CtRL-Sim: Reactive and Controllable Driving Agents with Offline Reinforcement Learning

Anonymous Author(s)
Affiliation
Address
email

Abstract: Evaluating autonomous vehicle stacks (AVs) in simulation typically involves replaying driving logs from real-world recorded traffic. However, agents replayed from offline data are not reactive and hard to intuitively control. Existing approaches address these challenges by proposing methods that rely on heuristics or generative models of real-world data but these approaches either lack realism or necessitate costly iterative sampling procedures to control the generated behaviours. In this work, we take an alternative approach and propose CtRL-Sim, a method that leverages return-conditioned offline reinforcement learning to efficiently generate reactive and controllable traffic agents. Specifically, we process real-world driving data through a physics-enhanced Nocturne simulator to generate a diverse offline reinforcement learning dataset, annotated with various reward terms. With this dataset, we train a return-conditioned multi-agent behaviour model that allows for fine-grained manipulation of agent behaviours by modifying the desired returns for the various reward components. This capability enables the generation of a wide range of driving behaviours beyond the scope of the initial dataset, including adversarial behaviours. We demonstrate that CtRL-Sim can generate diverse and realistic safety-critical scenarios while providing fine-grained control over agent behaviours.

Keywords: Autonomous Driving, Simulation, Offline Reinforcement Learning

1 Introduction

Recent advances in autonomous driving has enhanced their ability to safely navigate the complexities of urban driving [1]. Despite this progress, ensuring operational safety in long-tail scenarios, such as unexpected pedestrian behaviours and distracted driving, remains a significant barrier to widespread adoption. Simulation has emerged as a promising tool for efficiently validating the safety of autonomous vehicles (AVs) in these long-tail scenarios. However, a core challenge in developing a simulator for AVs is the need for other agents within the simulation to exhibit realistic and diverse behaviours that are reactive to the AV, while being easily controllable. The traditional approach for evaluating AVs in simulation involves fixing the behaviour of agents to the behaviours exhibited in pre-recorded driving data. However, this testing approach does not allow the other agents to react to the AV, which yields unrealistic interactions between the AV and the other agents.

To address the issues inherent in non-reactive log-replay testing, prior work has proposed rule-based methods [2, 3] to enable reactive agents. However, the behaviour of these rule-based agents often lacks diversity and is unrealistic. More recently, generative models learned from real-world data have been proposed to enhance the realism of simulated agent behaviours [4, 5, 6, 7, 8, 9]. While these methods produce more realistic behaviours, they are either not easily controllable [4, 5, 9] or require costly sampling procedures to control the agent behaviours [10, 8, 7, 11, 12].

Submitted to the 8th Conference on Robot Learning (CoRL 2024). Do not distribute.

In this paper, we propose CtRL-Sim to address these limitations of prior work. The CtRL-Sim framework utilizes return-conditioned offline reinforcement learning (RL) to enable reactive, *closed-loop*, controllable, and probabilistic behaviour simulation within a physics-enhanced Nocturne [13] environment. We process scenes from the Waymo Open Motion Dataset [14] through Nocturne to curate an offline RL dataset for training that is annotated with reward terms such as "vehicle-vehicle collision" and "goal achieved". We propose a return-conditioned multi-agent autoregressive Transformer architecture [15] within the CtRL-Sim framework to imitate the driving behaviours in the curated dataset. We then leverage exponential tilting of the predicted return distribution [16] as a simple yet effective mechanism to control the simulated agent behaviours. While [16] exponentially tilts towards more optimal outcomes for the task of reward-maximizing control, we instead propose to tilt in *either direction* to provide control over both good and bad simulated driving behaviours.

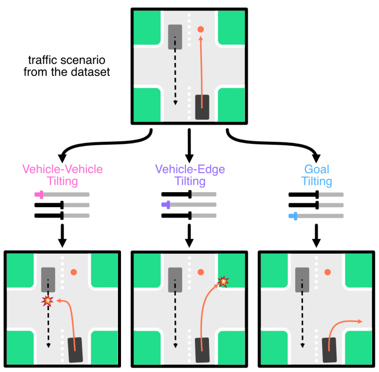


Figure 1: **CtRL-Sim allows for controllable agent behaviour** from existing datasets. This allows users to create interesting edge cases for testing and evaluating AV planners.

We show examples of how CtRL-Sim can be used to generate counterfactual scenes when exponentially tilting the different reward axes in Figure 1. For controllable generation, CtRL-Sim simply requires specifying a tilting coefficient along each reward axis, which circumvents the costly iterative sampling required by prior methods. CtRL-Sim scenarios are simulated within our physics-extended Nocturne environment. We summarize our main contributions: **1.** We propose CtRL-Sim, which is, to the best of our knowledge, the first framework applying return-conditioned offline RL for controllable and reactive behaviour simulation. Specifically, CtRL-Sim employs exponential tilting of factorized reward-to-go to control different axes of agent behaviours. **2.** We propose an autoregressive multi-agent encoder-decoder Transformer architecture within the CtRL-Sim framework that is tailored for controllable behaviour simulation. **3.** We extend the Nocturne simulator [13] with a Box2D physics engine, which facilitates realistic vehicle dynamics and collision interactions.

