HIFINVS: HIGH-FIDELITY NOVEL VIEW SYNTHESIS FROM UNPOSED IMAGES

Anonymous authors
Paper under double-blind review

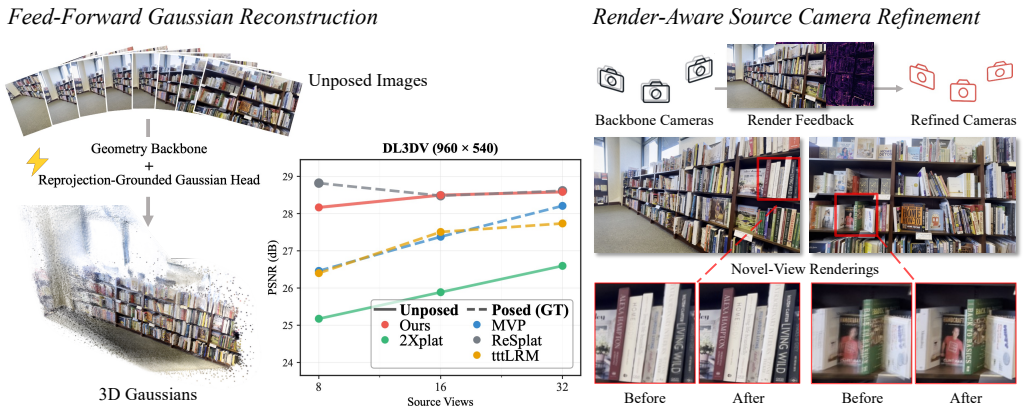


Figure 1: **High-fidelity feed-forward novel view synthesis from unposed images.** HiFiNVS builds on a frozen visual geometry backbone, using explicit 3D correspondences derived from predicted depth and cameras to adapt features across views for Gaussian reconstruction. Rendering feedback from the initial scene then guides feed-forward refinement of the predicted cameras. Reconstructing the scene once with the refined cameras suppresses rendering artifacts, as shown on the right. The center plot shows that HiFiNVS achieves competitive rendering quality compared with methods supplied with ground truth poses on the DL3DV dataset.

ABSTRACT

Feed-Forward Novel View Synthesis from unposed images has recently benefited from pretrained visual geometry models, which provide strong camera and dense geometry priors for Gaussian reconstruction. Nevertheless, the resulting Gaussian predictions can remain inconsistent across views, while errors in the estimated cameras often lead to ghosting and blurred details. We introduce HiFiNVS, a backbone-agnostic NVS framework that exploits predicted geometry not only for reconstruction, but also to establish explicit cross-view constraints for feature adaptation and camera refinement. Specifically, our Reprojection-Grounded Graph Attention constructs a patch graph by reprojecting 3D points across images and propagates information along geometrically grounded correspondence edges, thereby improving the cross-view consistency of Gaussian predictions. We further introduce a feed-forward camera refiner that updates the predicted cameras using rendering feedback from the initial Gaussian scene. Rerunning our reconstruction once with the refined cameras effectively suppresses rendering artifacts. HiFiNVS keeps the visual geometry backbone frozen and can be instantiated with DA3, π^3, and VGGT. Across all three backbones, camera refinement improves both camera accuracy and rendering quality. Experiments on unposed-view benchmarks show that HiFiNVS achieves state-of-the-art NVS performance.

1 INTRODUCTION

Feed-forward Gaussian reconstruction predicts 3D Gaussian parameters directly from images (Charatan et al., 2024; Chen et al., 2024; Zhang et al., 2024), replacing costly per-scene optimization in NeRF and 3D Gaussian Splatting (Mildenhall et al., 2020; Kerbl et al., 2023). Recent advances have

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

enabled high-fidelity novel view synthesis (NVS) at higher resolutions and with more input views (Xu et al., 2025a; 2026; Kang et al., 2026b; Chen et al., 2025). Despite this progress, reconstruction quality still depends on the accuracy of input camera poses, making high-fidelity reconstruction from unposed images challenging (Kang et al., 2026b).

Recent visual geometry models predict depth and cameras directly from images (Wang et al., 2025a; 2026b; Lin et al., 2026), enabling a new paradigm that builds on these pretrained models for feed-forward Gaussian reconstruction from unposed images (Jiang et al., 2025; Ye et al., 2026). To improve rendering quality, Selfi learns geometrically aligned features from a frozen backbone (Deng et al., 2026), while 2Xplat jointly trains separate geometry and appearance models (Jeong et al., 2026). Nevertheless, these methods largely draw on the architectures of visual geometry models, which are designed with minimal 3D inductive biases. We therefore investigate whether explicit 3D correspondences can guide networks to aggregate backbone features more effectively across views.

HiFiNVS uses Reprojection-Grounded Graph Attention (RGGA) to adapt features across views. RGGA simply derives coarse correspondences across views from predicted depth and cameras. Unlike ReSplat (Xu et al., 2026), which aggregates features over kNN neighborhoods of Gaussian centers, we use 3D reprojection to construct a correspondence graph at the patch level for feature aggregation. This design avoids costly attention over dense Gaussian points while retaining the native patch structure of the backbone features. RGGA thus promotes more consistent Gaussian predictions across views, leading to higher-fidelity novel-view renderings.

Errors in source camera predictions from visual geometry models can misalign observations across views, leading to inconsistent geometry and rendering artifacts. Existing methods address this challenge differently. YoNoSplat mixes ground-truth and predicted poses during training, while 2Xplat jointly trains camera estimation and Gaussian reconstruction (Ye et al., 2026; Jeong et al., 2026). These strategies account for camera errors during training, but neither introduces a dedicated stage to explicitly refine the predicted source cameras. Selfi instead refines cameras through bundle adjustment using learned correspondences, but requires per-scene optimization (Deng et al., 2026).

To explicitly refine predicted source cameras without per-scene optimization, we learn a feed-forward camera refiner that uses rendering feedback from the initial Gaussian scene. As the scene integrates observations from multiple views, discrepancies between the scene renderings and source images can provide cues about camera misalignment. Following the design of RGGA, the refiner uses 3D reprojection to align rendering feedback from multiple views with each source image patch, then combines the source patches with this feedback to predict camera updates. The refinement improves camera estimates from the backbone, and scenes rebuilt with the refined cameras show fewer rendering artifacts.

Experiments on DL3DV, Tanks & Temples, and ScanNet++ show that HiFiNVS synthesizes high-fidelity novel views from unposed images, surpassing several baselines supplied with ground-truth poses. The framework supports DA3, π^3, and VGGT, with camera refinement improving both source camera accuracy and NVS quality across all three backbones. Our frozen camera refiner further improves unposed NVS quality for other reconstruction methods without fine-tuning.

Contributions.

- We introduce HiFiNVS, a feed-forward Gaussian reconstruction framework for high-fidelity NVS from unposed images, instantiated with frozen DA3, π^3, and VGGT backbones.
- We propose RGGA, which constructs a patch correspondence graph through 3D reprojection and aggregates features along its edges, promoting more consistent Gaussian predictions across views.
- We develop a feed-forward camera refiner that uses rendering feedback to correct predicted source cameras, improving both source camera accuracy and NVS quality.
- Across three unposed-view benchmarks, HiFiNVS achieves state-of-the-art rendering quality, surpassing most strong methods supplied with ground-truth poses.

2 Related Work

Feed-forward visual geometry. Visual geometry models learn to recover scene geometry from unposed images. DUSt3R formulates pairwise reconstruction as point-map prediction, and MASt3R adds learned matching features (Wang et al., 2024; Leroy et al., 2024). MV-DUSt3R+ and Fast3R

extend point-map prediction to multiple views, while Spann3R and CUT3R support incremental reconstruction through spatial memory or persistent state (Tang et al., 2025; Yang et al., 2025; Wang & Agapito, 2025; Wang et al., 2025b). Models such as VGGT, π^3, DA3, and MapAnything jointly recover scene geometry and cameras from multi-view inputs (Wang et al., 2025a; 2026b; Lin et al., 2026; Keetha et al., 2026). HiFiNVS provides a general framework for high-fidelity NVS from unposed images using frozen visual geometry backbones. We instantiate it with DA3, π^3, and VGGT.

Feed-forward Gaussian reconstruction. Feed-forward methods reconstruct Gaussian scenes from calibrated images (Charatan et al., 2024; Chen et al., 2024; Zhang et al., 2024; Chen et al., 2025; Kang et al., 2026b) or unposed inputs (Ye et al., 2025; Xu et al., 2025b; Kang et al., 2025; Huang & Mikolajczyk, 2025; Zhang et al., 2025). Pretrained models provide geometric priors for reconstruction: DepthSplat and MonoSplat incorporate monocular depth features, while PF3plat combines pretrained depth and correspondence models for Gaussian alignment (Xu et al., 2025a; Liu et al., 2025; Hong et al., 2025). Splatt3R adds Gaussian heads to a frozen stereo backbone, whereas AnySplat and YoNoSplat fine-tune visual geometry networks for rendering (Smart et al., 2024; Jiang et al., 2025; Ye et al., 2026). Selfi instead learns feature adaptation through reprojection-based consistency supervision (Deng et al., 2026). Other approaches emphasize compact representations, using point fusion or learned query-based decoding (Smart et al., 2026; An et al., 2026). 2Xplat conditions a separate Gaussian reconstruction model on cameras predicted by a pretrained visual geometry model (Jeong et al., 2026). In contrast, HiFiNVS uses explicit 3D correspondences to guide the aggregation of pretrained backbone features across views, promoting more consistent Gaussian predictions.

Camera refinement and rendering feedback. Accurate camera estimates support alignment across views and high-quality novel view synthesis. Optimization-based approaches jointly refine cameras and scene representations through per-scene gradient optimization (Lin et al., 2021a). With a fixed scene representation, iNeRF optimizes camera poses by comparing rendered and observed images (Lin et al., 2021b). Generalizable NVS methods learn camera updates from feature reprojection costs (Chen & Lee, 2023; Li et al., 2024). For reconstruction with visual geometry models, Selfi matches its learned, geometrically aligned features across views to establish correspondences, then refines predicted cameras through bundle adjustment (Deng et al., 2026). HiFiNVS instead uses a feed-forward camera refiner driven by rendering feedback to update source cameras predicted by visual geometry models. Our frozen refiner further improves unposed NVS quality in other reconstruction pipelines without fine-tuning.

3 METHOD

3.1 PROBLEM FORMULATION AND OVERVIEW

Given unposed source images, HiFiNVS reconstructs a 3D Gaussian scene for novel view synthesis (Figure 2). A frozen visual geometry backbone extracts patch features and predicts cameras, while a trainable depth head initialized from the backbone predicts dense depth. The predicted cameras and depth define a reprojection graph connecting patches across views. Within the Gaussian head, alternating graph attention and frame attention adapt the patch features, and the Gaussian decoder predicts per-view Gaussian parameters to form the initial scene. A feed-forward camera refiner then uses feedback from source-view renderings of this scene to refine the predicted cameras. We rebuild the graph and rerun the Gaussian head once with the refined cameras, reusing the backbone features and depth predictions.

