Rethinking Closed-loop Training for Autonomous Driving

A COLLABORATION BETWEEN WAABI AND U OF TORONTO

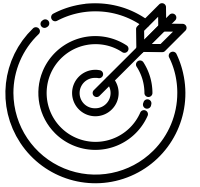
MOHAMMAD HOUDEIB

MILA, AUTONOMOUS VEHICLES, FALL 2025

Requirements of a SDV

Safety

Decision Making



Progress

Comfort

Robustness

Planning

Common Approaches and their Limitations

Modular Planners
perception → localization → prediction→ planning → control

- Hand Engineered
- Fails in dense, uncertain or long-horizon interaction
- Fragile to distribution shift and corner cases

Imitation Learning
Behavioral Cloning

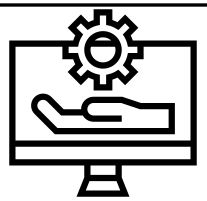
- Easy to scale with human data
- Open loop distribution shift
- Errors accumulate → unsafe

Existing RL Approaches

- Model-free = slow, expensive, unstable
- No multi-step future reasoning
- Too weak to handle complex traffic

Challenges of Real World Closed Loop Training

Unsafe



No Software Updates on Demand

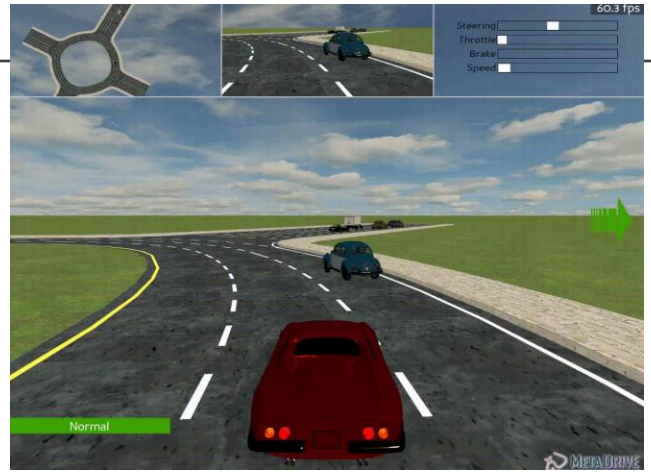
Interesting scenarios are rare to come by

Costly

Solution: Learn Closed Loop in a Simulation

CARLA

MetaDrive



NVDIA

Drive Sim

WAABI

World

Motivation

What type of scenarios do we need to learn to drive safely?

Why Scenario Design Matters?

Tasks like car racing or empty lanes

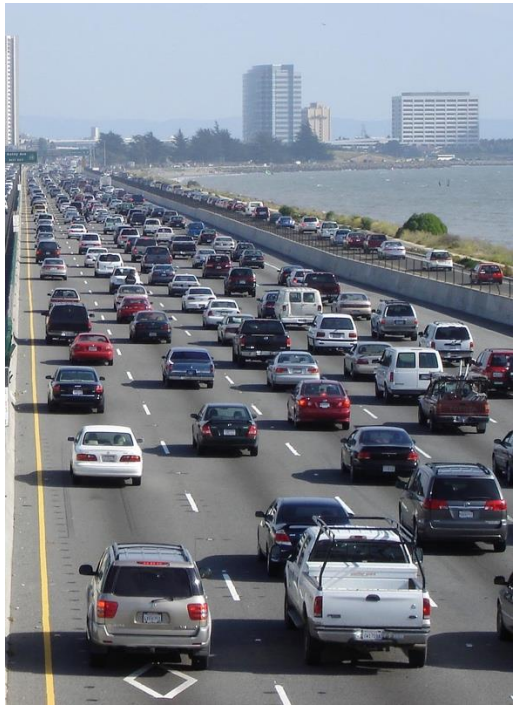
→ fail to capture complexity

Supervised learning has shown

→ scale improves generalization

But in closed loop training, we don't know:

- whether targeted scenarios help more than free flow traffic?
- how many scenario variations do we need?
- how does scenario diversity affect safety?



Benchmark design is a critical, unanswered question!