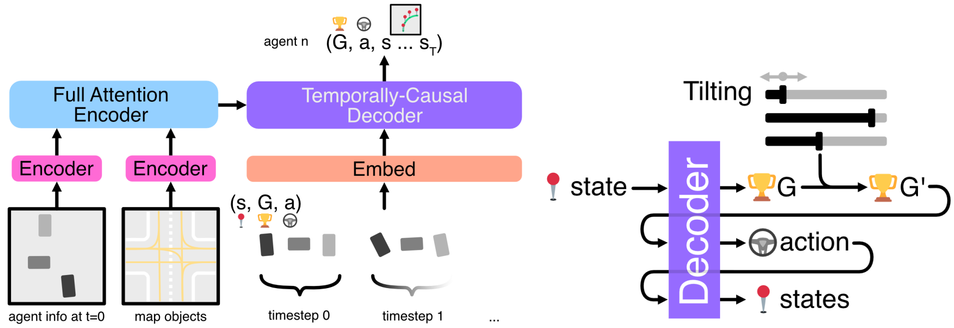
We demonstrate the effectiveness of CtRL-Sim at producing controllable and realistic agent behaviours compared to prior methods. We also show that finetuning our model in Nocturne with simulated adversarial scenarios enhances control over adversarial behaviours. CtRL-Sim has the potential to serve as a useful framework for enhancing the safety and robustness of AV planner policies through simulation-based training and evaluation.

2 CtRL-Sim

In this section, we present the proposed CtRL-Sim framework for behaviour simulation. We first introduce CtRL-Sim in the single-agent setting, and subsequently show how it extends to the multi-agent setting. Given the state of an agent s_t at timestep t and additional context (e.g., the road structure, the agent's goal), the behaviour simulation model employs a driving policy $\pi(a_t|s_t, m, s_G)$ and a forward transition model $\mathcal{P}(s_{t+1}|s_t, a_t)$ to control the agent in the scene. Note that a_t is the action, m is the map context, and s_G is the prescribed goal state. Using the physics-extended Nocturne simulator, we have access to a physically-realistic forward transition model $\mathcal{P}$. In this work, we are interested in modelling the policy $\pi(a_t|s_t, m, s_G)$ such that we can both imitate the real distribution of driving behaviour and control the agent's behavior to generate long-tail counterfactual scenes.

2.1 Our Approach to Controllable Simulation via Offline RL

We consider the common offline RL setup where we are given a dataset $\mathcal{D}$ of trajectories $\tau_i = \{\ldots, s_t, a_t, r_t, \ldots\}$, with states $s_t \in \mathcal{S}$, actions $a_t \in \mathcal{A}$ and rewards r_t. These trajectories are generated using a (suboptimal) behaviour policy $\pi_B(a_t|s_t)$ executed in a finite-horizon Markov



(a) **Model Overview** (b) **Inference Time Forward Pass**

Figure 2: 2a (left) The agent and map data at $t = 0$ are encoded and fed through a Transformer encoder as context for the decoder, similar to [9]. Trajectories are arranged first by agents, then by timesteps, embedded, and fed through the decoder. For each agent, we encode (s_t, G_t, a_t) (i.e. state, return-to-go, action) and we predict from these $(G_t, a_t, s_{t+1}, \dots, s_T)$. 2b (right) At inference time, the state predicts the return-to-go. The return-to-go is tilted (i.e., reweighed to encourage specific behaviors) and is used to predict the action, which in turn is used to predict the next states.

decision process. The return at timestep t is defined as the cumulative sum of scalar rewards obtained in the trajectory from timestep t, $G_t = \sum_{t'=t}^{T} r_{t'}$. The objective of offline RL is to learn policies that perform as well as or better than the best agent behaviours observed in $\mathcal{D}$.

The primary insight of this work is the observation that offline RL can be an effective way to perform controllable simulation. That is, the policy distribution over actions can be tilted at inference time towards desirable or undesirable behaviors by specifying different values of return-to-go G_t. This requires a different formulation of the policy such that it is conditioned on the return $\pi(a_t|s_t, G_t, s_G)$[1]. In Table 3, we outline how different approaches in offline RL have learned return-conditioned policies. In this work, we adopt an approach that learns the joint distribution of returns and actions of an agent in a given dataset. Specifically, $p_\theta(a_t, G_t|s_t, s_G) = \pi_\theta(a_t|s_t, s_G, G_t)p_\theta(G_t|s_t, s_G)$. We note that [17] found it helpful to also utilize a *model-based* return-conditioned policy, whereby the future state is modelled as part of the joint distribution being learned. This is shown to provide a useful regularizing signal for the policy, even though the future state prediction is not directly used at inference time. In this work, we also found it helpful to regularize the learned policy by predicting the full sequence of future states. The final distribution we are aiming to model is thus given by $p_\theta(s_{t+1:T}, a_t, G_t|s_t, s_G) = p_\theta(s_{t+1:T}|s_t, s_G, G_t, a_t)\pi_\theta(a_t|s_t, s_G, G_t)p_\theta(G_t|s_t, s_G)$.

At inference time, we obtain actions by first sampling returns $G_t \sim p_\theta(G_t|s_t, s_G)$ and then sampling actions $a_t \sim \pi_\theta(a_t|s_t, s_G, G_t)$. This sampling procedure corresponds to the imitative policy since the sampled returns are obtained from the learned density that models the data distribution. Following prior work in offline RL [16, 17, 18], we can also sample actions from an exponentially-tilted policy distribution. This is done by sampling the returns from the tilted distribution $G'_t \sim p_\theta(G_t|s_t, s_G)\exp(\kappa G_t)$, with G'_t being the tilted return-to-go and where κ represents the inverse temperature; higher values of κ concentrate more density around the best outcomes or higher returns, while negative values of κ concentrate on less favourable outcomes or lower returns.

