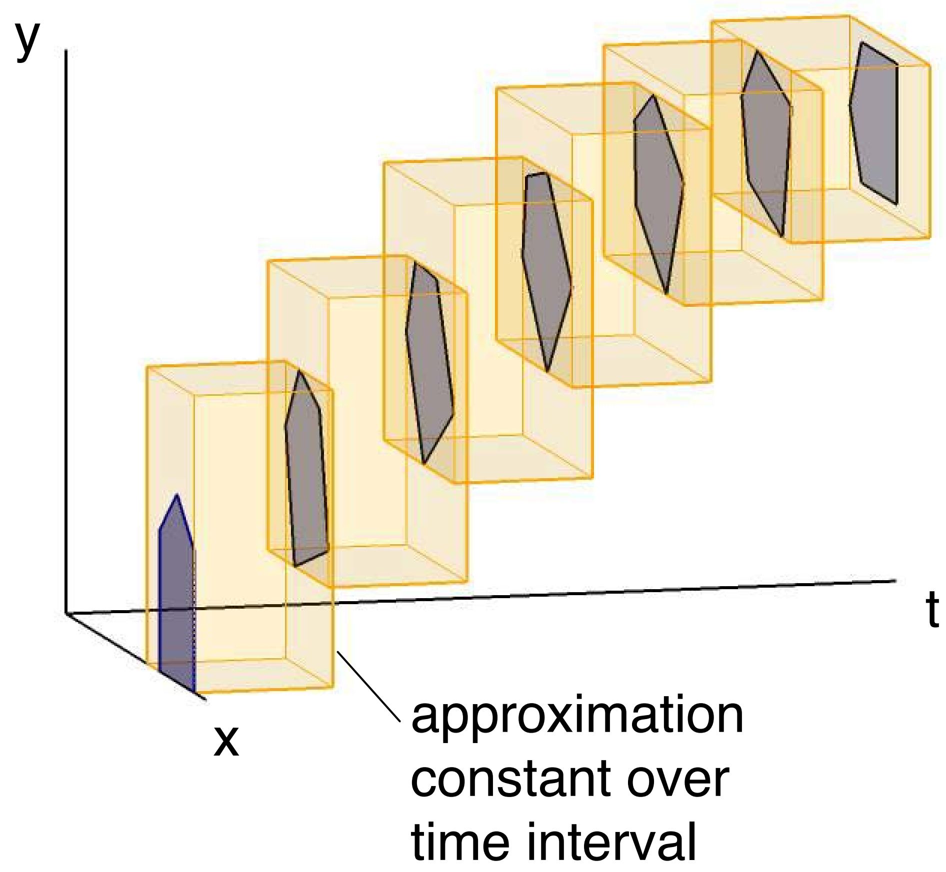
# Approximation in Space-Time



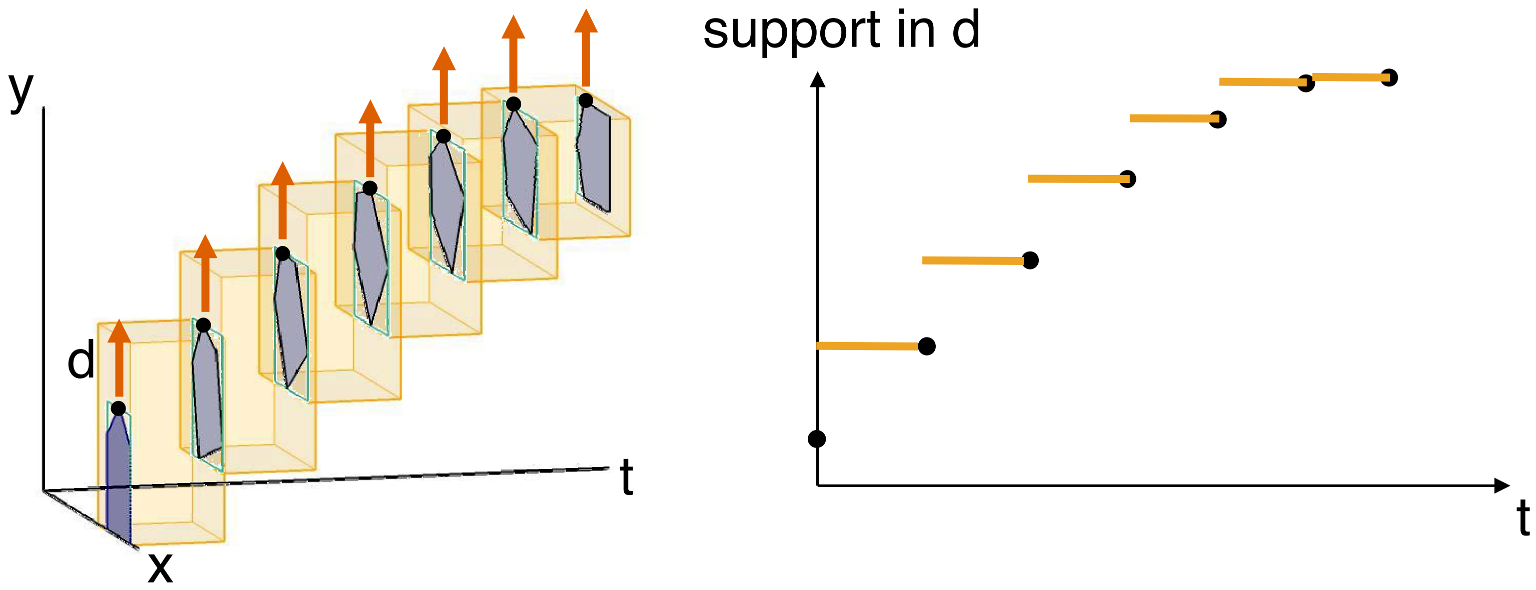
# Approximation in Space-Time



# Approximation in Space-Time



# Support Function over Time



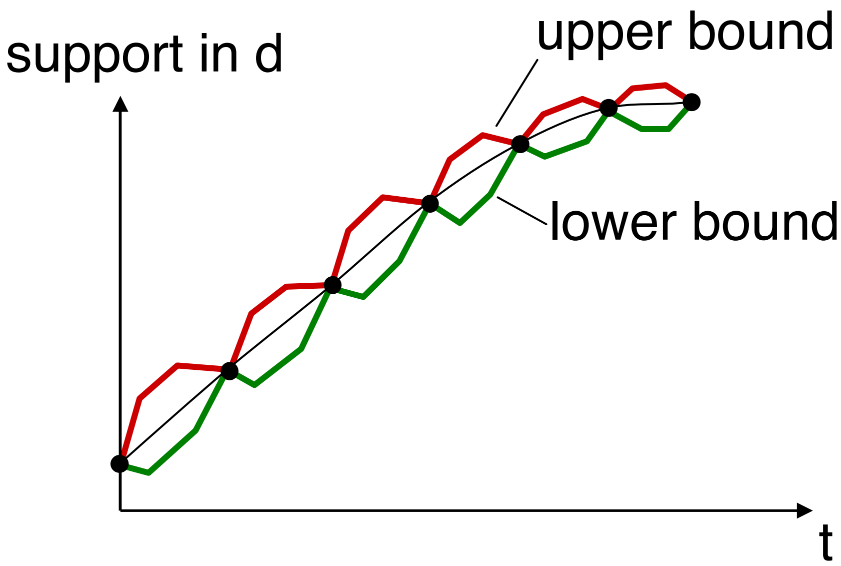

**convex set per time interval =**
**piecewise constant scalar functions**

# Support Function over Time

- **1st order Taylor approx.**
  CAV'11

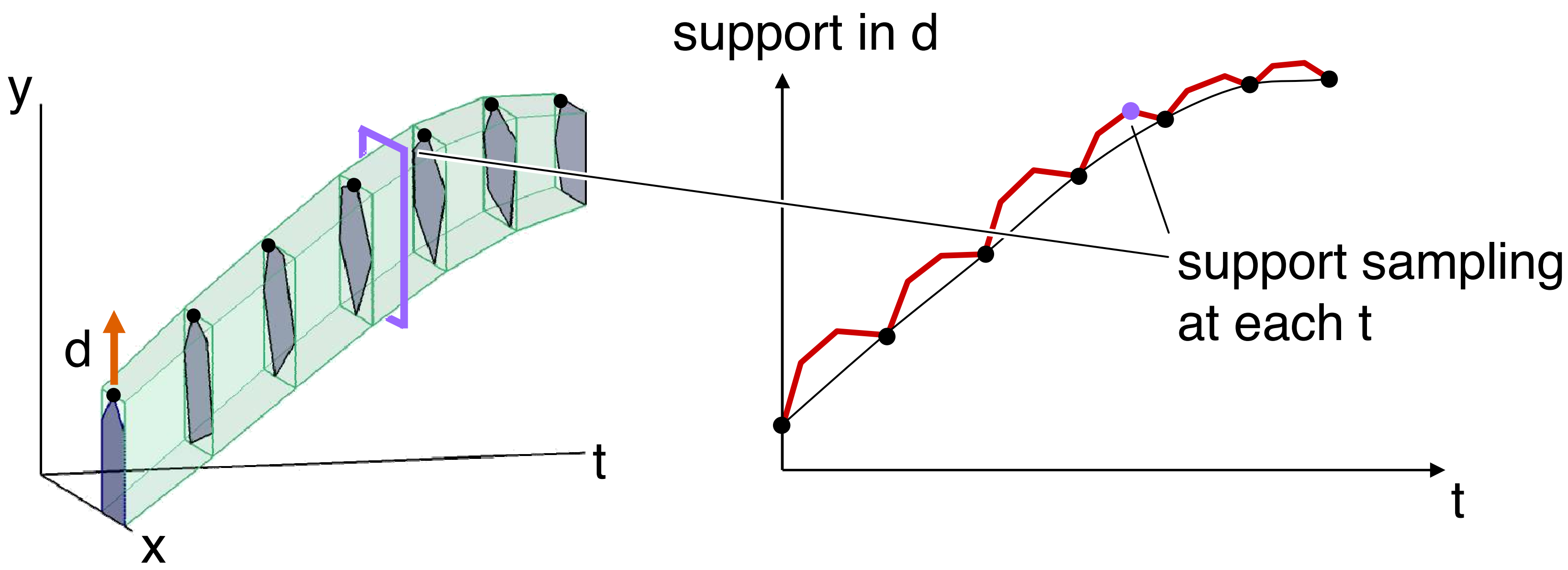
$$\Omega_t = (1-\tfrac{t}{\delta})\mathcal{X}_0 \oplus \tfrac{t}{\delta}e^{\delta A}\mathcal{X}_0$$
$$\oplus \left(\tfrac{t}{\delta}\mathcal{E}_\Omega^+ \cap (1-\tfrac{t}{\delta})\mathcal{E}_\Omega^-\right)$$
$$\oplus t\mathcal{U} \oplus \tfrac{t^2}{\delta^2}\mathcal{E}_\Psi$$

