Counterexample-Guided Synthesis of Perception Models and Control

University of California Berkeley

Shromona Ghosh*, Hadi Ravanbakhsh*, Sanjit A. Seshia

Abstract

Goal: State of the art ML-based perception systems are still prone to errors. We need to design control modules for autonomous systems which are robust to perception errors.

Contributions:

✦ A novel counterexample-guided method to synthesize controllers robust to perception errors

✦ Data-driven inference of simple models of complex perception modules, including ML-based perception

✦ Two case studies:

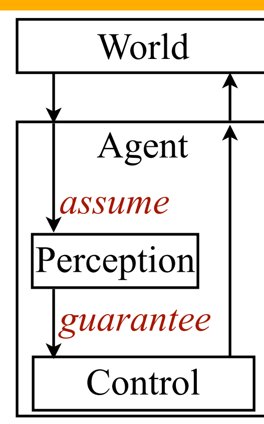
- ✦ Lane-keeping with a classical vision-based perception module
- ✦ Automatic braking with a neural network-based perception module

Approach

✦ **Common Approach:** Use assume-guarantee to and decompose the system into perception and control modules. Design each component separately.

- ✦ **Problem:** Currently we can't design provably correct perception modules

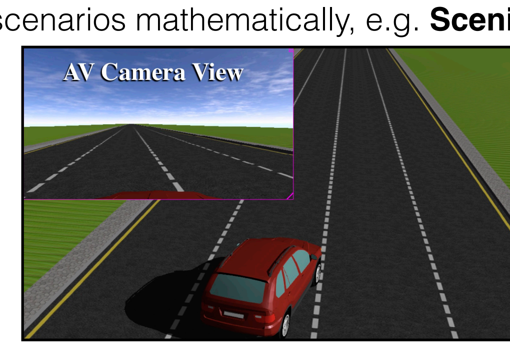
✦ **Our Approach:** Design a possibly faulty perception module. Afterwards, synthesize (a) assume-guarantee pairs for the perception module and (b) the control module



Problem

✦ **Simulation-based Verification:** Use high fidelity simulators for systematic safety analysis of autonomous agent in complex environments:

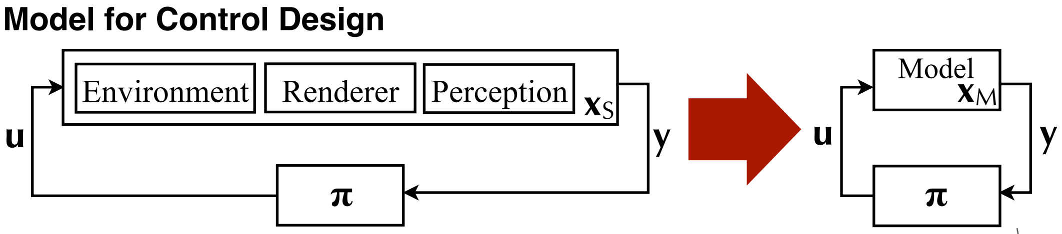
- ✦ **Requirements:** Define all possible scenarios mathematically, e.g. **Scenic**
- ✦ **Simulator:** We have access to internal state of the simulator and can enforce environment constraints by initializing the internal state.
- ✦ **Verification:** We have a verification oracle that can systematically tests the closed-loop system and finds counterexamples. E.g. **VerifAI**



✦ **Problem:** Given a parameterized control policy, and a faulty perception module, find a set of parameters s.t. the closed-loop system in the simulator remains safe w.r.t. the requirements.

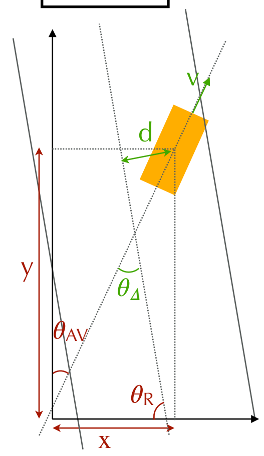
Assume-Guarantee through Modeling

✦ **Model for Control Design**



✦ **Lane-keeping Example**

- ✦ **Simulator state x_S:**
  - ✦ **Agent state:** x y θ_{AV} v
  - ✦ **Environment variables:** Target lane on the map, Time of day, weather, marks on the ground, etc.
- ✦ **Model state x_M:** d $\theta\Delta$ v
- ✦ **Perception output y:** d′ $\theta\Delta'$
- ✦ **Control input u:** steering