We are interested in modelling and controlling the individual components of the reward function rather than maximizing their weighted sum. For example, we would like to model an agent's ability to reach its goal, drive on the road, and avoid collisions. In general, given C reward components, our objective is to learn policies that are conditioned on *all* its factored dimensions as this would grant us control over each one at test time. This entails modelling separate return components as $G_t^c \sim p_\theta(G_t^c|s_t, s_G)$ for each return component c. Applying this factorization, we reformulate the learned policy to explicitly account for the conditioning on all return components $\pi_\theta(a_t|s_t, s_G, G_t^1, \dots G_t^C)$. At test time, each return component will be accompanied by its own inverse temperature κ^c to

[1]Note that we omit the additional context m for brevity.

enable control over each return component, which enables sampling actions that adhere to different behaviours specified by $\{\kappa^1, \ldots, \kappa^C\}$, as shown in Algorithm 2 in Appendix B.

To implement our framework for behaviour simulation, we extend the approach presented above to the multi-agent setting. Across all agents we have sets for the joint states $\mathbb{S}_t$, goal states $\mathbb{S}_G$, actions $\mathbb{A}_t$, and returns-to-go $\mathbb{G}_t$. The final multi-agent joint distribution we model is:

$$p_\theta(\mathbb{S}_{t+1:T}, \mathbb{A}_t, \mathbb{G}_t | \mathbb{S}_t, \mathbb{S}_G) = p_\theta(\mathbb{S}_{t+1:T} | \mathbb{S}_t, \mathbb{S}_G, \mathbb{G}_t, \mathbb{A}_t)\pi_\theta(\mathbb{A}_t | \mathbb{S}_t, \mathbb{S}_G, \mathbb{G}_t)p_\theta(\mathbb{G}_t | \mathbb{S}_t, \mathbb{S}_G), \quad (1)$$

where the returns and actions from the previous timesteps are shared across agents, while at the present timestep they are masked out so one can only observe one's own return and action.

2.2 Multi-Agent Behaviour Simulation Architecture

In this section, we introduce the proposed architecture for multi-agent behaviour simulation within the CtRL-Sim framework that parameterizes the multi-agent joint distribution presented in Equation (1). We propose an encoder-decoder Transformer architecture [19], as illustrated in Figure 2, where the encoder encodes the initial scene and the decoder autoregressively generates the trajectory rollout for all agents in the scene.

Encoder To encode the initial scene, we first process the initial agent states and goals $(\mathbf{s}_0, \mathbf{s}_G)$ and the map context m, where $\mathbf{s}_0$ is the joint initial state of all agents and $\mathbf{s}_G$ is the joint goal state of all agents. Each agent i's initial state information s_0^i, which includes the position, velocity, heading, and agent type, is encoded with an MLP. Similarly, each agent's goal s_G^i, which is represented as the ground-truth final position, velocity, and heading, is also encoded with an MLP. We then concatenate the initial state and goal embedding of each agent and embed them with a linear layer to get per-agent embeddings of size d. We additionally apply an additive learnable embedding to encode the agents' identities across the sequence of agent embeddings. The map context is encoded using a polyline map encoder, detailed more fully in Appendix F, which yields L road segment embeddings of size d. The initial agent embeddings and road segment embeddings are then concatenated into a sequence of length $N + L$ and processed by a sequence of E Transformer encoder blocks.

Decoder The proposed decoder architecture models the joint distribution in Equation (1) as a sequence modelling problem, where we model the probability of the next token in the sequence conditioned on all previous tokens $p_\theta(x_t | x_{<t})$ [15]. In this work, we consider trajectory sequences of the form: $x = \langle \ldots, (s_t^1, s_G^1), (G_t^{1,1}, \ldots, G_t^{C,1}), a_t^1, ..., (s_t^N, s_G^N), (G_t^{1,N}, \ldots, G_t^{C,N}), a_t^N, \ldots \rangle$. These sequences are an extension of the sequences considered in the Multi-Game Decision Transformer [16] to the multi-agent goal-conditioned setting with factorized returns. Unlike Decision Transformer [15], our model predicts the return distribution and samples from it at inference time, which enables flexible control over the agent behaviours and circumvents the need to specify an expert return-to-go. We obtain state-goal tuple (s_t^i, s_G^i) embeddings in the same way that $(\mathbf{s}_0, \mathbf{s}_G)$ are processed in the encoder. Following recent work that tokenizes driving trajectories [20, 9], we discretize the actions and return-to-gos into uniformly quantized bins. We then embed the action and return-to-go tokens with a linear embedding. To each input token, we additionally add two learnable embeddings representing the agent identity and timestep, respectively. The tokenized sequence is then processed by D Transformer decoder layers with a temporally causal mask that is modified to ensure that the model is permutation equivariant to the agent ordering (see Appendix F for details).

Training Given a dataset of offline trajectories (Section 3), we train our model by sampling sequences of length $H \times N \times 3$, where H is the number of timesteps in the context. The state, return-to-go, and action token embeddings output by the decoder are used to predict the next return token, action token, and future state sequence, respectively. We train the return-to-go and action headers with the standard cross-entropy loss function and the future state sequence header with an L2 regression loss function. The final loss function is of the form: $\mathcal{L} = \mathcal{L}_{\text{action}} + \mathcal{L}_{\text{return-to-go}} + \alpha\mathcal{L}_{\text{state}}$.

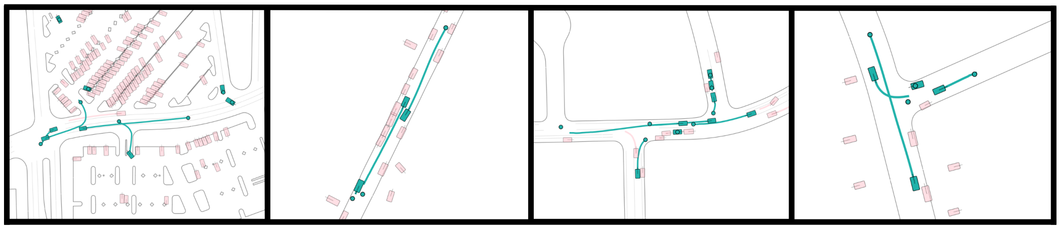


Figure 3: **Qualitative results of multi-agent simulation with CtRL-Sim.** The teal agents are controlled by CtRL-Sim, and other agents in pink are set to log-replay through physics.