3.2 REPROJECTION-GROUNDED GRAPH ATTENTION

RGGA uses correspondences derived from predicted cameras and depth to guide feature aggregation across views at the patch level. More specifically, we construct a directed graph G through 3D reprojection using the predicted camera C_v and depth map D_v of each source view v. Each graph node represents an image patch. An edge $j \to i$ identifies patch j as a geometrically corresponding cross-view neighbor of query patch i, from which i aggregates features.

A patch may span a depth discontinuity, so for each patch i in view v, we sample M regularly spaced pixels with coordinates $\{\boldsymbol{u}_i^m\}_{m=1}^M$. We unproject each sampled pixel into world space and project

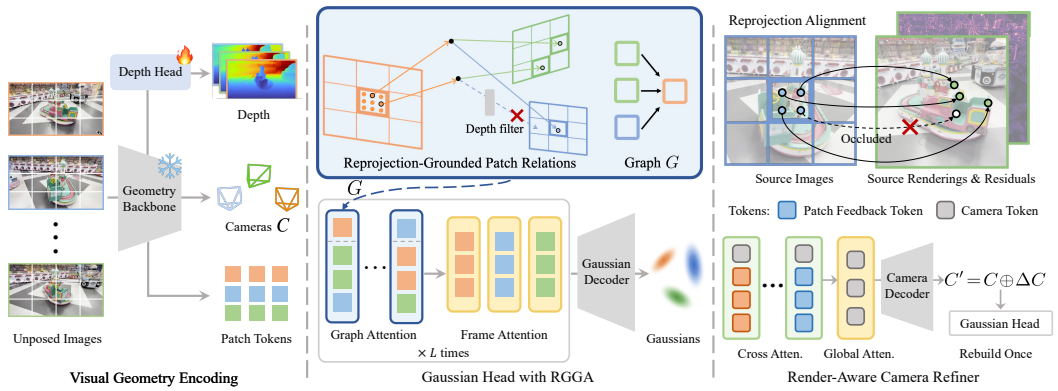


Figure 2: **Overview of HiFiNVS.** Given unposed source images, a frozen visual geometry backbone extracts patch features and predicts cameras C, while a trainable depth head predicts dense depth. RGGA aggregates features across views using a patch graph G constructed from predicted cameras and depth, alternating graph and frame attention to adapt patch features for Gaussian decoding. With the initial Gaussian scene built, the camera refiner uses Gaussian reprojection to align source-view rendering feedback with source patch features and combines both into patch feedback tokens. Camera tokens conditioned on embeddings of C aggregate patch feedback tokens within each view through cross-attention, followed by global attention across all camera tokens. The camera decoder predicts residuals ΔC to obtain refined cameras C'. We then rebuild the graph and Gaussian scene with C'.

the 3D point into every other source view v':

$$\boldsymbol{x}_i^m = \Pi_v^{-1}(\boldsymbol{u}_i^m, D_v(\boldsymbol{u}_i^m)), \qquad \widehat{\boldsymbol{u}}_{v' \leftarrow i}^m = \Pi_{v'}(\boldsymbol{x}_i^m), \quad v' \neq v. \tag{1}$$

Here, Π_v and Π_v^{-1} denote projection and depth-based unprojection under camera C_v. Each in-bounds projection identifies a candidate patch j in view v'. A sample is valid if the depth of $\boldsymbol{x}_i^m$ in view v' agrees with $D_{v'}$ at the projected location and its surface normal is consistent with the normal estimated there. For each query patch i, we retain up to k candidate patches with the highest valid-projection counts and add a directed edge $j \to i$ for each retained candidate.

Using G, graph attention aggregates features from cross-view neighbors through self-attention along its edges, while frame attention aggregates features among patches within each source view. We repeat graph attention followed by frame attention L times to obtain the adapted features for Gaussian decoding.

3.3 Gaussian Decoding

For each source view v, a DPT decoder (Ranftl et al., 2021) decodes the adapted features into a dense feature map, which is fused with image and depth features to predict one Gaussian per pixel.

The Gaussian decoder predicts a bounded per-pixel log-depth residual δ_v and places each Gaussian center along its pixel ray under camera C_v at depth $\widetilde{D}_v = D_v \exp(\delta_v)$. Although the frozen backbone provides a strong depth prior, its estimates can remain inaccurate, making it difficult to obtain suitable Gaussian center locations using only δ_v. We therefore retain the backbone depth-head architecture, initialize the depth head with the pretrained weights, and fine-tune it under rendering supervision. This allows the base depth D_v to adapt to Gaussian reconstruction, while δ_v provides a local correction.

For the remaining attributes, the Gaussian head predicts scales, rotations, and spherical-harmonic coefficients for view-dependent color and opacity, following prior work (Imtiaz et al., 2025; Kang et al., 2026b). The per-view Gaussians are concatenated to form the initial Gaussian scene.

3.4 Render-Aware Camera Refinement

We refine the predicted source cameras using rendering feedback from the initial Gaussian scene. As this scene integrates observations from multiple views, discrepancies between its source-view renderings and the source images can provide cues about camera misalignment. For each source view v, we render RGB I_v^{rend} and depth D_v^{rend} under C_v. A frozen ResNet-18 encoder Φ (He et al., 2016) extracts multiscale features from source and rendered images. Feedback comprises rendered RGB and features, RGB residuals $I_v - I_v^{\text{rend}}$, and feature residuals $\Phi(I_v) - \Phi(I_v^{\text{rend}})$.

Gaussian reprojection pairs each source patch with rendering feedback from corresponding locations across views, which connects observed patch appearance to local rendering discrepancies under the current cameras. For each source patch i, we sample M pixels on a regular grid as in Section 3.2 and encode their source RGB, ResNet features, and ray geometry into a patch query.

To gather the corresponding rendering feedback, we project the Gaussian center $\boldsymbol{\mu}_i^m$ predicted for each sampled pixel $\boldsymbol{u}_i^m$ into source view v' and encode the sampled feedback as an evidence token $\boldsymbol{e}_{i,v'}$:

$$\begin{aligned} \boldsymbol{p}_{i,v'}^m &= \Pi_{v'}(\boldsymbol{\mu}_i^m), \\ \boldsymbol{e}_{i,v'} &= \mathrm{Enc}\big([\boldsymbol{h}_{v'}(\boldsymbol{p}_{i,v'}^m)]_{m=1}^M, \boldsymbol{b}_{i,v'}\big)\,. \end{aligned} \tag{2}$$

Here, $\boldsymbol{h}_{v'}$ combines rendered appearance, residuals, and ray geometry. The mask $\boldsymbol{b}_{i,v'}$ marks in-bounds projections whose Gaussian center depths in view v' agree with the rendered depth at the projected locations. The encoder Enc masks invalid samples and maps their ordered representations and validity indicators to an evidence token with an MLP. Views are selected by valid-sample coverage, yielding one evidence token per retained view for each source patch.

Each patch query aggregates its evidence tokens from multiple views through cross-attention. Alternating this operation with frame self-attention within each source view produces one feedback token per patch. For each view, a camera token conditioned on the current camera parameters aggregates patch feedback tokens within this view through cross-attention and then interacts with camera tokens from other views through global self-attention.

The camera decoder predicts bounded residuals ΔC from the camera tokens, comprising per-view rotation and translation updates and a shared intrinsics update, yielding refined cameras C'. We rebuild G and the Gaussian scene once using C', reusing backbone features and depth predictions.

3.5 Training Objectives

We first train the Gaussian head and fine-tune the depth head while keeping the visual geometry backbone frozen. Following the mix-forcing strategy of YoNoSplat (Ye et al., 2026), we initially train reconstruction with GT poses, then mix posed and unposed settings to adapt the model to predicted source poses. The reconstruction objective is

$$\mathcal{L}_{\mathrm{NVS}} = \mathcal{L}_{\mathrm{rgb}} + \lambda_d \mathcal{L}_{\mathrm{depth}} + \lambda_\delta \mathcal{R}_\delta + \lambda_\alpha \mathcal{R}_\alpha, \qquad \mathcal{L}_{\mathrm{depth}} = \rho(\log D - \log D^{\mathrm{geo}})\,. \tag{3}$$

Here, $\mathcal{L}_{\mathrm{rgb}}$ combines MSE and LPIPS (Zhang et al., 2018) on novel views. D denotes the predicted source depth maps, D^{geo} denotes the frozen backbone depth prior, and ρ is the Huber loss. $\mathcal{R}_\delta$ regularizes log-depth residuals, while $\mathcal{R}_\alpha$ regularizes opacity SH coefficients. Rendering supervision adapts D for Gaussian placement, while $\mathcal{L}_{\mathrm{depth}}$ constrains deviations from the pretrained geometry.

We then freeze the reconstruction model and train the camera refiner, supervising the refined cameras C' with GT cameras. The camera objective is

$$\mathcal{L}_{\mathrm{cam}} = \mathcal{L}_{\mathrm{pose}} + \lambda_K \mathcal{L}_K. \tag{4}$$

The pose loss $\mathcal{L}_{\mathrm{pose}}$ combines geodesic rotation error with a Huber penalty on camera center errors. The intrinsics loss $\mathcal{L}_K$ supervises focal lengths and principal points. Loss details and training settings are provided in Appendix B.

4 Experiments

4.1 Experimental Setup

Implementation. For our main experiments, we use Depth Anything 3 (DA3) (Lin et al., 2026) as the frozen visual geometry backbone. We jointly train the depth and Gaussian heads at 448×256 for 70K iterations on 16 A800 GPUs, using batch sizes 8/4/2 for 8/16/32 source views. We then fine-tune both heads at 960×540 for 30K iterations, using batch sizes 2/1 for 8/16 source views. Following the mix-forcing strategy in Ye et al. (2026), we use ground-truth poses for the first 80% of each NVS training stage and sample pose-free batches with probability 0.1 during the remaining 20%. For posed inputs, we align GT poses to the backbone predictions via Sim(3), fitted using corresponding source camera poses. The reprojection graph samples a 7×7 grid per source patch and retains at most 48 neighboring patches from other views using a relative depth threshold of 0.1. After NVS

Table 1: Novel view synthesis at 960×540 on DL3DV and ScanNet++ with 8, 16, and 32 input views. Entries report PSNR↑/SSIM↑/LPIPS↓. Methods marked with * use DA3 camera predictions directly. Ours in the unposed setting includes camera refinement. Unposed methods share the same EPA settings. $\text{ReSplat}_{\text{init}}$ denotes the initial prediction, while $\text{ReSplat}_{\text{re}}$ recurrently updates Gaussians using rendering errors.

