Effective Multi-sensor Conditioning for Street-view Novel-view Synthesis

Zhengfei Kuang[1] **Adam Sun**[1] **Liyuan Zhu**[1] **Tong Wu**[1]
Shengqu Cai[1] **Jonathan Tremblay**[2] **Iro Armeni**[1] **Ehsan Adeli**[1]
Lior Yariv[1] **Gordon Wetzstein**[1]

[1]Stanford Univerity [2]NVIDIA

Abstract

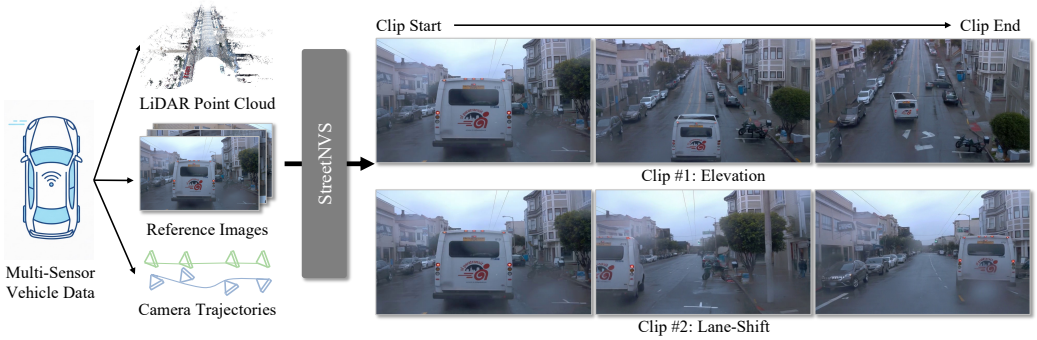
Modern vehicle platforms are equipped with a rich sensor suite, including LiDAR, calibrated multi-camera rigs, and accurate ego-motion, that in principle offers strong signal for re-rendering a driving scene from novel viewpoints. A growing line of recent work leverages video diffusion models for this task, using their generative priors to synthesize plausible novel views from sparse vehicle observations. In practice, however, existing methods exploit only a fragment of this signal, and their quality tends to degrade as the target trajectory departs from the recorded driving path. We argue that this is fundamentally a *multi-sensor fusion problem*: sparse LiDAR reprojections supply accurate but incomplete metric geometry, surround-view reference imagery supplies dense appearance but no metric depth, and camera poses tie the two together across views. We introduce **StreetNVS**, a video diffusion framework that jointly conditions on all three signals through a Reference-Enhanced Camera Attention module based on a relative ray-level positional encoding. We develop a two-stage curriculum training strategy that gradually exposes the model to increasingly sparse LiDAR. On the Waymo Open Dataset, **StreetNVS** substantially outperforms state-of-the-art baselines under sparse LiDAR conditioning, matches methods that rely on 10–100$\times$ denser point clouds. We further show capabilities of synthesizing coherent videos along extreme out-of-trajectory paths such as elevation, lane-shift, pullback, and rotation. Our website: https://streetnvs.github.io

1 Introduction

Reconstructing and digitally re-experiencing a driving scene from novel viewpoints is a critical capability with growing practical importance across automotive and transportation domains. High-quality street-view novel-view synthesis (NVS) underpins a wide range of downstream applications: rendering immersive user views for incident review and navigation, generating diverse training data for autonomous driving agents and world modeling, supporting driver-assistance features such as blind-spot visualization and parking guidance, and enabling realistic closed-loop simulation for safety evaluation. Unlike traditional 3D reconstruction settings, which typically operate offline on densely captured imagery with methods such as 3D Gaussian Splatting (13) or other neural rendering methods (32), vehicle-based streetscape reconstruction is fundamentally challenging: the scene is highly dynamic and source cameras are mounted directly on the vehicle, providing only sparse and biased coverage of the surrounding environment. As a result, target viewpoints often fall well outside the envelope of observed views, demanding strong extrapolation capabilities rather than interpolation for NVS.

To address these challenges, a growing body of recent work has turned to video diffusion models (34; 11; 46), which provide strong generative priors over natural driving scenes. Among these, methods

40th Conference on Neural Information Processing Systems (NeurIPS 2026).



(a) Multi-Sensor Conditioning (b) Novel View Synthesis

Figure 1: **StreetNVS for Street-View Novel-View Synthesis. (a)** Given multi-sensor data from a vehicle rig (LiDAR point clouds, reference images, and camera trajectories), **StreetNVS** synthesizes street-view videos along arbitrary novel trajectories. **(b)** Two NVS examples: an elevation trajectory lifting the camera toward a bird's-eye view (top), and a lane-shift trajectory displacing it from the original driving path (bottom). By jointly leveraging LiDAR geometry and cross-camera references, **StreetNVS** produces high-fidelity views far from the vehicle's own observations.

such as FreeVS (36), Gen3C (27), and StreetCrafter (48) leverage dense LiDAR point clouds as a powerful geometric anchor: they reproject the LiDAR into the target novel view and feed the resulting buffer directly to the diffusion model as a conditioning signal, effectively solving an inpainting task using the video generation model. This explicit geometric grounding shown to substantially improve consistency and reduce hallucination, yielding impressive reconstruction quality when the reprojected LiDAR point cloud is dense. However, as the target view moves farther from the observed viewpoints on the vehicle, the reprojected point cloud becomes increasingly sparse, making the inpainting task ambiguous.

Our method, dubbed **StreetNVS**, is designed to optimize the quality of synthesized views far from the vehicle-mounted source camera poses where reprojected point clouds are extremely sparse. Our key insight is that the vehicle rig also provides surround-view imagery from synchronized cameras, which complements LiDAR: as illustrated in Figure 2, LiDAR provides accurate but sparse and incomplete 3D structure, while reference views offer dense appearance coverage but lack metric geometry. Existing LiDAR-conditioned video models such as StreetCrafter (48) sidestep this issue with an image-to-video (I2V) formulation conditioned on the first frame, which is effective in mildly extrapolated cases but can fail catastrophically under less constrained trajectories such as large rotations away from the driving direction. In contrast, **StreetNVS** jointly leverages LiDAR and source camera views as conditioning signals, adopting recent advances in camera-controlled video generation using relative spatially aware positional embeddings (18; 54; 44) into our multi-sensor conditioned video generator.

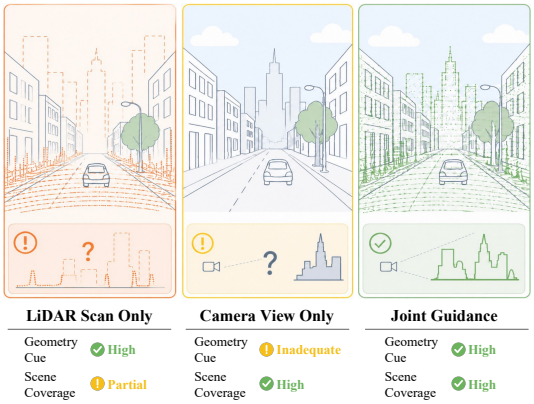


Figure 2: **Complementary Conditioning Signals.** LiDAR alone (left) lacks scene coverage and reference views alone (middle) lack metric geometry; jointly leveraging both (right) provides high geometric fidelity with full coverage.

Trained and evaluated on the Open Waymo Dataset (30) using a two-stage dense-to-balanced LiDAR curriculum, we show that StreetNVS achieves state-of-the-art performance, with improved image quality, scene faithfulness, and 3D consistency. We demonstrate the robustness of StreetNVS to varying sparse point cloud conditioning, where it exhibits little-to-no quality degradation for point clouds that are 10–100× sparser than the dense LiDAR measurements taken from the perspective of the vehicle. This capability uniquely unlocks "extreme" NVS scenarios where the synthesized target view is far from the vehicle-mounted camera observations, as exemplified in Figure 1 (b).

Our contributions include:

- We present a novel generative framework for street-view NVS that jointly conditions a video generation model on sparse LiDAR reprojections, camera poses, and reference images, substantially relaxing the dense-LiDAR requirement of prior approaches.
- We design a camera control module that incorporates multiple reference images and target video latents, providing robust appearance anchoring across changing viewpoint without relying on dense geometric input.
- We establish state-of-the-art robustness to sparse LiDAR conditioning and extreme novel viewpoints, matching or exceeding baselines that require roughly $10\times$ denser LiDAR, demonstrated through comprehensive experiments across various LiDAR sparsity levels.