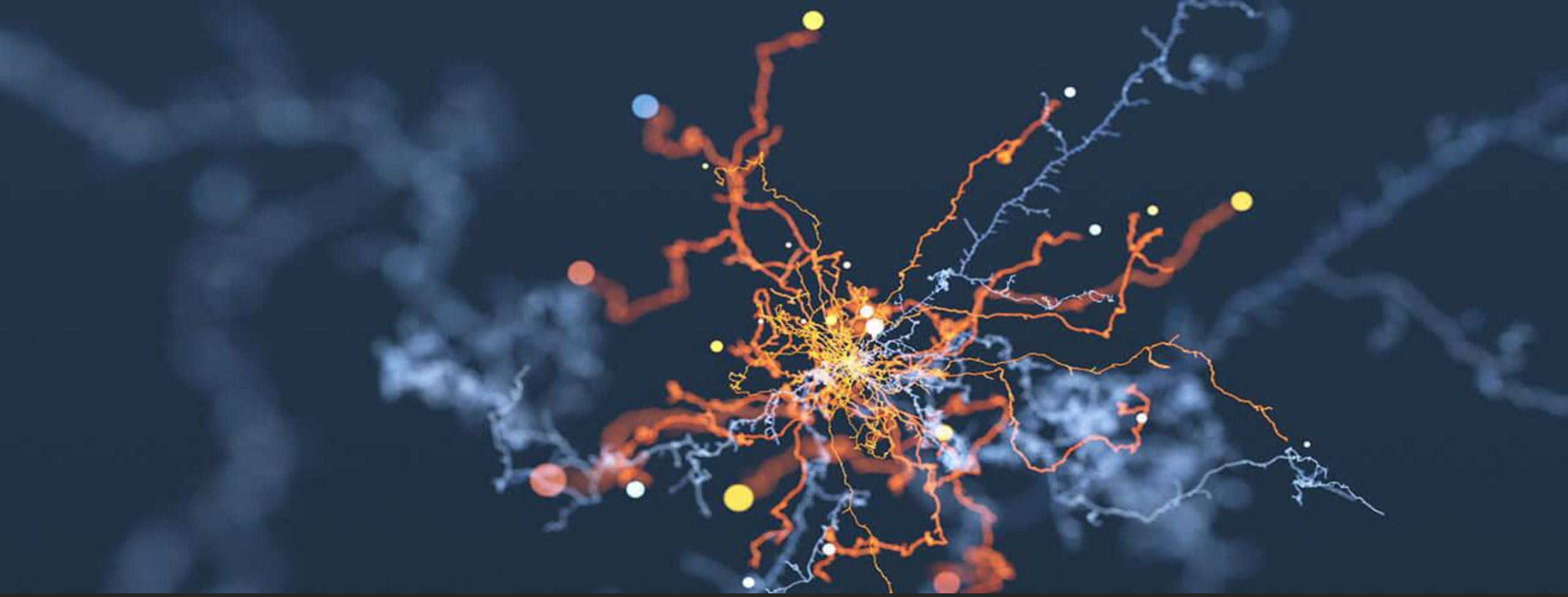
What the authors built

A new highway-driving simulator benchmark:

- Built on top of Waabi's simulator
- Can generate realistic free-flow traffic
- And targeted scenarios (cut-ins, braking, merging, blocking)
- With procedural variation ⟶ behavioral diversity
- Allows scaling scenario complexity

TRAVL: Trajectory Value Learning

- A new RL algorithm
- Multi-step lookahead
- Plans in trajectory space, not raw actions
- Uses imagined (counterfactual) trajectories to learn MUCH fasters
- Overcomes weaknesses of model-free RL



Reinforcement Learning for SDV's: Quick Recap

Model-Free, Model-Based and Off-Policy RL: The Basics

Model-Free RL:

- Learns a policy by interacting with the real environment
- No explicit model of future dynamics
- Learns “what to do” directly from trial and error

Analogy: Learning to drive only by practicing on the road

Model-Based RL:

- Learns or uses a model of the environment’s dynasmics
- Predict possible futures by simulating ahead
- Uses rollouts for planning and reasoning

Analogy: driving with a simulator that lets you test future scenarios before acting

Off-Policy RL:

- Learns from past experience, not only current policy
- Uses a replay buffer of previous interactions
- Can learn from data generated by **other policies** or **imagined data**

Limitations of Model-Free and Model-Based RL

Model-free RL:

Predicts instant control actions (steering, throttling):

→ no long-term reasoning

→ struggles in complex interactions

Learns only from real environment experience:

→ learns slowly from real samples only

Model based RL:

Uses a world (dynamics) model to simulate ahead

→ Good long-term reasoning

But model rollouts are too slow at inference

→ especially with many possible future trajectories

Off Policy RL:

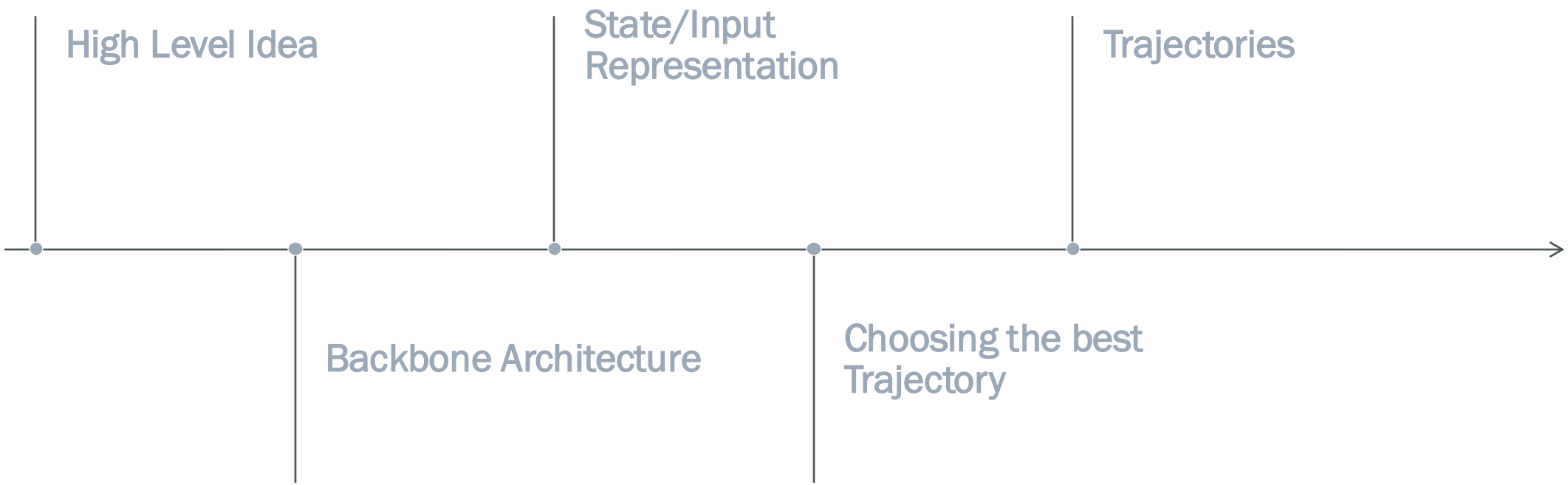
Still constrained by action-space parametrization

