DRIVEREGEN: GENERATING DRIVING SCENES AS GAUSSIANS IN RECONSTRUCTION SPACE

Anonymous authors
Paper under double-blind review

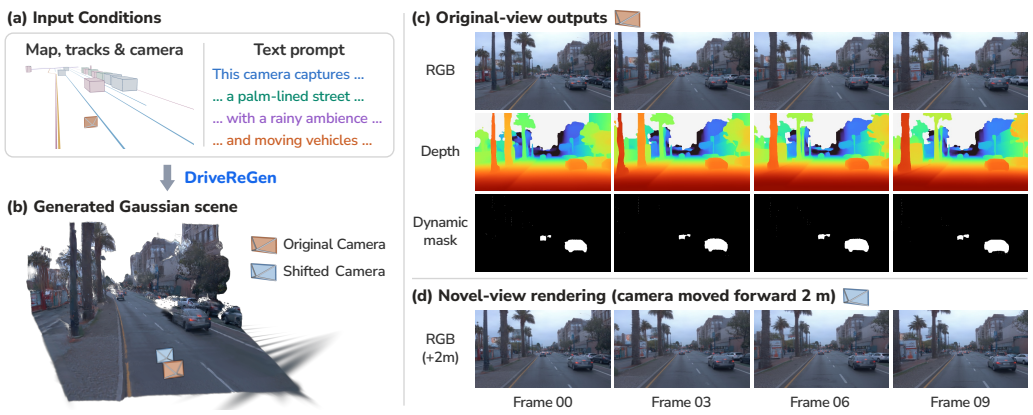


Figure 1: Given the input conditions, DriveReGen generates dynamic driving scenes as Gaussians within the reconstruction space of a feed-forward model. We show RGB renderings, depth, and dynamic masks across frames, together with novel views rendered from the same scene after moving the camera forward by 2 m. The actor tracks and Gaussian scene on the left are shown at frame 0.

ABSTRACT

Collecting real driving data is costly. Video models offer an alternative by generating driving scenes from text and layouts, yet they control these layouts only in 2D space and cannot ensure consistent geometry across frames and views. Recent methods generate Gaussian scenes from generated videos or video latents, but the reconstructed geometry may not fully match the generated appearance. General-scene methods instead generate in feed-forward reconstruction state space to jointly model geometry and appearance. However, conditioning on metric driving layouts requires the frozen decoder's scale and FOVs. We introduce DriveReGen, which generates driving scenes in reconstruction space with scene-specific scale and decoder fields of view for metric layout conditioning. Our two-stage layout control first aligns content with projected HD maps and actor tracks, then uses decoded calibration to encode metric layouts as reconstruction rays and depths. To supervise decoded outputs, Hierarchical Decoding Supervision (HDS) backpropagates feature, geometry, and image errors through frozen decoding. Given text, cameras, and layouts, DriveReGen generates 3D Gaussians without target images or LiDAR. On Waymo, DriveReGen reduces FVD by 58.0% and improves lane F1 by 3.77 percentage points over InfiniCube. Under physical cameras, its road/actor LiDAR AbsRel is 17.0% lower. Code will be made publicly available.

1 INTRODUCTION

Autonomous driving relies on diverse training data, but collecting real data is costly. Video diffusion models (Zhao et al., 2025b; Li et al., 2024a) offer an alternative by generating driving videos from text, with HD maps and actor tracks defining the scene layout. These layouts are projected into camera views to specify where map elements and actors should appear in each image. However, matching these 2D projections does not ensure consistent 3D geometry across frames and views.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

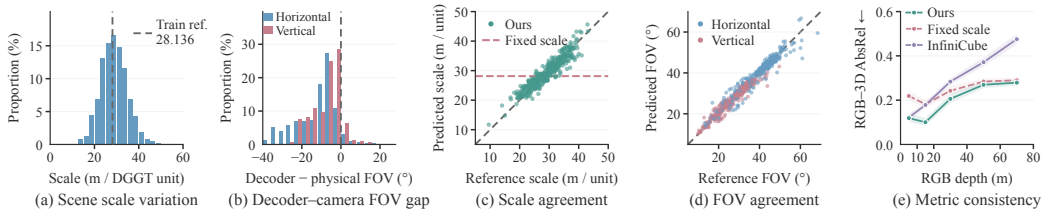


Figure 2: Scene calibration and metric RGB–3D consistency. **(a, b)** Metres per DGGT unit vary across training scenes, and decoder FOVs differ from physical-camera FOVs. **(c, d)** Generated scales and FOVs closely match their LiDAR-calibrated and DGGT-predicted references from real videos; dashed diagonals indicate exact agreement. **(e)** AbsRel compares Gaussian depth with metric pseudo-depth from each method's rendered RGB. The x-axis bins RGB pseudo-depth in metres; shading shows 95% scene-bootstrap CIs. Generated scale improves agreement over a fixed training scale, and DriveReGen has lower errors than InfiniCube in most depth ranges.

Due to the benefit of explicit 3D modeling, recent work (Wang et al., 2026a; Zhu et al., 2026; Zhang et al., 2025) builds Gaussian driving scenes from generated videos or video latents. These representations capture visual detail in camera views, while Gaussian reconstruction recovers a shared 3D scene. Other methods (Lu et al., 2025b; Yang et al., 2025a) first generate coarse voxels or occupancy grids, project them to guide video generation, and reconstruct Gaussians from the grids and videos. However, the reconstructed geometry may not fully match the appearance in those videos.

To jointly model geometry and appearance, general-scene methods (Gao et al., 2026c) instead generate Gaussian scenes in the state space of feed-forward reconstruction models. For driving scenes, models such as DGGT (Chen et al., 2026) provide such states, but we find that their use in layout-controlled generation requires matching decoder calibration: metres per DGGT unit vary across scenes, and Gaussian decoder fields of view (FOVs) differ from those of physical cameras (Fig. 2(a, b)). This calibration converts metric layout depths and camera translations to decoder units while matching its camera rays. With no target images at generation time, this calibration must be inferred.

To this end, we introduce DriveReGen, which generates a driving scene with scene-specific calibration: scale and decoder FOVs that adapt metric layouts to the frozen decoder. Our flow model uses shared attention to exchange information across scene, sky, and calibration tokens. To control scene generation with metric layouts, we use two-stage conditioning at each flow step. First, we project HD maps and actor tracks with known cameras to guide shared attention without requiring the generated calibration. Second, we use decoded calibration to express metric layouts as per-view rays and depths in reconstruction space. The model uses these cues to learn where to place objects in 3D reconstruction space and how large to make them. The resulting latents are decoded into Gaussians for rendering, but latent-space objectives alone do not directly constrain these outputs. We therefore train the generator with Hierarchical Decoding Supervision (HDS), backpropagating feature, geometry, and image errors through the frozen decoding path.

Given text, calibrated requested cameras, HD maps, and actor tracks, DriveReGen outputs 3D Gaussian scenes, depth maps, and dynamic maps. Inference requires neither target-scene images nor LiDAR. On Waymo, DriveReGen reduces FVD by 58.0% and increases lane F1 by 3.77 percentage points over InfiniCube. Physical-camera tests show better map alignment at 2 m and 4 m offsets and 17.0% lower original-view road/actor LiDAR AbsRel (Tab. 3). Our main contributions are:

(1) We propose DriveReGen to generate 3D Gaussian driving scenes from metric layouts in reconstruction space. Generated scale converts layout depths to reconstruction units, and decoder FOVs define their rays; this calibration improves layout alignment and depth consistency.

(2) We introduce two-stage layout control combining projected layouts with reconstruction-space rays and depths. At each flow step, generated scale and decoder FOVs define per-view metric conditions for the map and actor adapters, improving layout alignment and depth consistency.

(3) We train the generator with Hierarchical Decoding Supervision (HDS), backpropagating feature, geometry, and image errors through frozen decoding. Ablation studies on Waymo show that HDS improves rendered video quality and layout alignment without extra inference cost.

2 Related Work

2.1 Reconstruction States for Generation

Feed-forward reconstruction models (Wang et al., 2024; 2025; 2026c) recover geometry and camera parameters from unposed images without per-scene optimization. AnySplat (Jiang et al., 2025) extends this framework to Gaussian reconstruction, while DGGT (Chen et al., 2026) supports dynamic driving scenes. Recent work (Jang et al., 2026; Huang et al., 2026; Sun et al., 2026; Gao et al., 2026c; Weijler et al., 2026) generates in geometry foundation model features or token spaces, making reconstruction states an alternative to image latents. DriveReGen uses scene-specific scale and decoder FOVs to adapt metric layouts for generation in DGGT reconstruction space.

2.2 Driving Scene Generation with 3D Gaussians

General-domain methods decode 3D/4D Gaussians from generated RGB-D or video latents (Yang et al., 2025b; Pan et al., 2026); some also generate camera trajectories or use rendering-guided sampling (Go et al., 2025; Shabanov et al., 2026). For geometric supervision, Wu et al. (2026) align video and geometry features, while Gao et al. (2026b) use reconstruction and rendering losses but require target views. For scale, Yang et al. (2025b) use scale-invariant depth and Li et al. (2024b) normalize camera trajectories, but neither method generates metric scale. Metric generation has been studied in indoor scene synthesis (Ainetter et al., 2026) and unbounded world models (Joshi et al., 2026), but unposed image reconstruction lacks absolute scale, which is essential for metric driving conditions. DriveReGen uses scene-specific metric scale and decoder FOVs to encode metric layouts as per-view ray and depth cues that guide scene generation through the frozen decoder.

Driving scene generation uses multi-view videos (Gao et al., 2026a) and voxel or occupancy representations (Lu et al., 2025b; Yang et al., 2025a; Li et al., 2026). WorldSplat (Zhu et al., 2026) and CVD-STORM (Zhang et al., 2025) decode Gaussians from view-aligned latents, whereas ScenDi (Guo et al., 2026) uses voxel latents. In these Gaussian pipelines, reconstruction and rendering losses train tokenizers or Gaussian decoders, while latent objectives alone train the generators. Ost et al. (2026) apply such losses directly to splats but require thousands of steps per scene. DriveReGen trains the generator with feature, geometry, and image losses computed after decoding.

Another line of work starts from target-scene images. Observation-conditioned methods (Lu et al., 2025a; Mao et al., 2025; Chen & Peng, 2025; Zhao et al., 2025a; Ni et al., 2025; Yang et al., 2025c; Wang et al., 2026b; Park et al., 2026) reconstruct, extrapolate, or edit that scene rather than generating a new one. Song et al. (2026) backpropagate rendering losses through decoded Gaussians to a feed-forward 4D predictor, but condition on observations of the scene they extrapolate. DriveReGen generates 3D scenes without target-scene observations or per-scene fitting at inference.

3 Methods

3.1 Preliminaries

We first define the reconstruction state that DriveReGen generates, isolate the scale and field-of-view variables it does not expose, and state the resulting generation task over scene, sky and calibration.

Feed-forward reconstruction. DGGT (Chen et al., 2026) maps unposed images $\{I_i\}_{i=1}^{N}$ to a dynamic driving scene in one pass, taking the first camera as the world frame. Per frame it predicts extrinsics $[\mathbf{R}_i \mid \boldsymbol{\tau}_i]$ and fields of view $\phi_{x,i}, \phi_{y,i}$ parameterizing $\mathbf{K}_i$, a depth map d_i, a dynamic probability m_i, and pixel-aligned Gaussian attributes $\mathbf{a}_i$ (Kerbl et al., 2023) of colour, opacity, three axis scales $\boldsymbol{\sigma}$ of geometric mean $\bar{\sigma} = (\prod_j \sigma_j)^{1/3}$, and rotation. Its sky module renders backgrounds.

