UNIVERSITY OF MICHIGAN

EECS ECE CSE

Correct-by-construction control synthesis for highly dynamics systems: an application in automotive safety systems

Necmiye Ozay
EECS, University of Michigan

Joint work with:
Petter Nilsson, Jessy Grizzle, Yuxiao Chen, Huei Peng (UM)
Aaron Ames (TAMU), Paulo Tabuada (UCLA), Jun Liu (Waterloo)

Exploration of Correct-By-Construction Controller Synthesis Seminar
September 2, 2015, JPL, Pasadena, CA

Research partly funded by NSF

Motivation

Excerpt From "Dad's Sixth Sense" (Hyundai) [2014 Super Bowl]

Third Highest Scoring Ad of all Time According to Ace Metrix

Motivation

- **Crashes in which at least one driver was distracted:**
  - 3,267 fatalities (**2010**)
  - **735,000 nonfatal injuries**

Automation can help with:

- <u>safety</u>, aging society, energy/congestion

- **Costs $45.8 billion in 2010**
  - roughly 17 percent of all economic costs from motor vehicle crashes.

Trends in Automation

NHTSA “Preliminary Statement of Policy Concerning Automated Vehicles” defines levels of automation:

- Level 0 - no automation
- Level 1 - Function-specific automation
- Level 2 - Combined function automation
- Level 3 - Limited self-driving automation
- Level 4 - Full self-driving automation

Trends in Automation

NHTSA "Preliminary Statement of Policy Concerning Automated Vehicles" defines levels of automation:

- Level 0 - no automation
- **Level 1 - Function-specific automation**
- **Level 2 - Combined function automation**
- Level 3 - Limited self-driving automation
- Level 4 - Full self-driving automation

All new vehicles in the US at "Level 1" with electronic stability control since 2012.

Google

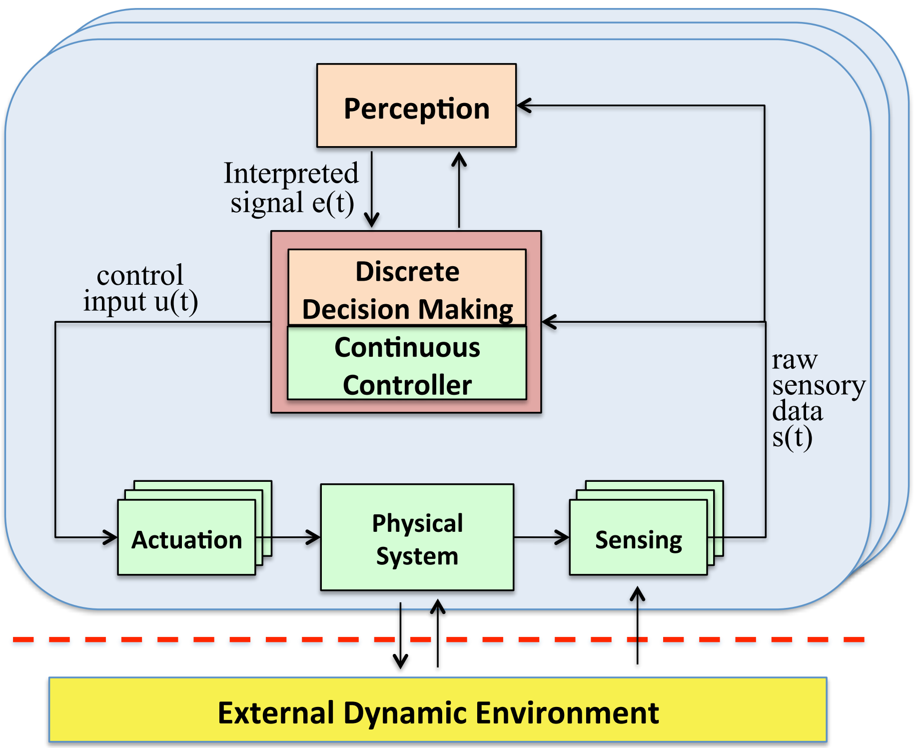
Challenges

- Many vehicle active safety systems are being conceived and deployed
  - extreme increase in software complexity
- Combining two safety systems can lead to unexpected interactions
  - hard to test and certify
- Many complaints (e.g., unintended acceleration) and recalls
  - cost to economy

How can we build (semi)autonomous systems that we can trust our lives with?

Can we guarantee correctness by design using formal methods?

Cyber-Physical Control Systems

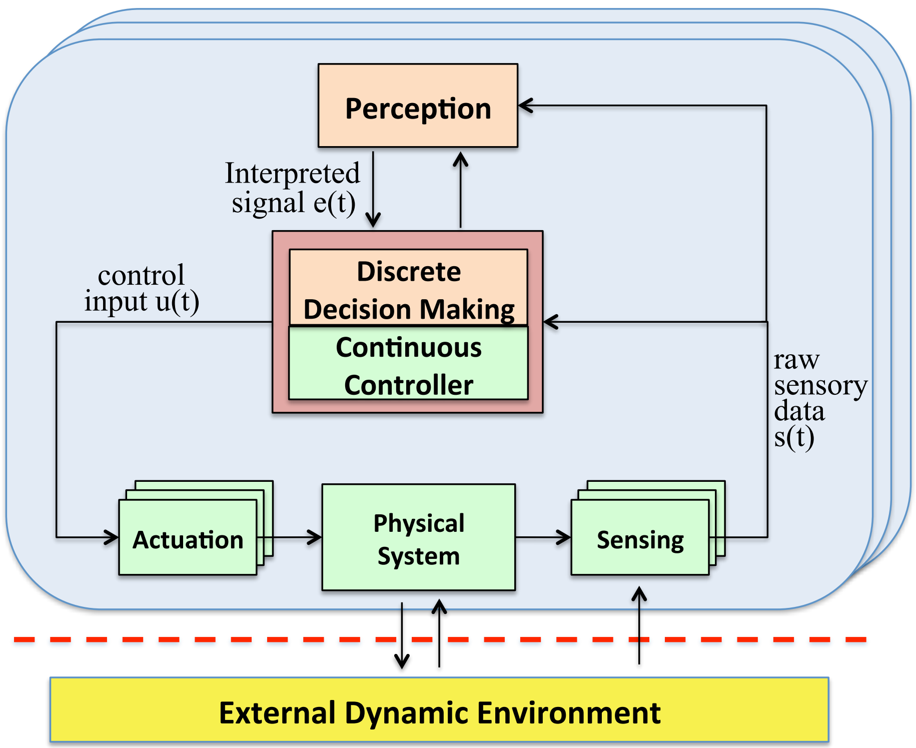


Several interacting feedback loops

computational, discrete
physical, continuous

Spatially distributed, communication between blocks/(sub)systems

Cyber-Physical Control Systems



physical, continuous

$$\dot{x}_p = f_p(x_p, u, \eta)$$
$$s = g_p(x_p, u, \mu)$$

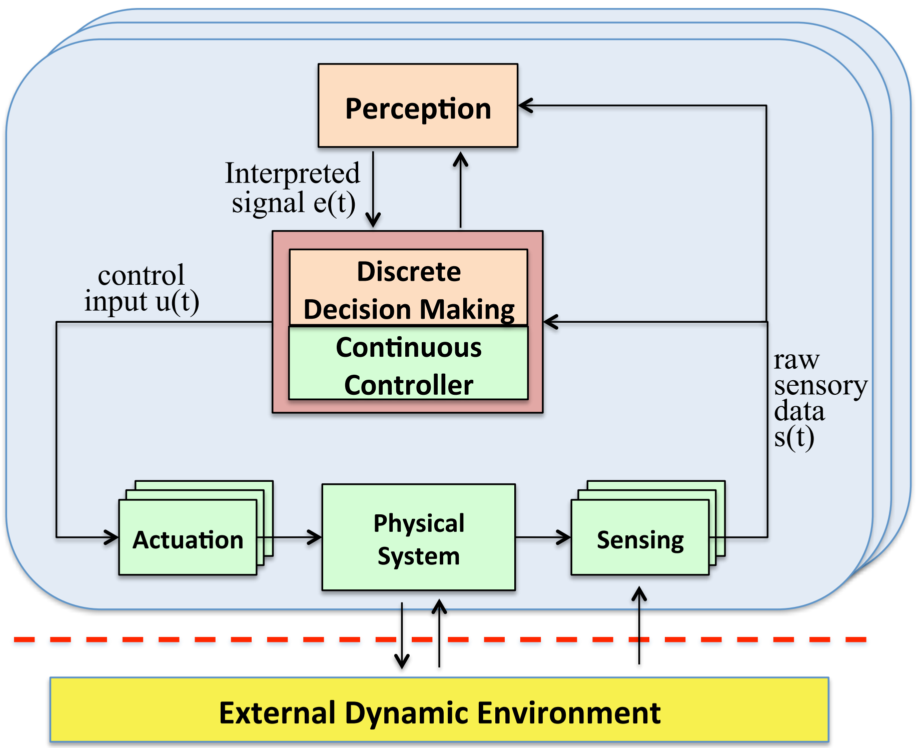
physical system

$$\dot{x}_c = f_c(x_c, s)$$
$$u = g_c(x_c, s)$$

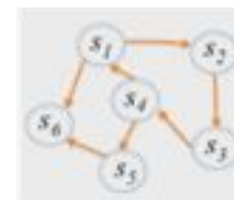
feedback controller

Models: Differential equation
Specifications: Stability, reference tracking, optimality

Cyber-Physical Control Systems



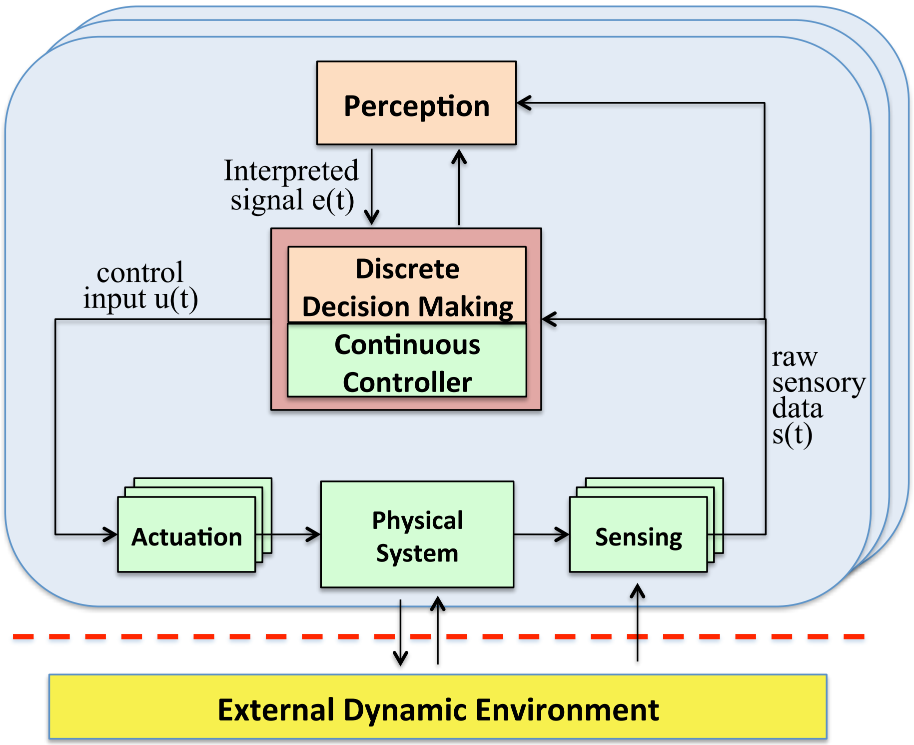
computational, discrete



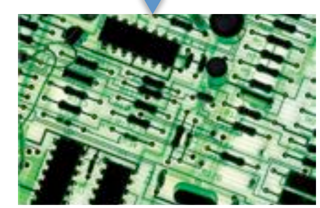
$$TS(S, Act, \rightarrow, AP)$$

Models: Discrete-event systems, automata, transition systems
Specifications: Safety, liveness, diagnosability

