

DriveWeaver: Point-Conditioned Video Inpainting for Controllable Vehicle Insertion in Autonomous Driving Simulation

Junzhe Jiang¹ Zipei Ma^{1,2} Zijie Pan¹ Li Zhang^{1,2*}

¹ School of Data Science, Fudan University

² Shanghai Innovation Institute

github.com/LogosRoboticsGroup/DriveWeaver

Abstract. A pivotal step in autonomous driving simulation involves inserting foreground vehicles with predefined trajectories into simulated scenes. This process enhances scene diversity and facilitates the creation of various corner cases for testing and improving autonomous driving models. However, existing methods often rely on pre-reconstructed 3D assets, which frequently lead to lighting inconsistencies between the inserted foreground and the background. Moreover, the reliance on limited, manually-curated 3D assets hinders large-scale deployment. To address these challenges, we propose **DriveWeaver**, a novel framework for controllable vehicle insertion in autonomous driving simulation. Specifically, for a masked target insertion area, DriveWeaver performs video inpainting conditioned on vehicle point clouds to generate high-quality, temporally consistent vehicles. This video-inpainting-based approach ensures seamless blending between the foreground and background, while the readily available point cloud conditions enable superior generalization. To support long-term generation, we further design a global-to-local hierarchical inpainting strategy, ensuring the consistent identity and appearance of the inserted vehicles. Meanwhile, we extract explicit 3D Gaussian representations of the inserted vehicles through an urban reconstruction pipeline to enable real-time rendering for autonomous driving simulation. Extensive experiments across diverse datasets demonstrate that our method outperforms existing baselines in visual realism and geometric consistency, providing a robust tool for scalable autonomous driving scene augmentation.

Keywords: Video inpainting · Autonomous driving simulation

1 Introduction

Autonomous driving simulators have garnered significant attention due to their ability to evaluate driving algorithms in a safe and cost-effective manner [3, 6, 19, 28, 53, 61]. Traditional simulators [6, 36] employ manually-crafted CAD models to render autonomous driving scenarios. While offering high controllability,

* Corresponding author (lizhangfd@fudan.edu.cn)

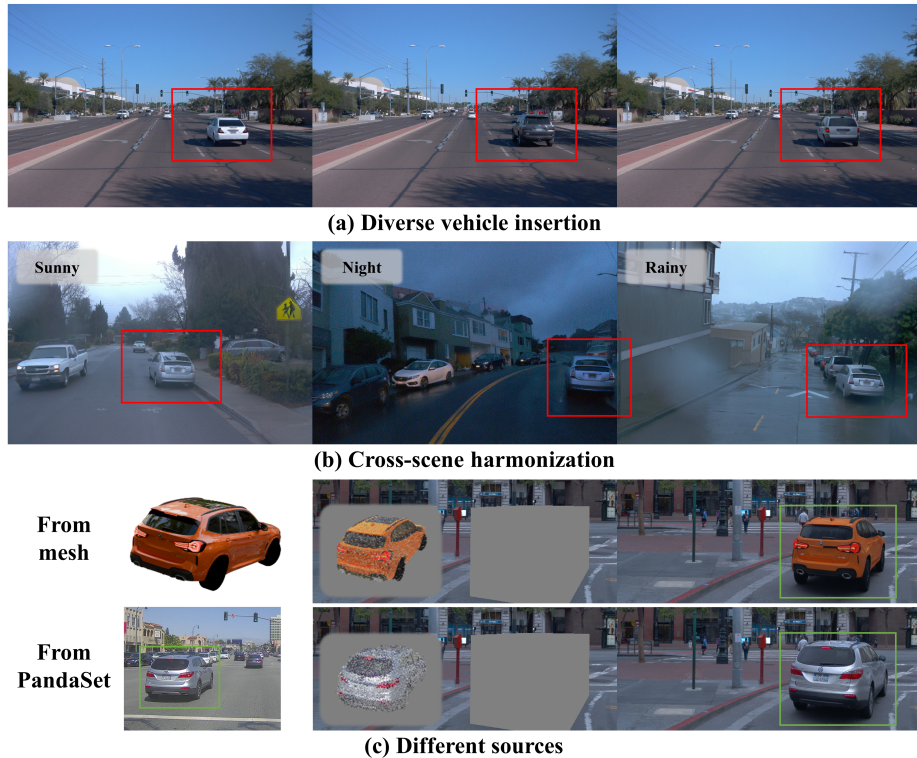


Fig. 1: Controllable high-fidelity vehicle insertion via DriveWeaver. Our point-conditioned inpainting framework provides flexible scene editing for autonomous driving simulation. It supports (a) diverse vehicle insertion options, (b) seamless insertion of vehicles with lighting in complex weather and illumination conditions and (c) flexible point-conditioned insertion using point clouds from diverse sources.

these simulators suffer from a pronounced domain gap compared to the real world. Recently, leveraging readily available real-world urban imagery, scene reconstruction techniques have provided a reliable solution for high-fidelity autonomous driving simulation. With advancements in neural rendering [2, 4, 30, 47, 49, 52] which further augmented by generative priors to refine novel view synthesis [50, 54, 58], contemporary methods have achieved high-quality, large-scale novel view synthesis of recorded scenes. Despite achieving photo-realistic rendering, these methods are primarily restricted to faithfully reproducing captured environments. The lack of the capability for flexible scene editing poses a significant barrier to evaluating driving algorithms in rare, long-tail corner cases.

To achieve high-fidelity foreground insertion for constructing safety-critical corner cases, several recent studies [47, 61, 64] leverage large-scale autonomous driving datasets [7, 13, 38] to pre-reconstruct 3D vehicle assets. These assets are then composed and rendered into background scenes according to predefined tra-

jectories. To mitigate the visual inconsistencies between the inserted objects and the environment, various post-processing techniques have been employed, such as selecting vehicles with matching illumination and synthesizing shadows. Despite these efforts, a persistent stylistic gap remains between the foreground and background. Furthermore, these pipelines heavily rely on labor-intensive processes, including manual pose calibration for unannotated datasets [7] and the heuristic selection of lighting-compatible assets. Such dependencies on manual intervention significantly limit the scalability and generalizability across diverse real-world scenarios. Alternative approaches [3, 19, 24, 29] employ manually-modeled 3D mesh assets as foreground objects and utilize diffusion models [35] as evaluators to provide guidance signals for physics-based inverse rendering. Through differentiable rendering, these methods enable the end-to-end optimization of lighting parameters. While achieving superior realism in vehicle insertion, these frameworks require per-scene optimization via diffusion models, which is computationally expensive. Furthermore, the acquisition of high-quality 3D meshes remains costly, making these methods difficult to adapt efficiently to diverse scenarios or to satisfy the requirements of large-scale deployment.

In this paper, we present **DriveWeaver**, a novel point-conditioned video inpainting diffusion model. As shown in Fig. 1, our framework achieves precise control over the synthesis of high-quality, geometrically consistent foreground vehicles using cost-effective point cloud data. Specifically, we leverage point cloud renderings as pixel-level conditions to provide rigorous geometric grounding for vehicle poses. By integrating these renderings as conditional inputs, DriveWeaver exploits the powerful generative priors of video diffusion models to fix missing structural details and ensure seamless environmental harmony. This pixel-level conditioning paradigm enables cross-dataset generalization during inference, allowing point clouds from diverse sources to guide the generation process. Consequently, our approach offers a scalable and versatile solution for foreground vehicle insertion in autonomous driving simulations. Furthermore, the superior geometric consistency of the synthesized videos enables the extraction of explicit 3D Gaussian representations for the inserted foreground vehicles via urban reconstruction pipelines. This capability facilitates real-time rendering, which is crucial for the large-scale training and evaluation of autonomous driving models.