2 Related Work

Street-View Novel View Synthesis. Early work on NVS for large-scale urban environments (63; 49) relied on Neural Radiance Fields (NeRF) to model scene geometry and dynamic objects (25; 31; 40). More recent approaches are based on 3D Gaussian Splatting (13) and its driving-scene extensions (47; 5; 64; 59; 3; 60), showing substantially improved rendering throughput and visual quality. However, these methods are fundamentally reconstructive: their fidelity is tightly coupled to the density and coverage of the source views, and they struggle to extrapolate beyond the captured trajectory. To overcome this, a growing line of work casts street-view NVS as a generative problem, leveraging diffusion-based video models conditioned on geometric priors (36; 60); StreetCrafter (48), for instance, trains a video generative model to synthesize novel views and then refines a Gaussian splatting representation from the generated views. Yet their performance still degrades when the target camera pose deviates significantly from the recorded trajectory, partly because the LiDAR point cloud reprojected into the target view becomes increasingly sparse.

3D Reprojection-Based Conditioning. A growing body of work treats 3D reconstruction as a structural scaffold for diffusion-based video generation. Typically, these methods condition the diffusion model on rendered geometry via ControlNet (56), VACE-style modules (11), or embedding residuals added to the noised latents, framing generation as a pixel-level inpainting task. A common approach projects geometry predicted from a 3D vision model (37; 20; 35; 10; 16; 22) and trains video diffusion models to generate missing texture in the reprojections (52; 50; 55; 43; 53; 27; 57; 9; 62). Diffusion-as-Shader (6), MotionStream (28), and Edit-by-Track (17) use 3D tracking videos as control signals, while MosaicMem (51) reprojects patchified tokens for memory efficiency. A related line of works (23; 41; 65) applies generative priors to repair artifacts and inpaint unseen regions in 3D Gaussian splats. While these methods establish a general-purpose paradigm for 3D-conditioned video generation, they rely on a feed-forward models predicting geometry from input images, hence can be metrically inconsistent. In this work, we focus on the automotive setting and exploit the accurate LiDAR and multi-view imagery directly available from the vehicle rig.

Camera Pose–Controlled Video Generation. Controlling camera motion in pretrained video generation models has recently emerged as a crucial problem, enabling controllable scene exploration and serving as a building block for world models. One line of work encodes camera parameters, such as extrinsic matrices (42; 2; 39) or Plücker ray maps (1; 7; 8; 46; 61; 19; 15), and fuses them into the latent features of pretrained video models. These approaches are lightweight but resemble absolute positional encoding, potentially limiting generalization to unseen motions. More recent works introduce relative camera-aware positional encodings (45; 26; 18; 54) into DiT-style architectures for improved 3D consistency, with RayRoPE (44) further injecting depth-based 3D correspondences. Differently from these works, we introduce LiDAR measurements to resolve scale ambiguities and further incorporate multiple reference views via cross-attention with camera-aware positional encodings, achieving significant gains in 3D reconstruction quality on street-view NVS.

3 Method

We consider autonomous driving scenarios in street-view environments within a temporal window of F frames. Modern vehicle platforms are equipped with multiple synchronized sensors, including N surround-view RGB cameras with fixed extrinsics relative to the vehicle, producing frames $\{\boldsymbol{v}_i^{(1,\dots,N)}\}_{i=1}^{F}$ with associated camera poses $\{\boldsymbol{T}_i^{(1,\dots,N)}\}_{i=1}^{F}$, and a LiDAR sensor providing per-

frame point cloud observations $\{\mathcal{L}_i\}_{i=1}^{F}$. Given the surround-view observations of the starting frame $\boldsymbol{v}^{\text{ref}} \doteq \boldsymbol{v}_1^{(1,\dots,N)}$ together with an aggregated and optionally subsampled point cloud $\mathcal{L}^{\text{agg}} \subseteq \bigcup_{i=1}^{F} \mathcal{L}_i$ from the LiDAR scan, our goal is to synthesize a novel-view video $\{\boldsymbol{v}_i^{\text{tgt}}\}_{i=1}^{F}$ along a target camera trajectory $\{\boldsymbol{T}_i^{\text{tgt}}\}_{i=1}^{F} \in \mathrm{SE}(3)^F$ that may differ substantially from the original sensor viewpoints (e.g., elevation or third-person views). To simulate the wide range of LiDAR densities encountered in practice, where reprojected point clouds become increasingly sparse as the target trajectory departs from the source cameras, we deliberately train and evaluate **StreetNVS** across multiple subsampling ratios of $\mathcal{L}^{\text{agg}}$. The synthesized video should (1) remain faithful to the accumulated RGB and LiDAR observations, (2) closely follow the prescribed target trajectory, and (3) preserve correct scene geometry and dynamic road behaviors.

3.1 Preliminaries

To address this challenging setting, we build our pipeline on top of a state-of-the-art large video diffusion model (34), leveraging the strong appearance prior learned from massive video pretraining. For controllable generation, our framework further builds on recent advances in relative camera pose encodings, which inject geometric structure directly into the transformer's attention layers.

Video Diffusion Models. Given user conditions c such as a text prompt, input frames, or camera poses, video diffusion models generate a video that follows the conditioning signal through an iterative denoising process. We adopt the Flow Matching formulation (21), which defines a probability path by linearly interpolating clean data $\boldsymbol{z}_0 \sim P_{\text{data}}$ and Gaussian noise $\boldsymbol{z}_1 \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ as $\boldsymbol{z}_t = t\boldsymbol{z}_1 + (1-t)\boldsymbol{z}_0$ for $t \in [0, 1]$, with associated velocity $\boldsymbol{z}_1 - \boldsymbol{z}_0$. A network $\boldsymbol{u}_\theta(\boldsymbol{z}_t, t, c)$ is trained to predict this velocity by minimizing the standard Flow Matching MSE loss:

$$\mathcal{L}_{\text{FM}} = \mathbb{E}_{t,c,\boldsymbol{z}_0,\boldsymbol{z}_1} \left\| \boldsymbol{u}_\theta(\boldsymbol{z}_t, t, c) - (\boldsymbol{z}_1 - \boldsymbol{z}_0) \right\|^2 . \tag{1}$$

At inference, samples are generated by integrating the learned ODE $\dot{\boldsymbol{z}}_t = \boldsymbol{u}_\theta(\boldsymbol{z}_t, t, c)$ from $t = 1$ to $t = 0$ in discrete steps, starting from pure Gaussian noise.

Relative Camera Pose Encodings. Transformer attention computes outputs as a weighted aggregation of value vectors $\boldsymbol{v}$, with weights given by the softmax of inner products between queries $\boldsymbol{q}$ and keys $\boldsymbol{k}$, all linearly projected from token features. Relative camera pose encoding injects geometric structure into this computation by associating each token with a per-token transformation $\boldsymbol{T} \in \mathrm{SE}(3)$ derived from the camera pose, and modifying $\boldsymbol{q}$, $\boldsymbol{k}$, and $\boldsymbol{v}$ through a block-diagonal matrix $\boldsymbol{D} = \boldsymbol{I}_{d/4} \otimes \boldsymbol{T}$ that repeats $\boldsymbol{T}$ along the diagonal to span the d feature channels (14; 26):

$$\mathrm{Attn}^{\text{relcam}}(\boldsymbol{q}, \boldsymbol{k}, \boldsymbol{v}) = \boldsymbol{D}^\top \mathrm{Attn}\big(\boldsymbol{D}^\top \boldsymbol{q},\, \boldsymbol{D}^{-1}\boldsymbol{k},\, \boldsymbol{D}^{-1}\boldsymbol{v}\big) . \tag{2}$$

Under this formulation, the attention score $A_{mn} \propto \boldsymbol{q}_m^\top \boldsymbol{D}_m \boldsymbol{D}_n^{-1} \boldsymbol{k}_n$ depends only on the relative transformation between token pairs, where m, n are token indices, providing explicit geometric awareness across views. We build on UCPE (54), which applies this idea at the ray level: rather than sharing a single $\boldsymbol{T}$ across all tokens of a frame, each token is assigned its own viewing ray, and a ray-to-world transformation $\boldsymbol{T}_m = \begin{bmatrix} \boldsymbol{R}_m & \boldsymbol{t}_m \\ \mathbf{0}^\top & 1 \end{bmatrix}$ is applied to its $\boldsymbol{q}$, $\boldsymbol{k}$, $\boldsymbol{v}$, where $\boldsymbol{R}_m = [\boldsymbol{x}_m, \boldsymbol{y}_m, \boldsymbol{z}_m]$ defines a local ray frame and $\boldsymbol{t}_m$ is the camera center.