Method	ADE (m) ↓	FDE (m) ↓	Goal Success Rate (%) ↑	JSD ($\times 10^{-2}$) ↓	Collision (%) ↓	Off Road (%) ↓	Per Scene Gen. Time (s) ↓
Replay-Physics*	0.47	0.97	87.3	7.6	2.8	10.7	1.1
Actions-Only [9]	4.81±0.52	11.89±1.42	32.7±1.4	10.4±0.3	19.9±1.2	27.6±1.0	3.3
Imitation Learning	1.24±0.05	1.95±0.10	77.4±1.3	8.3±0.1	5.8±0.2	12.1±0.2	3.4
DT (Max Return) [15]	1.56±0.04	3.07±0.16	63.3±0.8	8.4±0.1	5.3±0.3	11.0±0.2	20.7
CTG++† [11]	1.73±0.10	4.02±0.32	38.8±5.4	7.4±0.2	5.9±0.4	15.0±1.5	44.0
CtRL-Sim (No State Prediction)	1.32±0.03	2.21±0.06	72.4±0.8	8.2±0.2	6.1±0.4	12.0±0.3	
CtRL-Sim (Base)	1.29±0.04	2.13±0.08	73.0±1.3	8.1±0.2	5.8±0.4	11.8±0.2	8.2
CtRL-Sim (Positive Tilting)	1.25±0.03	2.04±0.08	72.9±1.5	7.9±0.1	5.3±0.2	11.0±0.2	
DT* (GT Initial Return)	1.10±0.02	1.58±0.07	77.5±1.5	8.4±0.1	5.3±0.3	11.9±0.3	20.8
CtRL-Sim* (GT Initial Return)	1.09±0.02	1.60±0.06	77.2±1.1	8.1±0.2	5.6±0.4	12.2±0.1	

Table 1: **Multi-agent simulation results over 1000 test scenes.** We report mean±std across 5 seeds. CtRL-Sim achieves a good balance between reconstruction performance, common sense, realism, and efficiency. * indicates privileged models requiring GT future. † indicates reimplementation.

3 Experiments

3.1 Experimental Setup

Offline Reinforcement Learning Dataset To train our model, we curate an offline reinforcement learning dataset derived from the Waymo Open Motion dataset [14]. We first extend Nocturne by integrating a physics engine based on the Box2D library for enabling realistic vehicle dynamics and collisions, detailed in Appendix B.1. Each scene in the Waymo dataset is fed through the physics-enhanced Nocturne simulator to compute the per-timestep actions and factored rewards for each agent. Refer to Appendix C for more details regarding the offline RL dataset collection.

Evaluation We evaluate CtRL-Sim on its ability to replicate the driving behaviours found in the Waymo Open Motion Dataset (*imitation*) and generate counterfactual scenes that are consistent with specified tilting coefficients (*controllability*). For both modes of evaluation, we use 1 second of history and simulate an 8 second future rollout. For *imitation*, we evaluate on up to 8 moving agents per scene that we control with CtRL-Sim, where the remaining agents are set to log replay through physics. We evaluate on 1000 random test scenes in both modes of evaluation. Following recent work [7], we use three types of metrics for imitation evaluation: *reconstruction* metrics, such as Final Displacement Error (**FDE**), Average Displacement Error (**ADE**), and **Goal Success Rate**; a *distributional realism* metric (**JSD**) defined by the mean of the Jensen-Shannon Distances computed on linear speed, angular speed, acceleration, and distance to nearest vehicle features between real and simulated scenes; and *common sense* metrics measured by **Collision** and **Offroad** rate.

For controllability evaluation, we evaluate on 1 selected "interesting" interactive agent that is controlled by CtRL-Sim, defined as an agent who is moving and whose goal is within 10 metres of another moving agent. All agents except for the CtRL-Sim-controlled interesting agent are set to log replay through physics. We evaluate the model's controllability through metrics aligned with the specified reward dimensions: we report the goal success rate for the goal reward control, collision rate for the vehicle-vehicle reward control, and offroad rate for the vehicle-road-edge reward control.

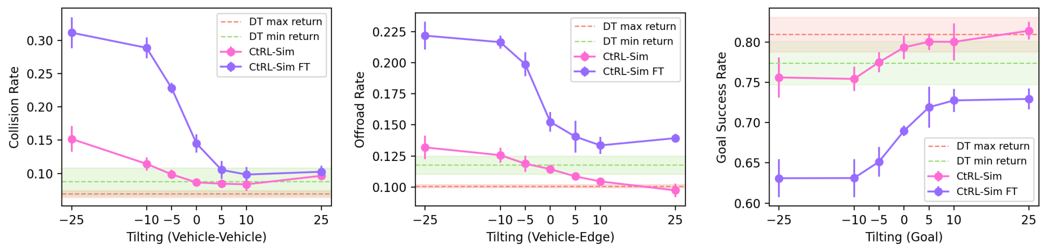


Figure 4: **Effects of exponential tilting.** Comparison of CtRL-Sim base model (magenta) and fine-tuned model (purple) across different reward dimensions. Rewards range from -25 to 25 for vehicle-vehicle collision (**left**), vehicle-edge collision (**middle**), and goal reaching (**right**). Results show smooth controllability, with fine-tuning enhancing this effect. We report mean$\pm$std over 5 seeds.