Our main contributions are summarized as follows: **(i)** We propose a novel video inpainting diffusion framework DriveWeaver designed for highly scalable and generalizable foreground vehicle insertion in autonomous driving simulation. Leveraging point cloud renderings as pixel-level conditions, DriveWeaver generates foreground vehicles with strong geometric consistency while preserving environmental illumination and style coherence. **(ii)** To support long-term generation, we develop a global-to-local hierarchical inpainting strategy that ensures the long-term temporal consistency of inserted foreground vehicles. Furthermore, the generated results can be directly reconstructed into 3D Gaussian representations, supporting efficient rendering for downstream autonomous driving model training and evaluation. **(iii)** Extensive experiments conducted on two large-scale benchmark datasets (Waymo [38] and PandaSet [46]) demonstrate

that DriveWeaver significantly outperforms existing state-of-the-art methods in terms of visual realism and geometric consistency.

2 Related work

Neural rendering for autonomous driving simulation. Recent advancements in neural rendering have fundamentally transformed autonomous driving simulation by enabling high-fidelity, data-driven scene reconstruction directly from sensor logs. Early neural simulators [3, 32, 39, 47, 52] primarily relied on neural radiance fields (NeRF) [31] to parameterize large-scale urban environments. To address the computational bottlenecks and representation limitations of NeRF, recent approaches [2, 4, 30, 49, 51, 62] have increasingly adopted 3D Gaussian splatting (3DGS) [21] for highly efficient rendering. Despite their remarkable reconstruction capabilities, these purely physical-based rendering methods often suffer from visual artifacts or blurring in unobserved regions when rendering extreme novel views, which frequently arise in autonomous driving simulation [10, 19, 62, 64]. To alleviate this, contemporary frameworks [43, 50, 54, 58] leverage the powerful generative priors of large-scale diffusion models to fix missing details and refine rendering quality. Although these neural rendering and generative frameworks achieve impressive realism when reconstructing recorded data, they lack the ability to edit scenes, which limits their applicability for evaluating driving algorithms in rare long-tail corner cases within autonomous driving simulators. This limitation motivates us to develop a point-conditioned video inpainting model that enables flexible editing of reconstructed scenes.

Urban scene editing. To test autonomous vehicles in safety-critical scenarios, simulators must support flexible scene editing, particularly foreground object insertion. Recent works [61, 64] curate large libraries of pre-reconstructed 3D vehicle assets and compose them with background scenes along predefined trajectories. To reduce lighting and style gaps, these pipelines rely on asset retrieval with matched illumination and shadows synthesis, which limits scalability and cross-scene generalization. Other approaches [3, 19, 24, 29] adopt physics-based neural rendering [45] pipelines that explicitly optimize object-scene interactions using high-quality 3D meshes and inverse rendering. While capable of producing realistic results, these methods are computationally expensive and depend on costly CAD meshes, hindering large-scale deployment. Concurrent works explore alternative editing paradigms that face similar scalability and temporal constraints. R3D2 [27] and G²Editor [23] compose foreground objects from pre-reconstructed 3D assets, which incur substantially higher acquisition costs than our point-based conditions. DriveEditor [25] and GenMM [37] instead rely on a single reference images for object insertion, which introduces self-occlusion artifacts and restricts insertion viewpoints to those matching the reference. Furthermore, these methods lack the temporal scale necessary for closed-loop simulation: R3D2 is image-only, while G²Editor, DriveEditor, and GenMM are limited to approximately 10 frames—far below the 193 frames our approach generates. In contrast, DriveWeaver replaces costly 3D assets and restrictive reference images

with accessible point cloud renderings, enabling viewpoint-flexible, high-fidelity insertion in a single forward pass. Beyond data efficiency, our hierarchical global-to-local generation strategy eliminates long-term temporal drift, and 3DGS distillation bypasses per-frame inference bottlenecks, achieving real-time interactive rendering suitable for closed-loop autonomous driving simulation.

Video generation and inpainting. Video generation and inpainting techniques have advanced significantly, enabling diverse editing capabilities such as dynamic object insertion and missing region recovery [12, 18, 42, 48, 63, 65]. Recently, architectural improvements in Diffusion transformers (DiT) [33] and the development of flow matching [8, 26] have led to a new generation of foundation video models [20, 22, 41, 55, 60]. These models demonstrate unprecedented visual fidelity, scalability, and temporal consistency in video synthesis, benefiting various downstream tasks [9, 34, 57, 59]. Building upon the strong generative capability of Wan2.1 VACE-14B [20], we formulate foreground vehicle insertion as a controlled video inpainting task. Specifically, we inject sparse point cloud renderings as explicit geometric conditions into the conditional branch of Wan2.1. This point-conditioned inpainting framework allows us to leverage the reliable texture and illumination priors of large-scale generative models, enabling the synthesis of high-quality foreground vehicles that are both geometrically consistent and harmoniously integrated with the autonomous driving scene.

3 Method

Given a recorded autonomous driving scene with camera images, vehicle point clouds, and predefined vehicle trajectories, our goal is to synthesize high-fidelity vehicles that are seamlessly harmonized with the background and strictly aligned with the provided point clouds and trajectory constraints. In Sec. 3.1, we briefly introduce flow matching models. Then, we introduce our point-conditioned video inpainting model DriveWeaver in Sec. 3.2 and long-term generation strategy in Sec. 3.3. Sec. 3.4 describes how the inserted vehicles generated by DriveWeaver are distilled into 3D Gaussian representations to enable real-time rendering. Finally, Sec. 3.5 details the training dataset construction and implementation specifics. An overview of our pipeline is provided in Fig. 2.

3.1 Preliminaries

Flow matching. DriveWeaver is built upon Wan2.1 VACE-14B [20], which employs the flow matching framework [8, 26] to establish a consistent denoising diffusion process for diverse video distributions. An intermediate latent $\mathbf{x}_t$ is constructed by interpolating between a video latent $\mathbf{x}_1$ and Gaussian noise $\mathbf{x}_0 \sim \mathcal{N}(0, I)$, where the timestep $t \in [0, 1]$ is drawn from a logit-normal distribution. Adopting the rectified flows [8] formulation, the latent at time t is expressed as $\mathbf{x}_t = (1 - t)\mathbf{x}_0 + t\mathbf{x}_1$. The corresponding target velocity is defined as:

$$\mathbf{v}_t = \frac{d\mathbf{x}_t}{dt} = \mathbf{x}_1 - \mathbf{x}_0. \quad (1)$$

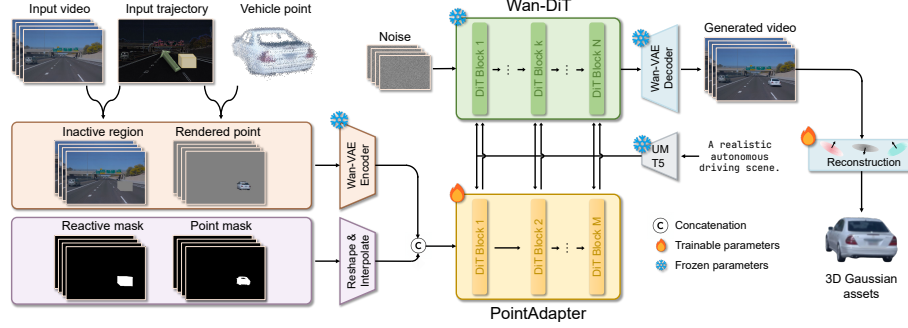


Fig. 2: Pipeline of DriveWeaver. Given an input driving video, a vehicle point cloud, and a target trajectory, DriveWeaver generates realistic scenarios with the vehicle seamlessly inserted. The pipeline leverages a frozen video diffusion backbone (Wan-DiT) and introduces a lightweight, trainable PointAdapter. Specifically, the processed RGB conditions and masks are channel-wise concatenated and fed into the PointAdapter, which injects geometric and spatial guidance into the main branch via residual connections, ensuring efficient training and high-quality generation. The inserted vehicles can be further distilled into 3D representations using an urban reconstruction pipeline, enabling real-time rendering in autonomous driving simulation.