3.2 Overview of StreetNVS

As motivated in Section 1, prior work such as StreetCrafter (48) conditions on only a subset of the complementary signals available from the vehicle rig. **StreetNVS** instead jointly leverages all geometric cues—camera poses, LiDAR, and surround-view reference imagery—in a unified Diffusion Transformer (DiT) framework, illustrated in Figure 3. We first render the aggregated LiDAR point cloud into the target trajectory, producing pixel-aligned RGB, validity mask, and normalized depth videos in the target view. These are encoded by the video diffusion model's VAE encoder $\mathrm{Enc}(\cdot)$. The resulting latents are concatenated channel-wise, passed through a lightweight LiDAR embedder for alignment, and added token-wise to the noisy target latent $\boldsymbol{z}^{\text{tgt}}$. In parallel, reference views $\boldsymbol{z}^{\text{ref}} = \mathrm{Enc}(\boldsymbol{v}^{\text{ref}})$ from the vehicle cameras are encoded by the same VAE and passed through the *same weight-shared* embedder. The reference image occupies the RGB channel while the LiDAR-specific channels are filled with placeholders (an all-ones mask and an all-zeros depth map), so that reference and target tokens share an identical channel layout and embedding

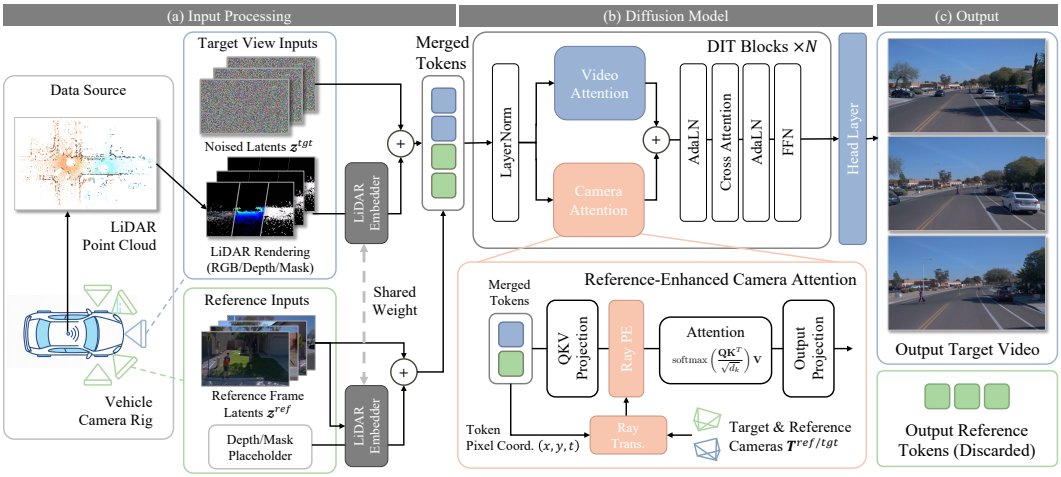


Figure 3: **Overview of StreetNVS.** Our framework performs street-view novel-view synthesis by conditioning a Diffusion Transformer (DiT) on LiDAR measurements, multi-view reference imagery and camera poses. **(a)** The **LiDAR Embedder** extracts features from LiDAR reprojections and merges them with the noised target latent z^{tgt}; reference latents z^{ref} are processed by the same weight-shared embedder with depth/mask placeholders. **(b)** Within each DiT block, our **Reference-Enhanced Camera Attention** branch runs in parallel with the video attention, applying a ray-level positional encoding (Ray PE) over target and reference poses to align features across views. **(c)** The target tokens are decoded into the **synthesized novel-view video**, while reference tokens are discarded.

pathway. The reference latents are then concatenated with the LiDAR-injected target latents to form the merged token sequence fed into the diffusion backbone.

The diffusion backbone consists of a stack of DiT blocks followed by a head layer that decodes the target tokens into the velocity prediction, while reference tokens are discarded at the output. To enable cross-view conditioning between target and reference frames while respecting their camera poses, we augment each DiT block with a dedicated camera-control branch that injects the poses of all frames into the attention computation. The original self-attention module and remaining layers are kept intact and jointly fine-tuned with the new branch.

3.3 Reference-Enhanced Camera Attention

The camera-control branch operates on the merged token sequence $\boldsymbol{z}_{1,\dots,M} = \{\boldsymbol{z}^{\text{tgt}}, \boldsymbol{z}^{\text{ref}}\}$ of length M, together with per-frame camera poses $\boldsymbol{T}^{\text{view}} = \{\boldsymbol{T}^{\text{tgt}}, \boldsymbol{T}^{\text{ref}}\}$ covering both the target trajectory and the reference views, and per-token frame indices $\tau_{1,\dots,M}$ that map each token to its corresponding pose in $\boldsymbol{T}^{\text{view}}$. We formulate the branch as global self-attention across all tokens, with relative camera-pose positional encoding applied to queries, keys, and values. Each token is first projected to query, key, and value:

$$\boldsymbol{q} = f_q(\boldsymbol{z}), \quad \boldsymbol{k} = f_k(\boldsymbol{z}), \quad \boldsymbol{v} = f_v(\boldsymbol{z}), \tag{3}$$

and then transformed by a relative camera positional encoding $\mathcal{E}$:

$$\tilde{\boldsymbol{q}} = \mathcal{E}(\boldsymbol{q}, \boldsymbol{T}, \tau), \quad \tilde{\boldsymbol{k}} = \mathcal{E}^{-1}(\boldsymbol{k}, \boldsymbol{T}, \tau), \quad \tilde{\boldsymbol{v}} = \mathcal{E}^{-1}(\boldsymbol{v}, \boldsymbol{T}, \tau), \tag{4}$$

where $\mathcal{E}^{-1}(\cdot, \boldsymbol{T}, \tau) = \mathcal{E}(\cdot, \boldsymbol{T}^{-1}, -\tau)$. The encoding $\mathcal{E}$ has two components. The first follows UCPE (54) and assigns each token a ray-level transformation $\boldsymbol{T}_m = \text{RayTrans}(\boldsymbol{T}^{\text{view}}, \tau_m, x_m, y_m)$ derived from the camera pose at frame τ_m and the token's image coordinates (x_m, y_m). The second injects the frame index τ_m via standard RoPE (29). The two encodings are concatenated channel-wise; we do not additionally encode pixel coordinates within the image plane, since the ray-level pose encoding already captures the corresponding geometry.

The encoded queries, keys, and values are passed through self-attention; the attention output is then re-encoded with $\mathcal{E}$ and projected via the output layer f_o before being passed to the feed-forward network:

$$\boldsymbol{o} = f_o\Big(\mathcal{E}\Big(\text{Attn}(\tilde{\boldsymbol{q}}, \tilde{\boldsymbol{k}}, \tilde{\boldsymbol{v}}), \boldsymbol{T}, \tau\Big)\Big). \tag{5}$$



Figure 4: **Qualitative comparison with baseline methods.** Baselines degrade substantially under sparse LiDAR conditioning. Our fine-tuned StreetCrafter (StreetCrafter*) recovers the scene only partially, with visible inconsistencies relative to the ground truth, while our model reconstructs the scene with the highest fidelity.

3.4 Two-Stage Progressive Training Curriculum

When the full pipeline is trained jointly from the start, we empirically observe that the model struggles to follow the LiDAR rendering guidance, especially in the sparse regime. We attribute this to two factors: (1) our backbone is a generic video diffusion model with no prior exposure to LiDAR-conditioned street-view synthesis, and (2) under high subsampling ratios, the LiDAR input is too sparse to provide a useful learning signal on its own.

To address this, we adopt a two-stage training curriculum. In the first stage, we disable the camera-control branch and the reference-frame inputs, training the model with only the target LiDAR renderings as guidance and emphasizing denser LiDAR samples to give the model a strong initial signal. In the second stage, we enable all components and fine-tune the full pipeline jointly, using a balanced sparsity distribution that exposes the model to the full sparsity range encountered at test time.

4 Experiments

4.1 Implementation Details

We train and evaluate our model on the Waymo Open Dataset (30), a large-scale autonomous driving benchmark captured by vehicles carrying five cameras and a LiDAR scanner. For each sample, we designate one camera as the target view over a 49-frame clip and use the initial frame of the remaining four as reference views. We use the official train and test splits. To cover a wide range of LiDAR densities, we aggregate LiDAR scans over every 10 frames per clip and subsample the aggregated points at ratios from 0.001 to 1 at order-of-magnitude intervals, providing validity masks and normalized depth maps as auxiliary inputs.

Our model is built on WAN-2.2-I2V-5B (34) and trained within the DiffSynth (4) framework using AdamW (24). A 500-iteration warm-up ramps the learning rate from 5×10^{-6} to 5×10^{-5}, followed by linear decay back to 5×10^{-6}. Each stage is trained for 10K iterations on 8 NVIDIA H100 GPUs, taking approximately 30 hours in total.