Reconstruction state. Its aggregator turns each image into P patch tokens, of which the dense heads read levels $\Lambda = \{4, 11, 17, 23\}$ of width $3C$ with $C = 1024$: a C-wide DINO part $\mathbf{D}_i^{\ell}$, and a $2C$-wide attention part $\mathbf{A}_i^{\ell}$ holding within-frame and cross-frame streams. The reconstruction state $\mathcal{F} = \{\mathbf{J}_i^{\ell}\}_{i,\ell}$, with $\mathbf{J}_i^{\ell} = [\mathbf{D}_i^{\ell}; \mathbf{A}_i^{\ell}]$, holds $4 \times 3C = 12{,}288$ values per patch, and the heads decode:

$$
\begin{aligned}
d_i &= \mathcal{H}_{\text{depth}}\big(\{\mathbf{A}_i^{\ell}\}_{\ell\in\Lambda}\big), \qquad \mathbf{a}_i = \mathcal{H}_{\text{gs}}\big(\{\mathbf{J}_i^{\ell}\}_{\ell\in\Lambda}\big), \\
m_i &= \mathcal{H}_{\text{dyn}}\big(\{\mathbf{D}_i^{\ell}\}_{\ell\in\Lambda}\big), \qquad \boldsymbol{\mu}_{i,p} = \mathbf{R}_i^{\top}\big(d_{i,p}\mathbf{K}_i^{-1}\bar{\mathbf{p}} - \boldsymbol{\tau}_i\big),
\end{aligned}
\tag{1}
$$

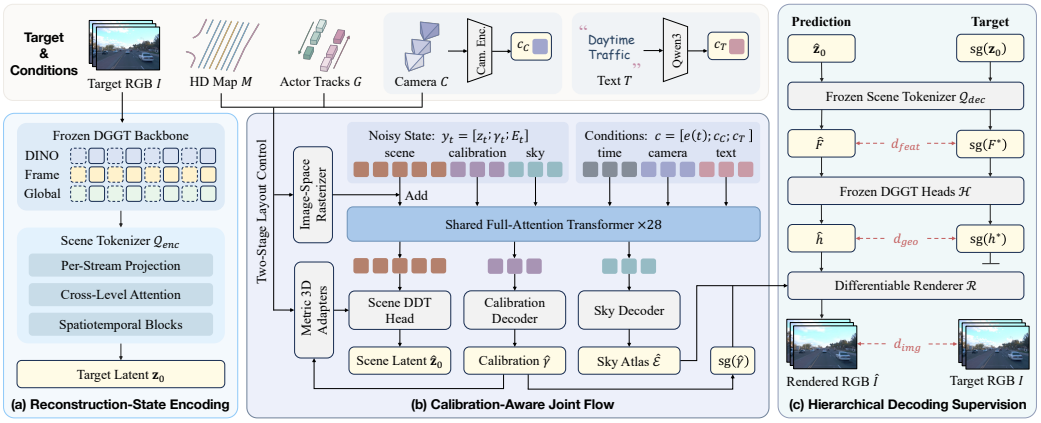


Figure 3: Training pipeline of DriveReGen. **(a)** The tokenizer compresses DGGT features of target RGB into scene latents while preserving decoded Gaussian size-to-depth ratios. **(b)** A joint flow denoises scene latents, calibration (metric scale and two decoder FOVs), and sky. Two-stage layout conditioning: image-space rasters align HD maps and actor tracks in requested views; calibration converts metric layouts into per-view ray and depth cues for map and actor adapters. **(c)** Hierarchical Decoding Supervision trains the flow with feature, geometry, and requested-view image losses backpropagated through frozen decoders and rendering. sg denotes stop-gradient.

with $\bar{\mathbf{p}}$ the homogeneous coordinate of pixel p. The head outputs are indexed by frame and pixel rather than by a 3D grid, are differentiable in $\mathcal{F}$, and are decoded from it without images. The camera head instead reads a separate camera token, so $\mathcal{F}$ itself carries no explicit camera parameterization.

Scale and field of view. The frozen heads decode geometry from $\mathcal{F}$, which lacks the calibration needed to align it with requested cameras and metric layouts. Metric scale is ambiguous: for $\lambda > 0$, scaling the scene and camera centres $\mathbf{O}_i$ alters neither the projection nor the size-to-depth ratio ρ,

$$\pi\big(\mathbf{K}_i\mathbf{R}_i(\lambda\mathbf{X} - \lambda\mathbf{O}_i)\big) = \pi\big(\mathbf{K}_i\mathbf{R}_i(\mathbf{X} - \mathbf{O}_i)\big), \qquad \rho_{i,p} = \frac{\bar{\sigma}_{i,p}}{d_{i,p}} = \frac{\lambda\bar{\sigma}_{i,p}}{\lambda d_{i,p}}. \tag{2}$$

Thus ρ is scale-invariant. DGGT inherits per-scene normalization of VGGT (Wang et al., 2025), making its unit clip-dependent and non-metric. FOV also requires calibration: decoder intrinsics can differ from the physical camera's $\mathbf{K}_i^{\text{phys}}$. Per-clip calibration $\gamma = (\eta_s, \eta_x, \eta_y)$ uses $\eta_s = \log s$ and $\eta_q = N^{-1}\sum_i \log\tan(\phi_{q,i}/2)$ for $q \in \{x, y\}$; s is metres per DGGT unit. During training, we compute median DGGT-to-LiDAR optical-axis depth ratios over valid pixels within frames, reject outlying frame estimates, and invert the median across retained frames to obtain s. FOV targets use DGGT predictions on target RGB. At inference, the model predicts γ without target RGB or LiDAR. Given text T, requested camera $\mathbf{C}$, HD map M, and actor tracks G, DriveReGen models $\mathcal{F}$, a directional RGB sky atlas $\mathcal{E}$ shared across views and used as the background at infinity, and γ:

$$p\big(\mathcal{F},\ \mathcal{E},\ \gamma \,\big|\, T,\ \mathbf{C},\ M,\ G\big). \tag{3}$$

3.2 Generating in the Reconstruction State

We compress the multi-level reconstruction state into a latent by a tokenizer, constrain decoded Gaussian size-to-depth ratios, and couple all outputs via full self-attention at a shared noise time.

Scene tokenizer. The frozen heads read the reconstruction state $\mathcal{F}$ from Sec. 3.1, but four levels with three 1,024-dimensional streams each give 12,288 variables per frame–patch—too large and redundant for a diffusion model to generate directly. We therefore encode each location as a 1,024-dimensional latent with $\mathcal{Q}_{\text{enc}}$. Since the levels describe the same location but the streams carry different signals, $\mathcal{Q}_{\text{enc}}$ projects each stream separately at every level and then pools the levels with a learned attention query. The tokenizer decoder $\mathcal{Q}_{\text{dec}}$ reconstructs four levels via this 12:1 bottleneck.

Preserving the size-to-depth ratio. Feature reconstruction of $\mathcal{F}$ does not directly constrain decoded size relative to depth. Rendering also admits size–opacity ambiguity: a smaller splat with higher

opacity can produce nearly the same image. The wrong ratio makes a splat too small or too large for its decoded depth. We therefore directly match the ratio at corresponding pixels:

$$\mathcal{L}_\rho = \mathbb{E}_{(i,p)\in\Omega} \left|\log \rho_{i,p}(\mathcal{Q}_{\mathrm{dec}}(\mathcal{Q}_{\mathrm{enc}}(\mathcal{F}))) - \log \rho_{i,p}(\mathcal{F})\right|. \tag{4}$$

Here $\rho_{i,p}(\mathcal{X})$ is decoded from state $\mathcal{X}$ by the frozen heads. The support Ω contains valid, static, non-sky pixels with sufficient opacity in the direct and round-trip decodes. The log difference assigns equal penalty to multiplicative ratio errors k and $1/k$ and vanishes exactly when size and depth change by the same local factor. App. A.1 details the support and auxiliary objectives. We pretrain the tokenizer with this loss and feature reconstruction, then freeze it for joint flow training.

Joint flow. The tokenizer encodes $\mathcal{F}$ into frame–patch tokens $\mathbf{z}$, whereas the other outputs have scene-wide scope: $\mathcal{E}$ is a directional RGB atlas shared by all views, and γ specifies one metric scale and two decoder FOVs for the clip. We concatenate these outputs as $\mathbf{y} = [\mathbf{z}; \mathcal{E}; \gamma]$ and train one conditional rectified flow to denoise them jointly (Liu et al., 2023). The three groups, indexed by r, share t but receive independent standard Gaussian noise. Following RAEv2's high-dimensional DiT design (Singh et al., 2026), a shared full-attention transformer processes all noisy outputs and conditions, giving the calibration decoder access to the current scene state. A DDT head predicts the clean scene component, while separate decoders predict the clean sky and calibration components, forming $\widehat{\mathbf{y}}_0 = [\widehat{\mathbf{z}}_0; \widehat{\mathcal{E}}; \widehat{\gamma}]$. The calibration decoder precedes the DDT head and supplies $\widehat{\gamma}$ for metric layout conditioning in the same flow step. The floor $\delta_t = \max(t, t_0)$ caps the loss weight.

$$\mathbf{y}_t = (1-t)\mathbf{y} + t\boldsymbol{\epsilon}, \quad \mathcal{L}_{\mathrm{RF}} = \sum_r \mathbb{E}\big[w_r(t)\delta_t^{-2}\|\widehat{\mathbf{y}}_0^r - \mathbf{y}^r\|_2^2\big]. \tag{5}$$

Apps. A.1 and A.2 detail sky atlas construction and calibration targets. App. A.4 specifies loss weights and t_0; App. A.5 gives the Euler grid, whose model calls all satisfy $t \geq t_0$.

Sliding-window generation. To generate longer clips, training includes crops that retain the parent clip's coordinates and calibration targets. Since all windows describe one scene, they must share its scale and FOV. We therefore blend overlapping-window velocities at each denoising step, jointly updating shared calibration and sky states with the full-sequence latent, as detailed in App. A.5.

3.3 Calibration-Aware Layout Conditioning

We use the generated decoder calibration $\widehat{\gamma}$ to express metric layouts M, G as per-view reconstruction rays and depths. Its scale $\widehat{s}$ also converts requested camera translations to reconstruction units. Our two-stage layout control uses these cues to guide scene content and learned 3D placement.

Layout encoding. Let $\mathbf{C} = \{\mathbf{K}_i^{\mathrm{phys}}, \mathbf{T}_i^{\mathrm{met}}\}_{i=1}^N$ specify physical intrinsics on the normalized canvas and metric camera-to-world poses. Generated calibration gives scale $\widehat{s} = e^{\widehat{\eta}_s}$ and decoder FOVs $\widehat{\phi}_q = 2\arctan(e^{\widehat{\eta}_q})$, $q \in \{x, y\}$, defining clip-shared $\mathbf{K}^{\mathrm{rec}} = \mathbf{K}(\widehat{\phi}_x, \widehat{\phi}_y)$. Requested intrinsics determine the layout projection, while generated FOVs define the decoder rays. For a metric camera-frame layout point $\mathbf{x}_i$ with positive axial depth $x_{i,z}$, physical projection gives the homogeneous pixel $\bar{\mathbf{p}}_i = (u_i, v_i, 1)^\top$. We then obtain its decoder ray $\mathbf{r}_i$ and conditioning coordinate $\mathbf{q}_i$ as

$$\bar{\mathbf{p}}_i = \mathbf{K}_i^{\mathrm{phys}}\mathbf{x}_i / x_{i,z}, \qquad \mathbf{r}_i = (\mathbf{K}^{\mathrm{rec}})^{-1}\bar{\mathbf{p}}_i, \qquad \mathbf{q}_i = (x_{i,z}/\widehat{s})\mathbf{r}_i. \tag{6}$$

Physical intrinsics determine layout pixels. Generated FOVs define decoder rays, and scale converts depth to reconstruction units. These coordinates guide generation through adapters, and frozen heads decode native Gaussians with decoder intrinsics and requested poses, bypassing the camera head. Default physical-K export maps centres and covariances per source camera and retains temporal fusion (App. A.2). Centre mapping equals depth backprojection with physical intrinsics. At inference, we render the exported scene with physical intrinsics and requested poses whose translations are divided by $\widehat{s}$. At a fixed timestamp, camera shifts change only the rendering camera.

Two-stage layout control. M, G and the requested camera $\mathbf{C}$ share the same metric world frame, so $\mathbf{C}$ directly determines their requested-view 2D projections. Our second stage uses the predicted $\widehat{\gamma}$, which is recovered by the calibration decoder only after the shared transformer, to encode this metric geometry as reconstruction-space cues. We thus use two stages per flow step: image-space conditioning before calibration decoding, then calibration-guided metric conditioning.

The first stage aligns scene content with the projected layout. $\mathbf{C}$ defines the requested views, so projecting M, G under $\mathbf{C}$ gives target locations for map elements and actors. We encode their dimensionless rasters as patch features and add them to the embeddings of $\mathbf{z}_t$ before the shared transformer. This yields view-aligned map and actor features on each frame's patch grid.

The second stage uses clean $\widehat{\gamma}$ to apply Eq. 6, expressing metric map samples and actor box corners as reconstruction rays and depths at the same pixels. For maps, each patch retains the nearest valid sample per semantic group, represented by ray ratios and log depth with validity flags for missing groups. For actors, we combine the eight mapped box corners with normalized camera-frame velocity. We encode map and actor features with separate MLPs, then pool actor embeddings onto patches using projected coverage and depth weights that favor nearer actors where projections overlap. The resulting patch-aligned features modulate DDT blocks through adaptive normalization and residual gating, providing position and size cues at the corresponding layout pixels.

Together, the first stage aligns scene content with the projected layout, while the second uses clean $\widehat{\gamma}$ to supply per-view ray and depth cues for learned 3D placement through the DDT head.

