Instant NuRec: Feed-Forward Neural Reconstruction for Driving Scene Simulation

NVIDIA[†]

https://github.com/nvidia/instant-nurec

Figure 1: ***Instant NuRec*** ingests a multi-camera driving sequence and, in a single feed-forward pass, produces a layered 3DGS world with static, dynamic, and sky components that the user can freely navigate spatially and temporally. For each scene (top and bottom), the location markers indicate the vehicle positions used to produce the renderings in the right columns.

Abstract

3D simulation platforms are critical for autonomous driving because they enable end-to-end policy evaluation, thereby reducing development costs and improving safety. In recent years, neural simulation has become predominant, with methods such as NuRec [1] playing a central role; however, these methods remain relatively slow and typically require per-scene tuning. In this work, we present *Instant NuRec*, a feed-forward neural reconstruction model that turns a short multi-view driving log into a fully simulatable 3D Gaussian Splatting (3DGS) world in a single forward pass. The model accepts multi-view input from a calibrated camera rig and emits a layered output consisting of static and dynamic 3DGS layers, a sky cubemap, and per-camera ISP corrections, while providing native support for non-pinhole camera models via 3DGUT [2]. It reconstructs a 10–20-second multi-camera scene in roughly 1.5 seconds and achieves 28.26 dB full-image PSNR on the Waymo Open Dataset, 2.01 dB above the strongest evaluated baseline. *Instant NuRec* is deeply integrated into NuRec and is compatible with AlpaSim [3] for closed-loop simulation.

[†]The full list of contributors is provided in Appendix A.

1. Introduction

Closed-loop simulation has become a central tool for developing and validating autonomous driving (AD) stacks: it allows planners and perception modules to be tested against rare, safety-critical, or otherwise hard-to-collect interactions in a controllable environment. A faithful simulator must combine three ingredients: *(i)* photorealistic rendering of the surrounding world, *(ii)* an explicit, physics-simulation-ready decomposition of the scene into a static background and individually editable dynamic agents, and *(iii)* the ability to freely re-pose the ego camera and re-time the scene so that counterfactual rollouts can be evaluated. Recent advances in 3D Gaussian Splatting (3DGS) [4] have made high-quality rendering computationally inexpensive, while per-scene optimization frameworks such as OmniRe [5] have shown that the resulting reconstructions can faithfully reproduce a real driving log. However, per-scene optimization is fundamentally a bottleneck to scaling: each new clip requires hours of computation, careful hyperparameter tuning, and a large set of auxiliary inputs (LiDAR sweeps, fitted poses, and semantic masks). This cost makes it impractical to ingest the millions of clips that modern AD fleets collect every day.

In this paper, we introduce *Instant NuRec*, a feed-forward neural reconstruction model that turns a short multi-view driving log into a fully simulatable 3DGS world in a single forward pass. The input is a sequence of images captured by a rig with 1–5 calibrated cameras. A single forward pass through *Instant NuRec* produces *(i)* a static 3DGS layer that captures roads, buildings, and other rigid background elements; *(ii)* a dynamic 3DGS layer with piecewise-linear trajectories that represents moving vehicles and vulnerable road users; and *(iii)* a cubemap that models the distant sky. The output can be exported in USDZ format and consumed directly by closed-loop simulation frameworks such as AlpaSim [3], allowing a change in clip selection to be reflected in the simulator within seconds rather than hours.

Instant NuRec couples a single shared encoder with a set of lightweight task-specific decoders. The encoder is an alternating-attention Vision Transformer [6] that interleaves intra-image and cross-image attention to fuse information across the input cameras and over time. Its shared features drive several decoders that predict dense depth, normals, semantics, a sky cubemap, and per-camera color corrections. The predicted depth lifts a set of query points into world space to anchor the Gaussians, the semantic head separates the static background from dynamic agents, and a Gaussian head supplies the remaining shape and appearance attributes needed to complete the layered output. Crucially, a motion head directly recovers piecewise-linear actor trajectories for the query points, so the dynamic layer requires no per-actor cuboid tracks at reconstruction time. We train the model in three stages that deliberately decouple geometry supervision from rendering supervision, keeping large-scale training on $\sim$40K internal driving clips efficient.

In short, *Instant NuRec* turns a multi-camera driving log into a complete 3DGS world with static, dynamic, and sky components in a single feed-forward pass, ready for direct use in closed-loop simulation. Across novel-view synthesis, geometry, and downstream simulator integration, it closes a large fraction of the quality gap to per-scene optimization while requiring 10^3–$10^4\times$ less computation per clip, making large-scale reconstruction of driving logs practical. Our code is available at `https://github.com/nvidia/instant-nurec`, with additional documentation at `https://docs.nvidia.com/nurec/index.html`.

2. Related Work

Per-scene neural reconstruction for driving. Neural Radiance Fields (NeRFs) [7] and 3DGS [4] have set the standard for photorealistic novel-view synthesis, and a substantial body of follow-up work has specialized them to outdoor driving scenes. Early urban radiance fields scaled reconstruction to street and city scales using posed RGB and LiDAR [8, 9], while Neural Scene Graphs [10] introduced the now-standard decomposition of a driving scene into a static background and a set of rigidly moving foreground actors. Later methods pushed this toward self-supervised dynamics and sensor-realistic simulation [11, 12] and replaced the volumetric field with

explicit Gaussians for faster, editable reconstruction [13, 14, 15]. OmniRe [5] extended this decomposition to non-rigid pedestrians and cyclists, while 3DGRT [16] and 3DGUT [2] introduced Gaussian renderers that natively support distorted cameras and secondary rays, removing the rectified-pinhole assumption. These methods produce extremely high-quality reconstructions but require minutes to hours of per-clip optimization and a rich set of auxiliary inputs (LiDAR, multi-traversal poses, semantic masks, and cuboid tracks). *Instant NuRec* preserves the same output format, namely a layered 3DGS world consumable by NuRec [1, 17] and AlpaSim [3], but amortizes the optimization cost into a single feed-forward pass.

Feed-forward 3D reconstruction. Feed-forward 3D models have rapidly progressed from single-object generation [18, 19, 20, 21, 22, 23, 24, 25] to scene-level prediction. At the scene level, generalizable Gaussian models predict per-pixel Gaussians from sparse posed views in a single pass using probabilistic depth [26], plane-sweep cost volumes [27], large transformer decoders [28], or monocular depth priors [29]. In parallel, another line of work regresses pointmaps or Gaussians in a pose-free manner directly from uncalibrated images [30, 31, 32]. This paradigm has recently been specialized to dynamic driving scenes: STORM [33] and DrivingForward [34] predict static and dynamic Gaussians from surround-view video, and a wave of contemporaneous models extends this to pose-free, tracker-free, and off-road settings [35, 36, 37, 38]. A separate line of feed-forward 3DGS work decouples Gaussian prediction from input pixels: instead of emitting one Gaussian per pixel, TokenGS [39] regresses Gaussians from a fixed set of learnable tokens, and related query- or scene-token formulations [40, 41] similarly unbind the primitive count from image resolution and view count. Unlike these reconstruction-focused models, which target bare RGB Gaussians, *Instant NuRec* emits a complete, simulation-ready layered 3D world.

Driving world models. A complementary and rapidly growing line of work learns generative models that synthesize plausible driving observations conditioned on the past and on control signals such as ego trajectory, layout, or text. Early latent-diffusion models [42, 43, 44, 45] generate controllable single- or multi-view street video. Subsequent work improves fidelity and controllability, extends the prediction horizon [46, 47, 48, 49, 50], adopts autoregressive video-GPT or diffusion backbones for long-range rollouts [51, 52], and culminates in large world foundation models trained across driving and other embodiments [53, 54]. These models are powerful content generators but, by construction, emit 2D pixels rather than an explicit 3D state: they cannot be freely re-posed by a planner and do not directly yield a simulatable scene. A second line therefore uses generative video priors to *repair or densify* neural reconstructions, synthesizing novel-trajectory frames offline as a data machine [55, 56] or removing ghosting and floater artifacts online as an inference-time enhancer [57, 58, 59]. Intermediate variants are conditioned on point clouds or pseudo-renders [60, 61]. Most recently, joint formulations such as the Xiaomi EV world model [62] tightly couple a feed-forward reconstructor with a video generator, allowing geometry to anchor generation while the generator fills unobserved regions. *Instant NuRec* occupies the reconstruction side of this framework: it does not invent content but emits an explicit, navigable 3D world in a single pass, making it complementary to generative world models and a natural, fast 3D backbone for the generative-repair and joint-model lines.

3. Method

At a high level, *Instant NuRec* takes a short multi-view driving log as input and produces a layered 3DGS world in a single feed-forward pass. We first describe the problem formulation (input/output representation) in § 3.1, then the shared encoder in § 3.2, and finally the decoder heads that produce the renderable representation in § 3.3. A schematic of the pipeline is shown in Fig. 2.