Method	PSNR ↑	SSIM ↑	LPIPS ↓	FVD ↓
FreeVS(36)	13.19	0.407	0.721	171.42
WAN VACE 14B(11)	14.67	0.430	0.623	60.68
Gen3C(27)	14.49	0.427	0.602	29.78
StreetCrafter(48)	16.81	0.492	0.494	40.78
StreetCrafter*	19.22	0.582	0.377	12.29
Ours	**20.82**	**0.619**	**0.311**	**9.12**

Table 1: **Quantitative Comparisons on Street View Synthesis** at LiDAR sparsity ratio 0.01. We additionally fine-tune StreetCrafter under the same setup for fair comparison (StreetCrafter*).



Figure 5: **Qualitative Ablation Study.** Without LiDAR projection, the model partially recovers scene identity but fails on geometry. Removing camera attention or reference views causes inconsistent generation. Our full model successfully reconstructs consistent content at the correct location.

4.2 Novel-View Synthesis Evaluation

We curate 402 evaluation clips from the Waymo Open Dataset test split, drawn from all five cameras and processed using the same protocol as the training set, and evaluate every model under sparse LiDAR conditioning at a sampling ratio of 0.01. We report PSNR, SSIM (38), LPIPS (58), and FVD (33). For baselines, we select three state-of-the-art street-view synthesis methods, namely StreetCrafter (48), FreeVS (36), and Gen3C (27), all trained on Waymo with dense LiDAR, as well as a general-purpose video-to-video model that supports point reprojection inputs, WAN-2.1-VACE-14B (11). To ensure a fair comparison, we additionally re-implement StreetCrafter on top of our backbone and retrain it across the same range of LiDAR sparsity levels, denoted as StreetCrafter*.

As shown in Table 1, our method substantially outperforms all baselines across all metrics. Qualitatively, Figure 4 shows that FreeVS, Gen3C, and the original StreetCrafter degrade severely under sparse LiDAR, either reproducing the scattered points directly, hallucinating incorrect scenes, or producing blurred output. Our retrained StreetCrafter*, the strongest comparison as it is exposed to the same sparsity range as our model, recovers more of the scene but still exhibits visible inconsistencies in geometry and object placement. In contrast, our model reconstructs the scene with the highest fidelity, faithfully reproducing both global layout and fine appearance details.

4.3 Ablation Study

To verify the effectiveness of each component, we ablate four variants under the same evaluation protocol: *Ours w/ Camera Only* relies solely on camera attention and discards LiDAR-derived inputs; *Ours w/ Projection Only* uses only LiDAR reprojection videos, without camera attention or reference frames; *Ours w/o Reference Views* combines LiDAR projections with camera attention but omits reference views; and *Ours w/o Progressive Training* trains end-to-end in a single stage. Table 2 shows that our full model consistently outperforms all variants, with two notable findings: *Ours w/o Reference Views* surpasses *Ours w/ Projection Only*, indicating that camera attention alone provides a meaningful boost; and omitting the two-stage curriculum substantially degrades performance.

Qualitative results in Figure 5 further illustrate these trends: without LiDAR, the model partially recovers scene identity from cross-view cues but produces incorrect geometry; without camera attention or reference views, it captures consistent layout but fails to fill regions uncovered by LiDAR

Method	PSNR ↑	SSIM ↑	LPIPS ↓	FVD ↓
Ours w/ Camera Only	17.60	0.541	0.452	15.77
Ours w/ Projection Only	19.98	0.598	0.354	12.00
Ours w/o Reference Views	20.34	0.607	0.334	10.28
Ours w/o Progressive Training	19.51	0.583	0.365	11.13
Ours Full	**20.82**	**0.619**	**0.311**	**9.12**

Table 2: **Ablation Study at LiDAR sparsity ratio 0.01.** Removing any of the LiDAR projection embedding, camera attention, reference views, or two-stage curriculum degrades performance, with the full model performing best across all metrics.

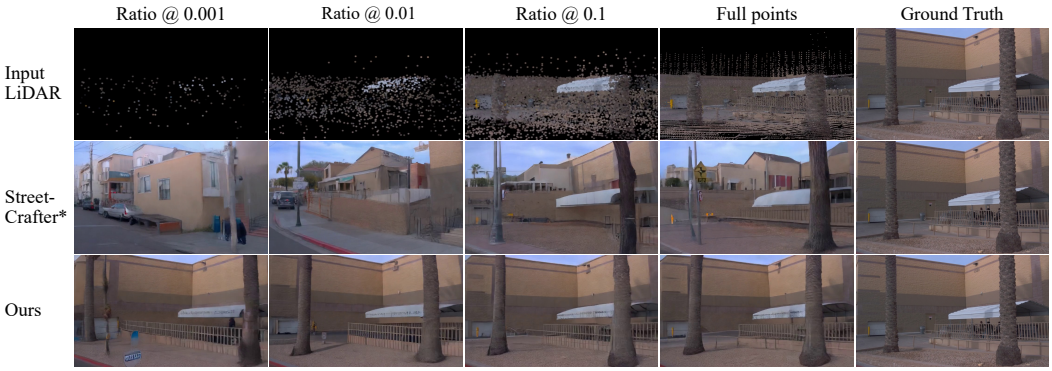


Figure 6: **Qualitative comparison across varying sparsities.** Each row shows a single frame from the synthesized video of all methods. Our method preserves high reconstruction fidelity across all levels of LiDAR sparsity, substantially outperforming the baseline.

(e.g., the top of the pillar in the first example). Only the full model handles both aspects, yielding results that most closely match the ground truth.

4.4 Comparison Across LiDAR Densities

Beyond the fixed-sparsity setting, we further evaluate across the full range of LiDAR ratios in our dataset, from 0.001 to 1. Figure 7 reports our full model, the ablation variants, and both versions of StreetCrafter. The original StreetCrafter, not exposed to sparse LiDAR during training, degrades sharply as density decreases, while the fine-tuned variant improves in the sparse regime but weakens in the dense one. Our model outperforms all baselines across the entire spectrum, and the ablation comparison shows that the gains from our camera attention, LiDAR embedding, and reference-view conditioning grow more pronounced as LiDAR becomes sparser, demonstrating the complementary nature of the signals from multi-sensors.

Figure 6 provides qualitative comparisons between StreetCrafter* and our method. At low sparsity ratios, rendered LiDAR points become widely spaced relative to their render radius, producing scattered blobs that no longer convey continuous scene structure; StreetCrafter* fails to recover the correct scene under these conditions. Our method, in contrast, produces nearly identical reconstructions across all sparsity levels, preserving fine details such as the awning, railing, and palm tree even at the sparsest settings, reflecting how reference views and camera-pose conditioning supply cues that remain stable independent of LiDAR density.

4.5 Unseen Novel-View Synthesis

To assess generalization beyond the training distribution, we evaluate our model on four categories of novel-view trajectories not seen during training and far from the recorded driving path: *elevation*, in which the camera is gradually raised above the vehicle; *lane-shift*, in which the camera is translated to adjacent lanes; *pull-back*, in which the camera dollies straight backward along its forward axis; and *rotation*, in which the camera turns left and right to look around.

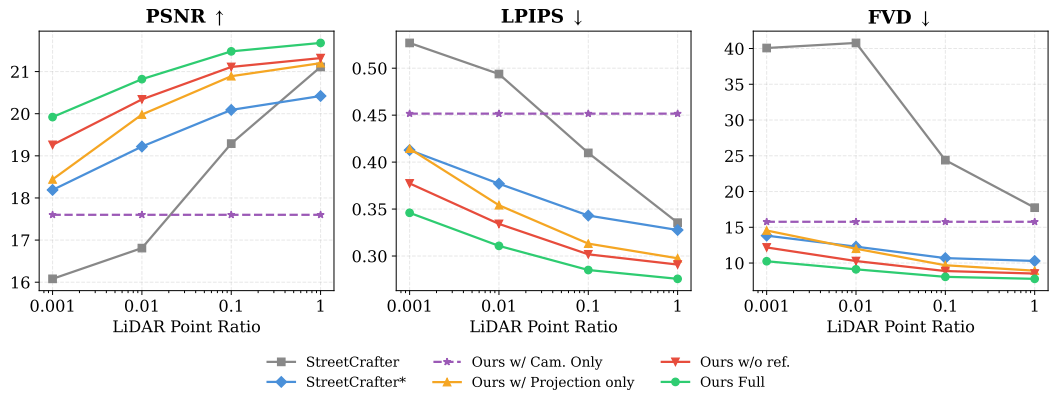


Figure 7: **Quantitative Comparison across LiDAR densities.** Our full model consistently outperforms all baselines and ablation variants. The gap narrows as density increases, since dense points gradually dominate the conditioning signal, but our model remains best throughout. Note that ratio 1 corresponds to full LiDAR and represents the upper bound of density achievable in our benchmark.