Methods under Comparison For imitation evaluation, we compare CtRL-Sim against several relevant baselines: **1.** *Replay-Physics* employs an inverse bicycle model to obtain the ground-truth log-replay actions and executes through the simulator. **2.** *Actions-Only* is an encoder-decoder model inspired by [9] where the decoder trajectory sequences only contain actions. **3.** *Imitation Learning (IL)* is identical to the architecture in Section 2.2 except with the removal of returns and the future state prediction. **4.** *Decision Transformer (DT)*: The *GT Initial Return* variant specifies the initial ground-truth return-to-go from the offline RL dataset, with the goal of acting as an imitative policy. *Max Return* follows the standard DT approach of selecting the maximum observable return in the dataset. The DT architecture is identical to that of CtRL-Sim except the return token precedes the state token, and the returns and future states are not predicted by the decoder. **4.** *CTG++* is a reimplementation of [11], a competitive Transformer-based diffusion model for behaviour simulation.

We evaluate the following variants of the proposed CtRL-Sim model: **1.** *CtRL-Sim (Base)* is the CtRL-Sim model trained on the offline RL dataset. **2.** *CtRL-Sim (No State Prediction)* is the base model trained without the state prediction task. **3.** *CtRL-Sim (Positive Tilting)* applies $\kappa_c = 10$ tilting to all components c of the base model. **4.** *CtRL-Sim (GT Initial Return)* is similar to *DT (GT Initial Return)*. For controllability evaluation, we evaluate on the CtRL-Sim base model and a finetuned CtRL-Sim model (*CtRL-Sim FT*). The finetuned model takes a trained base model and finetunes it on a dataset of simulated long-tail scenarios that we collect using an existing simulated collision generation method CAT [21]. This allows CtRL-Sim to be exposed to more long-tail collision scenarios during training, as the Waymo Open Motion dataset mainly contains nominal driving. We refer readers to Appendix H for details of CAT and our proposed finetuning procedure.

3.2 Results

In Table 1, we present the multi-agent imitation results comparing the CtRL-Sim model and its variants with imitation baselines. The CtRL-Sim models perform competitively with the imitation baselines, with the CtRL-Sim (Positive Tilting) model achieving a good balance between distributional realism (2nd in JSD), reconstruction performance (2nd in FDE, ADE), common sense (Tied 1st in Collision and Offroad Rate), and efficiency ($5.4\times$ faster than CTG++). Although DT (Max Return) attains equal collision and offroad rates as CtRL-Sim, this comes at the cost of substantially worse reconstruction performance. We further validate the importance of the future state prediction task, with CtRL-Sim (Base) outperforming CtRL-Sim (No State Prediction) across all metrics. The CtRL-Sim (Positive Tilting) model attains the best collision rate and offroad rate, demonstrating the effectiveness of exponential tilting for steering the model towards good driving behaviours.

We emphasize that a distinctive feature of CtRL-Sim that distinguishes it from the imitation baselines in Table 1 is that it additionally enables intuitive control over the agent behaviours through exponential tilting of the return distribution. This contrasts with DT, which, although capable of generating suboptimal behaviours by specifying low initial return-to-gos, lacks intuitive control due to the prerequisite knowledge about the return-to-go values and an absence of an interpretable mechanism

Adv. Method	Tilt	Reactive?	Control?	Planner Metrics: Progress (m) ↓	Planner Metrics: Coll. w/ Adv. (%) ↑	Adversary Realism: JSD ($\times 10^{-2}$) ↓	Adversary Realism: Coll. Speed (m/s) ↓
CAT		✗	✗	53.3	61.4	18.6	6.9
CtRL-Sim	-10	✓	✓	57.5±0.1	10±0.5	13.3±0.5	7.4±0.5
	10			57.7±0.1	8.7±0.5	12.7±0.4	8.3±0.6
CtRL-Sim FT	-10	✓	✓	56.1±0.2	33.8±1.9	19.6±0.4	6.3±0.2
	10			57.1±0.1	18.5±1.6	14.9±0.2	6.1±0.2
	50			57.4±0.2	14.6±1.8	15.6±1.2	6.0±0.3

Table 2: **Adversarial scenario generation results over 1000 test scenes.** We report the mean±std over 5 seeds for the CtRL-Sim models. Finetuning CtRL-Sim on CAT data improves ability to generate adversarial scenarios compared with base CtRL-Sim model. Compared with CAT, CtRL-Sim is reactive and controllable, while exhibiting better collision realism.

for behaviour modulation. By constrast, the exponential tilting of the predicted return distribution employed in CtRL-Sim has a clear interpretation: negative exponential tilting yields behaviours that are worse than the average behaviours learned from the dataset, while positive exponential tilting yields better-than-average behaviours. This provides a more intuitive interface to a practitioner who may aim to produce behaviours that are either less or more optimal than nominal driving behaviours.

We show the results of our controllability evaluation in Figure 4. For each reward dimension c, we exponentially tilt κ_c between -25 and 25 and observe how this affects the corresponding metric of interest. We also show the results of DT when conditioning on the minimum and maximum possible return. For both the base and finetuned CtRL-Sim models, we observe a relatively monotonic change in each metric of interest as the tilting coefficient is increased from -25 to 25. As the finetuned model is exposed to collision scenarios during finetuning, it demonstrates significant improvements over the base model in generating bad driving behaviours. Specifically, at -25 tilting, the finetuned model is able to generate $2.1\times$ as many collisions and $1.8\times$ as many offroad violations as the base model. Figure 5 (and 6, 7 in Appendix J) shows qualitatively the effects of tilting.