Method	DL3DV			ScanNet++		
	8V	16V	32V	8V	16V	32V
Posed (GT poses)						
3DGS_{7k}	22.79/0.783/0.272	26.04/0.850/0.216	28.62/0.904/0.180	21.99/0.804/0.302	25.83/0.875/0.249	29.05/0.918/0.202
$\text{ReSplat}_{\text{init}}$	27.31/0.894/0.169	27.08/0.877/0.207	26.88/0.883/0.205	26.47/0.931/0.192	27.03/0.935/0.208	26.97/0.930/0.204
$\text{ReSplat}_{\text{re}}$	**28.82/0.916**/0.144	28.48/**0.901**/0.177	28.61/0.909/0.173	**28.09/0.944**/0.176	**28.65/0.950**/0.191	28.88/**0.947**/0.187
Long-LRM	15.98/0.676/0.339	22.99/0.788/0.249	26.33/0.836/0.219	16.56/0.729/0.345	23.78/0.841/0.241	26.78/0.875/0.208
tttLRM	26.40/0.854/0.177	27.51/0.867/0.166	27.74/0.879/0.158	26.06/0.883/0.192	28.14/0.911/0.171	28.75/0.916/0.161
iLRM-wide	24.83/0.817/0.239	26.53/0.842/0.214	26.71/0.854/0.206	23.37/0.816/0.284	25.64/0.866/0.243	26.29/0.876/0.228
AdaptSplat	26.92/0.870/0.161	28.39/0.888/0.145	**29.40**/0.909/0.130	24.18/0.831/0.232	27.59/0.893/0.184	28.42/0.910/0.165
MVP	26.46/0.869/0.169	27.38/0.883/0.157	28.21/0.904/0.142	25.46/0.859/0.213	28.50/0.909/0.172	**29.44**/0.922/0.157
Ours_{DA3}	28.28/0.897/**0.136**	**28.79**/0.900/**0.132**	29.21/**0.913/0.124**	26.87/0.911/**0.172**	28.64/0.928/**0.157**	29.35/0.931/**0.149**
Unposed						
2Xplat	25.18/0.830/0.194	25.89/0.840/0.186	26.60/0.861/0.175	24.50/0.839/0.223	26.95/0.884/0.189	27.11/0.888/0.182
$\text{ReSplat}^{*}_{\text{re}}$	26.54/0.840/0.187	25.96/0.832/0.236	25.37/0.814/0.256	25.61/0.894/0.216	25.87/0.891/0.243	25.59/0.877/0.250
tttLRM*	25.96/0.857/0.192	26.76/0.859/0.186	26.52/0.860/0.191	25.25/0.892/0.212	26.87/**0.916**/0.195	27.01/**0.912**/0.192
AdaptSplat*	26.11/0.841/0.181	26.84/0.846/0.175	26.87/0.842/0.181	24.02/0.869/0.228	25.89/0.897/0.207	25.96/0.893/0.204
Ours_{DA3}	**28.17/0.891/0.141**	**28.50/0.890/0.140**	**28.58/0.895/0.138**	**26.54/0.896/0.180**	**27.76**/0.909/**0.170**	**28.16**/0.908/**0.165**

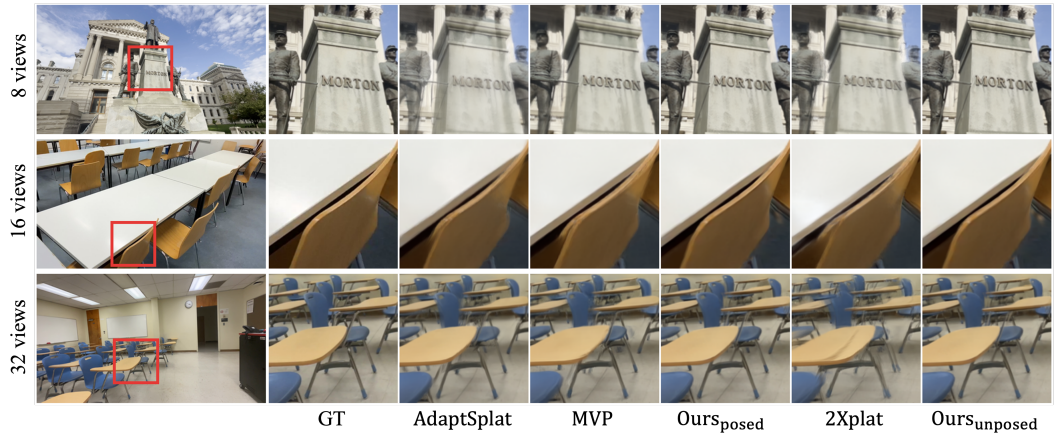


Figure 3: Qualitative NVS comparison at 960×540. AdaptSplat, MVP, and $\text{Ours}_{\text{posed}}$ use ground truth poses. 2Xplat fine tunes DA3 for camera prediction; $\text{Ours}_{\text{unposed}}$ refines predictions from frozen DA3. Full images are provided in Figure 6.

training, we freeze both heads and train the camera refiner at 480p and 960p on 8 GPUs for 20K iterations. The DA3 Giant backbone contains 1.10B parameters, and the depth and Gaussian heads contain 44.1M and 145.8M parameters, respectively. The camera refiner contains 18.3M parameters, including 11.7M in its frozen ResNet-18 encoder (He et al., 2016).

Datasets. We train HiFiNVS on DL3DV-10K (Ling et al., 2024). Following prior work (Kang et al., 2026b; Ye et al., 2026), we uniformly sample 8, 16, and 32 source images with maximum frame intervals of 50, 100, and 150, respectively, and randomly select 8 target images from the same interval, excluding the source images. We evaluate on the official 140-scene DL3DV benchmark using the same sampling protocol.

We evaluate generalization without additional training on Tanks & Temples and ScanNet++ (Riegler & Koltun, 2020; Yeshwanth et al., 2023). For Tanks & Temples, we follow the FVS evaluation protocol on Truck, M60, Playground, and Train. For each group of 8 official target images, we select 8, 16, or 32 source images according to view overlap. For ScanNet++, we use the 20-scene validation subset released with DA3-BENCH (Lin et al., 2026) and sample 8, 16, or 32 source images together with 8 target images from local iPhone trajectories.

Table 2: Novel view synthesis at 448×256 on DL3DV and Tanks & Temples with 8, 16, and 32 input views. Metrics, camera settings, and method notation follow Table 1. LagerNVS follows its official target-query protocol without EPA.

Method	DL3DV			Tanks & Temples		
	8V	16V	32V	8V	16V	32V
Posed (GT poses)						
$3DGS_{7k}$	22.53/0.750/0.238	25.94/0.837/0.168	28.61/0.907/0.121	14.93/0.457/0.447	17.12/0.611/0.335	20.02/0.746/**0.240**
DepthSplat	22.11/0.728/0.173	21.01/0.695/0.213	19.17/0.638/0.283	18.10/0.606/0.307	17.68/0.572/0.347	16.92/0.536/0.390
IDESplat	23.95/0.824/0.154	21.51/0.753/0.221	19.26/0.669/0.306	18.11/0.602/0.327	17.13/0.542/0.382	16.37/0.498/0.441
$ReSplat_{init}$	25.67/0.858/0.159	26.15/0.867/0.147	26.65/0.886/0.133	18.36/0.625/0.338	18.80/0.656/0.312	20.11/0.704/0.275
$ReSplat_{re}$	**27.36/0.889**/0.131	**28.10/0.901**/0.117	**28.84/0.920/0.102**	19.12/0.667/0.309	**19.66/0.699**/0.285	**21.14/0.755/0.240**
TokenGS	24.38/0.776/0.286	23.33/0.729/0.334	22.23/0.693/0.377	18.64/0.594/0.441	18.40/0.570/0.467	18.34/0.552/0.499
$Ours_{DA3}$	26.93/0.880/**0.117**	27.47/0.887/**0.112**	27.99/0.903/**0.102**	**19.36/0.693/0.259**	18.97/0.684/**0.278**	20.25/0.727/0.253
Unposed						
AnySplat	21.15/0.682/0.237	21.04/0.665/0.251	20.80/0.662/0.259	15.44/0.465/0.377	16.26/0.484/0.374	17.09/0.492/0.375
LagerNVS	23.72/0.722/0.182	22.99/0.688/0.197	22.88/0.691/0.195	18.38/0.524/0.302	17.96/0.499/0.318	18.00/0.488/0.326
$DA3_{GS}$	20.20/0.597/0.254	20.85/0.627/0.241	20.73/0.630/0.242	16.96/0.456/0.372	17.32/0.478/0.358	17.68/0.474/0.358
$ReSplat^{*}_{re}$	25.64/0.843/0.160	25.72/0.838/0.158	25.85/0.845/0.155	18.47/0.591/0.357	18.66/0.596/0.350	19.16/0.596/0.351
$Ours_{DA3}$	**26.85/0.874/0.121**	**27.17/0.875/0.119**	**27.48/0.887/0.113**	**19.31/0.671/0.269**	**18.89/0.664/0.288**	**19.91/0.689/0.273**

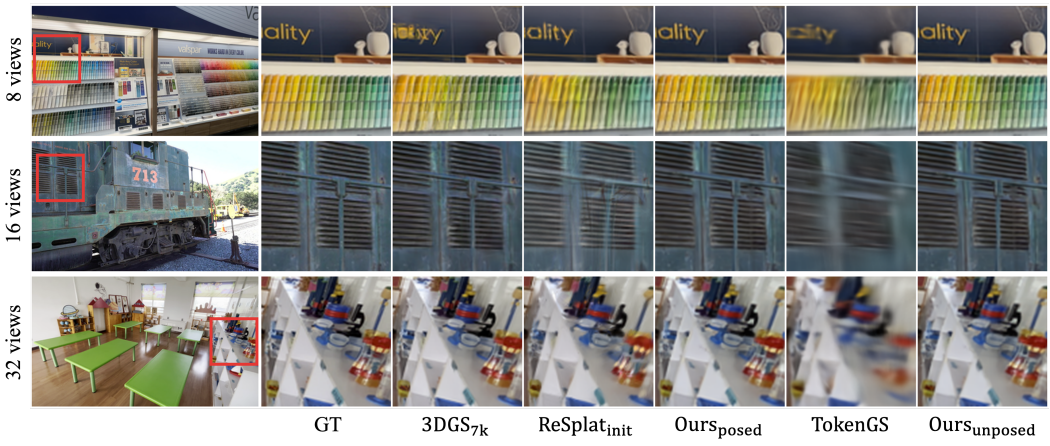


Figure 4: Qualitative NVS comparison at 448×256. All reconstruction methods use ground truth poses except $Ours_{unposed}$. Full images are provided in Figure 7.