→ Evaluationg one-step actions, not full trajectories

→ Insufficient for reasoning about complex driving maneuvers

TRAVL: Trajectory Value Learning

TRAVL


High Level Idea
Backbone Architecture
State/Input Representation
Choosing the best Trajectory
Trajectories

TRAVL : Trajectory Value Learning (High level)

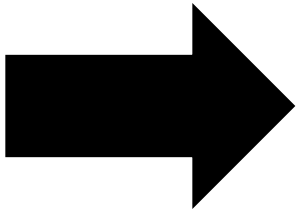
- Instead of outputting actions, outputs **trajectories**
- Provides **explicit multi-step lookahead**
- **Avoids** expensive model-based rollouts
- Learns from **real + imagined** trajectories
- Uses **off-policy RL** (replay buffer)

State Representation (BEV Rasterization)

State Space *S* includes:

1) HD Map of the surrounding road network

2) Past T' seconds of motion history for:
- All actors (surrounding vehicles)
- Ego vehicle



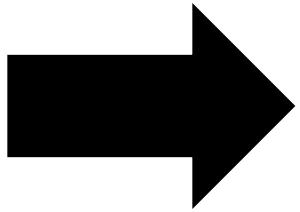
Rasterized as BEV Tensor

1) For each past frame (T'), draw **actor bounding boxes** → **T' channels**

2) Draw ego history the same way → **another T' channels**

3) Encode HD map across **M binary layer**s (lane centerlines, boundaries, target route, sop lines…)

4) **Add 2 positional** channls (x,y coordinate of each pixel) so CNN knows the absolute location



Final Input Tensor

H x W x (2T' +M + 2)

What is a trajectory?

- TRAVL samples candidate future trajectories

$$\tau = \{(x^0, y^0), (x^1, y^1), \cdots, (x^T, y^T)\}$$

⟶ trajectory = 2D points that represents ego's location over next T timesteps

- Typically, composed of 10-20 timesteps at 10Hz

⟶ 1-2 seconds long

How TRAVL Generates Candidate Trajectories

- Samples physically feasible trajectories from trajectory sampler
- Samples user map priors so candidates follow lanes, curves, merges…

Key components:

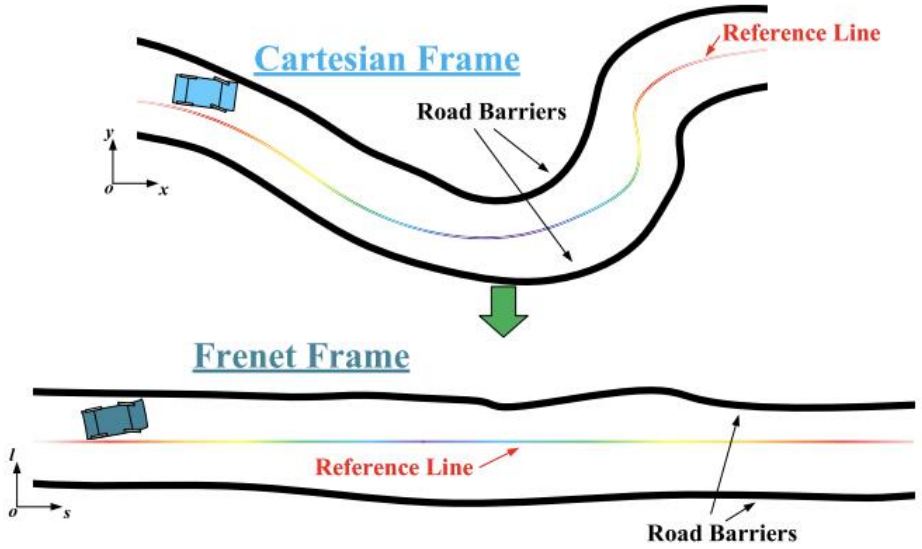
•-Uses a bicycle model for physically feasible vehicle motion

•Samples in **Frenet frame** (along lane coordinates)

→ Naturally handles curved roads

→ Represents speed profiles (longitude)

→ Represents lateral motion within lane



At inference:

