Runtime Assurance: Problem Statement, State-of-Art Overview and Proposed Approach

Ratnesh Kumar
Iowa State Univ.

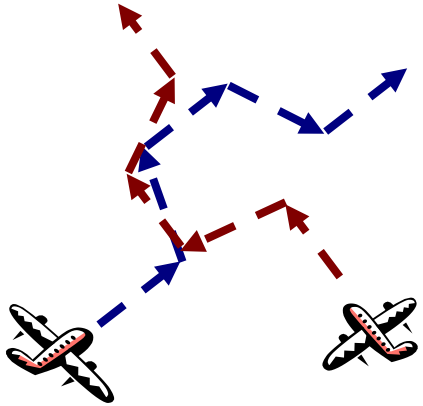
George Pappas
Univ. of Penn

Problem Statement

- Given:
  - Verified **B**aseline fail-safe control B
  - Unverified **A**dvanced control A
- Design:
  - Runtime strategy to allow use of advanced control to the extent possible while ensuring safety

Runtime Assurance: Boundary Methods Overview

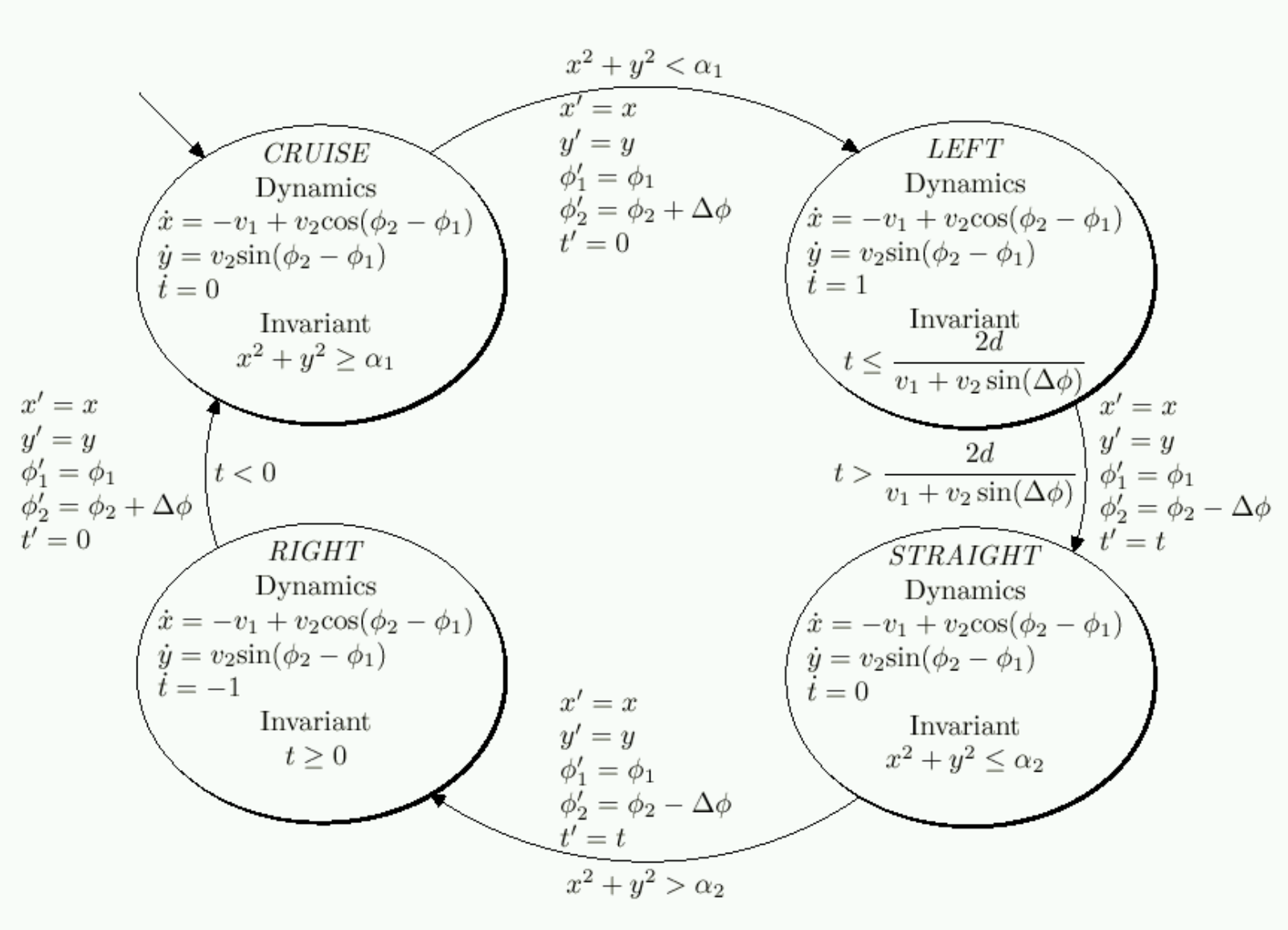
Illustrative Example: Conflict Resolution Protocol



1. Cruise until a_1 miles away
2. Change heading by $\Delta\Phi$
3. Maintain heading until lateral distance d
4. Change to original heading
5. Change heading by - $\Delta\Phi$
6. Maintain heading until lateral distance - d
7. Change to original heading

Is this protocol safe?