Baselines. We benchmark HiFiNVS at 960×540 and 448×256 and include both posed and unposed methods. At 960×540, posed baselines include ReSplat (Xu et al., 2026), Long-LRM (Chen et al., 2025), tttLRM (Wang et al., 2026a), iLRM-wide (Kang et al., 2026a), AdaptSplat (Xing et al., 2026), and MVP (Kang et al., 2026b); 2Xplat (Jeong et al., 2026) is the unposed baseline. At 448×256, posed baselines include DepthSplat (Xu et al., 2025a), IDESplat (Long et al., 2026), ReSplat, and TokenGS (Ren et al., 2026); unposed baselines include AnySplat (Jiang et al., 2025), LagerNVS (Szymanowicz et al., 2026), and $DA3_{GS}$ (Lin et al., 2026). We also evaluate posed methods with DA3 camera predictions: $ReSplat_{re}$, tttLRM, and AdaptSplat at 960×540, and $ReSplat_{re}$ at 448×256. These variants are marked with *. We evaluate the baselines using their released checkpoints. $ReSplat_{init}$ and $ReSplat_{re}$ denote its initial and recurrent predictions.

For posed evaluation, we replace the predicted source cameras with GT cameras for graph construction and Gaussian reconstruction, without camera refinement. In the unposed setting, we rebuild the graph and Gaussian scene with refined cameras. Following Ye et al. (2026) and Jeong et al. (2026), we apply evaluation-time pose alignment (EPA) to target cameras for unposed evaluation, with the reconstructed scene fixed. LagerNVS follows its official target-query protocol without EPA. Evaluation details are provided in Appendix C.2.

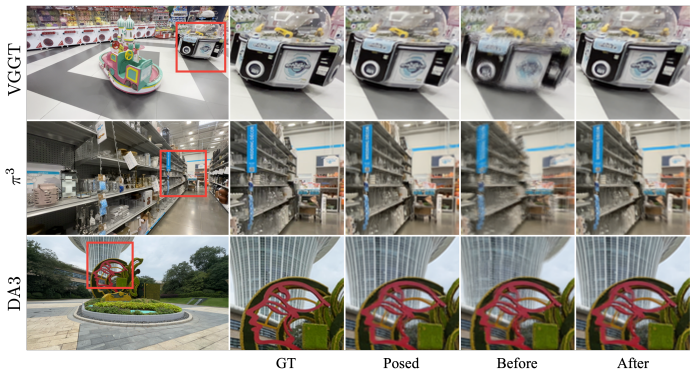


Figure 5: Camera refinement in HiFiNVS with VGGT, π^3, and DA3 using 16 input views. Posed, Before, and After use ground truth, predicted, and refined source cameras, respectively.

Table 3: Quantitative results of camera refinement across backbones on DL3DV with 16 views.

NVS quality (PSNR ↑)

Backbone	Posed	Before	After
VGGT	26.669	24.090	25.914
π^3	26.902	26.055	26.702
DA3	27.550	26.401	27.299

Source-camera accuracy (AUC ↑)

Backbone	AUC@3	AUC@5	AUC@15
VGGT	0.55→0.60	0.70→0.73	0.90→0.91
π^3	0.80→0.89	0.87→0.93	0.95→0.98
DA3	0.59→0.63	0.70→0.72	0.89→0.90

4.2 Feed-Forward Novel View Synthesis

Results at 960×540. Our method produces detailed reconstructions with fewer ghosting artifacts and cleaner object boundaries (Figure 3), consistent with the low LPIPS reported in Table 1. With GT source cameras, our method also achieves strong reconstruction quality. Among the compared feed-forward baselines, only ReSplat$_{\text{re}}$ directly updates Gaussian parameters using rendering errors. In the unposed setting, our camera refiner mitigates the effect of source camera errors, enabling the complete pipeline to surpass several baselines supplied with GT poses.

Results at 448×256. At 448×256, our method also delivers strong unposed reconstruction on DL3DV and Tanks & Temples (Table 2 and Figure 4). On DL3DV, reconstruction quality improves with additional input views, while the gap between posed and unposed results remains small. The Tanks & Temples results further support generalization beyond the training dataset.

4.3 Cross-Backbone Reconstruction and Camera Refinement

To evaluate our framework with different geometry backbones, we adapt HiFiNVS to VGGT and π^3. We initialize the Gaussian head and camera refiner from our DA3 model, then fine-tune the depth and Gaussian heads for 50K iterations and the camera refiner for 20K iterations. Table 3 reports NVS quality and source camera pose AUC on DL3DV at 480×270 with 16 input views. The results demonstrate that the same reconstruction design supports NVS with all three backbones. Camera refinement further improves pose accuracy and NVS quality in each instance, narrowing the gap to reconstruction with GT poses. Figure 5 shows corresponding reductions in blur and ghosting. Full results are provided in Appendix Table 8.

4.4 Generalizability of Camera Refinement

To evaluate generalization across NVS methods, we apply the same frozen camera refiner to MVP, AdaptSplat, ReSplat$_{\text{re}}$, and tttLRM without fine-tuning. We evaluate on DL3DV at 960×540 with 16 input views, initializing all pipelines with DA3 camera predictions. Each method provides source RGB and depth renderings for camera refinement. We backproject the rendered depths to obtain the 3D anchors used in Eq. (2). By operating on these renderings rather than internal reconstruction features, the refiner can be coupled to different NVS architectures. Each method then reconstructs the scene using the refined cameras. ReSplat$_{\text{re}}$ uses two recurrent iterations before and after camera refinement. Refinement improves PSNR and narrows the gap to posed reconstruction across all evaluated methods (Table 4), showing that the camera refiner generalizes to rendering feedback from different NVS models. See Table 10 and Figure 11 for additional results.

4.5 Runtime Breakdown

Table 5 reports runtime on DL3DV with 16 input views on a single NVIDIA A100 GPU. The breakdown separates initial reconstruction from feedback refinement. Evidence includes image feature extraction, reprojection, and construction of patch queries and evidence tokens. Refiner covers feedback aggregation and camera update prediction. Rebuild recomputes the graph and Gaussian

scene with refined cameras. FPS measures average rendering speed after Gaussian reconstruction. Backbone inference dominates initial reconstruction. By reusing backbone features and predicted depth, feedback refinement takes less time than initial reconstruction.

Table 4: Generalization of camera refiner (PSNR↑).

Method	Posed	Before	After
Ours_{DA3}	28.79	26.79	28.50
MVP	27.38	26.18	27.17
AdaptSplat	28.39	26.84	28.12
$\text{ReSplat}_{\text{re}}$	28.48	25.96	27.07
tttLRM	27.51	26.76	27.31

Table 5: Runtime breakdown on DL3DV at 16 views.

	Initial Gaussian construction (ms)					
Resolution	Backbone	Depth	Graph	Attention	Gaussian	Total
480×270	368.3	60.6	21.1	37.1	83.4	570.5
960×540	2050.6	241.4	71.0	142.2	308.9	2814.2
	Feedback refinement (ms)					
Resolution	Render	Evidence	Refiner	Rebuild	Total	FPS↑
480×270	85.1	175.9	14.4	140.4	415.8	197.40
960×540	240.5	461.4	38.0	520.5	1260.3	69.43

4.6 Ablation Studies

We train the NVS ablation models for 70K iterations and the camera refiner ablation models for 20K iterations on DL3DV using 8 GPUs. All ablations are evaluated at 480×270 with 16 input views.

Table 6: Component ablations on DL3DV with 16 input views. (a) Gaussian reconstruction variants. (b) Camera refiner variants use predicted poses and share the same frozen reconstruction model.

(a) Gaussian Reconstruction

Setting	PSNR↑	SSIM↑	LPIPS↓
w/o depth refinement	25.68	0.840	0.146
w/o depth residual	27.04	0.877	0.122
w/o token adaptation	26.28	0.870	0.130
w/o graph attention	26.35	0.870	0.131
Global + frame attention	26.47	0.873	0.128
Full	27.16	0.881	0.120

(b) Camera Refiner

Setting	PSNR↑	SSIM↑	LPIPS↓	AUC@5↑
Initial cameras	26.40	0.852	0.139	0.703
w/o rendering feedback	26.49	0.854	0.139	0.708
w/o image feature	26.50	0.853	0.138	0.707
w/o rendering residuals	27.22	0.875	0.124	0.722
Extrinsics only	26.98	0.870	0.127	0.724
w/o graph rebuilding	27.20	0.875	0.123	0.725
Full	27.28	0.877	0.122	0.725

Gaussian reconstruction. We ablate depth processing and token adaptation. "w/o depth refinement" uses frozen backbone depth D^{geo} and disables the Gaussian depth residual; "w/o depth residual" retains D but removes this residual. Figures 9 and 10 show depth predictions before and after fine-tuning. "w/o token adaptation" bypasses the adaptation module, while "w/o graph attention" retains only frame attention within its attention blocks. "Global + frame attention" replaces graph attention with global self-attention while retaining frame attention. Within our feature adaptation module, reprojection-guided attention yields better reconstruction quality than global self-attention.

Camera refiner. We ablate rendering evidence, camera updates, and graph rebuilding. For the three input ablations, we train separate refiners with the corresponding inputs removed during both training and evaluation. "w/o rendering feedback" excludes rendering evidence and yields reconstruction quality close to the initial result. "w/o image feature" removes ResNet features and their residuals while retaining RGB evidence. The smaller improvement from refinement highlights the importance of image features for interpreting rendering feedback. "w/o rendering residuals" removes RGB and feature residuals but retains the rendered appearance. For "Extrinsics only", we train a separate refiner to update only extrinsics, keeping the predicted intrinsics unchanged. "w/o graph rebuilding" uses the full refiner but reuses the initial graph connectivity after camera refinement during evaluation.

5 Conclusion

Limitations. Our method relies on the initial geometry predicted by the frozen backbone. Errors in depth or camera estimates can compromise both reprojection correspondences and rendering feedback, even with a fine-tuned depth head and a single camera update.

We present HiFiNVS, which exploits predicted geometry not only for reconstruction but also to establish explicit cross-view constraints for feature adaptation and camera refinement. HiFiNVS achieves state-of-the-art unposed NVS performance across the benchmarks. Camera refinement improves both pose accuracy and NVS quality across DA3, π^3, and VGGT. These results support a general framework for high-fidelity feed-forward NVS built on pretrained visual geometry backbones.

AI USE STATEMENT

We used LLMs to assist with developing evaluation scripts that integrate official implementations for consistent comparison across methods, reviewing experimental protocols and results, and editing the manuscript. The authors reviewed all AI-assisted content and take responsibility for the final manuscript.

REPRODUCIBILITY STATEMENT

Section 3 describes the model and training objectives, and Section 4.1 details the experimental settings. Appendices A, B, and C provide further details on the architecture, training, image preprocessing, and evaluation protocols. We plan to release model checkpoints and inference code publicly.

REFERENCES

Honggyu An, Jaewoo Jung, Mungyeom Kim, Chaehyun Kim, Minkyeong Jeon, Jisang Han, Kazumi Fukuda, Takuya Narihira, Hyunah Ko, Junsu Kim, Sunghwan Hong, Yuki Mitsufuji, and Seungryong Kim. Learning compact 3D representations from feed-forward novel view synthesis. In *CVPR*, 2026.

David Charatan, Sizhe Lester Li, Andrea Tagliasacchi, and Vincent Sitzmann. pixelSplat: 3D gaussian splats from image pairs for scalable generalizable 3D reconstruction. In *CVPR*, 2024.