Even though action is a full trajectory, vehicle only executes the first part, then replans (MPC-style)

The TRAVL Architecture: Feature Extraction & Cost Prediction

Backbone Network

- A CNN (ResNet style) processes the **H x W x (2T' +M + 2) BEV tensor**
- Produces feature map encoding scene context (lanes, other actors, motion history)
- Trajectory sampler proposes physically feasible candidate paths, concatenated with kinematics function
- Two MLP's predict:
  - Rθ(s, τ): **short-term cost** of the trajectory **(within T)**
  - Vθ(s, τ): **long-term value of** the trajectory **(beyond T)**

  Final Q-value: **Q(s, τ) = Rθ + Vθ**

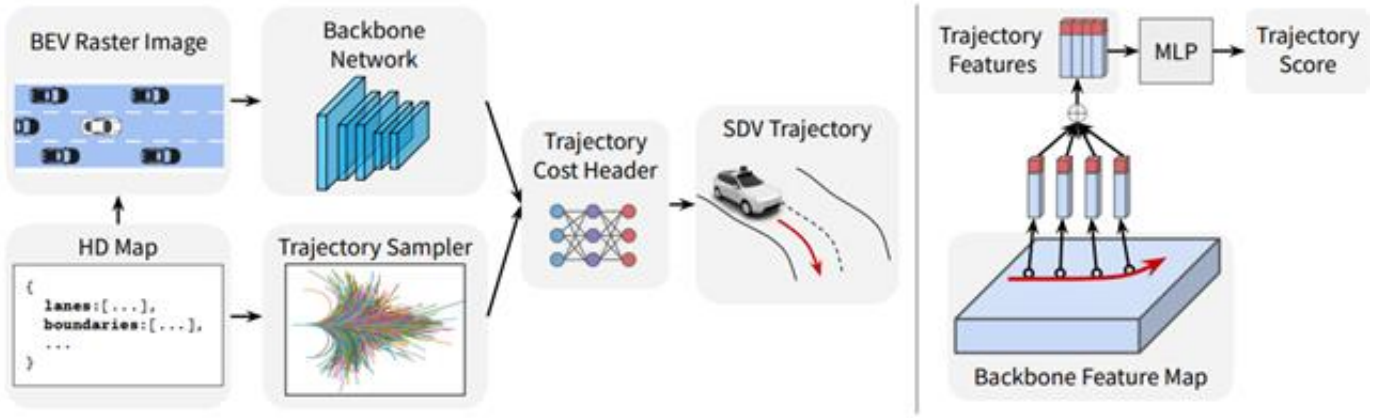


Fig. 2: TRAVL leverages rich backbone features to predict the cost of following a trajectory. The lowest costed trajectory is selected as the SDV's plan

Picking the Best Trajectory

$$\tau^* = \arg\max_{\tau \in \mathcal{T}} Q(s, \tau).$$

Learning from Real and Imagined Trajectories

Real experience: trajectories executed in the simulator

Imagined experience: counterfactual trajectories τ′ ego could have taken (generated cheaply using approximate world model)

→ Freshly generated alternative futures used only for training

→ The alternative trajectory only lasts fractions of a second

→ Other actors stay the same; only ego path changes

Intuition: What if the ego had taken this other trajectory instead?

Why Counterfactual Trajectories Matter

Benefits:

1. Adds many cheap training samples → better supervision
2. Strong short-term cost learning ($R\theta$) (x10 faster)
3. Stored with real data in the replay buffer (off-policy)
4. Dramatically improves sample efficiency

Counterfactual rewards loss enables quicker Q-learning

$$Q^{k+1} \leftarrow \arg\min_{Q_\theta} \mathbb{E}_D \left[\underbrace{\left(Q_\theta(s,\tau) - \mathcal{B}_\pi^k Q^k(s,\tau)\right)^2}_{\text{Q-learning}} + \alpha_k \underbrace{\mathbb{E}_{\tau' \sim \mu(\tau'|s)} \left(R_\theta(s,\tau') - r'\right)^2}_{\text{Counterfactual Reward Loss}} \right],$$

$$s.t. \quad Q_\theta = R_\theta + V_\theta, \quad V_\theta = \gamma \mathcal{P}_\pi^k Q^k. \tag{3}$$

Algorithm 1 TRAVL: TRAjectory Value Learning

Require: Simulator, Training Scenario Set

Initialization: $\mathcal{D} \leftarrow \emptyset$, $\pi(\tau|s) \leftarrow \text{Uniform}(\tau)$, TRAVL network $\leftarrow$ random weights.