Conflict Resolution Maneuver



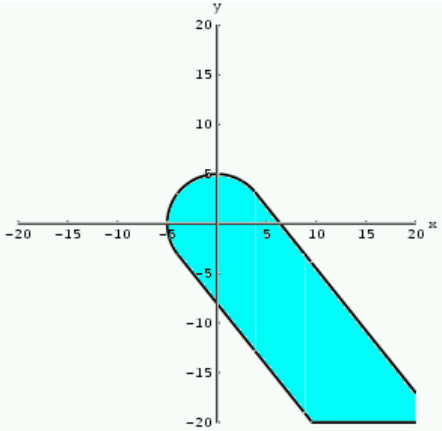
Computing Unsafe Sets

$$
\begin{aligned}
& v_1 = 4; v_2 = 5; \lambda = 0 \\
\texttt{unsafeCruise} &= \texttt{Resolve}\left[\exists t > 0 \wedge (x - v_1 t + \lambda v_2 t)^2 + (y + \sqrt{1-\lambda^2} v_2 t)^2 \leq 25\right] \\
&= \left(y < -\tfrac{20}{\sqrt{41}} \wedge -\sqrt{41} - \tfrac{4y}{5} \leq x \leq \sqrt{41} - \tfrac{4y}{5}\right) \vee \left(y = -\tfrac{20}{\sqrt{41}} \wedge -\sqrt{41} - \tfrac{4y}{5} < x \leq \sqrt{41} - \tfrac{4y}{5}\right) \vee \\
&\quad \left(y = \tfrac{20}{\sqrt{41}} \wedge -\sqrt{25 - y^2} < x < \sqrt{41} - \tfrac{4y}{5}\right) \vee \left(\tfrac{20}{\sqrt{41}} \leq y < 5 \wedge -\sqrt{25 - y^2} < x < \sqrt{25 - y^2}\right) \vee \\
&\quad \left(-\tfrac{20}{\sqrt{41}} < y < \tfrac{20}{\sqrt{41}} \wedge -\sqrt{25 - y^2} < x \leq \sqrt{41} - \tfrac{4y}{5}\right)
\end{aligned}
$$

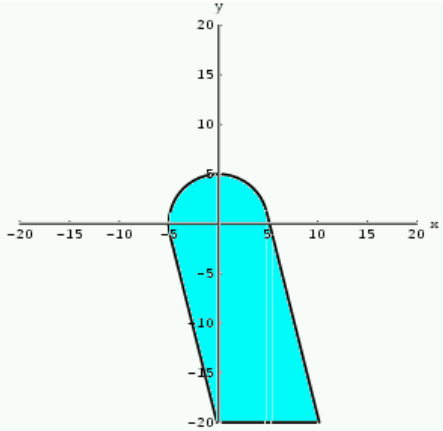
$$
\begin{aligned}
& v_1 = 4; v_2 = 5; \lambda = \tfrac{3}{5} \\
\texttt{unsafeLeft} &= \texttt{Resolve}\left[\exists t > 0 \wedge (x - v_1 t + \lambda v_2 t)^2 + (y + \sqrt{1-\lambda^2} v_2 t)^2 \leq 25\right] \\
&= \left(y < -\tfrac{5}{\sqrt{17}} \wedge -\tfrac{5\sqrt{17}}{4} - \tfrac{y}{4} \leq x \leq \tfrac{5\sqrt{17}}{4} - \tfrac{y}{4}\right) \vee \left(y = -\tfrac{5}{\sqrt{17}} \wedge -\tfrac{5\sqrt{17}}{4} - \tfrac{y}{4} < x \leq \tfrac{5\sqrt{17}}{4} - \tfrac{y}{4}\right) \vee \\
&\quad \left(y = \tfrac{5}{\sqrt{17}} \wedge -\sqrt{25 - y^2} < x < \tfrac{5\sqrt{17}}{4} - \tfrac{y}{4}\right) \vee \left(\tfrac{5}{\sqrt{17}} < y < 5 \wedge -\sqrt{25 - y^2} < x < \sqrt{25 - y^2}\right) \vee \\
&\quad \left(-\tfrac{5}{\sqrt{17}} < y < \tfrac{5}{\sqrt{17}} \wedge -\sqrt{25 - y^2} < x \leq \tfrac{5\sqrt{17}}{4} - \tfrac{y}{4}\right)
\end{aligned}
$$

$$
\begin{aligned}
& v_1 = 4; v_2 = 5; \lambda = -\tfrac{3}{5} \\
\texttt{unsafeRight} &= \texttt{Resolve}\left[\exists t > 0 \wedge (x - v_1 t + \lambda v_2 t)^2 + (y + \sqrt{1-\lambda^2} v_2 t)^2 \leq 25\right] \\
&= \left(y < -7\sqrt{\tfrac{5}{13}} \wedge -\tfrac{5\sqrt{65}}{4} - \tfrac{7y}{4} \leq x \leq \tfrac{5\sqrt{65}}{4} - \tfrac{7y}{4}\right) \vee \left(y = -7\sqrt{\tfrac{5}{13}} \wedge -\tfrac{5\sqrt{65}}{4} - \tfrac{7y}{4} < x \leq \tfrac{5\sqrt{65}}{4} - \tfrac{7y}{4}\right) \vee \\
&\quad \left(y = 7\sqrt{\tfrac{5}{13}} \wedge -\sqrt{25 - y^2} < x < \tfrac{5\sqrt{65}}{4} - \tfrac{7y}{4}\right) \vee \left(7\sqrt{\tfrac{5}{13}} < y < 5 \wedge -\sqrt{25 - y^2} < x < \sqrt{25 - y^2}\right) \vee \\
&\quad \left(-7\sqrt{\tfrac{5}{13}} < y < 7\sqrt{\tfrac{5}{13}} \wedge -\sqrt{25 - y^2} < x \leq \tfrac{5\sqrt{65}}{4} - \tfrac{7y}{4}\right)
\end{aligned}
$$

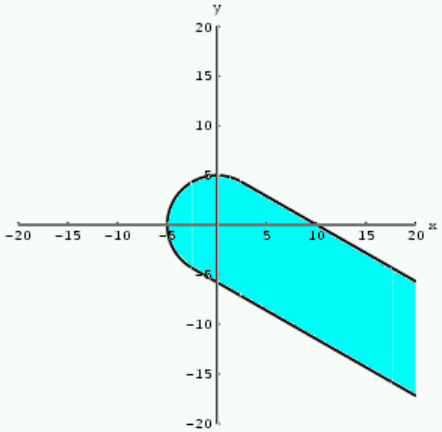
Safe Sets



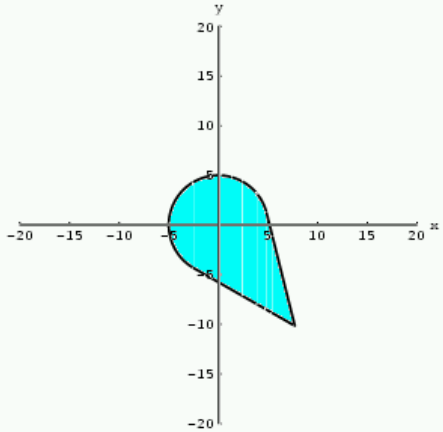
(a) `unsafeCruise`



(b) `unsafeLeft`



(c) `unsafeRight`



(d) `unsafeCruise` $\wedge$ `unsafeLeft` $\wedge$ `unsafeRight`

Time vs. Event Triggered Hybrid Systems

- Time-triggered:
  - Switching depends on time only
  - Switching and dynamics are decoupled
  - Switching times are known a priori
  - **Switched systems** more appropriate
- Event-triggered:
  - Switching also depends on state
  - Switching and dynamics are coupled
  - Switching times not known a priori
  - **Hybrid automata** more appropriate
- More specialized examples
  - Piece-wise affine systems (PWA)
  - Mixed Logical Dynamical
  - Linear Complementarity models