$$\Phi_2(A,\delta) = A^{-2}\left(e^{\delta A} - I - \delta A\right)$$
$$\mathcal{E}_\Omega^+(\mathcal{X}_0,\delta) = \boxdot\left(\Phi_2(|A|,\delta)\boxdot\left(A^2\mathcal{X}_0\right)\right),$$
$$\mathcal{E}_\Omega^-(\mathcal{X}_0,\delta) = \boxdot\left(\Phi_2(|A|,\delta)\boxdot\left(A^2e^{\delta A}\mathcal{X}_0\right)\right),$$
$$\mathcal{E}_\Psi(\mathcal{U},\delta) = \boxdot\left(\Phi_2(|A|,\delta)\boxdot(A\mathcal{U})\right).$$



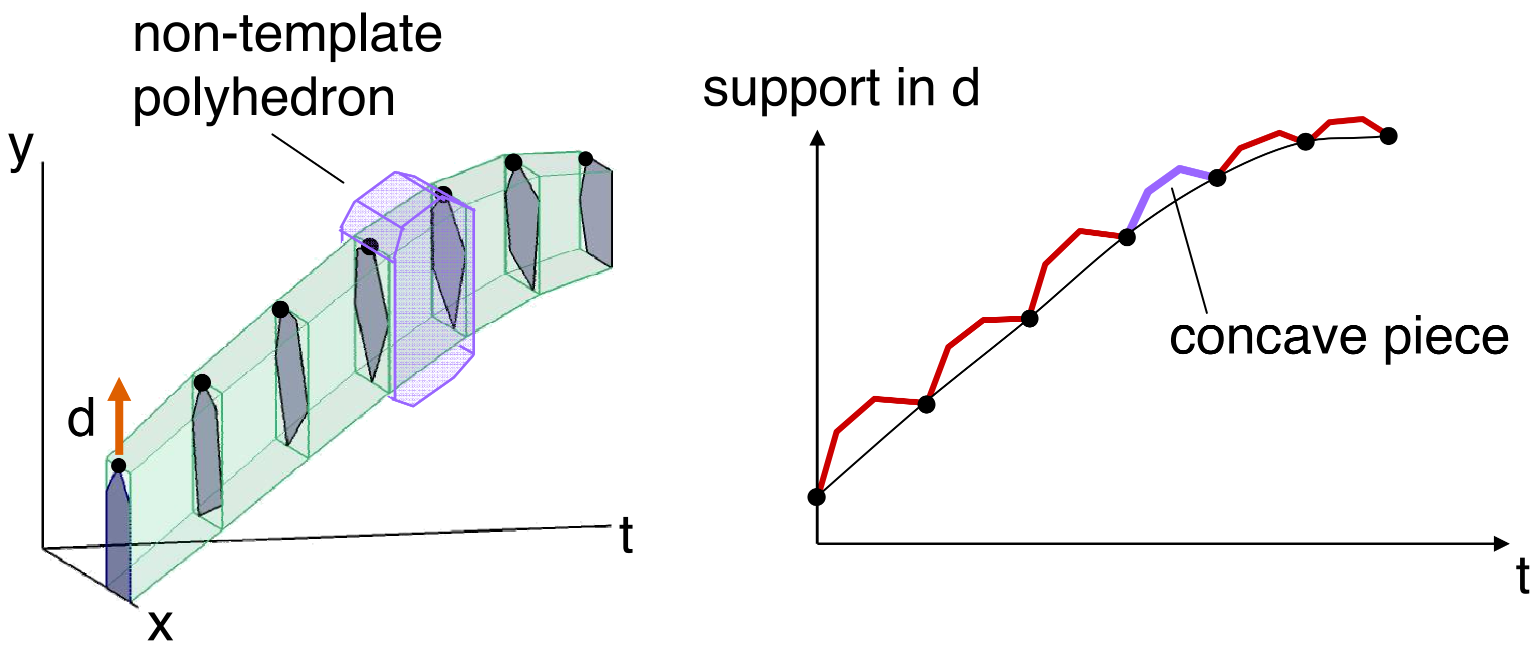

**interpolation with**
**piecewise linear scalar functions**

# Support Function over Time



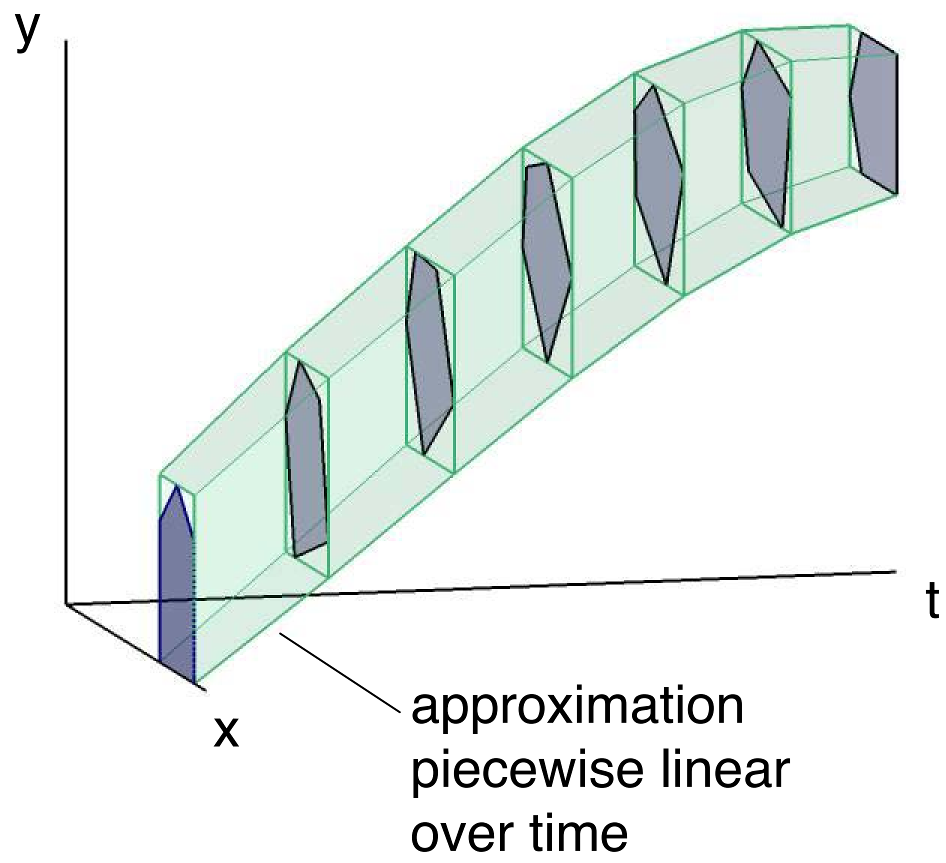

**infinite** **union of** **template** **polyhedra**
**(one for each t)**

# Convexification

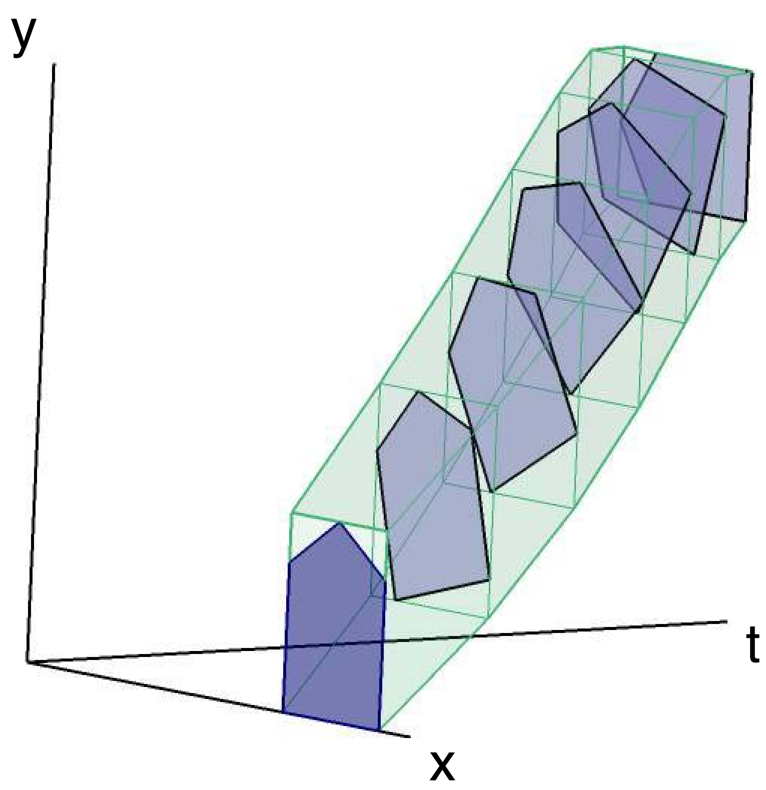