Figure 8: **Results on unseen trajectories far from the recorded path.** The red arrow in the camera visualization indicates the facing direction of the vehicle. Our method handles a variety of extreme novel trajectories absent from the training data while maintaining high coherence and consistency.

As shown in Figure 8, our model preserves scene coherence across all four trajectory types. The rotation case is particularly demanding: the model successfully reconstructs the forward-facing scene at the end of the trajectory to match the starting frame, despite an extended interval in the middle during which the frontal region is entirely unobserved. The pull-back case similarly stresses geometric and amodal completion: the camera retreats far behind its original position, requiring the model to synthesize plausible content for large disoccluded regions while keeping the originally visible structure stable across frames.

5 Conclusion

We presented **StreetNVS**, a unified framework for street-view NVS that jointly conditions a video diffusion backbone on three complementary signals from the vehicle rig: LiDAR reprojections, per-frame camera poses, and surround-view reference imagery. LiDAR pins down geometry where observed, while pose-aware reference views, fused through our Reference-Enhanced Camera Atten-

tion module with relative ray-level positional encoding, fill in appearance and structure elsewhere; a two-stage curriculum further enables robustness across LiDAR sparsity levels at test time. **StreetNVS** substantially outperforms state-of-the-art baselines under sparse LiDAR, matches methods that rely on 10–100$\times$ denser point clouds, and generalizes to extreme out-of-trajectory viewpoints such as elevation, lane-shift, spiral, and rotation. On the other hand, our current evaluation is limited to clips of moderate length, and scaling to longer driving sequences is a natural next direction. We also focus exclusively on novel-view generation from existing scene content; extending **StreetNVS** toward generative scene editing, such as inserting or removing objects, is a promising direction for future work.

Societal Impact. **StreetNVS** is one step toward multi-sensor street-view synthesis as a practical building block for autonomous-driving world models, closed-loop safety simulation, and large-scale data augmentation. At the same time, like other photorealistic video generators, it could in principle be misused to fabricate misleading driving footage, and synthesized views that look convincing but are subtly geometrically wrong should not be over-trusted in safety-critical pipelines without careful validation.

Acknowledgement. The authors gratefully acknowledge support from Rivian and the Toyota Research Institute (TRI). Some of the computing for this project was performed on the Marlowe (12) cluster managed by Stanford Data Science and administered by Stanford Research Computing. We thank the support from NVIDIA Academic Grant Program for this project.

References

[1] S. Bahmani, I. Skorokhodov, G. Qian, A. Siarohin, W. Menapace, A. Tagliasacchi, D. B. Lindell, and S. Tulyakov. Ac3d: Analyzing and improving 3d camera control in video diffusion transformers. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 22875–22889, 2025.

[2] J. Bai, M. Xia, X. Fu, X. Wang, L. Mu, J. Cao, Z. Liu, H. Hu, X. Bai, P. Wan, and D. Zhang. Recammaster: Camera-controlled generative rendering from a single video, 2025.

[3] Z. Chen, J. Yang, J. Huang, R. De Lutio, J. M. Esturo, B. Ivanovic, O. Litany, Z. Gojcic, S. Fidler, M. Pavone, et al. Omnire: Omni urban scene reconstruction. *arXiv preprint arXiv:2408.16760*, 2024.

[4] Z. Di, G. Zhu, Z. Duan, Z. Chu, Y. Chen, and W. Lu. Diffsynth-engine: a high-performance diffusion inference engine. https://github.com/modelscope/diffsynth-engine, 2025.

[5] T. Fischer, J. Kulhanek, S. R. Bulo, L. Porzi, M. Pollefeys, and P. Kontschieder. Dynamic 3d gaussian fields for urban areas. *arXiv preprint arXiv:2406.03175*, 2024.

[6] Z. Gu, R. Yan, J. Lu, P. Li, Z. Dou, C. Si, Z. Dong, Q. Liu, C. Lin, Z. Liu, W. Wang, and Y. Liu. Diffusion as shader: 3d-aware video diffusion for versatile video generation control. *SIGGRAPH*, 2025.

[7] H. He, Y. Xu, Y. Guo, G. Wetzstein, B. Dai, H. Li, and C. Yang. Cameractrl: Enabling camera control for text-to-video generation. In *ICLR*, 2025.

[8] H. He, C. Yang, S. Lin, Y. Xu, M. Wei, L. Gui, Q. Zhao, G. Wetzstein, L. Jiang, and H. Li. Cameractrl ii: Dynamic scene exploration via camera-controlled video diffusion models. *arXiv preprint arXiv:2503.10592*, 2025.

[9] C. Hou and Z. Chen. Training-free camera control for video generation. *arXiv preprint arXiv:2406.10126*, 2024.

[10] J. Huang, Q. Zhou, H. Rabeti, A. Korovko, H. Ling, X. Ren, T. Shen, J. Gao, D. Slepichev, C.-H. Lin, J. Ren, K. Xie, J. Biswas, L. Leal-Taixe, and S. Fidler. Vipe: Video pose engine for 3d geometric perception. In *NVIDIA Research Whitepapers*, 2025.

[11] Z. Jiang, Z. Han, C. Mao, J. Zhang, Y. Pan, and Y. Liu. Vace: All-in-one video creation and editing. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 17191–17202, 2025.

[12] C. Kapfer, K. Stine, B. Narasimhan, C. Mentzel, and E. Candes. Marlowe: Stanford's gpu-based computational instrument, Jan. 2025.

[13] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4), July 2023.

[14] X. Kong, S. Liu, X. Lyu, M. Taher, X. Qi, and A. J. Davison. Eschernet: A generative model for scalable view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9503–9513, 2024.

[15] Z. Kuang, S. Cai, H. He, Y. Xu, H. Li, L. Guibas, and G. Wetzstein. Collaborative video diffusion: Consistent multi-video generation with camera control. In *arXiv*, 2024.

[16] Z. Kuang, T. Zhang, K. Zhang, H. Tan, S. Bi, Y. Hu, Z. Xu, M. Hasan, G. Wetzstein, and F. Luan. Buffer anytime: Zero-shot video depth and normal from image priors. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 17660–17670, 2025.

[17] Y.-C. Lee, Z. Zhang, J. Huang, J.-H. Wang, J.-Y. Lee, J.-B. Huang, E. Shechtman, and Z. Li. Generative video motion editing with 3d point tracks. *arXiv preprint arXiv:2512.02015*, 2025.

[18] R. Li, B. Yi, J. Liu, H. Gao, Y. Ma, and A. Kanazawa. Cameras as relative positional encoding. *arXiv preprint arXiv:2507.10496*, 2025.

[19] T. Li, G. Zheng, R. Jiang, T. Wu, Y. Lu, Y. Lin, X. Li, et al. Realcam-i2v: Real-world image-to-video generation with interactive complex camera control. *arXiv preprint arXiv:2502.10059*, 2025.

[20] H. Lin, S. Chen, J. H. Liew, D. Y. Chen, Z. Li, G. Shi, J. Feng, and B. Kang. Depth anything 3: Recovering the visual space from any views. *arXiv preprint arXiv:2511.10647*, 2025.

[21] Y. Lipman, R. T. Chen, H. Ben-Hamu, M. Nickel, and M. Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

[22] S. Liu, K. W. Ng, W. Jang, J. Guo, J. Han, H. Liu, Y. Douratsos, J. C. Pérez, Z. Zhou, C. Phung, et al. Scaling sequence-to-sequence generative neural rendering. *arXiv preprint arXiv:2510.04236*, 2025.

[23] X. Liu, C. Zhou, and S. Huang. 3dgs-enhancer: Enhancing unbounded 3d gaussian splatting with view-consistent 2d diffusion priors. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

[24] I. Loshchilov and F. Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[25] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021.

[26] T. Miyato, B. Jaeger, M. Welling, and A. Geiger. Gta: A geometry-aware attention mechanism for multi-view transformers. *arXiv preprint arXiv:2310.10375*, 2023.

[27] X. Ren, T. Shen, J. Huang, H. Ling, Y. Lu, M. Nimier-David, T. Müller, A. Keller, S. Fidler, and J. Gao. Gen3c: 3d-informed world-consistent video generation with precise camera control. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 6121–6132, 2025.

[28] J. Shin, Z. Li, R. Zhang, J.-Y. Zhu, J. Park, E. Shechtman, and X. Huang. MotionStream: Real-Time Video Generation with Interactive Motion Controls. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2026.