Basic problems of Hybrid Systems Verification/Synthesis

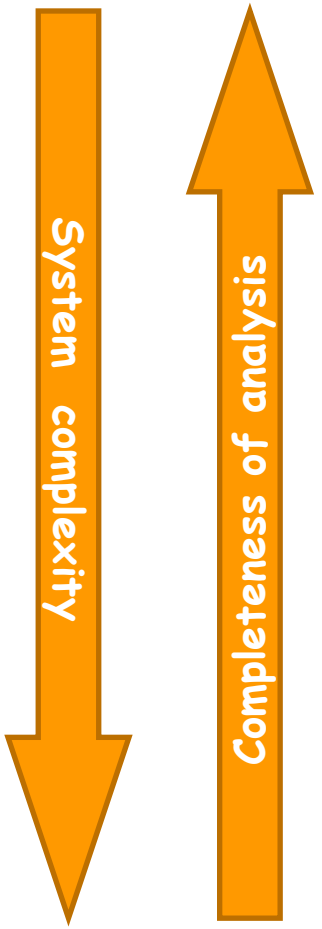
Safety verification: Is $\text{Reach}(S) \cap S_F$ empty?

Model checking: Does S satisfy temporal logic formula ?

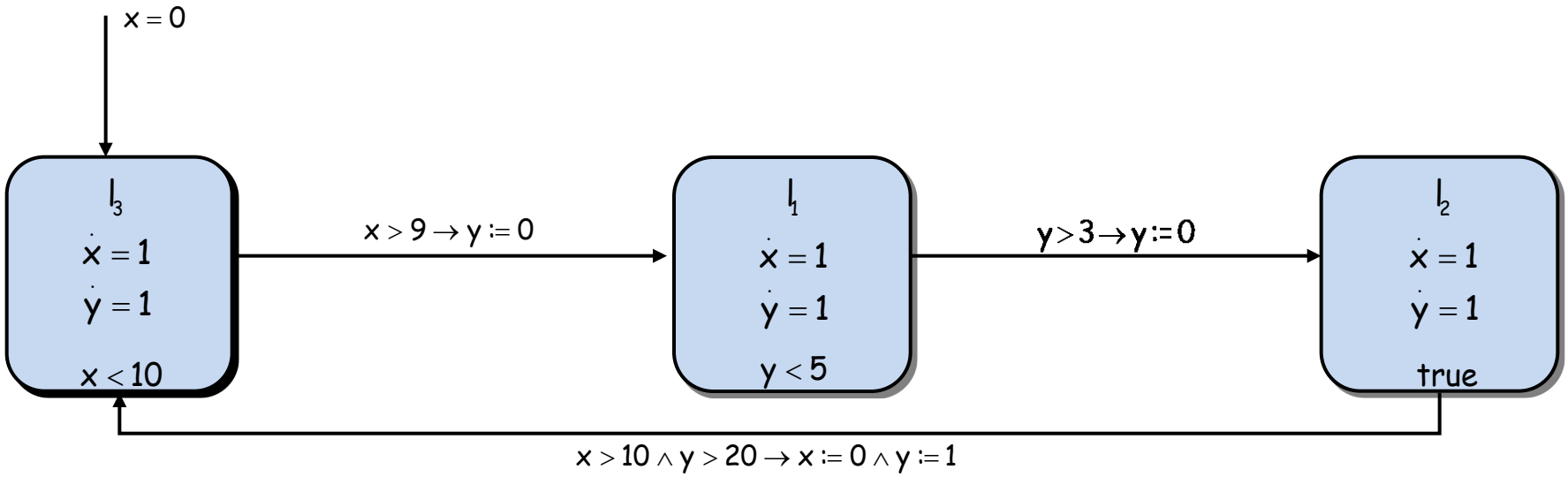
Controller Synthesis: Does S||C satisfy temporal logic formula ?

Hybrid Systems verification approaches

- Identifying decidable classes
- Discrete/predicate abstractions
- Barrier certificates / invariants
- Reachability algorithms
- Robust testing
- Systematic simulations / model based testing
- Statistical techniques