The network is optimized to estimate this velocity field. Specifically, the objective function is defined as the mean squared error (MSE) between the predicted velocity $\mathbf{u}(\mathbf{x}_t, \mathbf{c}_{txt}, t; \boldsymbol{\theta})$ and the ground truth $\mathbf{v}_t$:

$$\mathcal{L} = \mathbb{E}_{\mathbf{x}_0, \mathbf{x}_1, \mathbf{c}_{txt}, t} \|\mathbf{u}(\mathbf{x}_t, \mathbf{c}_{txt}, t; \boldsymbol{\theta}) - \mathbf{v}_t\|^2, \quad (2)$$

where $\mathbf{c}_{txt}$ represents the text embeddings extracted from UMT5 [5] and $\boldsymbol{\theta}$ denotes the trainable parameters.

3.2 DriveWeaver architecture

This section focuses on developing a diffusion model for high-fidelity vehicle insertion. Given an N -frame sequence of recorded driving scenes, the vehicle point clouds, and their predefined trajectories, the model renders corresponding point cloud condition maps. It then generates a sequence of N video frames with the inserted vehicles, thereby enabling the construction of rare long-tail scenarios for autonomous driving simulation. We employ Wan2.1 VACE-14B [20] as our frozen latent video flow matching diffusion backbone, which comprises a 3D VAE and a diffusion transformer (DiT) denoiser. To enable the backbone to be conditioned on our rendered point cloud maps, DriveWeaver introduces a lightweight point cloud adapter. This adapter modulates the frozen backbone, allowing it to incorporate the geometric guidance from the rendered point cloud maps during the generation process.

Point cloud rendering. We employ a point cloud rendering technique to generate geometric conditions. For a driving sequence consisting of N frames, we

first project the LiDAR points onto the calibrated images and colorize them by querying the corresponding pixel values. We then extract the point clouds for each vehicle instance using 3D bounding boxes, resulting in instance point clouds $\{\mathbf{p}_j^i\}_{j=1}^N$ defined in the canonical coordinate system of each vehicle instance i in frame j . To render the point condition at frame j , we aggregate LiDAR points within a temporal window of length l . These points are integrated into a unified point cloud $\mathbf{p}^i$ represented in the canonical coordinate system. Subsequently, the point cloud is transformed based on the respective vehicle instance poses and camera poses, and the point clouds from all individual instances are aggregated to form the combined point cloud $\mathbf{p}$.

Since direct projection of the combined point cloud $\mathbf{p}$ onto the image plane often leaves significant gaps and occluded regions, we assign a fixed radius to each point and perform point rasterization following [50]. As illustrated in Fig. 2, this procedure yields a set of rendered point cloud maps $\{\mathbf{I}_j^{pc}\}_{j=1}^N$ and their associated masks $\{\mathbf{M}_j^{pc}\}_{j=1}^N$. Simultaneously, the 3D bounding boxes of the inserted vehicles are projected onto the image plane to differentiate between regions requiring inpainting and the original background areas. This projection generates inactive background RGB images $\{\mathbf{I}_j^{in}\}_{j=1}^N$, and binary inpainting masks $\{\mathbf{M}_j^{re}\}_{j=1}^N$ which explicitly define the reactive area for vehicle synthesis. By using point cloud rendering as a pixel-level condition for video inpainting, the model only needs to complete the geometry indicated by the point cloud while maintaining environmental coherence. This formulation enables cross-dataset conditioning on vehicle point clouds and improves large-scale generalization.

PointAdapter. We respectively concatenate the rendered point cloud maps $\{\mathbf{I}_j^{pc}\}_{j=1}^N$ and their associated masks $\{\mathbf{M}_j^{pc}\}_{j=1}^N$, the background RGB of the inactive regions $\{\mathbf{I}_j^{in}\}_{j=1}^N$, and the masks for the reactive regions $\{\mathbf{M}_j^{re}\}_{j=1}^N$. The two RGB maps ($\{\mathbf{I}_j^{pc}\}_{j=1}^N$, $\{\mathbf{I}_j^{in}\}_{j=1}^N$) are encoded using the same 3D VAE [41] employed for the video latents. And following the practices in [20, 41], the masks ($\{\mathbf{M}_j^{pc}\}_{j=1}^N$, $\{\mathbf{M}_j^{re}\}_{j=1}^N$) are reshaped and interpolated to match the latent resolution. These components are then concatenated along the channel dimension and aligned with the latent video tokens.

Taking the concatenated conditions above as input, PointAdapter is designed as a lightweight DiT branch, as shown in Fig. 2. It maintains the same token dimension as the Wan-DiT blocks but consists of significantly fewer layers. A PointAdapter block $\mathcal{T}^{pc}$ is associated with every k -th Wan-DiT block $\mathcal{T}$, and the output of the PointAdapter block is linearly projected back to the backbone’s feature space by projection function $\mathcal{P}$ and incorporated into the corresponding DiT block as a residual modulation:

$$\mathbf{z}_{j+1}^{pc} = \mathcal{T}_j^{pc}(\mathbf{z}_j^{pc}, \mathbf{c}_{txt}, t) \quad (3)$$

$$\mathbf{z}_{i+1} = \mathcal{T}_i(\mathbf{z}_i, \mathbf{c}_{txt}, t) + \mathcal{P}_{i/k}(\mathbf{z}_{i/k}^{pc}) \cdot \mathbf{1}[i \equiv 0 \pmod{k}] \quad (4)$$

where i and j denote the block indices of the Wan-DiT and the PointAdapter, respectively. $\mathbf{z}$ and $\mathbf{z}^{pc}$ represent the latent features of the backbone and the PointAdapter. $\mathbf{c}_{txt}$ refers to the text embeddings encoded by umT5 [5], t is the timestep, and $\mathbf{1}[\cdot]$ is an indicator function that determines the specific backbone

blocks where the geometric information is injected. This adapter-based conditioning mechanism injects geometric information into the backbone using only a minimal number of additional parameters, ensuring that the geometric guidance effectively affects each backbone layer while maintaining high training efficiency.