Yu Chen and Gim Hee Lee. DBARF: Deep bundle-adjusting generalizable neural radiance fields. In *CVPR*, 2023.

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 *ECCV*, 2024.

Ziwen Chen, Hao Tan, Kai Zhang, Sai Bi, Fujun Luan, Yicong Hong, Fuxin Li, and Zexiang Xu. Long-LRM: Long-sequence large reconstruction model for wide-coverage gaussian splats. In *ICCV*, 2025.

Youming Deng, Songyou Peng, Junyi Zhang, Kathryn Heal, Tiancheng Sun, John Flynn, Steve Marschner, and Lucy Chai. Selfi: Self-improving reconstruction engine via 3D geometric feature alignment. In *CVPR*, 2026.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

Sunghwan Hong, Jaewoo Jung, Heeseong Shin, Jisang Han, Jiaolong Yang, Chong Luo, and Seungryong Kim. PF3plat: Pose-free feed-forward 3D gaussian splatting for novel view synthesis. In *ICML*, 2025.

Ranran Huang and Krystian Mikolajczyk. No pose at all: Self-supervised pose-free 3D gaussian splatting from sparse views. In *ICCV*, 2025.

Tooba Imtiaz, Lucy Chai, Kathryn Heal, Xuan Luo, Jungyeon Park, Jennifer Dy, and John Flynn. LVT: Large-scale scene reconstruction via local view transformers. In *SIGGRAPH Asia*, 2025.

Hwasik Jeong, Seungryong Lee, Gyeongjin Kang, Seungkwon Yang, Xiangyu Sun, Seungtae Nam, and Eunbyung Park. 2Xplat: Decoupling geometry and appearance modeling for feed-forward 3D gaussian splatting. *arXiv:2603.21064*, 2026.

Lihan Jiang, Yucheng Mao, Linning Xu, Tao Lu, Kerui Ren, Yichen Jin, Xudong Xu, Mulin Yu, Jiangmiao Pang, Feng Zhao, Dahua Lin, and Bo Dai. AnySplat: Feed-forward 3D gaussian splatting from unconstrained views. *ACM TOG*, 2025.

Gyeongjin Kang, Jisang Yoo, Jihyeon Park, Seungtae Nam, Hyeonsoo Im, Sangheon Shin, Sangpil Kim, and Eunbyung Park. SelfSplat: Pose-free and 3D prior-free generalizable 3D gaussian splatting. In *CVPR*, 2025.

Gyeongjin Kang, Seungtae Nam, Seungkwon Yang, Xiangyu Sun, Sameh Khamis, Abdelrahman Mohamed, and Eunbyung Park. iLRM: An iterative large 3D reconstruction model. In *CVPR*, 2026a.

Gyeongjin Kang, Seungkwon Yang, Seungtae Nam, Younggeun Lee, Jungwoo Kim, and Eunbyung Park. Multi-view pyramid transformer: Look coarser to see broader. In *CVPR*, 2026b.

Nikhil Keetha, Norman Müller, Johannes Schönberger, Lorenzo Porzi, Yuchen Zhang, Tobias Fischer, Arno Knapitsch, Duncan Zauss, Ethan Weber, Nelson Antunes, Jonathon Luiten, Manuel Lopez-Antequera, Samuel Rota Bulò, Christian Richardt, Deva Ramanan, Sebastian Scherer, and Peter Kontschieder. MapAnything: Universal feed-forward metric 3D reconstruction. In *3DV*, 2026.

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3D gaussian splatting for real-time radiance field rendering. *ACM TOG*, 2023.

Vincent Leroy, Yohann Cabon, and Jerome Revaud. Grounding image matching in 3D with MASt3R. In *ECCV*, 2024.

Hao Li, Yuanyuan Gao, Chenming Wu, Dingwen Zhang, Yalun Dai, Chen Zhao, Haocheng Feng, Errui Ding, Jingdong Wang, and Junwei Han. GGRt: Towards pose-free generalizable 3D gaussian splatting in real-time. In *ECCV*, 2024.

Chen-Hsuan Lin, Wei-Chiu Ma, Antonio Torralba, and Simon Lucey. BARF: Bundle-adjusting neural radiance fields. In *ICCV*, 2021a.

Haotong Lin, Sili Chen, Jun Hao Liew, Donny Y. Chen, Zhenyu Li, Yang Zhao, Sida Peng, Hengkai Guo, Xiaowei Zhou, Guang Shi, Jiashi Feng, and Bingyi Kang. Depth Anything 3: Recovering the visual space from any views. In *ICLR*, 2026.

Yen-Chen Lin, Pete Florence, Jonathan T. Barron, Alberto Rodriguez, Phillip Isola, and Tsung-Yi Lin. iNeRF: Inverting neural radiance fields for pose estimation. In *IROS*, 2021b.

Lu Ling, Yichen Sheng, Zhi Tu, Wentian Zhao, Cheng Xin, Kun Wan, Lantao Yu, Qianyu Guo, Zixun Yu, Yawen Lu, Xuanmao Li, Xingpeng Sun, Rohan Ashok, Aniruddha Mukherjee, Hao Kang, Xiangrui Kong, Gang Hua, Tianyi Zhang, Bedrich Benes, and Aniket Bera. DL3DV-10K: A large-scale scene dataset for deep learning-based 3D vision. In *CVPR*, 2024.

Yifan Liu, Keyu Fan, Weihao Yu, Chenxin Li, Hao Lu, and Yixuan Yuan. MonoSplat: Generalizable 3D gaussian splatting from monocular depth foundation models. In *CVPR*, 2025.

Wei Long, Haifeng Wu, Shiyin Jiang, Jinhua Zhang, Xinchun Ji, and Shuhang Gu. IDESplat: Iterative depth probability estimation for generalizable 3D gaussian splatting. In *CVPR*, 2026.

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 *ECCV*, 2020.

Guilherme Potje, Felipe Cadar, André Araujo, Renato Martins, and Erickson R. Nascimento. XFeat: Accelerated features for lightweight image matching. In *CVPR*, 2024.

René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. In *ICCV*, 2021.

Jiawei Ren, Michal Jan Tyszkiewicz, Jiahui Huang, and Zan Gojcic. TokenGS: Decoupling 3D gaussian prediction from pixels with learnable tokens. In *CVPR*, 2026.

Gernot Riegler and Vladlen Koltun. Free view synthesis. In *ECCV*, 2020.

Brandon Smart, Chuanxia Zheng, Iro Laina, and Victor Adrian Prisacariu. Splatt3R: Zero-shot gaussian splatting from uncalibrated image pairs. *arXiv:2408.13912*, 2024.

Brandon Smart, Chuanxia Zheng, Iro Laina, and Victor Adrian Prisacariu. Learning compact 3D gaussians via feed-forward point fusion. In *3DV*, 2026.

Stanislaw Szymanowicz, Minghao Chen, Jianyuan Wang, Christian Rupprecht, and Andrea Vedaldi. LagerNVS: Latent geometry for fully neural real-time novel view synthesis. In *CVPR*, 2026.

Zhenggang Tang, Yuchen Fan, Dilin Wang, Hongyu Xu, Rakesh Ranjan, Alexander Schwing, and Zhicheng Yan. MV-DUSt3R+: Single-stage scene reconstruction from sparse views in 2 seconds. In *CVPR*, 2025.

Chen Wang, Hao Tan, Yifan Wang, Zhiqin Chen, Yuheng Liu, Kalyan Sunkavalli, Sai Bi, Lingjie Liu, and Yiwei Hu. tttLRM: Test-time training for long context and autoregressive 3D reconstruction. In *CVPR*, 2026a.

Hengyi Wang and Lourdes Agapito. 3D reconstruction with spatial memory. In *3DV*, 2025.

Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. VGGT: Visual geometry grounded transformer. In *CVPR*, 2025a.

Qianqian Wang, Yifei Zhang, Aleksander Holynski, Alexei A. Efros, and Angjoo Kanazawa. Continuous 3D perception model with persistent state. In *CVPR*, 2025b.

Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jerome Revaud. DUSt3R: Geometric 3D vision made easy. In *CVPR*, 2024.

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

Mingwei Xing, Xinliang Wang, and Yifeng Shi. AdaptSplat: Adapting vision foundation models for feed-forward 3D gaussian splatting. *arXiv:2605.10239*, 2026.

Haofei Xu, Songyou Peng, Fangjinhua Wang, Hermann Blum, Daniel Barath, Andreas Geiger, and Marc Pollefeys. DepthSplat: Connecting gaussian splatting and depth. In *CVPR*, 2025a.

Haofei Xu, Daniel Barath, Andreas Geiger, and Marc Pollefeys. ReSplat: Learning recurrent gaussian splatting. In *ECCV*, 2026.

Jiale Xu, Shenghua Gao, and Ying Shan. FreeSplatter: Pose-free gaussian splatting for sparse-view 3D reconstruction. In *ICCV*, 2025b.

Jianing Yang, Alexander Sax, Kevin J. Liang, Mikael Henaff, Hao Tang, Ang Cao, Joyce Chai, Franziska Meier, and Matt Feiszli. Fast3R: Towards 3D reconstruction of 1000+ images in one forward pass. In *CVPR*, 2025.

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. In *ICLR*, 2025.

Botao Ye, Boqi Chen, Haofei Xu, Daniel Barath, and Marc Pollefeys. YoNoSplat: You only need one model for feedforward 3D gaussian splatting. In *ICLR*, 2026.

Chandan Yeshwanth, Yueh-Cheng Liu, Matthias Nießner, and Angela Dai. ScanNet++: A high-fidelity dataset of 3D indoor scenes. In *ICCV*, 2023.

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 *ECCV*, 2024.

Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018.

Shangzhan Zhang, Jianyuan Wang, Yinghao Xu, Nan Xue, Christian Rupprecht, Xiaowei Zhou, Yujun Shen, and Gordon Wetzstein. FLARE: Feed-forward geometry, appearance and camera estimation from uncalibrated sparse views. In *CVPR*, 2025.

A Architecture Details

We detail the depth head, Gaussian head, and camera refiner in execution order, using the notation of Section 3. The Gaussian head comprises feature adaptation and Gaussian decoding.

A.1 Depth Head

The depth head inherits the pretrained weights of the corresponding backbone's depth head and is fine tuned while the backbone remains frozen. From the backbone features, it predicts depth maps D for graph construction and Gaussian center placement, together with dense features that condition the Gaussian decoder. For π^3, projected decoder features are combined with the native point head outputs to provide 32 channels for Gaussian decoding. These features also predict a log-depth residual that adjusts the native depth prediction to obtain D.

A.2 Reprojection Graph and Feature Adaptation

Patch graph. On the backbone token grid with patch stride 14, we unproject a 7×7 grid of samples per patch using D and reproject them into the other source views. We discard projections outside the image or with relative depth error above 10%, and reject invalid or inconsistent normals estimated from depth (negative cosine similarity). Each remaining sample votes for the patch containing its projection. We retain up to 48 neighbors with the most votes, requiring at least five votes per neighbor. Depth and cameras are detached during graph construction.

