

ReinDriveGen: Reinforcement Post-Training for Out-of-Distribution Driving Scene Generation

Hao Zhang^{1*}, Lue Fan^{1,2*}, Weikang Bian¹, Zehuan Wu³, Lewei Lu³, Zhaoxiang Zhang², and Hongsheng Li^{1,4,5}

¹CUHK MMLab ²CASIA ³SenseTime Research

⁴Ace Robotics ⁵CPII under InnoHK

<https://drive-sim.github.io/ReinDriveGen/>

hzhangcc@link.cuhk.edu.hk, lue.fan@ia.ac.cn, hsli@ee.cuhk.edu.hk

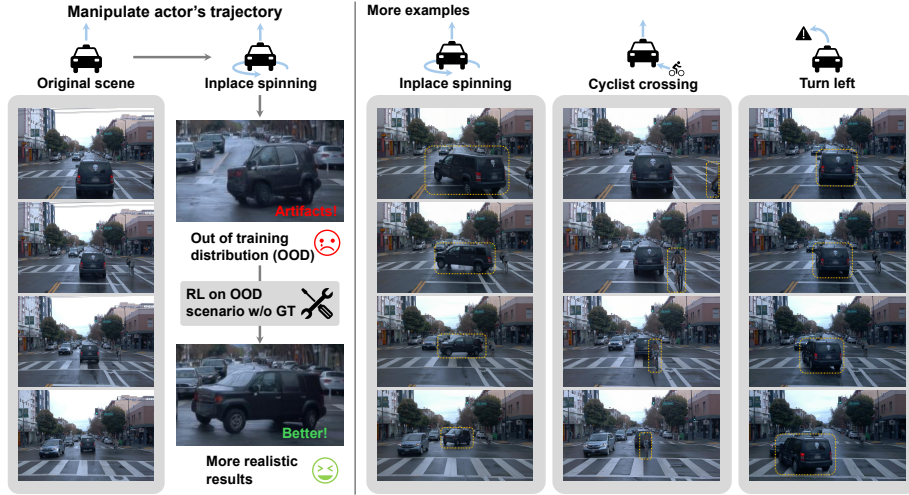


Fig. 1: ReinDriveGen enables photorealistic generation of OOD driving scenarios. ReinDriveGen can manipulate actor trajectories to synthesize diverse safety-critical corner cases such as vehicle in-place spinning, cyclist crossing, and left-turn collisions. Our RL-based post-training significantly improves generation quality for these OOD edits without requiring ground-truth supervision.

Abstract. We present ReinDriveGen, a framework that enables full controllability over dynamic driving scenes, allowing users to freely edit actor trajectories to simulate safety-critical corner cases such as front-vehicle collisions, drifting cars, vehicles spinning out of control, pedestrians jay-walking, and cyclists cutting across lanes. Our approach constructs a dynamic 3D point cloud scene from multi-frame LiDAR data, introduces a vehicle completion module to reconstruct full 360° geometry from partial observations, and renders the edited scene into 2D condition images that guide a video diffusion model to synthesize realistic driving videos.

* Equal contribution.

Since such edited scenarios inevitably fall outside the training distribution, we further propose an RL-based post-training strategy with a pairwise preference model and a pairwise reward mechanism, enabling robust quality improvement under out-of-distribution conditions without ground-truth supervision. Extensive experiments demonstrate that ReinDriveGen outperforms existing approaches on edited driving scenarios and achieves state-of-the-art results on novel ego viewpoint synthesis.

1 Introduction

Photorealistic driving scene simulation is indispensable for the development and validation of autonomous driving systems. A high-fidelity simulator allows arbitrary editing of driving scenes to generate diverse and realistic scenarios, including rare but safety-critical corner cases such as sudden cut-ins, near-collisions, vehicle drifting, and pedestrians jaywalking, thereby facilitating efficient data augmentation and thorough evaluation of driving policies.

A central challenge in building such a simulator lies in the inevitable **distribution gap** between training and inference. Current approaches, whether video generation-based [5, 17, 18] or reconstruction-based [4, 11, 15, 16, 30, 35], are fundamentally trained or optimized on recorded driving data. At inference time, however, a useful simulator must support arbitrary edits, such as repositioning vehicles, altering trajectories, and rendering from novel ego viewpoints, all of which produce conditions that deviate significantly from the training distribution. This gap is further exacerbated for safety-critical corner cases such as vehicles spinning in place, rollovers, and head-on collisions, which are inherently rare or entirely absent in normal driving logs, leaving the model with few or no corresponding training examples. Standard supervised fine-tuning (SFT) cannot close this gap, because ground-truth video only exists for the original recorded trajectories and viewpoints; no real reference can be obtained for any edited configuration. This out-of training distribution (OOD) degradation is most pronounced on vehicles, as they are the primary objects being edited and thus undergo the most drastic distribution shift, making vehicle quality the primary bottleneck for realistic simulation of safety-critical corner cases.

To address this fundamental challenge, we propose ReinDriveGen, a unified framework whose core insight is to leverage **reinforcement learning (RL) based post-training** to improve generation quality under OOD conditions, going beyond what SFT alone can achieve. Our framework comprises two components. The first is a Point Cloud Conditioned Video Diffusion Simulator. We aggregate and colorize multi-frame LiDAR point clouds to build a dynamic 3D scene, and introduce a vehicle completion module to reconstruct the full 360° geometry of partially observed vehicles. The completed scene is then rendered into 2D pseudo-images that serve as conditions for a video diffusion model to synthesize photorealistic driving videos.

The second is RL-based post-training, which finetunes the video diffusion model to bridge the distribution gap that SFT cannot close, enabling high-quality generation under OOD conditions. We extend DiffusionNFT [36] from

image generation to video generation as our RL post-training framework. For each OOD condition, the framework generates a group of candidate videos and uses a reward model to partition them into positive and negative subsets, defining a contrastive improvement direction via flow matching [12]. A critical question is how to design the reward model, as no existing reward model is tailored to evaluate the quality of generated driving vehicles. A straightforward approach is to train a pointwise reward model that assigns an absolute scalar score to each sample under the Bradley-Terry preference framework. However, such pointwise scoring is susceptible to reward hacking, where negligible score differences are disproportionately amplified into misleading training signals [27]. Moreover, pairwise comparison provides more robust judgments than evaluating each sample in isolation, as the model can directly contrast two samples rather than relying on absolute scores prone to systematic biases. Building on this insight, we curate a dataset of positive and negative driving vehicle pairs and train a dedicated **pairwise preference model**. To convert pairwise outcomes into per-sample rewards, we propose a **pairwise reward mechanism**: for each group of candidates, we perform all pairwise comparisons, and a sample’s reward is determined by how many comparisons it wins, with frequent winners receiving high rewards and frequent losers being penalized accordingly. This relative evaluation scheme avoids the pitfalls of absolute scoring and yields robust reward signals. Extensive experiments demonstrate that our RL-based post-training brings substantial quality improvements on OOD edited scenarios.