Cyber-Physical Control Systems



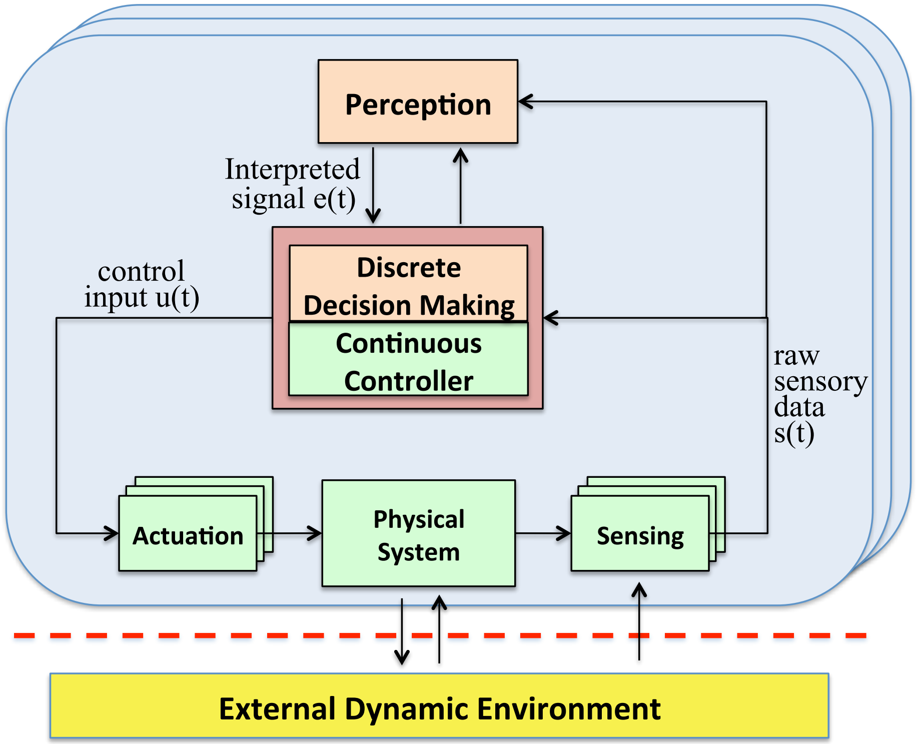
controller implementation



embedded system

Models: C code
Specifications: software specs

Cyber-Physical Control Systems



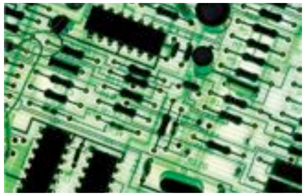
$$\dot{x}_p = f_p(x_p, u, \eta)$$
$$s = g_p(x_p, u, \mu)$$

physical system

$$\dot{x}_c^{(i)} = f_c(x_c^{(i)}, s^{(i)})$$
$$u^{(i)} = g_c(x_c^{(i)}, s^{(i)})$$

distributed feedback controllers

embedded systems

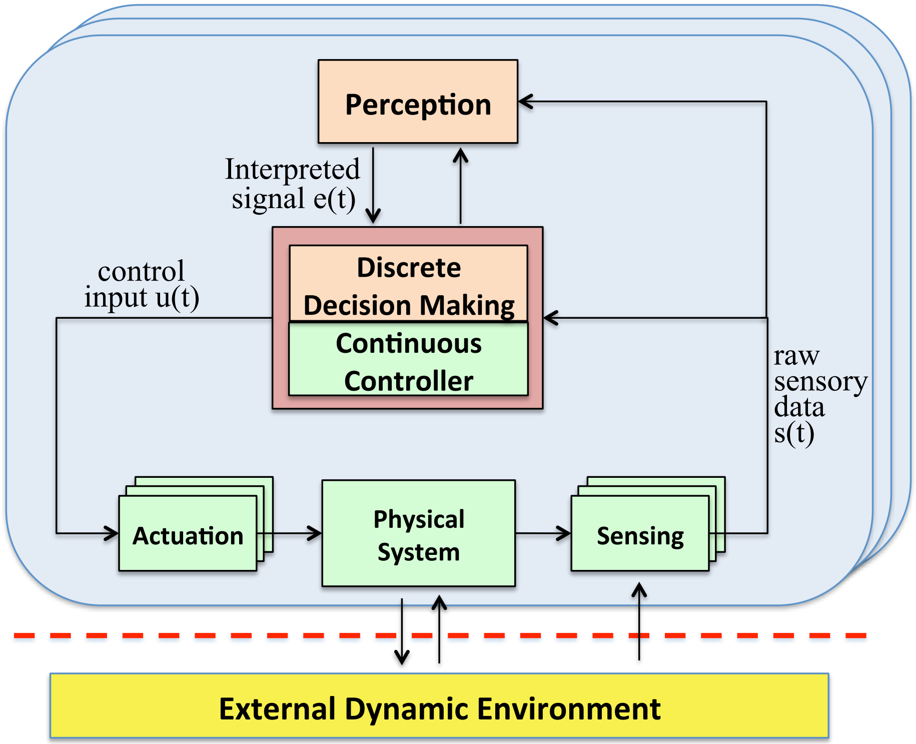


discrete events

We know how to design the pieces.
Integration is challenging!

Convergence of:
Control
Communication
Computer Science

Cyber-Physical Control Systems



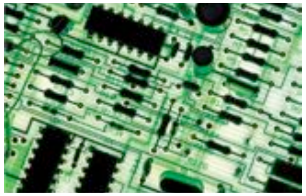
$$\dot{x}_p = f_p(x_p, u, \eta)$$
$$s = g_p(x_p, u, \mu)$$

physical system

$$\dot{x}_c^{(i)} = f_c(x_c^{(i)}, s^{(i)})$$
$$u^{(i)} = g_c(x_c^{(i)}, s^{(i)})$$

distributed feedback controllers

embedded systems

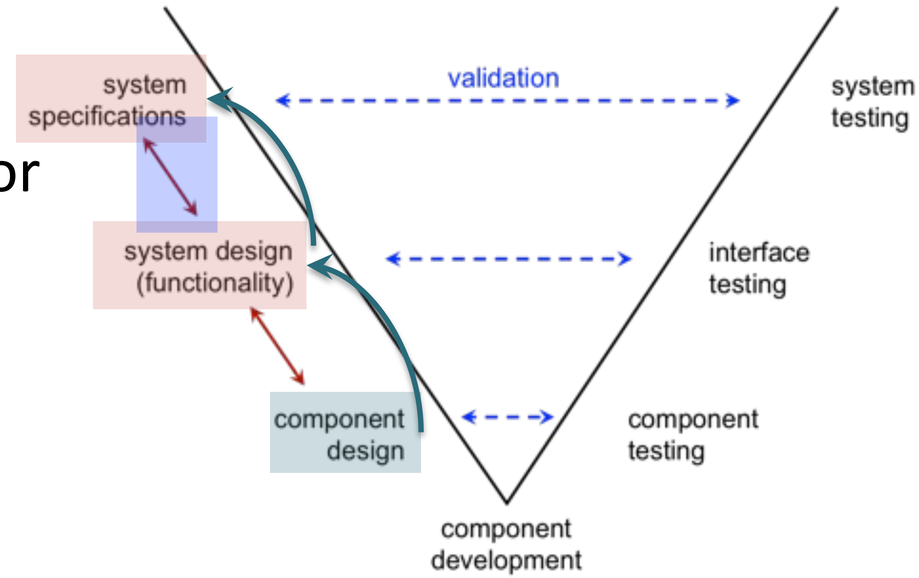


discrete events

Goal: Develop scalable tools for modular control synthesis!

Convergence of:
Control
Communication
Computer Science

Current Practice



- Current control design process for cyber-physical systems:
  - Given some spec (plain English) use **art of design** (engineering intuition, experience) and extensive testing/fine-tuning to come up with a single solution
  - little or no formal guarantees on correctness
  - no formal insight as to internal mechanisms

Better alternative: model-based approach, formal languages for specification, modular design, correct-by-construction embedded controller synthesis

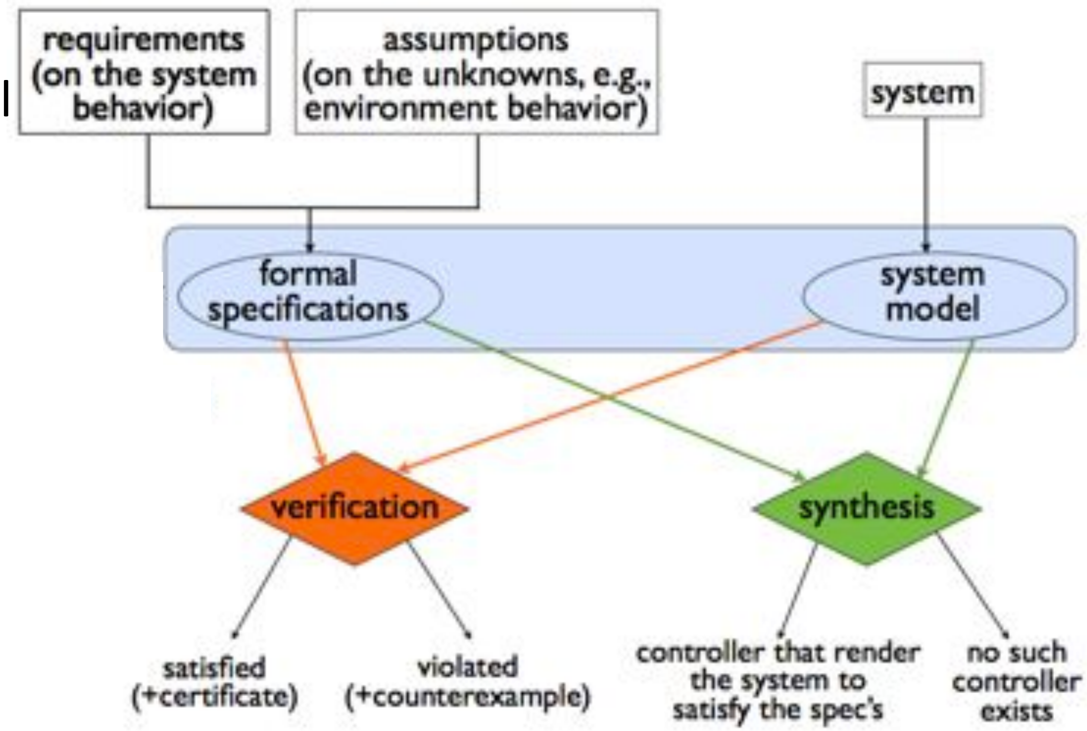
Formal Methods

- Mathematical and algorithmic techniques to reason about system/software behavior…
- Originally developed in computer science (software) domain. We integrate these methods with dynamics and control.
- Can we guarantee correctness by design using formal methods?
  - Reduce the need for extensive testing
  - Characterize explicitly all safe and unsafe conditions

Formal Methods for Control System Analysis and Synthesis

- Models for:
  - the system (usually hybrid/switched ODEs, with continuous/discrete inputs, disturbances and parametric uncertainty)
  - the environment (faults, external events)
- Formalized assumptions and requirements
  - linear temporal logic and its extensions
- Methods for verification and synthesis
  - algorithms that can process formal models and requirements to do analysis and control synthesis

Model-based approach



Correct by construction!

