Synthesis-guided Adversarial Example Generation for Gray-box Autonomous Systems with Sensing Imperfections

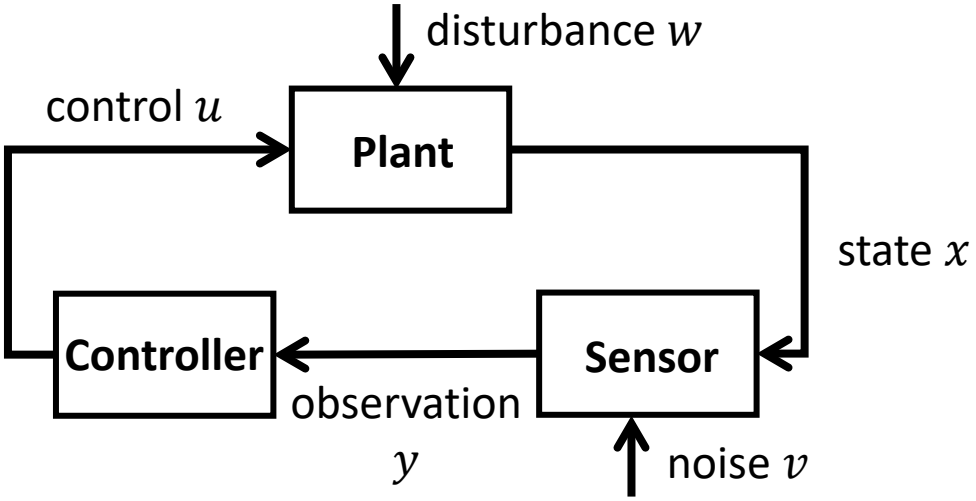
Liren Yang

4th March 2021

MICHIGAN ENGINEERING
UNIVERSITY of MICHIGAN ■ COLLEGE of ENGINEERING

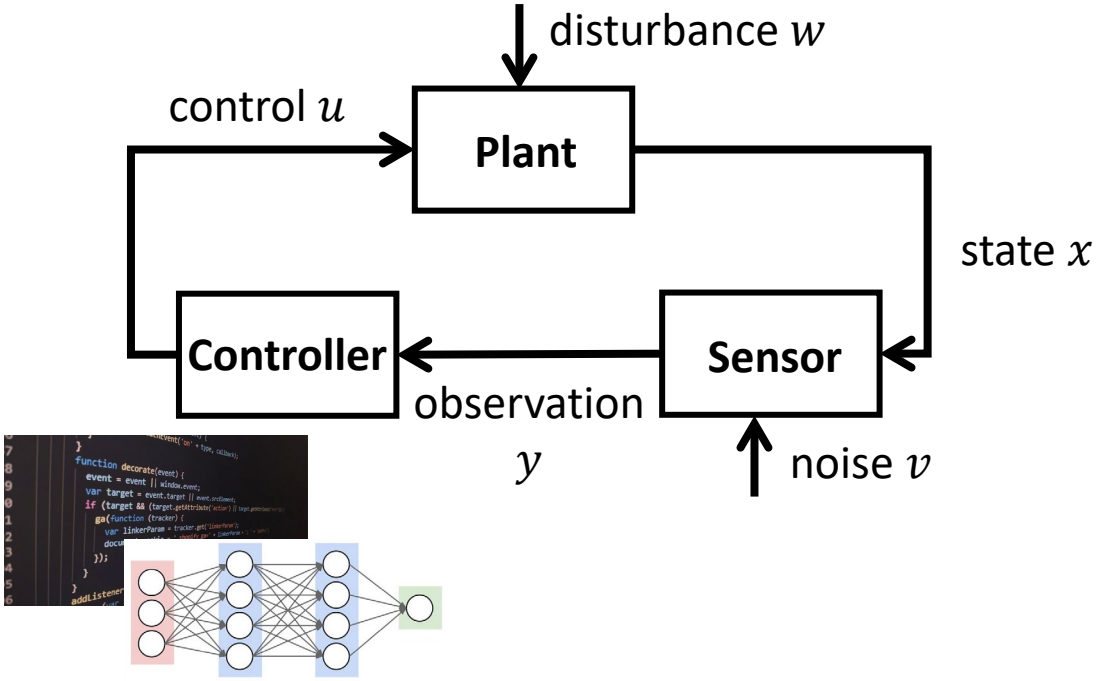
Collins Aerospace
Office of Naval Research ONR Science & Technology

Autonomous Control Systems



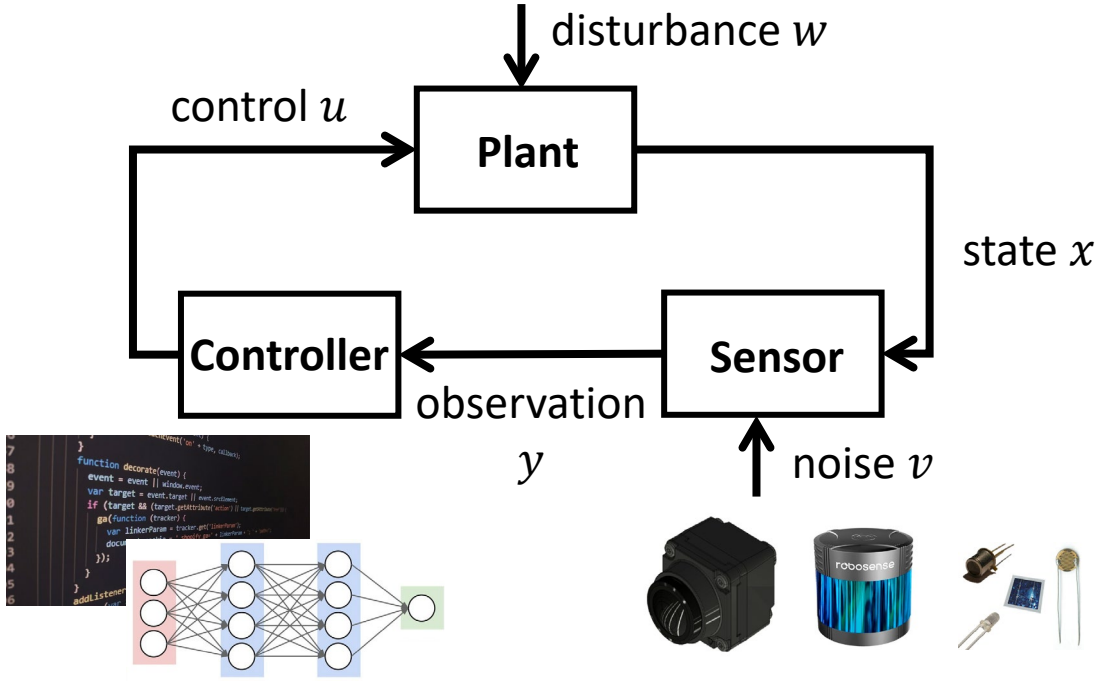
Autonomous Control Systems

- Complex components
  - Controller: millions of lines of code (learning, optimization…)



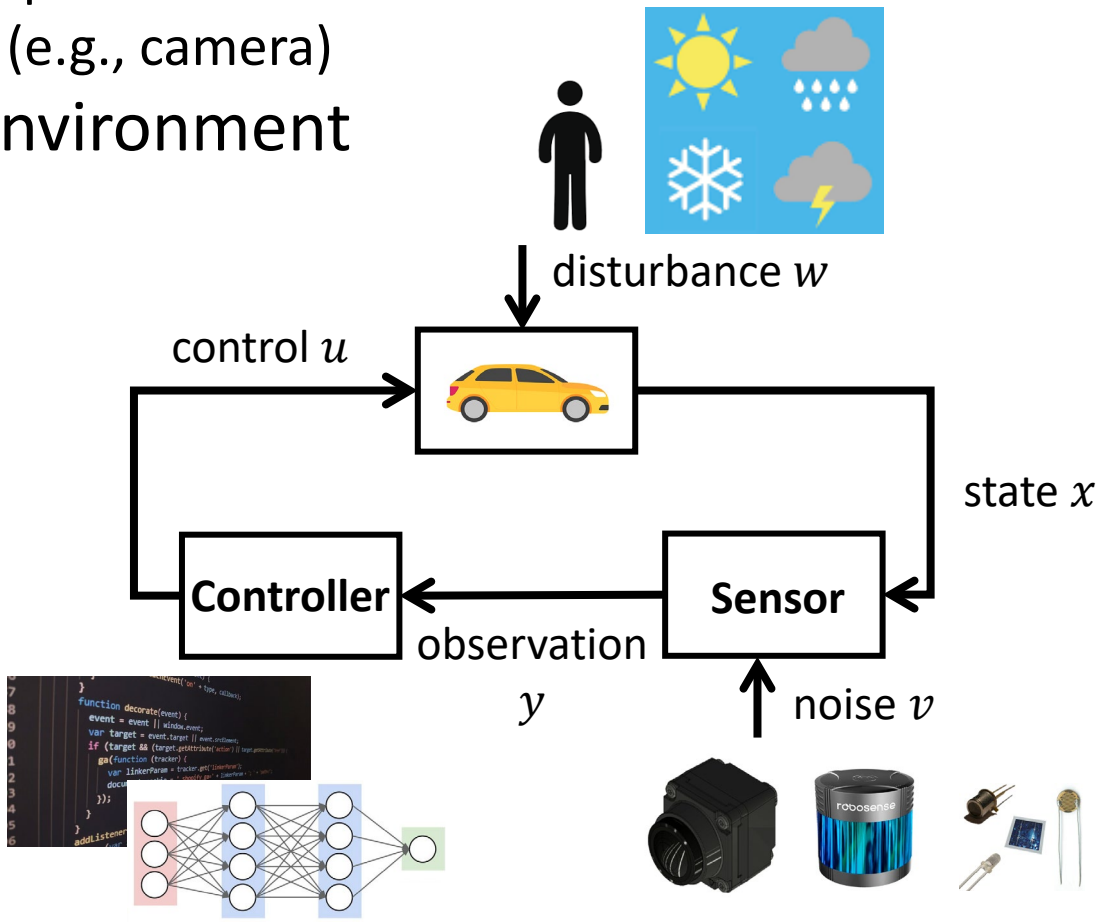
Autonomous Control Systems

- Complex components
  - Controller: millions of lines of code (learning, optimization…)
  - Sensor: imperfect & hard to formalize (e.g., camera)