**finite** **union of** **non-template** **polyhedra**
**(one for each concave piece)**

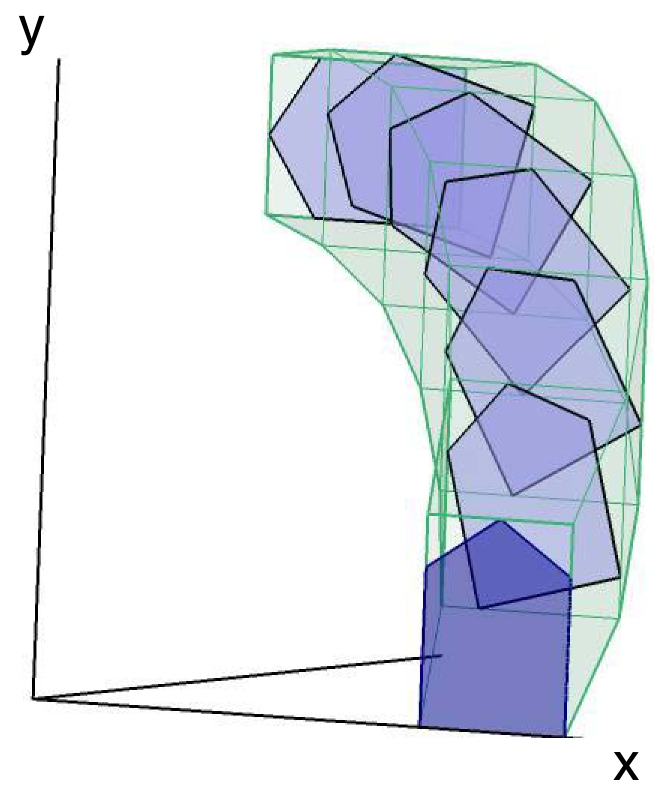
# Approximation in Space-Time



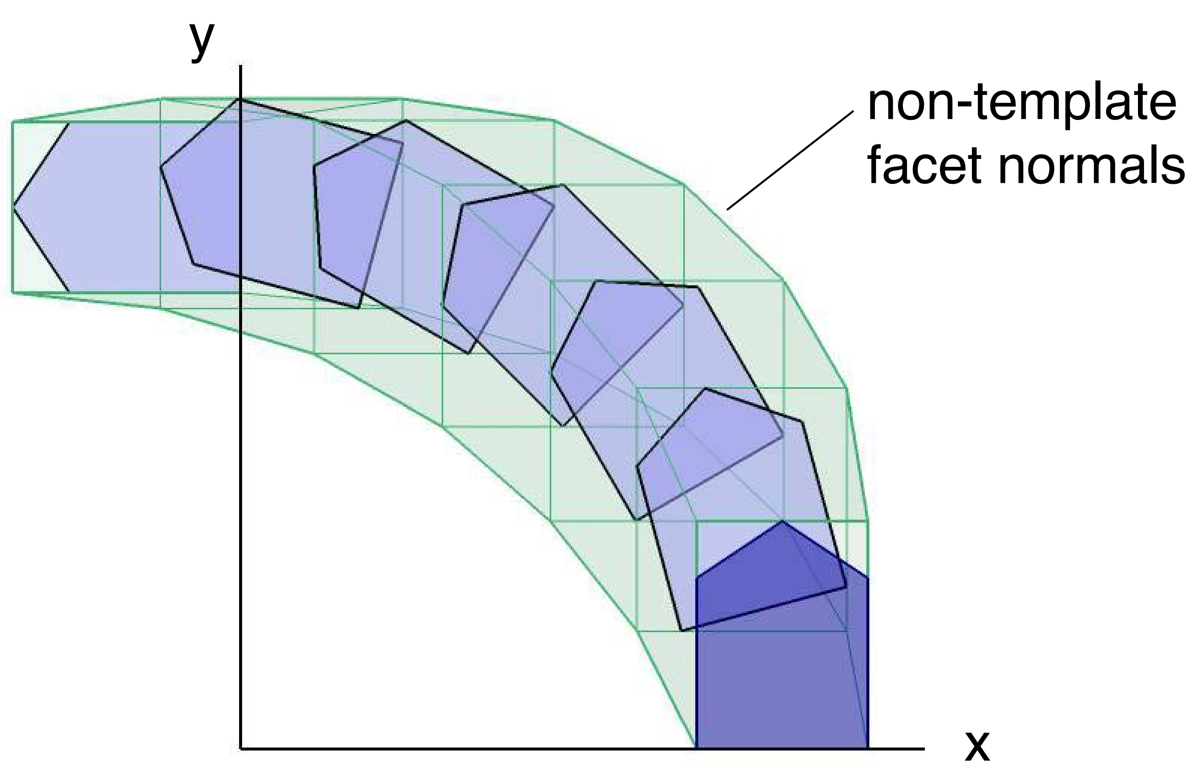
# Approximation in Space-Time



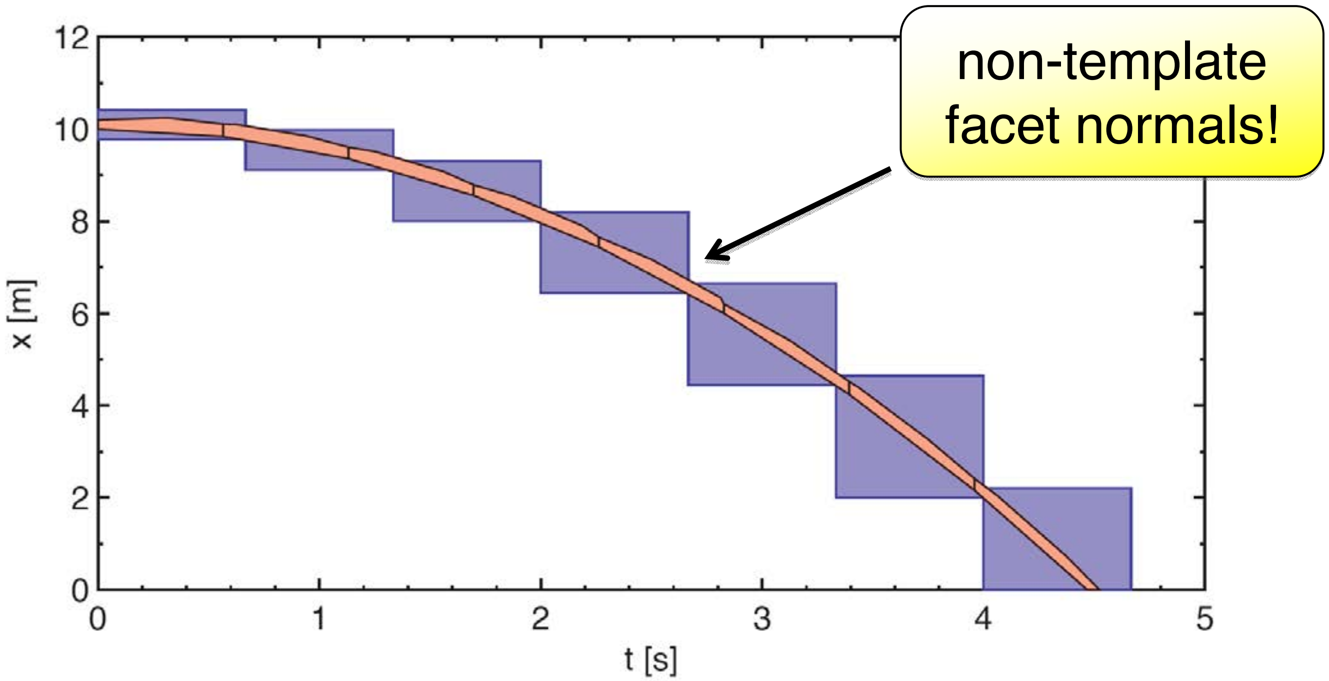
# Approximation in Space-Time



# Approximation in Space-Time

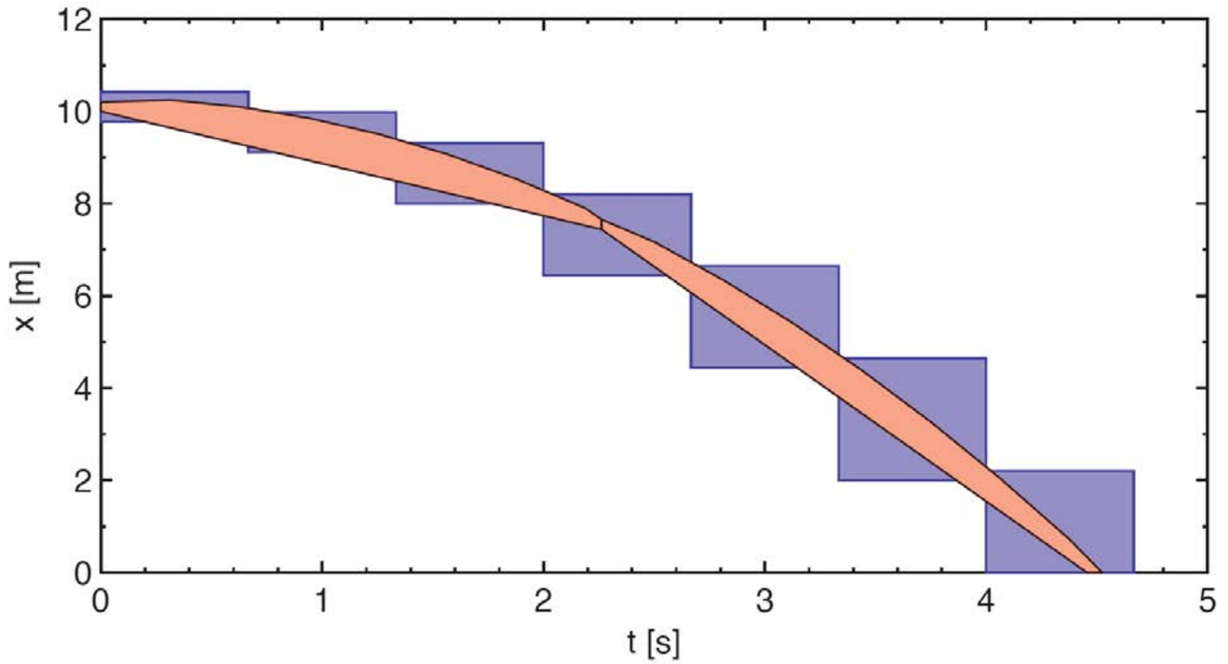


# Example: Bouncing Ball




Clustering up to total error 0.1 = 8 pieces

# Example: Bouncing Ball




Clustering up to total error 1.0 = 2 pieces

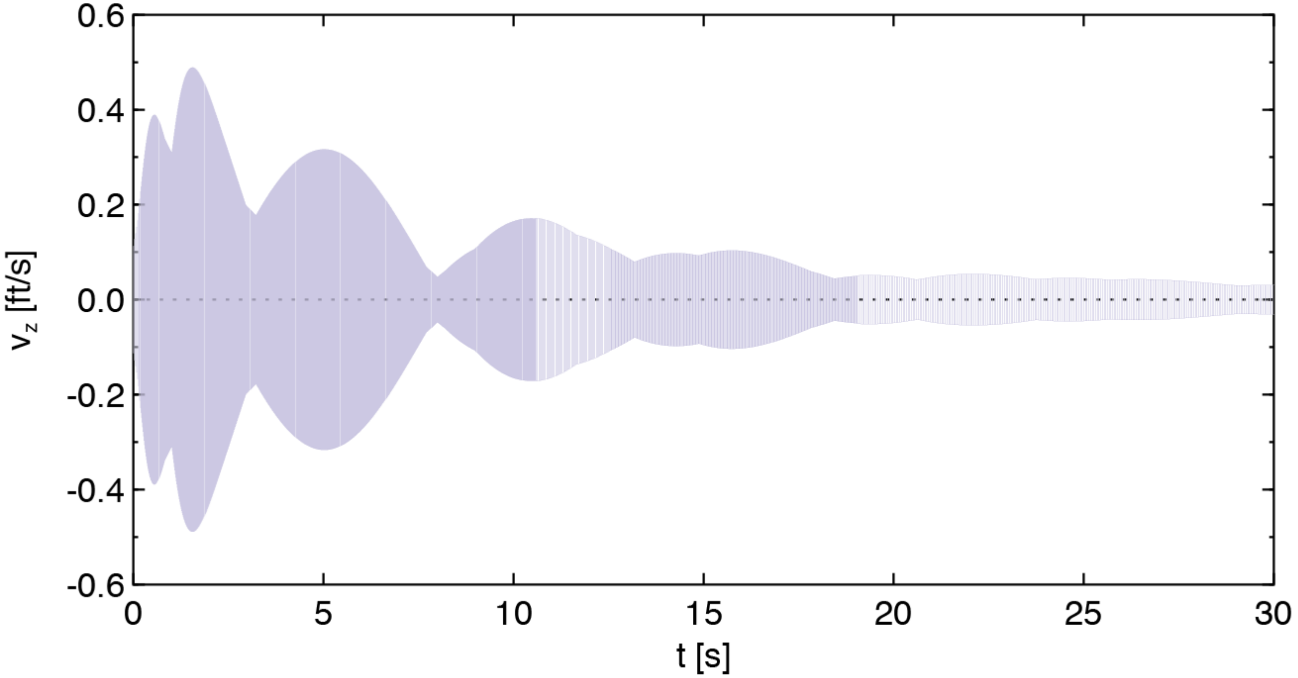
# Example: Controlled Helicopter


- **28-dim model of a Westland Lynx helicopter**
  - 8-dim model of flight dynamics
  - 20-dim continuous H∞ controller for disturbance rejection
  - stiff, highly coupled dynamics