Inference. During the inference stage, DriveWeaver benefits from the pixel-level conditioning of the rendered point cloud maps, which enables support for cross-dataset and non-LiDAR point cloud inputs. To further enhance the generation quality, we employ classifier-free guidance (CFG) [17]. Specifically, in addition to the conditional branch, we define an unconditional input by zeroing out the rendered point cloud maps $\{\mathbf{I}_j^{pc}\}_{j=1}^N$ and their corresponding masks $\{\mathbf{M}_j^{pc}\}_{j=1}^N$. This guidance strategy allows DriveWeaver to synthesize more realistic and consistent foreground vehicles within fewer sampling steps, effectively improving both visual fidelity and inference efficiency.

3.3 Long-term generation

In autonomous driving simulation, vehicle trajectories often span extended durations, necessitating the generation of long, temporally consistent videos. However, due to the high memory consumption of 3D-VAE and DiT architectures, modern video diffusion models are typically limited to generating short clips in a single pass (e.g., 13 to 81 frames). Existing approaches [1, 11, 14] often employ autoregressive or sliding window strategies, where the final frames of a previous clip serve as the condition for the next. While straightforward, this Markovian conditioning is prone to error accumulation. Over long durations, the generated foreground vehicles often suffer from significant semantic drift, structural degradation, and appearance inconsistency, which violates the strict 3D consistency required for driving simulation. To overcome these limitations, we propose a global-to-local hierarchical inpainting strategy. Instead of sequential generation, we decouple the process into a global low-frame-rate anchoring stage and a local high-frame-rate interpolation stage, as shown in Fig. 3.

Global anchoring stage. For a long input sequence, we first perform temporal subsampling on the background video and the corresponding point cloud conditions to create a sparse, low-frame-rate sequence. Given the reduced frame count, our model can process the entire duration in a single global forward pass. This step generates a coarse yet temporally consistent anchor video that establishes the global identity, geometry, and environmental lighting of the inserted vehicle across the full trajectory.

Local interpolation stage. Subsequently, we partition the original high-frame-rate timeline into shorter local segments. For each segment, the corresponding anchor frames serve as dense guidance to replace the sparse point cloud conditions. The model then performs conditional generation to synthesize the local high-frame-rate video.

To support this two-stage inference paradigm, we train DriveWeaver under two complementary modes. In the GT-Free mode, the model learns to generate frames solely from sparse point cloud conditions, preparing it for the global anchoring stage. In the GT-Aware mode, ground-truth frames are provided as

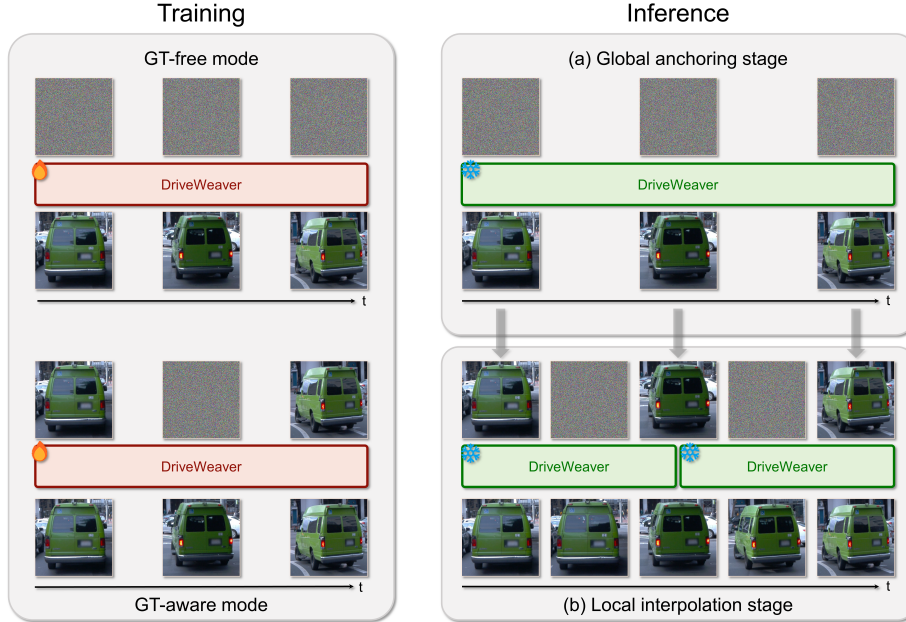


Fig. 3: Long-term generation. To prevent error accumulation and semantic drift common in autoregressive methods, DriveWeaver decouples the generation into two stages. (a) Global anchoring stage: The model processes a temporally subsampled sparse sequence in a single pass to establish a consistent vehicle identity and lighting across the entire trajectory. (b) Local interpolation stage: The synthesized sparse anchor frames are subsequently used as dense guidance to generate the local high-frame-rate segments, ensuring strict 3D and temporal consistency over extended durations. During training, DriveWeaver is trained under GT-Free and GT-Aware modes to respectively accommodate the global anchoring and local interpolation stages.

supervision, enabling the model to perform precise local interpolation guided by the anchor frames. By enforcing local generation to strictly follow the global anchors, our strategy effectively eliminates error accumulation and temporal drift. As a result, DriveWeaver ensures that the appearance and lighting of the vehicle remain perfectly consistent from the first frame to the last.

3.4 Distillation of inserted vehicles

While our point-conditioned video inpainting framework successfully synthesizes foreground vehicles with seamless environmental harmony and strict geometric consistency, the generative process is inherently bottlenecked by the slow, iterative nature of the denoising steps [16, 33]. For large-scale autonomous driving simulators that require high-frame-rate or real-time closed-loop evaluation [19, 53, 61, 64], relying exclusively on a video diffusion model for real-time rendering is computationally prohibitive. To bridge the gap between high-fidelity

offline generation and online simulation, it is imperative to distill the generated 2D video into an explicit 3D representation. Because our hierarchical generation strategy and precise point-conditioning enforce temporal consistency, the synthesized video frames meet the geometric constraints required for 3D reconstruction. Specifically, we leverage OmniRe [4], a state-of-the-art 3D Gaussian [21] urban reconstruction framework, to process the synthesized high-quality video. During the reconstruction pipeline, we explicitly decouple the inserted vehicle instance from the background scene representations, which encapsulates the complex illumination, realistic shadows, and stylistic properties generated by the diffusion model. This distilled 3D asset can subsequently be integrated into 3D Gaussian rasterization pipelines. Consequently, during the simulation phase, we can completely bypass the computationally heavy diffusion model, enabling real-time rendering of the edited corner cases for autonomous driving simulation.

3.5 Implementation details

We build our training dataset using the Waymo Open Dataset [38] and PandaSet [46], leveraging their 10Hz camera and the synchronized LiDAR. Following the instance point cloud rendering pipeline described in Sec. 3, we curated a total of approximately 10,000 training samples. We set the point rasterization radius to 0.005 for generating the rendered point cloud maps. We construct DriveWeaver based on the Wan2.1 VACE-14B [20] model. The PointAdapter comprises 3B parameters and is integrated into the Wan-DiT backbone at layers $[0, 5, 10, \dots, 30, 35]$ (i.e., every 5th block). During training, the Wan backbone remains frozen, and only the PointAdapter modules are updated. To ensure that the model generates results solely conditioned on the rendered point cloud maps, we fix the text prompt as “A realistic autonomous driving scene”. The model is trained on video sequences of 49 frames with a resolution of 480×720 . We use the Adam optimizer with a learning rate of 1×10^{-5} and a total batch size of 8. A linear warmup of 100 steps is applied at the beginning of the training process, during which the learning rate gradually increases. All experiments are conducted on 8 NVIDIA H200 (141GB) GPUs, and the training process runs for 20,000 iterations (approximately 5 days) to complete. During training, we employ the classifier-free guidance (CFG) [17] strategy, where the point cloud conditions are randomly dropped with a probability of 0.15. At inference time, we use 30 denoising steps with a CFG scale of 2.5. Generating a 49-frame 480p video clip on a single H200 GPU takes approximately 2 minutes.