3.4 Hierarchical Decoding Supervision

Sec. 3.2 trains a joint flow for the clean scene latent $\widehat{\mathbf{z}}_0$, sky atlas $\widehat{\mathcal{E}}$, and calibration $\widehat{\gamma}$ using $\mathcal{L}_{\mathrm{RF}}$ in Eq. 5. Because this loss precedes the tokenizer decoder, frozen reconstruction heads, and requested-view renderer, it does not directly constrain the decoded reconstruction state, dense-head predictions, or requested-view RGB. Hierarchical Decoding Supervision (HDS) adds feature, geometry, and requested-view image losses to improve rendering quality.

Our design principle is to use the frozen decoding path as a measurement function and compare predictions at increasingly task-specific interfaces. Let $\mathbf{z}_0 = \mathcal{Q}_{\mathrm{enc}}(\mathcal{F})$ be the clean training target. We decode $\widehat{\mathbf{z}}_0$ and $\mathbf{z}_0$ into $\widehat{\mathcal{F}} = \{\widehat{\mathbf{J}}_i^{\ell}\}_{i,\ell}$ and $\mathcal{F}^{\star} = \{\mathbf{J}_i^{\ell\star}\}_{i,\ell}$ with $\mathcal{Q}_{\mathrm{dec}}$ from Sec. 3.2. Feature-level supervision compares $\widehat{\mathbf{J}}_i^{\ell}$ and $\mathbf{J}_i^{\ell\star}$ at the four levels $\ell \in \Lambda$, constraining the complete multi-level state from which the frozen heads decode their outputs. Since token-level agreement does not directly constrain decoded quantities, geometry-level supervision compares all dense-head outputs from $\widehat{\mathcal{F}}$ and $\mathcal{F}^{\star}$: depth, Gaussian attributes, dynamics, and the depth and Gaussian confidence maps.

These two levels do not yet test whether the scene, sky, and calibration jointly form the target image. HDS renders the native decoded scene using $\Phi_{\mathrm{dec}}(\mathbf{C}, \widehat{\gamma})$, which rebases requested poses to the first-camera frame, divides translations by $\widehat{s}$, and uses generated decoder intrinsics. This training path precedes physical-camera export (App. A.2). We composite the native scene over $\widehat{\mathcal{E}}$ and compare it with target-window RGB I. Writing sg for stop-gradient, the objective is

$$
\begin{aligned}
\mathcal{L}_{\mathrm{HDS}} &= \lambda_{\mathrm{feat}} d_{\mathrm{feat}}(\widehat{\mathcal{F}}, \mathcal{F}^{\star}) + \lambda_{\mathrm{geo}} d_{\mathrm{geo}}(\widehat{\mathbf{h}}, \mathbf{h}^{\star}) + \lambda_{\mathrm{img}} d_{\mathrm{img}}(\widehat{I}, I) \\
&= \lambda_{\mathrm{feat}} d_{\mathrm{feat}}(\mathcal{Q}_{\mathrm{dec}}(\widehat{\mathbf{z}}_0), \mathrm{sg}[\mathcal{Q}_{\mathrm{dec}}(\mathbf{z}_0)]) && \text{(feature-level loss)} \\
&+ \lambda_{\mathrm{geo}} d_{\mathrm{geo}}\Big(\mathcal{H}(\widehat{\mathcal{F}}), \mathrm{sg}[\mathcal{H}(\mathcal{F}^{\star})]\Big) && \text{(geometry-level loss)} \\
&+ \lambda_{\mathrm{img}} d_{\mathrm{img}}\Big(\mathcal{R}\Big(\mathcal{H}(\widehat{\mathcal{F}}), \widehat{\mathcal{E}}, \Phi_{\mathrm{dec}}(\mathrm{sg}[\mathbf{C}], \mathrm{sg}[\widehat{\gamma}])\Big), I\Big). && \text{(image-level loss)}
\end{aligned}
\tag{7}
$$

Here $\mathcal{H}$ denotes the frozen heads and $\mathcal{R}$ the differentiable renderer. d_{feat} compares four reconstruction levels with normalized ℓ_1 and cosine distances. d_{geo} uses detached teacher depth confidence to weight robust distances on depth, Gaussian attributes, and dynamics while comparing confidences without this weighting. d_{img} combines masked Charbonnier, weighted by the same confidence, with LPIPS on requested-view RGB, jointly evaluating scene and sky rendering under fixed $\Phi_{\mathrm{dec}}(\mathbf{C}, \widehat{\gamma})$.

We stop target-branch gradients and freeze the tokenizer and heads. All three HDS terms supervise $\widehat{\mathbf{z}}_0$, and the image term also supervises $\widehat{\mathcal{E}}$. At each flow step, the same $\widehat{\gamma}$ conditions $\widehat{\mathbf{z}}_0$ through Stage 2 map and actor adapters, allowing all three terms to reach the calibration decoder. Stop-gradient in Φ_{dec} blocks only the direct camera-construction path. The training objective is

$$
\mathcal{L}_{\mathrm{train}} = \mathcal{L}_{\mathrm{RF}} + \mathcal{L}_{\mathrm{HDS}} + \mathcal{L}_{\mathrm{aux}}. \tag{8}
$$

Here $\mathcal{L}_{\mathrm{aux}}$ collects direct calibration and sky support objectives, detailed in App. A.4. During training, target frames supply RGB and FOV supervision, while LiDAR anchors metric scale.

Table 1: Visual quality and metric RGB–3D consistency of complete pipelines, including scale estimation (App. B.1). †References reconstruct observed images from DGGT features, directly or after tokenizer encoding and decoding. DriveReGen achieves the lowest FID, FVD, and AbsRel among generation methods, with strong text alignment measured by CLIP-T.

Method	Visual quality			Metric RGB–3D consistency		
	FID↓	FVD↓	CLIP-T↑	AbsRel↓	δ_1 ↑	Cov. (%)↑
InfiniCube	59.74	621.18	30.76	<u>0.2290</u>	<u>0.6435</u>	99.671
OpenDWM + DGGT	<u>52.72</u>	<u>410.66</u>	**31.21**	0.3170	0.5827	99.898
Cosmos3 + DGGT	64.83	782.83	30.82	0.4616	0.5452	<u>99.959</u>
DriveReGen (Ours)	**37.15**	**260.77**	<u>31.13</u>	**0.1581**	**0.7330**	**99.968**
DGGT direct†	17.27	85.46	31.67	0.1496	0.7388	99.985
Tokenizer Round Trip†	13.47	78.83	31.63	0.1512	0.7342	99.974

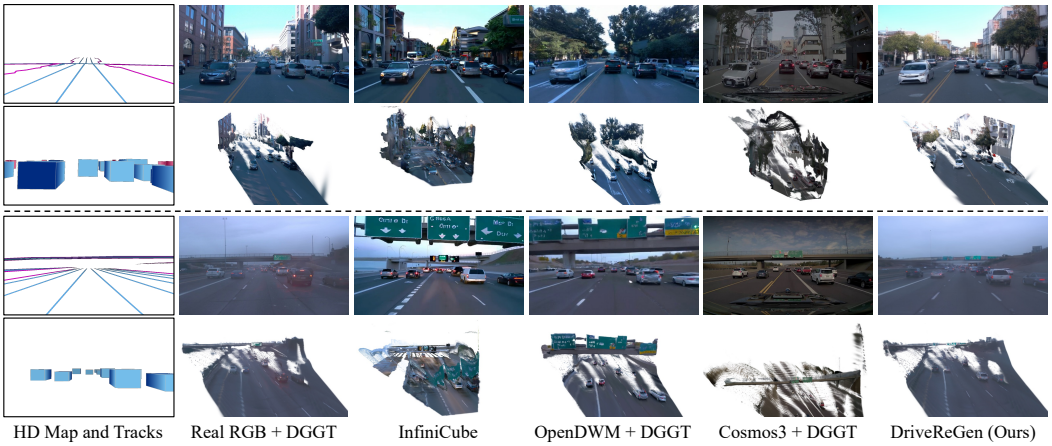


Figure 4: Qualitative comparison. RGB is shown above Gaussian renderings; static Gaussians are aggregated across all frames. Gaussians below real RGB are reconstructed by DGGT for visual reference only. InfiniCube has vehicle orientation errors; OpenDWM + DGGT blurs details; Cosmos3 shows a domain gap. DriveReGen yields more coherent geometry and more realistic rainy scenes.

4 Experiments

4.1 Experimental Setup

Benchmark. We evaluate scene generation, geometric control, camera offsets, and ablations on a shared benchmark of 404 front-view clips from the official Waymo validation split (Sun et al., 2020). We sample two clips per scene at fixed, temporally separated positions, with details in App. B.1.

Baselines. WorldSplat, ScenDi, and CVD-STORM lack complete pretrained Gaussian pipelines for direct evaluation (App. B.1). We compare (1) InfiniCube (Lu et al., 2025b), a voxel-guided video-to-Gaussian pipeline, and (2) two layout-controlled video models paired with frozen DGGT. OpenDWM CTSD-3.5 (OpenDWM) is SD3.5-based and Waymo-trained; Cosmos3-Nano (NVIDIA, 2026) supports map and actor control without Waymo-specific training. We include it to evaluate a practical alternative: Gaussian reconstruction from a strong pretrained video model. DriveReGen uses $(T, \mathbf{C}, M, G)$; baselines take (T, M, G), using the same $\mathbf{C}$ for HD-map rendering.

Metrics. (1) For *visual quality*, FID/FVD (Heusel et al., 2017; Unterthiner et al., 2019) measure image/video quality and CLIP-T text alignment. ΔFID measures changes after camera shifts. (2) For *metric RGB–3D consistency*, we compare Gaussian depth with frozen DA2 (Yang et al., 2024) pseudo-depth from the same rendered RGB, both in metres. AbsRel/δ_1 measure their agreement over valid pairs; coverage is the fraction of valid non-sky pseudo-depth pixels with Gaussian depth.

Table 2: Layout control under physical cameras at the original viewpoint. †Vehicle-only actor metrics, excluded from actor rankings. ‡Detections on real images. DriveReGen improves lane F1 by 1.35 percentage points and actor mAP by 0.0138 (12.4% relative) over OpenDWM + DGGT, with lower errors for matched lanes and actors, indicating closer alignment with the requested layout.

Method	HD map (lane detection)			Track (actor detection)			
	F1↑	R↑	X-Err↓	mAP↑	R↑	ATE↓	AOE↓
InfiniCube†	8.66	5.05	0.808	0.0945	0.1822	0.8179	0.3472
OpenDWM + DGGT	<u>11.08</u>	<u>6.55</u>	<u>0.746</u>	<u>0.1113</u>	<u>0.3283</u>	<u>0.7892</u>	<u>0.3616</u>
Cosmos3 + DGGT	10.17	6.18	0.788	0.0538	0.1426	0.8632	0.3775
DriveReGen (Ours)	**12.43**	**7.44**	**0.707**	**0.1251**	**0.3582**	**0.7682**	**0.3389**
Real images‡	16.14	9.80	0.702	0.1957	0.4616	0.7310	0.1689

Table 3: LiDAR-referenced road and actor depth. Generated scale improves decoder-K depth over the fixed training-set scale s_0. Full physical-K export outperforms InfiniCube on all three metrics, providing direct evidence of more accurate metric depth in layout-controlled regions.

Setting	AbsRel↓	RMSE (m)↓	δ_1↑
InfiniCube	0.0966	2.1598	0.8935
Ours, K^{rec}, s_0	0.1705	3.1743	0.6708
Ours, K^{rec}	0.0810	1.8740	0.9243
Ours, physical K	**0.0802**	**1.8289**	**0.9272**

LiDAR-referenced AbsRel/RMSE/δ_1 evaluate road and actor axial depth (Tab. 3). Calibration MAE uses LiDAR-calibrated log-scale and DGGT-predicted decoder-FOV references from real videos (FOV in degrees). (3) For *layout control*, frozen Waymo-pretrained monocular PGD (Wang et al., 2022) measures actor alignment via G-LET-mAP (Hung et al., 2024) (mAP), recall (R), ATE (translation error, m) and AOE (orientation error, rad). Frozen PersFormer (Chen et al., 2022) reports lane F1/R (%) and matched lateral X-error (m) against Waymo road-line/edge labels.

4.2 Scene Generation and Geometric Control

Generation quality. DriveReGen uses full physical-K export for all metrics in Tabs. 1 and 9, layout scores (Tabs. 2 and 6), and offsets (Tab. 4). Tab. 3 directly evaluates metric road and actor depth against LiDAR. DriveReGen leads generators in FID/FVD and RGB–3D AbsRel with strong text alignment, and outperforms original OpenDWM/Cosmos3 videos in FID/FVD. Fig. 4 shows InfiniCube's vehicle orientation errors and OpenDWM + DGGT's blurred details; DriveReGen renders clearer vehicles. Cosmos3's videos are high quality, but wide-angle views with visible hoods differ from Waymo, worsening FID/FVD. Our tokenizer uses longer training windows than DGGT; its round trip improves FID/FVD with similar RGB–3D consistency (App. B.5). App. B.7 reports 29-frame metrics and two 57-frame examples beyond 10-frame training.