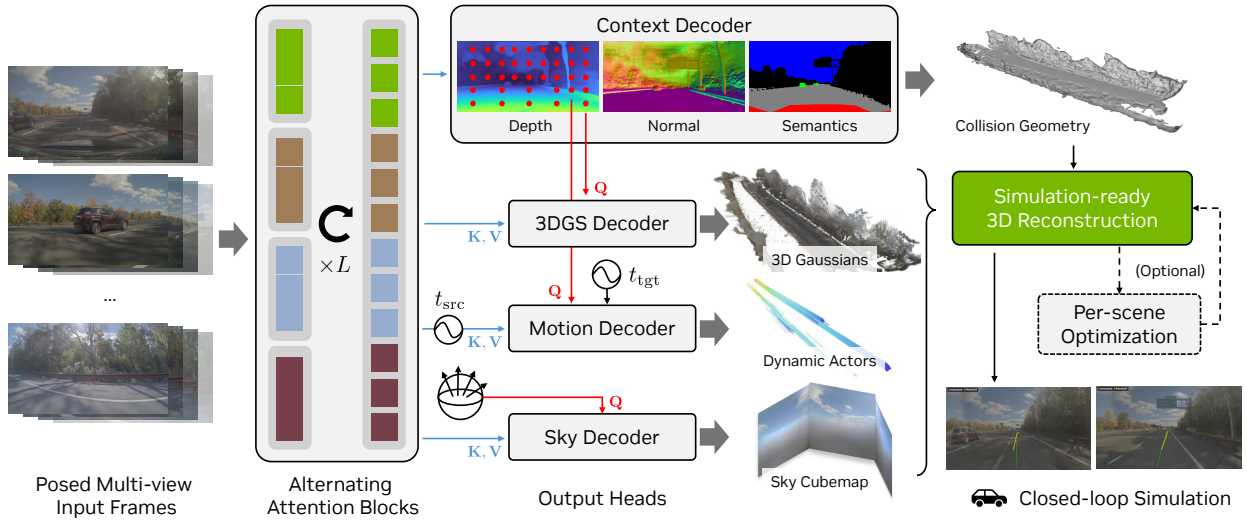

Figure 2: **Pipeline overview.** Multi-view driving images are tokenized into patches and processed by an alternating-attention ViT encoder. Several decoder heads share the resulting latent features and produce depth maps, semantic labels, motion estimates, a sky cubemap, and 3DGS attributes. Optionally, the output can be further optimized on a per-scene basis and used for downstream simulation tasks.

3.1. Problem Formulation

Input. The model accepts $V \times T$ RGB images, where V is the number of available cameras and T is the number of temporal frames sampled from the source camera videos at 2–4 Hz. Each image $\mathbf{I}_{v,t}$ is paired with its 6-DoF pose $\mathbf{T}_{v,t} \in SE(3)$ and a vector of camera-intrinsic parameters $\boldsymbol{\kappa}_{v,t}$. Optional cuboid tracks for dynamic actors can be supplied at inference time to further calibrate the trajectories of the dynamic Gaussians.

Output. *Instant NuRec* emits a layered representation:

- **Static layer** $\mathcal{G}^s$, a set of N_s static Gaussians, each parameterized by attributes $(\boldsymbol{\mu}_i, \boldsymbol{s}_i, \mathbf{q}_i, \alpha_i, c_i, \boldsymbol{n}_i, \ell_i)$, with world-space position $\boldsymbol{\mu} \in \mathbb{R}^3$, scale $\boldsymbol{s} \in \mathbb{R}^3$, rotation quaternion $\mathbf{q}$, opacity $\alpha \in [0,1]$, color $c \in \mathbb{R}^3$, world-space normal $\boldsymbol{n} \in \mathbb{S}^2$, and a semantic class ℓ (road/non-road) used by downstream simulators. The quaternion primarily parameterizes the Gaussian's rotation for view rendering, whereas the normal is used primarily to extract surfaces for simulation.
- **Dynamic layer** $\mathcal{G}^d$, a set of N_d dynamic Gaussians with the same per-Gaussian attributes as the static layer, except that the position $\boldsymbol{\mu}_i$ is replaced by a piecewise-linear trajectory $\boldsymbol{\mu}(t)$ defined by knots $\{(t_k, \boldsymbol{\mu}_k)\}_{k=1}^{3}$. For $t \in [t_k, t_{k+1}]$, the trajectory is evaluated as $\boldsymbol{\mu}(t) = (1-\lambda)\,\boldsymbol{\mu}_k + \lambda\,\boldsymbol{\mu}_{k+1}$ for $\lambda = (t-t_k)/(t_{k+1}-t_k)$. The opacity is gated by a smooth fade $f(x) = \exp(-x^{10})$ around t_1 and t_3.
- **Sky cubemap** $\mathbf{I}^{\text{sky}} \in \mathbb{R}^{6 \times H \times W \times 3}$, which represents the distant background.
- **ISP affine transforms** $\{\mathbf{A}_v\} \in \mathbb{R}^{V \times 3 \times 4}$, applied to rendered colors before the photometric loss to absorb residual per-camera color/exposure mismatch.

3.2. Shared Alternating-Attention Encoder

We largely follow the backbone design of Depth-Anything-3 [63], which repurposes the DINOv2 [64] architecture for multi-view inputs. Compared with other geometry backbones, it produces sharp and accurate depth maps without requiring an additional image tokenizer.

In our model, the encoder ϕ jointly tokenizes all input images and produces a shared latent $\mathbf{F} \in \mathbb{R}^{N \times C}$,

where N is the total number of tokens and C is the feature dimension. Each image is split into non-overlapping 14×14 patches, each of which is linearly projected to a token of dimension C. A per-image *class token*, derived from the camera's 6-DoF pose and field of view using sinusoidal positional encoding, is prepended to provide a global summary of the view and inject pose information directly into attention. The combined sequence is processed by a stack of L alternating attention blocks that interleave *intra-image* self-attention and *cross-image* self-attention. This interleaving design enables effective information exchange between patches in different views while maintaining computational efficiency in layers that apply intra-image attention.

3.3. Decoding Heads

Several decoding heads operate on the shared latent features to produce the final output. These decoders are lightweight, so adding an output head incurs little additional cost at inference.

Context-view decoders. We use *context views* to denote the input images observed by the model, as opposed to held-out target views used for rendering supervision and evaluation. The context-view decoders produce dense, pixel-aligned scene attributes for these input views. We use a set of DPT [65] fusion heads to fuse latent features from multiple layers into attribute maps. Specifically, we attach three DPT heads: one each for depth, normals, and semantic logits. The depth branch follows the Depth-Anything-3 model, but we remove the confidence channel because reconstruction tasks generally require dense predictions over the full image. For the semantic logits, we predict four classes: road, movable (vehicles and pedestrians), sky, and ego-car.

3DGS decoder. Most existing feed-forward reconstruction methods emit one Gaussian at every pixel. While this usually provides strong reconstruction quality, the number of Gaussians can be large and their pixel-aligned geometry may not reflect the underlying scene structure. In our design, we decouple the Gaussians from the latent features by selecting a set of query points from the lifted depth map. Each query point in $\mathbf{Q}$ is a 3D position lifted from the predicted depth map and embedded by a lightweight learned projection. The query points cross-attend to keys $\mathbf{K}$ and values $\mathbf{V}$ derived from the latent feature $\mathbf{F}$, and the decoder outputs the per-query Gaussian attributes, including scale, rotation, and opacity. After decoding, the Gaussians from all queries are combined and can then be rendered by 3DGUT [2].

Motion decoder. The motion decoder takes the same set of query points $\mathbf{Q}$ as input and predicts 3D displacements to the target timestamps t_{tgt}. For computational efficiency, we jointly predict two 3D displacements, one for each target timestamp, yielding a 6-D regression target. The motion decoder combines elements of V-DPM [66] and 4RC [67]: a shallow Transformer operates on the encoder features and is modulated by the source timestamps t_{src}. The Transformer's LayerNorms are AdaLN modules [68] conditioned on the target timestamps t_{tgt}, informing the head which source-to-target motion $t_{\text{src}} \to t_{\text{tgt}}$ to predict. For each source timestamp t_{src}, we select the immediately preceding and following timestamps from the same camera as targets. Given displacement predictions (Δ_-, Δ_+), the three final 3D positions of the dynamic layer are $\boldsymbol{\mu}_1 = \boldsymbol{\mu} + \Delta_-$, $\boldsymbol{\mu}_2 = \boldsymbol{\mu}$, and $\boldsymbol{\mu}_3 = \boldsymbol{\mu} + \Delta_+$, where $\boldsymbol{\mu}$ is the 3D position lifted from the depth map. Unlike STORM [33], in which all Gaussians are considered dynamic, we use the same query set throughout and subsequently apply a semantic-head mask, assigning only points classified as *movable* to the dynamic layer and the remainder to the static layer.

Query point selection. Query points anchor the predicted Gaussians in space, so their selection strongly affects reconstruction quality. We consider two strategies: (i) **Dense**, where every pixel in the context images is a query point; and (ii) **Selective**, which uses strided sampling and cross-attention to sparsify queries and includes a dedicated *road branch* for road-dominated regions. In Selective, query points, each associated with full-resolution per-pixel signals, are sampled with stride s and grouped into $p\times p$ token clusters. Each query token cross-attends to the latent feature map (with QK normalization and layer scale for stability) and predicts a fixed cluster of $M=p^2$ Gaussians, whose attributes are gathered from the corresponding group of

signals. For example, with $s=2$ and $p=8$, a single 448×784 context view yields $1{,}372$ tokens and roughly 88K Gaussians with the **Selective** strategy. This represents a $4\times$ reduction from the ~351K produced by **Dense** while empirically retaining most of the reconstruction quality.

Sky decoder. The sky decoder maps the shared latent $\mathbf{F}$ to the distant-background cubemap $\mathbf{I}^{\text{sky}}$. For each cubemap pixel, we form a query by sinusoidally encoding its world-space ray direction. The query cross-attends to encoder features augmented with per-camera ray embeddings so that each pixel can preferentially borrow color from the input view that observed a similar direction. A DPT-style fusion head upsamples the cross-attention output to full cubemap resolution.

