Verified Switched Control System Design using Real-Time Hybrid Systems Reachability

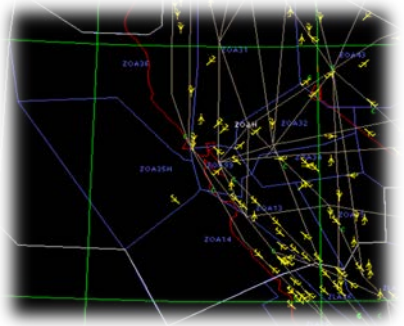
Stanley Bak, Taylor Johnson, Marco Caccamo, Lui Sha
Air Force Research Lab – Information Directorate – Rome, NY

Cyber-Physical Systems

- Include computational components interacting with the physical-world
- **Mistakes can have real-world consequences!**
- Ideally we would verify the system, but it may be too complicated for direct verification

Fault-Tolerant
Power Distribution

Autonomous Cars



Air Traffic Control

Outline

- Run-Time Assurance (RTA) Design

- RTA using Real-Time Reachability

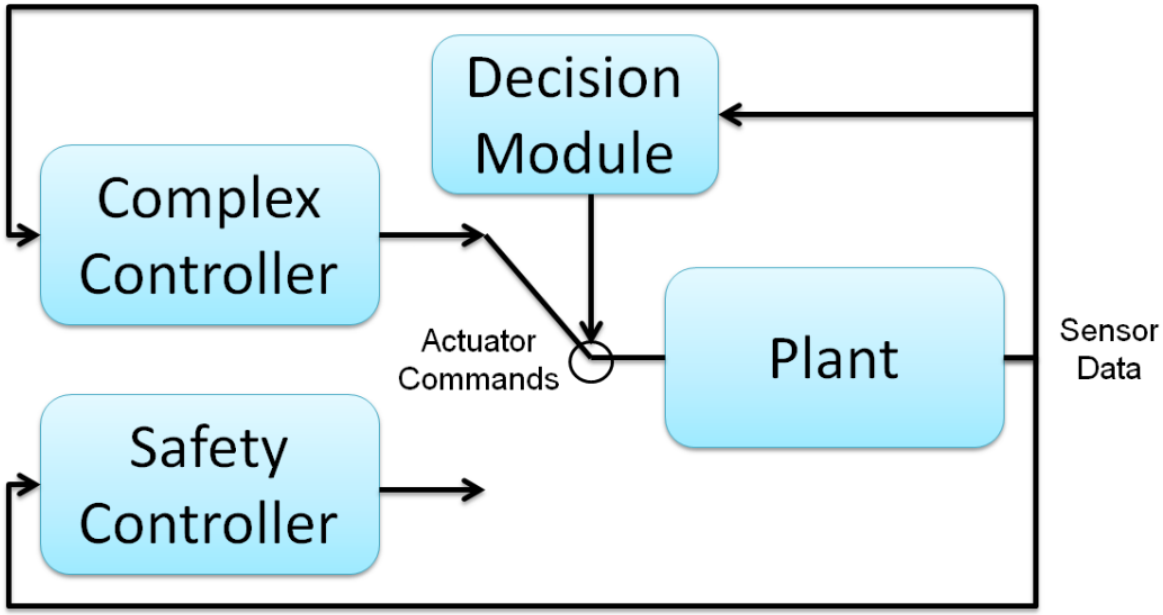
Outline

- **Run-Time Assurance (RTA) Design**

- RTA using Real-Time Reachability

Run-time Assurance (RTA) Design

- Sandbox untrusted controllers
- Lots of variants
- Key challenge is decision module



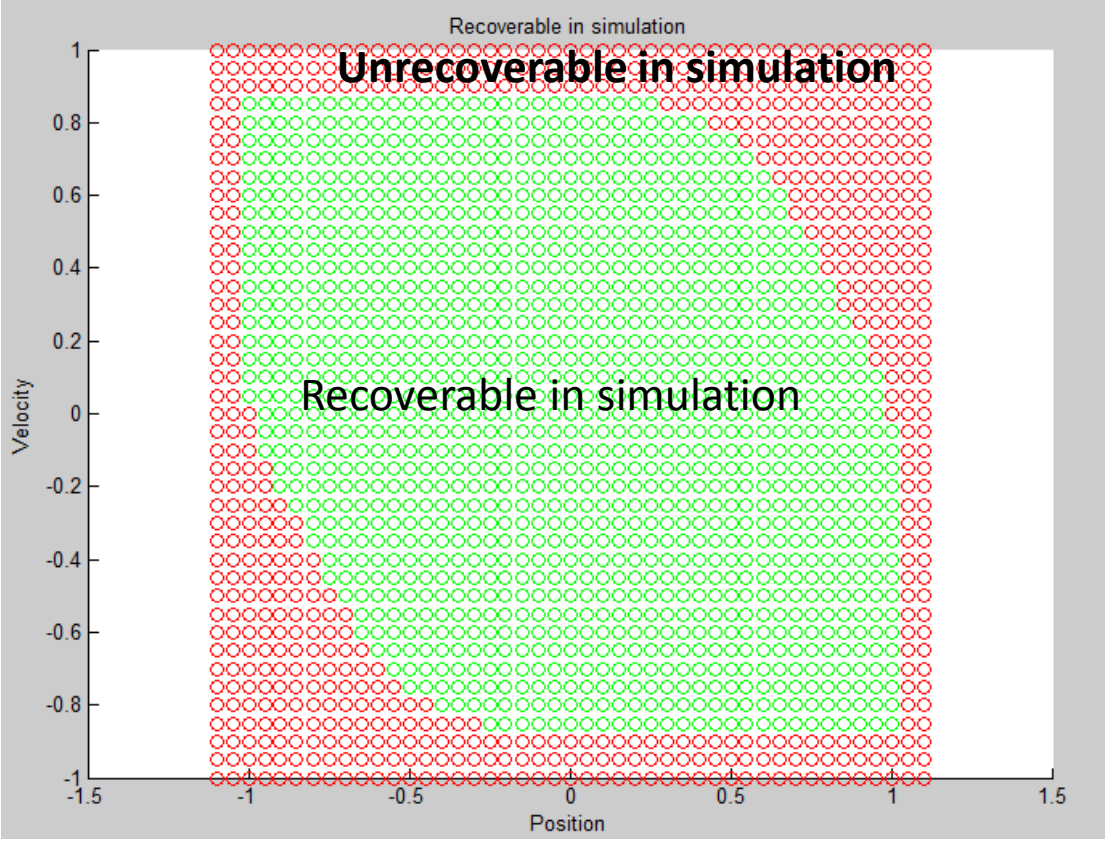
Run-time Assurance (RTA) Design

- Safe design is easy!

- The challenge is **conservatism**. The ‘best’ switching logic:
  - Predicts next state using the current command
  - Checks if the safety controller recovers

Offline Simulation Design

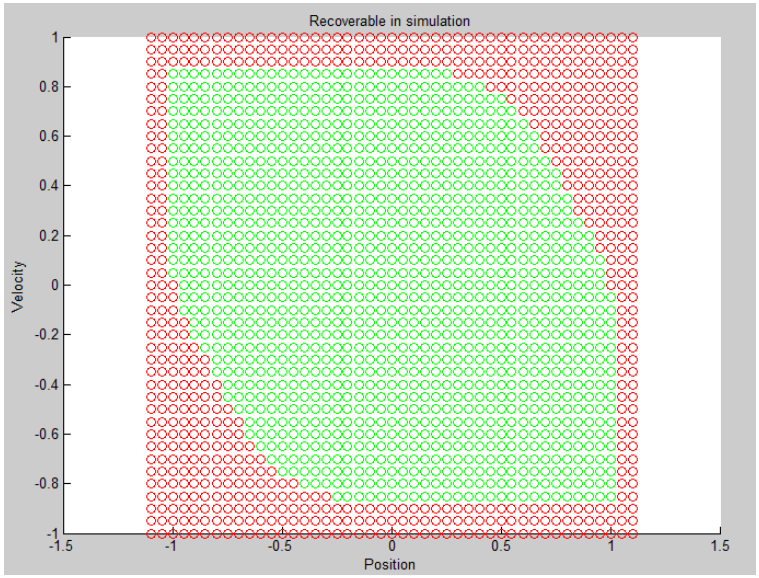
- Using simulations, we can grid the state space, then check which states are recoverable*



* M. Aiello, J. Berryman, J. Grohs, and J. Schierman, "Run-time assurance for advanced flight-critical control systems," in Proceedings of the American Institute of Aeronautics and Astronautics Guidance, Navigation, and Control Conference, ser. AIAA '10, 2010.