Feature adaptation. Following DA3's DPT decoding design, we adapt four intermediate feature levels from each backbone. We select DA3 layers 19, 27, 33, and 39 and VGGT aggregation stages 4, 11, 17, and 23. For π^3, we concatenate each local block output at indices 8, 16, 26, and 34 with the following global block output. All indices start from 0. For VGGT and π^3, separate linear projections map each feature level from 2048 to 3072 dimensions to match the Gaussian head initialized from our DA3 model. At each patch, the four features are separately projected to 512 dimensions, concatenated, and fused by an MLP into a token of dimension 1024. We apply 2 pairs of graph attention and frame attention, implemented as Transformer convolution and standard Transformer blocks, respectively, with 8 attention heads. Finally, each original feature is concatenated with the updated token and linearly projected back to 3072 dimensions for Gaussian decoding.

A.3 Gaussian Decoder

Feature fusion and prediction. The DPT decoder produces dense features with 128 channels. An RGB encoder with 3 convolutional layers and GELU activations produces image features of the same dimension, which are added to the DPT output. We concatenate the resulting features with detached depth features with 32 channels from the depth head described in Section A.1. Two convolutional layers separated by ReLU map the fused features from 160 channels through 32 hidden channels to the Gaussian parameters.

Gaussian parameters. A scaled tanh bounds the log-depth residual, and no offset is predicted in the image plane. The scale outputs are mapped to positive values and scaled by the pixel footprint and corrected depth. Quaternions are normalized and converted to scene coordinates. Color and opacity use SH degrees 1 and 2 during rendering, respectively, with sigmoid activation for opacity.

A.4 Render-Aware Camera Refiner

Image features. We extract features from the stem and first 2 residual stages of the frozen ResNet-18 (He et al., 2016), yielding 64, 64, and 128 channels at strides 2, 4, and 8. Source images and their renders share the same encoder.

Feedback token construction. At each sampled location, Enc projects appearance, ray geometry, and 2D sinusoidal positional encodings to 64 dimensions and sums the results. Ray geometry consists of the camera center, unit ray direction, and their cross product. The binary validity flag is separately projected to 64 dimensions and added to this sum.

Invalid sample vectors are zeroed, and each vector is concatenated with its validity flag. The 49 resulting vectors are flattened in the source patch's sampling order and passed through a two-layer MLP with GELU and input/output layer normalization, producing a 256-dimensional token. Evidence tokens receive an additional layer normalization.

Source patch queries reuse the RGB, feature, ray, and position projections and the MLP. They use source appearance without residuals and skip the validity projection. All concatenated validity flags are set to one.

Camera prediction and updates. The refiner first combines rendering feedback with source patch queries to produce patch feedback tokens. Each patch query first attends to its corresponding evidence tokens. Self-attention then combines the updated queries within each source view. We perform these two steps twice, using 256-dimensional tokens and 8 attention heads.

Camera tokens then summarize the patch feedback within each view and exchange information across views. Each token is initialized by adding a shared learned embedding to an MLP encoding of the current camera. The MLP maps a 9-dimensional vector comprising the camera center, camera-to-world rotation quaternion, and vertical and horizontal fields of view to 256 dimensions using two linear layers and GELU. Each camera token reads its view's patch feedback tokens through two cross-attention blocks. A single global self-attention block then updates the camera tokens across views.

The camera decoder predicts per-view extrinsic residuals and a shared intrinsic residual from these camera tokens. The extrinsic head operates on each token, while the intrinsic head uses the mean token combined with an encoding of the initial log focal lengths and image-normalized principal point. Each extrinsic residual defines a rotation and translation update applied by left multiplication to the current world-to-camera transform. The rotation angle is bounded by 10° and the translation magnitude by 10% of the median pairwise camera baseline. Intrinsic updates are bounded by 0.15 in log focal length and 4% of the corresponding image dimension for principal point coordinates.

B Training Details

B.1 Backbone and Camera Settings

During Gaussian head and depth head training, the backbone takes only source images as input. We estimate a Sim(3) transformation from corresponding GT and predicted source poses and apply it to both GT source and target poses, placing them in the coordinate system and scale of the backbone predictions.

When training with GT source poses, the aligned source poses are used for graph construction and Gaussian reconstruction, and the aligned target poses are used for rendering supervision. Source and target views share intrinsics derived from the GT source intrinsics.

In the unposed setting, reconstruction uses predicted source poses, and target poses are initialized from the aligned GT poses. Both source and target views use shared intrinsics derived from the predicted source intrinsics.

With predicted source cameras, the aligned GT target poses may not precisely match the reconstructed scene. To reduce the effect of this mismatch on reconstruction learning, we apply matching-based target pose alignment (MatchBA) before computing the rendering loss. We use MatchBA instead of EPA during training because it does not require iterative backpropagation through the renderer.

MatchBA refines each target pose through two iterations of rendering, matching, and pose estimation. At each iteration, we render RGB and depth at the current pose and match the rendered image to the target image using XFeat (Potje et al., 2024). Rendered depth lifts the matched rendering pixels into 3D points, establishing 3D–2D correspondences with the target image. PnP with RANSAC estimates the updated pose, followed by reprojection refinement on inlier matches. The Gaussian scene and camera intrinsics remain fixed during alignment. The refined target poses are detached before computing the rendering loss.

B.2 Depth and Gaussian Head Training

We train the depth and Gaussian heads on DL3DV with the backbone frozen. For DA3, training uses 16 A800 GPUs and comprises 70K iterations at 448×256 followed by 30K iterations at 960×540. Batch sizes are 8/4/2 for 8/16/32 source views at low resolution and 2/1 for 8/16 source views at high resolution. Each example contains 8 target views for rendering supervision. For VGGT and π^3, we initialize the Gaussian head from our DA3 model and the depth head from the corresponding backbone. We fine tune both heads for 50K iterations while keeping the selected backbone frozen.

The rendering loss combines MSE and LPIPS with weights of 1 and 0.02. The base depth weight is $\lambda_d = 10^{-4}$, and the regularization weights are $\lambda_\delta = 5 \times 10^{-4}$ and $\lambda_\alpha = 10^{-3}$. The Huber thresholds are $\beta = 0.03$ for log-depth supervision and 0.1 for Gaussian log-depth residual regularization $\mathcal{R}_\delta$. The opacity regularizer $\mathcal{R}_\alpha$ penalizes the mean opacity evaluated along random viewing directions. At high resolution, MSE is computed over the full image, while LPIPS uses corresponding random crops of 448×448 pixels.

We use AdamW with a peak learning rate of 10^{-4}, $(\beta_1, \beta_2) = (0.9, 0.95)$, weight decay 0.01, and bfloat16 activations. The gradient norm is clipped at 10.

B.3 Camera Refiner Training

We freeze the reconstruction model and train the camera refiner for 20K iterations per stage. The DA3 refiner is first trained at 480p with 8, 16, or 32 source views. Its weights initialize the refiners for VGGT and π^3, as well as the DA3 refiner at 960p. Training at 960p uses 8 or 16 source views.

Training cameras. We construct training cameras by perturbing aligned GT cameras with errors derived from backbone predictions. For all source camera pairs, we compare relative rotations and scale-normalized baselines between the predicted and aligned GT cameras. We first fit rotation perturbations for each view, then camera center perturbations after accounting for the fitted rotations. Both fits use least squares with equal weights for all pairs and constrain the perturbations to sum to zero across views. These perturbations and the shared intrinsic discrepancy are applied to the aligned GT cameras to obtain the training inputs. The refiner uses rendering feedback under these cameras and is supervised against the aligned GT cameras.

Optimization. The refiner is trained with camera supervision only, using aligned GT cameras as targets. In the objective of Section 3.5, the pose loss combines the geodesic angle between refined and GT rotations with a Huber penalty on camera center errors, weighted by 2 and 1, respectively. Camera center errors are normalized by the median pairwise baseline of the aligned GT source and target cameras, with a Huber threshold of 0.005. The intrinsics loss has weight $\lambda_K = 0.25$ and applies Huber penalties to errors in log focal lengths and principal point coordinates normalized by image dimensions.

We use AdamW with cosine learning-rate decay. The peak learning rate is 2×10^{-4} for initial DA3 training and 5×10^{-5} for adaptation to other backbones or 960p, with a 5% warmup during adaptation. Other AdamW settings follow Section B.2. Activations use bfloat16; camera geometry and losses use float32.

C Evaluation Details

C.1 Official Models and Evaluation Settings

Table 7 summarizes the evaluated baselines and additional related methods.

Table 7: Availability and reported configurations of evaluated baselines and related methods. Slashes in the training-view column separate checkpoints.

Year	Method	Code / Weights	Reported resolution	Training input views
Evaluated baselines				

Year	Method	Code / Weights	Reported resolution	Training input views
2023	3DGS	Code / –	Variable	Variable
2024	DepthSplat	Code / Weights	448×256	2–6
2024	Long-LRM	Code / Weights	960×540	32
2025	AnySplat	Code / Weights	448 (long side)	2–24
2025	iLRM-wide	Code / Weights	960×540	16 / 32
2025	ReSplat	Code / Weights	448×256	8 / 16 / 32
			960×512	8
			960×540	16
2025	DA3 GS head	Code / Weights	Multi-resolution	Not specified
2025	MVP	Code / Weights	960×540	16, 32, 64, 128
2026	IDESplat	Code / Weights	448×256	2–6
2026	tttLRM	Code / Weights	960×536	16–64
2026	LagerNVS	Code / Weights	512 (long side)	1–10
2026	2Xplat	Code / Weights	960×540	16, 32
2026	TokenGS	Code / Weights	448×256	2 / 4 / 6
2026	AdaptSplat	Code / Weights	960×540	16–128
Additional related methods				
2024	NoPoSplat	Code / Weights	256×256; 512×512	2 / 3
2025	YoNoSplat	Code / Weights	224×224	2–32
2025	Selfi	–	518×294	6

$\text{ReSplat}_{\text{re}}$ uses 4 recurrent updates at 448p and 4/2/2 updates for 8/16/32 views at 960p. ReSplat releases five separate checkpoints across these settings. For 2Xplat, we use the original high-resolution checkpoint and retain its use of both source and target RGB images for camera estimation. The appearance head predicts Gaussians from source images conditioned on the resulting source camera estimates.

C.2 Evaluation Protocol

Image preprocessing. For evaluation at 448×256, images are resized to 480×270 and center-cropped to 448×256. ScanNet++ images are undistorted, cropped to the valid region, and resized to 1024×768, then resized and center-cropped to 960×540.

For HiFiNVS, images are resized to 448×252 or 952×532 for inference and rendering, ensuring dimensions divisible by 14. Rendered RGB images are resized back to the respective evaluation resolutions of 448×256 or 960×540 using bicubic interpolation. Camera intrinsics are adjusted for all input resizing and cropping.