Autonomous Control Systems

- Complex components
  - Controller: millions of lines of code (learning, optimization…)
  - Sensor: imperfect & hard to formalize (e.g., camera)
- Uncertain environment



Autonomous Control Systems

- Complex components
  - Controller: millions of lines of code (learning, optimization…)
  - Sensor: imperfect & hard to formalize (e.g., camera)
- Uncertain environment
- Safety-critical requirement

disturbance w

control u

Plant

Controller

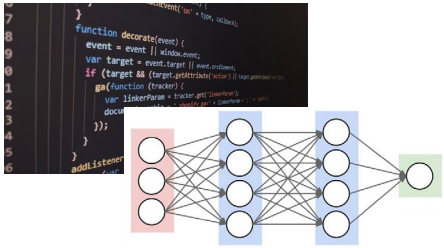
Sensor

observation y

noise v

We want

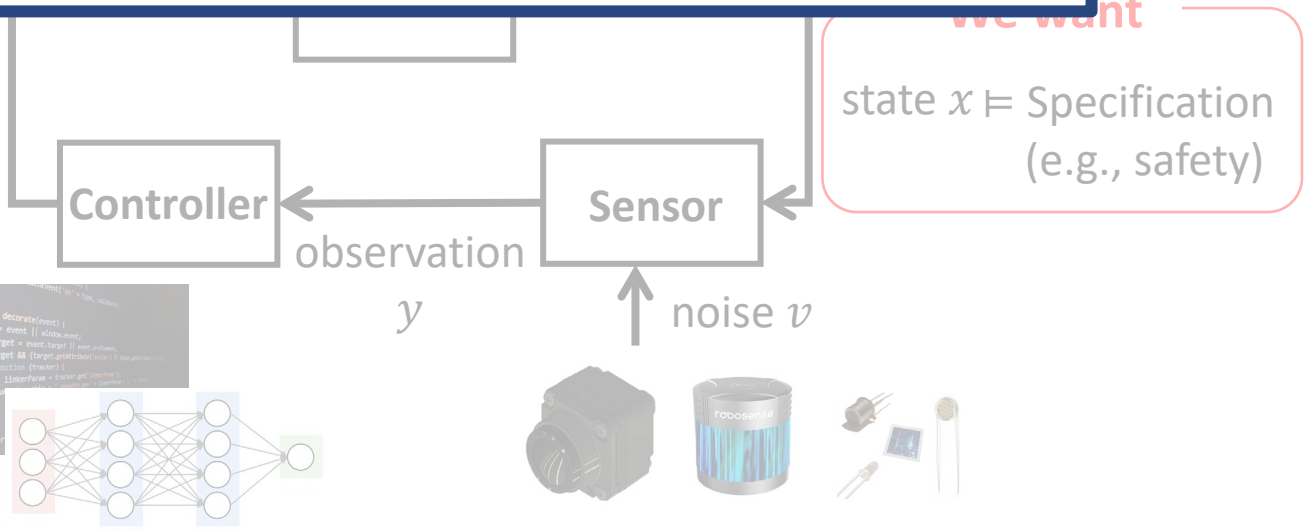
state $x \models$ Specification (e.g., safety)



Autonomous Control Systems

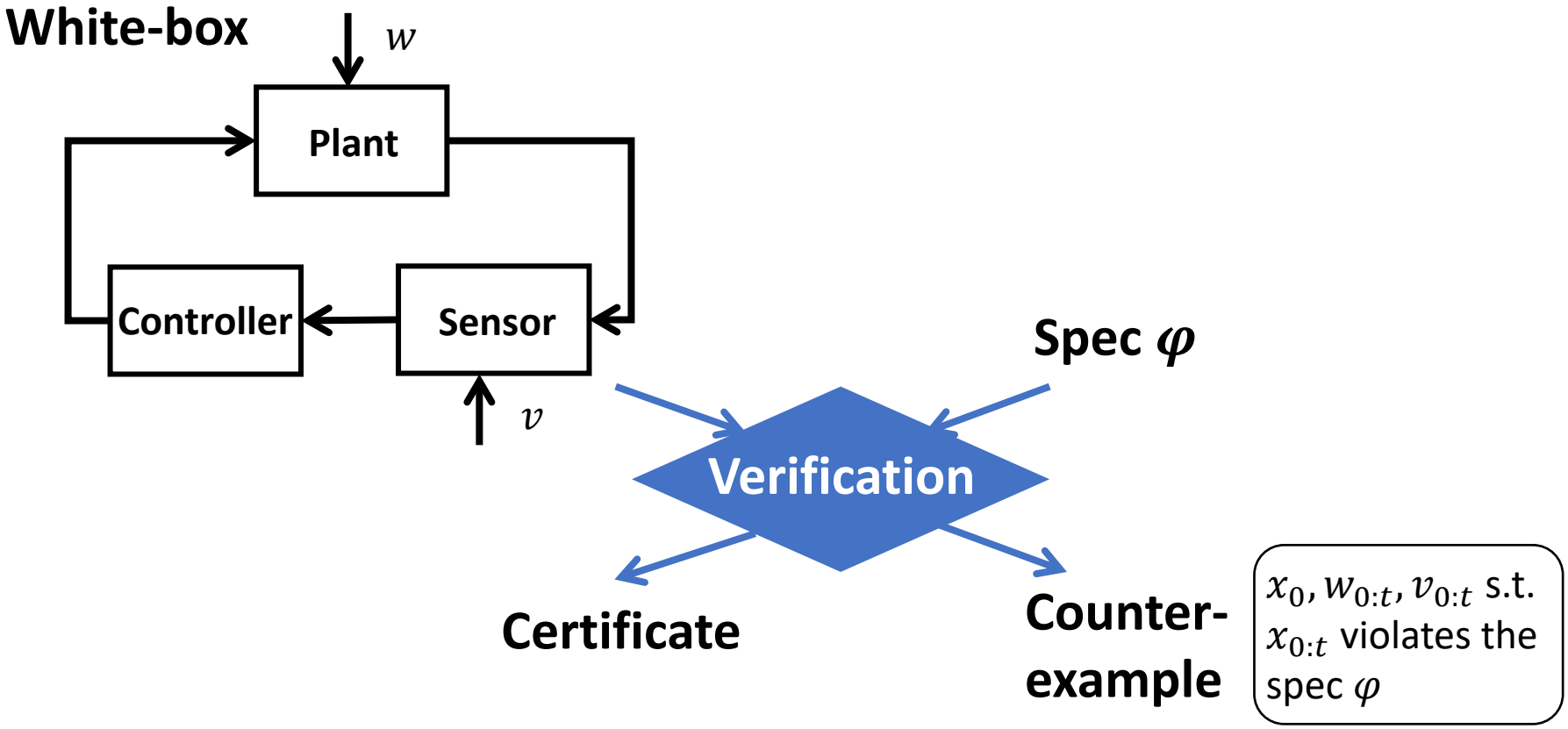
- Complex components
  - Controller: millions of lines of code (learning, optimization…)
  - Sensor: imperfect & hard to formalize (e.g., camera)
- Uncertain environment
- Safety-

Question: can we prove an autonomous system to be safe under all circumstances?