S. Skogestad and I. Postlethwaite, Multivariable Feedback Control: Analysis and Design. John Wiley & Sons, 2005.

# Example: Helicopter

- **28 state variables + clock**

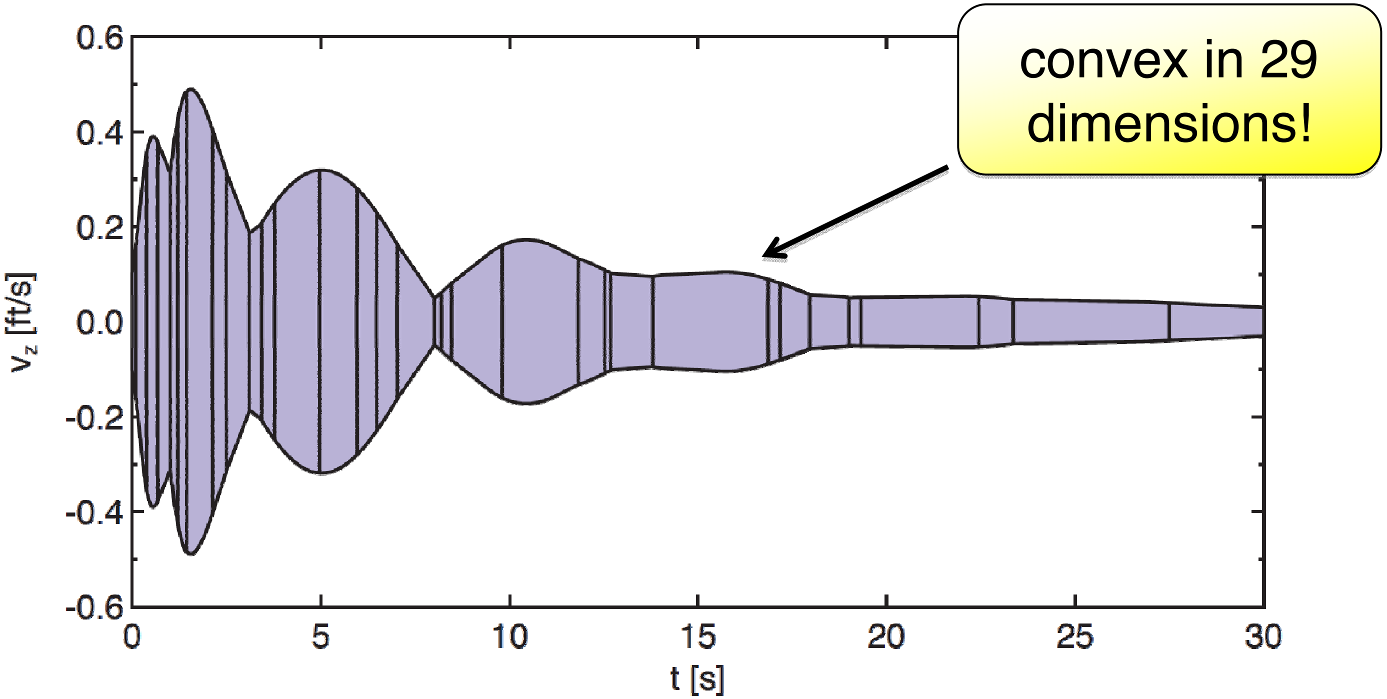



**CAV'11**: 1440 sets in 5.9s

1440 time steps

# Example: Helicopter

- **28 state variables + clock**

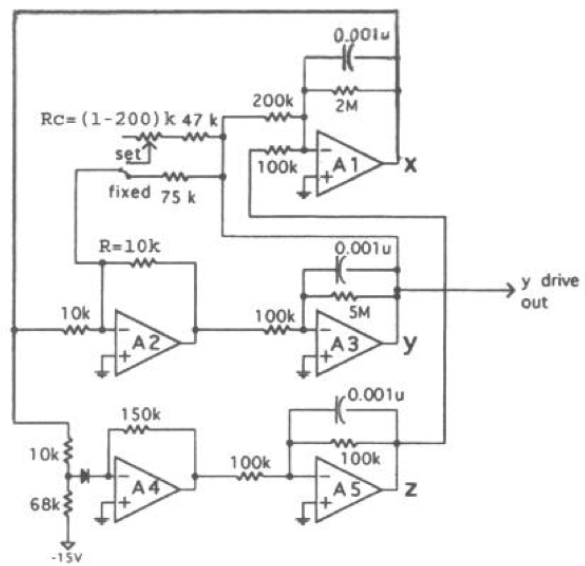


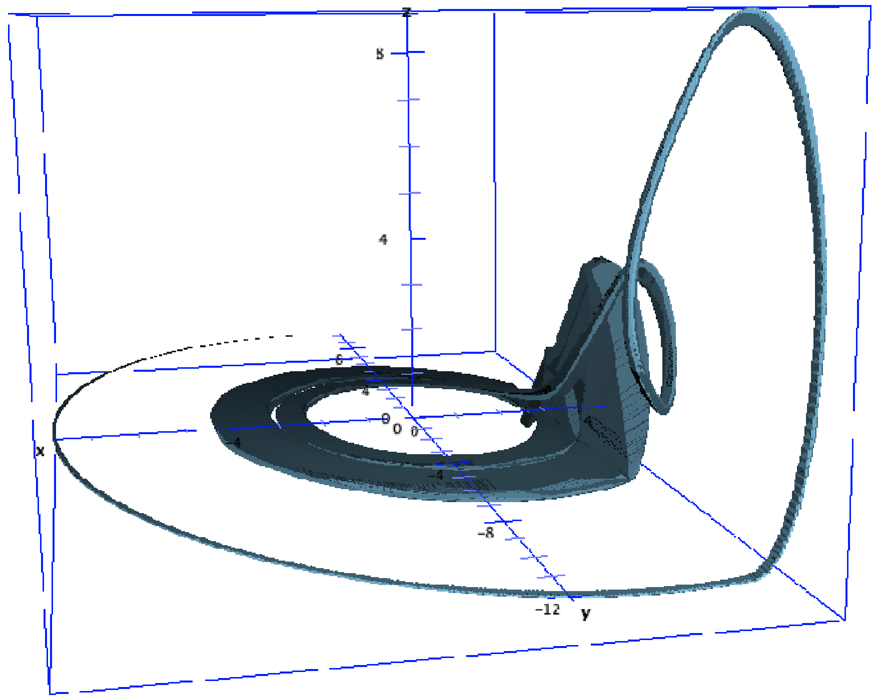

**HSCC'13**: **32** sets in 15.2s (4.8s clustering)

2 -- 3300 time steps, median 360

# Example: Chaotic Circuit

- **piecewise linear Rössler-like circuit**
  Pisarchik, Jaimes-Reátegui. ICCSDS'05
- **added nondet. disturbances**
- **3 variables, hard!**

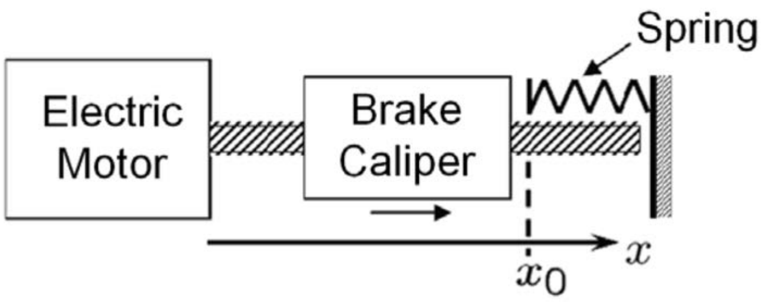


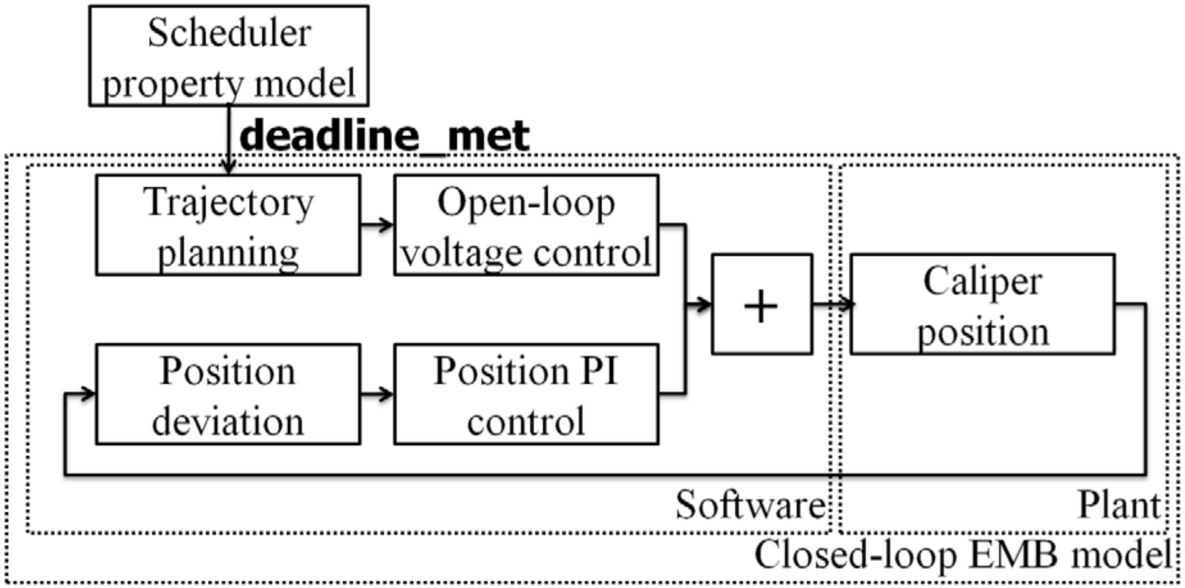