[29] J. Su, M. Ahmed, Y. Lu, S. Pan, W. Bo, and Y. Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

[30] P. Sun, H. Kretzschmar, X. Dotiwalla, A. Chouard, V. Patnaik, P. Tsui, J. Guo, Y. Zhou, Y. Chai, B. Caine, V. Vasudevan, W. Han, J. Ngiam, H. Zhao, A. Timofeev, S. Ettinger, M. Krivokon, A. Gao, A. Joshi, Y. Zhang, J. Shlens, Z. Chen, and D. 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.

[31] M. Tancik, V. Casser, X. Yan, S. Pradhan, B. Mildenhall, P. P. Srinivasan, J. T. Barron, and H. Kretzschmar. Block-nerf: Scalable large scene neural view synthesis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8248–8258, 2022.

[32] A. Tewari, J. Thies, B. Mildenhall, P. Srinivasan, E. Tretschk, W. Yifan, C. Lassner, V. Sitzmann, R. Martin-Brualla, S. Lombardi, et al. Advances in neural rendering. In *Computer Graphics Forum*, volume 41, pages 703–735. Wiley Online Library, 2022.

[33] T. Unterthiner, S. Van Steenkiste, K. Kurach, R. Marinier, M. Michalski, and S. Gelly. Towards accurate generative models of video: A new metric & challenges. *arXiv preprint arXiv:1812.01717*, 2018.

[34] T. Wan, A. Wang, B. Ai, B. Wen, C. Mao, C.-W. Xie, D. Chen, F. Yu, H. Zhao, J. Yang, et al. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025.

[35] J. Wang, M. Chen, N. Karaev, A. Vedaldi, C. Rupprecht, and D. Novotny. Vggt: Visual geometry grounded transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025.

[36] Q. Wang, L. Fan, Y. Wang, Y. Chen, and Z. Zhang. Freevs: Generative view synthesis on free driving trajectory. *arXiv preprint arXiv:2410.18079*, 2024.

[37] S. Wang, V. Leroy, Y. Cabon, B. Chidlovskii, and J. Revaud. Dust3r: Geometric 3d vision made easy. In *CVPR*, 2024.

[38] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004.

[39] Z. Wang, Z. Yuan, X. Wang, Y. Li, T. Chen, M. Xia, P. Luo, and Y. Shan. Motionctrl: A unified and flexible motion controller for video generation. In *ACM SIGGRAPH 2024 Conference Papers*, pages 1–11, 2024.

[40] C. Wu, J. Sun, Z. Shen, and L. Zhang. Mapnerf: Incorporating map priors into neural radiance fields for driving view simulation. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7082–7088. IEEE, 2023.

[41] J. Z. Wu, Y. Zhang, H. Turki, X. Ren, J. Gao, M. Z. Shou, S. Fidler, Z. Gojcic, and H. Ling. Difix3d+: Improving 3d reconstructions with single-step diffusion models. *CVPR*, 2025.

[42] R. Wu, R. Gao, B. Poole, A. Trevithick, C. Zheng, J. T. Barron, and A. Holynski. CAT4D: Create Anything in 4D with Multi-View Video Diffusion Models. *arXiv:2411.18613*, 2024.

[43] T. Wu, S. Yang, R. Po, Y. Xu, Z. Liu, D. Lin, and G. Wetzstein. Video world models with long-term spatial memory. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

[44] Y. Wu, M. Jeon, J.-H. R. Chang, O. Tuzel, and S. Tulsiani. Rayrope: Projective ray positional encoding for multi-view attention. *arXiv preprint arXiv:2601.15275*, 2026.

[45] K. Xiong, S. Gong, X. Ye, X. Tan, J. Wan, E. Ding, J. Wang, and X. Bai. Cape: Camera view position embedding for multi-view 3d object detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 21570–21579, 2023.

[46] D. Xu, W. Nie, C. Liu, S. Liu, J. Kautz, Z. Wang, and A. Vahdat. Camco: Camera-controllable 3d-consistent image-to-video generation. *arXiv preprint arXiv:2406.02509*, 2024.

[47] Y. Yan, H. Lin, C. Zhou, W. Wang, H. Sun, K. Zhan, X. Lang, X. Zhou, and S. Peng. Street gaussians: Modeling dynamic urban scenes with gaussian splatting. In *European Conference on Computer Vision*, pages 156–173. Springer, 2024.

[48] Y. Yan, Z. Xu, H. Lin, H. Jin, H. Guo, Y. Wang, K. Zhan, X. Lang, H. Bao, X. Zhou, and S. Peng. Streetcrafter: Street view synthesis with controllable video diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

[49] Z. Yang, Y. Chen, J. Wang, S. Manivasagam, W.-C. Ma, A. J. Yang, and R. Urtasun. Unisim: A neural closed-loop sensor simulator. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1389–1399, 2023.

[50] M. YU, W. Hu, J. Xing, and Y. Shan. Trajectorycrafter: Redirecting camera trajectory for monocular videos via diffusion models. *arXiv preprint arXiv:2503.05638*, 2025.

[51] W. Yu, R. Qian, Y. Li, L. Wang, S. Yin, D. Anthony, Y. Ye, Y. Li, W. Wan, A. Garg, et al. Mosaicmem: Hybrid spatial memory for controllable video world models. *arXiv preprint arXiv:2603.17117*, 2026.

[52] W. Yu, J. Xing, L. Yuan, W. Hu, X. Li, Z. Huang, X. Gao, T.-T. Wong, Y. Shan, and Y. Tian. Viewcrafter: Taming video diffusion models for high-fidelity novel view synthesis. *TPAMI*, 2024.

[53] S. Zhai, Z. Ye, J. Liu, W. Xie, J. Hu, Z. Peng, H. Xue, D. Chen, X. Wang, L. Yang, et al. Stargen: A spatiotemporal autoregression framework with video diffusion model for scalable and controllable scene generation. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 26822–26833, 2025.

[54] C. Zhang, B. Li, M. Wei, Y.-P. Cao, C. C. Gambardella, D. Phung, and J. Cai. Unified camera positional encoding for controlled video generation. *arXiv preprint arXiv:2512.07237*, 2025.

[55] J. Zhang, Y. Li, A. Chen, M. Xu, K. Liu, J. Wang, X.-X. Long, H. Liang, Z. Xu, H. Su, et al. Advances in feed-forward 3d reconstruction and view synthesis: A survey. *arXiv preprint arXiv:2507.14501*, 2025.

[56] L. Zhang, A. Rao, and M. Agrawala. Adding conditional control to text-to-image diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 3836–3847, October 2023.

[57] Q. Zhang, S. Zhai, M. A. B. Martin, K. Miao, A. Toshev, J. Susskind, and J. Gu. World-consistent video diffusion with explicit 3d modeling. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 21685–21695, 2025.

[58] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018.

[59] G. Zhao, C. Ni, X. Wang, Z. Zhu, X. Zhang, Y. Wang, G. Huang, X. Chen, B. Wang, Y. Zhang, et al. Drivedreamer4d: World models are effective data machines for 4d driving scene representation. In *Proceedings of the computer vision and pattern recognition conference*, pages 12015–12026, 2025.

[60] G. Zhao, X. Wang, C. Ni, Z. Zhu, W. Qin, G. Huang, and X. Wang. Recondreamer++: Harmonizing generative and reconstructive models for driving scene representation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 26718–26728, 2025.

[61] G. Zheng, T. Li, R. Jiang, Y. Lu, T. Wu, and X. Li. Cami2v: Camera-controlled image-to-video diffusion model. *arXiv preprint arXiv:2410.15957*, 2024.

[62] S. Zheng, Z. Peng, Y. Zhou, Y. Zhu, H. Xu, X. Huang, and Y. Fu. Vidcraft3: Camera, object, and lighting control for image-to-video generation. *arXiv preprint arXiv:2502.07531*, 2025.

[63] H. Zhou, J. Shao, L. Xu, D. Bai, W. Qiu, B. Liu, Y. Wang, A. Geiger, and Y. Liao. Hugs: Holistic urban 3d scene understanding via gaussian splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21336–21345, 2024.

[64] X. Zhou, Z. Lin, X. Shan, Y. Wang, D. Sun, and M.-H. Yang. Drivinggaussian: Composite gaussian splatting for surrounding dynamic autonomous driving scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21634–21643, 2024.

[65] L. Zhu, M. Narayana, M. Stary, W. Hutchcroft, G. Wetzstein, and I. Armeni. Gaussfusion: Improving 3d reconstruction in the wild with a geometry-informed video generator. *arXiv preprint arXiv:2603.25053*, 2026.

A More Results

Please check our website (https://streetnvs.github.io) for all animated results.