Asynchoronous Experience Collection:

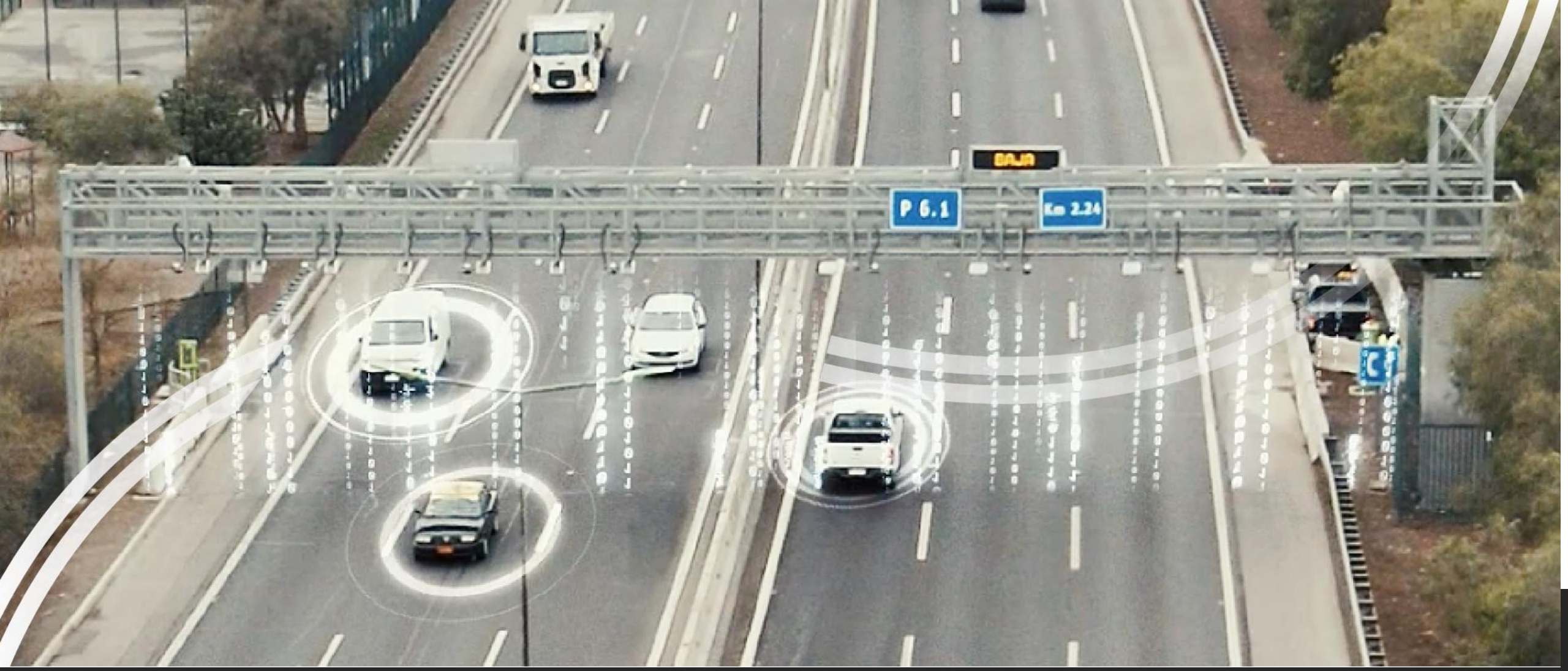
while Learning has not ended **do**
  Sample a scenario variation from the training scenario set.
  Produce $(s^t, \tau^t, r^t, s^{t+1})$ by interacting the policy π and the simulator on the sampled scenario.
  Store $(s^t, \tau^t, r^t, s^{t+1})$ to the replay buffer $\mathcal{D}$.
end while

Learning:

for $k = 0, \cdots,$ max_iter **do**
  Draw (mini-batch) samples $(s^t, \tau^t, r^t, s^{t+1})$ from $\mathcal{D}$.
  Draw a set of trajectory samples $\mathcal{T}$ given s^t.
  Compute $R_\theta(s^t, \tau^t)$, $V_\theta(s^t, \tau^t)$ and $R_\theta(s^t, \tau')$, $V_\theta(s^t, \tau')$ for $\tau' \in \mathcal{T}$ using TRAVL network.
  Evaluate $r' = R(s^t, \tau', s')$ for $\tau' \in \mathcal{T}$ using reward functions.
  Compute $\mathcal{L}$ using Eq. 7.
  Update network parameter θ using gradients of $\mathcal{L}$.
  $Q_\theta \leftarrow R_\theta + V_\theta$.
  $\pi(\tau|s) \leftarrow \begin{cases} \arg\max_\tau Q_\theta(s, \tau), & \text{with probability } 1 - \epsilon \\ \text{randomly sample } \tau, & \text{with probability } \epsilon. \end{cases}$
end for