For long-term generation, we process sequences of 193 frames by sampling anchor frames at a stride of 4. This results in a 49-frame anchor sequence used to establish long-term consistency. The generated anchor video is then equally partitioned into 4 segments to facilitate local high-frame-rate synthesis. To distill the 3D representations of the inserted vehicles, we follow the configuration of OmniRe [4] and disable LiDAR depth supervision within the insertion regions, as the synthesized foreground vehicles lack precise depth ground truth.

Regarding computational overhead for distillation of inserted vehicles, training a standard 193-frame sequence requires approximately 1.5 hours on single

NVIDIA RTX 4090 GPU. However, because OmniRe decouples the optimization of foreground instances from the background, we can freeze the pre-trained background 3DGS representation and exclusively optimize the inserted foreground vehicle. This targeted optimization converges in only 10 minutes per instance. Given that pre-reconstructed background 3DGS models serve as a standard prerequisite in practical autonomous driving simulation pipelines, this marginal offline distillation cost is readily justifiable.

4 Experiments

We evaluate both DriveWeaver and its distilled 3D Gaussian representation (denoted as DriveWeaver and DriveWeaver-D, respectively) in terms of visual realism and geometric consistency. All evaluation metrics are computed after resizing the outputs to the training resolution of DriveWeaver, i.e., 480×720 .

4.1 Experimental setup

Baselines. We compare our method with HUGSIM [61], StreetCrafter [50] and Wan2.1 VACE-14B [20]. HUGSIM [61] obtains 3D assets by reconstructing the 3DRealCar [7] dataset, combines them with the background for rendering, and renders shadows to enhance realism. StreetCrafter [50] takes LiDAR point cloud rendering as a condition to generate the entire scene. And Wan2.1 VACE-14B [20] is the backbone model we use, which supports the masked video to video (MV2V) generation mode. All methods are based on their official implementations.

Evaluation metrics. We conduct experiments on the Waymo Open Dataset [38] and PandaSet [46]. Specifically, we select 10 sequences of approximately 200 frames each from the Waymo and 5 sequences of 80 frames each from PandaSet. All evaluation sequences are strictly held out and do not appear in the training set. For visual realism, we compute FID [15], FVD [40], and LPIPS [56] between the videos with inserted vehicles and the original videos for all methods. For geometric consistency, we leverage the fact that 3D reconstruction quality depends on geometric coherence. We therefore apply the urban reconstruction method OmniRe [4] to the edited videos and report PSNR and SSIM [44] of the reconstructed results. Better geometric consistency facilitates more accurate foreground reconstruction, resulting in higher PSNR and SSIM, whereas poor consistency introduces blur and artifacts after reconstruction, leading to lower scores. In addition, due to the longer sequences in Waymo, we further evaluate long-term generation performance on this dataset.

4.2 Comparisons with the state-of-the-art

We first evaluate the short-term and long-term foreground insertion performance on Waymo [38] in terms of visual realism and geometric consistency. As presented in Tab. 1 and Fig. 4, DriveWeaver outperforms state-of-the-art methods [20, 50, 61] in visual realism across all evaluation metrics, and ranks second in geometric consistency, surpassed only by the computer-graphics-based

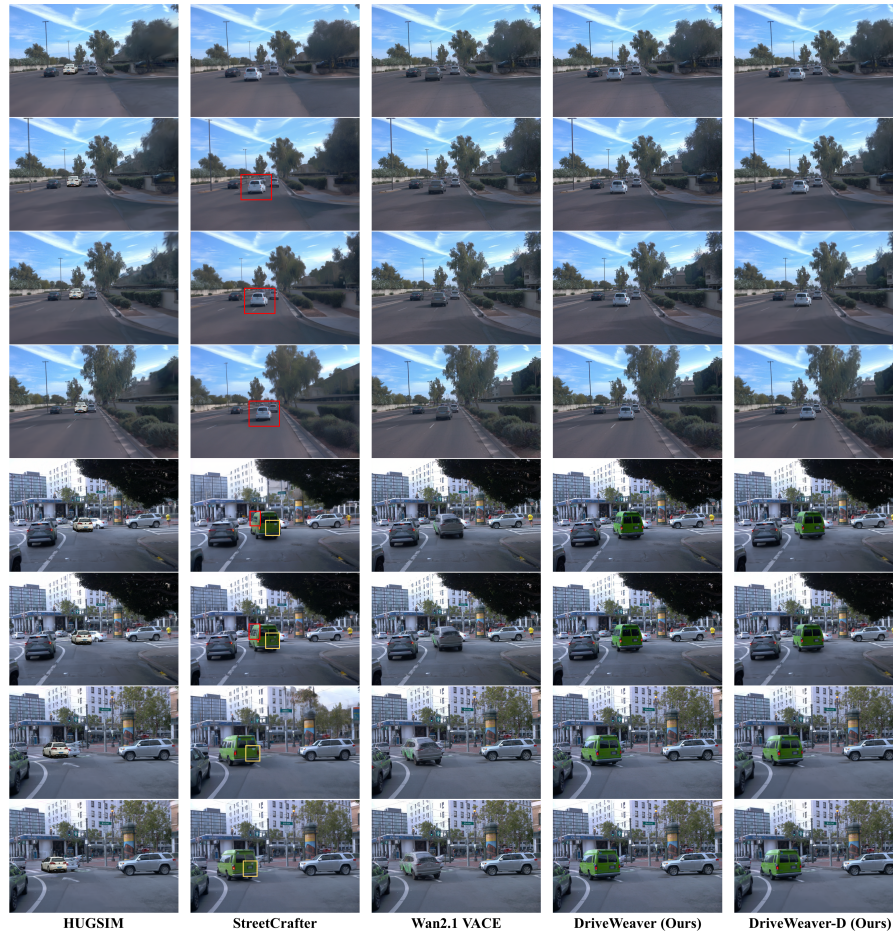


Fig. 4: Qualitative comparison on the Waymo [38] dataset.

insertion approach HUGSIM [61]. Meanwhile, the distilled 3DGS representation, DriveWeaver-D, incurs only marginal degradation in visual realism compared to the original generation results. In contrast, utilizing explicit 3D meshes, HUGSIM [61] naturally maintains structural rigidity across frames. However, without generative environmental understanding, it struggles to harmonize the asset with the target scene’s complex illumination and style, which results in a severe domain gap, leading to poor visual realism metrics like FID and FVD. StreetCrafter [50] generates long videos in an autoregressive manner. Although it can produce visually appealing individual frames—resulting in relatively superior FID scores among the baselines—the generated vehicles exhibit noticeable semantic drift and shape deformation over long simulated trajectories. This lack of long-term temporal consistency leads to inferior FVD and LPIPS perfor-