✦ **Simulator Traces**

- ✦ $\mathbf{y}(i) = h_S(\mathbf{x}_S(i))$
- ✦ $\mathbf{u}(i) = \pi(\mathbf{y}(i))$
- ✦ $\mathbf{x}_S(i+1) = f_S(\mathbf{x}_S(i), \mathbf{u}(i))$

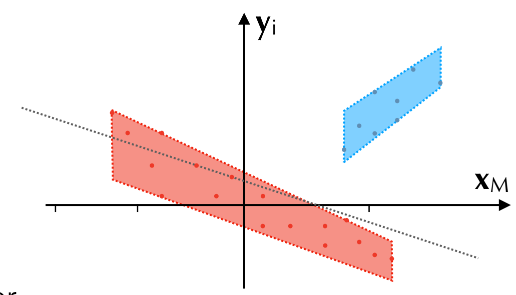
✦ **Model Traces**

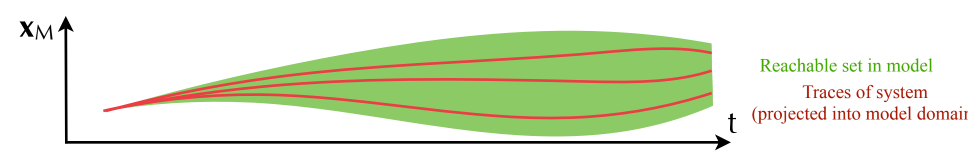
- ✦ $\mathbf{y}(i) = h_M(\mathbf{x}_M(i))$
- ✦ $\mathbf{u}(i) = \pi(\mathbf{y}(i))$
- ✦ $\mathbf{x}_M(i+1) = f_M(\mathbf{x}_M(i), \mathbf{u}(i))$

Inferring Sound Models

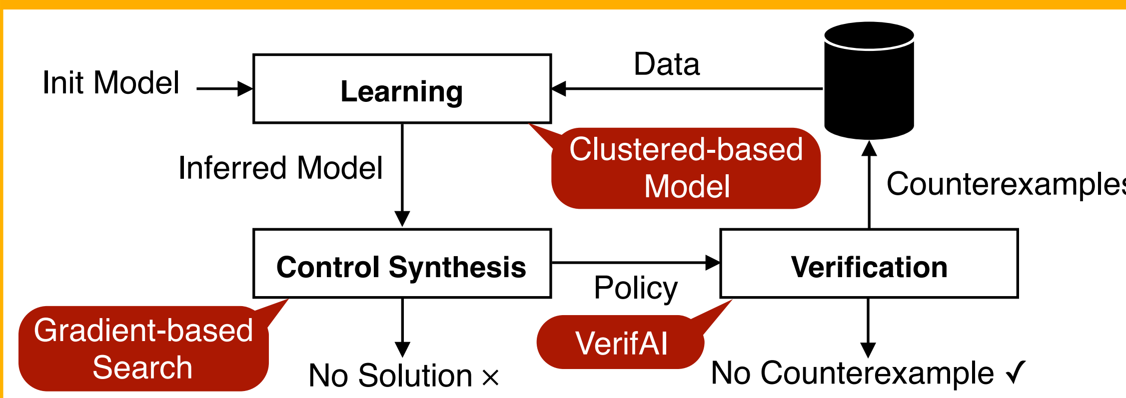
✦ **Infer h_M (and f_M) from simulator traces:**

- ✦ Extract ($\mathbf{x}_M$, $\mathbf{y}$) from traces
- ✦ Cluster data
- ✦ Learn linear model for each cluster
- ✦ Use a prior model
- ✦ Aggregate all models
- ✦ Model over-approximates the simulator