TRAVL: Summary

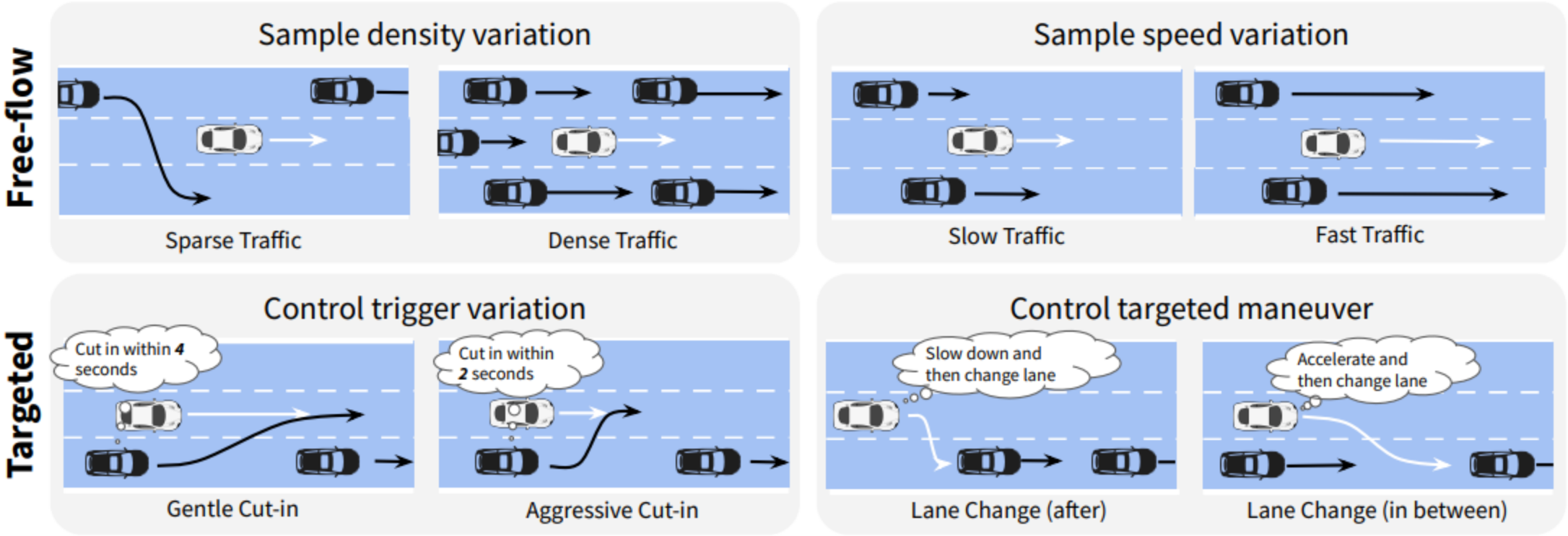
- Thinks in trajectories, not actions
- Scores trajectories with a learned Q-function
- Samples safe, map-respecting trajectories
- Learns from *real* and *imagined* experiences
- Achieves much higher sample efficiency than model-free RL
- Avoids expensive rollouts of model-based RL



Back to the Main Motivation:

What type of scenarios do we need to learn to drive safely?

Why Build a Closed Loop Benchmark?



Free Flow Scenarios (Uncontrolled Traffic)

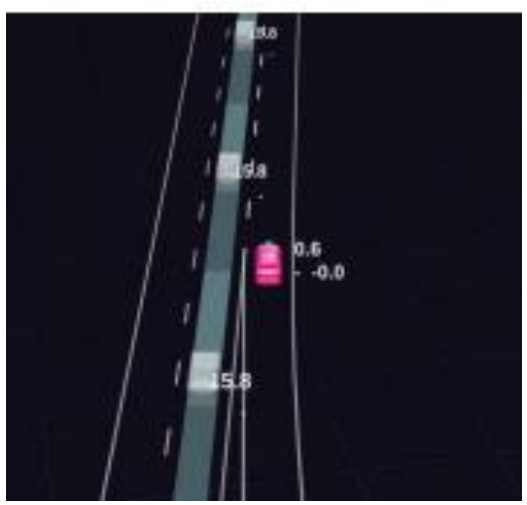
Mimic normal highway driving

No control over surrounding actors

Randomized parameters:

- Traffic density
- Actor initial speeds + target speeds
- Lane goals
- Map topology, curvature, number of lanes

Generated through distribution sampling



Targeted Scenario Set

Designed to test specific autonomous driving capacities.

Three ego intentions: Lane Following, Lane Change, Lane Merge

Five other actor behaviors: Braking, Accelerating, Blocking, Cut-in, Negotiation

→ 3 × 5 × actor placements ≈ **24 scenario types**

Each scenario type parametrized by:

- Ego initial heading and speed
- Relative speed and position of surrounding actors
- Time to collision / distance triggers
- IDM behaviour parameters
- Map Geometry parameters

Experiments

Goals:

1. Show the importance of targeted scenarios vs Free flow traffic
2. Study effects of data diversity and scale in learning a good policy

Metrics

Scenario Pass Rate	% of scenarios that reach their goal (e.g maintain target lane)
Collision Rate	% of scenarios where ego collides
Progress	Distance traveled (meters) before end of scenario
Minimum Time to Collision (MinTTC)	How many seconds would remain before ego collides with another actor
Minimum Distance to closest Actor	Minimum distance between ego and any other actor in the scene

Datasets

Free flow dataset: 834 training and 274 testing scenarios

Targeted scenario dataset: 783 training and 256 testing scenario

80 / 20 split

Average scenario duration: 15 seconds

Closed Loop Benchmarking

Method		Pass Rate ↑	Col. Rate ↓	Prog. ↑	MinTTC↑	MinDist↑
Imit. Learning	C	0.545	0.177	240	0.00	2.82
PPO [69]		0.173	0.163	114	0.00	5.56
A3C [49]		0.224	0.159	284	0.03	4.65
RAINBOW[7] [30]		0.435	0.270	234	0.00	1.38
Imit. Learning	T	0.617	0.261	**286**	0.00	1.49
PPO [69]		0.273	0.249	200	0.00	1.73
A3C [49]		0.362	0.137	135	0.30	6.14
RAINBOW[7] [30]		0.814	0.048	224	0.45	9.70
TRAVL (ours)		**0.865**	**0.026**	230	**0.82**	**12.62**