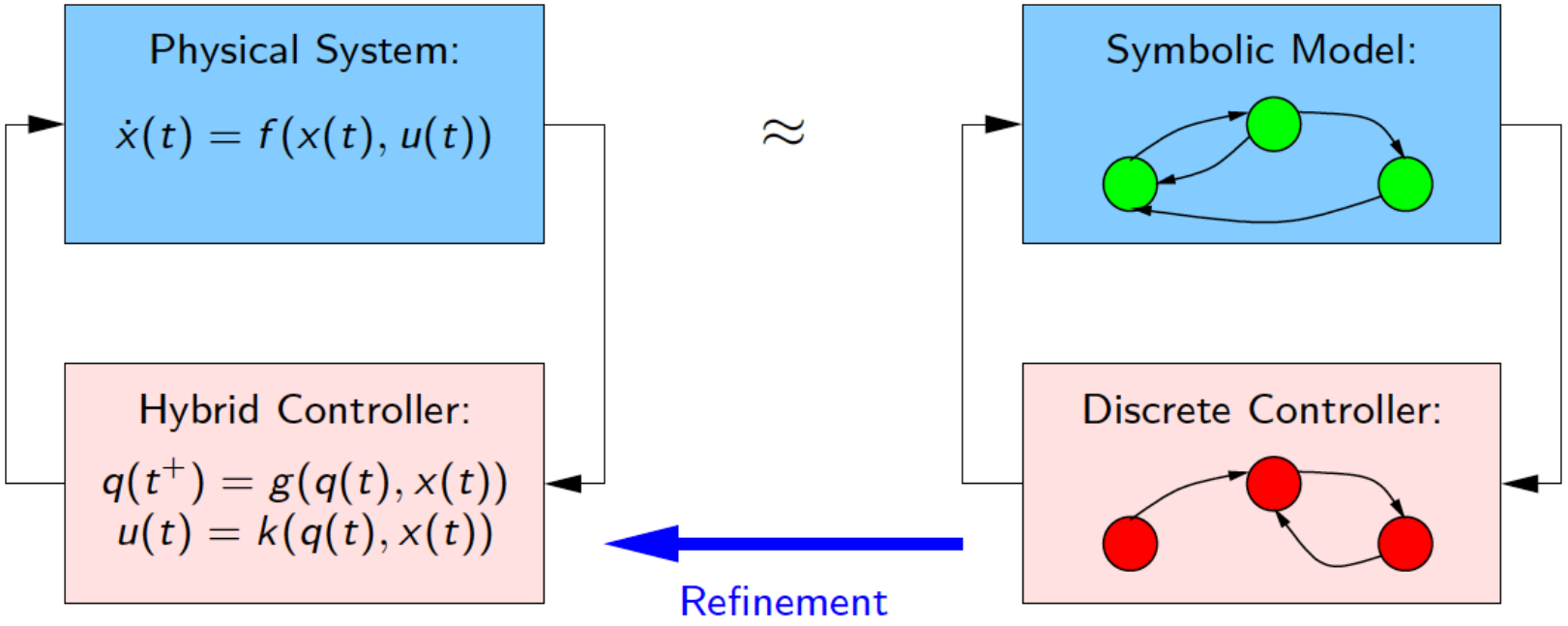
Finite abstractions of hybrid automata*



Theorem*: All initialized automata with rate-bounded flows (eg timed automata) admit a finite bisimulation. Also O-minimal systems.

*R. Alur and D. Dill, A theory of timed automata, Theoretical Computer Science, 1994
*G. Lafferriere, G.J. Pappas, S.S. Sastry, O-minimal hybrid systems, Mathematics of Control, Signals, and Systems, 2000
*R. Alur, T. Henzinger, G. Lafferriere, G.J. Pappas, Discrete abstractions of hybrid systems, Proceedings of IEEE, 2000
*A. Chutinam, B.H. Krogh, Verification of infinite-state systems using quotient transition systems, IEEE Transactions on Automatic Control, 2001

Symbolic abstraction for controller synthesis*



*P. Tabuada and G. J. Pappas, LTL control of discrete-time linear systems, IEEE Transactions on Automatic Control, 2006
*C. Belta, V. Isler, G.J. Pappas, Discrete abstractions for robot planning and control, IEEE Transactions in Robotics, 2005
*L. Habets, P.J. Collins, J. van Schuppen, Reachability and control synthesis for PWA hybrid systems, IEEE Transactions on Automatic Control, 2006

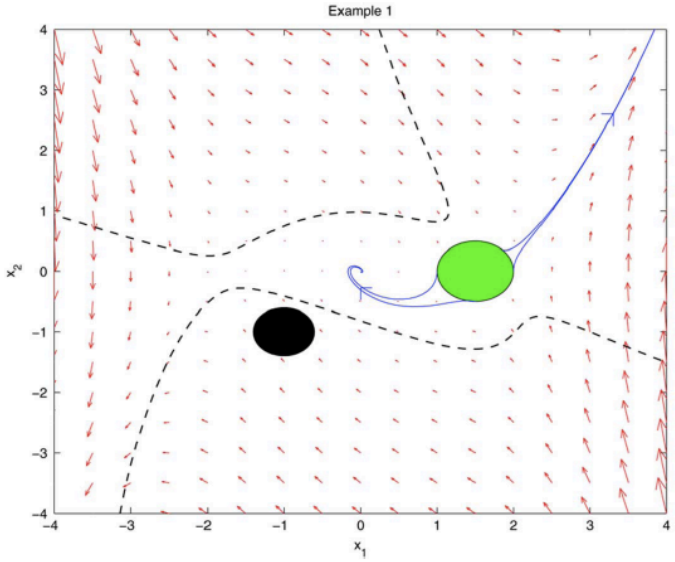
Safety verification using barrier certificates

- Given dynamics

$$\dot{x} = f(x, d)$$
$$x \in \mathbb{R}^n, d \in \mathcal{D},$$

- With the f a polynomial, set of initial set $\mathcal{X}_0$ and unsafe states $\mathcal{X}_u$ semialgebraic, input space $\mathcal{D}$ semialgebraic.

$$\mathcal{X}_0 = \{x \mid I(x) \geq 0\},$$
$$\mathcal{X}_u = \{x \mid U(x) \geq 0\},$$
$$\mathcal{D} = \{x \mid D(x) \geq 0\}.$$