Our main contributions are summarized as follows: (1) We propose a Point Cloud-Conditioned Video Diffusion Simulator capable of simulating edited actor trajectories and novel ego-vehicle viewpoints. (2) We extend RL-based post-training from image to video generation and propose a pairwise reward mechanism with a dedicated pairwise preference model, yielding robust training signals that effectively address vehicle quality degradation under out-of-distribution driving scenarios.

2 Related Work

Reconstruction-Based Driving Simulators. Neural Radiance Fields [15] and 3D Gaussian Splatting [11] have driven rapid progress in 3D scene reconstruction, and numerous works have adapted these representations to dynamic driving scenarios [3, 7, 8, 22, 23, 28, 29, 31, 37]. However, these reconstruction-centric methods are intrinsically limited to the training trajectory and struggle to extrapolate high-fidelity views when the camera deviates significantly from the recorded path. Recent hybrid approaches [4, 16, 30, 35] integrate video diffusion priors to improve novel-viewpoint rendering, yet balancing generation and reconstruction remains non-trivial, often leading to geometric artifacts or blurred details. More fundamentally, all reconstruction-based methods cannot properly handle vehicle trajectory editing: repositioning a vehicle exposes previously unobserved regions that produce visible holes, while baked-in visual effects such as lighting and shadows cannot be re-synthesized for the edited configuration.

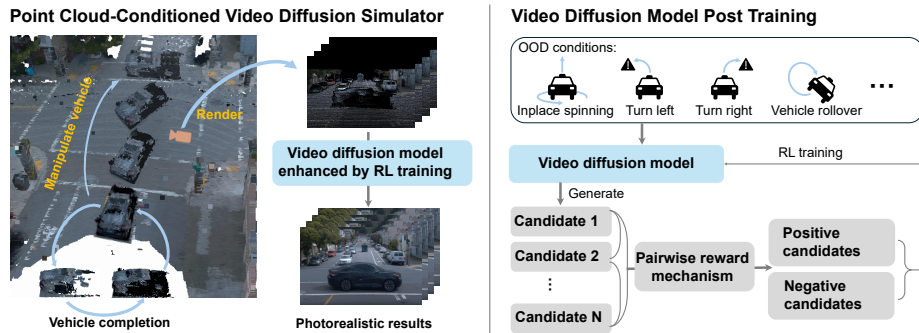


Fig. 2: Overview of ReinDriveGen’s two core components. **Left:** Our simulator edits dynamic point clouds and completes vehicle geometry to render structural pseudo-images, which condition the video diffusion model for photorealistic synthesis. **Right:** To enhance quality in out-of-distribution (OOD) scenarios, we employ RL post-training. A pairwise reward mechanism ranks generated candidates into positive and negative sets, providing robust contrastive supervision.

Diffusion-Based Driving Simulators. In parallel with reconstruction-based methods, diffusion models have emerged as powerful tools for driving simulation. Approaches such as MagicDrive-V2 [5], Vista [6], MaskGWM [17], and Cosmos [18] synthesize driving videos from structured inputs such as text prompts, HD maps, and bounding boxes. However, these video-generation models lack an explicit underlying 3D representation, which precludes precise control over camera trajectories and makes it difficult to maintain consistent 3D scene structure across frames. Furthermore, due to the inherent stochasticity of the diffusion sampling process, the generated content often varies significantly across different runs, making it difficult to perform multiple rounds of distinct edits—such as modifying different vehicles’ trajectories—while keeping the rest of the scene fixed and consistent. More fundamentally, constrained by their training data distribution, these methods struggle to faithfully simulate unusual driving behaviors such as vehicle drifting or in-place spins that are rarely represented in standard driving datasets. Although FreeVS [24] attempts to mitigate control issues by leveraging LiDAR point clouds for geometric guidance, it remains limited by the sensor’s range and fails to recover visual content outside the LiDAR scan, leading to incomplete scene synthesis. Asset insertion methods such as R3D2 [14] and Mirage [25] can augment recorded driving videos by compositing vehicle assets into real scenes. However, they mainly insert new objects without modifying existing scene dynamics, and often struggle with occlusion reasoning and interactions with the background. Therefore, they are more suitable for visual augmentation than controllable driving simulation.

3 Method

Our method consists of two core components as illustrated in Fig. 2: a Point Cloud-Conditioned Video Diffusion Simulator (Sec. 3.1) and an RL-based post-training framework (Sec. 3.2).

3.1 Point Cloud-Conditioned Video Diffusion Simulator

Without an explicit 3D scene representation, video diffusion models often struggle to maintain multi-view geometric consistency, and it remains challenging to perform multiple rounds of targeted scene edits while keeping the rest of the environment intact. We therefore adopt dynamic 3D point clouds as our 3D backbone and render them into 2D pseudo-images that provide pixel-level conditions for a video diffusion model, inspired by [24, 30].

Dynamic 3D Point Cloud Construction. A single LiDAR scan is too sparse to serve as a reliable condition. We therefore aggregate points across frames in a category-aware manner, leveraging the annotated 3D bounding boxes provided by Waymo [21] for all dynamic actors (vehicles, pedestrians, and cyclists). For the static background, we accumulate all 200 frames of each scene into a unified world coordinate system. For each vehicle, we transform its per-frame LiDAR points into the object’s canonical frame using the annotated bounding boxes and aggregate all frames. Pedestrians and cyclists are deformable, so we only aggregate ± 2 frames (5 in total) around each time step. Each point is colorized by projecting it onto the corresponding camera image and retrieving the RGB value.

Vehicle Completion. Even after multi-frame aggregation, vehicles are often only partially observed due to occlusion and limited recorded viewpoints. We apply AdaPoinTr [33], a point cloud completion method, to reconstruct full 360° vehicle geometry. The model is initialized from a ShapeNet-55 pre-trained checkpoint and fine-tuned on Projected ShapeNet-55 (car category only; see supplementary for details). After completion, each vehicle can be freely repositioned and observed from arbitrary viewpoints without geometric holes, ensuring multi-view geometric consistency across edits. Note that the completed points carry no color information and are assigned black; we leave the texture synthesis of these surfaces to the downstream video diffusion model.