LiDAR-referenced depth. Tab. 3 compares rendered axial depth with observed LiDAR on layout-controlled roads and actors. Full physical-K export lowers AbsRel/RMSE by $17.0\%/15.3\%$ relative to InfiniCube and raises δ_1 by 3.37 percentage points. These results support accurate metric depth in layout-controlled regions, complementing the monocular-prior agreement in Tab. 1.

Layout control. In Tab. 2, DriveReGen raises lane F1 from 11.08% to 12.43% over OpenDWM + DGGT and reduces lateral error. It also leads the actor rankings in mAP/recall and matched pose errors. InfiniCube reports vehicle-only scores; under this protocol, our mAP/recall is $0.3504/0.4846$ versus $0.0945/0.1822$ (Tab. 11). Higher detection scores and lower matched errors show closer alignment with the requested layouts than the video-to-Gaussian baselines. Fig. 5 shows qualitative examples; Tab. 10 additionally compares against original videos before DGGT reconstruction.

Camera offsets. We divide lateral offsets by each method's scale, keeping scenes and intrinsics fixed. At 4 m, our physical-K export reaches 7.31% lane F1 versus 2.67% for InfiniCube and ΔFID of 26.04 versus 33.28 (Tab. 4). This supports layout alignment under physical camera shifts.

Table 4: Lateral offsets of fixed Gaussian scenes. DriveReGen renders its physical-K export and leads all metrics at both offsets. Lane F1/recall and lateral X-error measure layout alignment; ΔFID compares each method with its zero-offset baseline against original-view real images.

Method	Lateral camera offset: 2 m				Lateral camera offset: 4 m			
	ΔFID↓	F1↑	R↑	X-Err↓	ΔFID↓	F1↑	R↑	X-Err↓
InfiniCube	<u>11.06</u>	<u>5.75</u>	<u>3.18</u>	<u>0.860</u>	<u>33.28</u>	<u>2.67</u>	<u>1.41</u>	<u>0.976</u>
OpenDWM + DGGT	23.07	5.63	3.11	0.935	64.25	2.24	1.18	1.100
Cosmos3 + DGGT	26.19	4.39	2.42	1.040	73.65	1.52	0.79	1.134
DriveReGen (Ours)	**9.27**	**11.52**	**6.64**	**0.748**	**26.04**	**7.31**	**4.01**	**0.822**

Table 5: Calibration ablations after sampling with fixed latents. Physical K for decoding replaces decoder intrinsics. MAE compares with LiDAR scale and DGGT FOVs; u: unchanged.

Setting	Reference MAE			RGB–depth consistency		
	log-s MAE↓	FOV$_x$ (°)↓	FOV$_y$ (°)↓	AbsRel↓	δ_1 ↑	Cov. (%)↑
Ours	**0.0541**	**1.589**	**1.121**	**0.1581**	**0.7330**	**99.968**
Ours w/ fixed scale	0.1600	1.589^{u}	1.121^{u}	0.2123	0.5880	99.968^{u}
Physical K for decoding	0.0541^{u}	8.287	5.332	0.1747	0.6912	99.959

4.3 Ablation Studies

Tab. 5 changes calibration after sampling with scene latents fixed. Physical K for decoding is an ablation of decoder intrinsics, distinct from the default physical-K export. Tab. 6 retrains conditioning and supervision variants; App. B.1 details their settings and retained objectives.

In Tab. 5, predicted scale changes only depth conversion, reducing AbsRel by 25.5% and improving δ_1 by 14.50 percentage points over a training-set constant. Generated decoder FOVs improve layout control over training-set constants after sampling (Tab. 8). These results support scene-specific calibration for metric RGB–3D consistency and requested-view layout alignment.

Table 6: Training ablations of calibration and decoded supervision. HDS variants cumulatively add RGB, geometry, and feature supervision. Full performs best across all metrics.

Variant	FVD↓	AbsRel↓	F1↑	mAP↑
Fixed units	283.97	0.2413	11.51	0.0862
Fixed S2 γ	264.32	0.1668	11.40	0.0873
w/o HDS	288.33	0.1672	11.81	0.0919
HDS: RGB	274.85	0.1640	11.77	0.1150
HDS: Geo.	263.12	0.1616	12.08	0.1179
HDS: Full	260.77	0.1581	12.43	0.1251

Fixed units uses training-set scale and decoder-FOV constants during training and inference. *Fixed S2 γ* uses these constants only in Stage 2 encoding, retaining 3D inputs, adapters, and generated calibration for decoding. Fixed units worsens RGB–depth consistency and layout alignment, supporting scene-specific calibration. Fixing only Stage 2 weakens layout control, reducing actor mAP from 0.1251 to 0.0873. Full and Fixed S2 γ both retain joint generation, so their comparison isolates the contribution of calibration-aware conditioning.

w/o HDS retains flow and auxiliary objectives. We cumulatively add requested-view image losses (*HDS: RGB*), decoded geometry losses (*HDS: Geo.*), and reconstruction-feature losses (*HDS: Full*). RGB supervision improves FVD, depth consistency, and actor alignment, but not lane F1. Geometry losses improve all four metrics, supporting direct constraints on 3D outputs. Adding feature matching to *HDS: Geo.* lowers FVD from 263.12 to 260.77. Full lowers FVD by 27.56 over *w/o HDS*. HDS adds about 16% training time at the same update count but no inference overhead.

5 Conclusion

DriveReGen generates 3D Gaussian driving scenes from metric layouts in reconstruction space. Generated scale and decoder FOVs provide per-view ray and depth cues for two-stage layout control, while HDS supervises decoded geometry and rendering. On Waymo, results show better visual quality, physical-camera layout control and shifted-view map alignment, with lower road/actor LiDAR depth errors than InfiniCube. We discuss limitations and failure cases in App. C.

AI USE STATEMENT

In this work, we used generative AI tools to assist with experimental design and manuscript translation, and MiniMax-M3 to generate scene captions. We did not use generative AI to develop the core research ideas. Additionally, AI tools assisted with data-processing and evaluation code, parameter selection, and manuscript organization, drafting, and language editing. The authors inspected and tested the AI-assisted code, manually spot-checked the generated captions, and reviewed the manuscript. We take responsibility for the final content of this work, including AI-assisted manuscript text, scientific claims, implementation code, and other research artifacts.

ETHICS STATEMENT

DriveReGen generates controllable 3D driving scenes to support simulation and data augmentation research. Generated scenes could also be misrepresented as real traffic recordings. Moreover, generated scenes may inherit dataset biases or contain geometric errors. We therefore recommend clearly labeling synthetic outputs and checking their physical realism and scenario coverage before safety-critical use; they should complement, not replace, real-world safety validation. Data should be used and shared in accordance with the applicable terms of use and requirements for protecting personal privacy in source recordings and generated outputs.

REPRODUCIBILITY STATEMENT

We train on the Waymo training split using 32 NVIDIA A800 GPUs, BF16 data-parallel training, and a global batch size of 64. Front-view images are resized to a height of 350 pixels and a width of 518 pixels. Training clips contain 10–14 frames for tokenizer pretraining and 10 frames for flow training. The tokenizer is pretrained for 100,000 iterations using AdamW with a learning rate of 6×10^{-4} and weight decay of 0.05. Its schedule combines a 4,000-step linear warm-up with cosine decay. We freeze the pretrained tokenizer during joint flow training.

We train the joint flow for 100,000 iterations with GMuon for two-dimensional parameters and AdamW for the rest, using weight decay 0.01. GMuon uses momentum 0.95, Nesterov updates, and five Polar Express Newton–Schulz iterations. The learning rate stays at 10^{-4} for 4,000 steps, then decays linearly to 10^{-5}. Both stages use `LambdaLR` and AdamW $\epsilon = 10^{-8}$. HDS ramps linearly from zero at step 5,000 to full weight at step 10,000. We evaluate EMA weights with decay 0.9999.

Sampling uses 50 Euler steps, layout CFG scale 1, and text CFG scale 4 for scene, calibration, and sky. The 57-frame examples use 10-frame windows $[0, 10)$, $[7, 17)$, $[14, 24)$, $[19, 29)$, $[26, 36)$, $[33, 43)$, $[40, 50)$, and $[47, 57)$, with zero-based indexing. All CFG branches retain the requested cameras. App. A details architecture, loss weights, and window blending.

REFERENCES

Stefan Ainetter, Thomas Deixelberger, Edoardo A. Dominici, Philipp Drescher, Konstantinos Vardis, and Markus Steinberger. Scene generation at absolute scale: Utilizing semantic and geometric guidance from text for accurate and interpretable 3D indoor scene generation, 2026.

Li Chen, Chonghao Sima, Yang Li, Zehan Zheng, Jiajie Xu, Xiangwei Geng, Hongyang Li, Conghui He, Jianping Shi, Yu Qiao, and Junchi Yan. PersFormer: 3D lane detection via perspective transformer and the OpenLane benchmark. In *European Conference on Computer Vision (ECCV)*, 2022.

Shijie Chen and Peixi Peng. FreeGen: Feed-forward reconstruction-generation co-training for free-viewpoint driving scene synthesis, 2025.

Xiaoxue Chen, Ziyi Xiong, Yuantao Chen, Gen Li, Nan Wang, Hongcheng Luo, Long Chen, Haiyang Sun, Bing Wang, Guang Chen, Hongyang Li, Ya-Qin Zhang, Hangjun Ye, and Hao Zhao. DGGT: Feedforward 4D reconstruction of dynamic driving scenes using unposed images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1265–1276, June 2026.

Ruiyuan Gao, Kai Chen, Zhihao Li, Lanqing Hong, Zhenguo Li, and Qiang Xu. MagicDrive3D: Controllable 3D generation for any-view rendering in street scenes. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 5724–5733, March 2026a.

Sensen Gao, Zhaoqing Wang, Qihang Cao, Dongdong Yu, Changhu Wang, and Jia-Wang Bian. PixWorld: Unifying 3D scene generation and reconstruction in pixel space, 2026b.

Sensen Gao, Zhaoqing Wang, Qihang Cao, Dongdong Yu, Changhu Wang, Tongliang Liu, Mingming Gong, and Jiawang Bian. OneWorld: Taming scene generation with 3D unified representation autoencoder, 2026c.

Hyojun Go, Byeongjun Park, Jiho Jang, Jin-Young Kim, Soonwoo Kwon, and Changick Kim. SplatFlow: Multi-view rectified flow model for 3D gaussian splatting synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 21524–21536, June 2025. doi: 10.1109/CVPR52734.2025.02005.

Hanlei Guo, Jiahao Shao, Xinya Chen, Xiyang Tan, Sheng Miao, Yujun Shen, and Yiyi Liao. ScenDi: 3D-to-2D scene diffusion cascades for urban generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 40291–40302, June 2026.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs trained by a two time-scale update rule converge to a local Nash equilibrium. In *Advances in Neural Information Processing Systems*, volume 30, pp. 6626–6637, 2017.

Jiaxin Huang, Yuanbo Yang, Bangbang Yang, Lin Ma, Yuewen Ma, and Yiyi Liao. Gen3R: 3D scene generation meets feed-forward reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 25358–25369, June 2026.

Wei-Chih Hung, Vincent Casser, Henrik Kretzschmar, Jyh-Jing Hwang, and Dragomir Anguelov. LET-3D-AP: Longitudinal error tolerant 3D average precision for camera-only 3D detection. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 8272–8279. IEEE, 2024. doi: 10.1109/ICRA57147.2024.10609986.

Wooseok Jang, Seonghu Jeon, Jisang Han, Jinhyeok Choi, Minkyung Kwon, Seungryong Kim, Saining Xie, and Sainan Liu. Repurposing geometric foundation models for multi-view diffusion, 2026.

Lihan Jiang, Yucheng Mao, Linning Xu, Tao Lu, Kerui Ren, Yichen Jin, Xudong Xu, Mulin Yu, Jiangmiao Pang, Feng Zhao, Dahua Lin, and Bo Dai. AnySplat: Feed-forward 3D gaussian splatting from unconstrained views. *ACM Transactions on Graphics*, 44(6):1–16, December 2025. ISSN 1557-7368. doi: 10.1145/3763326.

Amogh Joshi, Julian Ost, and Felix Heide. WorldFlow3D: Flowing through 3D distributions for unbounded world generation, 2026.

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3D gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4):139:1–139:14, July 2023. doi: 10.1145/3592433.