Specifying Correct Behavior Using Linear Temporal Logic

Extends propositional logic with temporal operators

$\wedge$ (and), $\vee$ (or), $\rightarrow$ (implies), $\neg$ (not),

$\Diamond$ (eventually), $\Box$ (always), $\mathcal{U}$ (until), $\bigcirc$ (next),

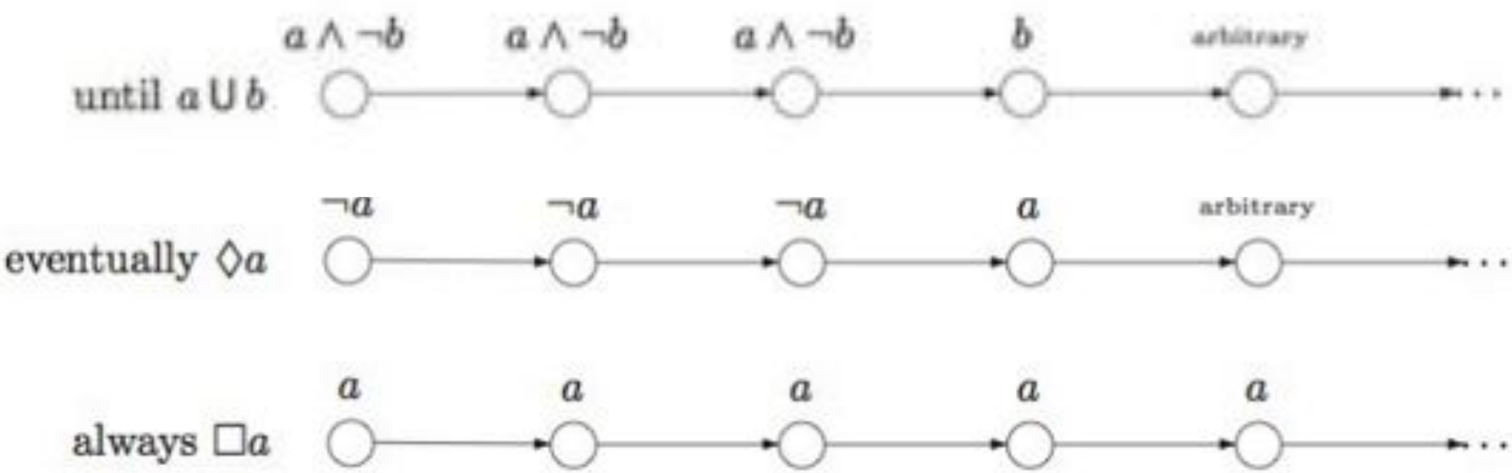
- Building blocks: atomic propositions + logic operators
  - An **atomic proposition** p is a subset of the state-space (e.g. R^n). We say that a state x(t) at time t satisfies p if x(t) $\in$ p.
- Allows to reason about infinite sequences of states
- Specifications (formulas) describe sets of allowable and desired behavior
  - safety: what actions/states are "not bad" or allowed
  - liveness: when an action can be/should be taken (e.g., infinitely often)

Specifying Correct Behavior Using Linear Temporal Logic

Extends propositional logic with temporal operators

$\wedge$ (and), $\vee$ (or), $\rightarrow$ (implies), $\neg$ (not),

$\Diamond$ (eventually), $\Box$ (always), $\mathcal{U}$ (until), $\bigcirc$ (next),



- LTL operators can be combined to specify interesting behavior:

$$\Box[(\text{engine temperature} \leq 240F) \rightarrow (\text{valve 1 closed})]$$

Formalizing the problem

$$\Sigma : \begin{array}{c} \dot{x} = f(x, u, \delta) \\ y = g(x, u, \delta) \\ x(0) \in \mathcal{X}_0 \end{array}$$

u: control inputs

δ: disturbance

y: outputs available to control

Propositions:

$$\Pi \doteq \{\pi_{init}, \pi_1, \ldots, \pi_{n_p}\}$$

$$h : X \rightarrow 2^{\Pi}$$

Formalizing the problem

$$\Sigma : \begin{array}{c} \dot{x} = f(x, u, \delta) \\ y = g(x, u, \delta) \\ x(0) \in \mathcal{X}_0 \end{array}$$

u: control inputs

δ: disturbance

y: outputs available to control

Propositions:

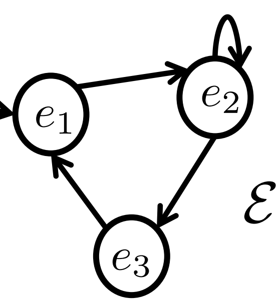
$$\Pi \doteq \{\pi_{init}, \pi_1, \dots, \pi_{n_p}\}$$

$$h : X \rightarrow 2^{\Pi}$$

Environment:

$$e(t) \in \{e_1, e_2, \dots, e_N\}; \forall t$$

Continuous-time discrete-valued signal (*with finite variability*)



Formalizing the problem

$$\Sigma : \begin{array}{c} \dot{x} = f(x, u, \delta) \\ y = g(x, u, \delta) \\ x(0) \in \mathcal{X}_0 \end{array}$$

u: control inputs

δ: disturbance

y: outputs available to control

Propositions:

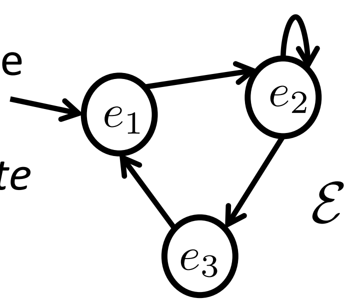
$$\Pi \doteq \{\pi_{init}, \pi_1, \ldots, \pi_{n_p}\}$$

$$h : X \to 2^{\Pi}$$

Environment:

$$e(t) \in \{e_1, e_2, \ldots, e_N\}; \forall t$$

Continuous-time discrete-valued signal (*with finite variability*)

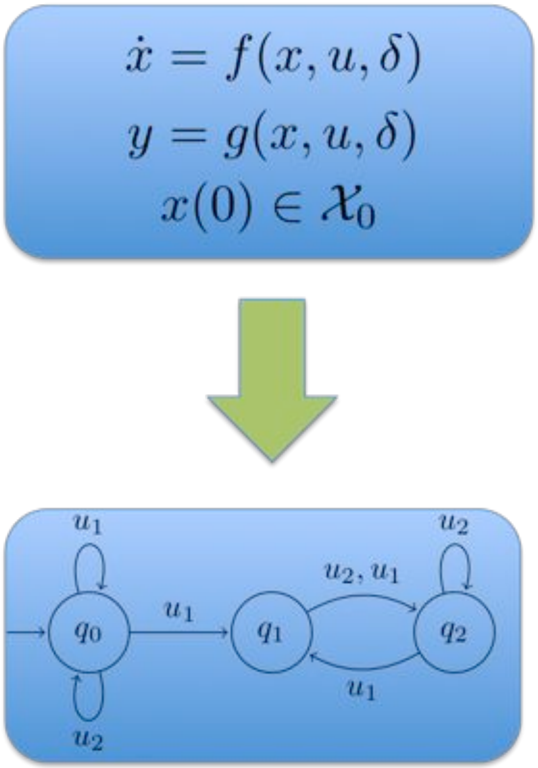


Problem statement:

Given a dynamical system Σ, a set of propositions over its state space (Π, h), an environment description $\mathcal{E}$ and some LTL (without next) specification φ, design a controller $u(y(t), e(t))$ such that the trajectories of the system satisfies the spec for all initial conditions x(0) in a given set, for all disturbances d, and for all environment behaviors.

Tools for reactive synthesis and control

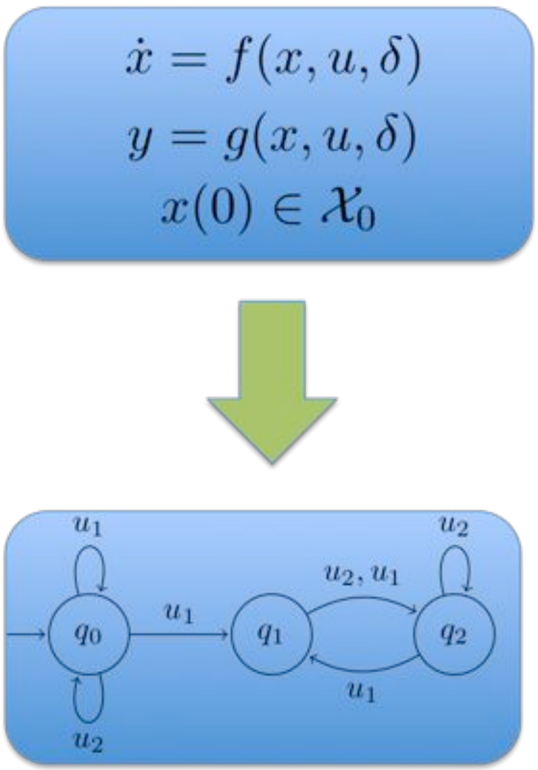
Abstraction-based



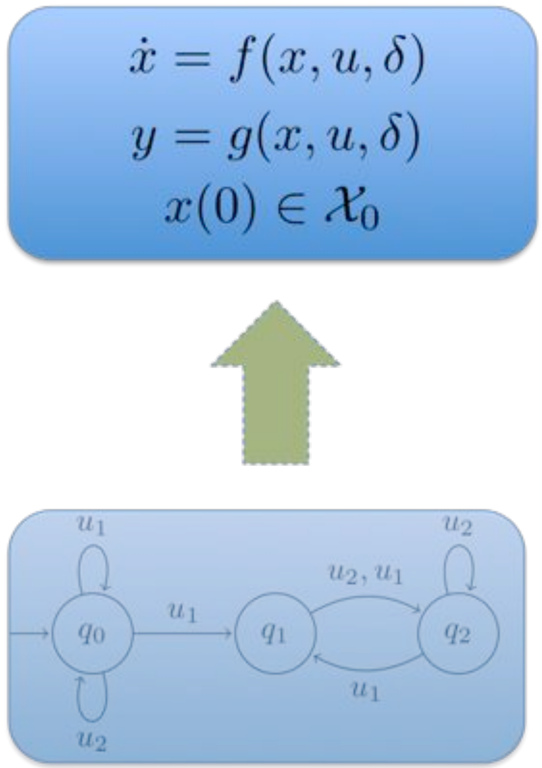
OLPM ACC 13, LOTM TAC 13
LO HSCC 14

Tools for reactive synthesis and control

Abstraction-based



Fixed-point operations on continuous domain



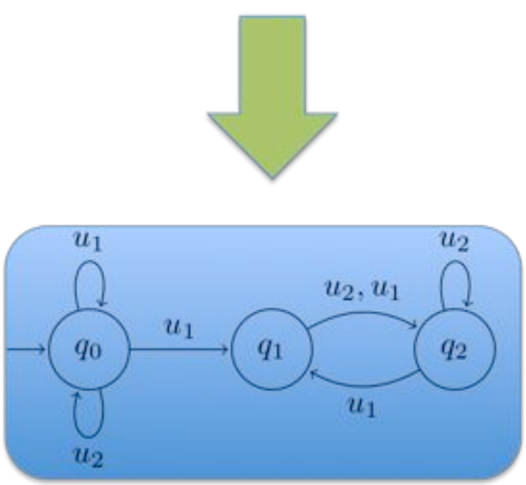
OLPM ACC 13, LOTM TAC 13
LO HSCC 14

N+ CDC 14, CST 15 (s)

Tools for reactive synthesis and control

Abstraction-based

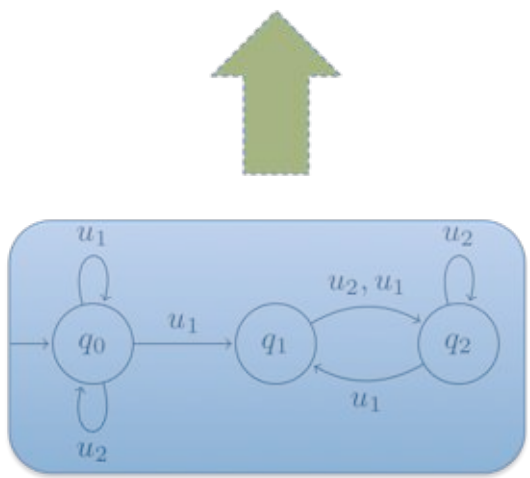
$$\dot{x} = f(x, u, \delta)$$
$$y = g(x, u, \delta)$$
$$x(0) \in \mathcal{X}_0$$



OLPM ACC 13, LOTM TAC 13
LO HSCC 14

Fixed-point operations on continuous domain

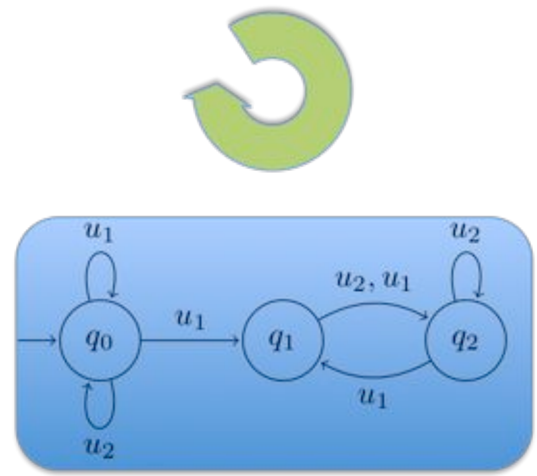
$$\dot{x} = f(x, u, \delta)$$
$$y = g(x, u, \delta)$$
$$x(0) \in \mathcal{X}_0$$



N+ CDC 14, CST 15 (s)

Incremental synergistic approach

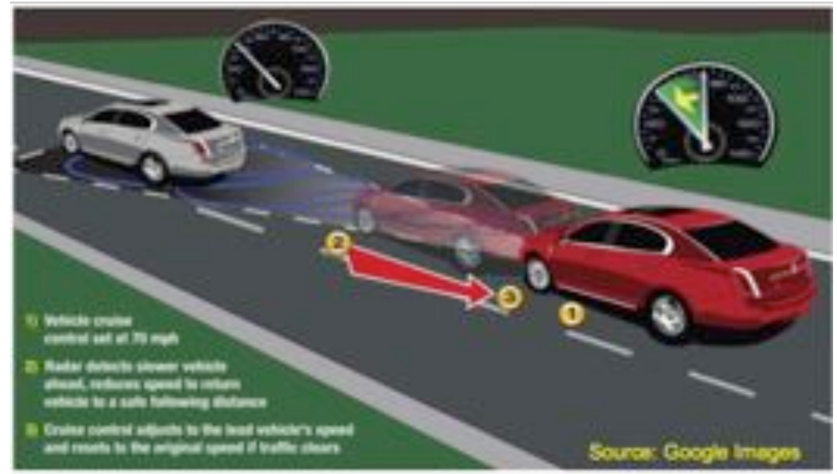
$$\dot{x} = f(x, u, \delta)$$
$$y = g(x, u, \delta)$$
$$x(0) \in \mathcal{X}_0$$



NO CDC 14

Applications in automotive safety: Adaptive cruise control

Adaptive Cruise Control (ACC)



- Two modes of operation
  - If there is no lead car in front (M1), regulate velocity (v)
  - If there is a car close enough (M2), regulate headway (h), distance to the lead car

  \+ in each mode hard safety constraints: acceleration limits, minimum allowed "time headway" (h/v)
- Mode is determined by a sensor (radar) reading: is there a car within the radar range and if so, how close it is?