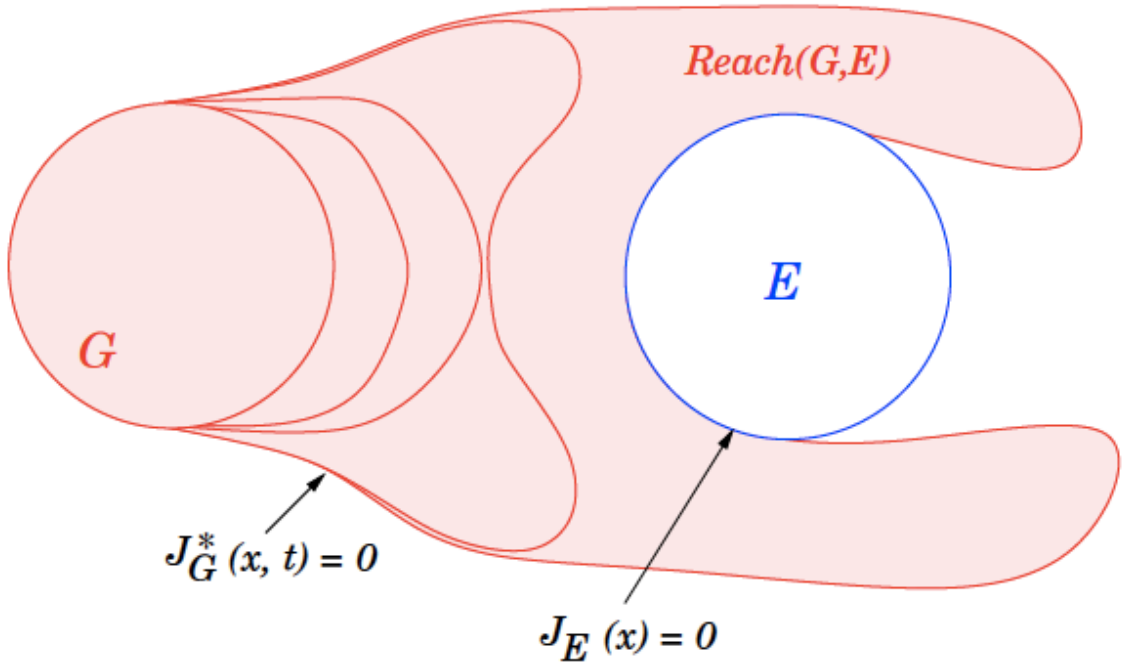


- A barrier certificate $B(x)$ should satisfy

$$B(x) > 0, \forall x \in \mathcal{X}_u,$$
$$B(x) \leq 0, \forall x \in \mathcal{X}_0,$$
$$\frac{\partial B}{\partial x} \cdot f(x, d) \leq 0, \forall x \in \mathbb{R}^n, d \in \mathcal{D}.$$

- This problem can be casted as a SOSP.

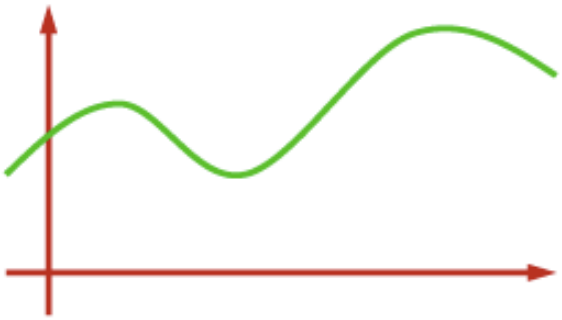
Exact reach set computation using HJB equation



$$\frac{\partial J_G(x,t)}{\partial t} + \min(0, H(x, \frac{\partial J_G(x,t)}{\partial x})) = 0$$

$$\text{subject to } J_G(x,t) \geq J_E(x)$$

TALIRO : Verification using robust testing



A/D

Boolean abstraction

LTL / MTL

$\Phi = \mathrm{G}\, p_1 \wedge \mathrm{F}_{[0,T]} \mathrm{G}\, p_2$

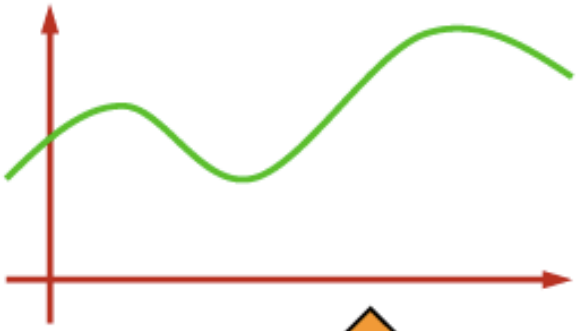
Monitoring Algorithm

Truth Value

$\{\bot, \top\}$

[Maler and Nickovic '04]
[Thati and Rosu '04]
[Rosu and Havelund '05]
[Geilen '01]
others ...