ISP decoder. This decoder predicts a per-camera 3×4 affine RGB transform, analogous to appearance compensation in neural rendering [69], that absorbs residual differences in white balance, gamma, and vignetting between cameras observing the same scene. A learned token cross-attends to the deepest encoder features, grouped by camera index, and a final linear projection emits the affine matrix and bias. The transform is initialized to the identity and applied to rendered colors before the photometric loss is computed. At reconstruction time, downstream simulators may either apply the transform to match the training rig's color response or omit it to render in the canonical color space.

Forward-pass summary. Putting the pieces together, the encoder ϕ maps the input clip to a shared latent representation $\mathbf{F}$, from which the context DPT heads predict dense depth, normals, and semantics. Query points $\mathbf{Q}$ are lifted from the depth map using either the Dense or Selective strategy. The 3DGS decoder cross-attends $\mathbf{Q}$ to $\mathbf{F}$ and assembles static Gaussians $\mathcal{G}^s$ with scale, rotation, opacity, color, normal, and semantic labels. The motion decoder uses the same queries $\mathbf{Q}$ to predict displacements; only queries subsequently masked as *movable* are assigned to dynamic Gaussians $\mathcal{G}^d$ with piecewise-linear trajectories. The sky and ISP heads produce $\mathbf{I}^{\text{sky}}$ and per-camera affine transforms $\{\mathbf{A}_v\}$ in parallel; no iterative optimization is performed at inference.

3.4. Long-clip Processing via Chunk Merging

Because the model is trained on short temporal windows for memory and computational efficiency, reconstructing an entire driving log requires processing it in overlapping chunks and merging the per-chunk predictions into a single global field.

Per-chunk pruning. Within each chunk, we drop Gaussians whose opacity is below a small threshold ($\alpha<10^{-2}$) and transform the surviving primitives into a global world frame using the chunk-level pose. In practice, this can reduce the number of Gaussians while maintaining rendering quality.

Frustum-ownership pruning. A Gaussian predicted in one chunk, particularly at a great distance from the chunk's cameras, often lands inside another chunk's frustum, where it competes with that chunk's own closer, better-conditioned Gaussians and creates floaters. We therefore apply a *frustum-ownership* test: a Gaussian g from chunk i is kept only if no other chunk j observes it from substantially closer range. Specifically, define D_{gi} as the Euclidean distance from the Gaussian's center to the optical center of its closest camera in chunk i, with $D_{gi}=\infty$ if g lies outside chunk i's frustum pyramid. We retain g from chunk i if and only if

$$D_{gi} \;\le\; \min_{j} D_{gj} + \delta \tag{1}$$

where δ is a small tolerance that keeps Gaussians on the boundary between chunks. Intuitively, each Gaussian is "owned" by the chunk that observed it at the closest range; downstream chunks see only their own near-field reconstruction and inherit the far field from their closer neighbors. Because dynamic Gaussians are active only over a short time window and rarely overlap with another chunk's active interval, we leave the dynamic layer untouched and apply the pruning to the static layer only.

4. Training

We describe the training objective and curriculum in § 4.1 and the implementation details in § 4.2.

4.1. Losses and Training Curriculum

We train the model with three groups of objectives:

$$\mathcal{L} = \mathcal{L}_{\text{context}} + \mathcal{L}_{\text{motion}} + \mathcal{L}_{\text{render}}, \tag{2}$$

where

- $\mathcal{L}_{\text{context}}$ directly supervises the outputs of the context decoders. Attributes such as depth, normals, and semantic logits are pixel-aligned and have corresponding ground truth. We supervise these attributes with L_1, cosine similarity, and cross-entropy losses, respectively.
- $\mathcal{L}_{\text{motion}}$ includes forward and backward 3D-flow losses, supervised by 3D scene flow derived from cuboid tracks for foreground pixels and a zero-displacement target for background pixels.
- $\mathcal{L}_{\text{render}}$ comprises differentiable rendering losses on novel, held-out training views sampled uniformly within the context time window. The loss consists of an L_1 term and an LPIPS term [70], computed after applying the predicted camera-ISP affine transform. This rendering loss also backpropagates to the sky cubemap decoder. We additionally use an opacity loss to encourage transparency in sky regions so that the sky is represented primarily by the cubemap rather than the 3D Gaussians.

In practice, introducing $\mathcal{L}_{\text{render}}$ at the beginning of training slows convergence. Moreover, rendering at every iteration is too expensive because of the large number of Gaussians. We therefore divide the full training process into a three-stage curriculum:

- **Stage 1: Pretraining.** We reproduced the Depth-Anything-3 pretraining protocol using the depth- and ray-prediction tasks from [63] and the data-curation pipeline described in [71]. This stage provides a general initialization for the full backbone.
- **Stage 2: Context training.** We enable the sky, depth-and-context, and motion decoders and fully fine-tune the encoder with these heads on the internally curated driving corpus with $\mathcal{L}_{\text{context}} + \mathcal{L}_{\text{motion}}$; the context heads receive dense, well-conditioned supervision from LiDAR and auto-labels.
- **Stage 3: GS training.** The encoder is frozen, and the camera-ISP and GS decoders are enabled. Because no ground-truth Gaussian attributes are available, we train these heads only after the geometry has stabilized, allowing rendering gradients to backpropagate through a sensible 3D scaffold. The Gaussians produced by the now-frozen geometry stack are rendered through 3DGUT [2] and additionally supervised with $\mathcal{L}_{\text{render}}$.

Freezing the encoder in Stage 3 is a deliberate choice: it prevents high-variance rendering gradients from corrupting the well-conditioned geometry features learned in Stage 2 and substantially stabilizes training while the GS attributes are still random.

4.2. Implementation Details

Data curation and auto-labeling. *Instant NuRec* is trained on ∼40K filtered driving clips drawn from NVIDIA's Autonomous Vehicle (AV) data platform. Each clip contains footage from up to five cameras, with ∼300–600 frames per camera at 30 Hz, along with the corresponding camera calibrations and rig-trajectory data. Each clip also includes LiDAR point-cloud data, from which we export depth-map ground truth. In addition to the raw sensor data and calibrations, we apply the following steps to obtain auxiliary training data:

- **Semantic segmentation** is obtained by independently applying a state-of-the-art semantic segmentation model [1] to every image from each camera.
- **3D bounding boxes** for movable actors are provided by an internal cuboid auto-labeler and tracker using LiDAR data.
- **Dense depth** is obtained by accumulating LiDAR points across the clip and projecting them into each image; missing pixels (sky, ego-car, and occluded surfaces) are masked out of the depth loss. Regions belonging to dynamic actors are accumulated according to their auto-labeled bounding boxes.

Training infrastructure. *Instant NuRec* is implemented in PyTorch, with 3DGUT [2] serving as the differentiable renderer. Training uses standard DDP across nodes with an effective per-GPU batch size of 1–2 clips. We train the model with several common camera configurations $V \in \{1, 3, 5\}$ (front only; front, front-left, and front-right; or front, front-left, front-right, rear-left, and rear-right) and varying temporal frame counts $T \in \{8, 12, 18\}$. The learning rate follows a cosine annealing schedule with a linear warmup to a maximum of 10^{-4}. The complete three-stage training process takes $\sim$6 days on 8 nodes.

5. Experiments

We evaluate *Instant NuRec* along three axes: *(i)* novel-view synthesis quality against recent feed-forward baselines on the public Waymo Open Dataset (§ 5.1); *(ii)* reconstruction quality and closed-loop simulation viability on our large-scale internal corpus (§ 5.2); and *(iii)* ablations of the main design choices (§ 5.3).

5.1. Comparison on the Waymo Open Dataset

For reproducibility and direct comparison with recent feed-forward reconstruction methods, we evaluate on the validation split of the Waymo Open Dataset [72]. Our evaluation follows the protocol used by STORM [33]. Each evaluation sample is a 2-second window containing 20 consecutive frames at 10 Hz. The model observes four context frames spaced 0.5 s apart and is evaluated on the held-out target frames. All RGB metrics are computed after resizing the predictions and ground truth to the STORM resolution of 240×160 pixels.

Baselines. We compare against representative feed-forward reconstruction methods: DepthSplat [29], STORM [33], Depth-Anything-3 [63], and DGGT [35]. We evaluate the baseline methods using their released checkpoints without further fine-tuning. Because DGGT does not take poses as input, we align its predicted trajectory with the ground-truth trajectory before computing the metrics.

Metrics. For image quality, we report PSNR and SSIM on both the full image and the dynamic region. We define the dynamic region using semantic segmentation masks that select vehicle and movable-object pixels, and we use the same mask for all methods. For geometry, we follow the sparse-LiDAR depth evaluation protocol used in StreetForward [37]. We project Waymo LiDAR points into the front camera and evaluate only pixels with valid sparse depth. Because several feed-forward baselines predict depth up to an arbitrary scale, we align each prediction with the LiDAR depth using a per-frame least-squares scale before reporting AbsRel and δ_1 for all valid LiDAR pixels and for the subset in dynamic regions.

As shown quantitatively in Tab. 1 and qualitatively in Fig. 3, our model achieves the best RGB reconstruction among the evaluated feed-forward baselines, with even larger margins in dynamic regions. Its depth quality is on par with that of the strongest baselines, if not better on some metrics. We attribute the improvements mainly to our choice of efficient backbones and more effective training strategies.