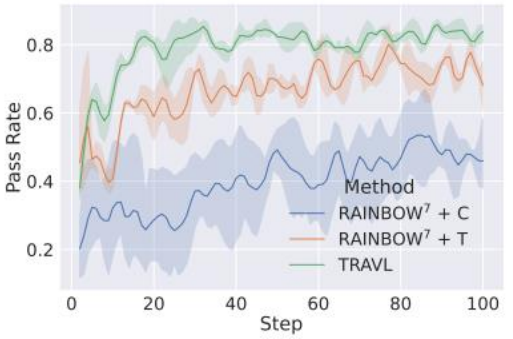


Fig. 5: Training curves for 3 runs. TRAVL has the least variance and converges faster.

Targeted vs Free Flow Benchmarking

			Test					
			Pass Rate ↑		Collision Rate ↓		Progress ↑	
			Free-flow	Targeted	Free-flow	Targeted	Free-flow	Targeted
Train	RB[7]+T	Free-flow	0.783	0.453	0.198	0.228	146	173
		Targeted	0.885	0.815	0.104	0.048	231	224
	TRAVL	Free-flow	0.784	0.696	0.198	0.177	229	219
		Targeted	0.903	0.865	0.089	0.026	172	230

Behavioral Scale and Diversity (1)

Fig. 4: Increasing scenario diversity improves performance across the board.

Behavioral Scale and Diversity (2)

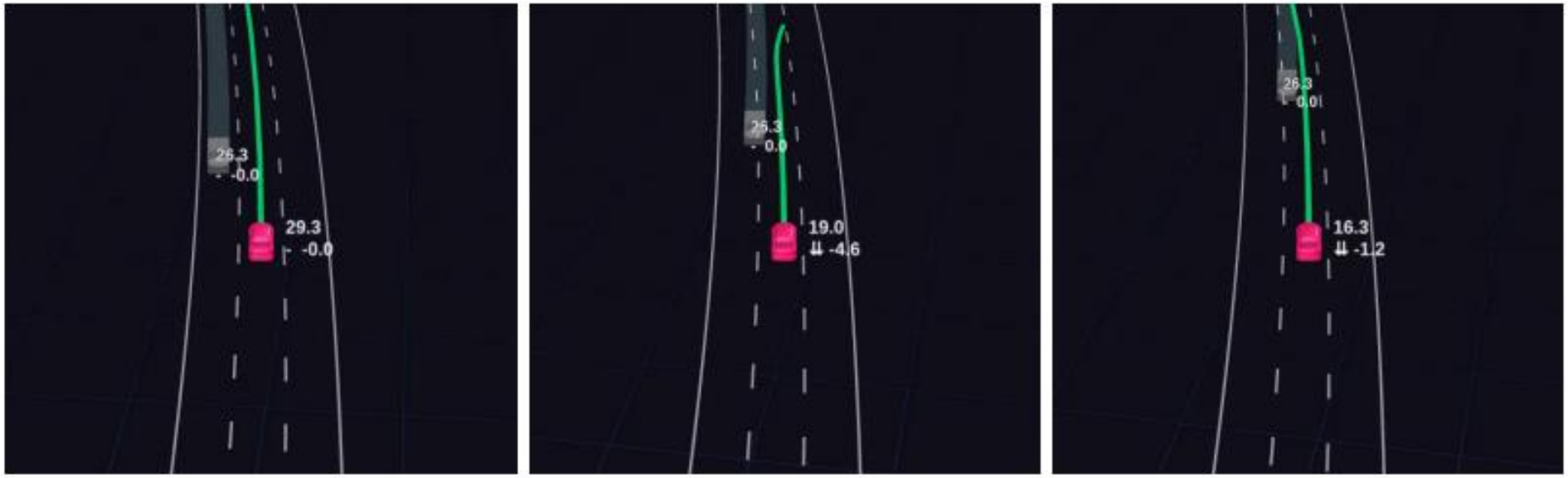
Method	Pass Rate ↑	Col. Rate ↓	Prog. ↑	MinTTC ↑	MinDist ↑
Map Variation	0.738	0.070	230	0.53	8.97
Beh. Variation	**0.872**	**0.022**	228	0.60	9.82
Both	0.865	0.026	**231**	**0.82**	**12.6**

Table 3: We train a TRAVL agent on datasets with different axes of variation. Behavioral variation has larger effects than map for learning robust driving policies.