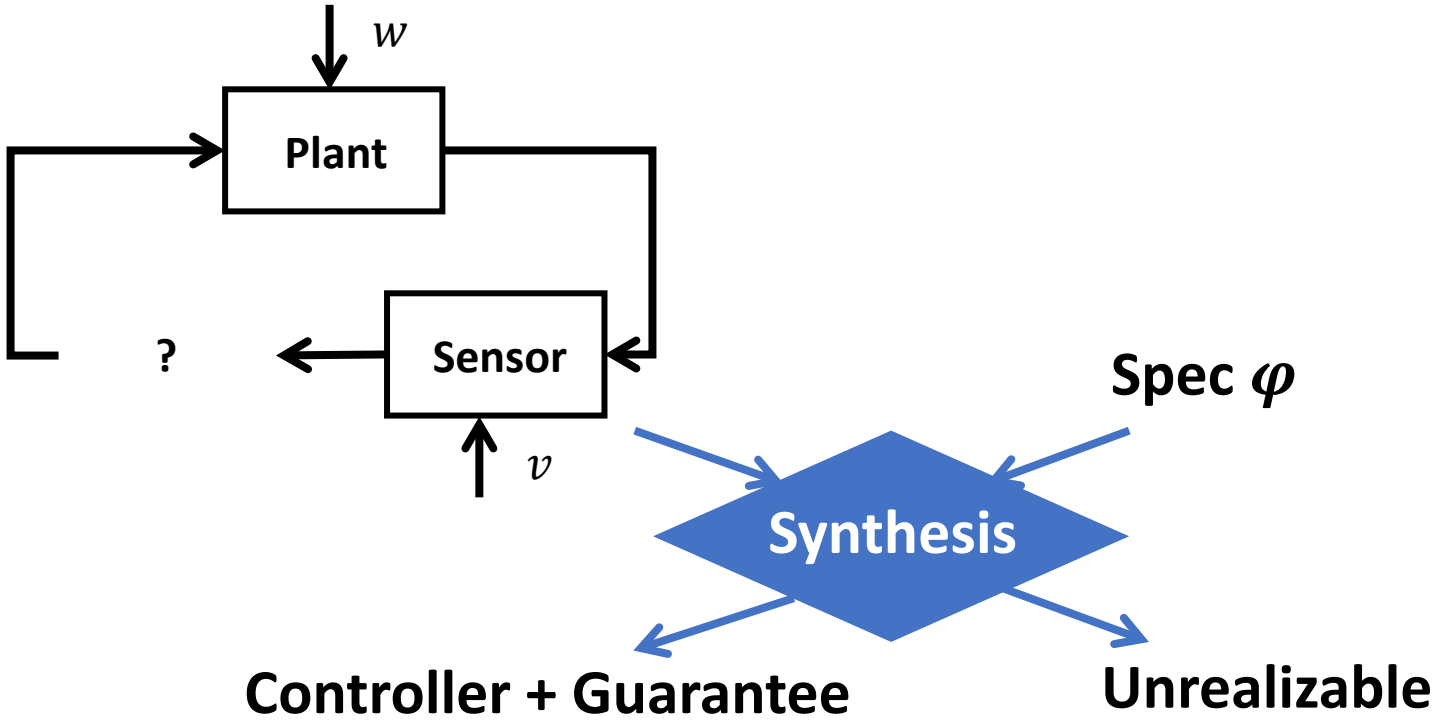
Verification

- Pro: strong conclusion
- Con: computationally challenging



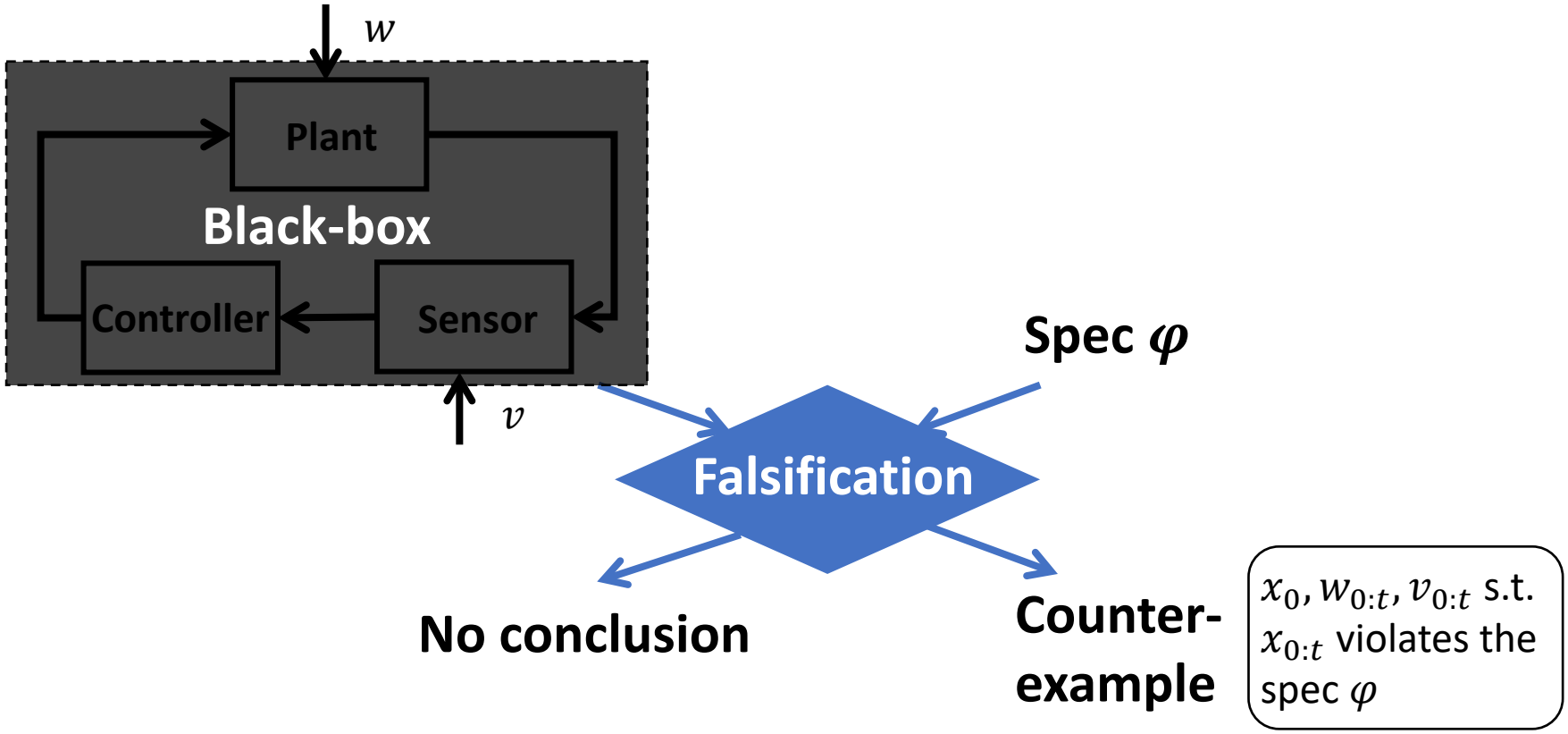
Alternative 1: Synthesis

- Pro: avoid the complexity of controller, "explain" fundamental limits (impossibility)
- Con: even more difficult, difficulty ↑ with sensing error
  difficulty ↑↑ with vision



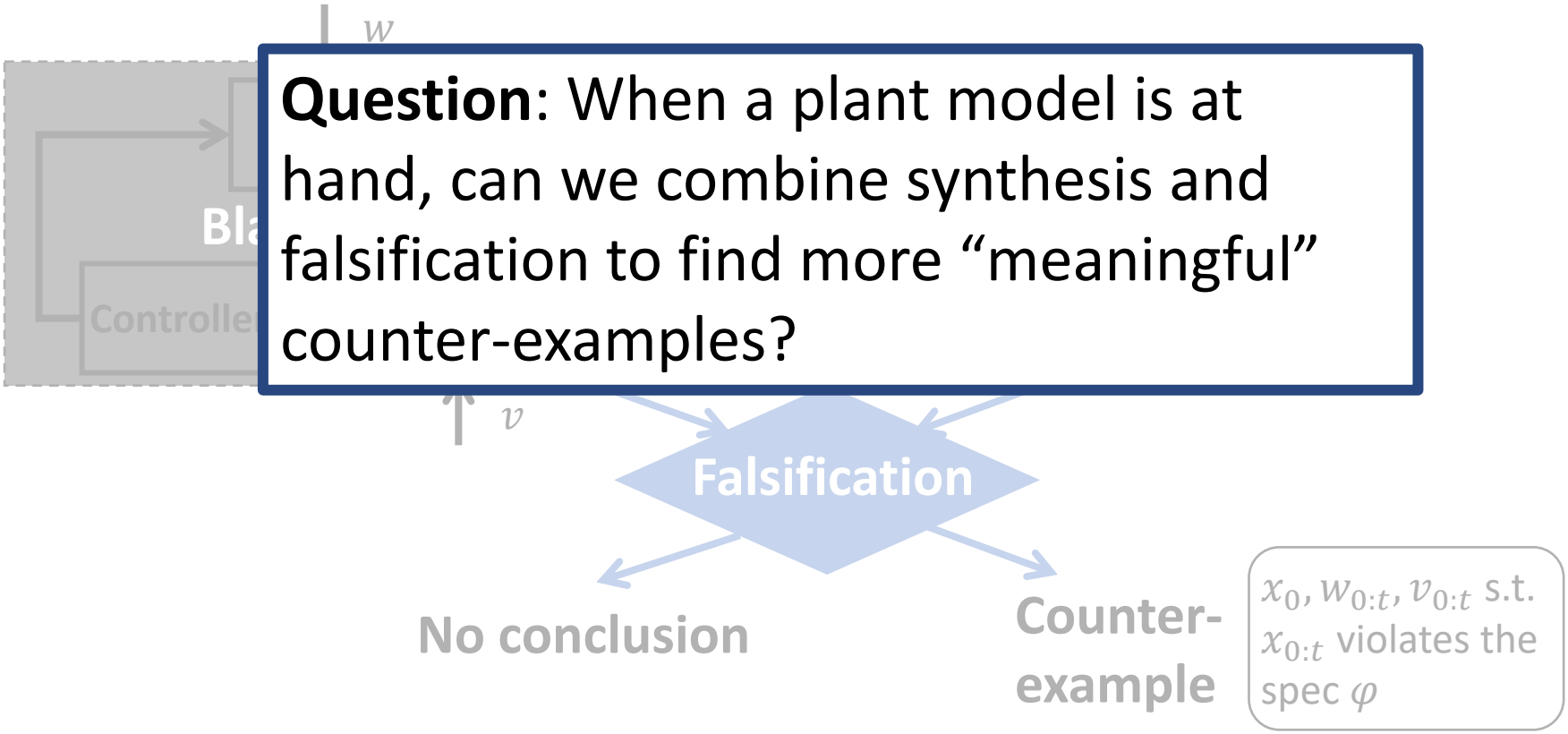
Alternative 2: Falsification

- Pro: computationally cheaper
- Con: weaker conclusion, lack of interpretability



Alternative 2: Falsification

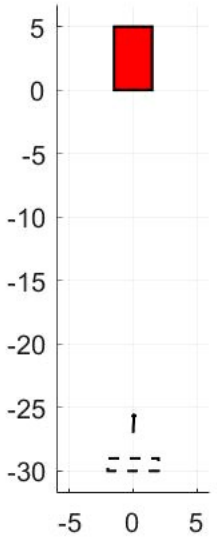
- Pro: computationally cheaper
- Con: weaker conclusion, lack of interpretability