Nilsson et al CDC 14

Formalizing specifications

- The ISO 15622 standard states:

  "When the ACC is active, the vehicle speed shall be controlled automatically either to maintain a time gap to a forward vehicle, or to maintain the set speed, whichever speed is lower. The change between these two control modes is made automatically by the ACC system."

Formalizing specifications

- The ISO 15622 standard states:

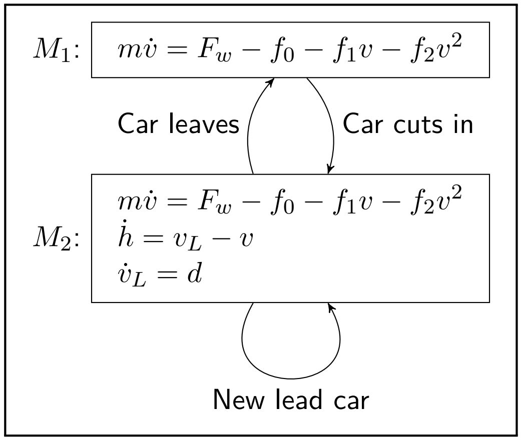
  "When the ACC is active, the vehicle speed shall be controlled automatically either to maintain a time gap to a forward vehicle, or to maintain the set speed, whichever speed is lower. The change between these two control modes is made automatically by the ACC system."

- The goal for each mode is to reach and stay in a desired goal set. This can be captured by

$$\Box\left(\Box M_i \Rightarrow \Diamond\Box G_i\right).$$

ACC: Model and Specifications

Model: Hybrid system with two modes:



- Input set: $S_2 = \{F_w \mid F_w \in [-0.3mg, 0.2mg]\}$.

Objectives: Goals for 'no lead car mode' M_1:

- Goal set: $G_1 = \{v \mid v \in [v_{des} - \Delta_v, v_{des} + \Delta_v\}$.

Goals for 'lead car mode' M_2:

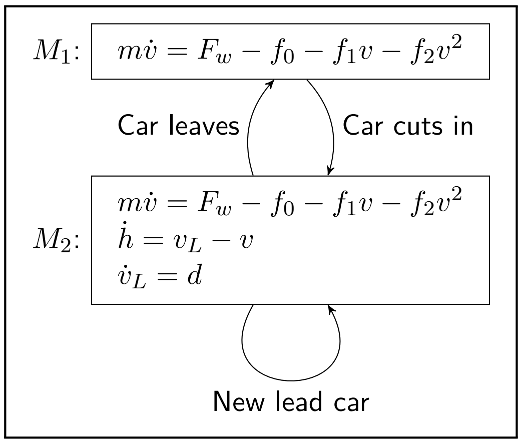
- Safe set: $S_1 = \{(v, h, v_L) \mid h/v \geq 1\}$.
- Goal set: $G_2 = \{(v, h, v_L) \mid h/v \geq 1.3,\ v < v_{des} + \Delta_v\}$.

Lead car assumptions

$$v_L \in [v_L^-, v_L^+]$$
$$a_L \in [a_L^-, a_L^+]$$

ACC: Model and Specifications

Model: Hybrid system with two modes:



- Input set: $S_2 = \{F_w \mid F_w \in [-0.3mg, 0.2mg]\}$.

Objectives: Goals for 'no lead car mode' M_1:

- Goal set: $G_1 = \{v \mid v \in [v_{des} - \Delta_v, v_{des} + \Delta_v\}$.

Goals for 'lead car mode' M_2:

- Safe set: $S_1 = \{(v, h, v_L) \mid h/v \geq 1\}$.
- Goal set: $G_2 = \{(v, h, v_L) \mid h/v \geq 1.3,\ v < v_{des} + \Delta_v\}$.

LTL Specification

$$\square S_U \wedge \square \left(\bigwedge_{i=1}^{2} (\square M_i \Rightarrow \Diamond \square G_i) \right).$$

ACC: LTL Specification

LTL Specification:

$$\Box\left((M_1 \vee S_1) \wedge S_2\right) \wedge \Box\left(\bigwedge_{i=1}^{2} \Box M_i \implies \Diamond\Box G_i\right).$$

Recall: $\Box$ means "always", $\Diamond\Box$ means "eventually always",
$M_1 \wedge S_1$ is equivalent to $M_2 \implies S1$

- In each mode, we need to satisfy certain hard safety constraints and if persistently in a mode, need to reach a goal set and remain in it.
- Need to be **reactive** to mode changes

Goal: Want to find a fixed point characterization!

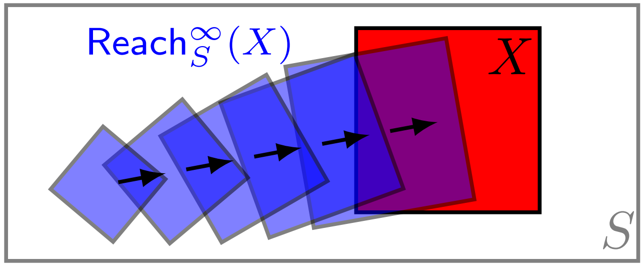
Set-valued operators: From every control theorists' tool set

Given a dynamical system $x^+ = f(x, u, d)$ with

- Input constraints $u \in \mathcal{U}$
- Disturbance assumptions $d \in \mathcal{D}$

Safe, robust reachability:

$$\text{Reach}_S^\infty(X) = \{x_0 \in S : X \text{ can be reached from } x_0\}$$



Set-valued operators: From every control theorists' tool set

Given a dynamical system $x^+ = f(x, u, d)$ with

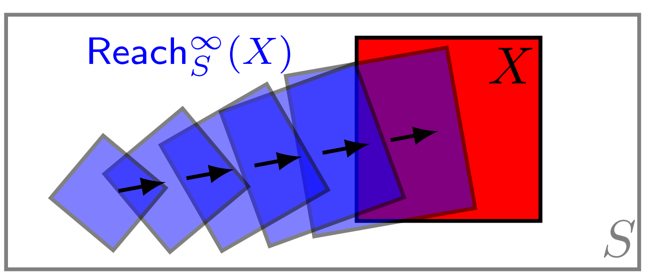
- Input constraints $u \in \mathcal{U}$
- Disturbance assumptions $d \in \mathcal{D}$

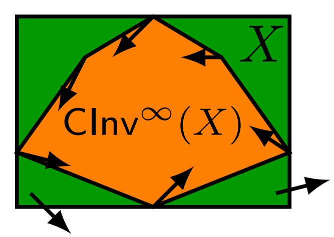
Safe, robust reachability:

$$\text{Reach}_S^\infty(X) = \{x_0 \in S : X \text{ can be reached from } x_0\}$$

Robust controlled invariance:

$$\text{CInv}^\infty(X) = \{x_0 \in X : X \text{ can be kept invariant starting from } x_0\}$$





Fixed point characterization

- Solve for specification of the form

$$\Box S \wedge \Box\left((\Box M_1 \implies \Diamond\Box G_1) \wedge (\Box M_2 \implies \Diamond\Box G_2)\right)$$

Search for sets C_1 and C_2 such that

$$C_1 \subset M_1 \cap \mathsf{Reach}_S^\infty \underbrace{\left(\mathsf{Inv}^\infty\left(G_1 \cap (C_1 \cup C_2)\right) \cup C_2\right)}_{D_1},$$

$$C_2 \subset M_2 \cap \mathsf{Reach}_S^\infty \underbrace{\left(\mathsf{Inv}^\infty\left(G_2 \cap (C_1 \cup C_2)\right) \cup C_1\right)}_{D_2}.$$

Correct control strategy if such C_1, C_2 are found:

- When in C_1, make progress toward D_1
- When in C_2, make progress toward D_2

Want to make C_1 and C_2 as large as possible to maximize controller domain

Fixed point characterization

- Solve for specification of the form

$$\Box S \wedge \Box\left(\left(\Box M_1 \implies \Diamond\Box G_1\right) \wedge \left(\Box M_2 \implies \Diamond\Box G_2\right)\right)$$

Search for sets C_1 and C_2 such that

$$C_1 \subset M_1 \cap \mathsf{Reach}_S^{\infty} \underbrace{\left(\mathsf{Inv}^{\infty}\left(G_1 \cap (C_1 \cup C_2)\right) \cup C_2\right)}_{D_1},$$

$$C_2 \subset M_2 \cap \mathsf{Reach}_S^{\infty} \underbrace{\left(\mathsf{Inv}^{\infty}\left(G_2 \cap (C_1 \cup C_2)\right) \cup C_1\right)}_{D_2}.$$

Correct control strategy if such C_1, C_2 are found:

- When in C_1, make progress toward D_1
- When in C_2, make progress toward D_2

Want to make C_1 and C_2 as large as possible to maximize controller domain

Fixed point characterization

- Solve for specification of the form

$$\Box S \wedge \Box\left((\Box M_1 \implies \Diamond\Box G_1) \wedge (\Box M_2 \implies \Diamond\Box G_2)\right)$$

Search for sets C_1 and C_2 such that

$$C_1 \subset M_1 \cap \mathsf{Reach}_S^\infty \underbrace{\left(\mathsf{Inv}^\infty \left(G_1 \cap (C_1 \cup C_2)\right) \cup C_2\right)}_{D_1},$$

$$C_2 \subset M_2 \cap \mathsf{Reach}_S^\infty \underbrace{\left(\mathsf{Inv}^\infty \left(G_2 \cap (C_1 \cup C_2)\right) \cup C_1\right)}_{D_2}.$$

Correct control strategy if such C_1, C_2 are found:

- When in C_1, make progress toward D_1
- When in C_2, make progress toward D_2

Want to make C_1 and C_2 as large as possible to maximize controller domain

Fixed point characterization

- Solve for specification of the form

$$\Box S \wedge \Box\left(\left(\Box M_1 \implies \Diamond\Box G_1\right) \wedge \left(\Box M_2 \implies \Diamond\Box G_2\right)\right)$$

Search for sets C_1 and C_2 such that

$$C_1 \subset M_1 \cap \mathsf{Reach}_S^\infty \underbrace{\left(\mathsf{Inv}^\infty\left(G_1 \cap (C_1 \cup C_2)\right) \cup C_2\right)}_{D_1},$$

$$C_2 \subset M_2 \cap \mathsf{Reach}_S^\infty \underbrace{\left(\mathsf{Inv}^\infty\left(G_2 \cap (C_1 \cup C_2)\right) \cup C_1\right)}_{D_2}.$$

Correct control strategy if such C_1, C_2 are found:

- When in C_1, make progress toward D_1
- When in C_2, make progress toward D_2

Want to make C_1 and C_2 as large as possible to maximize controller domain

Fixed point characterization

- Set computations directly on the continuous state space of a linearized system
- Conservative linearization
  - Reachability in linearized system implies reachability in original system
- Disturbance assumptions defined piecewise linearly

$$C_1^0 = M_1,\ C_2^0 = M_2$$

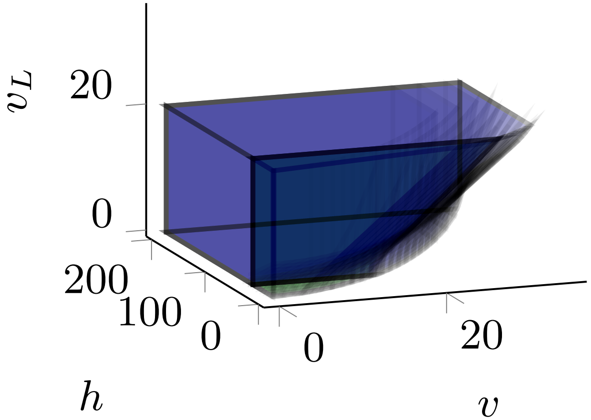
$$C_1^{k+1} = M_1 \cap \mathsf{Reach}_S^\infty \left(\mathsf{Inv}^\infty \left(G_1 \cap (C_1^k \cup C_2^k)\right) \cup C_2^k\right)$$

$$C_2^{k+1} = M_2 \cap \mathsf{Reach}_S^\infty \left(\mathsf{Inv}^\infty \left(G_2 \cap (C_1^k \cup C_2^k)\right) \cup C_1^k\right)$$

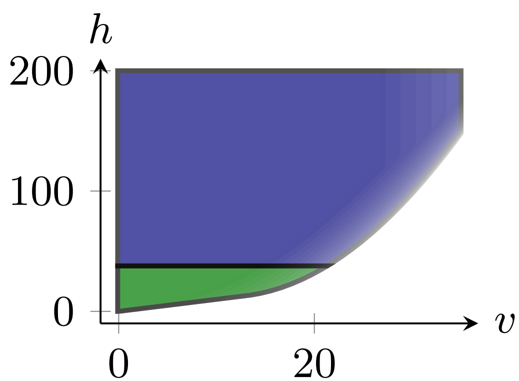
- Monotonically non-increasing sets (=> convergence)
- Use approximations (=> simpler sets and termination)
- Implementation: use MPC to move between sets (=> auto-generation of code)

Some results (safe control domain)

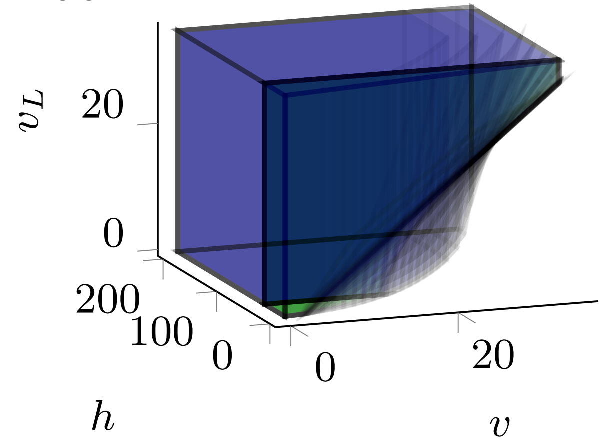
“Normal” lead car



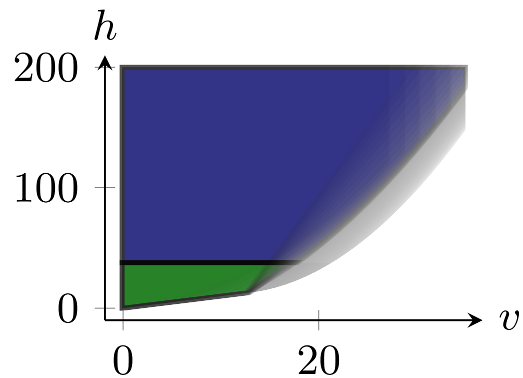
Cross section at $v_L = 10$ m/s



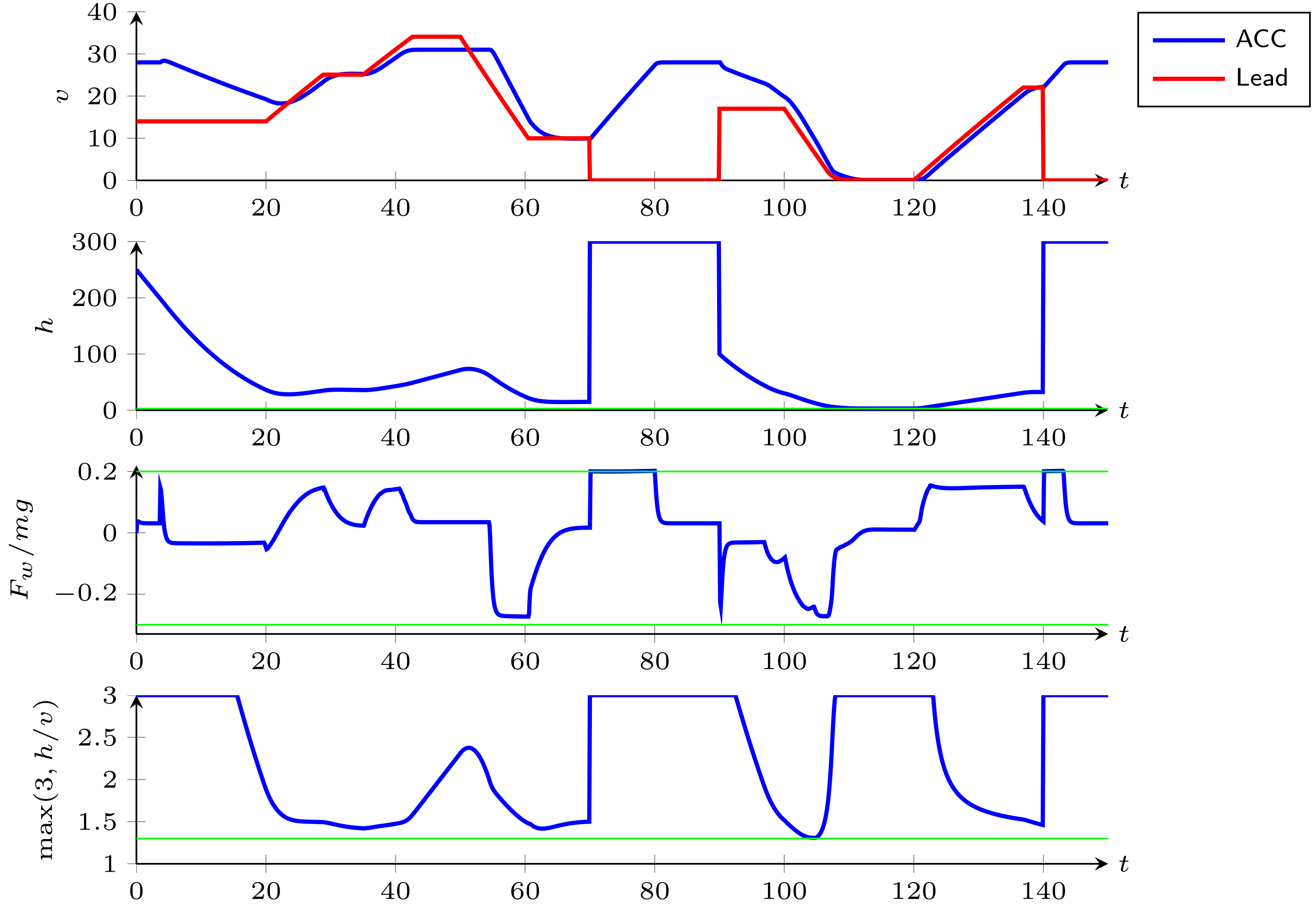
“Aggressive” lead car



Cross section at $v_L = 10$ m/s

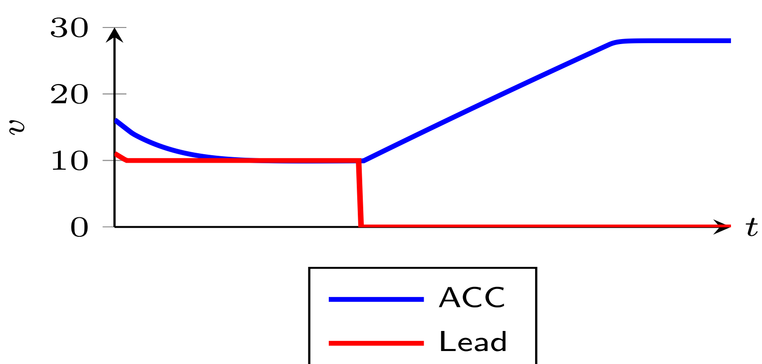


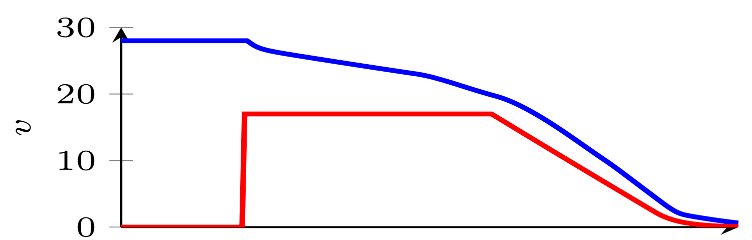
Simulations



CarSim Simulations

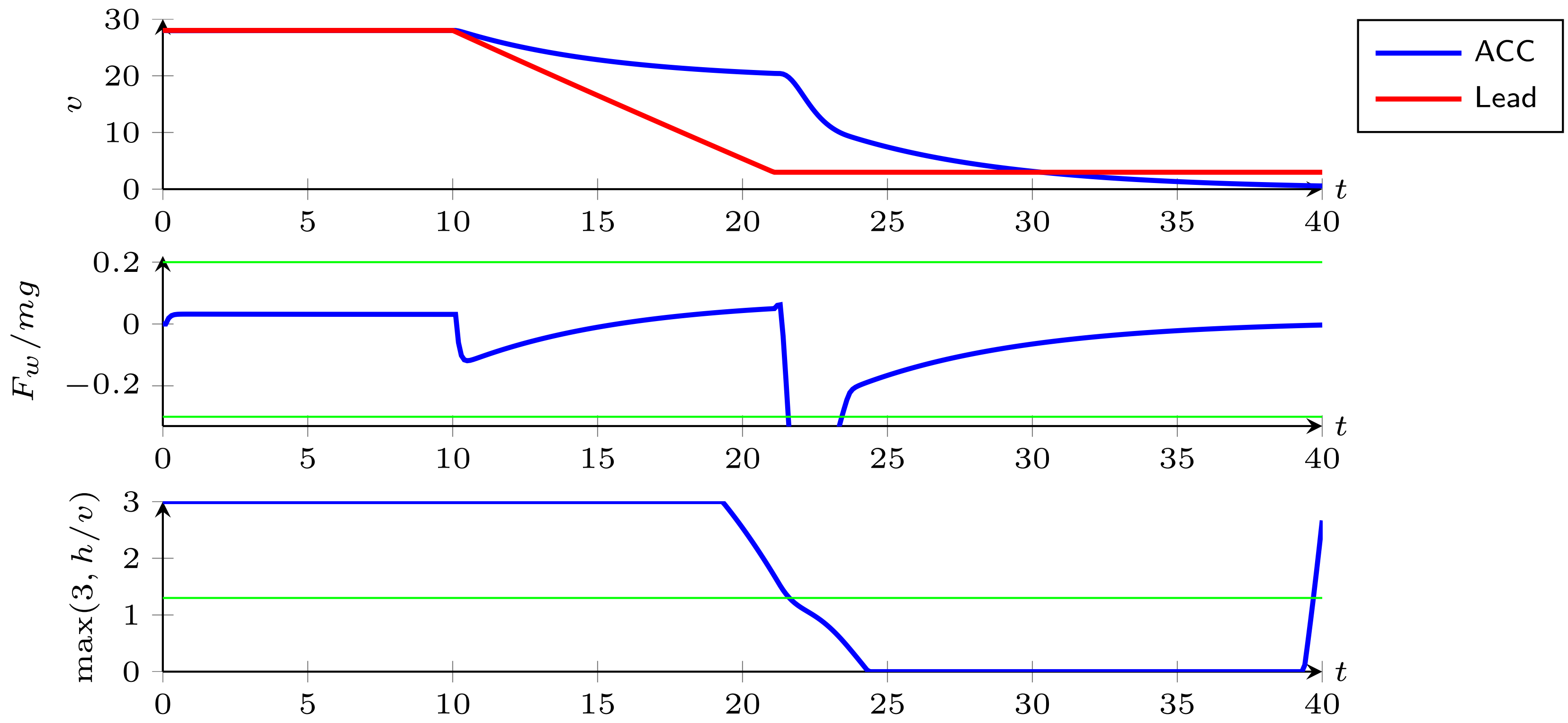
Are we robust enough with full (30dim state space) for non-linear vehicle dynamics?