Challenges with Simulation Design

- When do we stop the simulation?
- What if the real state is between simulation points?
- How accurate are the simulations?
- Most problematic: How well does this scale?



Scaling with Simulation Design

Most problematic: How does this scale?

- 100 partitions per dimension,
  11 dimensions = 100^11 =10^22 points,
  1 μs per simulation = **317 million years**
- Large online storage required
  - Lookup table?
  - Linear bounds*?

Continental plate configuration 317.1 million years ago

* S. Bak, "Industrial application of the System-Level Simplex Architecture for real-time embedded system safety," Master's Thesis, University of Illinois at Urbana-Champaign, 2009.

Simulation Scalability

How many partitions per dimension if we want the runtime to be:

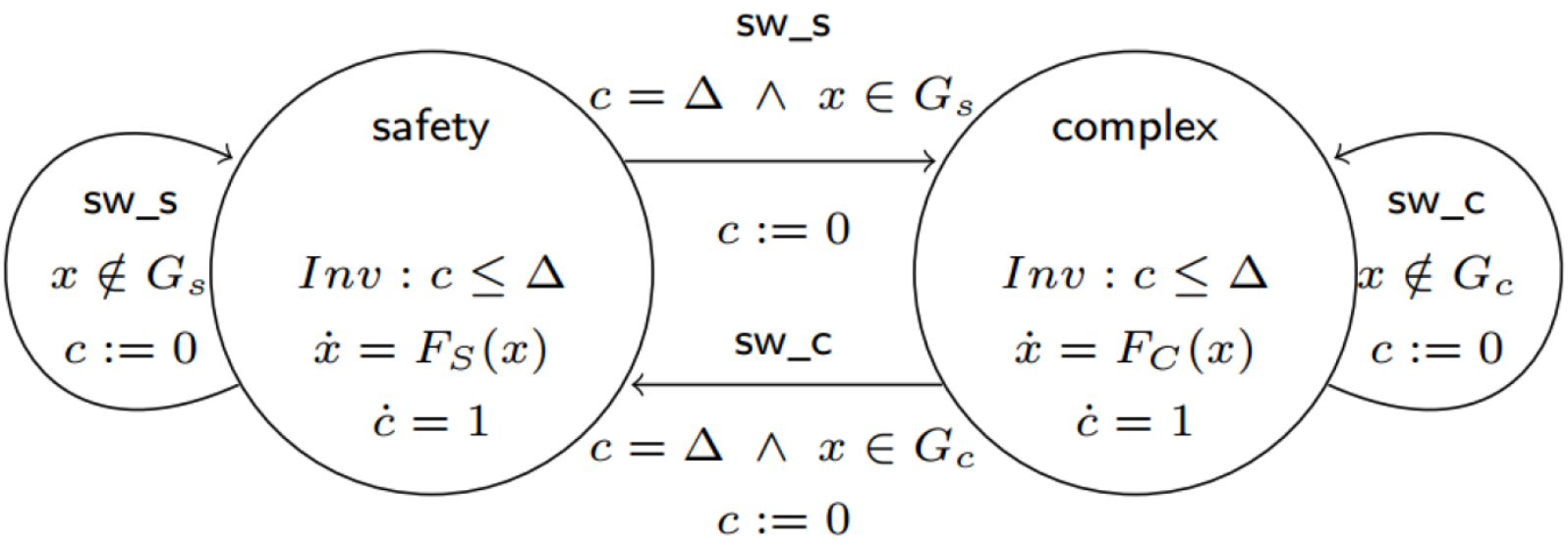
1 hour – ~7 partitions (~51 degrees)

1 day – ~ 10 partitions (~36 degrees)

1 year - ~ 17 partitions (~20 degrees)

Verified Design based on Reachability

- Instead of simulation, we can use more formal reasoning based on hybrid-systems reachability computation
- This reasons about sets of states, accounting for method inaccuracies by over-approximation



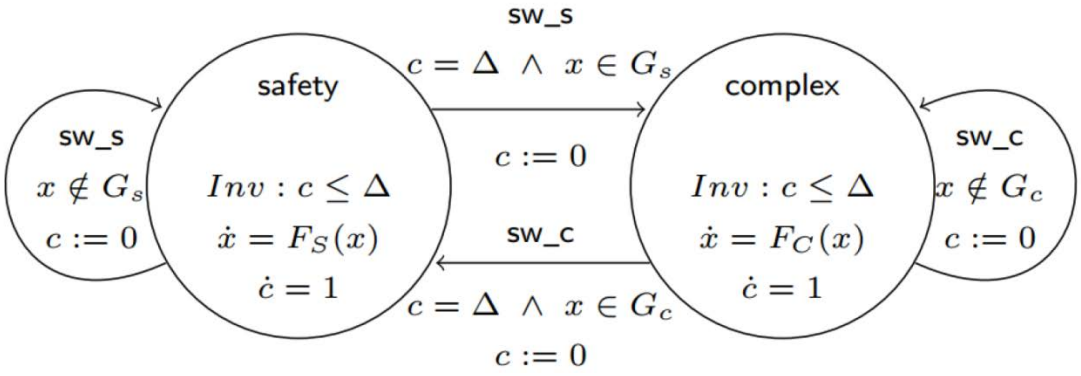
Reachability for RTA

- The ‘best’ switching logic can be defined in terms of reachability:

$$\textsc{Reach}(\textsc{Reach}_{\delta}(x, \text{CC}), \text{SC}) \cap U = \emptyset$$

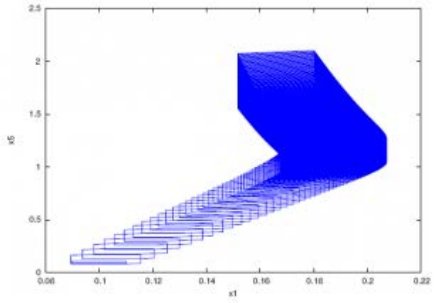
-or- $x \notin \textsc{BackReach}^{*}_{\leq \delta}(\textsc{BackReach}^{*}(U, \text{SC}), \text{CC}')$

- Drawbacks:
  - Achievable Accuracy
  - Online Representation

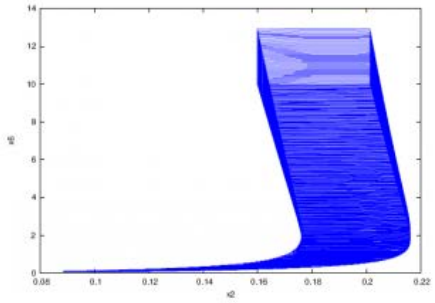


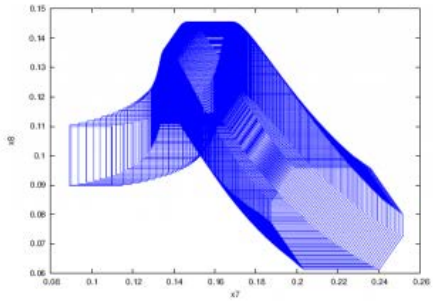
Representation Drawbacks

- Flow* is a tool which computes reachability for systems with nonlinear dynamics
  - Uses Taylor Models for representation



- Example: 9 dimensional biological model*
  - Order: 5, Step size: 0.001, Steps: 10
  - Output: 3 MB, ~300 KB per step