TALIRO : Verification using robust testing



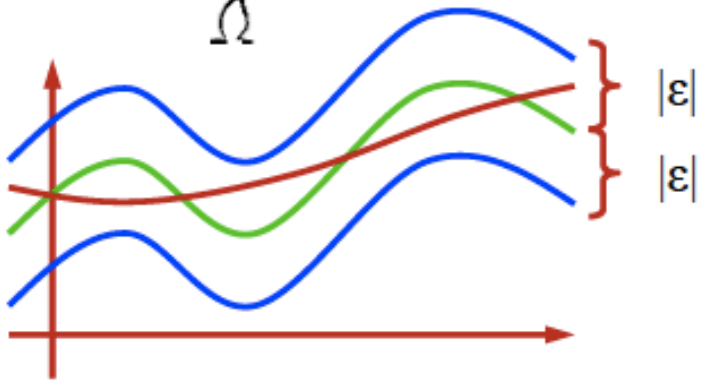
LTL / MTL

$\Phi = G(p_1 \rightarrow F_{\leq 2}\, p_2)$

Monitor/Tester

Robustness parameter

$\varepsilon \in \mathbb{R} \cup \{\pm\infty\}$

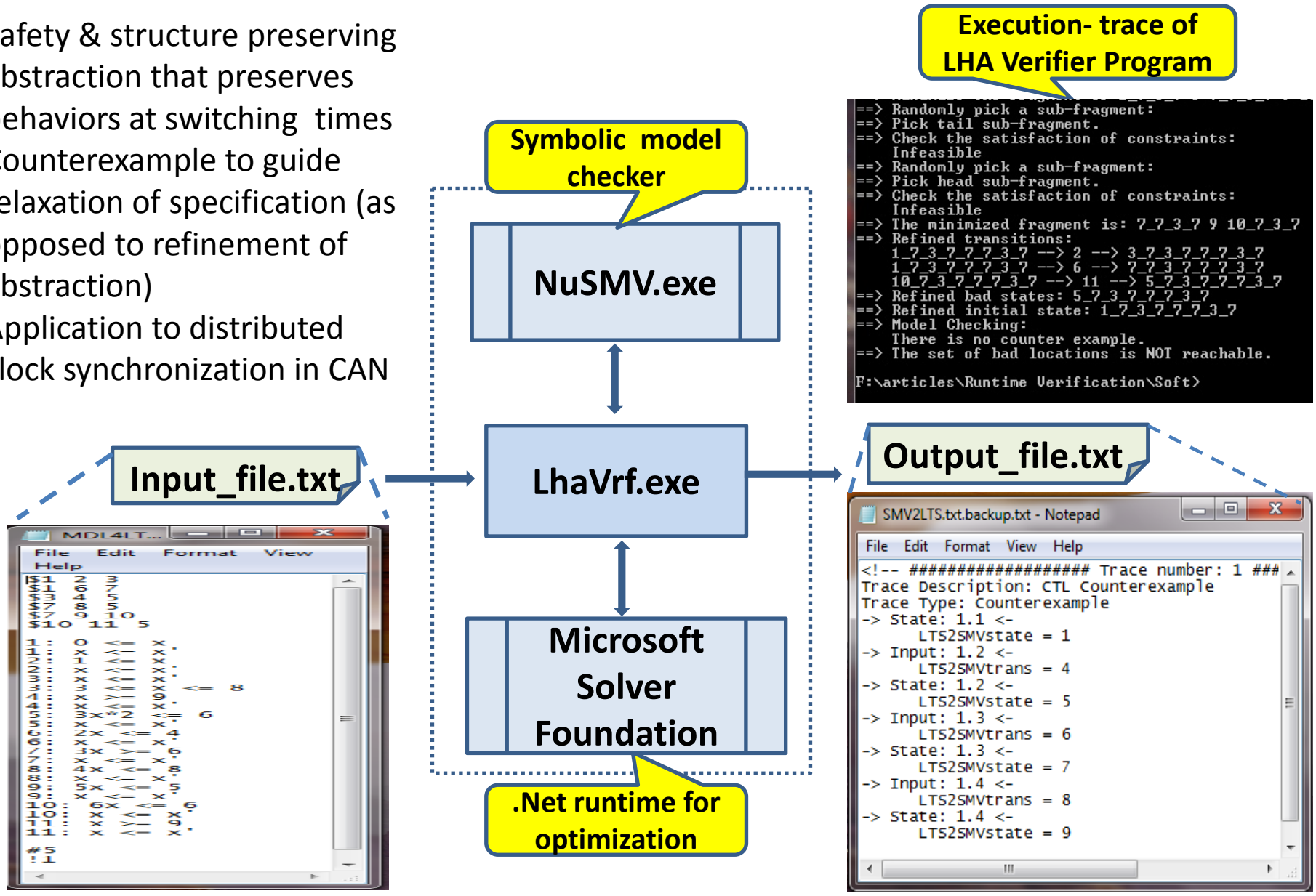


Fainekos and Pappas, *Robustness of temporal logic specifications for continuous-time signals*, TCS, 2009

Hybrid Automata Safety Verifier

Jiang (GM R&D) & Kumar (ISU)

- Safety & structure preserving abstraction that preserves behaviors at switching times
- Counterexample to guide relaxation of specification (as opposed to refinement of abstraction)
- Application to distributed clock synchronization in CAN



Tools

SpaceX (VERIMAG)

Reachability using zonotopes

LTLMOP (Cornell University)

- Temporal logic robot planning and control

PESSOA (U.C.L.A)

- Approximate symbolic control of nonlinear systems

MATISSE (U. Joseph Fourier)

- Approximate bisimulation computations

LTLcon (Boston University)

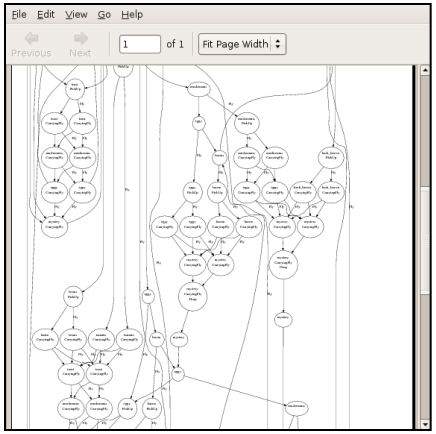
- LTL control of linear systems

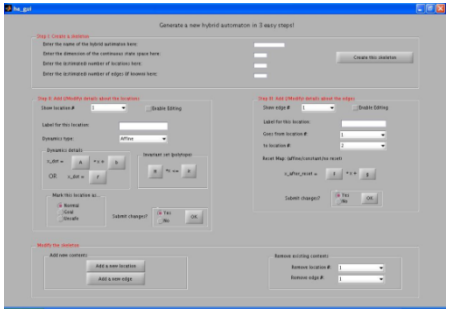
TALIRO (Arizona State, Colorado, NEC Labs)

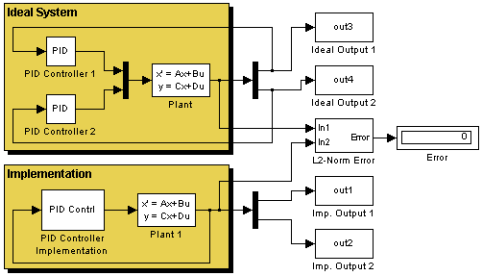
- Verification using robustness

TULIP (CalTech)

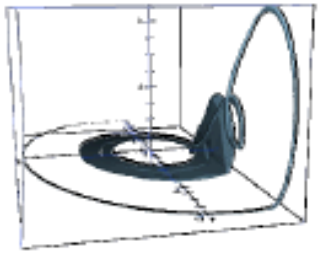
Model predictive temporal logic control







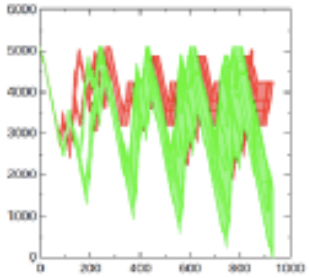
SpaceEx : State Space Explorer



Support Function Algo

–many continuous variables

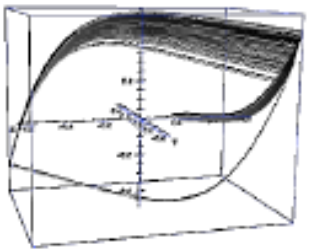
–low discrete complexity



PHAVer

–constant dynamics (LHA)