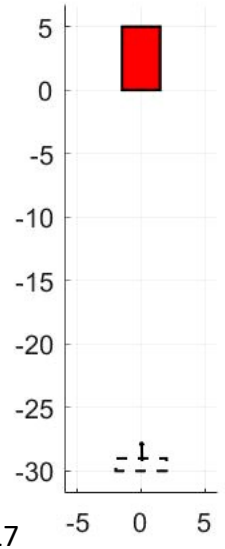
A Typical Adversarial Example

- Plant: integrator-like dynamics
- Sensor: a camera, simulated with Carla[1]
- End-to-end controller: a feedforward neural network
- Spec: avoid the obstacle (red) in the front, without decelerating

Under random noise



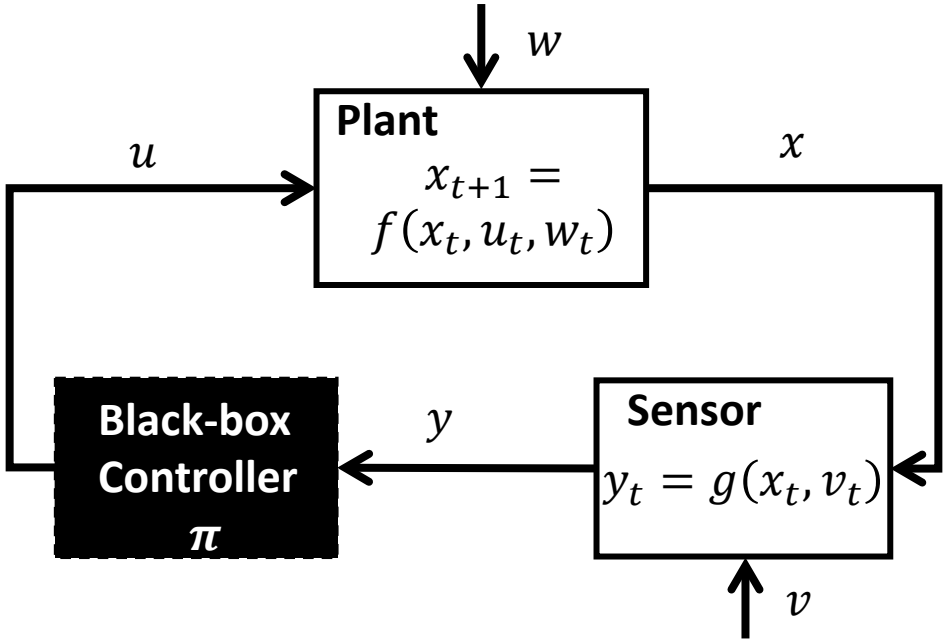
Under adversarial noise



1. A. Dosovitskiy et al. “CARLA: An open urban driving simulator”. 2017

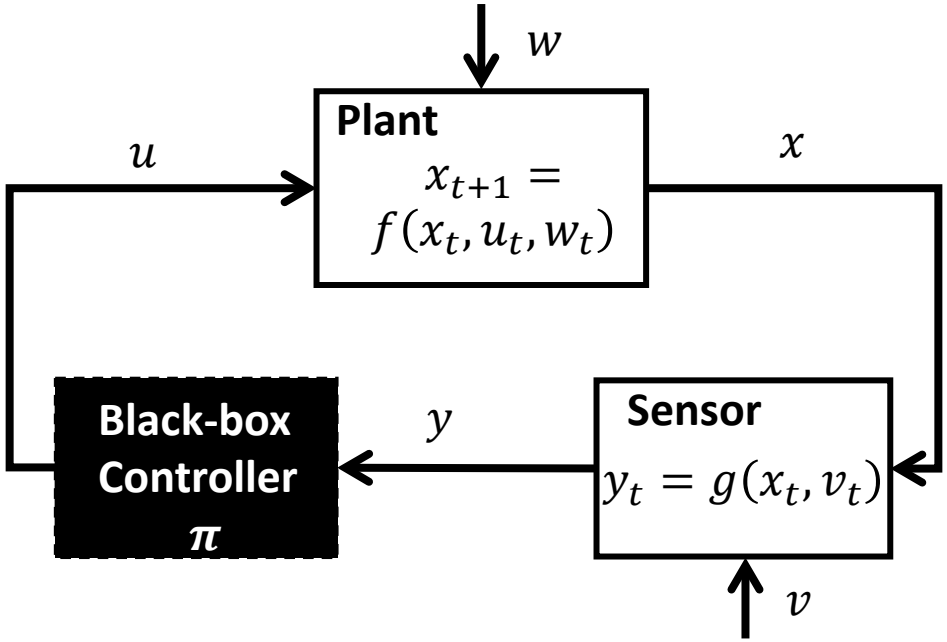
Our Gray-box Setting:

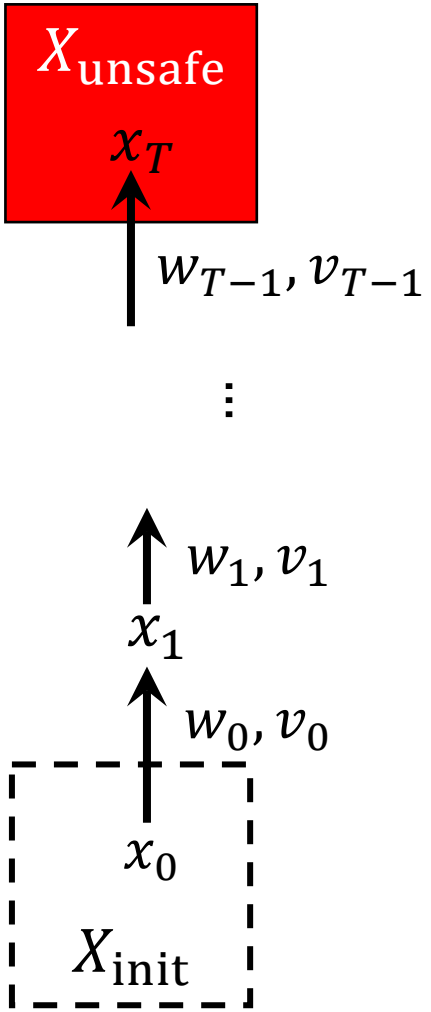
- White-box components:
  - Plant model (f)
  - Sensor model (g), to be relaxed later
- Black-box component
  - Controller (π): static, i.e., no memory (e.g., rule-based, MPC, feedforward neural network)
  - Can be "queried": given y, we get $u = \pi(y)$



Problem Statement

- Given:
  - A gray-box system
  - An unsafe set X_{unsafe}, and an initial set X_{init}
- Find one adversarial example:
  - A trajectory $x_0, x_1, \ldots, x_T$
  - External inputs $w_0, w_1, \ldots, w_{T-1}, v_0, v_1, \ldots, v_{T-1}$
  - $x_0 \in X_{\text{init}}$ and $x_T \in X_{\text{unsafe}}$





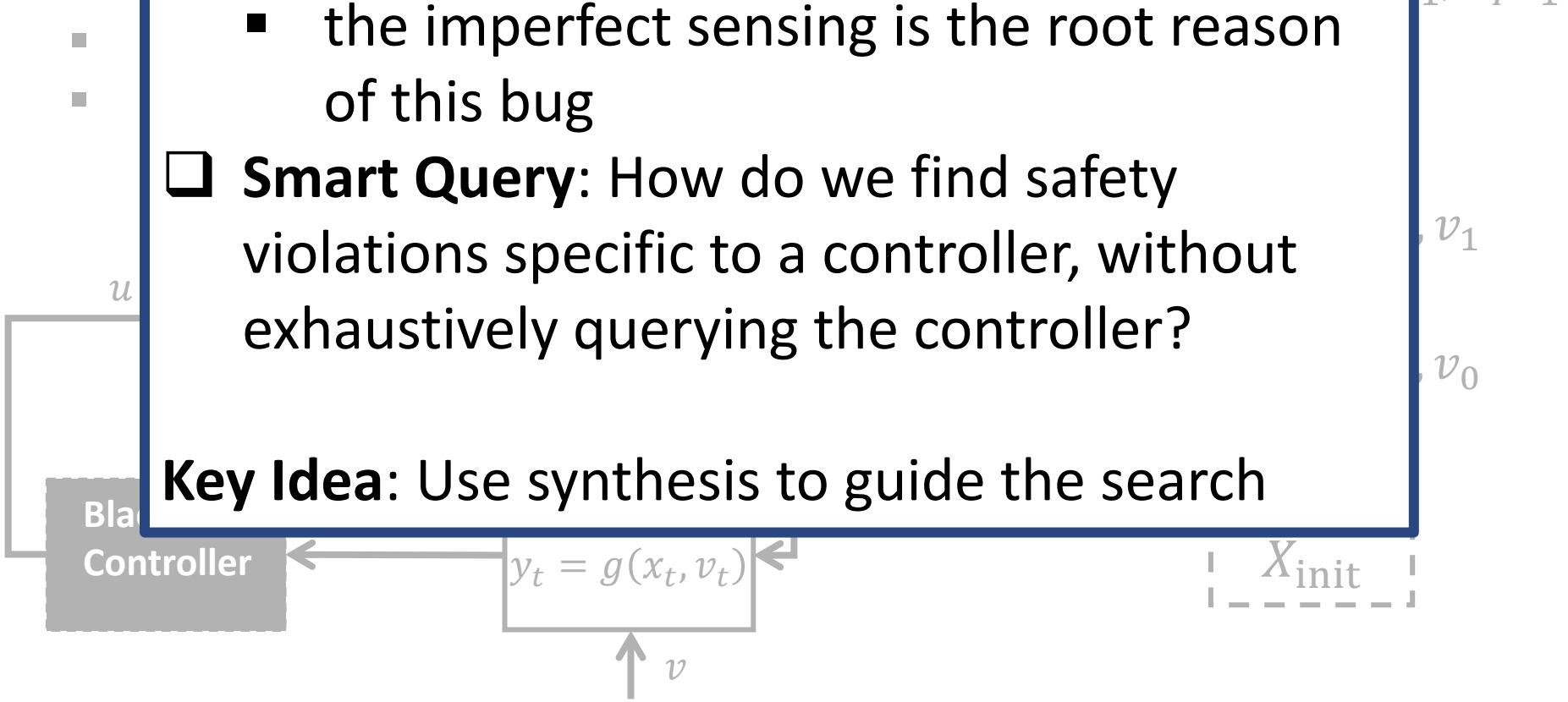
Problem Statement

- Given:
  -
  -
- Fin
  -
  -
  -

Questions:

- **Interpretability**: Can we say anything like:
  - the bug is specific to the given controller,
  - the imperfect sensing is the root reason of this bug
- **Smart Query**: How do we find safety violations specific to a controller, without exhaustively querying the controller?