* "Constructing Flowpipes for Continuous and Hybrid Systems: Case-Studies"
http://systems.cs.colorado.edu/research/cyberphysical/taylormodels/casestudies/

Verified Design based on LMI

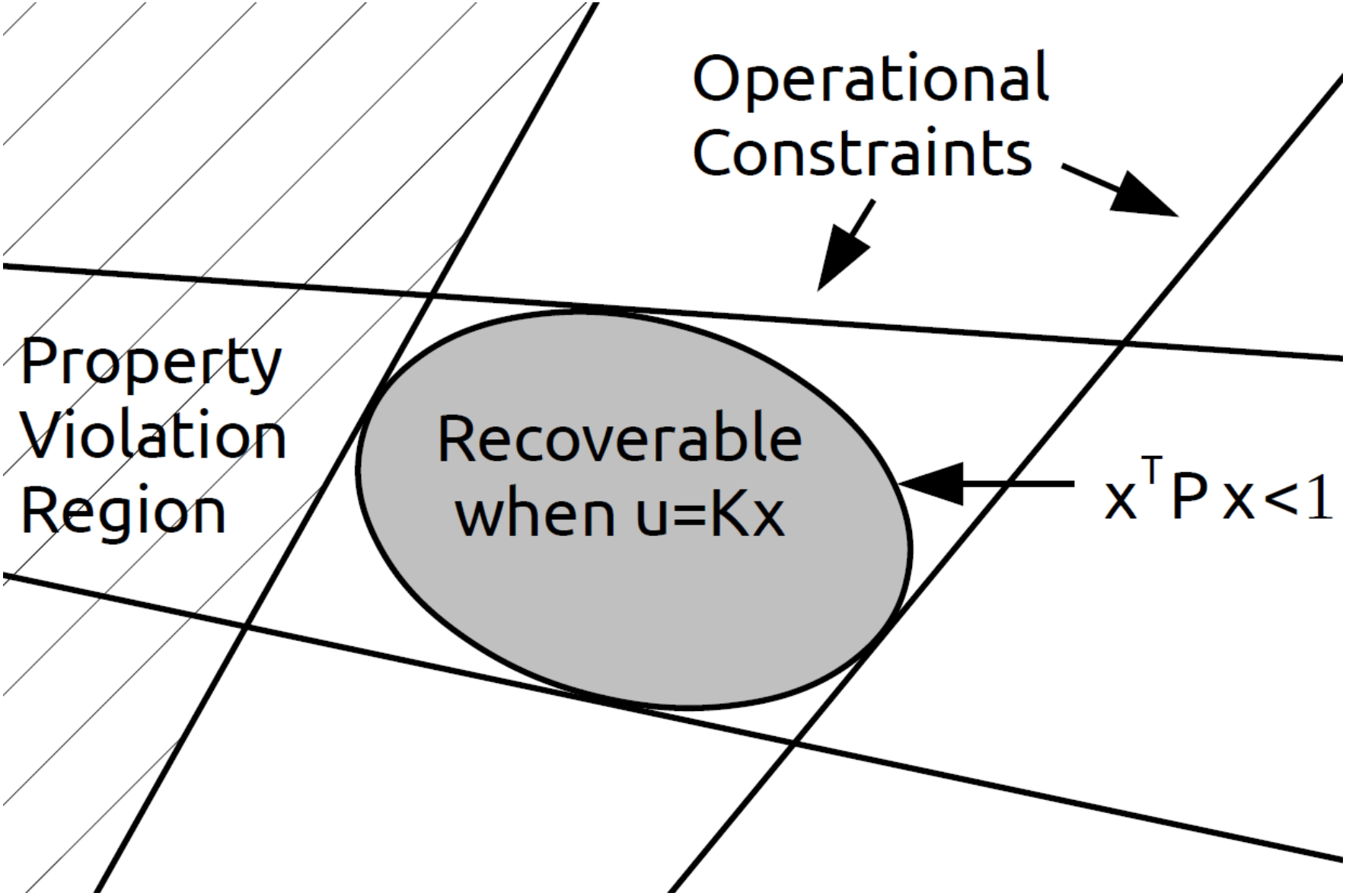
- For linear (and linearized) systems, you can find the largest ellipsoid inside the recoverable region by solving a linear matrix inequality (LMI)*

Linear Time-Invariant Control System: $x' = Ax + Bu$

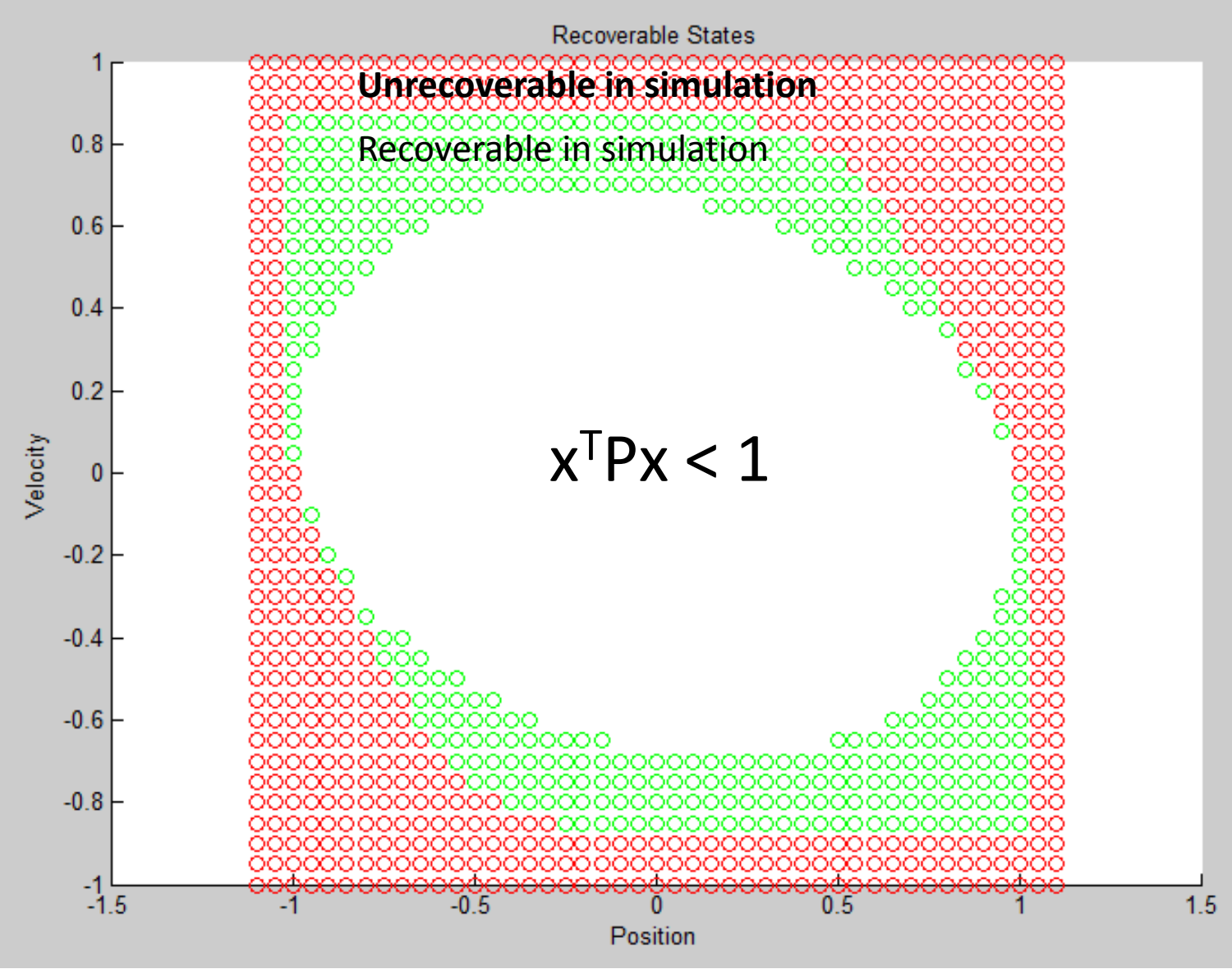
- Input: Matrices A, B, linear system constraints
- Output: Gain matrix K, Potential matrix P,
  where if you use $u=Kx$ then x^TPx is decreasing and all constraints are satisfied if $x^TPx < 1$

* D. Seto and L. Sha, "A case study on analytical analysis of the inverted pendulum real-time control system," CMU/ SEI, Tech. Rep., 1999.