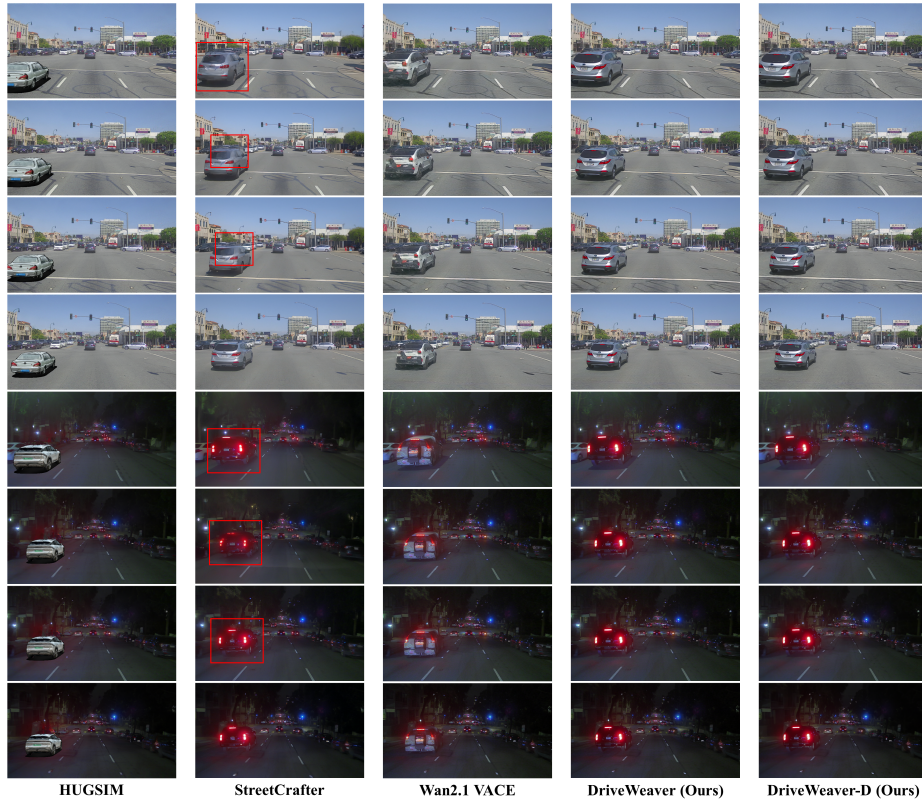


Fig. 5: Qualitative comparison on the PandaSet [46] dataset.

mance. More critically, temporal inconsistency breaks the geometric constraints required for 3D reconstruction, causing catastrophic failures in downstream reconstruction task, as reflected by extremely low PSNR and SSIM values. Without explicit point cloud conditioning, Wan2.1 VACE-14B [20] struggles to infer strict rigid-body dynamics from mask information alone, resulting in suboptimal performance in both visual realism and geometric consistency.

The same performance trends and the superiority of DriveWeaver are consistently observed on the PandaSet [46], as shown in Tab. 2 and Fig. 5, demonstrating the robustness of our paradigm across datasets. These results verify that DriveWeaver seamlessly synthesizes realistic and environmentally coherent textures for inserted vehicles, while preserving the strict and long-term geometric consistency required for reliable autonomous driving simulation. For a more comprehensive evaluation of visual realism and geometric consistency, please refer to our supplementary material.

Table 1: Comparison to state-of-the-art methods on Waymo [38]. We color the top results as **best** and **second best**.

Methods	Visual Realism						Geometric Consistency			
	Short-term			Long-term			Short-term		Long-term	
	FID↓	FVD↓	LPIPS↓	FID↓	FVD↓	LPIPS↓	PSNR↑	SSIM↑	PSNR↑	SSIM↑
Ground truth	0	0	0	0	0	0	36.44	0.974	33.56	0.953
HUGSIM [61]	49.06	325.89	0.304	56.99	150.95	0.308	36.05	0.973	33.96	0.966
StreetCrafter [50]	26.88	295.06	0.294	28.49	105.70	0.285	30.31	0.878	28.81	0.871
Wan2.1 VACE-14B [20]	54.67	264.74	0.171	35.12	63.90	0.169	34.47	0.956	31.99	0.930
DriveWeaver (Ours)	15.50	80.17	0.157	17.50	22.46	0.159	34.69	0.960	32.23	0.931
DriveWeaver-D (Ours)	22.35	119.36	0.168	24.28	38.50	0.172	—			

Table 2: Comparison to state-of-the-art methods on PandaSet [46]. We color the top results as **best** and **second best**.

Methods	Visual Realism			Geometric Consistency	
	FID↓	FVD↓	LPIPS↓	PSNR↑	SSIM↑
Ground truth	0	0	0	34.95	0.956
HUGSIM [61]	56.77	530.58	0.204	34.89	0.954
StreetCrafter [50]	35.46	359.25	0.295	29.10	0.833
Wan2.1 VACE-14B [20]	42.74	309.43	0.196	31.11	0.924
DriveWeaver (Ours)	21.54	146.15	0.182	31.34	0.927
DriveWeaver-D (Ours)	25.69	208.75	0.203	—	

4.3 Ablation study

We ablate key architectural designs on the challenging Waymo long-term generation, as shown in Tab. 3 and Fig. 6.

(a) PointAdapter vs. full finetuning. Directly finetuning the Wan backbone causes FVD to more than double from 22.46 to 62.59, indicating catastrophic forgetting of the pretrained prior. The limited autonomous driving data cannot preserve the texture and motion knowledge from large-scale pretraining, confirming that lightweight adapter-based conditioning is essential for visual fidelity.

(b) Point cloud rendering vs. naive projection. Replacing our rendering with naive 2D point projections yields overly sparse geometric conditions, causing the model to produce blurry outputs that lack high-frequency textural details. FVD degrades from 22.46 to 28.53, underscoring the importance of dense rasterized point guidance for reliable geometric supervision.

(c) Global-to-local vs. autoregressive generation. Reverting to a standard autoregressive paradigm primarily compromises temporal coherence: FVD rises from 22.46 to 29.41 as per-step prediction error progressively accumulates over time. Without global anchor frames to enforce long-range identity and pose constraints, the inserted vehicles gradually drift in appearance and deform in shape over extended trajectories, confirming that the global-to-local hierarchical strategy is indispensable for maintaining long-term consistency.

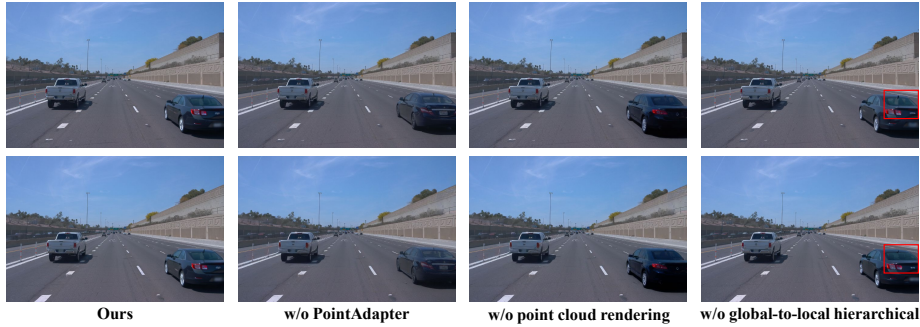


Fig. 6: Ablation study on architectural designs.

Table 3: Ablation study on architectural designs. We evaluate metrics on the challenging Waymo long-term generation.