Scene Editing and Rendering. With the dynamic 3D point cloud augmented by completed vehicles in place, we can freely edit actor trajectories and ego-vehicle viewpoints within the reconstructed scene. The edited point cloud is then rendered into 2D pseudo-images via projection; regions not covered by any LiDAR point are left black.

Conditional Video Generation. The pseudo-images rendered from the edited point cloud provide geometric guidance but are inevitably noisy and incomplete due to LiDAR sparsity, multi-frame accumulation artifacts, and untextured completed surfaces. A video diffusion model is employed to transform these coarse renderings into photorealistic driving videos.

To ensure that the regions beyond LiDAR coverage remain visually consistent across different edits, we anchor each generated sequence to a recorded frame

from the original capture, which serves as a reference image providing appearance priors. Conditioned on both the reference image and the rendered pseudo-image sequence, the video diffusion model fills in LiDAR-unobserved regions guided by the reference image, corrects multi-frame accumulation artifacts, synthesizes realistic textures on the completed but untextured vehicle surfaces, and generates plausible lighting and shadow effects.

We adopt the VACE-1.3B [10] model as our video diffusion backbone. The reference image and pseudo-image sequence are injected through VACE’s Video Condition Unit (VCU) as a composition of Reference-to-Video Generation (R2V) and Video-to-Video Editing (V2V). We jointly fine-tune the Context Adapter and the DiT on the Waymo dataset using paired pseudo-image sequences and recorded ground-truth videos. At inference time, novel ego-vehicle viewpoints often expose completed but untextured vehicle surfaces that never appear under the original recorded views. To improve robustness to such cases, we adopt the idea of pseudo-view simulation from GA-Drive [34]: we randomly mask regions adjacent to depth discontinuities, simulating the black untextured areas that emerge under novel viewpoints. We further apply random rectangular block masking to handle arbitrary occlusion patterns. Training videos are 49 frames at 480×832 resolution, and the text prompt is fixed as “A realistic autonomous driving scene.”

3.2 RL-based Post-training

The supervised fine-tuning (SFT) stage described in Sec. 3.1 trains the video diffusion model on pairs of rendered pseudo-images and ground-truth videos, both captured under the original recorded ego-vehicle viewpoints and actor trajectories. At inference time, however, we wish to edit vehicle trajectories (*e.g.*, simulating lead-vehicle collisions, drifting, spinning or other counterfactual maneuvers) or render from novel ego-vehicle viewpoints. In both cases, the rendered pseudo-image conditions deviate significantly from those seen during SFT, exhibiting novel occlusion patterns, unfamiliar vehicle orientations, and untextured completed surfaces, which creates a substantial domain gap.

Empirically, we observe that vehicles are the primary source of visual degradation under these out-of-distribution (OOD) conditions, exhibiting distorted geometry, unrealistic textures, and implausible lighting, while static background regions remain largely unaffected. To address this limitation, we propose an RL-based post-training framework that directly optimizes vehicle generation quality under OOD conditions without requiring paired ground-truth supervision.

Extending DiffusionNFT to Video Generation. We adopt DiffusionNFT [36] as our RL post-training framework, which achieves significantly faster training than FlowGRPO [13] by performing policy optimization directly on the forward diffusion process in a classifier-free-guidance-free (CFG-free) manner. This efficiency advantage is particularly important for video diffusion RL, where the high dimensionality of spatiotemporal outputs makes both sampling and training substantially more expensive than in the image setting.



Fig. 3: Qualitative demonstrations of ReinDriveGen on OOD safety-critical corner cases. Every two consecutive rows show two frames from the same scene, with the edit type and edited region annotated in each image.

We extend DiffusionNFT from image to video generation. Let π_θ denote the video diffusion policy parameterized by θ , and let $\mathbf{c}$ denote a condition consisting of a reference frame and a rendered pseudo-image sequence. We maintain a reference policy π^{old} (updated via EMA) alongside the current policy π_θ . In each iteration, we sample a batch of OOD conditions $\{\mathbf{c}_i\}$. For each condition, the reference policy generates a group of N candidate videos. A reward model scores each candidate, and the scores are used to optimize the following contrastive objective adapted from DiffusionNFT:

$$\mathcal{L}(\theta) = \mathbb{E}_{\mathbf{c}, \pi^{\text{old}}(\mathbf{x}_0|\mathbf{c}), t} \left[r \|v_\theta^+(\mathbf{x}_t, \mathbf{c}, t) - \mathbf{v}\|_2^2 + (1 - r) \|v_\theta^-(\mathbf{x}_t, \mathbf{c}, t) - \mathbf{v}\|_2^2 \right], \quad (1)$$

where $\mathbf{x}_0$ is a clean generated video sampled from π^{old} , $\mathbf{x}_t$ is its noised version at flow-matching timestep t , $\mathbf{v}$ is the flow-matching velocity target, $r \in [0, 1]$ is the reward associated with $\mathbf{x}_0$, and v_θ^+ , v_θ^- are the implicit positive and negative velocity predictions as defined in [36]. Intuitively, for high-reward samples ($r \rightarrow 1$) the loss encourages v_θ to reproduce similar high-quality outputs, while for low-reward samples ($r \rightarrow 0$) it steers v_θ away from low-quality outputs.

Pairwise Preference Model. A critical component of RL post-training is a reward model that evaluates the quality of vehicles in generated driving videos. No existing reward model serves this purpose; general-purpose image quality or aesthetics models fail to capture domain-specific artifacts such as distorted vehicle geometry, unrealistic textures, or implausible lighting patterns. We therefore train a dedicated pairwise preference model to evaluate generated vehicle quality.

Preference data construction. To train the pairwise preference model, we need paired data of relatively better and worse vehicle crops. We exploit a key empirical observation: when generating videos with the SFT model, subsampling the pseudo-image conditioning (*i.e.*, providing a rendered pseudo-image every k -th frame instead of every frame) progressively degrades the visual quality of vehicles in the generated output. This provides a natural and controllable mechanism for constructing preference pairs without manual annotation. Concretely, for each scene in the Waymo training dataset, we generate three versions of the output video using the SFT model with pseudo-image conditioning provided at (i) every frame, (ii) every 2nd frame, and (iii) every 4th frame. Since we focus exclusively on vehicle quality, we apply YOLO26 [19] to detect and crop all vehicle instances from each generated frame. For a given vehicle instance in a given frame, the crops from different conditioning frequencies form naturally ordered preference pairs: (i) $\succ$ (ii), (ii) $\succ$ (iii), and (i) $\succ$ (iii), all paired on the same vehicle, same frame, and same scene, ensuring that the vehicle quality is the only varying factor.