Table 2 evaluates CtRL-Sim's ability to produce adversarial agents that collide with a data-driven planner. We evaluate on a held-out test set of two-agent interactive scenarios from the Waymo dataset, where one interacting agent is controlled by the planner and the other is controlled by the adversary. We use a positively-tilted CtRL-Sim base model as our planner, due to its demonstrated ability to produce good driving behaviours in Table 1. For adversarial scenario generation, we compare the base CtRL-Sim model against the CtRL-Sim FT model. With -10 tilting applied, the finetuned model generates 238 more collisions with the planner than the base model over 1000 scenes, which we attribute to its exposure to simulated collision scenarios during finetuning. Notably, CtRL-Sim FT was finetuned in only 30 minutes on 1 NVIDIA A100-Large GPU and only 3500 CAT scenarios. This underscores CtRL-Sim's capability to flexibly incorporate data from various sources through finetuning, thereby enabling the generation of new kinds of driving behaviours. Importantly, after finetuning, CtRL-Sim FT largely retains its ability to produce good driving behaviours. This is evidenced by a 21.1 percentage point decrease in the planner's collision rate when using a +50 positively tilted finetuned model as the adversary.

In Table 2, we also compare CtRL-Sim against a state-of-the-art collision generation method, CAT [21], which uses a goal-conditioned trajectory forecasting model to select plausible adversarial trajectories that overlap with the ego plan. Although CAT generates more collisions with the planner, CAT is *not controllable* in that it cannot control the degree to which the agents are adversarial, which is a distinguishing feature of CtRL-Sim. Furthermore, CAT agents are *non-reactive* to the ego's actions as the trajectory is fixed at the beginning of the simulation, which severely limits the realism of agents controlled by CAT. This is evidenced by a larger adversary collision speed than all finetuned CtRL-Sim models and is also validated qualitatively in the attached supplementary videos. As collision realism is hard to quantitatively assess, we further conduct a user study to confirm that CtRL-Sim adversarial scenarios are indeed more realistic than CAT adversarial scenarios, with details and results reported in Appendix I.

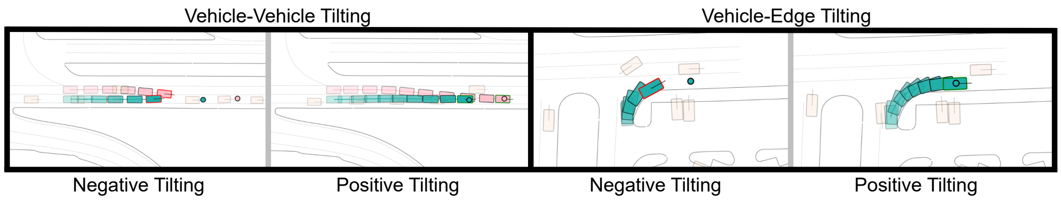


Figure 5: **Qualitative results of vehicle-vehicle and vehicle-edge tilting.** Two traffic scenes comparing positive tilting of the CtRL-Sim-controlled agent (shown in teal) with negative tilting for the same agent. Bounding boxes in red indicate traffic violations. Other agents log-replay through physics, with interacting agents in pink. Goals are marked by small circles.

4 Related Work

Agent behaviour simulation involves modelling the behaviour of other agents in simulation, such as vehicles and pedestrians, to enable diverse and realistic interactions with the AV. Agent behaviour simulation methods can be categorized into rule-based and data-driven methods. Rule-based methods rely on human-specified rules to produce plausible agent behaviours, such as adhering strictly to the center of the lane [2, 3]. These methods often yield unrealistic and rigid agent behaviours that fail to capture the full spectrum of driving behaviours. Moreover, we are most interested in modelling long-tail behaviours, which are difficult to model with rules alone.

To address these limitations, prior work has proposed learning generative models that aim to replicate agent behaviours found in real-world driving trajectory datasets [22, 23, 24, 4, 5, 25, 9]. These approaches draw significant inspiration from the extensive array of methods proposed for the task of joint motion prediction [26, 27, 28, 29, 30]; however, it's crucial to distinguish that, unlike the open-loop nature of joint motion prediction, behaviour simulation operates in a closed-loop manner [31]. To improve the realism of the learned behaviours, other work has proposed using adversarial imitation learning [32] to minimize the behavioural discrepancy between expert and model rollouts [33, 34, 6] or RL to improve traffic rule compliance [35, 36]. While such methods demonstrate improved realism over rule-based methods, they lack the necessary control over the behaviours to enable the generation of targeted simulation scenarios for AV testing.

More recent work has proposed more controllable behaviour simulation models by learning conditional models [10, 37, 7, 38, 8, 11, 12] that enable conditioning on a high-level latent variables [10, 37], route information [7], or differentiable constraints [8, 11, 39, 12, 40]. More recently, [41] used retrieval augmented generation to generate controllable traffic scenarios. However, these methods either lack interpretable control over the generated behaviours [37] or require costly test-time optimization procedures to steer the generated behaviours, such as latent variable optimization [10], Bayesian optimization [42, 43, 7], or the simulation of expensive diffusion processes [8, 44, 11, 39, 12, 40]. CtRL-Sim takes an alternative approach and learns a conditional multi-agent behaviour model that conditions on interpretable factorized returns. By exponentially tilting the predicted return distribution [16] at test time, CtRL-Sim enables *efficient, interpretable, and fine-grained control* over agent behaviours while being grounded in real-world data.

5 Conclusions

We presented CtRL-Sim, a novel framework applying offline RL for controllable and reactive behaviour simulation. Our proposed multi-agent behaviour Transformer architecture allows CtRL-Sim to employ exponential tilting at test time to simulate a wide range of interesting agent behaviours. We present experiments showing the effectiveness of CtRL-Sim at producing controllable and reactive behaviours, while maintaining competitive performance on the imitation task compared to baselines. We hope CtRL-Sim can be further explored in future work to handle more reward function components, such as driving comfort and respecting traffic signalization, as well as explored in domains outside of autonomous driving.