Supervision example

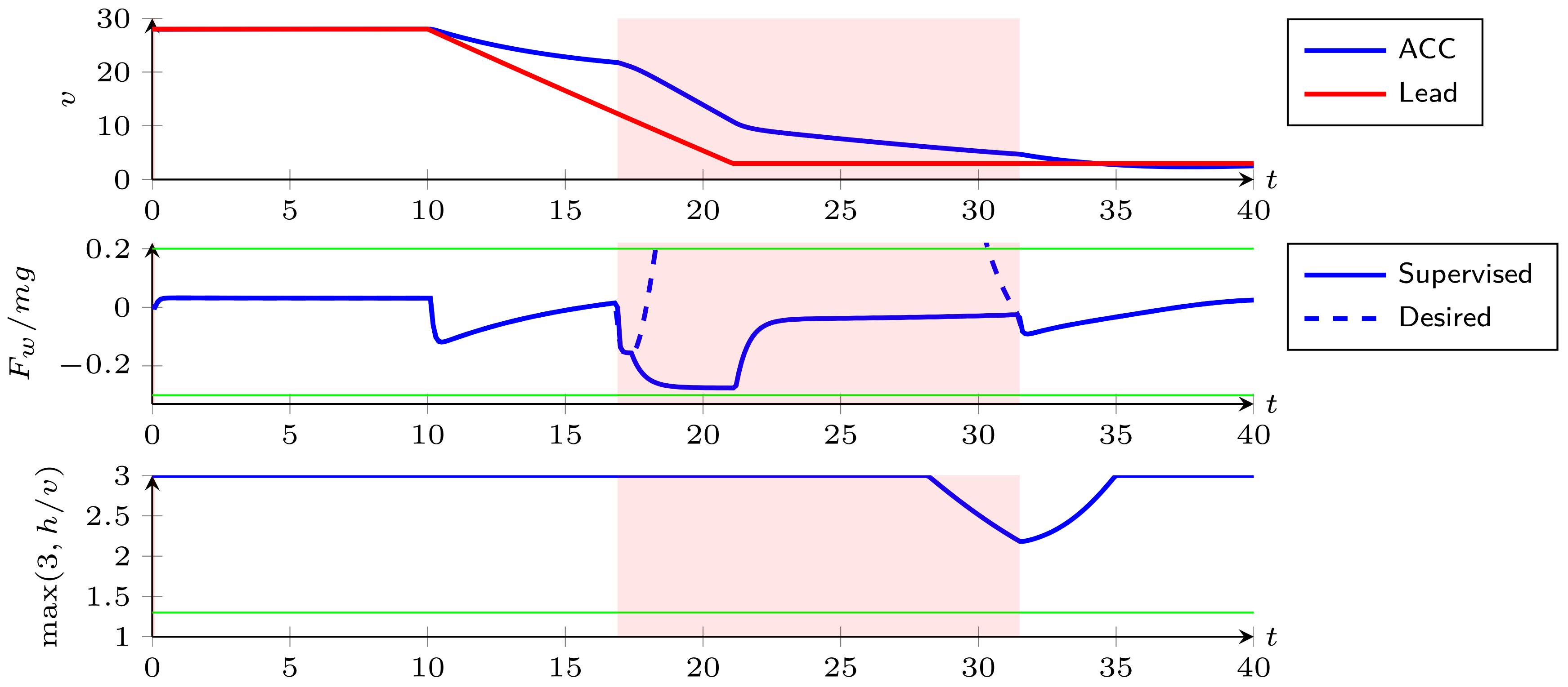
Mis-tuned controller from the literature



Controller has certain nice properties (stability, string stability) but causes a crash.

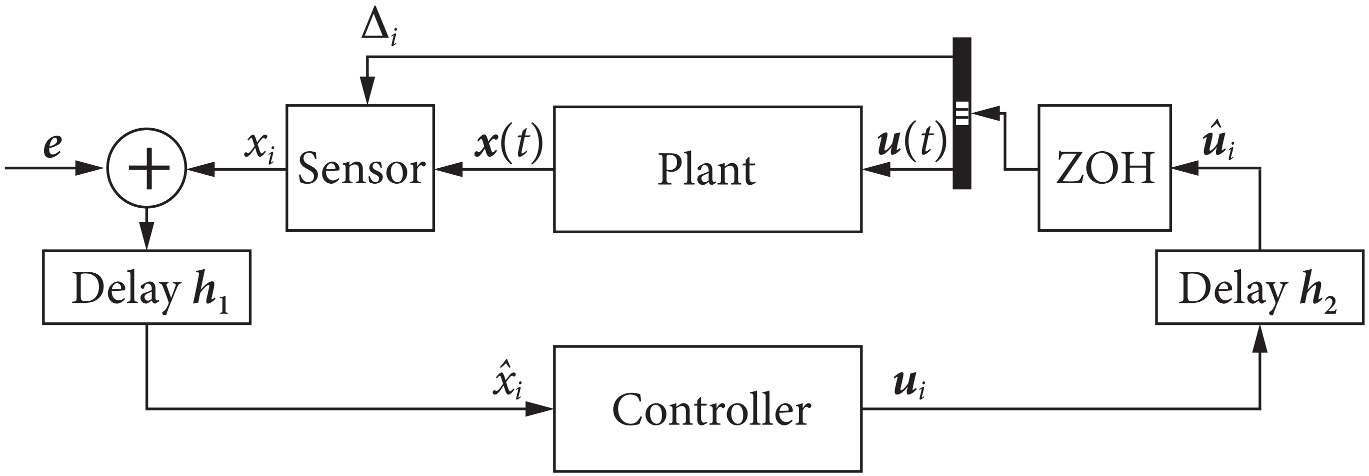
Supervision example

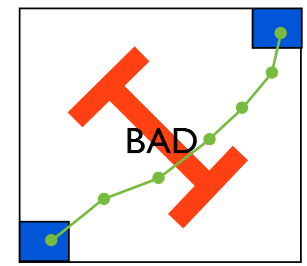
Supervised controller:



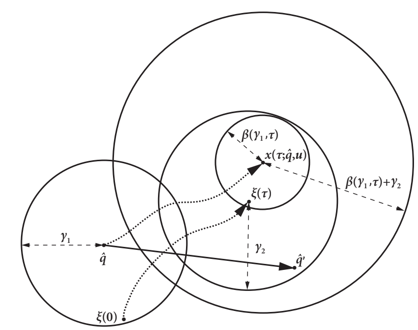
Original controller is used in non-red areas.

Can we formalize robustness?

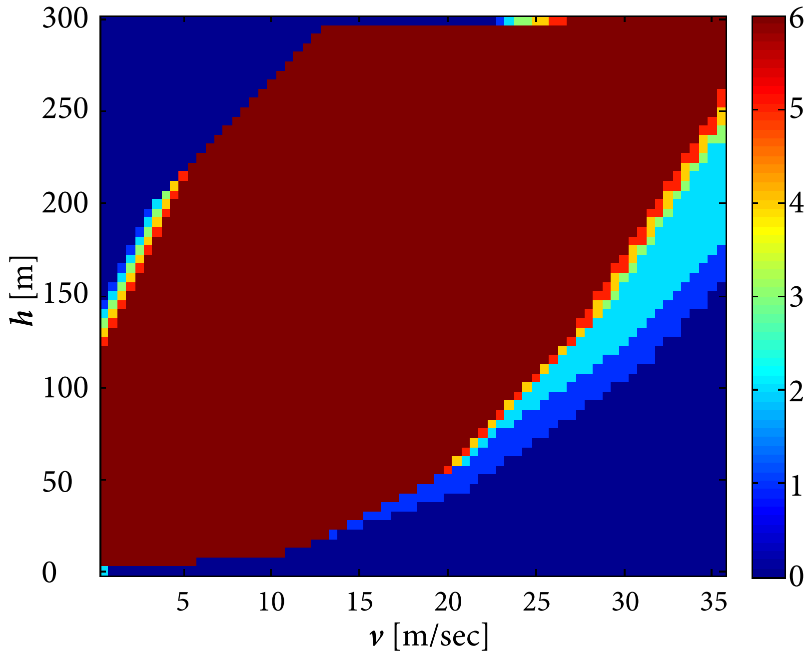


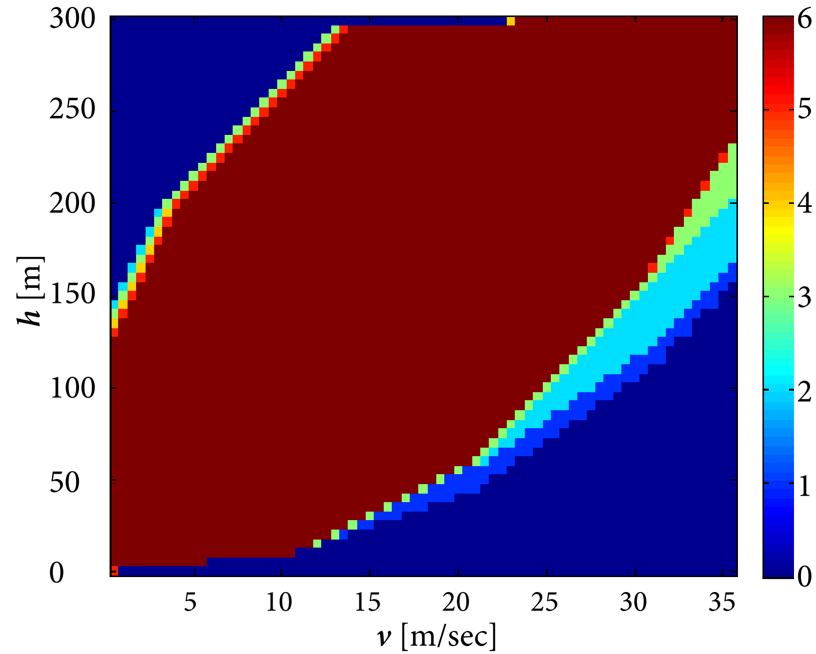


- Sources of imperfection
  - Most guarantees are on discrete-time behaviors (how about continuous-time)
  - Sensor, actuation, computation delays (jitter)
  - Delays, uncertainties in the model
  - Errors in measurements
- Idea is to introduce robustness margins: two additional balls are "enough"