Bohan Li, Zhuang Ma, Dalong Du, Baorui Peng, Zhujin Liang, Zhenqiang Liu, Xianda Guo, Zheng Zhu, Chao Ma, Yueming Jin, Xin Jin, Hao Zhao, and Wenjun Zeng. OmniNWM: Omniscient driving navigation world models. In *European Conference on Computer Vision (ECCV)*, 2026.

Xiaofan Li, Yifu Zhang, and Xiaoqing Ye. DrivingDiffusion: Layout-guided multi-view driving scenarios video generation with latent diffusion model. In *European Conference on Computer Vision*, pp. 469–485, 2024a. doi: 10.1007/978-3-031-73229-4_27.

Xinyang Li, Zhangyu Lai, Linning Xu, Yansong Qu, Liujuan Cao, Shengchuan Zhang, Bo Dai, and Rongrong Ji. Director3D: Real-world camera trajectory and 3D scene generation from text. In *Advances in Neural Information Processing Systems*, volume 37, pp. 75125–75151. Curran Associates, Inc., 2024b. doi: 10.52202/079017-2391.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *International Conference on Learning Representations*, 2023.

Hao Lu, Zhuang Ma, Guangfeng Jiang, Wenhang Ge, Bohan Li, Yuzhan Cai, Wenzhao Zheng, Yunpeng Zhang, and Yingcong Chen. 4D driving scene generation with stereo forcing, 2025a.

Yifan Lu, Xuanchi Ren, Jiawei Yang, Tianchang Shen, Zhangjie Wu, Jun Gao, Yue Wang, Siheng Chen, Mike Chen, Sanja Fidler, and Jiahui Huang. InfiniCube: Unbounded and controllable dynamic 3D driving scene generation with world-guided video models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 27272–27283, October 2025b.

Jiageng Mao, Boyi Li, Boris Ivanovic, Yuxiao Chen, Yan Wang, Yurong You, Chaowei Xiao, Danfei Xu, Marco Pavone, and Yue Wang. DreamDrive: Generative 4D scene modeling from street view images. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 367–374. IEEE, May 2025. doi: 10.1109/ICRA55743.2025.11127463.

Chaojun Ni, Guosheng Zhao, Xiaofeng Wang, Zheng Zhu, Wenkang Qin, Guan Huang, Chen Liu, Yuyin Chen, Yida Wang, Xueyang Zhang, Yifei Zhan, Kun Zhan, Peng Jia, Xianpeng Lang, Xingang Wang, and Wenjun Mei. ReconDreamer: Crafting world models for driving scene reconstruction via online restoration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1559–1569, June 2025.

NVIDIA. Cosmos 3: Omnimodal world models for physical AI. *arXiv preprint arXiv:2606.02800*, 2026.

OpenDWM. OpenDWM: Open driving world models, 2025. URL `https://github.com/SenseTime-FVG/OpenDWM`.

Julian Ost, Andrea Ramazzina, Amogh Joshi, Maximilian Bömer, Mario Bijelic, and Felix Heide. LSD-3D: Large-scale 3D driving scene generation with geometry grounding. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(10):8177–8187, 2026. doi: 10.1609/aaai.v40i10.37765.

Panwang Pan, Chenguo Lin, Chenxin Li, Jingjing Zhao, Yuchen Lin, Haopeng Li, Yunlong Lin, Kairun Wen, Yixuan Yuan, and Yadong Mu. Diff4Splat: Repurposing video diffusion models for dynamic scene generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4232–4244, June 2026.

Sung-Yeon Park, Adam Lee, Juanwu Lu, Can Cui, Luyang Jiang, Rohit Gupta, Kyungtae Han, Ahmadreza Moradipari, and Ziran Wang. SIMSplat: Language-aligned 4D gaussian splatting for driving scenario generation, 2026.

Xuanchi Ren, Yifan Lu, Tianshi Cao, Ruiyuan Gao, Shengyu Huang, Amirmojtaba Sabour, Tianchang Shen, Tobias Pfaff, Jay Zhangjie Wu, Runjian Chen, Seung Wook Kim, Jun Gao, Laura Leal-Taixe, Mike Chen, Sanja Fidler, and Huan Ling. Cosmos-Drive-Dreams: Scalable synthetic driving data generation with world foundation models, 2025.

Ahan Shabanov, Peter Hedman, Ethan Weber, Zhengqin Li, Denis Rozumny, Gael Le Lan, Naina Dhingra, Lei Luo, Andrea Vedaldi, Christian Richardt, Andrea Tagliasacchi, Bo Zhu, and Numair Khan. Free-range gaussians: Non-grid-aligned generative 3D gaussian reconstruction, 2026.

Jaskirat Singh, Boyang Zheng, Zongze Wu, Richard Zhang, Eli Shechtman, and Saining Xie. Improved baselines with representation autoencoders, 2026.

Qi Song, Yifei He, Chi Zhang, Zheng Fu, Xuhe Zhao, Mengmeng Yang, Kun Jiang, Rui Huang, and Diange Yang. Envision4D: Envisioning visual futures via feed-forward 4D gaussian splatting for autonomous driving, 2026.

Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, Vijay Vasudevan, Wei Han, Jiquan Ngiam, Hang Zhao, Aleksei Timofeev, Scott Ettinger, Maxim Krivokon, Amy Gao, Aditya Joshi, Yu Zhang, Jonathon Shlens, Zhifeng Chen, and Dragomir Anguelov. 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.

Xiangyu Sun, Shijie Wang, Fengyi Zhang, Lin Liu, Caiyan Jia, Ziying Song, Zi Huang, and Yadan Luo. VGGT-World: Transforming VGGT into an autoregressive geometry world model, 2026.

Thomas Unterthiner, Sjoerd van Steenkiste, Karol Kurach, Raphaël Marinier, Marcin Michalski, and Sylvain Gelly. FVD: A new metric for video generation. In *ICLR Workshop on Deep Generative Models for Highly Structured Data*, 2019.

Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. VGGT: Visual geometry grounded transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5294–5306, June 2025.

Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jerome Revaud. DUSt3R: Geometric 3D vision made easy. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 20697–20709, June 2024.

Tai Wang, Xinge Zhu, Jiangmiao Pang, and Dahua Lin. Probabilistic and geometric depth: Detecting objects in perspective. In *Proceedings of the 5th Conference on Robot Learning*, volume 164 of *Proceedings of Machine Learning Research*, pp. 1475–1485. PMLR, 2022.

Weijie Wang, Jiagang Zhu, Zeyu Zhang, Xiaofeng Wang, Zheng Zhu, Guosheng Zhao, Chaojun Ni, Haoxiao Wang, Guan Huang, Xinze Chen, Yukun Zhou, Wenkang Qin, Duochao Shi, Haoyun Li, Yicheng Xiao, Donny Y. Chen, and Jiwen Lu. DriveGen3D: Boosting feed-forward driving scene generation with efficient video diffusion, 2026a.

Yifan Wang, Francesco Pittaluga, Zaid Tasneem, Chenyu You, Manmohan Chandraker, and Ziyu Jiang. HorizonForge: Driving scene editing with any trajectories and any vehicles. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 24895–24905, June 2026b.

Yifan Wang, Jianjun Zhou, Haoyi Zhu, Wenzheng Chang, Yang Zhou, Zizun Li, Junyi Chen, Jiangmiao Pang, Chunhua Shen, and Tong He. π^3: Permutation-equivariant visual geometry learning. In *International Conference on Learning Representations*, 2026c.

Lisa Weijler, Irene Ballester, Guofeng Mei, Tolga Birdal, and Pedro Hermosilla. Latent riemannian flow matching for geometry-grounded 3D foundation models, 2026.

Haoyu Wu, Diankun Wu, Tianyu He, Junliang Guo, Yang Ye, Yueqi Duan, and Jiang Bian. Geometry forcing: Marrying video diffusion and 3D representation for consistent world modeling. In *International Conference on Learning Representations*, 2026.

Lihe Yang, Bingyi Kang, Zilong Huang, Zhen Zhao, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything V2. In *Advances in Neural Information Processing Systems*, volume 37, pp. 21875–21911, 2024.

Yu Yang, Alan Liang, Jianbiao Mei, Yukai Ma, Yong Liu, and Gim Hee Lee. X-Scene: Large-scale driving scene generation with high fidelity and flexible controllability. In *Advances in Neural Information Processing Systems*, volume 38, pp. 115563–115599. Neural Information Processing Systems Foundation, Inc., 2025a. doi: 10.52202/085713-3487.

Yuanbo Yang, Jiahao Shao, Xinyang Li, Yujun Shen, Andreas Geiger, and Yiyi Liao. Prometheus: 3D-aware latent diffusion models for feed-forward text-to-3D scene generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2857–2869, June 2025b.

Zeyu Yang, Zijie Pan, Yuankun Yang, Xiatian Zhu, and Li Zhang. Driving view synthesis on free-form trajectories with generative prior. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 28083–28092, October 2025c.

Tianrui Zhang, Yichen Liu, Zilin Guo, Yuxin Guo, Jingcheng Ni, Chenjing Ding, Dan Xu, Lewei Lu, and Zehuan Wu. CVD-STORM: Cross-view video diffusion with spatial-temporal reconstruction model for autonomous driving, 2025.

Guosheng Zhao, Chaojun Ni, Xiaofeng Wang, Zheng Zhu, Xueyang Zhang, Yida Wang, Guan Huang, Xinze Chen, Boyuan Wang, Youyi Zhang, Wenjun Mei, and Xingang Wang. DriveDreamer4D: World models are effective data machines for 4D driving scene representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12015–12026, June 2025a.

Guosheng Zhao, Xiaofeng Wang, Zheng Zhu, Xinze Chen, Guan Huang, Xiaoyi Bao, and Xingang Wang. DriveDreamer-2: LLM-enhanced world models for diverse driving video generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 10412–10420, 2025b. doi: 10.1609/aaai.v39i10.33130.

Ziyue Zhu, Zhanqian Wu, Zhenxin Zhu, Lijun Zhou, Haiyang Sun, Bing Wang, Kun Ma, Guang Chen, Hangjun Ye, Jin Xie, and Jian Yang. WorldSplat: Gaussian-centric feed-forward 4D scene generation for autonomous driving. In *International Conference on Learning Representations*, 2026.

A Additional Method Details

We supplement the main method with target construction, conditioning, losses, and sampling.

A.1 Reconstruction Targets and Scene Tokenizer

State and tokenizer. The tokenizer in Sec. 3.2 operates on patch tokens, excluding camera and register tokens. A local convolutional branch mixes neighboring patches, while attention within frames and across time at each patch adds broader context. The decoder uses the same attention axes and learned queries to recover the four reconstruction levels through separate output heads.

Tokenizer supervision. We combine normalized MSE and cosine feature losses, weighted by 1 and 0.2, with Huber matching of DGGT's dense-head outputs at weight 0.6. Dynamic-mask BCE and a lifespan penalty use weights 0.2 and 0.01 to supervise the static–dynamic split and static persistence. The size–depth loss in Eq. 4 has weight 0.3 and uses valid static, non-sky pixels with opacity above 0.05 in both direct and reconstructed predictions. Each rendering loss, against DGGT outputs or observed RGB, has weight 0.3 and combines MSE with 0.1 LPIPS. To make decoding robust to errors in generated latents, we also reconstruct DGGT features from $\mathbf{z} + \xi\boldsymbol{\epsilon}$, with $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and $\xi \sim \mathcal{U}(0, 0.8)$. Matching ρ constrains size relative to depth independently of their common scale.

Frozen generation interface. We standardize flow latents as $\widetilde{\mathbf{z}} = (\mathbf{z} - \boldsymbol{\mu}_z)/\boldsymbol{\sigma}_z$ using fixed training-set channel statistics. We undo this transform before decoding, retaining gradients for HDS.

Sky target. The sky atlas $\mathcal{E}$ covers the upper hemisphere in the teacher's first-camera frame. Each direction takes RGB from the visible frame with highest sky confidence. We fill missing regions from neighboring observations and retain an observation mask for loss weighting. The 128×256 atlas is packed into 8×8 RGB patches and standardized directly, bypassing the scene tokenizer.

A.2 Calibration Targets and Coordinate Interfaces

Calibration targets. We estimate scale from paired DGGT and LiDAR optical-axis depths. For valid pixels V_i and frames $\mathcal{I}$ retained after rejecting outlying frame estimates,

$$r_i = \mathrm{median}_{p \in V_i} \frac{d_{i,p}^{\mathrm{teacher}}}{d_{i,p}^{\mathrm{LiDAR}}}, \qquad s = \left(\mathrm{median}_{i \in \mathcal{I}}\, r_i\right)^{-1}, \qquad \eta_s = \log s. \tag{9}$$

We use finite, positive teacher depths and LiDAR depths between 1 and $80\,\mathrm{m}$, requiring at least 64 valid pairs per frame. Frame ratios farther than $3.5 \times 1.4826\,\mathrm{MAD}$ from their median are rejected, where MAD is the median absolute deviation. Scale and FOV validity are checked separately, and invalid channels are excluded from supervision.