Key Idea: Use synthesis to guide the search

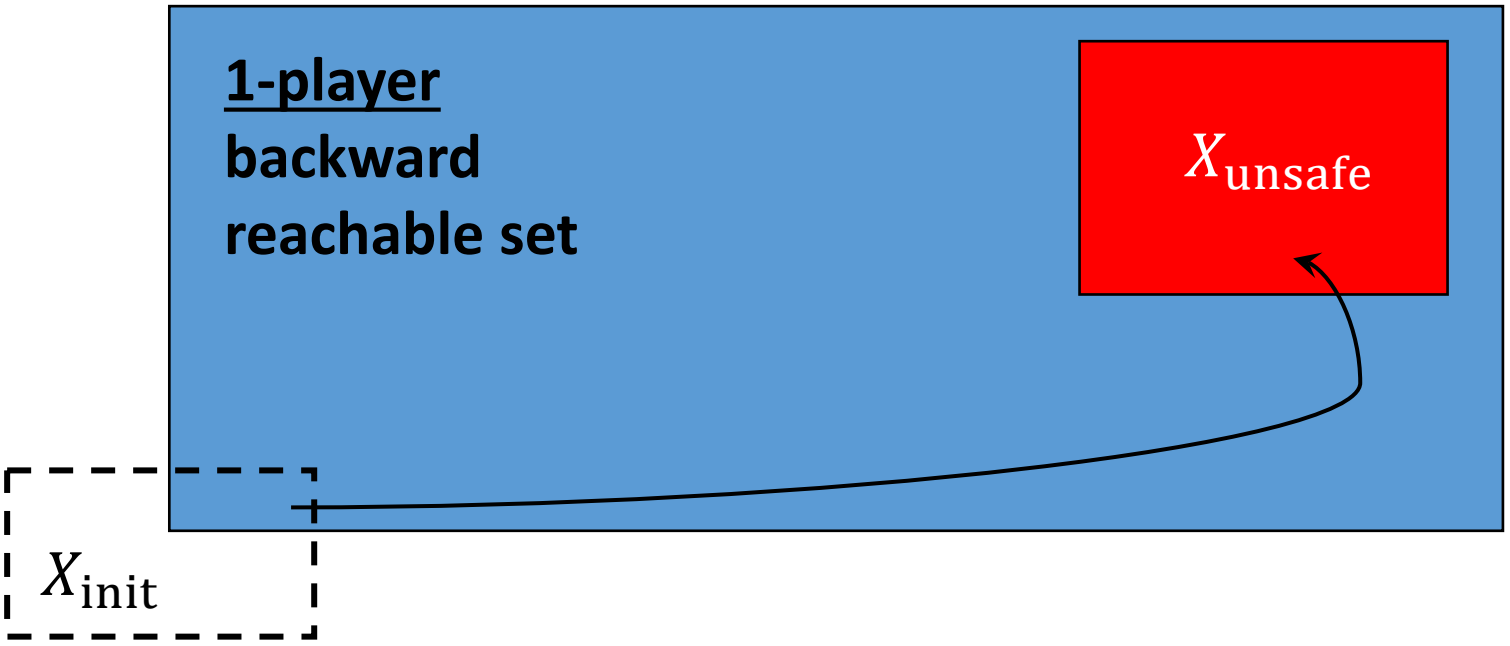


Key Idea

Verification: compute 1-player backward reachable set $= \bigcup X_k$

$$X_0 = X_{\text{unsafe}}$$
$$X_{k+1} = \mathbf{Pre}(X_k) := X_k \cup \{x | \exists w, v: \underbrace{f(x, \pi(x, v), w)}_{} \in X_k\}$$

Closed-loop dynamics: complex due to π, even if f itself is simple (precisely why verification is hard)



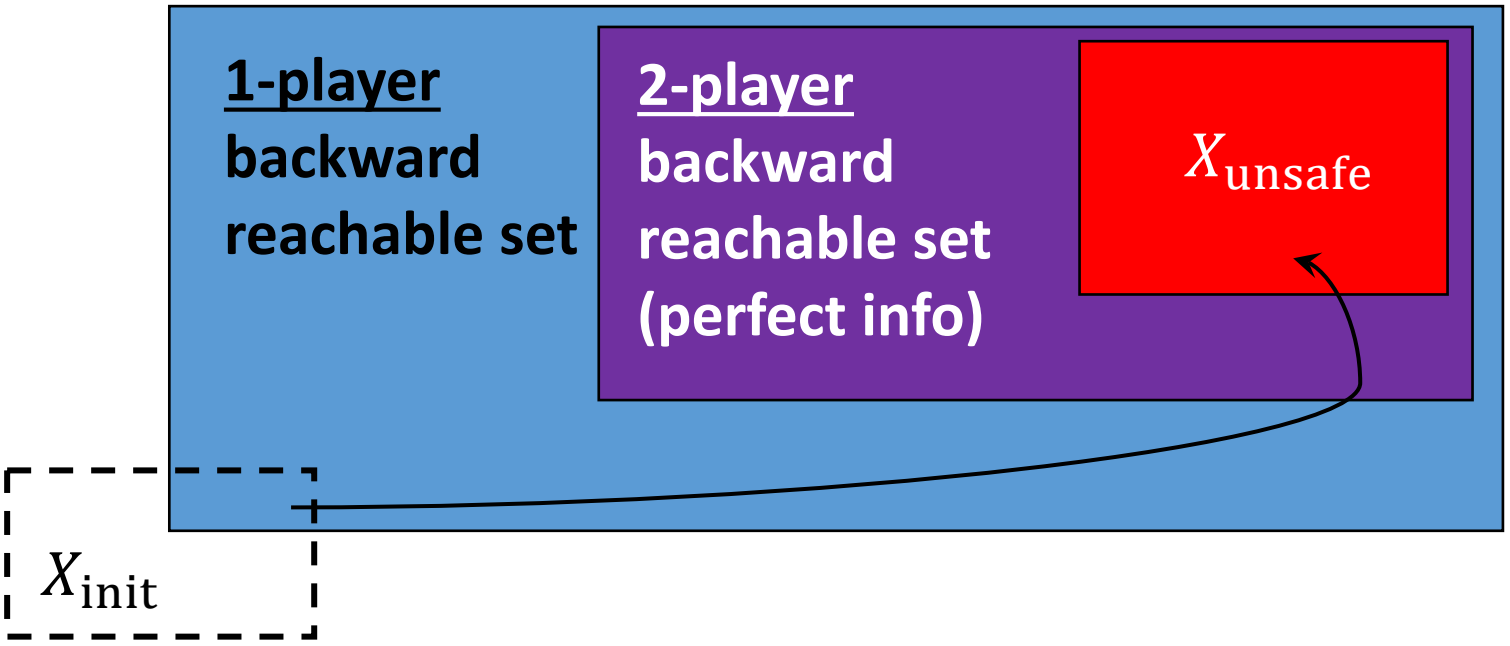
Key Idea

Synthesis: compute 2-player backward reachable set $= \bigcup \hat{X}_k$

$$\hat{X}_0 = X_{\text{unsafe}}$$
$$\hat{X}_{k+1} = \mathbf{CPre}(\hat{X}_k) := \hat{X}_k \cup \{x | \forall u: \exists w: f(x,u,w) \in \hat{X}_k\}$$

Open-loop dynamics: simple, independent of π

- The adversarial examples are trivial (generic)
- Noise v is not essential for violation