Pose AUC. Following the DA3 implementation (Lin et al., 2026), we compute angular errors in relative rotation and translation direction against GT poses for all source camera pairs, accounting for sign ambiguity in translation direction. Each pair's pose error is the larger of the two errors. For each threshold $\tau \in \{3^\circ, 5^\circ, 15^\circ\}$, we bin these errors over $[0, \tau]$ at 1° intervals and normalize the counts by the total number of pairs in the scene. AUC is the mean of the cumulative bin values, and the reported scores are averaged over scenes.

Evaluation-time pose alignment. For unposed evaluation, we apply EPA to all methods except LagerNVS, optimizing target camera rotations and translations while keeping the reconstructed scene and camera intrinsics fixed. EPA minimizes the MSE between rendered and target RGB images using Adam for 100 iterations with a learning rate of 10^{-4}. This evaluation procedure is separate from our learned camera refinement.

For HiFiNVS, target poses are initialized by transforming GT target cameras into the reconstruction coordinate system using a Sim(3) fitted between GT and initially predicted source cameras. We evaluate the final target poses and perform alignment separately for each reconstructed scene, including before and after camera refinement in Table 3 and Table 8.

D Additional Quantitative Results

D.1 Full Results across Backbones

Table 8 reports the full reconstruction and camera refinement results across backbones with 8, 16, and 32 input views.

Table 8: Reconstruction and camera refinement across backbones on DL3DV at 480×270. NVS reports PSNR↑/LPIPS↓; Pose reports AUC@3/5/15↑.

Source cameras	Metric	HiFiNVS (VGGT)			HiFiNVS (π^3)			HiFiNVS (DA3)		
		8V	16V	32V	8V	16V	32V	8V	16V	32V
Posed (GT)	NVS	26.256/0.131	26.669/0.127	27.127/0.118	26.265/0.135	26.902/0.126	27.327/0.117	27.013/0.120	27.550/0.115	28.101/0.105
Backbone	NVS	24.445/0.177	24.090/0.192	24.029/0.196	25.528/0.155	26.055/0.147	26.217/0.144	26.300/0.135	26.401/0.139	26.304/0.145
	Pose	0.61/0.74/0.91	0.55/0.70/0.90	0.52/0.67/0.88	0.80/0.87/0.96	0.80/0.87/0.95	0.81/0.87/0.95	0.72/0.80/0.93	0.59/0.70/0.89	0.56/0.68/0.88
Refined	NVS	25.888/0.142	25.914/0.148	26.000/0.146	25.985/0.145	26.702/0.133	26.967/0.127	26.964/0.123	27.299/0.122	27.603/0.117
	Pose	0.68/0.79/0.93	0.60/0.73/0.91	0.56/0.69/0.89	0.90/0.94/0.98	0.89/0.93/0.98	0.88/0.92/0.97	0.75/0.82/0.94	0.63/0.72/0.90	0.60/0.71/0.89

D.2 Additional Unposed NVS Results

Table 9 supplements Tables 1 and 2 with results before camera refinement. Camera refinement substantially improves NVS quality on DL3DV, while also yielding consistent gains on Tanks & Temples and ScanNet++ in cross-dataset evaluation.

Table 9: Unposed NVS before and after source-camera refinement (PSNR↑/SSIM↑/LPIPS↓).

960×540, unposed NVS						
Method	DL3DV			ScanNet++		
	8V	16V	32V	8V	16V	32V
2Xplat	25.18/0.830/0.194	25.89/0.840/0.186	26.60/0.861/0.175	24.50/0.839/0.223	26.95/0.884/0.189	27.11/0.888/0.182
Ours_{DA3}-Before	26.78/0.854/0.170	26.79/0.845/0.176	26.29/0.832/0.191	25.17/0.864/0.201	26.05/0.875/0.194	25.44/0.851/0.209
Ours_{DA3}	28.17/0.891/0.141	28.50/0.890/0.140	28.58/0.895/0.138	26.54/0.896/0.180	27.76/0.909/0.170	28.16/0.908/0.165
448×256, unposed NVS						
Method	DL3DV			Tanks & Temples		
	8V	16V	32V	8V	16V	32V
Ours_{DA3}-Before	26.11/0.853/0.135	26.12/0.846/0.139	26.11/0.847/0.142	19.25/0.667/0.272	18.84/0.655/0.293	19.53/0.657/0.296
Ours_{DA3}	26.85/0.874/0.121	27.17/0.875/0.119	27.48/0.887/0.113	19.31/0.671/0.269	18.89/0.664/0.288	19.91/0.689/0.273

D.3 Additional Metrics for Camera Refiner Transfer

Table 10 supplements Table 4 with SSIM and LPIPS from the same DL3DV evaluation. All three metrics improve after camera refinement for each evaluated method.

Table 10: Camera refiner transfer on DL3DV at 960×540 with 16 views.

	Before			After		
Method	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
Ours_{DA3}	26.79	0.845	0.176	28.50	0.890	0.140
MVP	26.18	0.844	0.186	27.17	0.874	0.161
AdaptSplat	26.84	0.846	0.175	28.12	0.878	0.149
$\text{ReSplat}_{\text{re}}$	25.96	0.832	0.236	27.07	0.866	0.207
tttLRM	26.76	0.859	0.186	27.31	0.880	0.173

E Qualitative Results

Additional qualitative results are provided in the demo video.

E.1 Full Image NVS Comparisons

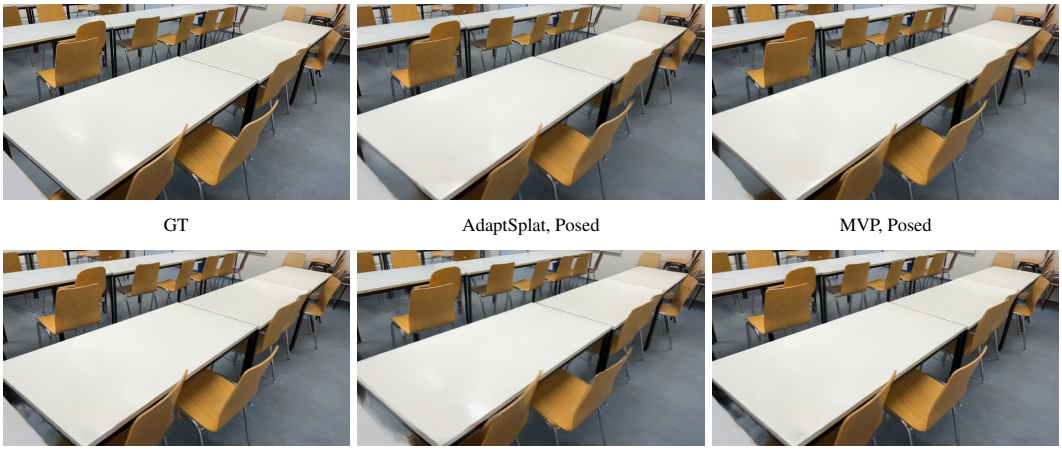
(a) DL3DV, 8 Input Views



GT | AdaptSplat, Posed | MVP, Posed

Ours, Posed | 2Xplat, Unposed | Ours, Unposed, with Refined Camera

(b) ScanNet++, 16 Input Views



GT | AdaptSplat, Posed | MVP, Posed

Ours, Posed | 2Xplat, Unposed | Ours, Unposed, with Refined Camera

(c) DL3DV, 32 Input Views



GT | AdaptSplat, Posed | MVP, Posed

Ours, Posed | 2Xplat, Unposed | Ours, Unposed, with Refined Camera

Figure 6: Full images corresponding to the 960×540 qualitative examples in Figure 3.

(a) DL3DV, 8 Input Views



GT | $3DGS_{7k}$, Posed | $ReSplat_{init}$, Posed



Ours, Posed | TokenGS, Posed | Ours, Unposed, with Refined Camera

(b) Tanks & Temples, 16 Input Views



GT | $3DGS_{7k}$, Posed | $ReSplat_{init}$, Posed



Ours, Posed | TokenGS, Posed | Ours, Unposed, with Refined Camera

(c) DL3DV, 32 Input Views



GT | $3DGS_{7k}$, Posed | $ReSplat_{init}$, Posed



Ours, Posed | TokenGS, Posed | Ours, Unposed, with Refined Camera

Figure 7: Full images corresponding to the 448×256 qualitative examples in Figure 4.

E.2 Reprojection Graph

We visualize the reprojection graph for two scenes, each with eight input views. Four spatially separated query patches are selected in the reference view, and their neighbors in the graph are marked with matching colors in the other views.

A query patch may span a depth discontinuity, so its sampled points can lie on surfaces at different depths. When reprojected into another view, these points can fall into spatially separated patches, connecting the query patch to multiple patches in that view. Markers are enlarged for visibility and do not indicate patch extent.

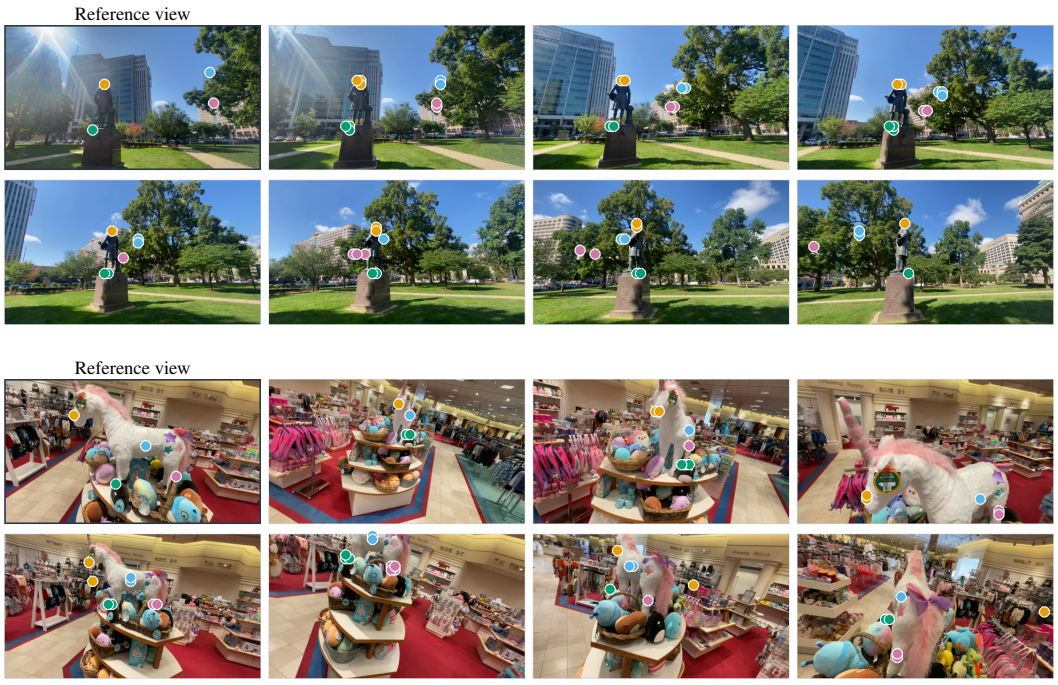


Figure 8: Reprojection graph visualization on two scenes. Colors identify four query patches in the reference view and their connected patches in the other input views. For visibility, the circular markers are slightly larger than the corresponding patches and may therefore overlap.