# Outline

- **Modeling with Hybrid Automata**
- **Reachability versus Simulation**
- **Reachability Algorithms**
  - piecewise constant dynamics
  - piecewise affine dynamics
- **Case Study: Controller Implementation**
- **SpaceEx Tool Platform**
- **Bibliography**

# Case Study: Electro-Mechanical Brake



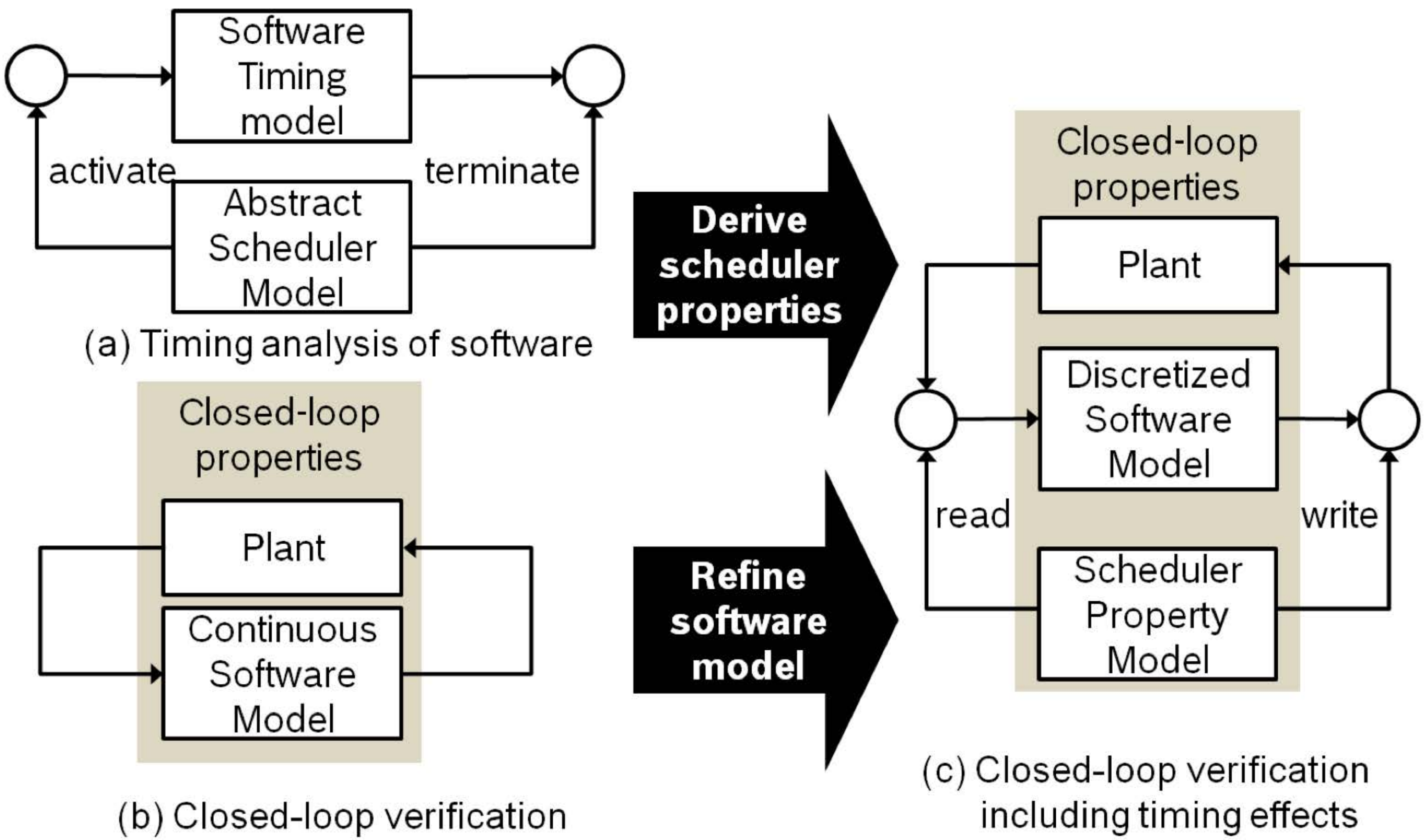




Frehse, Hamann, Quinton, Woehrle. Formal analysis of timing effects on closed-loop properties of control software. RTSS'14

# Case Study: Electro-Mechanical Brake

- **Controller Implementation**
  - discrete time
  - fixed-point arithmetic
  - multi-tasking processor: **scheduling with uncertain frequency**
  - worst-case analysis too conservative