LMI-design for RTA



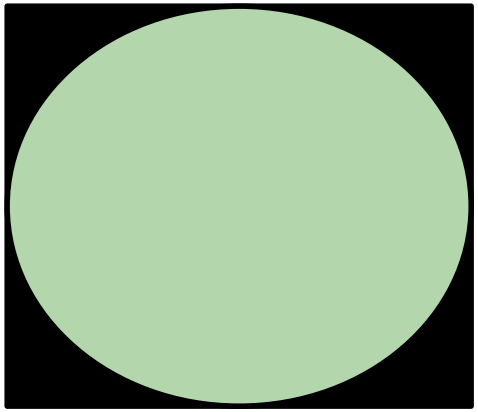
LMI-design for RTA (2)

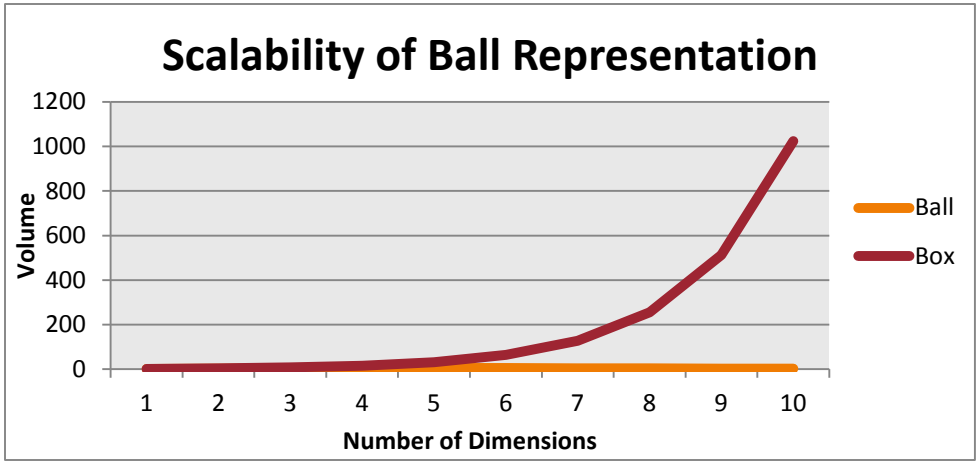


LMI-drawbacks

1. Ellipsoid guarantees input will not saturate, which is pessimistic

2. Ellipsoid may trim out recoverable states because of its shape restriction





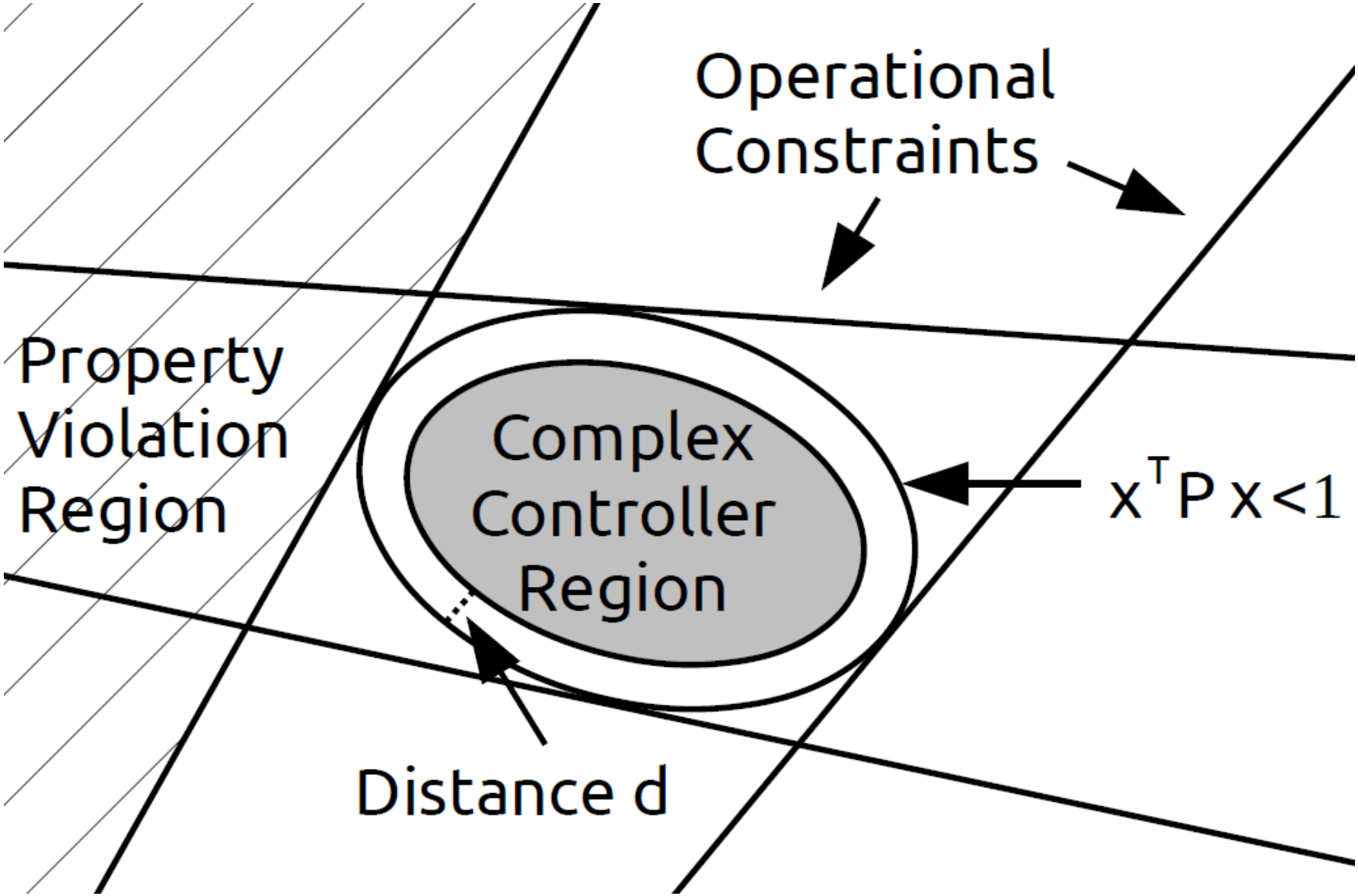
# Dims	1	2	3	4	5	6	7	8	9	10
Ball Volume	2	3.141	4.189	4.935	5.264	5.168	4.725	4.059	3.299	2.55
Box Volume	2	4	8	16	32	64	128	256	512	1024

LMI-drawbacks (2)

- From before: The ‘best’ switching logic:
  - **Predicts next state using the current command**
  - Checks if the safety controller recovers

- For the offline LMI approach, you consider all possible commands and how much state space can be covered in one control iteration, and then create a ‘buffer’

LMI Image


Operational
Constraints
Property
Violation
Region
Complex
Controller
Region
$x^T P x < 1$
Distance d

Outline

- Run-Time Assurance (RTA) Design

- **RTA using Real-Time Reachability**

RTA using Real-Time Reachability

- Don’t determine the switching set offline, do it online!
  - No large enumeration
  - Not limited to ellipsoid shape
  - No complex state representation

- How do we do it?
  - Use aspects of both LMI-based and reachability-based Simplex design

Unified Design

- We use the forward-time definition of a switching set

$$\textsc{Reach}(\textsc{Reach}_{\delta}(x, \textsc{cc}), \textsc{sc}) \cap U = \emptyset$$