Reconstruction cameras. Let $\mathbf{T}_a$ be the reference camera's metric camera-to-world pose. We rebase requested poses to this frame and divide translations by $\widehat{s}$ to match reconstruction units:

$$\begin{aligned} \mathbf{T}_a^{-1}\mathbf{T}_i^{\mathrm{met}} = \begin{bmatrix} \mathbf{R}_i^a & \mathbf{t}_i^a \\ \mathbf{0}^\top & 1 \end{bmatrix}, \qquad \mathbf{T}_i^{\mathrm{rec}} = \begin{bmatrix} \mathbf{R}_i^a & \mathbf{t}_i^a/\widehat{s} \\ \mathbf{0}^\top & 1 \end{bmatrix}, \\ \Phi_{\mathrm{dec}} = \left\{(\mathbf{T}_i^{\mathrm{rec}})^{-1}, \mathbf{K}^{\mathrm{rec}}\right\}_i, \qquad \Phi_{\mathrm{phys}} = \left\{(\mathbf{T}_i^{\mathrm{rec}})^{-1}, \mathbf{K}_i^{\mathrm{phys}}\right\}_i. \end{aligned} \tag{10}$$

Native Gaussian construction and HDS use Φ_{dec}, bypassing the camera head. Let $\mathbf{c}_i = \mathbf{t}_i^a/\widehat{s}$, $\mathbf{A}_i = (\mathbf{K}_i^{\mathrm{phys}})^{-1}\mathbf{K}^{\mathrm{rec}}$, and $\mathbf{J}_i = \mathbf{R}_i^a \mathbf{A}_i (\mathbf{R}_i^a)^\top$. For predicted depth $d_{i,p}$ in reconstruction units, the centre construction in Eq. 1 and its source-camera conversion for physical export give

$$\begin{aligned} \boldsymbol{\mu}_{i,p} &= \mathbf{R}_i^a (\mathbf{K}^{\mathrm{rec}})^{-1} \bar{\mathbf{p}}\, d_{i,p} + \mathbf{c}_i, \\ \boldsymbol{\mu}'_{i,p} &= \mathbf{J}_i(\boldsymbol{\mu}_{i,p} - \mathbf{c}_i) + \mathbf{c}_i = \mathbf{R}_i^a (\mathbf{K}_i^{\mathrm{phys}})^{-1} \bar{\mathbf{p}}\, d_{i,p} + \mathbf{c}_i. \end{aligned} \tag{11}$$

Thus centre conversion is equivalent to physical-camera backprojection of the same predicted depth. Covariances transform as $\boldsymbol{\Sigma}'_{i,p} = \mathbf{J}_i \boldsymbol{\Sigma}_{i,p} \mathbf{J}_i^\top$. Colour, opacity and dynamic attributes remain unchanged. We retain generated scale and temporal fusion, fix each Gaussian's source camera, and render the exported scene under Φ_{phys} without LiDAR fitting, reusing it across camera offsets.

Table 7: Shared-attention mRoPE positions on an $H_p \times W_p$ patch grid. Here $r_i = 24 f_i/\text{fps}$ for frame index f_i, and (y, x) are patch indices. Sky coordinates use unit directions $\mathbf{d}_j$ and $\mathbf{1} = (1, 1, 1)$.

Token / feature	mRoPE coordinates
Scene patch; map / actor features	(r_i, y, x)
Camera	$(r_i, \lfloor H_p/2 \rfloor, \lfloor W_p/2 \rfloor)$ (grid centre)
Sky atlas patch j	$15000\,\mathbf{1} + 8\mathbf{d}_j$
Calibration	$(15100, 15100, 15100)$
Text / noise-time	$(0, 0, 0)$ (no rotary phase)

Layout encoding. Both intrinsic matrices use the same normalized image canvas and have last row $(0, 0, 1)$. For a metric camera-frame layout point $\mathbf{x}_i$ with positive depth, $\bar{\mathbf{p}}_i = (u_i, v_i, 1)^\top = \mathbf{K}_i^{\text{phys}} \mathbf{x}_i / x_{i,z}$. With $\widehat{s} > 0$ and nonsingular intrinsics, Eq. 6 gives

$$\mathbf{K}^{\text{rec}} \mathbf{q}_i = \mathbf{K}_i^{\text{phys}} \mathbf{x}_i / \widehat{s}, \qquad q_{i,z} = x_{i,z}/\widehat{s} > 0, \qquad \mathbf{A}_i \mathbf{q}_i = \mathbf{x}_i / \widehat{s}. \tag{12}$$

Perspective division removes $\widehat{s}$, so $\pi(\mathbf{K}^{\text{rec}} \mathbf{q}_i) = \pi(\mathbf{K}_i^{\text{phys}} \mathbf{x}_i)$: the condition preserves the physical-camera pixel and converts its axial depth to reconstruction units. Decoder intrinsics have zero skew, principal point $(1/2, 1/2)$, and focal lengths $f_q = [2 \tan(\widehat{\phi}_q/2)]^{-1}$ for $q \in \{x, y\}$. Physical intrinsics retain the supplied principal point on the same canvas. The identity $\mathbf{A}_i \mathbf{q}_i = \mathbf{x}_i / \widehat{s}$ reverses the condition's intrinsic remapping in its source camera. These coordinates provide per-view adapter conditions. Gaussian centres are constructed separately from predicted depths as in Eq. 11.

A.3 Joint Generator and Structured Conditioning

Shared attention gives calibration decoding access to the scene state, and the decoded calibration supplies ray and depth cues to the scene head within the same flow step.

Stage 1: Image-space conditioning. The shared transformer has 28 blocks with 1440 channels and 20 attention heads. We encode map and actor rasters with separate convolutional stems, adding their features to scene patches. Three-axis mRoPE aligns scene and condition tokens (Tab. 7).

Stage 2: Metric conditioning. After shared attention, a lightweight decoder predicts clean $\widehat{\gamma}$. For the physical-camera pixel (u_i, v_i) and axial depth $x_{i,z}$ in Eq. 6, the map and actor features are

$$\psi_i = \begin{bmatrix} q_{i,x}/q_{i,z} \\ q_{i,y}/q_{i,z} \\ \log q_{i,z} \end{bmatrix} = \begin{bmatrix} 2\tan(\widehat{\phi}_x/2)(u_i - \frac{1}{2}) \\ 2\tan(\widehat{\phi}_y/2)(v_i - \frac{1}{2}) \\ \log x_{i,z} - \log \widehat{s} \end{bmatrix}. \tag{13}$$

Since $\tan(\widehat{\phi}_q/2) = e^{\widehat{\eta}_q}$ and $\log \widehat{s} = \widehat{\eta}_s$, the first two entries depend on decoder FOVs, while scale converts depth. The map and actor adapters encode these features.

DDT head. The DDT head has two blocks with 2048 channels, 16 attention heads, and SwiGLU feed-forward layers with 5461 intermediate channels. Each block uses RMSNorm with condition-dependent shifts, scales, and residual gates. The modulation and final output projections are zero-initialized. The two blocks contain 151.1M parameters.

A.4 Training Objectives

We give the loss weights and comparison spaces that supplement Eqs. 5–8.

Joint flow. With $t_0 = 0.05$, Eq. 5 weights squared clean-prediction errors by δ_t^{-2} in standardized coordinates. The implementation computes the same loss by comparing $(\mathbf{y}_t - \widehat{\mathbf{y}}_0)/\delta_t$ with $(\mathbf{y}_t - \mathbf{y})/\delta_t$. Training can sample $t < t_0$, where this weighting factor is capped at 400 and the clean target remains $\mathbf{y}$. We aggregate these errors across the three output groups as

$$\mathcal{L}_{\text{RF}} = D_z + 0.5 D_{\mathcal{E}} + 0.1 D_\gamma. \tag{14}$$

Here D_z averages over scene tokens and D_γ over valid calibration channels. $D_{\mathcal{E}}$ separately averages observed and completed sky regions with weights 1 and 0.05, giving completed values weaker supervision independently of region size. Samples without observed sky contribute zero to $D_{\mathcal{E}}$.

Decoded supervision. For d_{feat}, mean absolute error uses LayerNorm across token channels, while cosine distance uses the original tokens. We sum these distances to compare normalized values and token directions, averaging over the four reconstruction levels. For d_{geo}, Smooth-ℓ_1 compares log depth, Gaussian RGB, opacity and log scales, sigmoid dynamic probabilities, and confidences in $\log(1+c)$ space. Rotation uses $1-|\widehat{q}^{\top}q^{\star}|$ for unit quaternions, treating opposite signs equally. The four Gaussian attribute groups are averaged, with confidence weighting as defined in Sec. 3.4.

Requested-view images and gradients. Image supervision covers the full canvas, combining confidence-weighted Charbonnier with spatial LPIPS at coefficient 0.01. Feature and geometry losses use weight $(1-t)^2$, multiplied by a cubic smoothstep gate that decreases from one at $t=0.3$ to zero at $t=0.7$. Following Free-Range Gaussians (Shabanov et al., 2026), the Charbonnier weight is $\min(1,(1-t)/0.9)^5$. LPIPS uses the same weight but is disabled for $t \geq 0.7$. All outer HDS coefficients are 1. We fix Φ_{dec} and stop image-loss gradients through the predicted sky mask. Stage 2 gradients to calibration use a multiplier of zero for 5k training steps, rising linearly to one at 15k.

Auxiliary supervision. Direct calibration uses valid-channel Smooth-ℓ_1 with weight 1 in the unstandardized log coordinates of App. A.2. BCE and Dice supervise patch-level and refined sky masks with weights 0.05 and 0.1, respectively. Photometric and perceptual sky-view losses compare the reprojected atlas with observed sky RGB, with overall weight 1. This path bypasses Gaussian rendering but retains predicted-FOV gradients, complementing the fixed-camera HDS image loss.

A.5 Training and Sampling

Conditional sampling. Starting from $\mathbf{y}^{(0)} \sim \mathcal{N}(\mathbf{0},\mathbf{I})$, we take $N=50$ Euler steps with time shift $a=10$. Set $u_k = 1-k/N$ and $t_k = au_k/[1+(a-1)u_k]$ for $k=0,\dots,N$. At each $k<N$, we obtain the guided clean prediction $\widehat{\mathbf{y}}_{0,k}$ and update

$$\widehat{\mathbf{v}}_k = (\mathbf{y}^{(k)} - \widehat{\mathbf{y}}_{0,k})/t_k, \qquad \mathbf{y}^{(k+1)} = \mathbf{y}^{(k)} + (t_{k+1}-t_k)\widehat{\mathbf{v}}_k.$$

The smallest model-call time is $t_{N-1} = 10/59 \approx 0.1695 > t_0$, so $\delta_{t_k} = t_k$ at every call and the denominator floor is inactive during sampling. The endpoint $t_N = 0$ requires no model evaluation, and the final Euler step returns the guided clean prediction.

For classifier-free guidance (CFG), we use $\mathrm{TC} = (T, \mathbf{C})$ and $\mathrm{TCMG} = (T, \mathbf{C}, M, G)$. Text CFG removes only T from the TCMG branch, while layout CFG compares TCMG with TC. All branches retain $\mathbf{C}$. Layout guidance affects only the scene latent; sky, calibration, and sky masks use text guidance with layout conditions retained.

Sliding-window training and inference. Training allocates 75% of updates to independent windows and 25% to crops in the parent reconstruction's coordinates, retaining its frame indices and calibration targets. Inference maintains one full-sequence latent and shared calibration and sky states. At each denoising step, we blend overlapping-window velocities and update all states together.

Decoding and rendering. The sky atlas supplies background colour through the remaining transmittance. New camera views reuse the decoded Gaussian scene and sky atlas without resampling.

B Additional Experimental Details

We supplement the main experiments with evaluation details and inference costs, a direct comparison with video generators, tokenizer fidelity, and additional qualitative results.

B.1 Evaluation Details

Clips and baselines. For each Waymo validation scene, we take the first 10 front-view frames from each of two fixed, temporally separated segments at 10 Hz. For fairness, failed outputs from all methods remain in evaluation. Outputs share timestamps and a 518×350 front-view canvas. OpenDWM generates multiple views; we score the front camera. Cosmos3-Nano uses map and actor-box controls rendered with Cosmos-Drive-Dreams (Ren et al., 2025). DGGT direct uses observed RGB; the tokenizer reference additionally compresses and reconstructs the DGGT state.