Figure 3: **Qualitative comparison on the Waymo Open Dataset.** Novel-view synthesis results for held-out target frames from the validation set. Compared with recent feed-forward baselines, *Instant NuRec* produces sharper imagery and better-preserved thin structures.

Table 1: Image and depth quality comparison on the Waymo Open Dataset. Higher PSNR/SSIM/δ_1 and lower AbsRel are better.

Method	Full Image				Dynamic Region			
	PSNR ↑	SSIM ↑	AbsRel ↓	δ_1 ↑	PSNR ↑	SSIM ↑	AbsRel ↓	δ_1 ↑
DepthSplat [29]	22.48	0.645	0.295	0.592	18.59	0.391	0.445	0.512
STORM [33]	21.88	0.752	0.123	0.870	19.89	0.556	0.178	0.789
Depth-Anything-3 [63]	20.30	0.557	0.434	0.313	17.59	0.250	0.467	0.354
DGGT [35]	26.25	0.805	0.135	0.841	21.76	0.652	0.249	0.689
Ours	**28.26**	**0.859**	**0.076**	**0.937**	**24.93**	**0.793**	**0.085**	**0.922**

5.2. Comparisons and Applications on Internal Data

Public driving datasets are relatively small and cover only a limited range of conditions. To demonstrate real-world applicability, we evaluate on a held-out validation split of our internal corpus and assess reconstruction quality and simulator integration in settings that public benchmarks rarely capture, including non-pinhole cameras, longer sequences (up to 30 s), and complex driving scenarios. Throughout this section, our baseline is NVIDIA NuRec [1, 17], a per-scene-optimized 3DGS reconstruction that requires roughly 75 min of optimization per scene and serves as our performance reference for production-quality reconstruction.

Reconstruction quality. Fig. 4 shows single-camera reconstructions in which re-posed and bird's-eye views remain geometrically consistent and preserve thin structures across diverse scenes. *Instant NuRec* also reconstructs the full multi-camera surround view in a single forward pass: Fig. 5 shows the result across all five cameras (front, front-left, front-right, rear-left, rear-right), with consistent appearance and geometry across the rig in scenes ranging from tunnels to dense urban streets.

As described in § 3, *Instant NuRec* provides two query-point strategies, **Dense** and **Selective**, that trade Gaussian budget for reconstruction quality. On our validation set, the Selective strategy cuts the Gaussian budget by roughly 3×—from ∼351K per context view for Dense to ∼120K (∼6.3M to ∼2.2M per 18-frame

scene)—with only marginal losses in image and depth accuracy, as shown in Fig. 6. Fig. 7 further compares both variants with NuRec: both reconstruct a scene in ∼1.5 s versus ∼75 min for NuRec, a speedup of more than three orders of magnitude, while trailing only modestly in appearance quality.

We further evaluate reconstruction quality by applying an in-house 3D detection pipeline to the reconstructed scenes. We render each scene from novel viewpoints and compare the resulting detection precision and recall with those obtained from ground-truth images captured by the ego cameras. This evaluation checks that reconstruction inaccuracies do not bias downstream policies. Dense remains the stronger reconstruction variant, while Selective incurs a modest reduction in appearance and detection metrics while using roughly one-third of the Gaussian budget. Selective is therefore useful when memory or rendering cost is the primary constraint, while Dense provides the highest-quality feed-forward reconstruction.

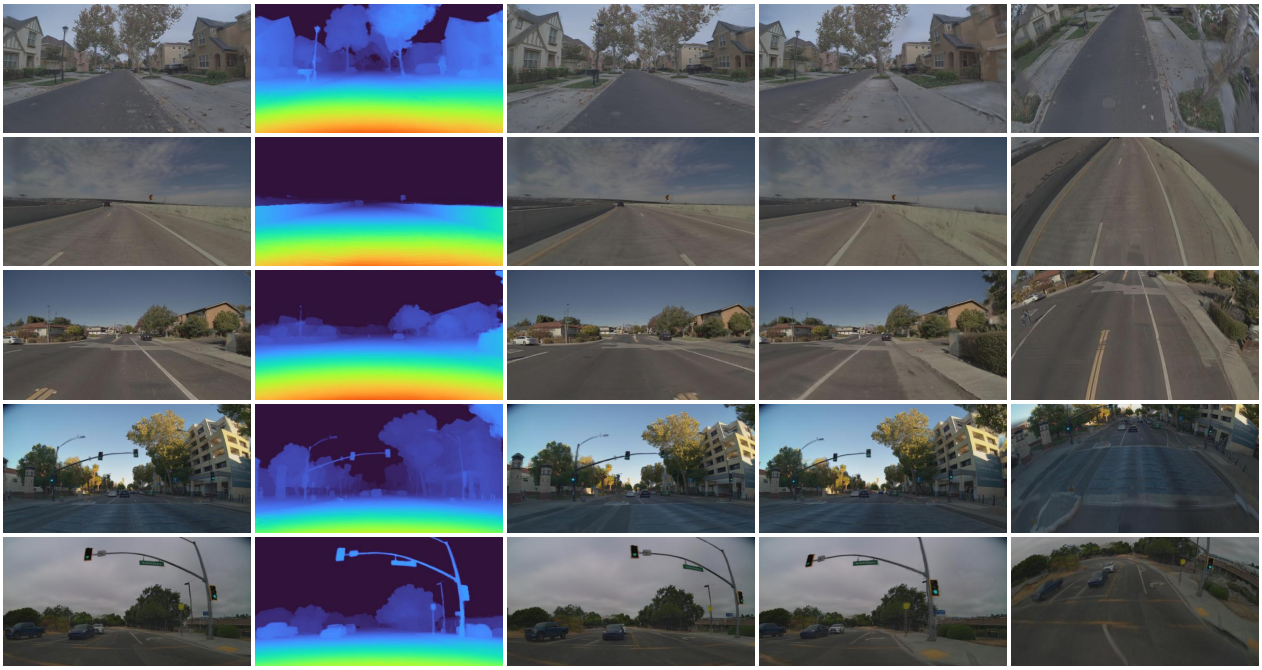

Figure 4: **Single-camera reconstruction.** From a single front-camera input, *Instant NuRec* reconstructs a navigable 3DGS scene in one forward pass. The re-posed and top-down renderings remain geometrically consistent and preserve thin structures such as poles and traffic lights.

Closed-loop simulation viability. A key question for any feed-forward reconstruction method is whether the reconstruction produced by a single forward pass is sufficient for reliable closed-loop policy simulation. To test this, we compare *Instant NuRec* with NuRec by running the same set of policies through the same AlpaSim [3] closed-loop stack and changing only the reconstructions. Our evaluation set contains 140 scenes. Each scene is rolled out for 20 s in closed loop, with policy replanning every 500 ms and six independent trials per scene. We simulate five driving policy configurations: VaVAM [73], Alpamayo R1 [74] (A-R1), and the 1-camera, 2-camera, and 4-camera variants of Alpamayo 1.5 [74] (A-1.5). We evaluate the policies using the average of two key metrics: *Collision (Front + Lateral + Rear)* and *Offroad*. As shown in Fig. 8, the policy ranking under *Instant NuRec* is identical to that under NuRec, meaning that policy selection decisions made with *Instant NuRec* would match those made with the more expensive per-scene reconstruction. This makes the method a reliable substitute for per-scene optimization in closed-loop policy evaluation, at a fraction of the reconstruction cost.

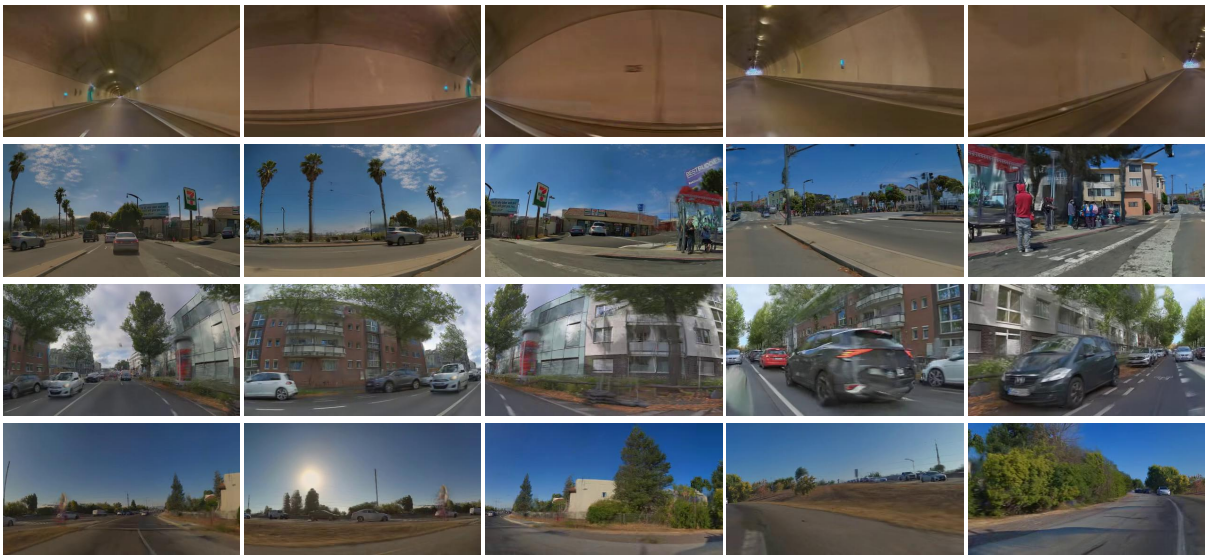

Figure 5: **Multi-camera surround-view reconstruction.** From a five-camera input rig, *Instant NuRec* jointly reconstructs the full surround view. Each row shows a different scene, and the rendered views remain consistent in appearance and geometry across overlapping cameras.