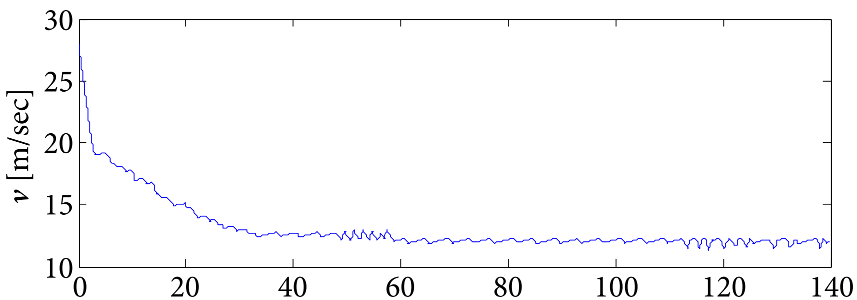
Robustness-performance trade-offs

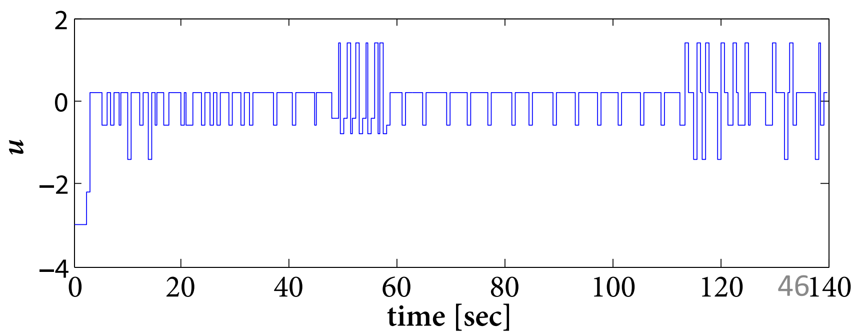




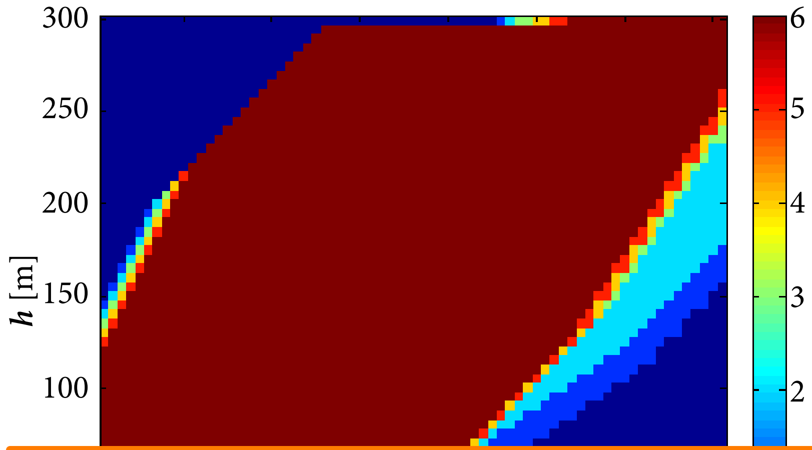
Safety domain as delay/jitter increases from 0-0.2 seconds

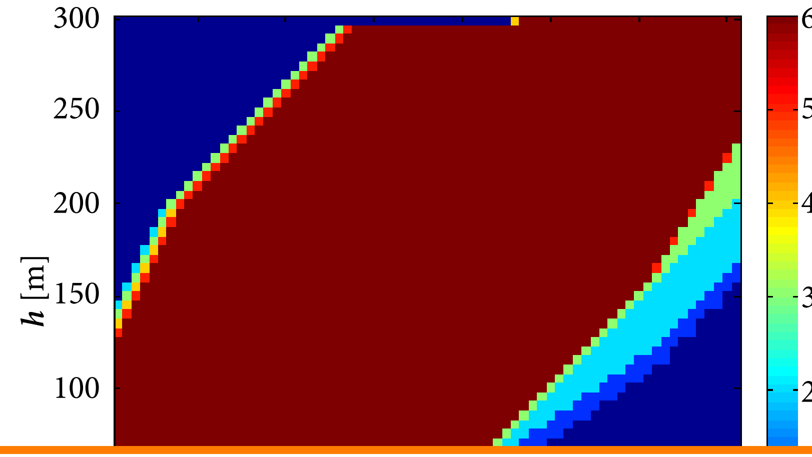
Safety domain as measurement error increases from 0-25cm





Robustness-performance trade-offs

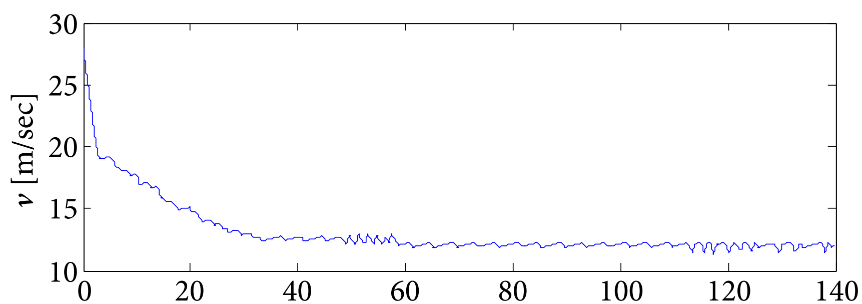


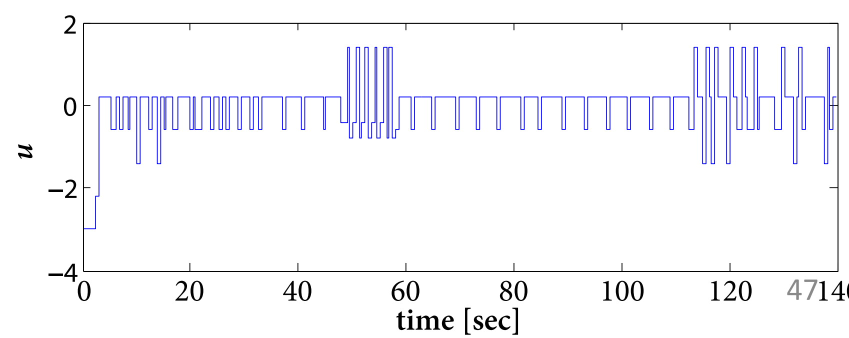


Can we prove non-existence of controllers?
Nilsson & Ozay, CDC 2014

Safety domain as delay/jitter increases from 0-0.2 seconds

Safety domain as measurement error increases from 0-25cm





Lane keeping

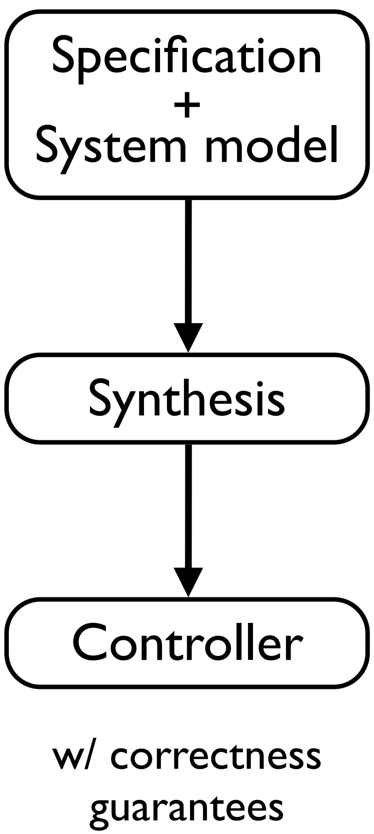
- Similar problem (just constrained reachability)

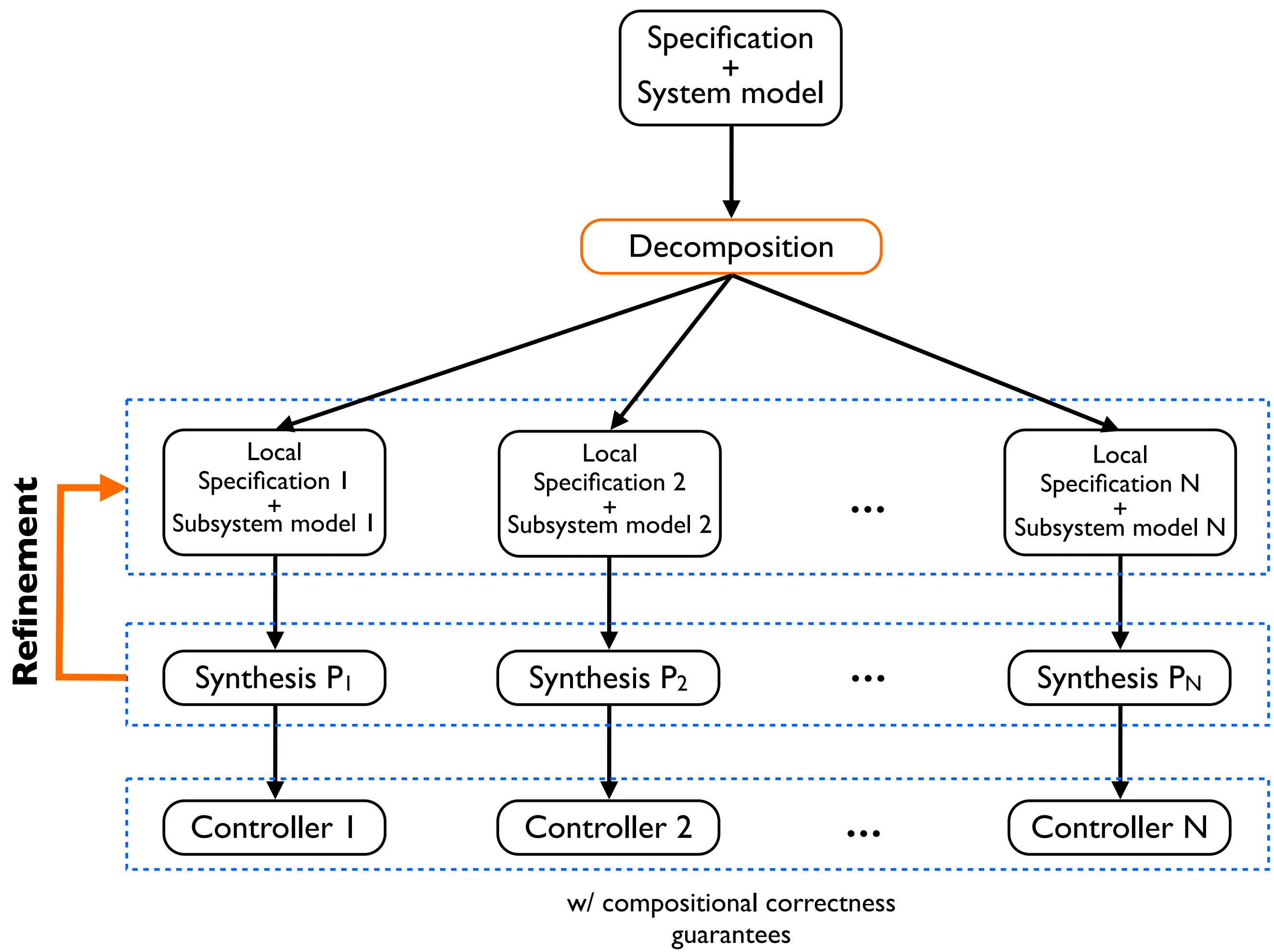
Would is still work when combined with ACC?

Future directions

Decompositions

Decomposition





Decomposition of dynamically coupled systems

$$\begin{bmatrix} x_1^+ \\ x_2^+ \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} B_1 & 0 \\ 0 & B_2 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} + \begin{bmatrix} E_1 & 0 \\ 0 & E_2 \end{bmatrix} \begin{bmatrix} d_1 \\ d_2 \end{bmatrix}$$

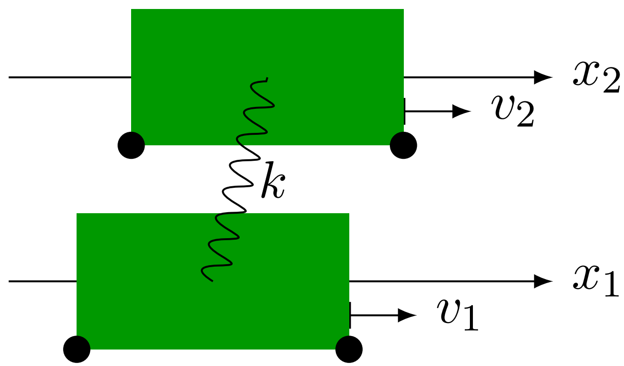
Find decompositions based on set-invariance.
Solve local problems in each invariant set.