–formally sound and exact



Simulation

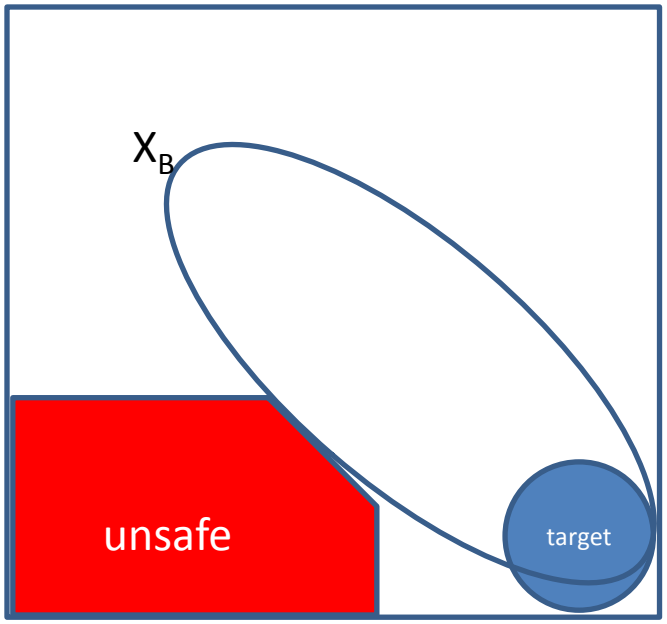
–nonlinear dynamics

–based on CVODE

Runtime Assurance:
A Proposed Approach

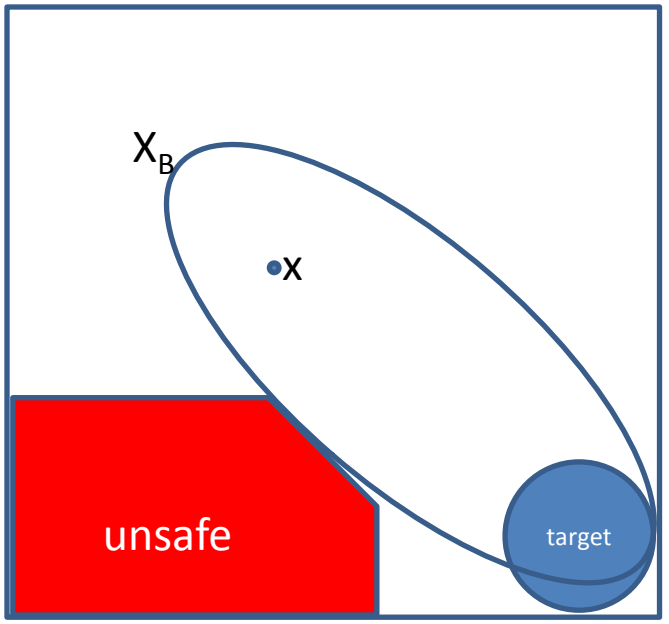
Safety Region of Baseline Control

- X_B: Safe states under baseline control
  - Set of states from where system is safely steerable to target states and invariance in target set maintainable
- To compute X_B:
  - Compute invariant subset of target
  - Compute backward reach/avoid set starting from invariant subset
  - Need to allow stochastic model of disturbance in reach/avoid set



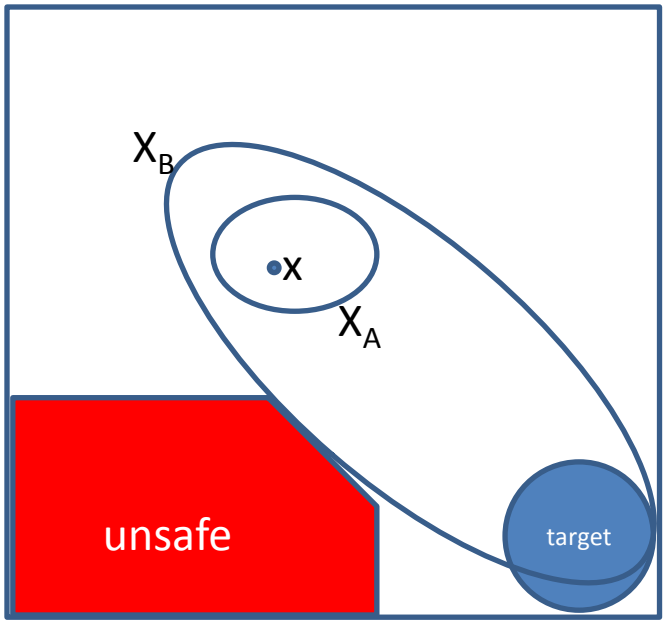
Current State versus Safety Region

- Current State x under advanced control must lie within X_B
  - Need to estimate current state and its variance to estimate
  - Current state **set** [x]



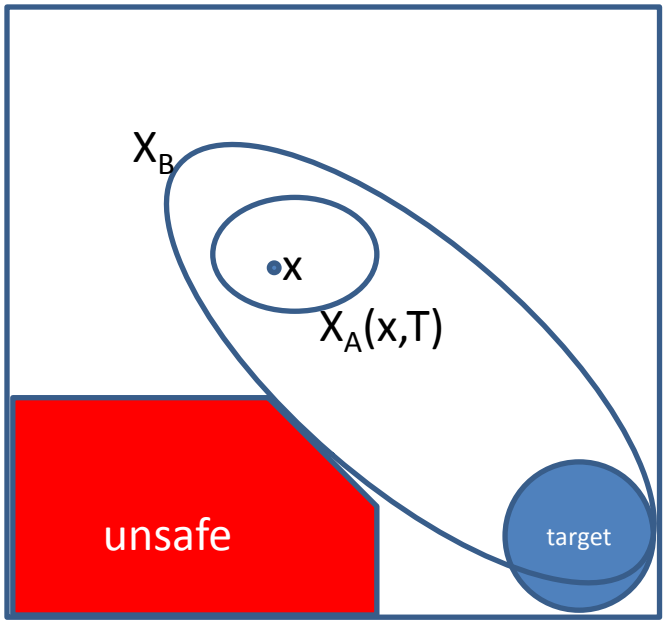
Active Safety Margin for Current State

- Need "safety margin" X_A to accommodate
  - Noisy estimate of current state
  - Reaction time of switching from advanced to baseline control
- Reaction time T:
  - Duration between decision vs. implementation of control switch
  - Need real-time models to compute/bound T



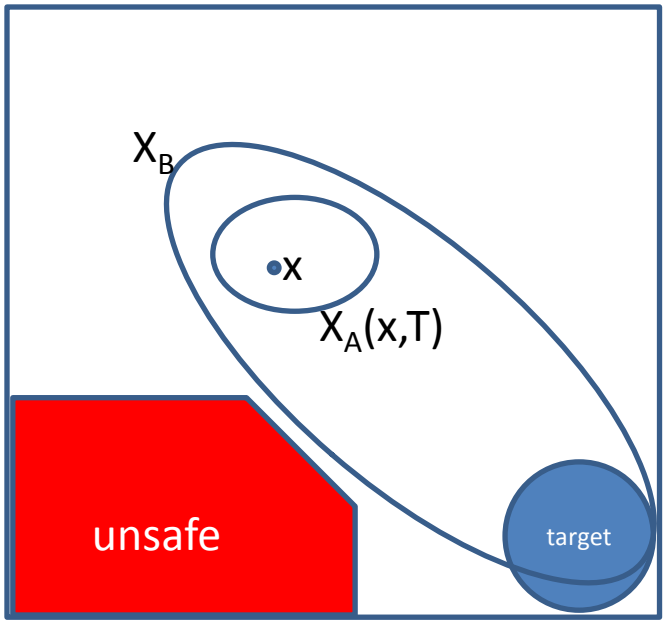
Active Safety Margin Characterization

- Active safety margin $X_A(x,T)$: States reachable from current state x within reaction time T under advanced control
- Over reaction time T, states under advanced control must remain inside X_B
  - Switch control at x if $X_A(x,T) \not\subset X_B$
- Need at least an abstract model of advance control to compute/bound $X_A(x,T)$