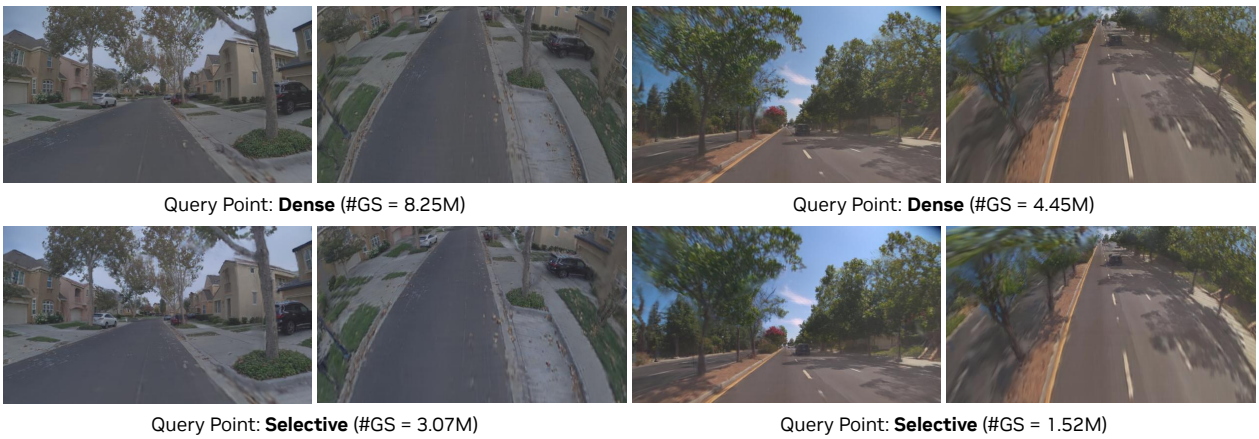

Figure 6: **Query-point selection.** Comparison of the **Dense** and **Selective** query-point strategies. The Selective strategy produces far fewer Gaussians (#GS) while maintaining comparable reconstruction quality.

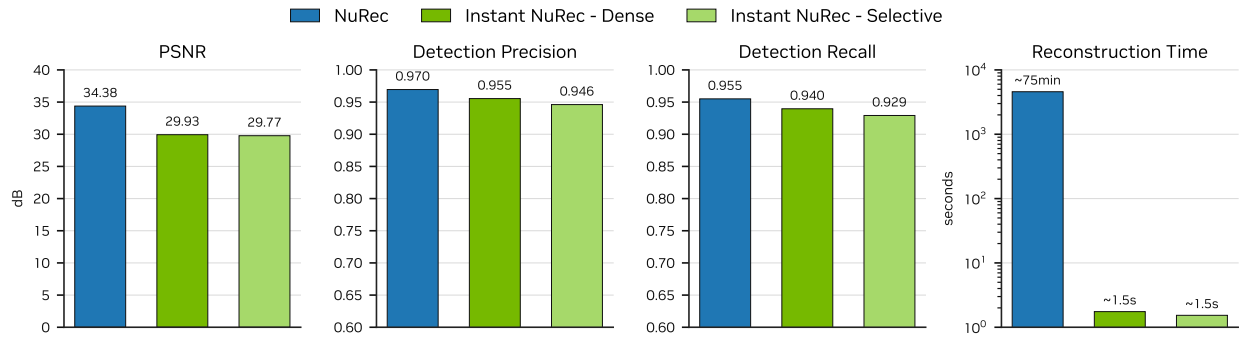

Figure 7: **Quantitative reconstruction comparison on internal data.** *Instant NuRec* (Dense and Selective) versus the per-scene-optimized NuRec baseline on PSNR, downstream 3D detection precision and recall, and reconstruction time. *Instant NuRec* reaches comparable appearance and detection quality while reconstructing a scene in ∼1.5 s instead of ∼75 min.

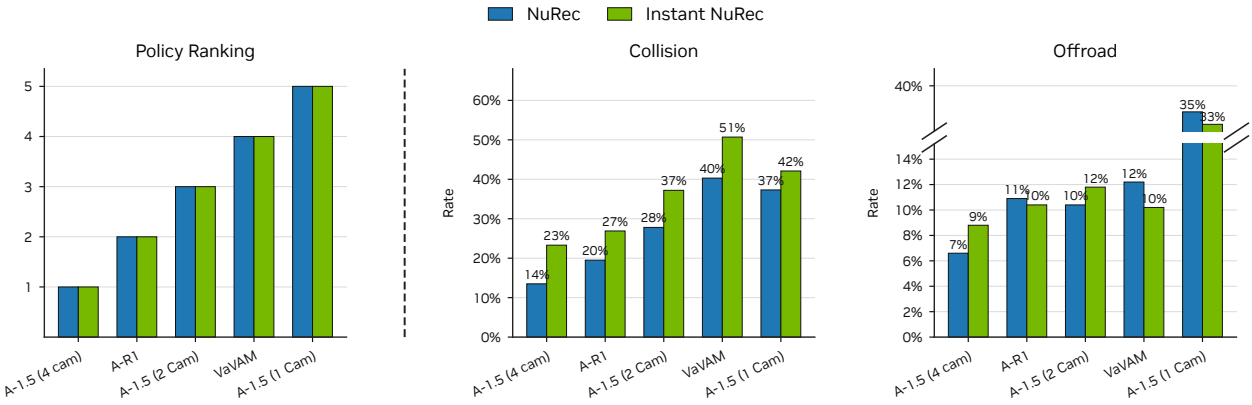

Figure 8: **Closed-loop comparison between NuRec and *Instant NuRec* across five policy configurations.** Each bar pair compares the same policy under NuRec (blue) versus *Instant NuRec* (green), averaged over 140 scenes × 6 trials. Despite requiring only a single forward pass, *Instant NuRec* reproduces the same policy ranking as NuRec, validating it as a reliable simulator for policy comparison.

5.3. Ablation Study

We ablate the main design choices of *Instant NuRec* on the validation set introduced in § 5.2. We summarize the results quantitatively in Tab. 2 and qualitatively in Fig. 9; each variant removes one component to show its individual effect on reconstruction quality.

Training strategy. We implement a single-stage variant that trains Stage 2 and Stage 3 jointly rather than following our sequential schedule. This variant takes longer to train and requires more iterations to converge, possibly because of the additional complexity introduced by gradients from the renderer. Nevertheless, its final metrics remain worse (Tab. 2), so we retain the multi-stage schedule.

Sky model. We replace the sky cubemap branch with an MLP sky model similar to STORM [33]. Because of the network's low-frequency inductive bias, this variant fails to recover high-frequency sky textures such as clouds and the sun (Fig. 9).

Loss terms. We ablate the two auxiliary losses in turn. Removing the LPIPS loss makes the reconstructions noticeably blurrier, while removing the depth loss causes the model to overfit to appearance and degrade

Figure 9: **Qualitative ablation.** Effects of the sky cubemap branch, LPIPS loss, and frustum-ownership merging strategy. In each comparison, the top row omits the named component, while the bottom row shows the full model.

geometry, as reflected in the quantitative depth metrics.

Primitive merging strategy. We replace frustum-ownership merging with a naive concatenation of Gaussians without pruning. Without this pruning, far-field Gaussians from the outer region of one chunk can appear as floaters inside neighboring chunks, where they compete with better-conditioned local primitives and degrade quality, especially for longer sequences.

Table 2: **Ablations on internal data.** Reconstruction quality when each component is removed.

Variant	PSNR ↑	SSIM ↑	AbsRel ↓	δ_1 ↑
Instant NuRec (full)	**29.93**	**0.793**	0.069	0.950
single-stage training	27.65	0.751	0.077	**0.955**
sky MLP instead of cubemap	26.73	0.692	**0.068**	0.950
w/o LPIPS loss	26.81	0.705	0.073	0.934
w/o depth loss	28.92	0.782	0.103	0.862
w/o frustum-ownership merging	26.10	0.648	0.098	0.891

5.4. Extension to LiDAR Reconstruction

Beyond image reconstruction, we examine whether *Instant NuRec* can be extended to the LiDAR modality with only minimal changes.

Model adaptation. We retain the pretrained *Instant NuRec* backbone and add a lightweight LiDAR branch that mirrors the camera path and predicts Gaussian primitives in a single forward pass. Each LiDAR sweep is represented as a *range map* with per-beam range and intensity; a range-map encoder produces tokens, and a decoder maps them to M Gaussian primitives for each valid input ray. These primitives are instantiated only for valid returns and are anchored near the corresponding measurements with bounded 3D offsets; the branch also predicts their scales, orientations, opacities, and intensities. To respect the sparse angular sampling of LiDAR, we impose an anisotropic scale floor inspired by Mip-Splatting [75]. Specifically, at range r, each primitive's lateral footprint is lower-bounded by the local inter-beam spacing induced by the horizontal and vertical angular resolutions. The branch is trained by rendering range and intensity for supervision frames captured from different poses, using an L_1 range loss and an MSE intensity loss.

Dataset and evaluation. We evaluate on a custom split of our internal data collected with a Pandar128 LiDAR. Ground truth consists of the valid returns in the recorded LiDAR point cloud. Geometry quality is measured by the Chamfer distance between the predicted and ground-truth geometry. We also report the per-return intensity MAE on rays for which both the filtered prediction and ground truth contain a return.