To further diversify the degradation patterns and improve robustness, we augment both positive and negative samples with motion blur and elastic distortion, where the degradation strength applied to negative samples is always strictly greater than that applied to positive samples. This asymmetric augmentation ensures that the reward model learns to discriminate between relatively better and worse samples even when both exhibit significant artifacts. This property is essential for the early stages of RL training, where the policy generates low-quality outputs and the reward model must still provide a meaningful ranking among candidates to guide optimization. Further details are provided in the supplementary material.

Model architecture. Given two vehicle crops I_1 and I_2 , we extract their CLS-token representations using a shared DINOv3 [20] ViT-H+ backbone f , yielding $\mathbf{z}_1 = f(I_1)$ and $\mathbf{z}_2 = f(I_2)$ with $\mathbf{z}_1, \mathbf{z}_2 \in \mathbb{R}^d$. The two feature vectors are concatenated in both orders and passed through a shared MLP head h to produce scalar logits $s_{12} = h([\mathbf{z}_1; \mathbf{z}_2])$ and $s_{21} = h([\mathbf{z}_2; \mathbf{z}_1])$. The preference probability is then defined via an antisymmetric formulation:

$$P(I_1 \succ I_2) = \sigma\left(\frac{s_{12} - s_{21}}{2}\right), \quad (2)$$

where σ denotes the sigmoid function. This design guarantees logical consistency, *i.e.*, $P(I_1 \succ I_2) = 1 - P(I_2 \succ I_1)$, since $\sigma(-x) = 1 - \sigma(x)$. The pairwise preference model is trained with binary cross-entropy loss on the constructed preference pairs.

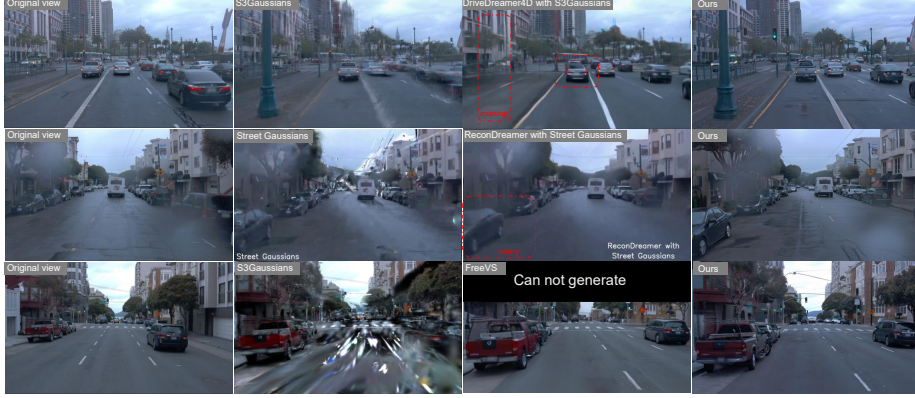


Fig. 4: Qualitative comparison of novel trajectories in the lane-change scenario.

Pairwise Reward Mechanism. Given the trained pairwise preference model, we propose a pairwise reward mechanism to aggregate pairwise comparison outcomes into per-candidate rewards within each group, providing robust training signals for RL-based post-training.

For each OOD condition $\mathbf{c}$, we generate $N = 16$ candidate videos $\{\mathbf{x}^1, \dots, \mathbf{x}^N\}$. Each candidate video is decomposed into individual frames. We run YOLO26 [19] on the first candidate to detect and localize all vehicles in every frame, yielding a set of bounding boxes. These bounding boxes are then applied identically to all N candidates, so that exactly the same spatial region is cropped across candidates for fair comparison. We treat each (frame index, bounding box) tuple as an independent evaluation unit; let M denote the total number of such units across all frames. For each unit m , we crop the corresponding region from every candidate, producing N crops $\{a_m^1, \dots, a_m^N\}$. We then perform all $\binom{N}{2}$ pairwise comparisons using our pairwise preference model (Eq. 2). For each pair (a_m^i, a_m^j) , the model outputs a preference probability $P(a_m^i \succ a_m^j)$. The per-unit reward for candidate i at unit m is defined as its win rate:

$$R_m^i = \frac{1}{N-1} \sum_{j \neq i} \mathbb{1}[P(a_m^i \succ a_m^j) > \tau], \quad (3)$$

where $\tau = 0.85$ is a confidence threshold that filters out ambiguous comparisons. The video-level reward $\hat{R}(\mathbf{x}^i)$ is computed as an area-weighted average of the per-unit win rates across all M units, where each unit is weighted by its bounding-box pixel area so that artifacts on vehicles with larger area are prioritized. Compared to pointwise scores, pairwise win rates span a wider and more discriminative range and are inherently robust to systematic biases in the reward model. The resulting $\hat{R}(\mathbf{x}^i)$ is used in place of the pointwise reward r in the DiffusionNFT loss (Eq. 1), which is seamless as DiffusionNFT only requires a scalar reward per sample.

With the pairwise reward mechanism in place, we can now perform RL post-training on OOD conditions, such as lead-vehicle collisions, drifting, spinning, and counterfactual trajectories that place vehicles in extreme orientations. In

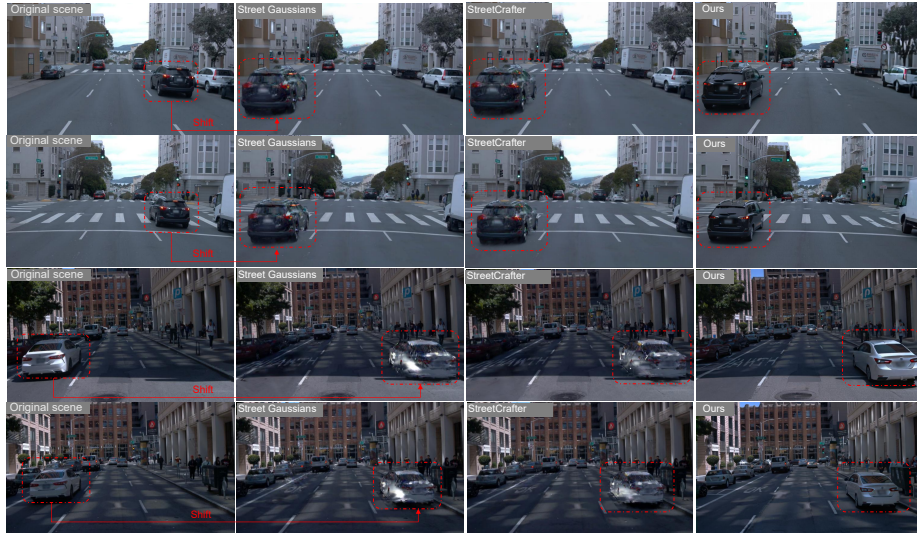


Fig. 5: Qualitative comparison of vehicle trajectory editing. Top two rows: the target vehicle is shifted 6m left and 4m backward. Bottom two rows: the target vehicle is shifted 6m to the right and 1m backward.