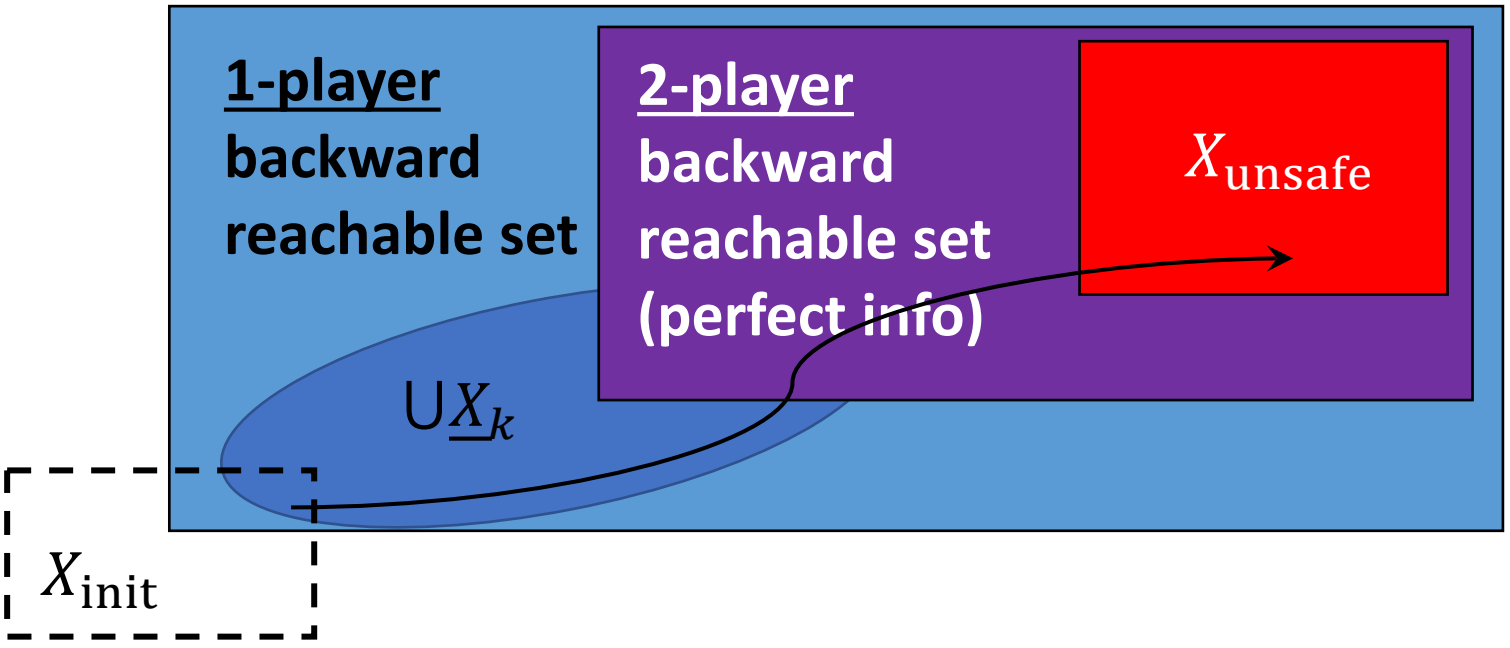


Key Idea

Synthesis guided Falsification: compute $\cup \underline{X}_k$, where

1. $\underline{X}_0 \subseteq$ 2−player backward reachable set
2. $Y_{k+1} = \mathbf{CPre}_y(\underline{X}_k) := \{y | \forall u: \exists x, v: y = g(x, v), f(x, u, w) \in \underline{X}_k\}$
3. $y_{k+1} \in Y_{k+1}$, $u_{k+1} = \pi(y_{k+1})$,
4. $\underline{X}_{k+1} = \mathbf{Pre}(\underline{X}_k | y_{k+1}) := \{x | \exists w, v: y_{k+1} = g(x, v), f(x, u_{k+1}, w) \in \underline{X}_k\}$

independent of π!

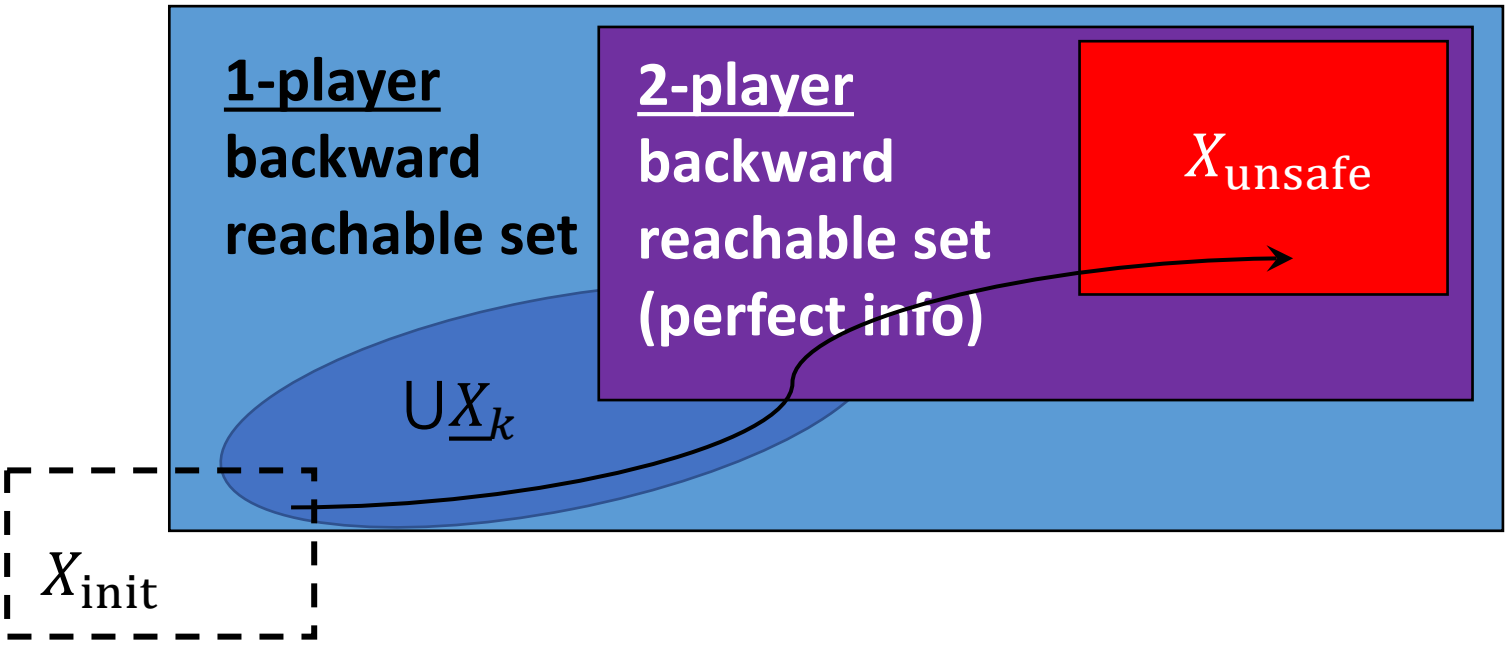


Key Idea

Synthesis guided Falsification: compute $\cup \underline{X}_k$, where

1. $\underline{X}_0 \subseteq$ 2−player backward reachable set
2. $Y_{k+1} = \mathbf{CPre}_y(\underline{X}_k)$: where it is tricker to satisfy the spec
3. query at the tricky place
4. $\underline{X}_{k+1} = \mathbf{Pre}(\underline{X}_k | y_{k+1})$ 1-player backward reachability based on the query result

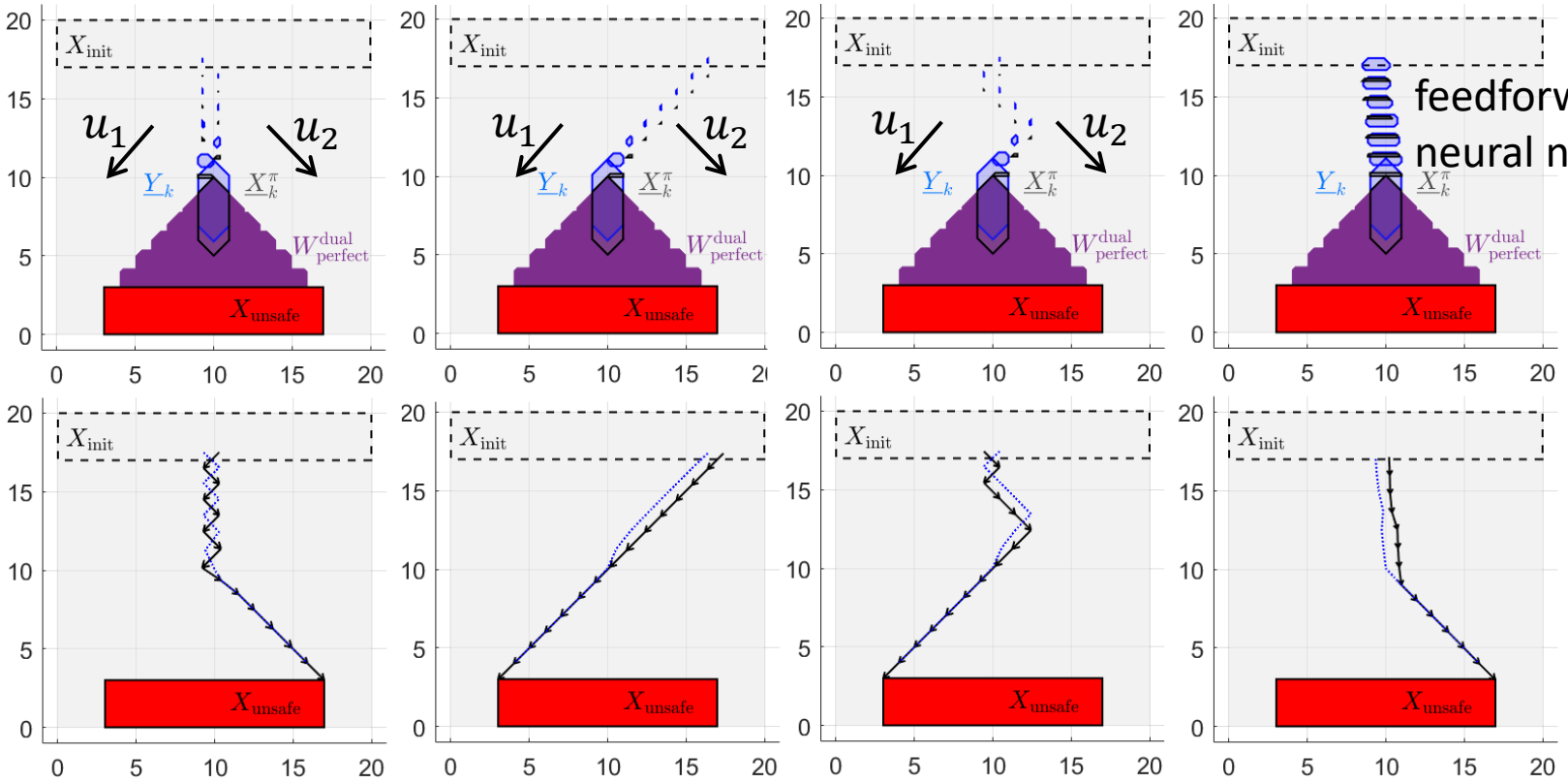
independent of π!