Table 3: LiDAR reconstruction benchmark on internal data.

Method	CD ↓	Prec. ↓	Cov. ↓	Int. MAE ↓	Recon. time ↓	Speedup ↑
NuRec [1]	0.204	0.080	0.124	0.028	∼75 min	1×
Instant NuRec LiDAR	0.286	0.113	0.173	0.027	∼20 s	∼225×
w/o scale floor	0.936	0.195	0.741	0.036	–	–

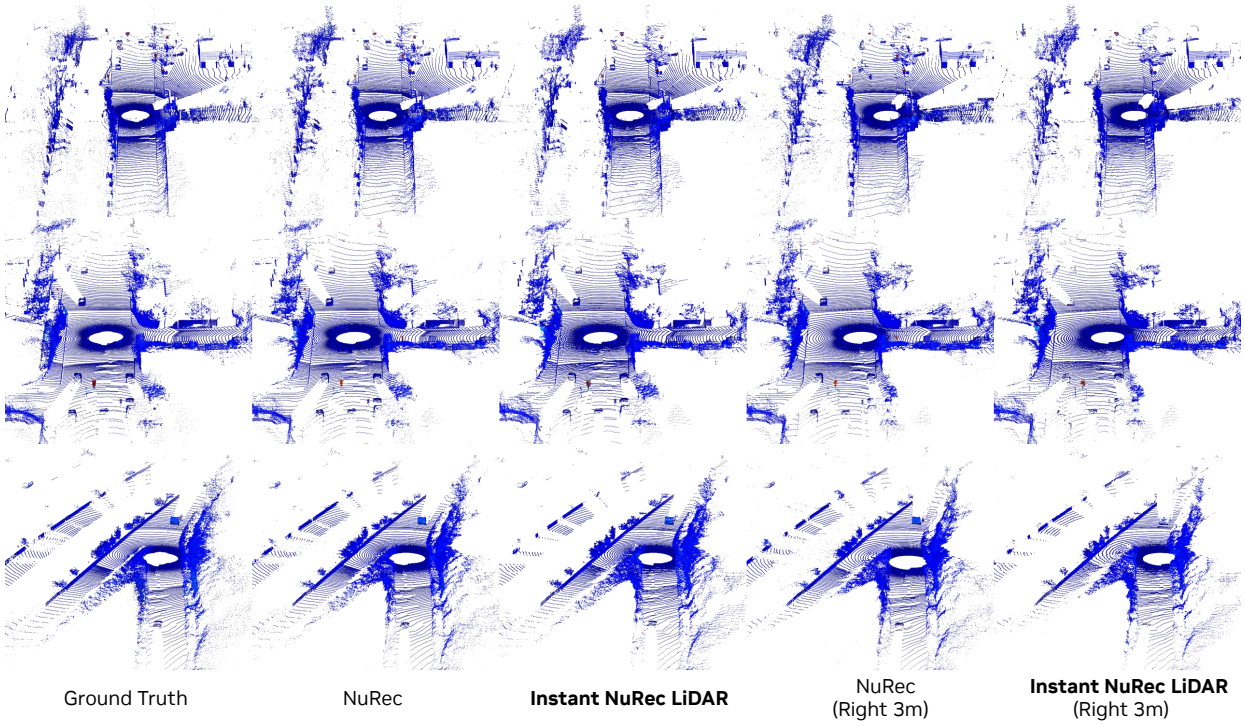

Figure 10: Qualitative comparison of LiDAR reconstruction on internal data for recorded and right-shifted trajectories. We highlight road and curb boundaries, vertical structures, and dynamic vehicles. *Instant NuRec LiDAR* preserves shape details and achieves overall quality comparable to NuRec.

Tab. 3 compares *Instant NuRec LiDAR* with the per-scene-optimized NuRec. Compared with NuRec, our method has higher CD, mainly because of coverage rather than precision, indicating well-localized but less complete geometry. Intensity prediction remains comparable. The ablation without the anisotropic scale floor confirms its importance for LiDAR reconstruction; otherwise, Gaussian primitives collapse around observed rays, producing fragmented and incomplete geometry.

Despite the quantitative gap, *Instant NuRec LiDAR* preserves the dominant scene geometry (Fig. 10): it reconstructs the main road layout, nearby structures, and scene boundaries without evident distortion. To obtain the best quality, we increase the number of context LiDAR frames, forming roughly 20 windows per clip. Although this increases the reconstruction time from 1 s to ∼20 s, the method remains orders of magnitude faster than NuRec. Overall, we conclude that *Instant NuRec* can reconstruct LiDAR geometry with qualitatively comparable results despite incomplete coverage, while remaining orders of magnitude faster.

6. Discussion and Conclusion

Limitations. *Instant NuRec* faces an inherent tension between the number of Gaussians and reconstruction quality: matching the quality of per-scene optimization can require many primitives, while a smaller budget under-samples thin structures. Test-time training that refines the predicted Gaussians on the input clip is a promising way to close this gap. Generalization is also limited by the training corpus, so rigs that differ substantially from those seen in training (e.g., low-mounted or fisheye-only setups) currently require fine-tuning. Finally, the three-keyframe piecewise-linear actor trajectories cannot capture sub-second non-rigid motion, such as pedestrian articulation. Denser keyframes could address this limitation. Separately, a streaming formulation in which past reconstructions condition the current forward pass is a promising direction for scaling to long logs.

Conclusion. *Instant NuRec* demonstrates that the cost of high-quality driving-scene reconstruction can be amortized into a single feed-forward pass. It turns a short multi-view log into a layered, fully simulatable 3DGS world consisting of static and dynamic Gaussians, a sky cubemap, and per-camera ISP corrections, directly consumable by NuRec and AlpaSim. By approaching per-scene quality at a cost that is orders of magnitude lower, we hope that *Instant NuRec* will lower the barrier to using neural reconstruction at the fleet scale that AD development now demands.

References

[1] NVIDIA. NuRec. Website, 2026. URL https://research.nvidia.com/labs/sil/nurec/. Accessed 2026-03-08. 1, 3, 8, 9, 14

[2] Qi Wu, Janick Martinez Esturo, Ashkan Mirzaei, Nicolas Moenne-Loccoz, and Zan Gojcic. 3DGUT: Enabling distorted cameras and secondary rays in gaussian splatting. *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025. 1, 3, 5, 7, 8

[3] NVIDIA, Yulong Cao, Riccardo de Lutio, Sanja Fidler, Guillermo Garcia Cobo, Zan Gojcic, Maximilian Igl, Boris Ivanovic, Peter Karkus, Janick Martinez Esturo, Marco Pavone, Aaron Smith, Ellie Tanimura, Michal Tyszkiewicz, Michael Watson, Qi Wu, and Le Zhang. AlpaSim: A modular, lightweight, and data-driven research simulator for autonomous driving. Software, October 2025. URL https://github.com/NVlabs/alpasim. 1, 2, 3, 10

[4] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3D Gaussian Splatting for real-time radiance field rendering. *ACM Transactions on Graphics (SIGGRAPH)*, 2023. 2

[5] Ziyu Chen, Jiawei Yang, Jiahui Huang, Riccardo de Lutio, Janick Martinez Esturo, Boris Ivanovic, Or Litany, Zan Gojcic, Sanja Fidler, Marco Pavone, Li Song, and Yue Wang. OmniRe: Omni urban scene reconstruction. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=11xgiMEI5o. 2, 3

[6] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *ICLR*, 2021. 2

[7] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: Representing scenes as neural radiance fields for view synthesis. In *European Conference on Computer Vision (ECCV)*, 2020. 2

[8] Konstantinos Rematas, Andrew Liu, Pratul P. Srinivasan, Jonathan T. Barron, Andrea Tagliasacchi, Thomas Funkhouser, and Vittorio Ferrari. Urban radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 2

[9] Matthew Tancik, Vincent Casser, Xinchen Yan, Sabeek Pradhan, Ben Mildenhall, Pratul P. Srinivasan, Jonathan T. Barron, and Henrik Kretzschmar. Block-NeRF: Scalable large scene neural view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 2

[10] Julian Ost, Fahim Mannan, Nils Thuerey, Julian Knodt, and Felix Heide. Neural scene graphs for dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2

[11] Jiawei Yang, Boris Ivanovic, Or Litany, Xinshuo Weng, Seung Wook Kim, Boyi Li, Tong Che, Danfei Xu, Sanja Fidler, Marco Pavone, and Yue Wang. EmerNeRF: Emergent spatial-temporal scene decomposition via self-supervision. In *International Conference on Learning Representations (ICLR)*, 2024. 2

[12] Adam Tonderski, Carl Lindström, Georg Hess, William Ljungbergh, Lennart Svensson, and Christoffer Petersson. NeuRAD: Neural rendering for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 2

[13] Xiaoyu Zhou, Zhiwei Lin, Xiaojun Shan, Yongtao Wang, Deqing Sun, and Ming-Hsuan Yang. DrivingGaussian: Composite gaussian splatting for surrounding dynamic autonomous driving scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 3

[14] Yunzhi Yan, Haotong Lin, Chenxu Zhou, Weijie Wang, Haiyang Sun, Kun Zhan, Xianpeng Lang, Xiaowei Zhou, and Sida Peng. Street Gaussians: Modeling dynamic urban scenes with gaussian splatting. In *European Conference on Computer Vision (ECCV)*, 2024. 3