each iteration, the reference policy generates $N = 16$ candidate videos per condition, the pairwise reward mechanism produces a scalar reward $\hat{R}(\mathbf{x}^i)$ for each candidate, and the policy is updated via the DiffusionNFT objective (Eq. 1) to reinforce high-win-rate generation patterns while suppressing low-quality ones. This enables the model to progressively improve vehicle quality under OOD conditions without any paired ground-truth supervision.

4 Experiment

4.1 Demonstration of Actor and Viewpoint Manipulation

Fig. 3 demonstrates that our method can manipulate vehicles, pedestrians, and cyclists, including maneuvers outside the training distribution such as in-place spinning, turning, vehicle rollover, and laterally shifted ego-vehicle viewpoints. We encourage the reader to refer to the supplementary material for video results.

Table 1: Quantitative comparison in a lane-change scenario where the trajectory gradually shifts 4 m to the left.

Models	PVG	DD4D w/. PVG	Recon. w/. PVG	S^3 Gauss.	DD4D w/. S^3 Gauss.	Recon. w/. S^3 Gauss.	Deform.-GS	DD4D w/. Deform.-GS	Recon. w/. Deform.-GS	Ours
NTA-IoU $\uparrow$	0.256	0.438	0.464	0.175	0.495	0.413	0.240	0.335	0.443	0.549
NTL-IoU $\uparrow$	50.70	53.06	53.21	49.05	53.42	51.62	51.62	52.93	53.78	56.13
FID $\downarrow$	105.29	71.52	74.32	124.90	66.93	123.61	92.24	77.32	76.24	51.99

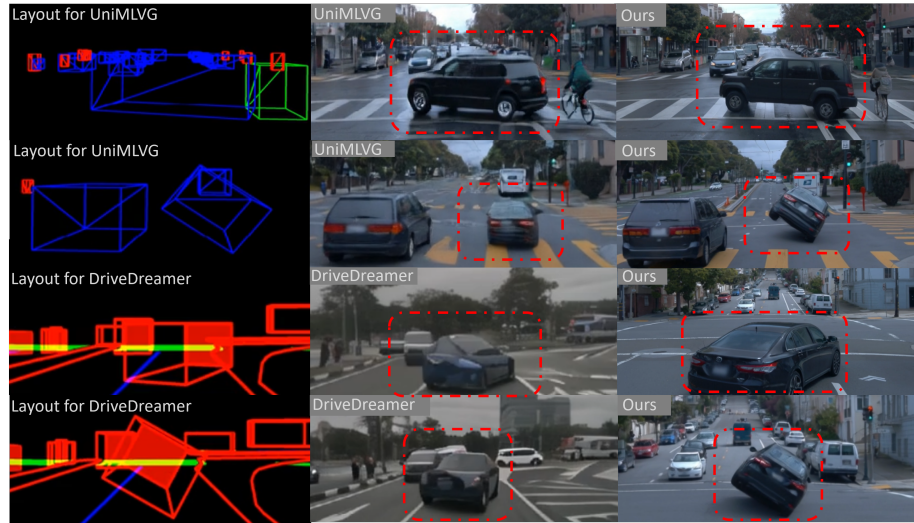


Fig. 6: Qualitative comparison with layout-conditioned methods. As DriveDreamer is restricted to nuScenes, it runs on its native data, while ours uses similar Waymo scenes for side-by-side reference.

4.2 Implementation Details

Conditional Video Diffusion Model. We fine-tune the video diffusion model on 1,000 scenes from the Waymo dataset [21]. Each scene comprises five camera views; we use only the front, front-left, and front-right views for training. Starting from the VACE-1.3B model [10], we fine-tune for approximately 500 H100 GPU hours with a batch size of 32, using the Adam optimizer with a constant learning rate of 1×10^{-5} .

RL-Based Post-Training. We curate 20 representative OOD conditions covering in-place spinning, left/right turning, vehicle rollover, and laterally shifted ego-vehicle viewpoints. For each condition, we generate a group of $N = 16$ candidate videos per rollout for policy optimization. To ensure training stability, we adopt LoRA for RL-based fine-tuning rather than updating full model weights. We use the Adam optimizer with a constant learning rate of 1×10^{-5} . All other hyperparameters of the DiffusionNFT framework, including β and the data collection policy update strategy, follow the default settings in [36]. Post-training takes approximately 400 H100 GPU hours over 500 optimization steps.

For implementation details of the vehicle completion model and the pairwise preference model, please refer to the supplementary material.

4.3 Comparison

We compare our method against existing approaches under two evaluation settings: novel ego-vehicle trajectory and vehicle trajectory editing.



Fig. 7: Ablation study on vehicle completion. Vehicle completion fills in unobserved surfaces in the pseudo-images, effectively eliminating vehicle distortion and texture artifacts in the diffusion outputs.

Table 2: Quantitative comparison on the vehicle trajectory editing scenario with a 5m lateral shift.

Method	Motion smoothness $\uparrow$	Background consistency $\uparrow$	Image quality $\uparrow$	FID $\downarrow$
Street Gauss.	0.981111	0.930975	0.639775	84.7970
StreetCrafter	0.981715	0.906872	0.650856	83.6278
Ours	0.977355	0.939697	0.692809	80.6267

Novel Ego-Vehicle Trajectory. We evaluate on the off-trajectory benchmark proposed by DriveDreamer4D [35], which introduces Novel Trajectory Agent IoU (NTA-IoU) and Novel Trajectory Lane IoU (NTL-IoU) to measure spatial consistency of vehicles and lane boundaries under novel viewpoints. To simulate a realistic lane-change scenario, we apply a lateral shift of 0.1 m per frame over 40 frames, yielding a cumulative lateral shift of 4 m. We compare with PVG [3], DriveDreamer4D [35], ReconDreamer [16], S3Gaussian [8], and Deformable-GS [32]. We additionally report FID between original-trajectory and novel-trajectory images to evaluate photorealism. As shown in Tab. 1, our method achieves the best performance across all metrics. The qualitative comparison in Fig. 4 is consistent with the quantitative results: our method produces the most photorealistic outputs. Since FreeVS [24] only generates partial novel views, we include it in qualitative comparisons only.

