Traffic Scenario Orchestration from Language via Constraint Satisfaction

Frieda Rong, Chris Zhang, Kelvin Wong, Raquel Urtasun

Improving Scenario-Based Testing

Motivation: Autonomous driving needs simulation for testing. But sourcing test cases is a bottleneck: scenario authoring and execution is tedious and inefficient.

Solution: To automatically convert written scenario descriptions into executable, closed-loop simulations that can reliably and precisely test ego behaviors.

Existing Approaches

Data-driven: Lack fine-grained granularity of control over reactive specifications and generalization to out of distribution scenarios that do not follow observed data.

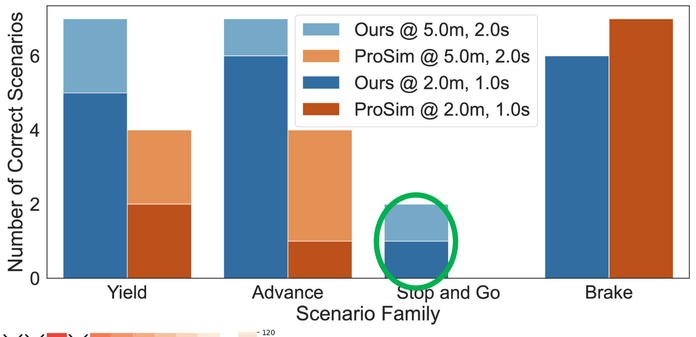
✘ Poor controllability and O.O.D. generalization

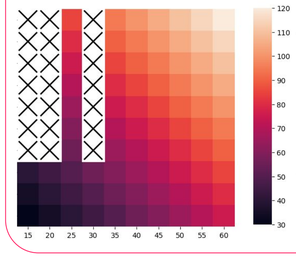
Optimization-based: Intuitive interface but laborious to program by hand at large scales. ✘ Not scalable

Our approach: Constraint-based, with automatic formalization from natural language into code.

✔ Supports flexible specifications (e.g. time-to-collision)
✔ Avoids time-consuming manual conversion

Comparison to a Data-Driven Baseline

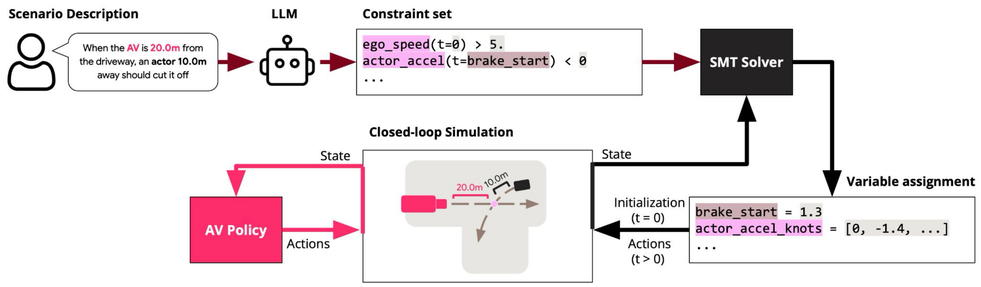




Top. Our method achieves challenging stop-and-go scenarios that are likely out of distribution for the learned baseline.

Left. The learned baseline systematically fails to orchestrate the crossed out parameter settings, lessening coverage.

Closed-Loop and Reactive Traffic Simulation



Given a text input description of a scenario, our orchestration pipeline prompts an LLM to convert it into a set of constraints which are then fed to an SMT solver.

Given these constraints and the current actor states of the ego and hero actor, the SMT solver generates a kinematically feasible motion plan for the hero actor.