[15] Yurui Chen, Chun Gu, Junzhe Jiang, Xiatian Zhu, and Li Zhang. Periodic vibration gaussian: Dynamic urban scene reconstruction and real-time rendering. *arXiv preprint arXiv:2311.18561*, 2023. 3

[16] Nicolas Moenne-Loccoz, Ashkan Mirzaei, Or Perel, Riccardo de Lutio, Janick Martinez Esturo, Gavriel State, Sanja Fidler, Nicholas Sharp, and Zan Gojcic. 3D Gaussian Ray Tracing: Fast tracing of particle scenes. *ACM Transactions on Graphics (SIGGRAPH Asia)*, 2024. 3

[17] CARLA Team. CARLA documentation: NVIDIA NuRec. Website, 2026. URL https://carla.readthedocs.io/en/latest/nvidia_nurec/. Accessed 2026-03-09. 3, 9

[18] Ruoshi Liu, Rundi Wu, Basile Van Hoorick, Pavel Tokmakov, Sergey Zakharov, and Carl Vondrick. Zero-1-to-3: Zero-shot one image to 3D object. *arXiv preprint arXiv:2303.11328*, 2023. 3

[19] Yichun Shi, Peng Wang, Jianglong Ye, Mai Long, Kejie Li, and Xiao Yang. MVDream: Multi-view diffusion for 3D generation. *arXiv preprint arXiv:2308.16512*, 2023. 3

[20] Peng Wang and Yichun Shi. ImageDream: Image-prompt multi-view diffusion for 3D generation. *arXiv preprint arXiv:2312.02201*, 2023. 3

[21] Jiaxiang Tang, Zhaoxi Chen, Xiaokang Chen, Tengfei Wang, Gang Zeng, and Ziwei Liu. LGM: Large multi-view gaussian model for high-resolution 3D content creation. *arXiv preprint arXiv:2402.05054*, 2024. 3

[22] Jianfeng Xiang, Zelong Lv, Sicheng Xu, Yu Deng, Ruicheng Wang, Bowen Zhang, Dong Chen, Xin Tong, and Jiaolong Yang. Structured 3D latents for scalable and versatile 3D generation. *arXiv preprint arXiv:2412.01506*, 2024. 3

[23] Jianfeng Xiang, Xiaoxue Chen, Sicheng Xu, Ruicheng Wang, Zelong Lv, Yu Deng, Hongyuan Zhu, Yue Dong, Hao Zhao, Nicholas Jing Yuan, and Jiaolong Yang. Native and compact structured latents for 3D generation. *arXiv preprint arXiv:2512.14692*, 2025. 3

[24] Zibo Zhao, Zeqiang Lai, Qingxiang Lin, et al. Hunyuan3D 2.0: Scaling diffusion models for high resolution textured 3D assets generation. *arXiv preprint arXiv:2501.12202*, 2025. 3

[25] SAM 3D Team, Xingyu Chen, Fu-Jen Chu, Pierre Gleize, Kevin J Liang, Alexander Sax, Hao Tang, Weiyao Wang, Michelle Guo, Thibaut Hardin, Xiang Li, Aohan Lin, Jiawei Liu, Ziqi Ma, Anushka Sagar, Bowen Song, Xiaodong Wang, Jianing Yang, Bowen Zhang, Piotr Dollár, Georgia Gkioxari, Matt Feiszli, and Jitendra Malik. SAM 3D: 3Dfy anything in images, 2025. URL https://arxiv.org/abs/2511.16624. 3

[26] David Charatan, Sizhe Li, Andrea Tagliasacchi, and Vincent Sitzmann. pixelSplat: 3D gaussian splats from image pairs for scalable generalizable 3D reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 3

[27] Yuedong Chen, Haofei Xu, Chuanxia Zheng, Bohan Zhuang, Marc Pollefeys, Andreas Geiger, Tat-Jen Cham, and Jianfei Cai. MVSplat: Efficient 3D gaussian splatting from sparse multi-view images. In *European Conference on Computer Vision (ECCV)*, 2024. 3

[28] Kai Zhang, Sai Bi, Hao Tan, Yuanbo Xiangli, Nanxuan Zhao, Kalyan Sunkavalli, and Zexiang Xu. GS-LRM: Large reconstruction model for 3D gaussian splatting. In *European Conference on Computer Vision (ECCV)*, 2024. 3

[29] Haofei Xu, Songyou Peng, Fangjinhua Wang, Hermann Blum, Daniel Barath, Andreas Geiger, and Marc Pollefeys. DepthSplat: Connecting gaussian splatting and depth. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025. 3, 8, 9

[30] 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)*, 2024. 3

[31] Botao Ye, Sifei Liu, Haofei Xu, Xueting Li, Marc Pollefeys, Ming-Hsuan Yang, and Songyou Peng. No pose, no problem: Surprisingly simple 3D gaussian splats from sparse unposed images. *arXiv preprint arXiv:2410.24207*, 2024. 3

[32] 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. *arXiv preprint arXiv:2505.23716*, 2025. 3

[33] Jiawei Yang, Jiahui Huang, Boris Ivanovic, Yuxiao Chen, Yan Wang, Boyi Li, Yurong You, Apoorva Sharma, Maximilian Igl, Peter Karkus, et al. STORM: Spatio-temporal reconstruction model for large-scale outdoor scenes. In *International Conference on Learning Representations*, volume 2025, pages 50446–50465, 2025. 3, 5, 8, 9, 12

[34] Qijian Tian, Xin Tan, Yuan Xie, and Lizhuang Ma. DrivingForward: Feed-forward 3D gaussian splatting for driving scene reconstruction from flexible surround-view input. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2025. 3

[35] Xiaoxue Chen, Ziyi Xiong, Yuantao Chen, Gen Li, Nan Wang, Hongcheng Luo, Long Chen, Haiyang Sun, Bing Wang, Guang Chen, Hangjun Ye, Hongyang Li, Ya-Qin Zhang, and Hao Zhao. DGGT: Feedforward 4D reconstruction of dynamic driving scenes using unposed images. *arXiv preprint arXiv:2512.03004*, 2025. 3, 8, 9

[36] Haibao Yu, Kuntao Xiao, Jiahang Wang, Ruiyang Hao, Yuxin Huang, Guoran Hu, Haifang Qin, Bowen Jing, Yuntian Bo, and Ping Luo. ReconDrive: Fast feed-forward 4D gaussian splatting for autonomous driving scene reconstruction. *arXiv preprint arXiv:2603.07552*, 2026. 3

[37] Zhongrui Yu, Zhao Wang, Yijia Xie, Yida Wang, Xueyang Zhang, Yifei Zhan, and Kun Zhan. StreetForward: Perceiving dynamic street with feedforward causal attention. *arXiv preprint arXiv:2603.19552*, 2026. 3, 8

[38] Shuo Wang, Jilin Mei, Fuyang Liu, Wenfei Guan, Fanjie Kong, Zhihua Zhao, Shuai Wang, Chen Min, and Yu Hu. Ground4D: Spatially-grounded feedforward 4D reconstruction for unstructured off-road scenes. *arXiv preprint arXiv:2605.04435*, 2026. 3

[39] Jiawei Ren, Michal Tyszkiewicz, Jiahui Huang, and Zan Gojcic. TokenGS: Decoupling 3D gaussian prediction from pixels with learnable tokens. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026. 3

[40] Chensheng Peng, Quentin Herau, Jiezhi Yang, Yichen Xie, Yihan Hu, Wenzhao Zheng, Matthew Strong, Masayoshi Tomizuka, and Wei Zhan. UniQueR: Unified query-based feedforward 3D reconstruction. *arXiv preprint arXiv:2603.22851*, 2026. 3

[41] Roni Itkin, Noam Issachar, Yehonatan Keypur, Xingyu Chen, Anpei Chen, and Sagie Benaim. GlobalSplat: Efficient feed-forward 3D gaussian splatting via global scene tokens. *arXiv preprint arXiv:2604.15284*, 2026. 3

[42] Anthony Hu, Lloyd Russell, Hudson Yeo, Zak Murez, George Fedoseev, Alex Kendall, Jamie Shotton, and Gianluca Corrado. GAIA-1: A generative world model for autonomous driving. *arXiv preprint arXiv:2309.17080*, 2023. 3

[43] Xiaofeng Wang, Zheng Zhu, Guan Huang, Xinze Chen, Jiagang Zhu, and Jiwen Lu. DriveDreamer: Towards real-world-driven world models for autonomous driving. *arXiv preprint arXiv:2309.09777*, 2023. 3

[44] Ruiyuan Gao, Kai Chen, Enze Xie, Lanqing Hong, Zhenguo Li, Dit-Yan Yeung, and Qiang Xu. MagicDrive: Street view generation with diverse 3D geometry control. *arXiv preprint arXiv:2310.02601*, 2023. 3

[45] Yuqing Wen, Yucheng Zhao, Yingfei Liu, Fan Jia, Yanhui Wang, Chong Luo, Chi Zhang, Tiancai Wang, Xiaoyan Sun, and Xiangyu Zhang. Panacea: Panoramic and controllable video generation for autonomous driving. *arXiv preprint arXiv:2311.16813*, 2023. 3

[46] Yuqi Wang, Jiawei He, Lue Fan, Hongxin Li, Yuntao Chen, and Zhaoxiang Zhang. Driving into the future: Multiview visual forecasting and planning with world model for autonomous driving, 2023. URL https://arxiv.org/abs/2311.17918. 3