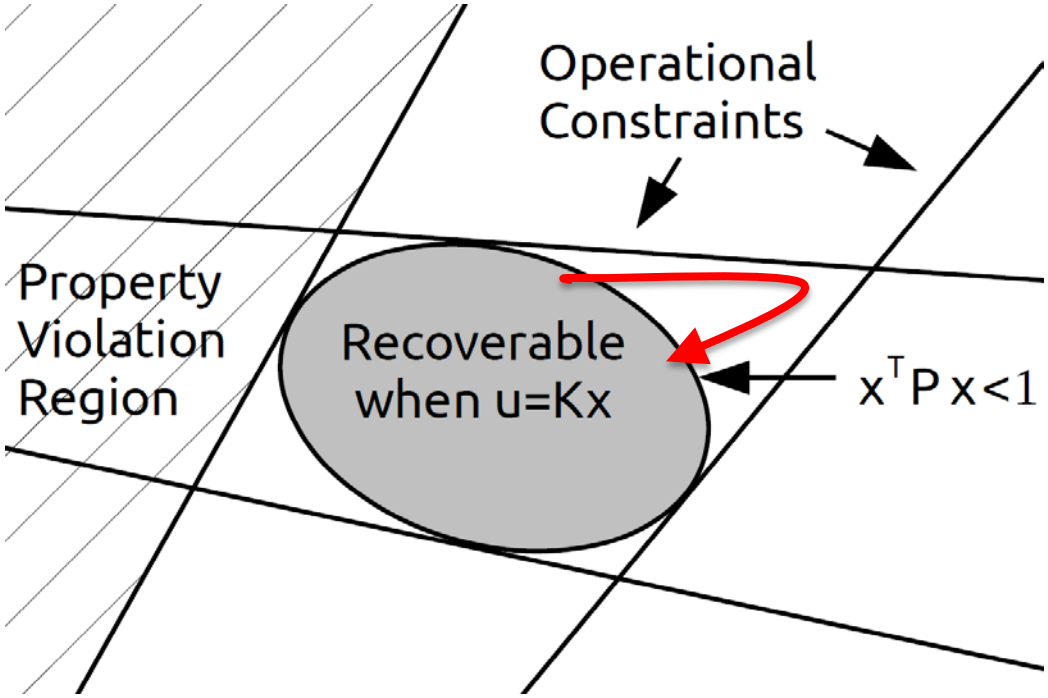
- However, we don't need infinite time reachability; we only need to get back into the LMI ellipsoid because

$$\textsc{Reach}(\mathcal{R}, \textsc{sc}) \cap U = \emptyset$$

$$\mathcal{R} = \{x | x^T P x < 1\}$$

Unified Design (2)

- **Key Idea**: allow the system to leave the safe ellipsoid, as long as we can guarantee (1) no constraints are violated when this happens, and (2) the state is guaranteed to go back into the ellipsoid



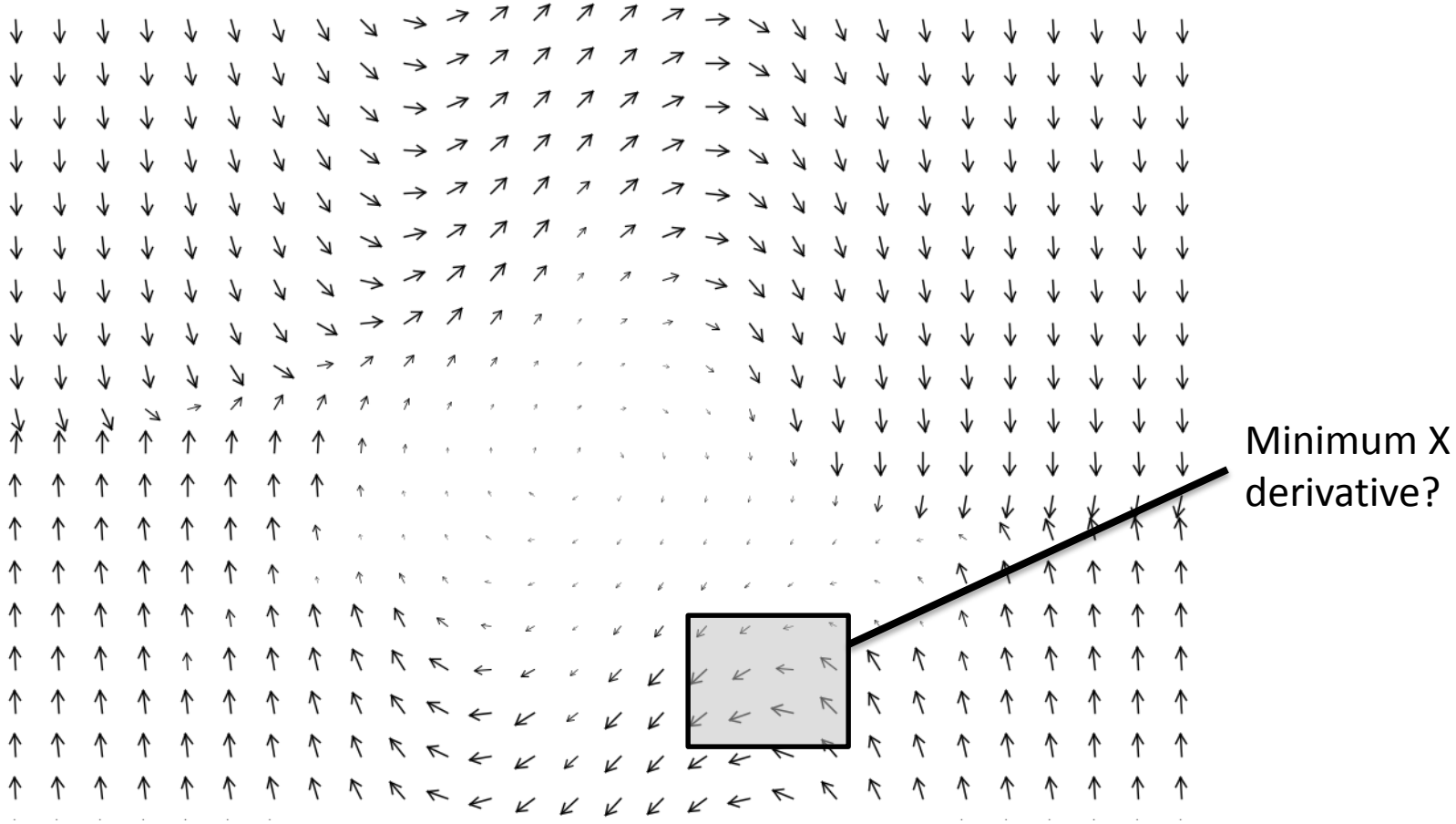
Unified Design (3)

- This requires reachability at runtime
  - Tools aren't meant for this…

- Let's make one!
  - Based on mixed-face lifting
  - Quick computation is more important than long-term error control
  - Assumes piecewise dynamics, with bounded derivatives, and a user-provided **`DerivativeBounds`** function

DerivativeBounds Function

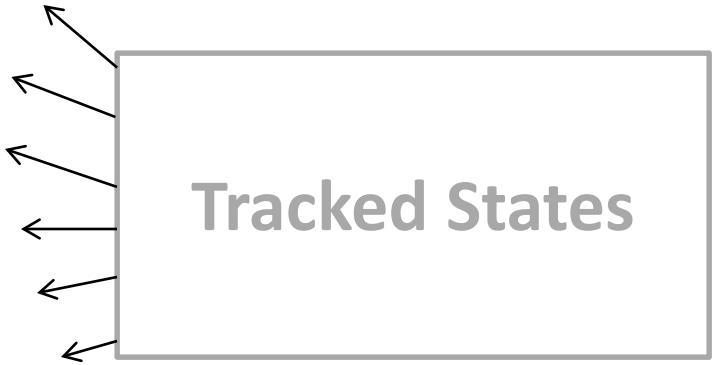
- For our algorithm, the user must provide a function that bounds the derivative for each direction in an arbitrary box