	Visual Realism			Geometric Consistency	
	FID↓	FVD↓	LPIPS↓	PSNR↑	SSIM↑
(a) w/o PointAdapter	28.82	62.59	0.166	32.13	0.931
(b) w/o point cloud rendering	21.08	28.53	0.160	31.90	0.929
(c) w/o global-to-local hierarchical	18.31	29.41	0.160	31.89	0.929
DriveWeaver (Ours)	17.50	22.46	0.159	32.23	0.931

5 Conclusion

We present DriveWeaver, a novel point-conditioned video inpainting diffusion model designed for controllable vehicle insertion in autonomous driving simulation. By conditioning the inpainting process on rendered vehicle point cloud maps, the model generates high-quality foreground vehicles that are both environmentally harmonious and geometrically consistent. To support extended durations, we introduce a global-to-local hierarchical inpainting strategy, which ensures the long-term temporal consistency of the inserted objects. Furthermore, by distilling the generated foreground into 3D Gaussian representations, our framework facilitates the real-time rendering requirements of autonomous driving simulators. Extensive comparative evaluations across multiple datasets demonstrate the effectiveness of the proposed method, providing a robust tool for scalable scene augmentation in autonomous driving simulation.

Acknowledgements

This work was supported in part by New Generation Artificial Intelligence-National Science and Technology Major Project (2025ZD0123004), Ningbo grant (2025Z038) and National Natural Science Foundation of China (Grant No.62376060)

References

1. Chen, X., Wang, Y., Zhang, L., Zhuang, S., Ma, X., Yu, J., Wang, Y., Lin, D., Qiao, Y., Liu, Z.: Seine: Short-to-long video diffusion model for generative transition and prediction. In: ICLR (2023)
2. Chen, Y., Gu, C., Jiang, J., Zhu, X., Zhang, L.: Periodic vibration gaussian: Dynamic urban scene reconstruction and real-time rendering. arXiv preprint (2023)
3. Chen, Y., Zhang, J., Xie, Z., Li, W., Zhang, F., Lu, J., Zhang, L.: S-nerf++: Autonomous driving simulation via neural reconstruction and generation. IEEE TPAMI (2025)
4. Chen, Z., Yang, J., Huang, J., de Lutio, R., Esturo, J.M., Ivanovic, B., Litany, O., Gojcic, Z., Fidler, S., Pavone, M., Song, L., Wang, Y.: Omnire: Omni urban scene reconstruction. arXiv preprint (2024)
5. Chung, H.W., Constant, N., Garcia, X., Roberts, A., Tay, Y., Narang, S., Firat, O.: Unimax: Fairer and more effective language sampling for large-scale multilingual pretraining. arXiv preprint (2023)
6. Dosovitskiy, A., Ros, G., Codevilla, F., Lopez, A., Koltun, V.: Carla: An open urban driving simulator. In: CoRL (2017)
7. Du, X., Wang, Y., Sun, H., Wu, Z., Sheng, H., Wang, S., Ying, J., Lu, M., Zhu, T., Zhan, K., Yu, X.: 3drealcar: An in-the-wild rgb-d car dataset with 360-degree views. In: ICCV (2025)
8. Esser, P., Kulal, S., Blattmann, A., Entezari, R., Müller, J., Saini, H., Levi, Y., Lorenz, D., Sauer, A., Boesel, F., et al.: Scaling rectified flow transformers for high-resolution image synthesis. In: ICML (2024)
9. Gao, C., Ding, L., Cai, X., Huang, Z., Wang, Z., Xue, T.: Lora-edit: Controllable first-frame-guided video editing via mask-aware lora fine-tuning. arXiv preprint (2025)
10. Gao, H., Chen, S., Jiang, B., Liao, B., Shi, Y., Guo, X., Pu, Y., Yin, H., Li, X., Zhang, X., Zhang, Y., Liu, W., Zhang, Q., Wang, X.: Rad: Training an end-to-end driving policy via large-scale 3dgs-based reinforcement learning. arXiv preprint (2025)
11. 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: NIPS (2024)
12. Guo, Y., Yang, C., Rao, A., Wang, Y., Qiao, Y., Lin, D., Dai, B.: Animatediff: Animate your personalized text-to-image diffusion models without specific tuning. arXiv preprint (2023)
13. H. Caesar, J. Kabzan, K.T.e.a.: Nuplan: A closed-loop ml-based planning benchmark for autonomous vehicles. In: CVPR ADP3 workshop (2021)
14. Henschel, R., Khachatryan, L., Hayrapetyan, D., Poghosyan, H., Tadevosyan, V., Wang, Z., Navasardyan, S., Shi, H.: Streamingt2v: Consistent, dynamic, and extendable long video generation from text. In: CVPR (2025)
15. Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Hochreiter, S.: Gans trained by a two time-scale update rule converge to a local nash equilibrium. In: NIPS (2017)
16. Ho, J., Jain, A., Abbeel, P.: Denoising diffusion probabilistic models. arXiv preprint (2020)
17. Ho, J., Salimans, T.: Classifier-free diffusion guidance. arXiv preprint (2022)
18. Hu, Y.T., Wang, H., Ballas, N., Grauman, K., Schwing, A.G.: Proposal-based video completion. In: ECCV (2020)

19. Jiang, J., Song, N., Li, J., Zhu, X., Zhang, L.: Realengine: Simulating autonomous driving in realistic context. *arXiv preprint* (2025)
20. Jiang, Z., Han, Z., Mao, C., Zhang, J., Pan, Y., Liu, Y.: Vace: All-in-one video creation and editing. *arXiv preprint* (2025)
21. Kerbl, B., Kopanas, G., Leimkühler, T., Drettakis, G.: 3d gaussian splatting for real-time radiance field rendering. In: *ACM TOG* (2023)
22. Kong, W., Tian, Q., Zhang, Z., Min, R., Dai, Z., Zhou, J., Xiong, J., Li, X., Wu, B., Zhang, J., et al.: Hunyuanvideo: A systematic framework for large video generative models. *arXiv preprint* (2024)
23. Li, J., Jiang, J., Miao, J., Long, M., Wen, T., Jia, P., Liu, S., Yu, C., Liu, M., Cai, Y., et al.: Realistic and controllable 3d gaussian-guided object editing for driving video generation. *arXiv preprint* (2025)
24. Liang, R., Gojcic, Z., Nimier-David, M., Acuna, D., Vijaykumar, N., Fidler, S., Wang, Z.: Photorealistic object insertion with diffusion-guided inverse rendering. In: *ECCV* (2024)
25. Liang, Y., Yan, Z., Chen, L., Zhou, J., Yan, L., Zhong, S., Zou, X.: Driveeditor: A unified 3d information-guided framework for controllable object editing in driving scenes. In: *AAAI* (2025)
26. Lipman, Y., Chen, R.T.Q., Ben-Hamu, H., Nickel, M., Le, M.: Flow matching for generative modeling. In: *ICLR* (2023)
27. 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: *CVPR Workshops* (2026)
28. Ljungbergh, W., Tonderski, A., Johnander, J., Caesar, H., Åström, K., Felsberg, M., Petersson, C.: Neuroncap: Photorealistic closed-loop safety testing for autonomous driving. In: *ECCV* (2024)
29. Lu, Y., Cai, Y., Zhang, S., Zhou, H., Hu, H., Yu, H., Geiger, A., Liao, Y.: Urbancad: Towards highly controllable and photorealistic 3d vehicles for urban scene simulation. In: *CVPR* (2025)
30. Ma, Z., Jiang, J., Chen, Y., Zhang, L.: Béziergs: Dynamic urban scene reconstruction with bézier curve gaussian splatting. In: *ICCV* (2025)
31. 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)
32. Ost, J., Mannan, F., Thuerey, N., Knodt, J., Heide, F.: Neural scene graphs for dynamic scenes. In: *CVPR* (2021)
33. Peebles, W., Xie, S.: Scalable diffusion models with transformers. In: *ICCV* (2023)
34. Qiu, H., Yu, N., Huang, Z., Debevec, P., Liu, Z.: Cinescale: Free lunch in high-resolution cinematic visual generation. *arXiv preprint* (2025)
35. Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B.: High-resolution image synthesis with latent diffusion models. In: *CVPR* (2022)
36. Shah, S., Dey, D., Lovett, C., Kapoor, A.: Airsim: High-fidelity visual and physical simulation for autonomous vehicles. In: *FSR* (2017)
37. Singh, B., Kulharia, V., Yang, L., Ravichandran, A., Tyagi, A., Shrivastava, A.: Genmm: Geometrically and temporally consistent multimodal data generation for video and lidar. *arXiv preprint* (2024)
38. Sun, P., Kretschmar, H., Dotiwala, X., Chouard, A., Patnaik, V., Tsui, P., Guo, J., Zhou, Y., Chai, Y., Caine, B., et al.: Scalability in perception for autonomous driving: Waymo open dataset. In: *CVPR* (2020)