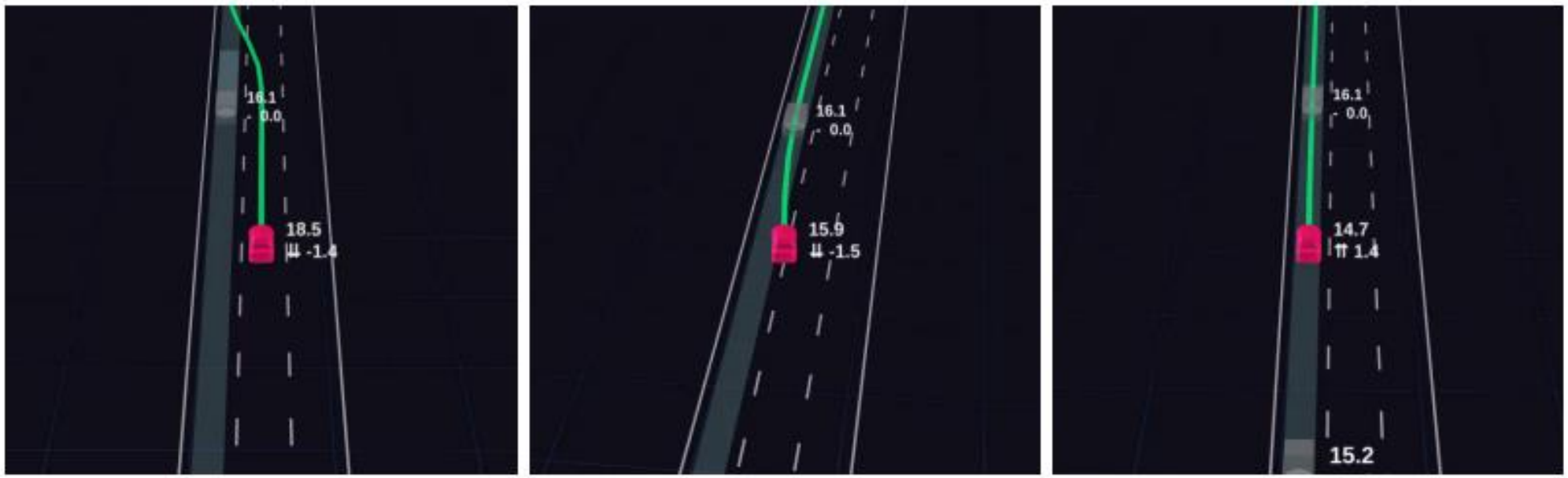
Qualitative Results

Free Flow: Merge

Targeted: Reacting to a cut-in

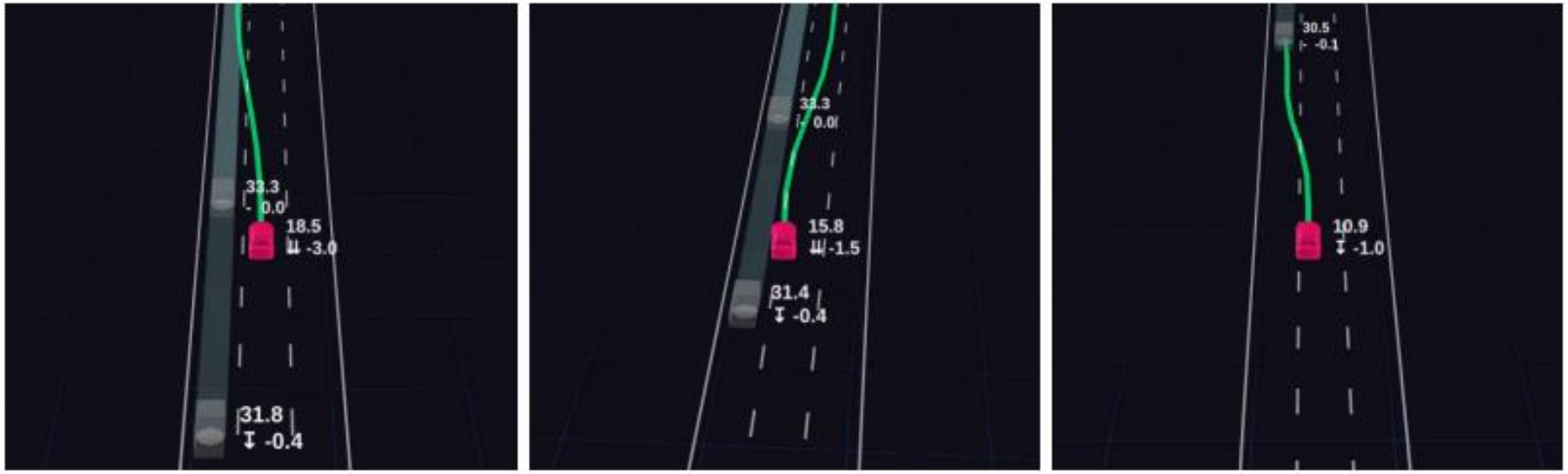


Targeted: Agent learnt to slow down to lane change between 2 actors



Targeted: Agent learns to speed up to merge into traffic

Failure Case: Unsafe Lane Change (But Collision successfully averted)



Conclusion

1) **Scenario Design Matters:** Targeted scenarios produce better results in closed loop

2) **Diversity Matters: More variation in actors' behavioral diversity leads to safer and more robust policies**

3) Trajectory-based learning works better than control-based RL: Better long-term horizon

4) TRAVL is a highly efficient way to train closed loop driving policies due to counterfactual data

Questions?