Real-Time Reachability Algorithm

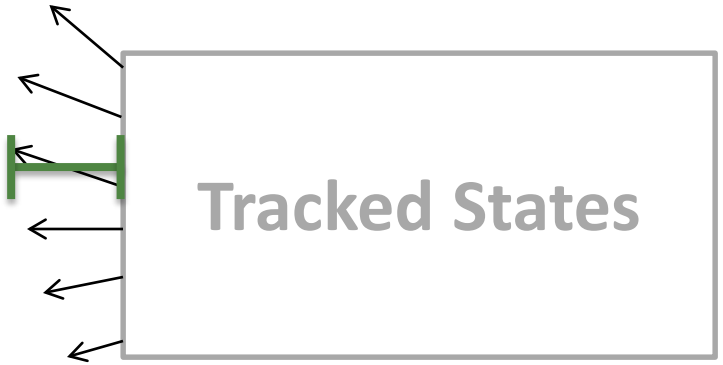
Tracked States

Real-Time Reachability Algorithm



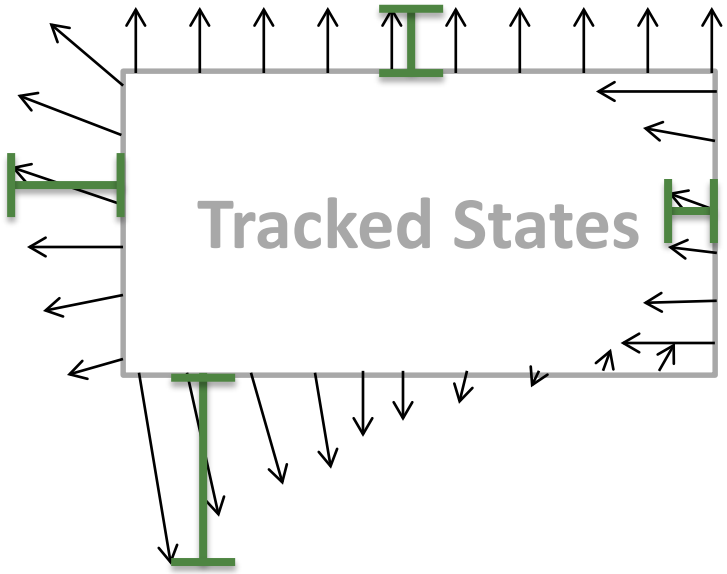
Real-Time Reachability Algorithm

Maximum advancement of face after `desiredStep` time



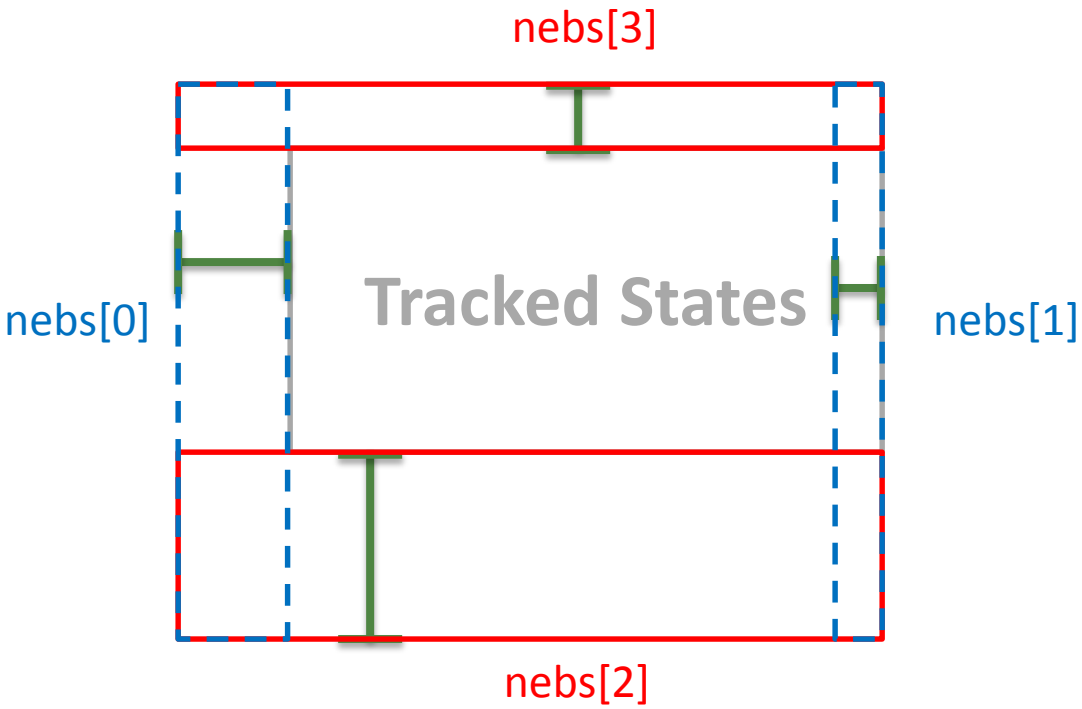
Real-Time Reachability Algorithm

Maximum advancement of each face after `desiredStep` time



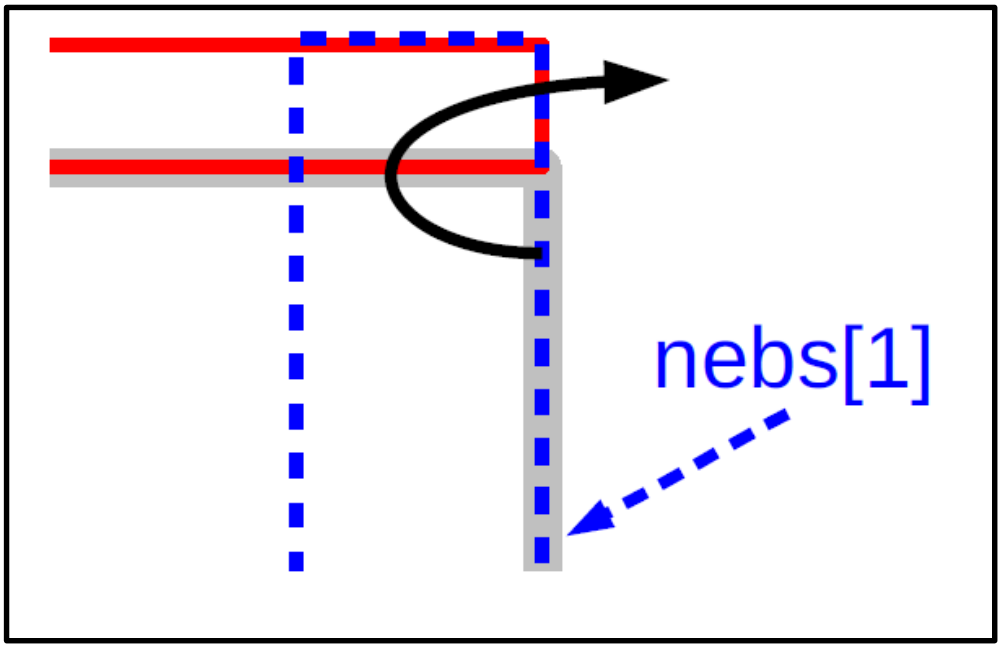
Real-Time Reachability Algorithm

Construct neighborhood for each face based on the widths



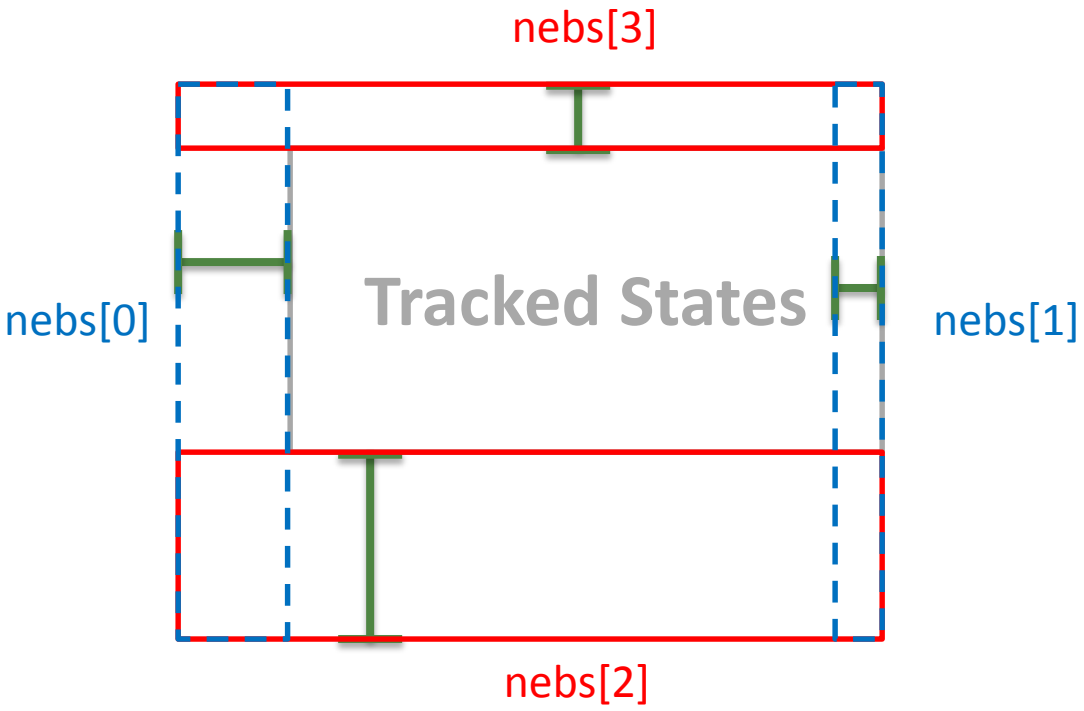
Real-Time Reachability Algorithm

- Next step: resample derivatives within neighborhoods