A Toy Example

- We find controller-specific violations using the same algorithm

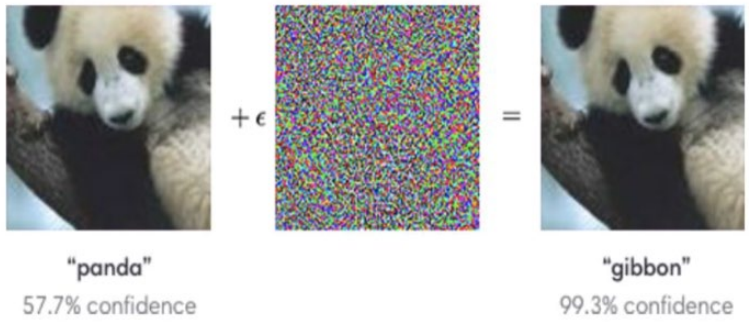
- Violations occur at "decision boundaries"



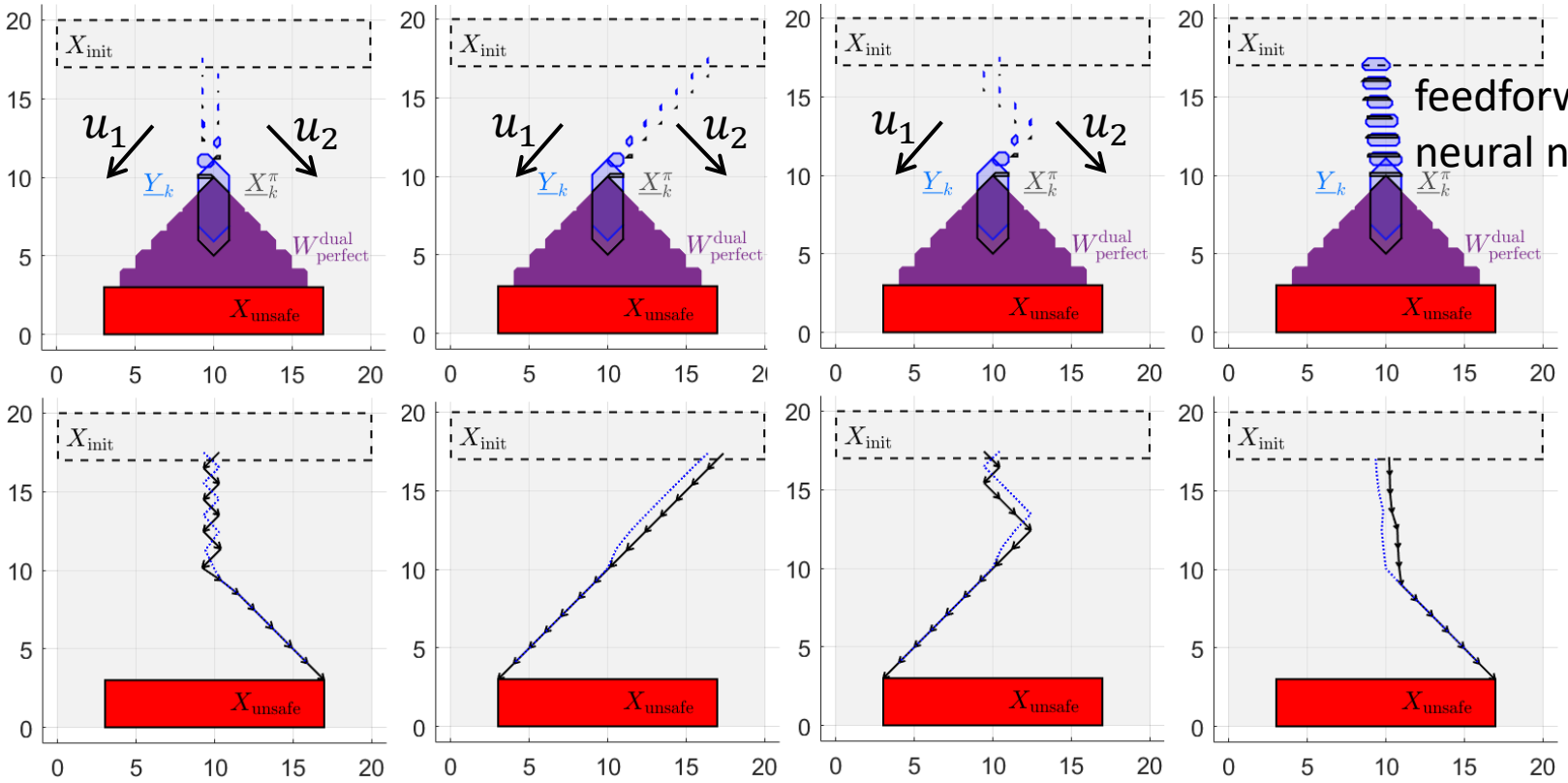
A Toy Example

- We find controller-specific violations using the same algorithm

- Violations occur at "decision boundaries"



Adversarial examples occur at decision boundaries in classification

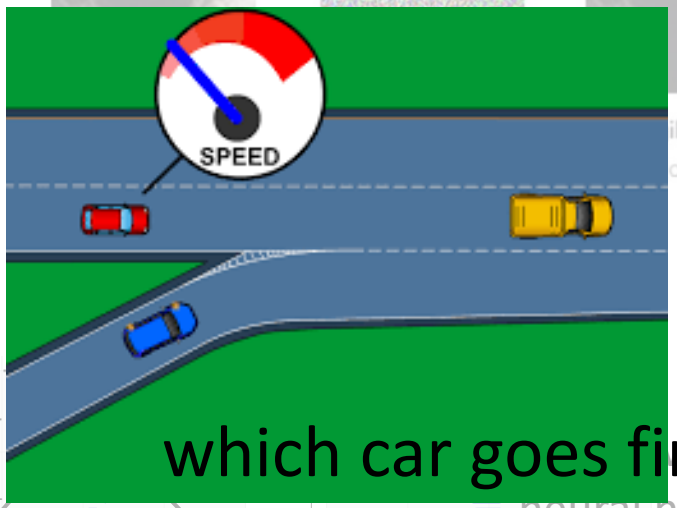


feedforward neural network

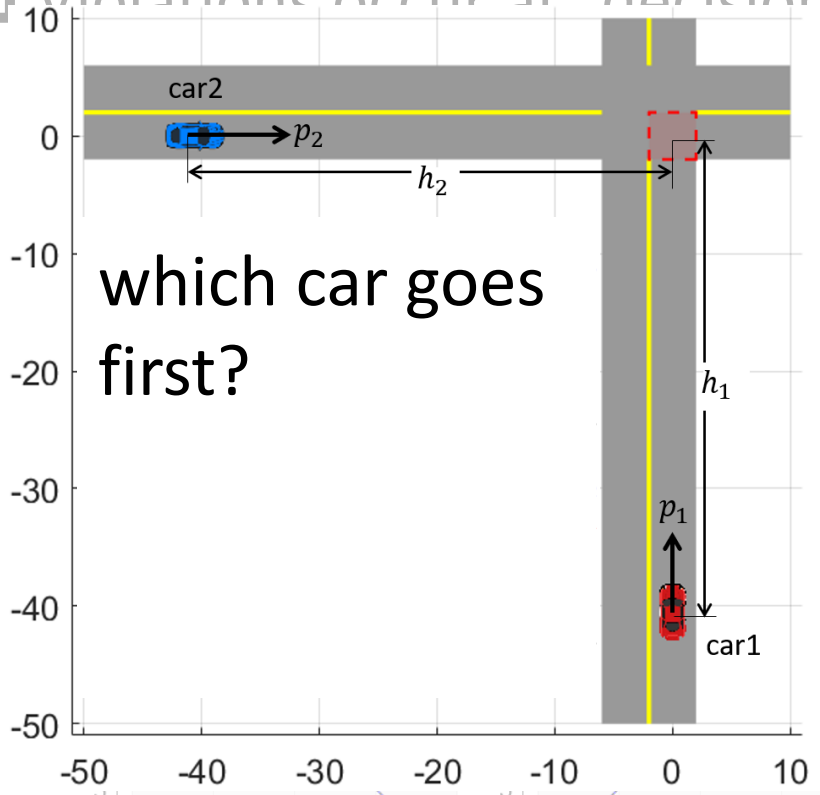
A Toy Example

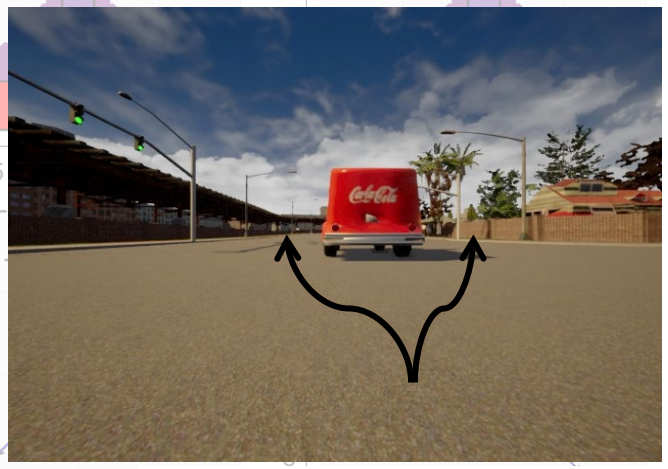
- We find controller-specific violations using the same algorithm
- Violations occur at "decision boundaries"

Decision boundaries are everywhere



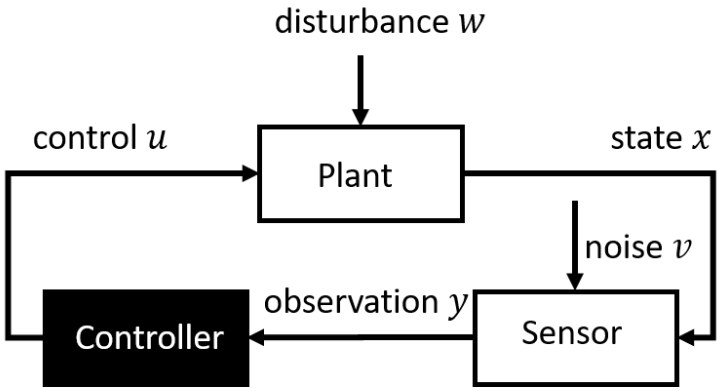
which car goes first?