References

[1] K. D. Kusano, J. M. Scanlon, Y. Chen, T. L. McMurry, R. Chen, T. Gode, and T. Victor. Comparison of waymo rider-only crash data to human benchmarks at 7.1 million miles. *arXiv preprint arXiv.2312.12675*, 2023.

[2] M. Treiber, A. Hennecke, and D. Helbing. Congested traffic states in empirical observations and microscopic simulations. *Physical review E*, 62(2):1805, 2000.

[3] A. Kesting, M. Treiber, and D. Helbing. General lane-changing model mobil for car-following models. *Transportation Research Record*, 1999(1):86–94, 2007.

[4] S. Suo, S. Regalado, S. Casas, and R. Urtasun. Trafficsim: Learning to simulate realistic multi-agent behaviors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[5] D. Xu, Y. Chen, B. Ivanovic, and M. Pavone. BITS: bi-level imitation for traffic simulation. In *Proceedings of the International Conference on Robotics and Automation (ICRA)*, 2023.

[6] M. Igl, D. Kim, A. Kuefler, P. Mougin, P. Shah, K. Shiarlis, D. Anguelov, M. Palatucci, B. White, and S. Whiteson. Symphony: Learning realistic and diverse agents for autonomous driving simulation. In *Proceedings of the International Conference on Robotics and Automation (ICRA)*, 2022.

[7] S. Suo, K. Wong, J. Xu, J. Tu, A. Cui, S. Casas, and R. Urtasun. MIXSIM: A hierarchical framework for mixed reality traffic simulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[8] Z. Zhong, D. Rempe, D. Xu, Y. Chen, S. Veer, T. Che, B. Ray, and M. Pavone. Guided conditional diffusion for controllable traffic simulation. In *Proceedings of the International Conference on Robotics and Automation (ICRA)*, 2023.

[9] J. Philion, X. B. Peng, and S. Fidler. Trajeglish: Learning the language of driving scenarios. *arXiv preprint arXiv.2312.04535*, 2023.

[10] D. Rempe, J. Philion, L. J. Guibas, S. Fidler, and O. Litany. Generating useful accident-prone driving scenarios via a learned traffic prior. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[11] Z. Zhong, D. Rempe, Y. Chen, B. Ivanovic, Y. Cao, D. Xu, M. Pavone, and B. Ray. Language-guided traffic simulation via scene-level diffusion. *arXiv preprint arXiv.2306.06344*, 2023.

[12] W. Chang, F. Pittaluga, M. Tomizuka, W. Zhan, and M. Chandraker. Controllable safety-critical closed-loop traffic simulation via guided diffusion. *arXiv preprint arXiv.2401.00391*, 2024.

[13] E. Vinitsky, N. Lichtlé, X. Yang, B. Amos, and J. Foerster. Nocturne: a scalable driving benchmark for bringing multi-agent learning one step closer to the real world. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

[14] S. Ettinger, S. Cheng, B. Caine, C. Liu, H. Zhao, S. Pradhan, Y. Chai, B. Sapp, C. R. Qi, Y. Zhou, Z. Yang, A. Chouard, P. Sun, J. Ngiam, V. Vasudevan, A. McCauley, J. Shlens, and D. Anguelov. Large scale interactive motion forecasting for autonomous driving : The waymo open motion dataset. In *Proceedings of the IEEE/CVF International Conference on Computer Vision, (ICCV)*, 2021.

[15] L. Chen, K. Lu, A. Rajeswaran, K. Lee, A. Grover, M. Laskin, P. Abbeel, A. Srinivas, and I. Mordatch. Decision transformer: Reinforcement learning via sequence modeling. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[16] K. Lee, O. Nachum, M. Yang, L. Lee, D. Freeman, S. Guadarrama, I. Fischer, W. Xu, E. Jang, H. Michalewski, and I. Mordatch. Multi-game decision transformers. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

[17] N. Gontier, P. R. López, I. H. Laradji, D. Vázquez, and C. J. Pal. Language decision transformers with exponential tilt for interactive text environments. *arXiv preprint arXiv.2302.05507*, 2023.

[18] A. Piché, R. Pardinas, D. Vazquez, and C. Pal. A probabilistic perspective on reinforcement learning via supervised learning. In *ICLR 2022 Workshop on Generalizable Policy Learning in Physical World*, 2022.

[19] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

[20] A. Seff, B. Cera, D. Chen, M. Ng, A. Zhou, N. Nayakanti, K. S. Refaat, R. Al-Rfou, and B. Sapp. Motionlm: Multi-agent motion forecasting as language modeling. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023.

[21] L. Zhang, Z. Peng, Q. Li, and B. Zhou. CAT: closed-loop adversarial training for safe end-to-end driving. In *Conference on Robot Learning, CoRL 2023, 6-9 November 2023, Atlanta, GA, USA*, volume 229 of *Proceedings of Machine Learning Research*, pages 2357–2372. PMLR, 2023.

[22] L. Bergamini, Y. Ye, O. Scheel, L. Chen, C. Hu, L. D. Pero, B. Osinski, H. Grimmett, and P. Ondruska. Simnet: Learning reactive self-driving simulations from real-world observations. In *Proceedings of the International Conference on Robotics and Automation (ICRA)*, 2021.

[23] A. Kamenev, L. Wang, O. B. Bohan, I. Kulkarni, B. Kartal, A. Molchanov, S. Birchfield, D. Nistér, and N. Smolyanskiy. Predictionnet: Real-time joint probabilistic traffic prediction for planning, control, and simulation. In *Proceedings of the International Conference on Robotics and Automation (ICRA)*, 2022.