Formal controller 1

Formal controller 2

- Natural decomposition if A_{12} and A_{21} are "small"
- Each subsystem need to be robust w.r.t. influence from other subsystems

Robot-UAV example

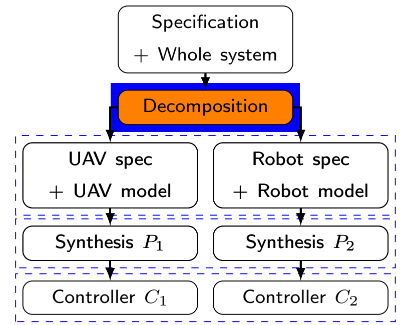


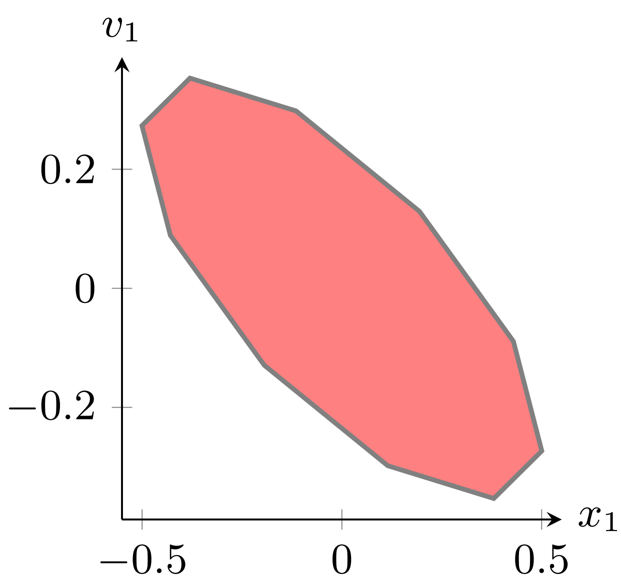
- 1D-robots connected by a spring
- Integrator dynamics

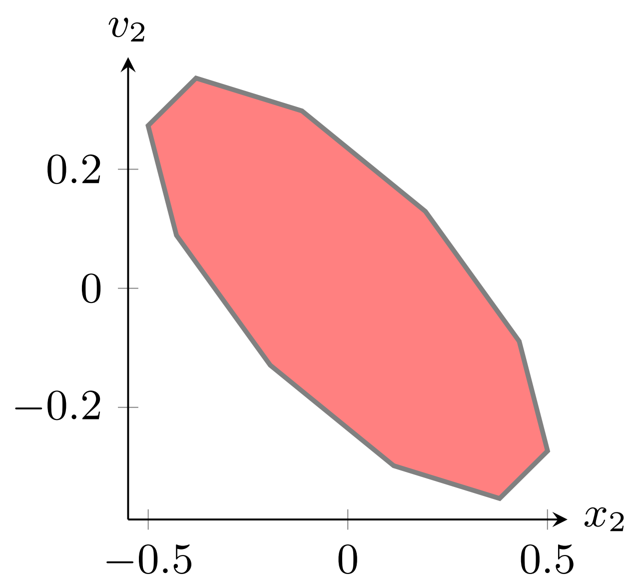
$$
\begin{aligned}
{x_1}^+ &= x_1 + v_1 \\
{v_1}^+ &= kx_1 + v_1 - kx_2 \\
{x_2}^+ &= + x_2 + v_2 \\
{v_2}^+ &= -kx_1 + kx_2 + v_2
\end{aligned}
$$

Robot-UAV example

1. Find invariant sets

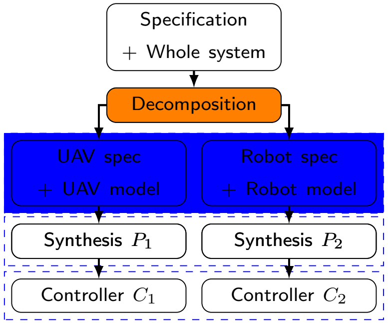




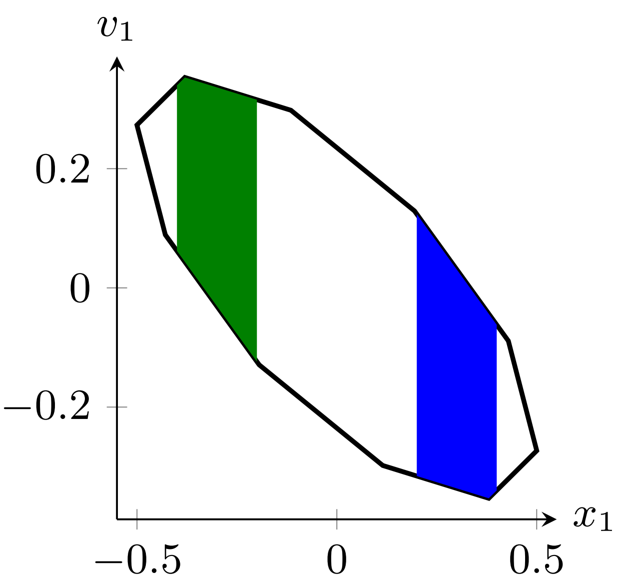


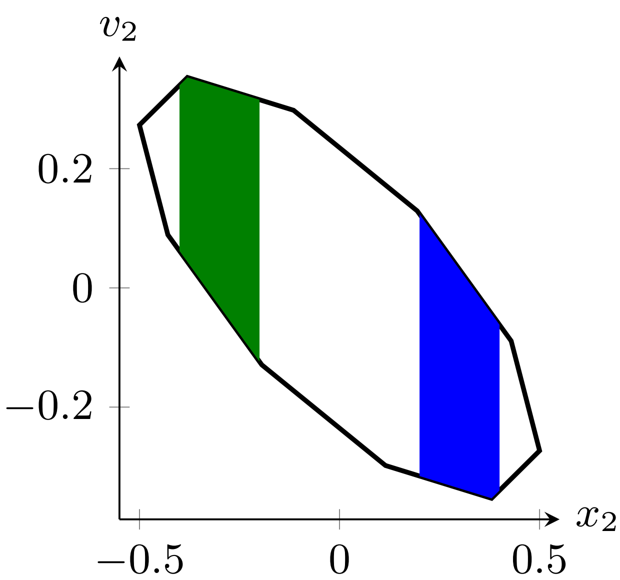
Robot-UAV example

1. Find invariant sets
2. With any method, do local synthesis for



$$\bigwedge_{i=1}^{2} \Box\Diamond \left(x_i \in {\color{green}\blacksquare}\right) \wedge \Box\Diamond \left(x_i \in {\color{blue}\blacksquare}\right)$$

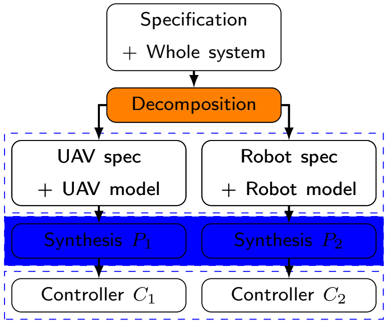


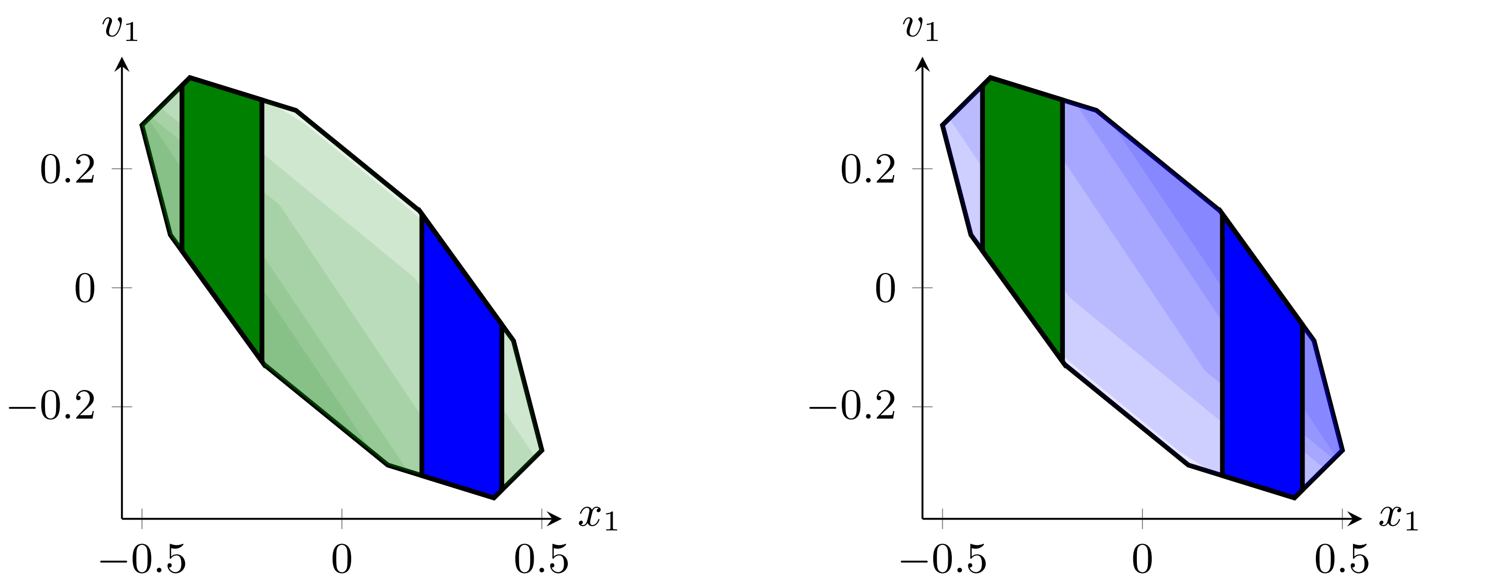


Robot-UAV example

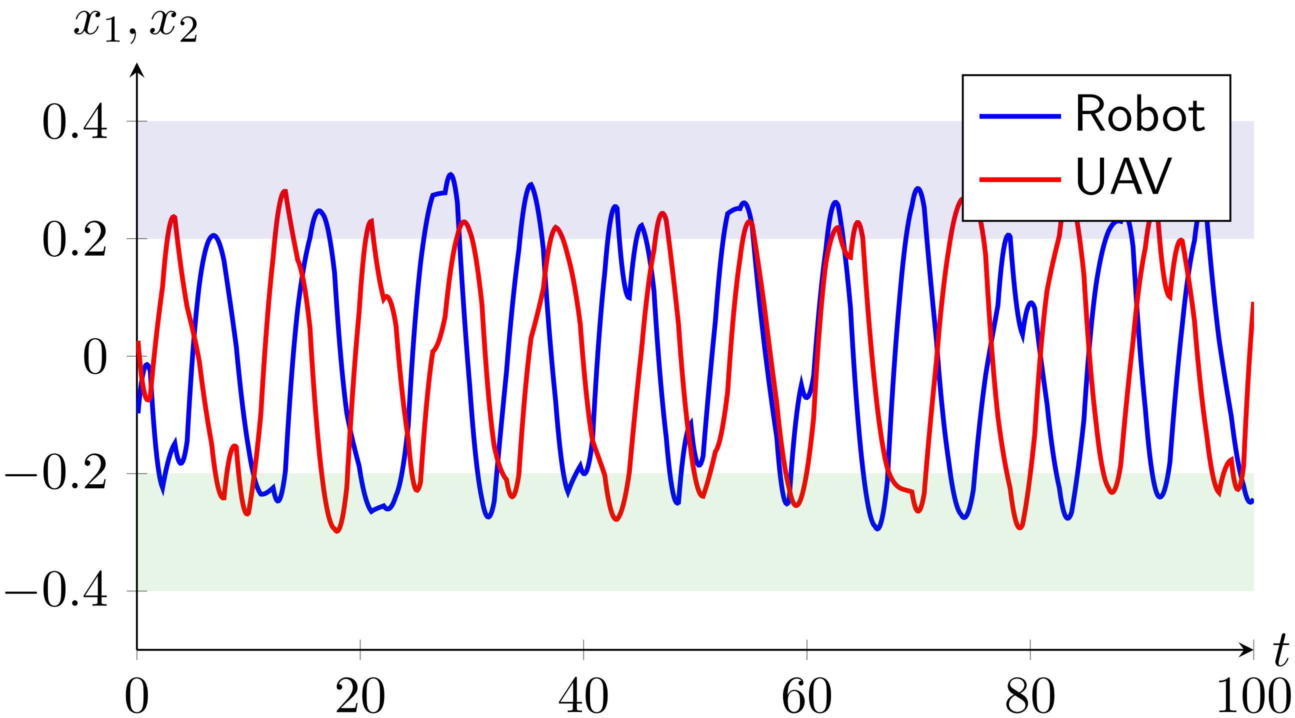
1. Find invariant sets
2. With any method, do local synthesis for

$$\bigwedge_{i=1}^{2} \Box\Diamond\left(x_i \in \text{[green]}\right) \wedge \Box\Diamond\left(x_i \in \text{[blue]}\right)$$





Robot-UAV example



- Both visit green and blue areas infinitely often
- Solved two 2-dimensional problems instead of one 4-dimensional

Summary & Current Directions

- Goal: go from **sensor** to **information** to **action** in a rigorous way with correctness guarantees.
- **Directions:**

How to formally combine different functionality?

- Compositional protocols: reduces complexity, enables local implementations, improves design modularity

Scalability, robustness

Other applications

Lane keeping – similar reach stay while avoiding problem

More info @ dynamiccps.org
@ web.eecs.umich.edu/~necmiye/