Vehicle Trajectory Editing. We first compare our method with reconstruction-based driving simulators for vehicle trajectory editing. Since several recent methods, including DriveDreamer4D [35], ReconDreamer [16], and FreeSim [4], have not released their source code, we compare with Street Gaussians [29] and StreetCrafter [30]. In this setting, we keep the ego-vehicle trajectory fixed and manually edit the trajectory of a target vehicle, thereby evaluating each method’s ability to synthesize scenes under modified vehicle motion. The qualitative comparison is shown in Fig. 5. Street Gaussians and StreetCrafter struggle to synthesize plausible content in regions that are unobserved in the original recorded views, and produce inconsistent lighting on the edited vehicles. In contrast, our method generates realistic content for newly exposed regions and preserves pho-



Fig. 8: Ablation study on RL-based post-training. From top to bottom: doubled vehicle speed, in-place spinning, and vehicle rollover. As the RL training step increases, the generated vehicles exhibit progressively better geometry, texture fidelity, and lighting plausibility.

torealistic lighting on the edited vehicles. For quantitative evaluation, we adopt three metrics from VBench [9], namely image quality, background consistency, and motion smoothness, and evaluate all methods on the five scenes released by StreetCrafter with a 5m lateral vehicle shift. As shown in Tab. 2, our method achieves the best performance in FID, image quality, and background consistency.

We further compare our method with diffusion-based driving simulators, including UniMLVG [2] and DriveDreamer [26]. These methods synthesize driving videos conditioned on HD maps, bounding boxes, and reference frames, and vehicle trajectories can be edited by manipulating the corresponding bounding boxes over time. To evaluate their ability to follow edited vehicle trajectories, we consider two representative out-of-distribution cases: in-place spinning and vehicle rollover. We compare our method with UniMLVG on the same scenes and under the same conditioning settings. Since DriveDreamer only supports the nuScenes dataset [1], we run it on its native nuScenes scenes and select visually similar Waymo scenes for our method to provide side-by-side references. As shown in Fig. 6, diffusion-based baselines struggle to synthesize these edited trajectories: they generate severely distorted vehicles under in-place spinning and fail to produce physically plausible rollovers. These failure cases suggest that purely data-driven video diffusion models have difficulty extrapolating to vehicle motions that are absent or rarely observed in their training data. In contrast, our method maintains realistic geometry, appearance, and temporal coherence, validating the necessity of our design for OOD trajectory editing.

4.4 Ablation Studies

We conduct ablation studies to validate the effectiveness of two key components of our framework: vehicle point cloud completion (Sec. 3.1) and RL-based post-training (Sec. 3.2).

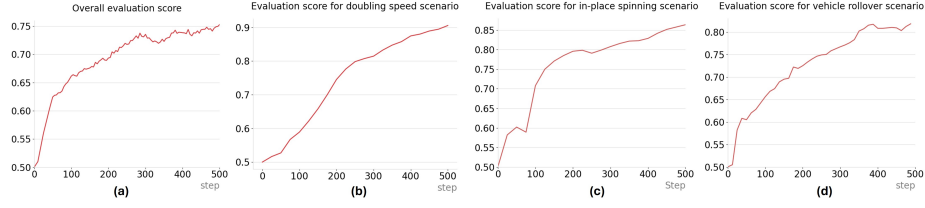


Fig. 9: Evaluation score curves during RL training. (a) Overall evaluation score averaged across all OOD conditions. (b)–(d) Per-scenario scores for doubling speed, in-place spinning, and vehicle rollover, respectively.

Effect of Vehicle Completion. To isolate the contribution of vehicle completion, we evaluate on a vehicle trajectory editing scenario where a target vehicle is repositioned along a modified path, exposing surfaces that are unobserved in the original recording. This setting is specifically chosen because repositioned vehicles are viewed from novel angles, making geometric completeness critical. Fig. 7 presents the rendered pseudo-images with and without vehicle completion, along with the corresponding video diffusion outputs. With vehicle completion, the pseudo-images provide geometrically complete vehicle silhouettes, enabling the diffusion model to produce vehicles with consistent geometry and plausible textures. Without it, large holes on occluded surfaces lead to noticeable vehicle distortion and severe texture artifacts, confirming that vehicle completion is essential for high-quality generation under trajectory edits.

Effect of RL-Based Post-Training. Since the pairwise reward mechanism (Eq. 3) computes win rates within each group of N candidates sampled for a given condition, the resulting rewards reflect only relative quality rankings within that group. They are not comparable across different groups. We therefore design a dedicated evaluation method that is executed every 5 training steps. Specifically, we fix the random seed to 0 to eliminate stochasticity, and for each OOD condition $\mathbf{c}$, we generate one video from the SFT baseline π_{sft} and one from the current RL-trained model π_{θ_k} at step k , both under identical conditions and the same seed. We then apply the same vehicle detection and cropping procedure described in Sec. 3.2 to obtain M evaluation units. For each unit m , we compute the pairwise preference probability $P(a_m^{\text{rl}} \succ a_m^{\text{sft}})$ using our pairwise preference model. The evaluation score for a single condition is defined as the area-weighted average of these preference probabilities across all M units:

$$E(\mathbf{c}) = \frac{\sum_{m=1}^M w_m \cdot P(a_m^{\text{rl}} \succ a_m^{\text{sft}})}{\sum_{m=1}^M w_m}, \quad (4)$$

where w_m is the bounding-box pixel area of unit m . The overall evaluation score is then obtained by averaging $E(\mathbf{c})$ across all evaluation conditions. At step 0, the RL model is identical to the SFT baseline, so both models produce the same video, yielding $P(a_m^{\text{rl}} \succ a_m^{\text{sft}}) = 0.5$ for all units and hence an overall score

of 0.5. A score rising above 0.5 indicates that the RL-trained model generates higher-quality vehicles than the SFT baseline.