Scene captions. For evaluation, MiniMax-M3 generates captions offline from all ten chronologically ordered front-view RGB frames of each reference clip. The prompt requests an 80–110-word

English paragraph describing roads, surroundings, visible actors, and lighting/weather, mentioning motion only when clear and omitting uncertain details. All generation methods receive the same frozen per-clip caption through their native text-conditioning interfaces, with Cosmos3 prompt upsampling disabled. Generators receive these reference-derived captions and the specified controls, without receiving the reference RGB frames.

Baseline availability. At the submission deadline, WorldSplat provides nuScenes training/inference code, but its stage checkpoints remain listed as "Coming soon." ScenDi's project page provides neither model code nor checkpoints. CVD-STORM links OpenDWM's released video models, but its Gaussian-decoder weights and inference configuration are not listed.

Metric RGB–3D consistency. Following CVD-STORM (Zhang et al., 2025) and WorldSplat (Zhu et al., 2026), we define metric RGB–3D consistency as agreement between rendered Gaussian optical-axis depth and frozen DA2 Metric VKITTI-Large pseudo-depth inferred from the same RGB, with both expressed in metres. AbsRel and δ_1 use valid Gaussian–DA2 pairs; coverage is their fraction of the non-sky, valid pseudo-depth domain. DriveReGen multiplies depth by its generated scale, learned from training LiDAR calibration targets; InfiniCube outputs native metres. Video-to-DGGT baselines and reconstruction references use the scale ratio between reconstructed and requested camera trajectories, retained throughout evaluation. These scores compare complete pipelines, including generation and scale estimation.

Layout control. (1) For *HD-map control*, we score PersFormer against Waymo road lines/edges, not centerlines. Hungarian matching uses distance/coverage thresholds of 1.5 m and 0.75. F1/recall pool all-frame counts, including empty-GT false positives. X-error averages 0–40 m matched errors $\leq$ 5 m. Waymo maps lack image-specific visibility and have denser boundaries; F1 is low even on real images (16.14%), so absolute F1 depends on the evaluation protocol. (2) For *actor control*, we score PGD detections within 30 m, since more distant actors occupy too few pixels for reliable position and pose evaluation. G-LET-mAP averages raw-score LET-3D-AP over vehicles, pedestrians, and cyclists. InfiniCube reports vehicle-only metrics as its official pipeline supports only vehicle control. Within 30 m, the full three-class evaluation contains $10{,}781$ ground-truth actor boxes; the vehicle-only subset contains $7{,}387$. Recall and pose errors follow the same PGD matching protocol for all methods. All methods share preprocessing and retain empty layouts for false-positive evaluation.

Camera offsets. For Tab. 4, we reuse the full physical-K export defined in App. A.2. Scenes stay fixed across offsets; PersFormer uses each method's rendering K. Relative poses and first-camera alignment map predictions into the source ground frame. We average left/right scores; ΔFID subtracts each method's zero-offset FID, using the same original-view real images.

Training ablations. *Fixed units* uses training-set scale and decoder-FOV constants throughout training and inference. *Fixed S2 γ* uses these constants only for Stage 2 layout encoding during training and sampling, retaining Full's 3D inputs and adapters. Calibration prediction and supervision remain active, and decoding uses generated calibration. All other training and evaluation settings match Full. Full and *Fixed S2 γ* retain the same joint generator; their difference is whether Stage 2 sees the scene-specific calibration used in decoding. *w/o HDS* removes all three terms in Eq. 7. The HDS variants add supervision cumulatively: *HDS: RGB* uses only the requested-view image term, *HDS: Geo.* adds geometry supervision to the RGB loss, and *HDS: Full* also adds feature supervision. All four settings keep the flow and auxiliary objectives, including direct calibration and sky supervision. Retained HDS terms use Full's loss weights and gradient paths, with the same training updates and evaluation settings on the shared 404-clip benchmark.

Calibration ablations. Tabs. 5 and 8 hold the model and sampled scene latents fixed. *Fixed scale* uses the training-set scale $s_0 = 28.136$ metres per reconstruction unit with generated FOVs. *Fixed FOV* uses 36.13° horizontally and 25.85° vertically, from training-set means of $\log\tan(\phi/2)$ with equal scene weights, retaining the principal point. In Tab. 5, *Physical K for decoding* rebuilds Gaussian centres and rerenders the scene and sky with requested intrinsics and generated scale. In Tab. 8, Fixed FOV constructs centres and renders with its constants. The physical-K export rows convert centres alone or centres and covariances in their source cameras, retaining generated scale and temporal fusion. PersFormer and PGD use requested physical K throughout.

For reference calibration, we measure log-scale and decoder FOV errors against references from real-video DGGT reconstruction: scale is calibrated with LiDAR, and FOV comes from DGGT's

Table 8: Calibration and physical-K export for requested-view layout control. Native uses generated FOVs; Fixed FOV uses training-set constants. Both physical-K exports retain generated scale and temporal fusion. Converting centres and covariances improves lane and actor detection and reduces lateral error relative to Native in the requested camera view.

Setting	HD map (lane detection)			Track (actor detection)			
	F1 ↑	R ↑	X-Err ↓	mAP↑	R↑	ATE ↓	AOE ↓
Native	12.08	7.22	0.713	0.1202	0.3448	**0.7680**	0.3321
Fixed FOV	11.21	6.63	0.720	0.1077	0.3130	0.7803	0.3403
Centres only	**12.55**	**7.54**	0.716	0.0976	0.2888	0.7904	**0.3183**
Centres + covariances	12.43	7.44	**0.707**	**0.1251**	**0.3582**	0.7682	0.3389

Table 9: Visual quality against video generators. Method labels indicate videos before DGGT reconstruction (video) or Gaussian renderings (3D). DriveReGen achieves the lowest FID and FVD while retaining text alignment, showing its visual-quality advantage holds against the video outputs.

Method	FID↓	FVD↓	CLIP-T↑
InfiniCube (3D)	59.74	621.18	30.76
OpenDWM (video)	<u>44.74</u>	<u>369.32</u>	**31.38**
Cosmos3 (video)	60.92	725.90	30.41
DriveReGen (Ours, 3D)	**37.15**	**260.77**	<u>31.13</u>

camera head. For RGB–depth consistency, we compare each setting's Gaussian depth with DA2 predictions from its own rendered RGB while retaining that setting's scale.

Inference cost. On a single NVIDIA A800, DriveReGen's full 10-frame pipeline averages 142.10 s with 19.12 GiB peak GPU memory, versus 419.66 s and 38.42 GiB for InfiniCube Vehicle v1's native Gaussian generation. DriveReGen is $2.95\times$ faster with 50.2% lower peak memory.

InfiniCube uses three sequential stages (mean time / peak memory: voxel diffusion, 189.88 s / 21.02 GiB; guidance and Wan-14B video generation, 209.86 s / 38.42 GiB; Gaussian reconstruction, 19.92 s / 11.55 GiB). Pipeline peak memory is the maximum across stages, reached during Wan generation. Rendering the center and offset trajectories adds 29.94 s with 1.89 GiB peak memory, bringing total time to 449.60 s per clip.

B.2 Calibration Ablations and Requested-View Placement

Generated FOVs improve layout alignment over fixed FOVs (Tab. 8). Full physical-K export further improves lane F1/recall and actor mAP/recall over Native, while reducing X-Err from 0.713 to 0.707 m; ATE is nearly unchanged and AOE slightly higher. Adding covariance conversion raises actor mAP from 0.0976 to 0.1251 and recall from 0.2888 to 0.3582.

B.3 LiDAR Depth Evaluation Protocol

For Tab. 3, LiDAR is projected with physical intrinsics and axial depth is evaluated on layout-controlled road and actor pixels. Both decoder-K rows retain $\mathbf{K}^{\mathrm{rec}}$. Fixed scale rescales camera translations by $\widehat{s}/s_0$ and multiplies rerendered depth by s_0. Physical-K export follows App. A.2, dividing rebased metric translations by $\widehat{s}$ and multiplying rendered depth by $\widehat{s}$. All methods share reference pixels selected independently of predictions and the same validity rule: finite positive depth and finite rendered alpha $\geq 10^{-4}$. AbsRel/RMSE use each method's valid predictions; δ_1 uses all reference pixels and counts invalid predictions as failures. Empty reference support gives N/A. With nonempty support but no valid prediction, AbsRel/RMSE are N/A and $\delta_1 = 0$. Available values are averaged over frames, then clips and scenes.

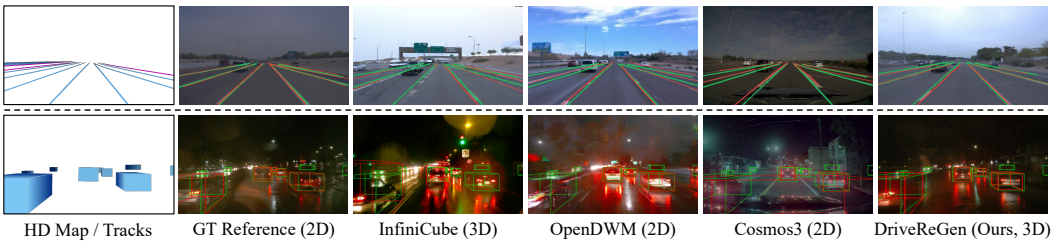


Figure 5: HD-map and track control. Top: lane detections; bottom: actor detections. Green marks projected Waymo ground truth; red marks predictions. OpenDWM and Cosmos3 use original videos (2D), whose layout scores exceed those after DGGT reconstruction (Tabs. 10 and 2). DriveReGen retains comparable alignment to OpenDWM while producing a renderable 3D scene.

Table 10: Layout control against original video outputs. Video rows precede DGGT reconstruction; 3D rows use Gaussian renderings. †Vehicle-only actor metrics, excluded from actor rankings. ‡Real-image reference. DriveReGen has slightly higher lane F1 and actor mAP than OpenDWM, while also producing Gaussians, depth, and dynamic maps.

Method	HD map (lane detection)			Track (actor detection)			
	F1↑	R↑	X-Err↓	mAP↑	R↑	ATE↓	AOE↓
InfiniCube (3D)†	8.66	5.05	0.808	0.0945	0.1822	0.8179	0.3472
OpenDWM (video)	<u>11.37</u>	<u>6.81</u>	<u>0.732</u>	<u>0.1144</u>	<u>0.3329</u>	<u>0.7877</u>	<u>0.3458</u>
Cosmos3 (video)	10.85	6.27	0.785	0.0656	0.1955	0.8553	0.3669
DriveReGen (Ours, 3D)	**12.43**	**7.44**	**0.707**	**0.1251**	**0.3582**	**0.7682**	**0.3389**
Real images‡	16.14	9.80	0.702	0.1957	0.4616	0.7310	0.1689

Table 11: Vehicle-only actor detection under the InfiniCube-aligned protocol, which omits pedestrians and cyclists. OpenDWM and Cosmos3 use original videos; InfiniCube and DriveReGen use Gaussian renderings. DriveReGen has slightly better detection and pose scores than OpenDWM while additionally providing an explicit 3D scene.

Method	mAP↑	R↑	ATE (m)↓	AOE (rad)↓
InfiniCube (3D)	0.0945	0.1822	0.8179	0.3472
OpenDWM (video)	<u>0.3346</u>	<u>0.4526</u>	<u>0.7525</u>	<u>0.2578</u>
Cosmos3 (video)	0.1880	0.2686	0.8386	0.3200
DriveReGen (Ours, 3D)	**0.3504**	**0.4846**	**0.7415**	**0.2523**

B.4 Comparison with Video Generators

Visual quality. We compare our Gaussian renderings with the original OpenDWM and Cosmos3 videos before DGGT reconstruction. Tab. 9 uses the same 404 clips, timestamps, and quality metrics as Tab. 1. DriveReGen achieves the lowest FID and FVD: compared with OpenDWM, FID decreases from 44.74 to 37.15 and FVD from 369.32 to 260.77. CLIP-T is similar (31.13 vs. 31.38).

Layout control. Tab. 10 compares layout control using the protocol of Tab. 2. Layout scores are close to OpenDWM's, with small improvements: lane F1 is 12.43% versus 11.37%, and actor mAP is 0.1251 versus 0.1144. ATE measures position errors on matched detections: 0.7682 m for DriveReGen versus 0.7877 m for OpenDWM. Compared with these original videos, DriveReGen also provides renderable 3D Gaussians, depth maps, and dynamic maps without reconstructing generated video, enabling re-rendering of the same scene from shifted cameras. Under the InfiniCube-aligned vehicle-only protocol (Tab. 11), our mAP/recall is $0.3504/0.4846$ versus InfiniCube's $0.0945/0.1822$. The pooled pedestrian–cyclist AOE is 1.117 rad for Cosmos3, compared with 1.438 for DriveReGen and 1.542 for OpenDWM.