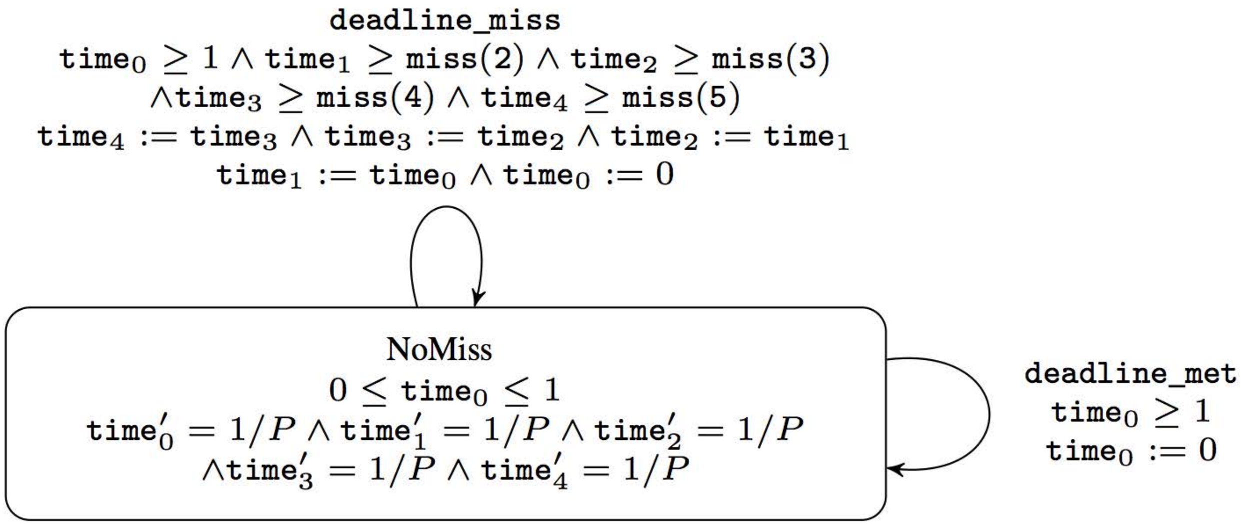
# Case Study: Electro-Mechanical Brake



# Case Study: Electro-Mechanical Brake

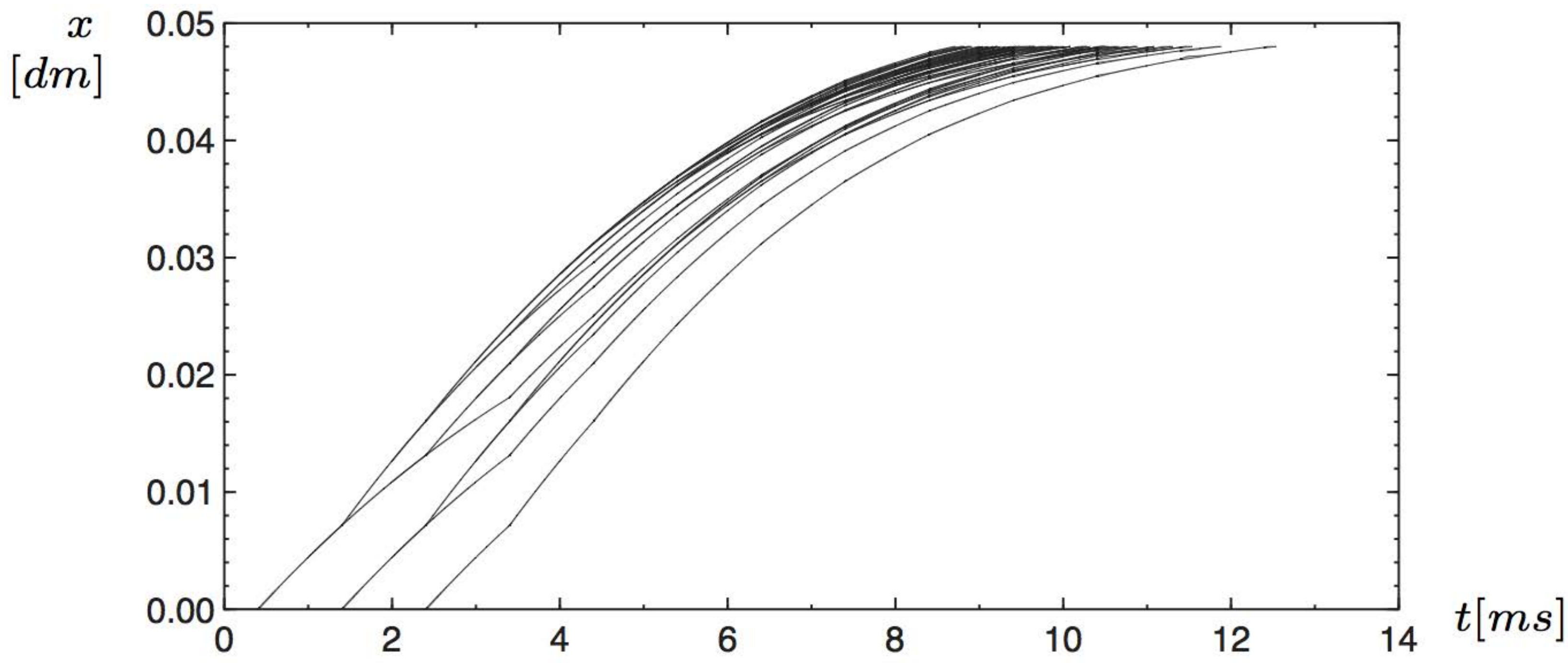
- **Typical Worst-Case Execution Time**
  - limit missed schedules per time interval

| # deadline misses | consecutive executions |
|---|---|
| 2 | 2 |
| 3 | 18 |
| 4 | 20 |
| 5 | 56 |