[47] Shenyuan Gao, Jiazhi Yang, Li Chen, Kashyap Chitta, Yihang Qiu, Andreas Geiger, Jun Zhang, and Hongyang Li. Vista: A generalizable driving world model with high fidelity and versatile controllability. *Advances in Neural Information Processing Systems (NeurIPS)*, 2024. 3

[48] 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. *arXiv preprint arXiv:2403.06845*, 2024. 3

[49] Lloyd Russell, Anthony Hu, Lorenzo Bertoni, George Fedoseev, Jamie Shotton, Elahe Arani, and Gianluca Corrado. GAIA-2: A controllable multi-view generative world model for autonomous driving, 2025. URL https://arxiv.org/abs/2503.20523. 3

[50] Mariam Hassan, Sebastian Stapf, Ahmad Rahimi, Pedro M B Rezende, Yasaman Haghighi, David Brüggemann, Isinsu Katircioglu, Lin Zhang, Xiaoran Chen, Suman Saha, Marco Cannici, Elie Aljalbout, Botao Ye, Xi Wang, Aram Davtyan, Mathieu Salzmann, Davide Scaramuzza, Marc Pollefeys, Paolo Favaro, and Alexandre Alahi. GEM: A generalizable ego-vision multimodal world model for fine-grained ego-motion, object dynamics, and scene composition control. *arXiv preprint arXiv:2412.11198*, 2024. 3

[51] Xiaotao Hu, Wei Yin, Mingkai Jia, Junyuan Deng, Xiaoyang Guo, Qian Zhang, Xiaoxiao Long, and Ping Tan. DrivingWorld: Constructing world model for autonomous driving via video GPT. *arXiv preprint arXiv:2412.19505*, 2024. 3

[52] Kaiwen Zhang, Zhenyu Tang, Xiaotao Hu, Xingang Pan, Xiaoyang Guo, Yuan Liu, Jingwei Huang, Li Yuan, Qian Zhang, Xiao-Xiao Long, Xun Cao, and Wei Yin. Epona: Autoregressive diffusion world model for autonomous driving. *arXiv preprint arXiv:2506.24113*, 2025. 3

[53] NVIDIA, Niket Agarwal, Arslan Ali, Maciej Bala, Yogesh Balaji, et al. Cosmos world foundation model platform for physical AI. *arXiv preprint arXiv:2501.03575*, 2025. 3

[54] 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. *arXiv preprint arXiv:2506.09042*, 2025. 3

[55] 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. *arXiv preprint arXiv:2410.13571*, 2024. 3

[56] 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. *arXiv preprint arXiv:2501.00601*, 2025. 3

[57] 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. *arXiv preprint arXiv:2411.19548*, 2024. 3

[58] Jay Zhangjie Wu, Yuxuan Zhang, Haithem Turki, Xuanchi Ren, Jun Gao, Mike Zheng Shou, Sanja Fidler, Zan Gojcic, and Huan Ling. Difix3D+: Improving 3D reconstructions with single-step diffusion models. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025. 3

[59] Yuxuan Zhang, Katarína Tóthová, Zian Wang, Kangxue Yin, Haithem Turki, Riccardo de Lutio, Yen-Yu Chang, Or Litany, Sanja Fidler, and Zan Gojcic. DiffusionHarmonizer: Bridging neural reconstruction and photorealistic simulation with online diffusion enhancer. *arXiv preprint arXiv:2602.24096*, 2026. URL https://arxiv.org/abs/2602.24096. 3

[60] Yunzhi Yan, Zhen Xu, Haotong Lin, Haian Jin, Haoyu Guo, Yida Wang, Kun Zhan, Xianpeng Lang, Hujun Bao, Xiaowei Zhou, and Sida Peng. StreetCrafter: Street view synthesis with controllable video diffusion models. *arXiv preprint arXiv:2412.13188*, 2024. 3

[61] Qitai Wang, Lue Fan, Yuqi Wang, Yuntao Chen, and Zhaoxiang Zhang. FreeVS: Generative view synthesis on free driving trajectory. *arXiv preprint arXiv:2410.18079*, 2024. 3

[62] Lijun Zhou, Hongcheng Luo, Zhenxin Zhu, Cheng Chi, Mingfei Tu, Kaixin Xiong, Lei Gong, Zhanqian Wu, Zehan Zhang, Fangzhen Li, Hao Li, Yingying Shen, Jiale He, Haohui Zhu, Shan Zhao, Kai Wang, Zhiwei Zhan, Yuechuan Pu, Kaiyuan Tan, Ruiling Yang, Xianqi Wang, Tianyi Yan, Jiawei Zhou, Lei Zhang, Jingyang Zhao, Xi Zhou, Chitian Sun, Chenming Wu, Jiong Deng, Hongwei Xie, Ming Lu, Kun Ma, Long Chen, Guang Chen, Hangjun Ye, Bing Wang, and

Haiyang Sun. Xiaomi EV World Model: A joint world model integrating reconstruction and generation for autonomous driving. *arXiv preprint arXiv:2605.18137*, 2026. 3

[63] Haotong Lin, Sili Chen, Junhao Liew, Donny Y Chen, Zhenyu Li, Guang Shi, Jiashi Feng, and Bingyi Kang. Depth Anything 3: Recovering the visual space from any views. *arXiv preprint arXiv:2511.10647*, 2025. 4, 7, 8, 9

[64] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. DINOv2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023. 4

[65] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 12179–12188, 2021. 5

[66] Edgar Sucar, Eldar Insafutdinov, Zihang Lai, and Andrea Vedaldi. V-DPM: 4D video reconstruction with dynamic point maps. *arXiv preprint arXiv:2601.09499*, 2026. 5

[67] Yihang Luo, Shangchen Zhou, Yushi Lan, Xingang Pan, and Chen Change Loy. 4RC: 4D reconstruction via conditional querying anytime and anywhere. *arXiv preprint arXiv:2602.10094*, 2026. 5

[68] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023. 5

[69] Ricardo Martin-Brualla, Noha Radwan, Mehdi S. M. Sajjadi, Jonathan T. Barron, Alexey Dosovitskiy, and Daniel Duckworth. NeRF in the wild: Neural radiance fields for unconstrained photo collections. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 6

[70] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 586–595, June 2018. doi: 10.1109/CVPR.2018.00068. 7

[71] Alessandro Burzio, Tobias Fischer, Sven Elflein, Qunjie Zhou, Riccardo de Lutio, Jiawei Ren, Jiahui Huang, Shengyu Huang, Marc Pollefeys, Laura Leal-Taixé, et al. Déjà View: Looping transformers for multi-view 3D reconstruction. *arXiv preprint arXiv:2605.30215*, 2026. 7

[72] 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)*, 2020. 8

[73] Florent Bartoccioni, Elias Ramzi, Victor Besnier, Shashanka Venkataramanan, Tuan-Hung Vu, Yihong Xu, Loick Chambon, Spyros Gidaris, Serkan Odabas, David Hurych, Renaud Marlet, Alexandre Boulch, Mickael Chen, Éloi Zablocki, Andrei Bursuc, Eduardo Valle, and Matthieu Cord. VaViM and VaVAM: Autonomous driving through video generative modeling. *arXiv preprint arXiv:2502.15672*, 2025. 10

[74] NVIDIA, Yan Wang, Wenjie Luo, Junjie Bai, Yulong Cao, Tong Che, Ke Chen, Yuxiao Chen, Jenna Diamond, Yifan Ding, Wenhao Ding, Liang Feng, Greg Heinrich, Jack Huang, Peter Karkus, Boyi Li, Pinyi Li, Tsung-Yi Lin, Dongran Liu, Ming-Yu Liu, Langechuan Liu, Zhijian Liu, Jason Lu, Yunxiang Mao, Pavlo Molchanov, Lindsey Pavao, Zhenghao Peng, Mike Ranzinger, Ed Schmerling, Shida Shen, Yunfei Shi, Sarah Tariq, Ran Tian, Tilman Wekel, Xinshuo Weng, Tianjun Xiao, Eric Yang, Xiaodong Yang, Yurong You, Xiaohui Zeng, Wenyuan Zhang, Boris Ivanovic, and Marco Pavone. Alpamayo-R1: Bridging reasoning and action prediction for generalizable autonomous driving in the long tail. *arXiv preprint arXiv:2511.00088*, 2025. 10

[75] Zehao Yu, Anpei Chen, Binbin Huang, Torsten Sattler, and Andreas Geiger. Mip-Splatting: Alias-free 3D gaussian splatting. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 19447–19456, 2024. 13

A. Contributors

Research. Jiahui Huang, Jiawei Ren, Michal Tyszkiewicz, Xin Kang, Seung Wook Kim, Shengyu Huang, Laura Leal-Taixe[‡], Zan Gojcic[‡], Sanja Fidler[‡].

Engineering. Bjoern Haefner, Michael Shelley, Ning Xu, Qi Wu, Janick Martinez Esturo, Nick Schneider[‡].

B. Acknowledgments

We greatly appreciate the contributions of the following individuals:

Alessandro Burzio, Alex Perec, Bingxin Ke, Daniel Dworakowski, Despoina Paschalidou, Emmanuel Attia, Jun Gao, Katarina Tothova, Lei Zhang, Lucrezia Shen, Murat Arar, Naveen Kumar Rai, Nicolas Moenne-Loccoz, Rodolfo Lima, Sangeetha Grama Srinivasan, Sean Pieper, Sergio Agostinho, Sherwin Bahmani, Shikhar Solanki, Sipeng Zhang, Tianchang Shen, Tobias Fischer, Weihua Zhang, Xuanchi Ren, Yixin Cao.

[‡]Leadership.