We provide a finer-grained comparison with baselines by splitting the evaluation data into an *easy* set, consisting of frontal cameras (Camera #0, #1, #2) whose initial views align closely with the target trajectory, and a *hard* set, consisting of side cameras (Camera #3, #4) whose initial views overlap only marginally with the rest of the video. As shown in Table 3, our method outperforms all baselines on both subsets, with a particularly large margin on the hard cases where baselines degrade substantially while our method maintains performance close to that of the easy cases.

We further compare against StreetCrafter*, the strongest baseline in our comparison, on our novel-view trajectories. As shown in Figure 9, by jointly leveraging reference views and camera-pose conditioning in addition to LiDAR, our model produces results that are more structurally coherent and more geometrically faithful than the baseline across all three trajectory types.

Beyond the qualitative results in the main paper, we provide additional comparisons in Figure 10 on the evaluation set, and in Figure 11 on the novel-view trajectories.

Table 3: **Quantitative Comparison by Camera Viewpoint.** Evaluated at sparsity ratio 0.01. Frontal cameras $(0/1/2)$ provide initial views well-aligned with the target trajectory, while side cameras $(3/4)$ provide initial views that overlap only marginally with the rest of the video.

	Frontal Cameras (Easy)				**Side Cameras (Hard)**			
Method	PSNR ↑	SSIM ↑	LPIPS ↓	FVD ↓	PSNR ↑	SSIM ↑	LPIPS ↓	FVD ↓
FreeVS (36)	12.51	0.4116	0.7245	197.93	13.87	0.4030	0.7172	179.41
Gen3C (27)	14.49	0.4517	0.5487	38.49	14.49	0.4028	0.6542	45.61
VACE (11)	15.27	0.4726	0.5548	78.69	14.07	0.3880	0.6914	77.52
StreetCrafter (48)	17.92	0.5636	0.4150	43.48	15.70	0.4211	0.5726	61.47
StreetCrafter*	19.65	0.6065	0.3186	15.31	18.80	0.5584	0.4354	23.31
StreetNVS (Ours)	**20.93**	**0.6364**	**0.2710**	**11.60**	**20.71**	**0.6018**	**0.3506**	**16.27**

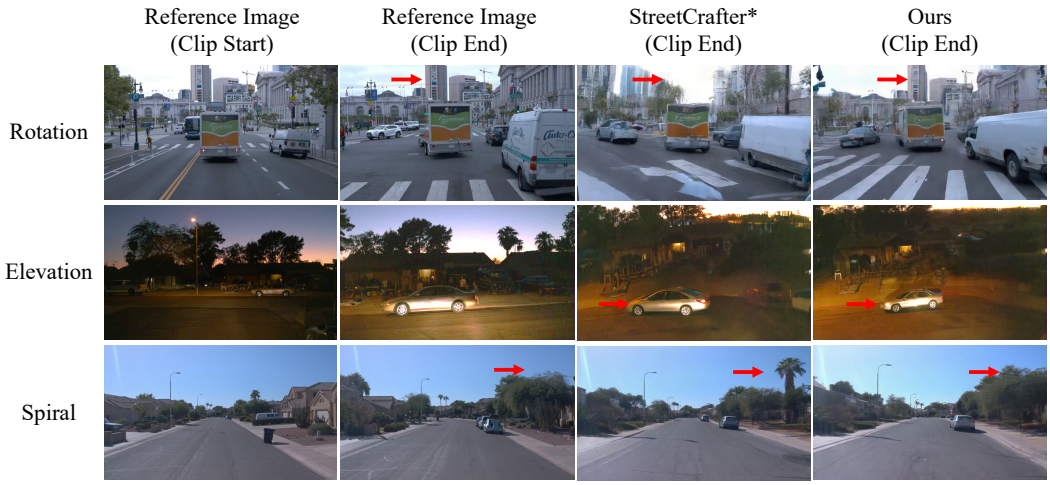


Figure 9: **Qualitative Comparison on Unseen Novel Views.** For each trajectory, we show the start and end frames of the original vehicle video (left two columns) alongside the end-frame predictions of StreetCrafter* and our model. Across rotation, elevation, and spiral cases, our model produces more coherent and geometrically faithful results: it preserves correct background structure under rotation, respects the elevated viewpoint's perspective, and maintains the correct metric scale in the spiral case where the baseline still shows scene content the camera should have already passed.

B Additional Implementation Details

B.1 Model

Backbone and Module Integration. Our model is built on top of the WAN-2.2-I2V-5B (34) backbone. Aside from the original model, the LiDAR embedding layer consists of two convolutional layers followed by a two-layer MLP that projects the LiDAR features into the DiT's hidden dimension, and the Camera Attention branch is added in parallel within each DiT block as described in Section 3. To accommodate reference inputs, we encode reference views frame-by-frame with the WAN VAE encoder, producing the same frame count in token space. To disambiguate reference tokens from target tokens within the noisy latent, we offset their temporal index by a constant of 64, well beyond the maximum temporal length of the latent space in our setup, $\lfloor 81/4 \rfloor = 20$. The reference tokens are then padded with constant LiDAR placeholder channels and concatenated with the target tokens, after which they are treated identically to target tokens by the network.

Camera Pose Encoding. We use the registered camera poses provided with the dataset. Because the video VAE compresses several frames into a single token along the temporal axis, we use the pose of the first frame within each compressed group as the representative pose for the corresponding token. Empirically, alternative choices (e.g., the middle or last frame) yield similar results. Since UCPE (54) is scale-sensitive, we rescale the scene by a factor of 0.1 relative to the metric scale to match the regime in which UCPE performs best. In all experiments, the camera-pose encoding occupies $3/4$ of each attention head's channels (96 channels), while the remaining $1/4$ (32 channels) is reserved for the frame-index encoding.

B.2 Data

Data Processing. For each scene in the Waymo Open Dataset, we sample LiDAR scans at every 10th frame (i.e., frames $0, 10, 20, \dots$) and aggregate them into a single point cloud, which we then uniformly subsample at several sparsity ratios. The aggregated points are rendered into the target trajectory using the differentiable point renderer of StreetCrafter (48) with a point radius of 0.02 in NDC space. Rendered depth maps are normalized to $[0, 1]$ and converted to RGB via a perceptual colormap (Jet). Training clips are extracted as 49-frame subsequences with a stride of 10 frames, yielding approximately $61{,}000$ clips per sparsity ratio. The evaluation set is constructed analogously, but uses only the first 49 frames of each scene, producing 402 evaluation clips in total.

Novel-View Trajectory Construction. To evaluate generation under unseen viewpoints, we construct five families of novel camera trajectories that deviate from the original driving path. Let $f \in \{0, \dots, F-1\}$ index frames, $t = f/(F-1) \in [0, 1]$ denote normalized time, and $\boldsymbol{T}_f$ the original front-camera-to-world pose. We decompose $\boldsymbol{T}_f$ as

$$\boldsymbol{T}_f = \begin{bmatrix} \boldsymbol{R}_f & \mathbf{c}_f \\ \mathbf{0} & 1 \end{bmatrix} = \big[\mathbf{r}_f \mid \mathbf{d}_f \mid \mathbf{f}_f \mid \mathbf{c}_f\big],$$

where $\boldsymbol{R}_f = [\mathbf{r}_f, \mathbf{d}_f, \mathbf{f}_f]$ is the rotation matrix whose columns are the camera's right, down, and forward axes, and $\mathbf{c}_f$ is the camera center. The four trajectory families are defined as follows.

Elevation. The camera is gradually lifted toward a bird's-eye view by interpolating between the original pose and an elevated, pitched target pose:

$$\boldsymbol{T}_f^{\text{ele}}(t) = \text{Lerp}(\boldsymbol{T}_f, \boldsymbol{T}_f^{\text{BEV}}, t),$$

where $\boldsymbol{T}_f^{\text{BEV}}$ is constructed by lifting the camera to height $h = 5\,\text{m}$, shifting it backward by $b = 2\,\text{m}$, and applying a downward pitch of $\theta = 30^\circ$.

Spiral. The camera orbits the original viewpoint along a circular path of radius $r = 1.5\,\text{m}$, completing $K = 2$ full revolutions while keeping its orientation fixed:

$$\mathbf{c}_f^{\text{spi}}(t) = \mathbf{c}_f + r\big[(\cos\alpha - 1)\,\mathbf{r}_f + \sin\alpha\,\mathbf{d}_f\big], \quad \alpha = 2\pi K t, \quad \boldsymbol{R}_f^{\text{spi}} = \boldsymbol{R}_f.$$

Lane-Shift. The camera is laterally displaced by a sinusoidal offset of amplitude $A = 3.5\,\text{m}$ along the lateral axis $\hat{\mathbf{r}}_f$ (orthogonal to the driving direction), simulating lane-changing behavior.

$$\mathbf{c}_f^{\text{lane}}(t) = \mathbf{c}_f + sA\sin(2\pi t)\,\hat{\mathbf{r}}_f, \quad \boldsymbol{R}_f^{\text{lane}} = \boldsymbol{R}_f,$$

where $s \in \{-1, +1\}$ is a per-scene sign that determines whether the camera shifts to the left or right adjacent lane.