# Case Study: Electro-Mechanical Brake

caliper position

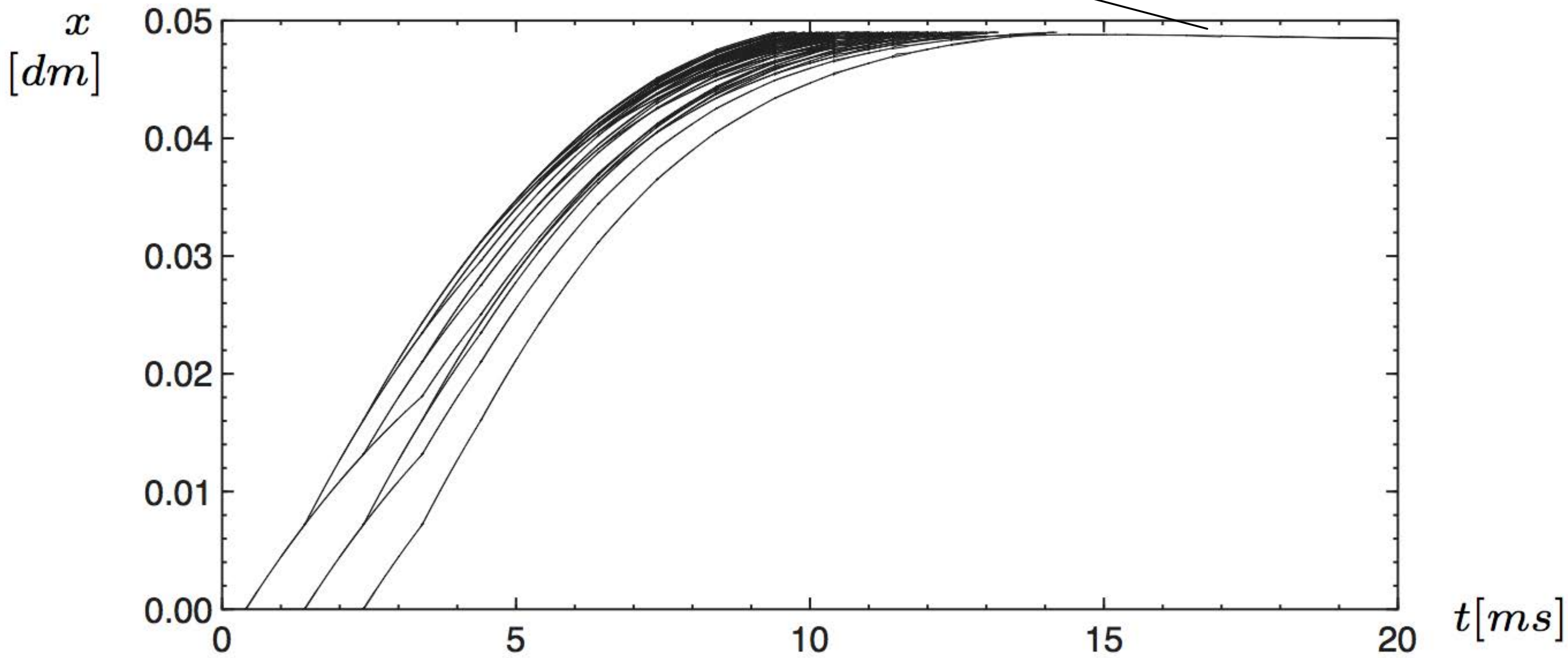



time

# Case Study: Electro-Mechanical Brake

caliper position

**only failure – hard to detect**



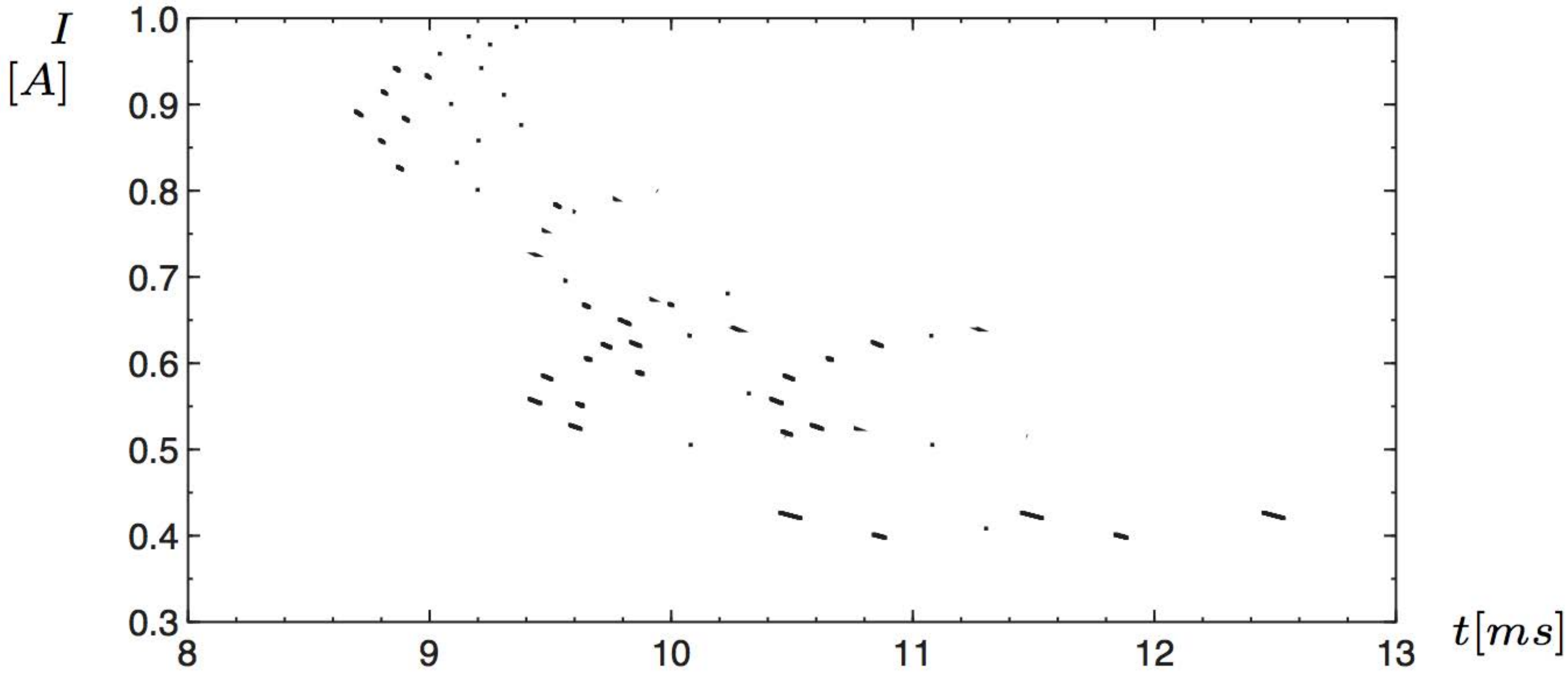

artificial failure case

(inconsistent with classical theory)

time

# Case Study: Electro-Mechanical Brake

current



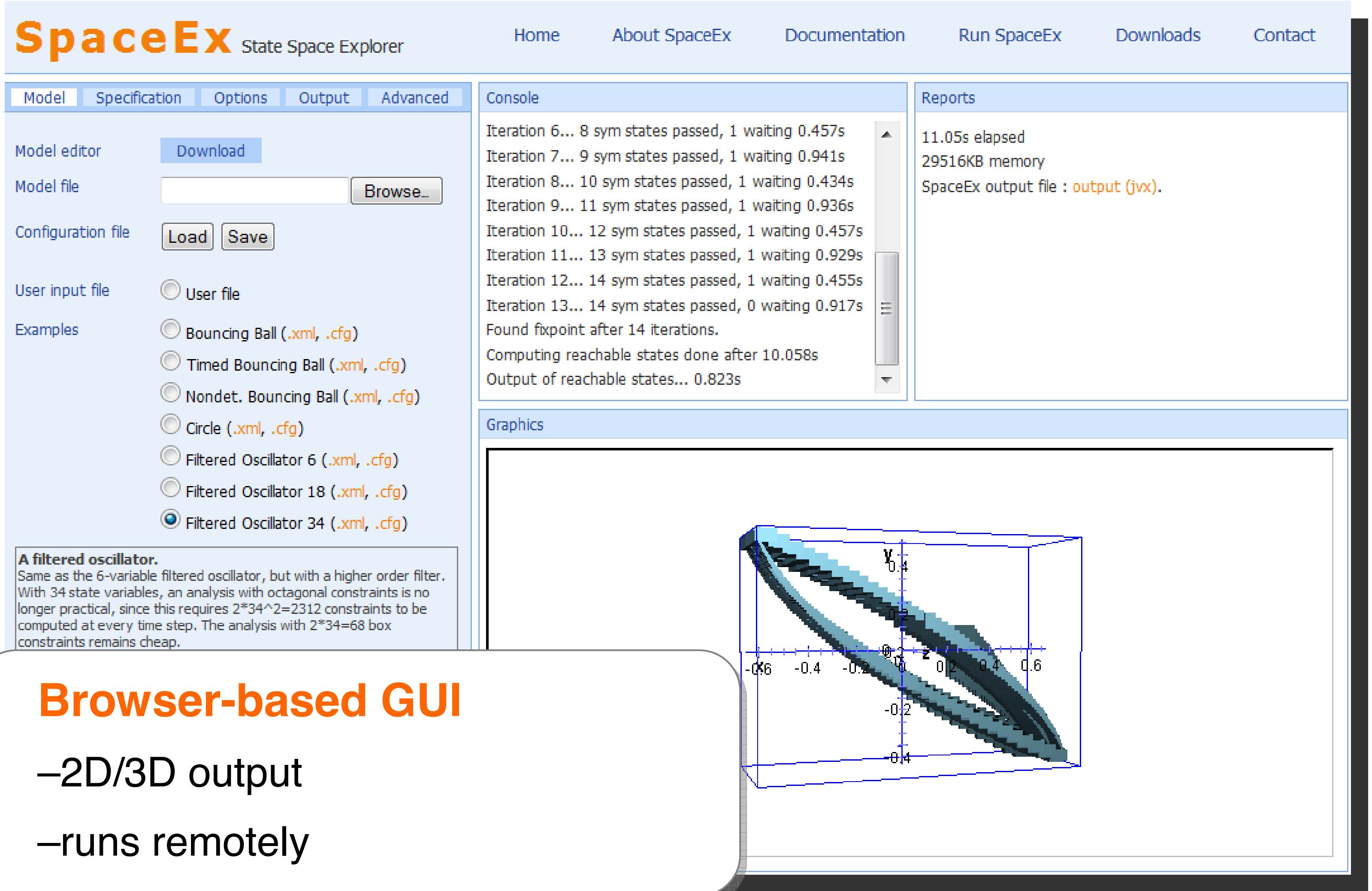

time

**physical properties:** maximum impulse on contact (measured via current)

# Outline

- **Modeling with Hybrid Automata**
- **Reachability versus Simulation**
- **Reachability Algorithms**
  - piecewise constant dynamics
  - piecewise affine dynamics
- **Case Study: Controller Implementation**
- **SpaceEx Tool Platform**
- **Bibliography**