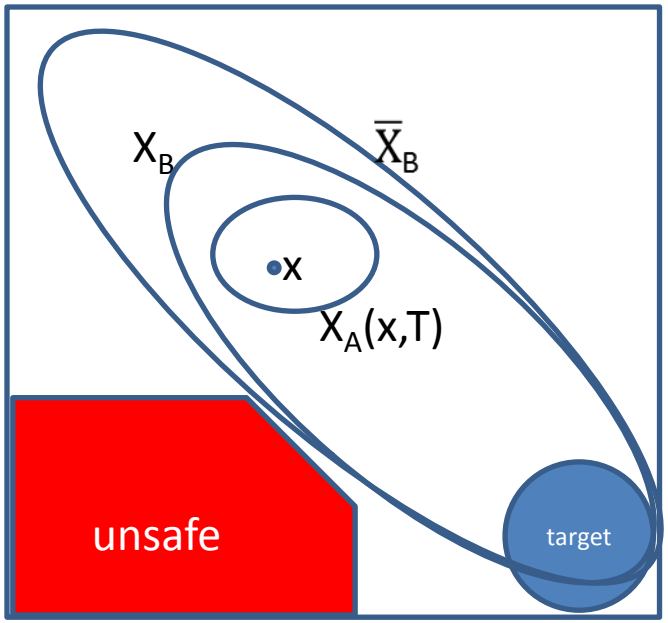
Additional Safety Against Faults

- Unverified advance control may also be faulty, so also:
  - Monitor input/output behaviors of advance control
  - Switch control at x if $[X_A(x,T) \not\subset X_B] \vee$ [Fault detected]
- Need to perform FDI by:
  - Monitoring of control software
  - Monitoring of controlled system to isolate any control software fault
  - Must account for noise



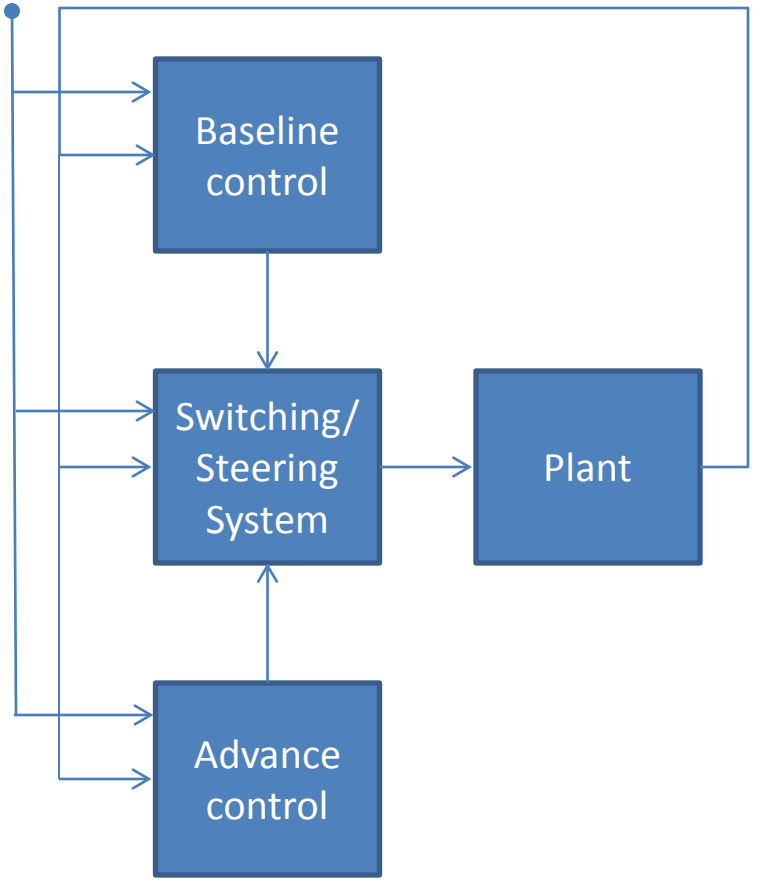
Enhancement: Steering Control

- Steering control may enlarge X_B to $\overline{X}_B$
  - $\overline{X}_B$: States safely steerable to X_B
  - Switch to steering control if $[X_A(x,T) \not\subset \overline{X}_B] \vee$ [advanced control fault detected]



Approach Summary

- Offline:
  - Compute safe states of baseline control X_B
  - Compute steerable states $\overline{X}_B$
  - Compute control switch reaction time T
  - Verify switching/steering system
- Runtime:
  - Estimate current state set x
  - Bound active safety margin $X_A(x,T)$, reachable states of x within T under advance control
  - Monitor i/o behaviors at software/system levels against their specs
  - Perform FDI to detect/isolate software faults
- Switch from advance to steering and next to baseline control at current state x if:
  - $[X_A(x,T) \not\subset \overline{X}_B] \vee$ [advanced control fault]



In Closing…

- Models Needed
  - Plant
  - Baseline control
  - Real-time analysis model of underlying distributed computing platform
  - Model/abstraction of advance control for bounding
  - Input/output correctness properties of advance control and of its controlled plant
  - Model of noise distribution
- Approach inspired from past works (Barron/CMU)

Acronyms

UAS: Unmanned Aerial System
V&V: Verification and Validation
RTA: Run Time Assurance
LTL: Linear-time Temporal Logic
CTL*: Computational-tree Temporal Logic
GUAS: Global Uniform Asymptotic Stability
LTI: Linear Time Invariant
PDE: Partial Differential Equation
SOS: Sum Of Squares