Fig. 9 (a) shows the overall evaluation score over the course of RL training, demonstrating a steady improvement as training progresses. We additionally present three representative OOD scenarios in Fig. 8, with all visualizations generated using a fixed seed of 0 for fair comparison across different training steps. From top to bottom, the three scenarios are: (1) doubling the original vehicle speed, (2) in-place vehicle spinning, and (3) vehicle rollover. The visual results clearly show a progressive refinement in vehicle geometry, texture fidelity, and lighting plausibility as RL training proceeds, confirming that the post-training framework effectively improves generation quality under OOD conditions. The per-scenario evaluation scores are reported in Fig. 9 (b), (c), and (d), respectively, all exhibiting consistent upward trends that corroborate the qualitative observations.

5 Limitations and Conclusion

We have presented ReinDriveGen, a unified framework for photorealistic and freely editable driving scene simulation that combines explicit 3D point cloud editing with diffusion-based video synthesis. A vehicle completion module enables free-form manipulation of vehicle trajectories without geometric artifacts, and an RL-based post-training strategy with a pairwise reward mechanism substantially improves vehicle quality under out-of-distribution scenarios. Currently, our method is constrained by GPU memory to generating 49-frame clips, with each clip taking approximately one minute, which does not yet support real-time applications.

More broadly, our work demonstrates that RL-based post-training can effectively improve diffusion model performance beyond the coverage of supervised training data. In autonomous driving, safety-critical corner cases, such as in-place spinning and vehicle rollover, are inherently rare, making the resulting distribution gap a fundamental challenge rather than a data collection problem. The proposed approach offers a principled solution to this challenge, and we believe the same paradigm can be extended to other domains where training data cannot fully cover the space of desired test-time conditions.

Acknowledgements

This work is supported in part by NSFC-RGC Project N_CUHK498/24, in part by Guangdong Basic and Applied Basic Research Foundation (No. 2023B1515130008, XW), in part by Shenzhen Loop Area Institute, and in part by the Centre for Perceptual and Interactive Intelligence (CPII) Ltd under the Innovation and Technology Commission (ITC)’s InnoHK.

References

1. Caesar, H., Bankiti, V., Lang, A.H., Vora, S., Liong, V.E., Xu, Q., Krishnan, A., Pan, Y., Baldan, G., Beijbom, O.: nuscenes: A multimodal dataset for autonomous driving. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. pp. 11621–11631 (2020)
2. Chen, R., Wu, Z., Liu, Y., Guo, Y., Ni, J., Xia, H., Xia, S.: Unimlv: Unified framework for multi-view long video generation with comprehensive control capabilities for autonomous driving. *arXiv preprint arXiv:2412.04842* (2024)
3. Chen, Y., Gu, C., Jiang, J., Zhu, X., Zhang, L.: Periodic vibration gaussian: Dynamic urban scene reconstruction and real-time rendering. *International Journal of Computer Vision* (2026)
4. Fan, L., Zhang, H., Wang, Q., Li, H., Zhang, Z.: Freesim: Toward free-viewpoint camera simulation in driving scenes. In: *Proceedings of the Computer Vision and Pattern Recognition Conference*. pp. 12004–12014 (2025)
5. Gao, R., Chen, K., Xiao, B., Hong, L., Li, Z., Xu, Q.: MagicDrive-V2: High-resolution long video generation for autonomous driving with adaptive control. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision* (2025)
6. Gao, S., Yang, J., Chen, L., Chitta, K., Qiu, Y., Geiger, A., Zhang, J., Li, H.: Vista: A generalizable driving world model with high fidelity and versatile controllability. In: *Advances in Neural Information Processing Systems (NeurIPS)* (2024)
7. Guo, J., Deng, N., Li, X., Bai, Y., Shi, B., Wang, C., Ding, C., Wang, D., Li, Y.: Streetsurf: Extending multi-view implicit surface reconstruction to street views. *arXiv preprint arXiv:2306.04988* (2023)
8. Huang, N., Wei, X., Zheng, W., An, P., Lu, M., Zhan, W., Tomizuka, M., Keutzer, K., Zhang, S.: S3gaussian: Self-supervised street gaussians for autonomous driving. *arXiv preprint arXiv:2405.20323* (2024)
9. Huang, Z., He, Y., Yu, J., Zhang, F., Si, C., Jiang, Y., Zhang, Y., Wu, T., Jin, Q., Chanpaisit, N., Wang, Y., Chen, X., Wang, L., Lin, D., Qiao, Y., Liu, Z.: VBench: Comprehensive benchmark suite for video generative models. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2024)
10. Jiang, Z., Han, Z., Mao, C., Zhang, J., Pan, Y., Liu, Y.: Vace: All-in-one video creation and editing. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. pp. 17191–17202 (2025)
11. Kerbl, B., Kopanas, G., Leimkühler, T., Drettakis, G., et al.: 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.* **42**(4), 139–1 (2023)
12. Lipman, Y., Chen, R.T., Ben-Hamu, H., Nickel, M., Le, M.: Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747* (2022)
13. Liu, J., Liu, G., Liang, J., Li, Y., Liu, J., Wang, X., Wan, P., Zhang, D., Ouyang, W.: Flow-grpo: Training flow matching models via online rl. *arXiv preprint arXiv:2505.05470* (2025)
14. Ljungbergh, W., Taveira, B., Zheng, W., Tonderski, A., Peng, C., Kahl, F., Petersson, C., Felsberg, M., Keutzer, K., Tomizuka, M., et al.: R3d2: Realistic 3d asset insertion via diffusion for autonomous driving simulation. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. pp. 925–934 (2026)
15. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: Nerf: Representing scenes as neural radiance fields for view synthesis. In: *ECCV* (2020)