[24] A. Ścibior, V. Lioutas, D. Reda, P. Bateni, and F. Wood. Imagining the road ahead: Multi-agent trajectory prediction via differentiable simulation. In *Proceedings of the IEEE International Intelligent Transportation Systems Conference (ITSC)*, 2021.

[25] Y. Wang, T. Zhao, and F. Yi. Multiverse transformer: 1st place solution for waymo open sim agents challenge 2023. *arXiv preprint arXiv.2306.11868*, 2023.

[26] S. Casas, C. Gulino, S. Suo, K. Luo, R. Liao, and R. Urtasun. Implicit latent variable model for scene-consistent motion forecasting. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020.

[27] R. Girgis, F. Golemo, F. Codevilla, M. Weiss, J. A. D'Souza, S. E. Kahou, F. Heide, and C. J. Pal. Latent variable sequential set transformers for joint multi-agent motion prediction. In *International Conference on Learning Representations*, 2021. URL https://api.semanticscholar.org/CorpusID:246824069.

[28] J. Ngiam, V. Vasudevan, B. Caine, Z. Zhang, H. L. Chiang, J. Ling, R. Roelofs, A. Bewley, C. Liu, A. Venugopal, D. J. Weiss, B. Sapp, Z. Chen, and J. Shlens. Scene transformer: A unified architecture for predicting future trajectories of multiple agents. In *International Conference on Learning Representations (ICLR)*, 2022.

[29] Y. Chen, B. Ivanovic, and M. Pavone. Scept: Scene-consistent, policy-based trajectory predictions for planning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[30] L. Rowe, M. Ethier, E.-H. Dykhne, and K. Czarnecki. FJMP: factorized joint multi-agent motion prediction over learned directed acyclic interaction graphs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[31] H. Caesar, J. Kabzan, K. S. Tan, W. K. Fong, E. M. Wolff, A. H. Lang, L. Fletcher, O. Beijbom, and S. Omari. nuplan: A closed-loop ml-based planning benchmark for autonomous vehicles. *arXiv preprint arXiv.2106.11810*, 2021.

[32] J. Ho and S. Ermon. Generative adversarial imitation learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2016.

[33] R. P. Bhattacharyya, D. J. Phillips, B. Wulfe, J. Morton, A. Kuefler, and M. J. Kochenderfer. Multi-agent imitation learning for driving simulation. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018.

[34] G. Zheng, H. Liu, K. Xu, and Z. Li. Objective-aware traffic simulation via inverse reinforcement learning. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2021.

[35] Y. Lu, J. Fu, G. Tucker, X. Pan, E. Bronstein, R. Roelofs, B. Sapp, B. White, A. Faust, S. Whiteson, D. Anguelov, and S. Levine. Imitation is not enough: Robustifying imitation with reinforcement learning for challenging driving scenarios. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2023.

[36] C. Zhang, J. Tu, L. Zhang, K. Wong, S. Suo, and R. Urtasun. Learning realistic traffic agents in closed-loop. In *Conference on Robot Learning, CoRL 2023, 6-9 November 2023, Atlanta, GA, USA*, volume 229 of *Proceedings of Machine Learning Research*, pages 800–821. PMLR, 2023.

[37] Z. Zhang, A. Liniger, D. Dai, F. Yu, and L. V. Gool. Trafficbots: Towards world models for autonomous driving simulation and motion prediction. In *Proceedings of the International Conference on Robotics and Automation (ICRA)*, 2023.

[38] W. Chang, C. Tang, C. Li, Y. Hu, M. Tomizuka, and W. Zhan. Editing driver character: Socially-controllable behavior generation for interactive traffic simulation. *IEEE Robotics Autom. Lett.*, 2023.

[39] C. Xu, D. Zhao, A. Sangiovanni-Vincentelli, and B. Li. Diffscene: Diffusion-based safety-critical scenario generation for autonomous vehicles. In *The Second Workshop on New Frontiers in Adversarial Machine Learning*, 2023.

[40] Z. Guo, X. Gao, J. Zhou, X. Cai, and B. Shi. Scenedm: Scene-level multi-agent trajectory generation with consistent diffusion models. *arXiv preprint: arXiv.2311.15736*, 2023.

[41] W. Ding, Y. Cao, D. Zhao, C. Xiao, and M. Pavone. Realgen: Retrieval augmented generation for controllable traffic scenarios. *arXiv preprint arXiv.2312.13303*, 2023.

[42] Y. Abeysirigoonawardena, F. Shkurti, and G. Dudek. Generating adversarial driving scenarios in high-fidelity simulators. In *International Conference on Robotics and Automation (ICRA)*, 2019.

[43] J. Wang, A. Pun, J. Tu, S. Manivasagam, A. Sadat, S. Casas, M. Ren, and R. Urtasun. Advsim: Generating safety-critical scenarios for self-driving vehicles. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[44] C. M. Jiang, A. Cornman, C. Park, B. Sapp, Y. Zhou, and D. Anguelov. Motiondiffuser: Controllable multi-agent motion prediction using diffusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[45] J. Lee, Y. Lee, J. Kim, A. R. Kosiorek, S. Choi, and Y. W. Teh. Set transformer: A framework for attention-based permutation-invariant neural networks. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2019.

[46] A. Ibrahim, B. Thérien, K. Gupta, M. L. Richter, Q. Anthony, T. Lesort, E. Belilovsky, and I. Rish. Simple and scalable strategies to continually pre-train large language models. *arXiv preprint arXiv.2403.08763*, 2024.