# SpaceEx Verification Platform

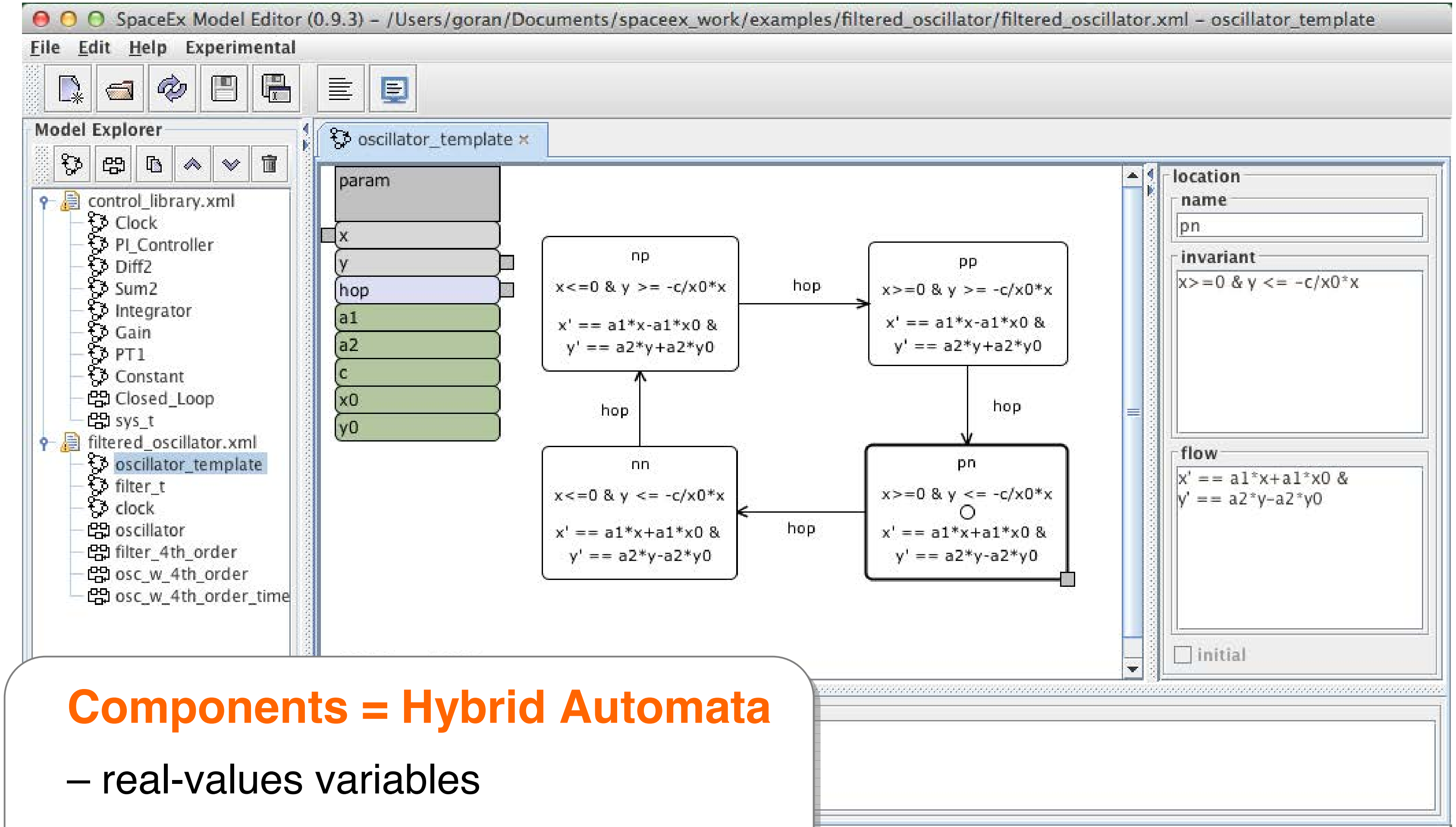



**Browser-based GUI**

–2D/3D output

–runs remotely

# SpaceEx Model Editor

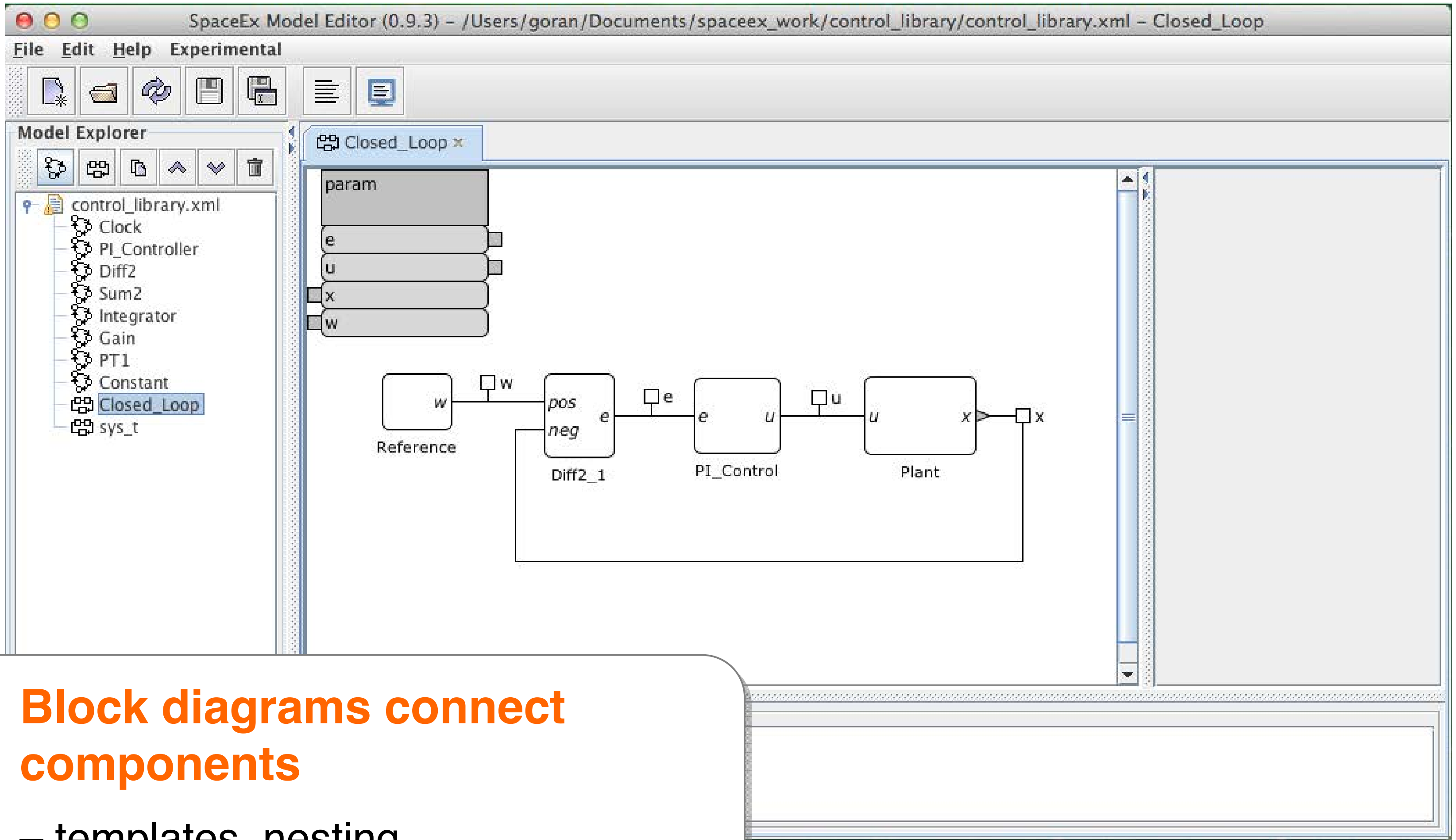



**Components = Hybrid Automata**

– real-values variables

– ODE, linear DAE

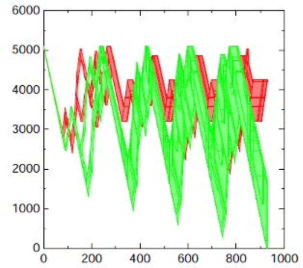
# SpaceEx Model Editor




**Block diagrams connect components**

– templates, nesting

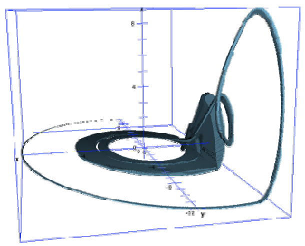
# SpaceEx Reachability Algorithms




**PHAVer**

–constant dynamics (LHA)

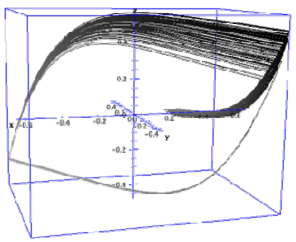
–formally sound and exact



**Support Function Algo**

–many continuous variables

–low discrete complexity



**Simulation**

–nonlinear dynamics

–based on CVODE

**spaceex.imag.fr**

# Outline

- **Modeling with Hybrid Automata**
- **Reachability versus Simulation**
- **Reachability Algorithms**
  - piecewise constant dynamics
  - piecewise affine dynamics
- **Case Study: Controller Implementation**
- **SpaceEx Tool Platform**
- **Bibliography**

# Bibliography


- **Hybrid Systems Theory**
  - Alur, Courcoubetis, Halbwachs, Henzinger, Ho, Nicollin, Olivero, Sifakis, Yovine. The algorithmic analysis of hybrid systems. Theoretical Computer Science, 1995
  - Henzinger. The theory of hybrid automata. LICS'96
- **Linear Hybrid Automata**
  - Henzinger, Ho, Wong-Toi, HyTech: The next generation. RTSS'95
  - Frehse. PHAVer: Algorithmic Verification of Hybrid Systems past HyTech. HSCC'05
  - Frehse. Tools for the verification of linear hybrid automata models. Handbook of Hybrid Systems Control. 2009.

# Bibliography


- **Affine Dynamics**
  - Asarin, Bournez, Dang, Maler. Approximate Reachability Analysis of Piecewise-Linear Dynamical Systems. HSCC'00
  - Girard, Le Guernic, Maler. Efficient computation of reachable sets of linear time-invariant systems with inputs. HSCC'06
- **Support Functions**
  - Le Guernic, Girard. Reachability analysis of hybrid systems using support functions. CAV'09
  - Frehse, Le Guernic, Donzé, Ray, Lebeltel, Ripado, Girard, Dang, Maler. SpaceEx: Scalable verification of hybrid systems. CAV'11.
  - Frehse, Kateja, Le Guernic. Flowpipe approximation and clustering in space-time. HSCC'13

# Bibliography


- **Ellipsoids**
  - Kurzhanskiy, Varaiya. Ellipsoidal toolbox (et). CDC, 2006.
  - Kurzhanskiy, Varaiya. Ellipsoidal techniques for reachability analysis of discrete-time linear systems. IEEE Transactions on Automatic Control, 2007.
- **Zonotopes**
  - Antoine Girard. Reachability of uncertain linear systems using zonotopes. HSCC'05
  - M. Althoff and B. Krogh. Reachability analysis of nonlinear differential-algebraic systems. IEEE Transactions on Automatic Control, 2013

# Verification Tools for Hybrid Systems

- **HyTech: LHA**
  - http://embedded.eecs.berkeley.edu/research/hytech/
- **Matisse Toolbox: zonotopes**
  - http://www.seas.upenn.edu/~agirard/Software/MATISSE/
- **Cora Toolbox: zonotopes, nonlinear systems**
  - http://www6.in.tum.de/Main/SoftwareCORA
- **HSOLVER: nonlinear systems**
  - http://hsolver.sourceforge.net/
- **Flow*: nonlinear systems**
  - http://systems.cs.colorado.edu/research/cyberphysical/taylormodels/
- **and more…: http://wiki.grasp.upenn.edu/hst/**

# Conclusions

- **Reachability in continuous time is hard**
  - even for simple dynamics
- **Handle affine systems with 100+ variables**
  - exploiting properties of affine dynamics
  - lazy set representations (support functions)

- **Further Work...**
  - abstraction refinement
  - extension to nonlinear dynamics