39. Turki, H., Zhang, J.Y., Ferroni, F., Ramanan, D.: Suds: Scalable urban dynamic scenes. *arXiv preprint* (2023)
40. Unterthiner, T., van Steenkiste, S., Kurach, K., Marinier, R., Michalski, M., Gelly, S.: Towards accurate generative models of video: A new metric & challenges. *arXiv preprint* (2018)
41. Wan, T., Wang, A., Ai, B., Wen, B., Mao, C., Xie, C.W., Chen, D., Yu, F., Zhao, H., Yang, J., Zeng, J., Wang, J., Zhang, J., Zhou, J., Wang, J., Chen, J., Zhu, K., Zhao, K., Yan, K., Huang, L., Feng, M., Zhang, N., Li, P., Wu, P., Chu, R., Feng, R., Zhang, S., Sun, S., Fang, T., Wang, T., Gui, T., Weng, T., Shen, T., Lin, W., Wang, W., Wang, W., Zhou, W., Wang, W., Shen, W., Yu, W., Shi, X., Huang, X., Xu, X., Kou, Y., Lv, Y., Li, Y., Liu, Y., Wang, Y., Zhang, Y., Huang, Y., Li, Y., Wu, Y., Liu, Y., Pan, Y., Zheng, Y., Hong, Y., Shi, Y., Feng, Y., Jiang, Z., Han, Z., Wu, Z.F., Liu, Z.: Wan: Open and advanced large-scale video generative models. *arXiv preprint* (2025)
42. Wang, C., Huang, H., Han, X., Wang, J.: Video inpainting by jointly learning temporal structure and spatial details. In: *AAAI* (2019)
43. Wang, Q., Fan, L., Wang, Y., Chen, Y., Zhang, Z.: Freevs: Generative view synthesis on free driving trajectory. In: *ICLR* (2025)
44. Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P.: Image quality assessment: from error visibility to structural similarity. *TIP* (2004)
45. Wang, Z., Shen, T., Gao, J., Huang, S., Munkberg, J., Hasselgren, J., Gojcic, Z., Chen, W., Fidler, S.: Neural fields meet explicit geometric representations for inverse rendering of urban scenes. In: *CVPR* (2023)
46. Xiao, P., Shao, Z., Hao, S., Zhang, Z., Chai, X., Jiao, J., Li, Z., Wu, J., Sun, K., Jiang, K., Wang, Y., Yang, D.: Pandaset: Advanced sensor suite dataset for autonomous driving. In: *ITSC* (2021)
47. Xie, Z., Zhang, J., Li, W., Zhang, F., Zhang, L.: S-nerf: Neural radiance fields for street views. In: *ICLR* (2023)
48. Xu, R., Li, X., Zhou, B., Loy, C.C.: Deep flow-guided video inpainting. In: *CVPR* (2019)
49. 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)
50. Yan, Y., Xu, Z., Lin, H., Jin, H., Guo, H., Wang, Y., Zhan, K., Lang, X., Bao, H., Zhou, X., Peng, S.: Streetcrafter: Street view synthesis with controllable video diffusion models. In: *CVPR* (2025)
51. Yang, J., Huang, J., Chen, Y., Wang, Y., Li, B., You, Y., Sharma, A., Igl, M., Karkus, P., Xu, D., et al.: Storm: Spatio-temporal reconstruction model for large-scale outdoor scenes. *arXiv preprint* (2024)
52. Yang, J., Ivanovic, B., Litany, O., Weng, X., Kim, S.W., Li, B., Che, T., Xu, D., Fidler, S., Pavone, M., Wang, Y.: Emernerf: Emergent spatial-temporal scene decomposition via self-supervision. *arXiv preprint* (2023)
53. Yang, Z., Chen, Y., Wang, J., Manivasagam, S., Ma, W.C., Yang, A.J., Urtasun, R.: Unisim: A neural closed-loop sensor simulator. In: *CVPR* (2023)
54. Yang, Z., Pan, Z., Yang, Y., Zhu, X., Zhang, L.: Driving scene synthesis on free-form trajectories with generative prior. In: *ICCV* (2025)
55. Yang, Z., Teng, J., Zheng, W., Ding, M., Huang, S., Xu, J., Yang, Y., Hong, W., Zhang, X., Feng, G., et al.: Cogvideox: Text-to-video diffusion models with an expert transformer. *arXiv preprint* (2024)
56. Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The unreasonable effectiveness of deep features as a perceptual metric. In: *CVPR* (2018)

57. Zhang, Y., Cao, C., Yu, C., Zhu, J.: Lion-lora: Rethinking lora fusion to unify controllable spatial and temporal generation for video diffusion. In: ICCV (2025)
58. 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. In: CVPR (2025)
59. Zheng, S., Yin, M., Hu, W., Li, X., Shan, Y., Fu, Y.: Versecrafter: Dynamic realistic video world model with 4d geometric control. arXiv preprint (2026)
60. Zheng, Z., Peng, X., Yang, T., Shen, C., Li, S., Liu, H., Zhou, Y., Li, T., You, Y.: Open-sora: Democratizing efficient video production for all. arXiv preprint (2024)
61. Zhou, H., Lin, L., Wang, J., Lu, Y., Bai, D., Liu, B., Wang, Y., Geiger, A., Liao, Y.: Hugsim: A real-time, photo-realistic and closed-loop simulator for autonomous driving. arXiv preprint (2024)
62. Zhou, H., Shao, J., Xu, L., Bai, D., Qiu, W., Liu, B., Wang, Y., Geiger, A., Liao, Y.: Hugs: Holistic urban 3d scene understanding via gaussian splatting. In: CVPR (2024)
63. Zhou, S., Li, C., Chan, K.C., Loy, C.C.: ProPainter: Improving propagation and transformer for video inpainting. In: ICCV (2023)
64. Zhou, Y., Ye, N., Ljungbergh, W., Li, T., Yang, J., Yang, Z., Zhu, H., Petersson, C., Li, H.: Decoupled diffusion sparks adaptive scene generation. In: ICCV (2025)
65. Zou, X., Yang, L., Liu, D., Lee, Y.J.: Progressive temporal feature alignment network for video inpainting. In: CVPR (2021)