  - Reconstruct if either (1) neighborhood flips from inwards-facing to outwards-facing, or (2) estimated width doubles in size



Real-Time Reachability Algorithm

Compute time until one of the faces can leave its neighborhood



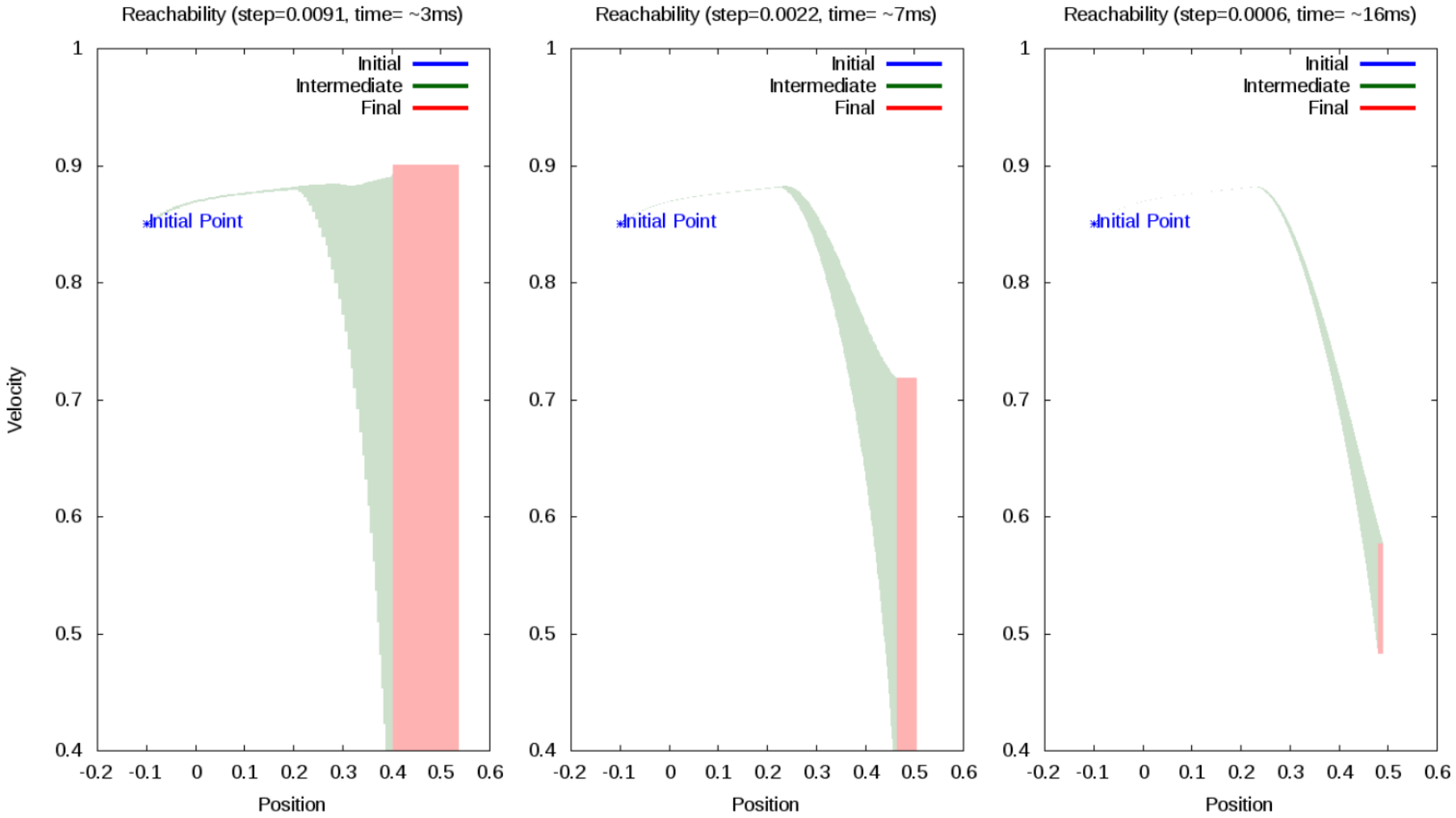
Real-Time Reachability

Advance time and repeat

Reachability Properties

- Construction time is bounded (finite derivative can only double a finite number of times)
- Time to advance per step is at least half of desired step size (number of steps is bounded)
- Typically, O(n) to advance time, but poor long-term error control due to wrapping effect
- After finishing the computation for the desired reach-time, cut desired step size in half and repeat until deadline exhausted