Rotation. The camera yaw oscillates within a range of $\psi_{\max} = 60°$, smoothly interpolating between a set of precomputed camera-to-world poses $\{\boldsymbol{T}_f^{(j)}\}_{j=0}^{4}$ corresponding to canonical yaw angles $\psi^{(j)} \in [-90, -45, 0, 45, 90]°$:

$$\psi(t) = \psi_{\max} \sin(2\pi t), \quad \boldsymbol{T}_f^{\mathrm{rot}}(t) = \mathrm{Lerp}(\boldsymbol{T}_f^{(j)}, \boldsymbol{T}_f^{(j+1)}, \beta),$$

where j is the index such that $\psi^{(j)} \le \psi(t) < \psi^{(j+1)}$, and $\beta = (\psi(t) - \psi^{(j)})/(\psi^{(j+1)} - \psi^{(j)}) \in [0, 1]$ is the linear interpolation weight. The wider precomputed pose set supports flexible yaw selection; in all our experiments, the actual oscillation stays within $\psi_{\max} = 60°$.

In all uses of $\mathrm{Lerp}(\cdot)$, the rotation component of the interpolated pose is re-orthonormalized to ensure the result lies in $\mathrm{SE}(3)$.

B.3 Training

We fine-tune the WAN backbone end-to-end together with the newly introduced modules. Since fine-tuning all parameters of the original model already requires ~ 70 GB of GPU memory, we apply LoRA with rank 1024 to the FFN layers, reducing memory consumption to ~ 40 GB and leaving sufficient headroom for the additional modules introduced by our framework.

We adopt a two-stage progressive training curriculum. In the first stage, we sample data from the four sparsity ratios (from 0.001 to 1) with weights $\{0, 0.25, 0.25, 0.5\}$, disable the Camera Attention branch, and train only the original WAN backbone together with the LiDAR embedding layers, using the rendered LiDAR buffers as the sole conditioning signal. In the second stage, we rebalance the sparsity distribution to $\{0.15, 0.35, 0.35, 0.15\}$ and fine-tune the full model with all modules enabled. Both stages run for 10K iterations under the same learning-rate schedule.

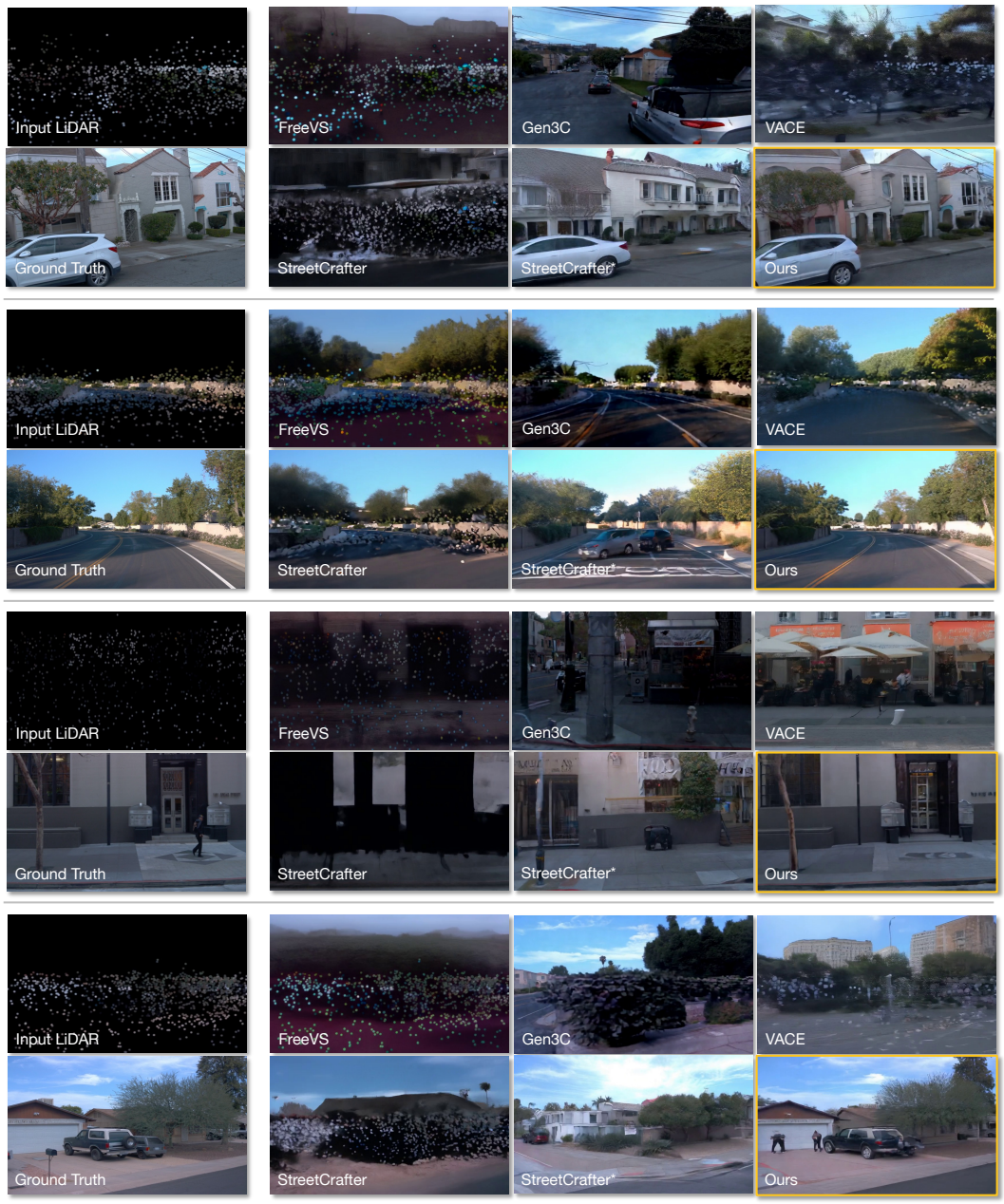


Figure 10: **More Qualitative comparison on the evaluation dataset**. All models are evaluated with 0.01 LiDAR sparsity ratio. Our method substantially outperforms all baselines.



Figure 11: **More results on Unseen Novel Views**.

NeurIPS Paper Checklist

1. **Claims**
   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?
   Answer: [Yes]
   Justification: The abstract and introduction states the contributions of the paper.
   Guidelines:
   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.
2. **Limitations**
   Question: Does the paper discuss the limitations of the work performed by the authors?
   Answer: [Yes]
   Justification: Limitations are discussed in the Sec. 5.
   Guidelines:
   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.
3. **Theory assumptions and proofs**
   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?
   Answer: [N/A]
   Justification: Paper does not include theoretical results.
   Guidelines:
   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.

- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**
   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?
   Answer: [Yes]
   Justification: Implementation details and experiment setups are included in Sec. 4 and the Appendix.
   Guidelines:
   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**
   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?
   Answer: [No]
   Justification: We are not releasing our code and data at the reviewing stage, but plan to make them available upon publication.
   Guidelines:
   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not

including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**
   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?
   Answer: [Yes]
   Justification: See Sec. 4 and Appendix.
   Guidelines:
   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.
7. **Experiment statistical significance**
   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?
   Answer: [No]
   Justification: We report the average score of the method results.
   Guidelines:
   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
   - If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.
8. **Experiments compute resources**
   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?
   Answer: [Yes]
   Justification: See Sec. 4 and the Appendix.
   Guidelines:
   - The answer [N/A] means that the paper does not include experiments.

- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: Our work conform with all ethic requirements.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: See Sec. 5

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: Our paper does not have such risks.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.

- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**
    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?
    Answer: [Yes]
    Justification: We cite the original contributors of the data and models used in this work.
    Guidelines:
    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.
13. **New assets**
    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?
    Answer: [N/A].
    Justification: Our paper does not release new assets.
    Guidelines:
    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.
14. **Crowdsourcing and research with human subjects**
    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?
    Answer: [N/A].
    Justification: The paper does not involve study on human subjects.
    Guidelines:
    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.
15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?
Answer: [N/A]
Justification: The paper does not involve study on human subjects.
Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**
Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.
Answer: [N/A]
Justification: The core contribution does not involve LLMs.
Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.