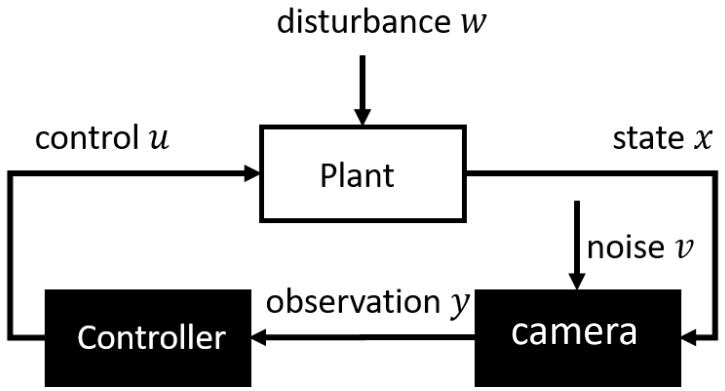


left or right?

Vision-in-the-loop Systems



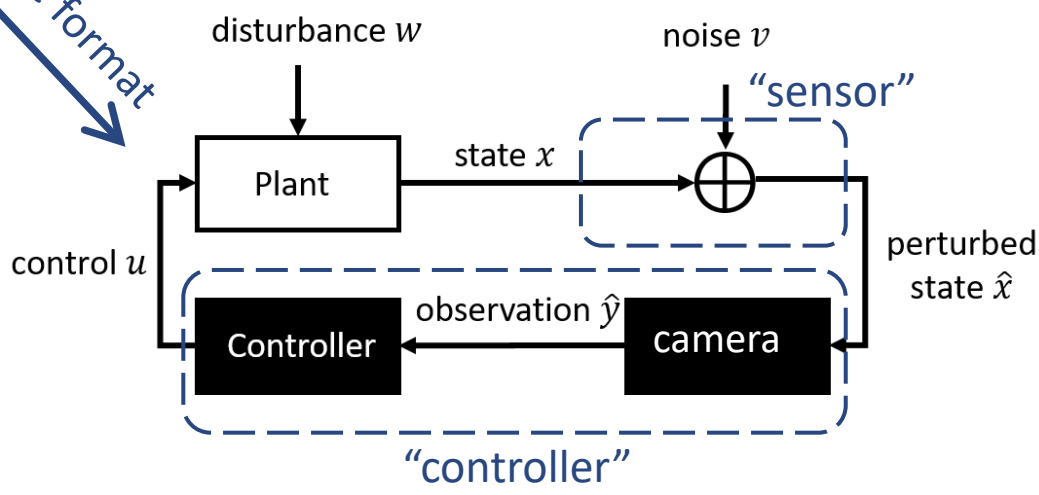
Σ_1: I know how to falsify this



Σ_2: What I want to falsify

In the same format

Idea: instead of perturbing the images (y, high dim), perturb the state (x, low dim) where the image is taken.

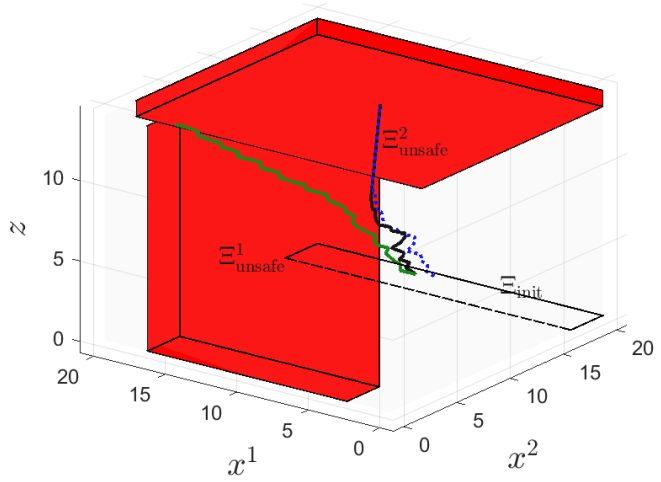


Σ_3: What I will falsify instead

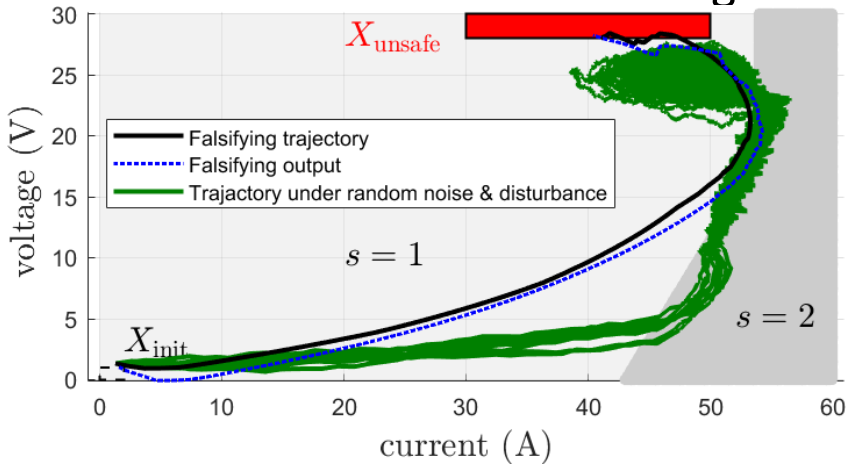
More Examples

More complex specification including a deadline

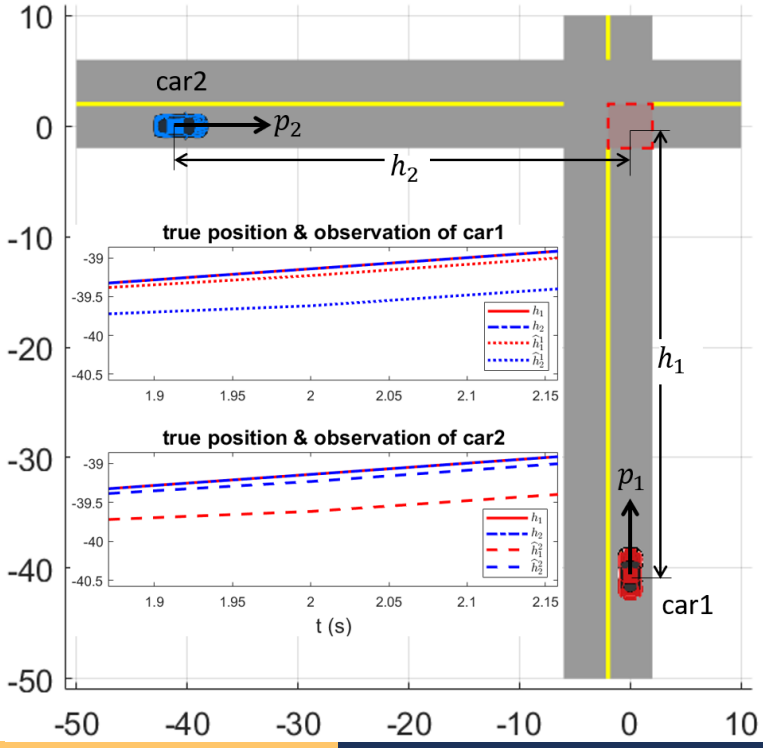
$$\square(x \in X_{\text{safe}}) \wedge \Diamond_{[0,T]} \square(x \in X_{\text{target}})$$



Buck converter with rule-based switching controller → forced to overvoltage

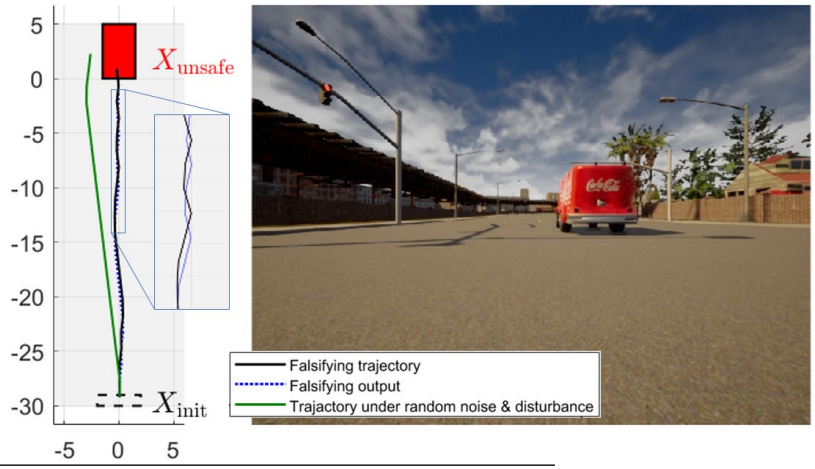


Two cars at an intersection, 8-D observation space, complex hybrid MPC controllers, each car has partial information of the other



Conclusion

- A new framework: synthesis-guided falsification:
  - Leads to explainable counterexamples
  - Works with black-box controllers (code)
  - Extends to vision/perception-based control or end-to-end learning controllers
- An interesting connection between adversarial examples in machine learning and those in decision-making



Thank you!
Q & A

G. Chou et al., "**Using control synthesis to generate corner cases: A case study on autonomous driving**", IEEE TCAD 2018
L. Yang & N. Ozay, "**Synthesis-guided Adversarial Example Generation for Gray-box Autonomous Systems with Sensing Imperfections**", (under review) preprint upon request