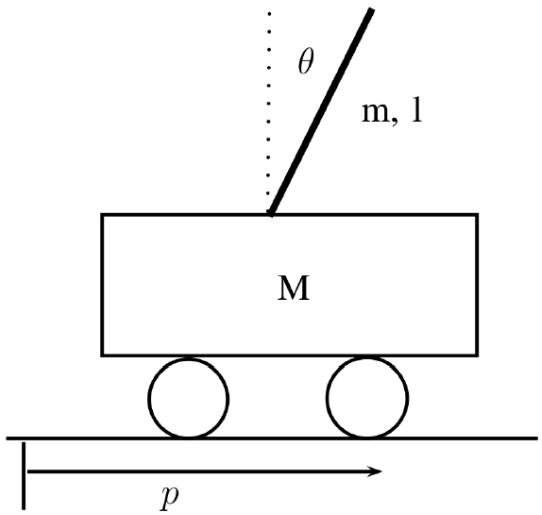
Iterative Improvement



Evaluation

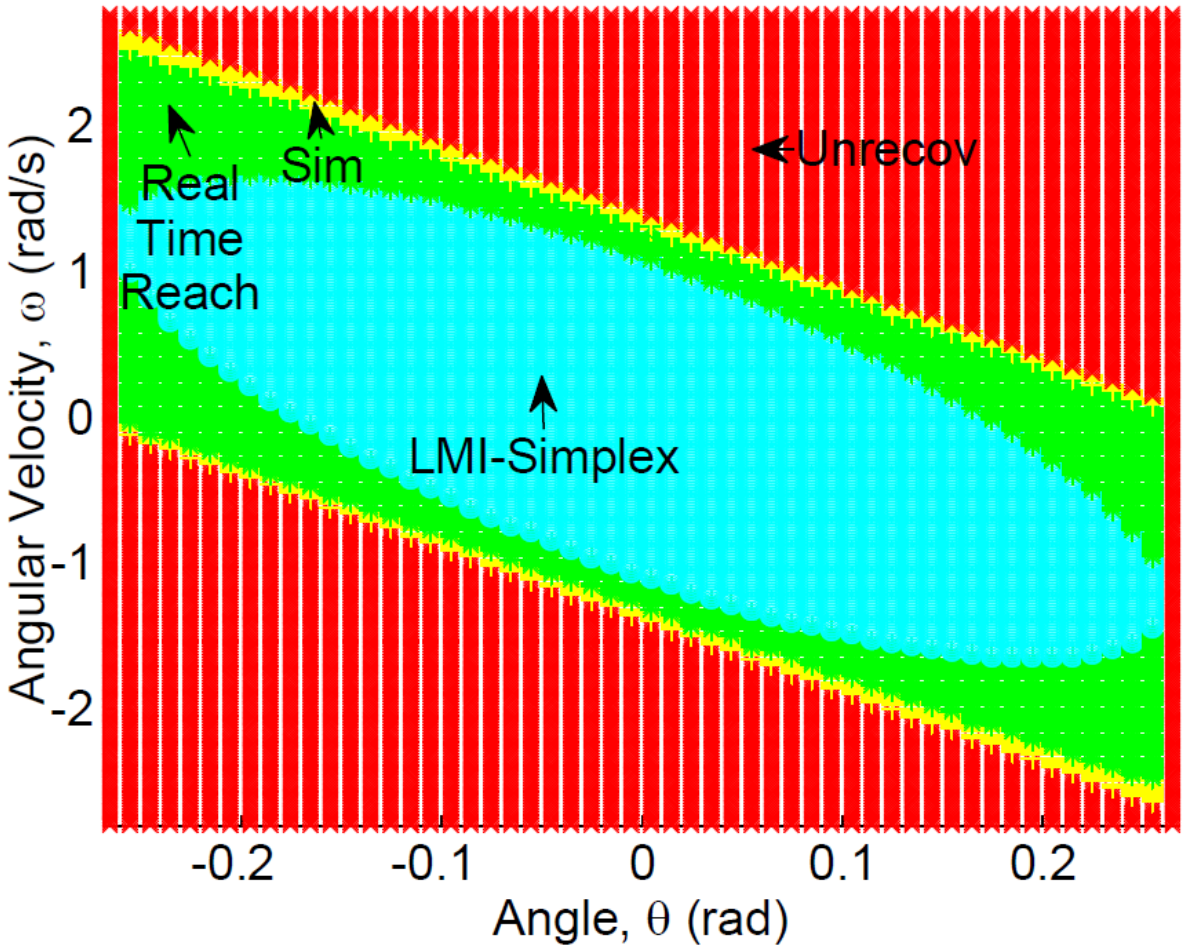
- How well does this compare to LMI-based reachability and simulation (optimal)?

- Use 4-D saturated inverted pendulum (same system as the one used for LMI-based design)



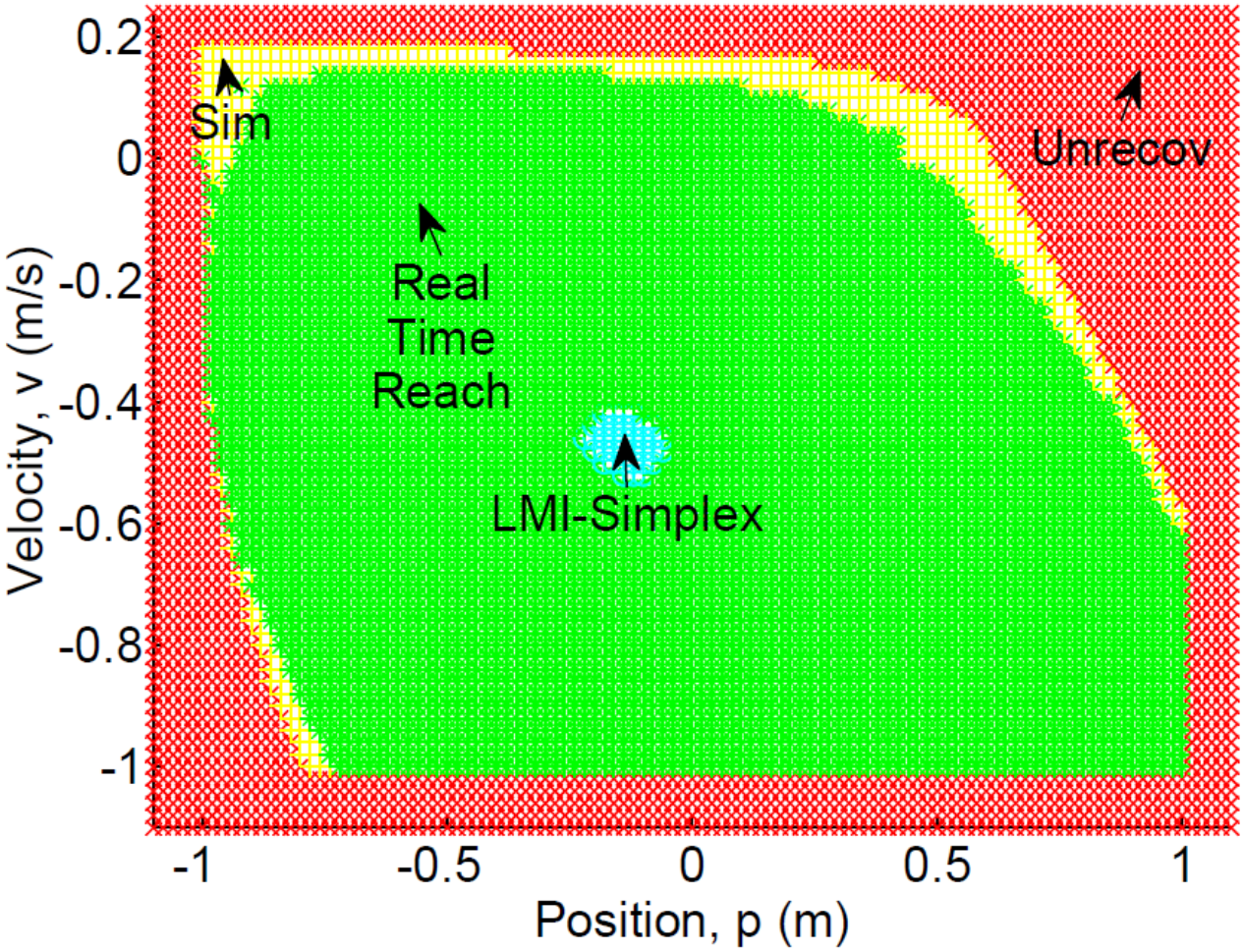
Evaluation (2)

- Projection where position = velocity = 0



Evaluation (3)

- Projection where θ=0.19, ω=0.18



Evaluation of Runtime

- Partition state space and count the points in each region
- “Best” improvement (based on simulations) is 247%

Runtime (ms)	LMI	RealTime	Sim	Unrecov	Improve
5	5473	6180	1376	37596	213%
20	5473	6948	608	37596	227%
40	5473	7059	497	37596	229%
50	5473	7108	448	37596	230%
75	5473	7133	423	37596	230%
100	5473	7216	340	37596	232%
200	5473	7286	270	37596	233%
500	5473	7338	218	37596	234%
1000	5473	7382	174	37596	235%

Conclusion

- **“A Unified Run-Time Assurance Scheme using Real-Time Reachability”**

- Run-Time Assurance Design
  - Simulation
  - Reachability
  - Linear Matrix Inequality (LMI)

- RTA using a Unified Approach
  - LMI ellipsoid + online reachability
  - Real-time reachability is feasible, even for small runtimes
  - Unified approach greatly expands the complex controller region (**227%** on the saturated inverted pendulum system)