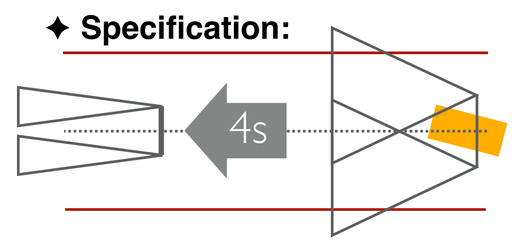
Counter-example Guided Search



Case Study: Lane-keeping

✦ **Problem**

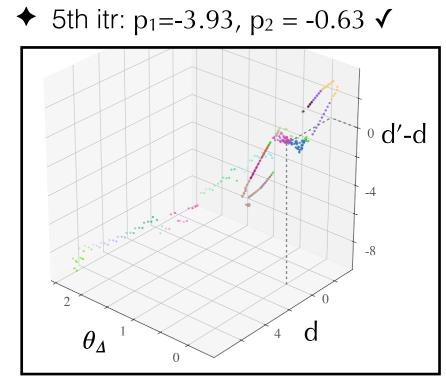
- ✦ **Policy:** $\pi : p_1\, \theta_\Delta + p_2\, d$
- ✦ **Perception:** θ_Δ' and v′ are accurate, but d′ is inaccurate
- ✦ **Specification:**



- ✦ **Modeling:** $d' \in h_M(d, \theta_\Delta, \underline{v})$

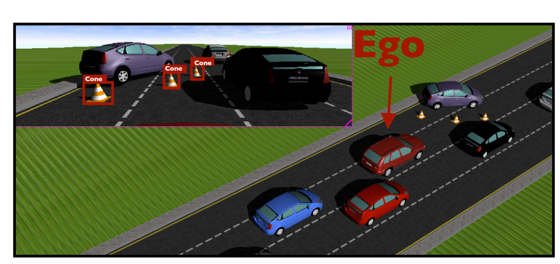
✦ **Counterexample-guided Search**

- ✦ 1st itr: $p_1=-0.5$, $p_2=-0.8$ ×
  - ✦ **VerifAI** finds counterexamples
- ✦ 5th itr: $p_1=-3.93$, $p_2=-0.63$ ✓



Case Study: Automatic Braking

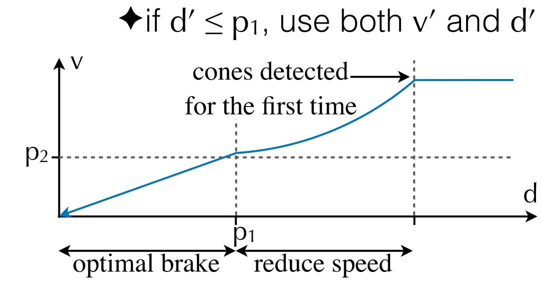
✦ **Problem:** Brake after cone detection, avoiding crash



- ✦ **Environment Parameters:** traffic speed, color of broken car, orientation of broken car…
- ✦ **Model state x_M:**
  - ✦ d: distance to cones
  - ✦ v: speed
- ✦ **Perception:** v′ is accurate, d′ is not: use size of detection boxes to measure distance to cones
- ✦ **Control input:** thrust
- ✦ **Modeling:** $d' \in h_M(d, \underline{v})$

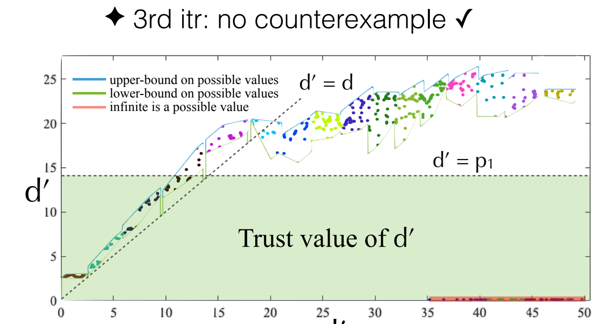
✦ **Policy:**

- ✦ if $d' > p_1$, reduce speed to p_2
- ✦ if $d' \leq p_1$, use both v′ and d′



✦ **Counterexample-guided Search**

- ✦ 1st itr: easy to find counterexamples x
- ✦ 2nd itr: only ew counterexamples. In all cases the color of broken car and cones color are similar x
- ✦ 3rd itr: no counterexample ✓



Acknowledgement

This work is supported in part by NSF grants CPS-1545126 (VeHICaL), CCF-1837132, by the DARPA Assured Autonomy grant, and by Berkeley Deep Drive.