16. Ni, C., Zhao, G., Wang, X., Zhu, Z., Qin, W., Huang, G., Liu, C., Chen, Y., Wang, Y., Zhang, X., et al.: Recondreamer: Crafting world models for driving scene reconstruction via online restoration. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 1559–1569 (2025)
17. Ni, J., Guo, Y., Liu, Y., Chen, R., Lu, L., Wu, Z.: Maskgwm: A generalizable driving world model with video mask reconstruction. arXiv preprint arXiv:2502.11663 (2025)
18. NVIDIA, :, Ali, A., Bai, J., Bala, M., Balaji, Y., Blakeman, A., Cai, T., Cao, J., Cao, T., Cha, E., Chao, Y.W., Chattopadhyay, P., Chen, M., Chen, Y., Chen, Y., Cheng, S., Cui, Y., Diamond, J., Ding, Y., Fan, J., Fan, L., Feng, L., Ferroni, F., Fidler, S., Fu, X., Gao, R., Ge, Y., Gu, J., Gupta, A., Gururani, S., Hanafi, I.E., Hassani, A., Hao, Z., Huffman, J., Jang, J., Jannaty, P., Kautz, J., Lam, G., Li, X., Li, Z., Liao, M., Lin, C.H., Lin, T.Y., Lin, Y.C., Ling, H., Liu, M.Y., Liu, X., Lu, Y., Luo, A., Ma, Q., Mao, H., Mo, K., Nah, S., Narang, Y., Panaskar, A., Pavao, L., Pham, T., Ramezanali, M., Reda, F., Reed, S., Ren, X., Shao, H., Shen, Y., Shi, S., Song, S., Stefaniak, B., Sun, S., Tang, S., Tasmeen, S., Tchapmi, L., Tseng, W.C., Varghese, J., Wang, A.Z., Wang, H., Wang, H., Wang, H., Wang, T.C., Wei, F., Xu, J., Yang, D., Yang, X., Ye, H., Ye, S., Zeng, X., Zhang, J., Zhang, Q., Zheng, K., Zhu, A., Zhu, Y.: World simulation with video foundation models for physical ai (2026), <https://arxiv.org/abs/2511.00062>
19. Sapkota, R., Cheppally, R.H., Sharda, A., Karkee, M.: Yolo26: key architectural enhancements and performance benchmarking for real-time object detection. arXiv preprint arXiv:2509.25164 (2025)
20. Siméoni, O., Vo, H.V., Seitzer, M., Baldassarre, F., Oquab, M., Jose, C., Khalidov, V., Szafraniec, M., Yi, S., Ramamonjisoa, M., Massa, F., Haziza, D., Wehrstedt, L., Wang, J., Darcet, T., Moutakanni, T., Sentana, L., Roberts, C., Vedaldi, A., Tolan, J., Brandt, J., Couprie, C., Mairal, J., Jégou, H., Labatut, P., Bojanowski, P.: DINOv3 (2025), <https://arxiv.org/abs/2508.10104>
21. Sun, P., Kretschmar, H., Dotiwalla, X., Chouard, A., Patnaik, V., Tsui, P., Guo, J., Zhou, Y., Chai, Y., Caine, B., Vasudevan, V., Han, W., Ngiam, J., Zhao, H., Timofeev, A., Ettinger, S., Krivokon, M., Gao, A., Joshi, A., Zhang, Y., Shlens, J., Chen, Z., Anguelov, D.: Scalability in perception for autonomous driving: Waymo open dataset. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (June 2020)
22. Tonderski, A., Lindström, C., Hess, G., Ljungbergh, W., Svensson, L., Petersson, C.: Neurad: Neural rendering for autonomous driving. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 14895–14904 (2024)
23. Turki, H., Zhang, J.Y., Ferroni, F., Ramanan, D.: Suds: Scalable urban dynamic scenes. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 12375–12385 (2023)
24. Wang, Q., Fan, L., Wang, Y., Chen, Y., Zhang, Z.: Freevs: Generative view synthesis on free driving trajectory. arXiv preprint arXiv:2410.18079 (2024)
25. Wang, S., Sun, H., Wang, B., Ye, H., Yu, X.: Mirage: One-step video diffusion for photorealistic and coherent asset editing in driving scenes. arXiv preprint arXiv:2512.24227 (2025)
26. Wang, X., Zhu, Z., Huang, G., Chen, X., Zhu, J., Lu, J.: Drivedreamer: Towards real-world-drive world models for autonomous driving. In: European conference on computer vision. pp. 55–72. Springer (2024)

27. Wang, Y., Li, Z., Zang, Y., Zhou, Y., Bu, J., Wang, C., Lu, Q., Jin, C., Wang, J.: Pref-grpo: Pairwise preference reward-based grpo for stable text-to-image reinforcement learning. arXiv preprint arXiv:2508.20751 (2025)
28. Wu, Z., Liu, T., Luo, L., Zhong, Z., Chen, J., Xiao, H., Hou, C., Lou, H., Chen, Y., Yang, R., et al.: Mars: An instance-aware, modular and realistic simulator for autonomous driving. In: CAAI International Conference on Artificial Intelligence. pp. 3–15. Springer (2023)
29. Yan, Y., Lin, H., Zhou, C., Wang, W., Sun, H., Zhan, K., Lang, X., Zhou, X., Peng, S.: Street gaussians: Modeling dynamic urban scenes with gaussian splatting. In: ECCV (2024)
30. Yan, Y., Xu, Z., Lin, H., Jin, H., Guo, H., Wang, Y., Zhan, K., Lang, X., Bao, H., Zhou, X., et al.: Streetcrafter: Street view synthesis with controllable video diffusion models. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 822–832 (2025)
31. Yang, Z., Chen, Y., Wang, J., Manivasagam, S., Ma, W.C., Yang, A.J., Urtasun, R.: Unisim: A neural closed-loop sensor simulator. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 1389–1399 (2023)
32. Yang, Z., Gao, X., Zhou, W., Jiao, S., Zhang, Y., Jin, X.: Deformable 3d gaussians for high-fidelity monocular dynamic scene reconstruction. arXiv preprint arXiv:2309.13101 (2023)
33. Yu, X., Rao, Y., Wang, Z., Lu, J., Zhou, J.: AdaPoinTr: Diverse Point Cloud Completion With Adaptive Geometry-Aware Transformers . IEEE Transactions on Pattern Analysis & Machine Intelligence **45**(12), 14114–14130 (Dec 2023). <https://doi.org/10.1109/TPAMI.2023.3309253>, <https://doi.ieeecomputersociety.org/10.1109/TPAMI.2023.3309253>
34. Zhang, H., Fan, L., Wang, Q., Li, W., Wu, Z., Lu, L., Zhang, Z., Li, H.: Ga-drive: Geometry-appearance decoupled modeling for free-viewpoint driving scene generation. arXiv preprint arXiv:2602.20673 (2026)
35. Zhao, G., Ni, C., Wang, X., Zhu, Z., Zhang, X., Wang, Y., Huang, G., Chen, X., Wang, B., Zhang, Y., Mei, W., Wang, X.: Drivedreamer4d: World models are effective data machines for 4d driving scene representation (2024)
36. Zheng, K., Chen, H., Ye, H., Wang, H., Zhang, Q., Jiang, K., Su, H., Ermon, S., Zhu, J., Liu, M.Y.: Diffusionnft: Online diffusion reinforcement with forward process. arXiv preprint arXiv:2509.16117 (2025)
37. Zhou, X., Lin, Z., Shan, X., Wang, Y., Sun, D., Yang, M.H.: Drivinggaussian: Composite gaussian splatting for surrounding dynamic autonomous driving scenes. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 21634–21643 (2024)