(a) Size–depth ratio residuals (b) Local round-trip diagnostics

Figure 6: Size–depth preservation after tokenizer compression. **(a)** Scene-balanced residuals concentrate near zero. **(b)** Median and P90 are medium-error and high-error examples, near the 50th and 90th percentiles of scene mean $|r|$. Renderings remain similar; the third column maps local ratio residuals. Positive residuals mean that round-trip splats are larger relative to depth than Direct DGGT; negative residuals mean that they are smaller. Grey marks invalid support.

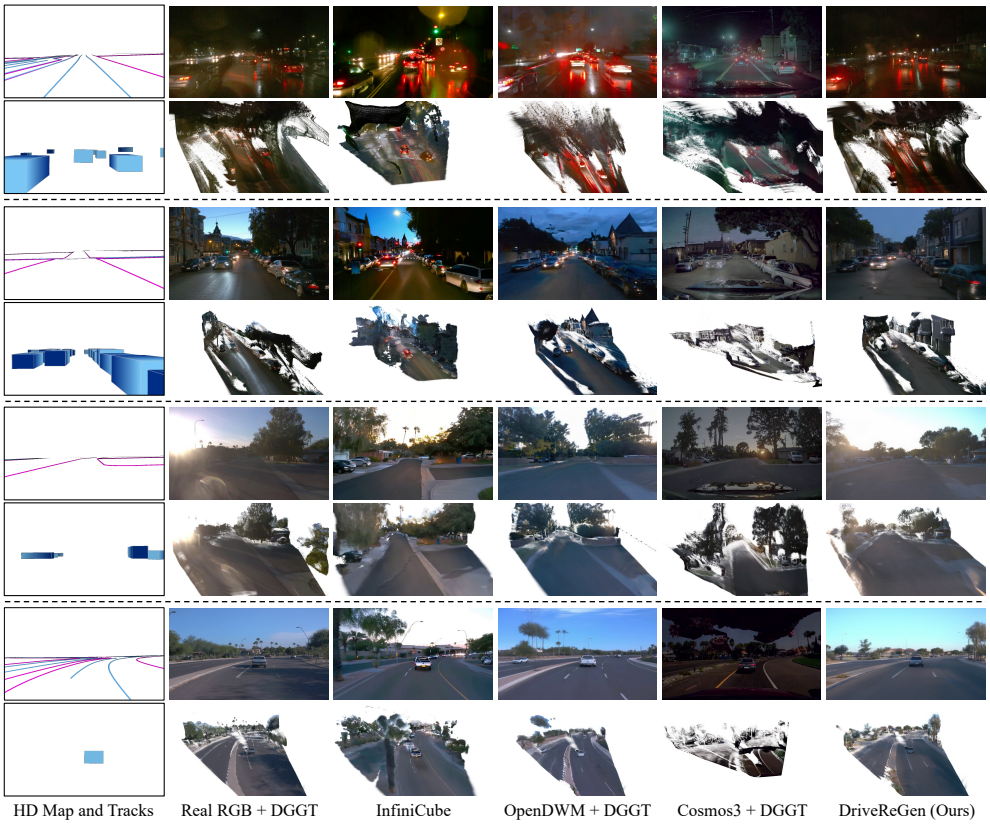


Figure 7: Additional qualitative comparisons. Each scene shows RGB results above and Gaussian renderings below. Gaussians below real RGB are reconstructed by DGGT and shown only as visual references. DriveReGen preserves visual quality and geometry even on a rainy night, while maintaining requested vehicle headings, prompt-aligned backlighting, and plausible vehicle sizes.

B.5 Tokenizer Fidelity

We assess size–depth preservation using $r = \log(\rho_{\mathrm{RT}}/\rho_{\mathrm{Direct}})$, where ρ is the geometric mean of Gaussian axis scales divided by depth. We use 100 scenes from the tokenizer's validation partition, one 12-frame clip per scene, with shared frozen heads and cameras and no scale fitting.

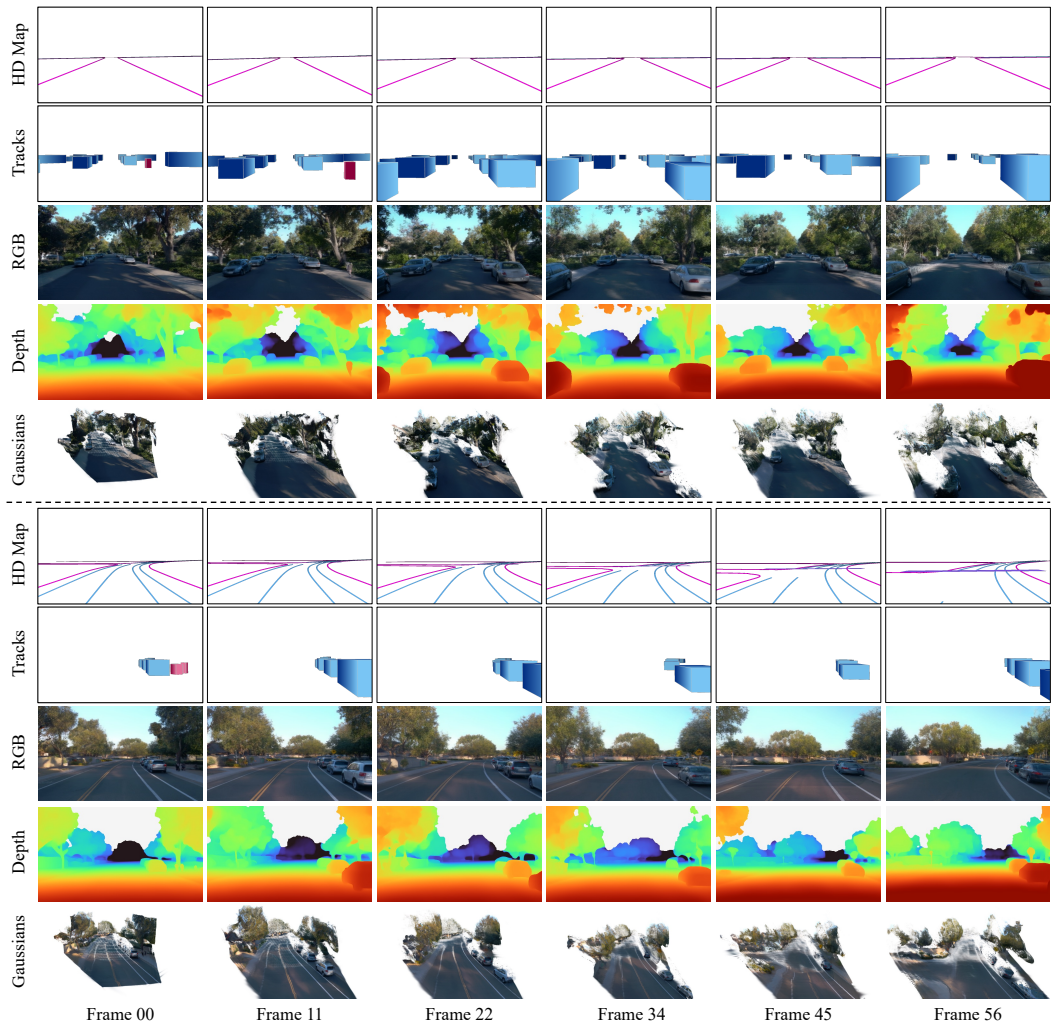


Figure 8: Long-video inference with sliding windows. Each example spans 57 frames; rows contain the HD map, actor tracks, RGB renderings, predicted depth maps, and Gaussian scene renderings at matching timestamps. DriveReGen maintains coherent road geometry and vehicle appearance across the 57-frame sequences, extending generation beyond its 10-frame training window.

We compare tokenizers trained without and with size–depth consistency and depth-bias supervision. The paired ratio increases from 0.7964 to 0.99985 (ideal: 1), using exponentiated hierarchical medians of r. This comparison evaluates the joint effect of size–depth consistency and depth-bias supervision. Tab. 1 shows high rendering quality and similar RGB–3D consistency after the tokenizer round trip; we further inspect its size–depth residuals and local renderings in Fig. 6. Panel (a) shows close size–depth agreement with Direct DGGT (mean $r = -0.0008$; mean $|r| = 0.0389$). The selected crops also retain similar RGB appearance at both Median and P90.

B.6 Additional Qualitative Comparisons

Fig. 7 presents four additional visual comparisons. In the first, DriveReGen retains visual quality and coherent geometry even in nighttime rain. In the second scene, InfiniCube produces incorrect vehicle headings, while OpenDWM + DGGT shows blurred vehicle details. DriveReGen follows the requested headings. Its second stage of layout control explicitly encodes the eight ordered corners of each actor's 3D box, providing orientation and size cues to the denoising decoder. In the third scene, DriveReGen reproduces the prompt-specified backlighting. In the fourth scene, our vehicles have plausible sizes and fit naturally into the scene, while Cosmos3 generates a corrupted video.

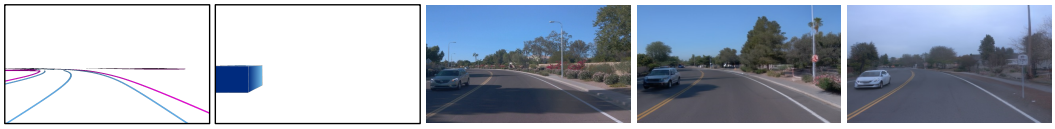


Figure 9: Rare-weather failure. From left to right: HD map, actor tracks, real RGB, and generations with the sunny and snow prompts. The snow prompt produces overcast lighting but no visible snow cover on the road or surrounding vegetation.

B.7 Long-Video Inference

We extend inference beyond the 10-frame training window using overlapping 10-frame windows with shared calibration and sky (App. A.5).

Fig. 8 shows 57-frame tree-lined street and curved-road sequences, with RGB, depth and Gaussian views at multiple timestamps. On a single NVIDIA A800, 57-frame inference averages $1258\,\mathrm{s}$, with peak GPU memory similar to that of 29-frame inference.

We compare DriveReGen with InfiniCube on the same 404 clips, extended to 29 frames, using the metrics in Tab. 1. DriveReGen achieves FID/FVD of $39.96/242.44$ versus $57.58/684.07$ and CLIP-T of 30.99 versus 30.85. For metric RGB–3D consistency, AbsRel is 0.1765 versus 0.2502, δ_1 is 0.7277 versus 0.6307, and coverage is 99.996% versus 99.596%. On an NVIDIA A800, 29-frame inference averages $583.36\,\mathrm{s}$, with an observed peak GPU memory of $22.73\,\mathrm{GiB}$.

C Limitations and Failure Cases

Domain transfer. DriveReGen uses a DGGT checkpoint trained on Waymo. Training our method on other datasets relies on DGGT to provide reliable reconstruction targets. For example, Cosmos3 produces high-quality videos, but their wide-angle dashcam views show the vehicle hood and differ from Waymo's front-camera views. Under this distribution shift, DGGT's reconstructions show geometric distortions (Fig. 4). Using DriveReGen on substantially different datasets or camera setups may therefore require fine-tuning the DGGT reconstruction backbone.

Temporal extent. DGGT is trained with 4–8 input frames per iteration, although it supports longer sequences (Chen et al., 2026). Generating much longer sequences requires this reconstruction prior to generalize beyond its training window. We use 10-frame flow training with sliding-window augmentation to balance reconstruction quality and cost, sharing calibration across windows at inference (App. A.5). Our 57-frame examples demonstrate this extension (Fig. 8); longer temporal contexts in reconstruction pretraining are a direction for further scaling.

Failure case. Our flow model is trained from scratch on driving data, with limited generalization to rare or unseen weather. For example, even when the prompt explicitly requests snowfall and white snow cover, the model produces an overcast scene without snow-covered roads or surroundings (Fig. 9). Broader weather coverage during training could improve generalization.

> *Original scene prompt (sunny weather):* This front-camera perspective shows a suburban road during the daytime under clear blue skies. The road is a two-lane street with a white dashed line dividing the lanes, bordered by sidewalks and landscaped areas with shrubs and a few scattered flowers. Streetlights line the road, and a few palm trees are scattered along the sides. A silver car is seen driving ahead, and another vehicle approaches from the opposite direction. The surroundings suggest a peaceful neighborhood with houses partially visible behind the landscaping. The overall atmosphere is calm and sunny.
>
> *Edited scene prompt (snowy weather):* This front-camera perspective shows a suburban road during a snowy winter daytime beneath an overcast white sky with steady white snowfall. The road is a two-lane street with a white dashed line dividing the lanes, bordered by sidewalks and landscaped areas with shrubs and a few scattered flowers; the road, sidewalks, and landscaping are blanketed in fresh

white snow. Streetlights line the road, and a few snow-covered palm trees are scattered along the sides. A silver car is seen driving ahead, and another vehicle approaches from the opposite direction. The surroundings suggest a peaceful snow-covered neighborhood with houses partially visible behind the snow-covered landscaping. The overall atmosphere is cold, wintry, white, and snowy.