Scenario Orchestration for Autonomous Driving

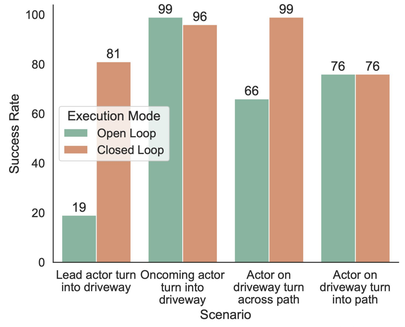
1. Model the state of an actor as a set of piecewise polynomials over time in 1D, recalling the *kinematic equations of motion*:

$$x_t = x_0 + v_0 t + \frac{1}{2} a_0 t^2 \qquad v_t = v_0 + a_0 t \qquad a_t = a_0$$

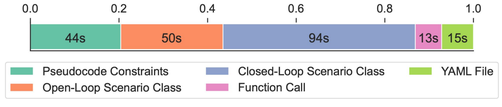
2. Symbolically expand expressions for actor state by applying rewrite rules to terms.
3. Encode the constraints into an SMT solver, e.g. Z3.
4. Solve for the constraints by invoking the solver.
5. Decode the solution assignments back into actor states by substituting values.

Repeat steps 1-5 in closed-loop against an uncontrolled ego policy and observe.

Symbolic Solving is Fast and Effective

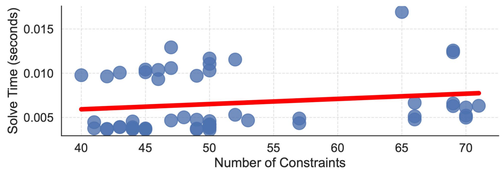


Top. Reactivity is key on two out of four hand-crafted scenarios, where closed-loop orchestration achieves 23% higher success rate compared to open-loop orchestration.



Top. LLM inference time across stages takes a total of a few minutes and is a one-time cost.

Bottom. SMT solver time takes seconds or less, and scales well with scenario complexity.



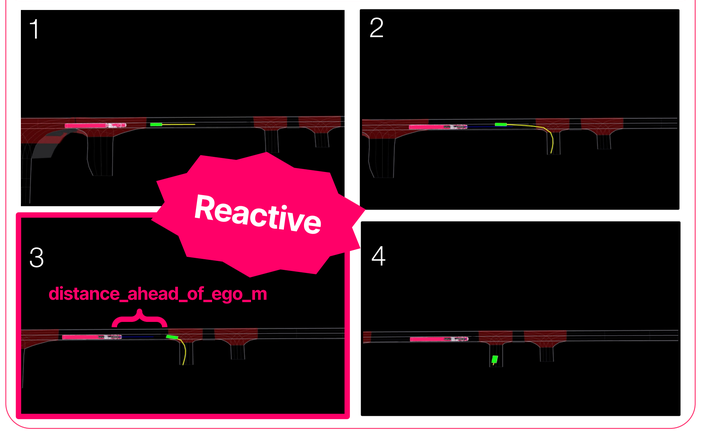
1. Input Description

Ego is lane-following straight along the road at {ego_initial_speed_mps}. There is a driveway {distance_to_driveway} in front of the ego that merges into the ego's lane. There is a lead vehicle that is initially driving at {initial_speed_mps}. ***When the lead vehicle is {distance_ahead_of_ego_m} ahead of the ego****, it turns right into the driveway and gradually comes to a stop.*

2. Intermediate Representation

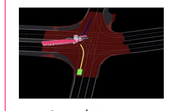
Actor 0:	A0v(t0) == ego_initial_speed_mps
- E	A0x(t0) == turn - distance_to_driveway_m
- [t0, go, t1, go, t2]	A1v(t0) == initial_speed_mps
	A1x(t1) - A0x(t1) == distance_ahead_of_ego_m
Actor 1:	A1x(t1) == turn
- E, S	A1v(t2) == 0
- [t0, go, t1, dec, t2]	A0(t1) == A1(t1)

3. Output Simulation



Future Work: Scaling Up

stop sign actor right of way



stop sign ego right of way

- Support more map topologies and more background and hero actors.
- Prevent hallucinations caused by the LLM during the conversion stage, e.g. via generating a custom DSL parser
- Measure and target statistical realism using evaluation metrics for capturing real world data distributions