E.3 Depth Adaptation

Figure 9 shows that fine-tuning sharpens thin structures and makes the depth separation between foreground objects and distant backgrounds more distinct. Preserving these depth boundaries is important for novel view synthesis, as errors in the depths used to position Gaussians can alter their projections and occlusion relationships in other views.

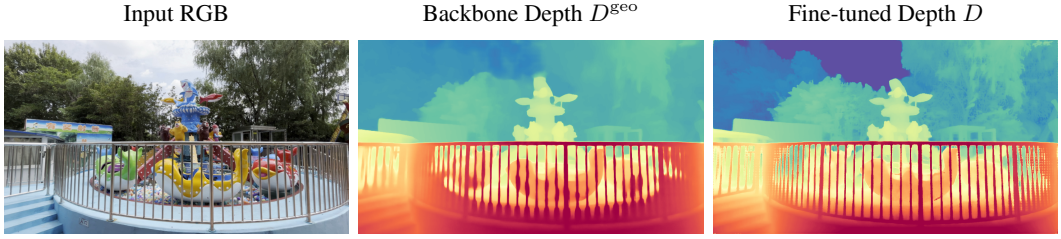
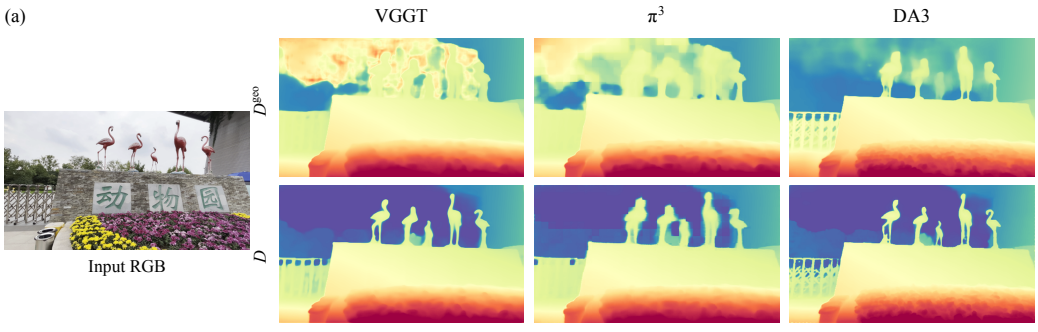




Figure 9: Depth predictions before and after fine-tuning. Each row shows an input image, frozen backbone depth D^{geo}, and fine-tuned depth D. Both depth predictions are visualized using identical inverse-depth normalization bounds and the same colormap within each scene.

Figure 10 provides additional examples of depth adaptation with VGGT, π^3, and DA3 at 480p.



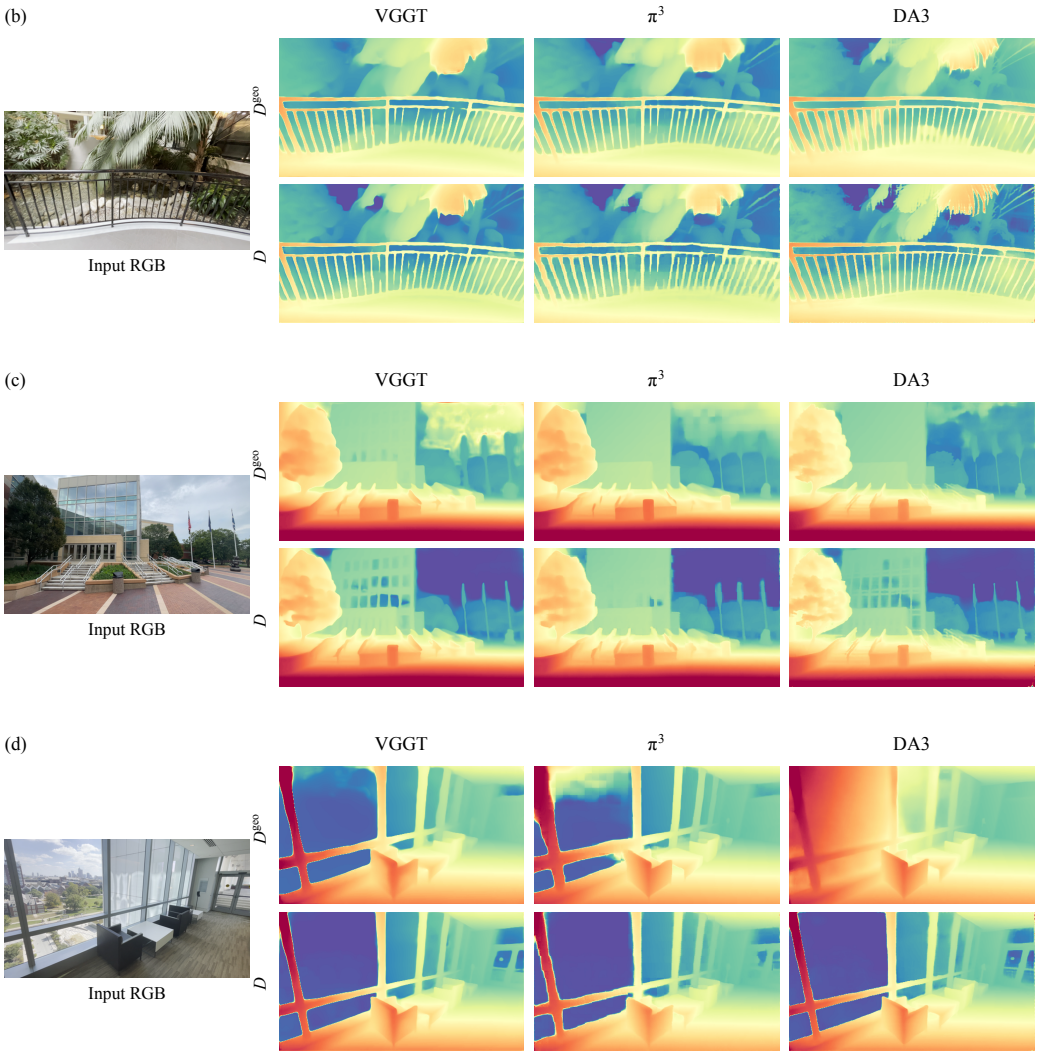
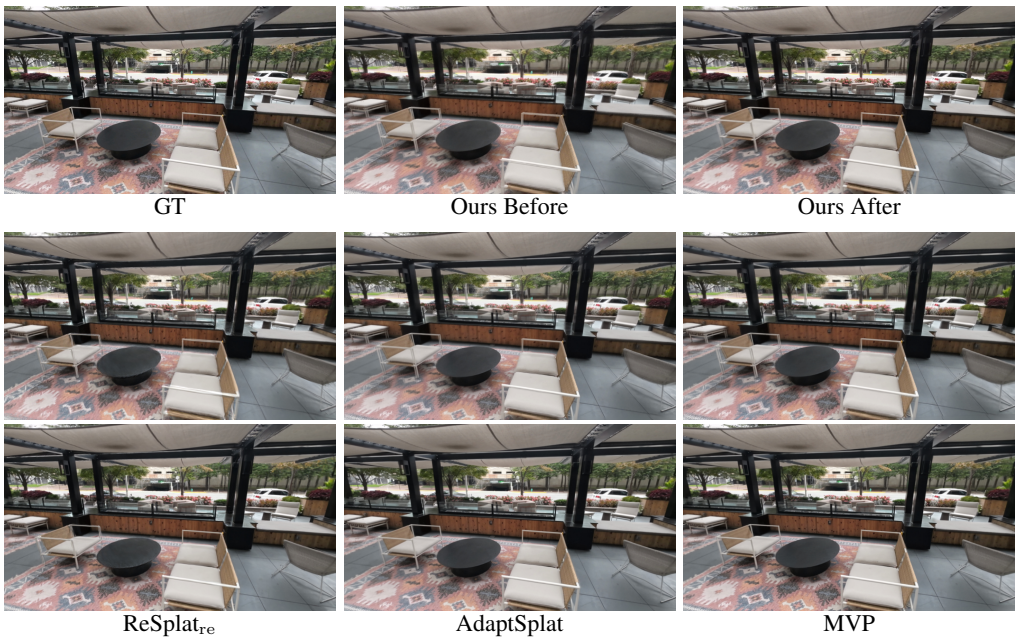


Figure 10: Depth adaptation across three backbones at 480p. Each example shows the input image and frozen backbone depth D^{geo} (top) and fine-tuned depth D (bottom) for each backbone. Visualization follows the normalization protocol described above.

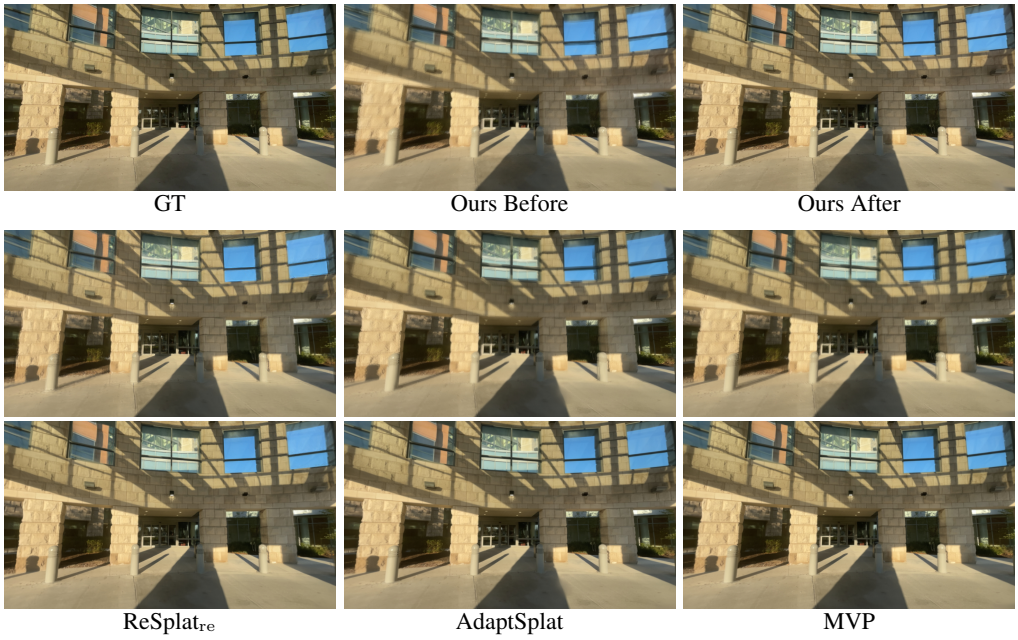
E.4 Camera Refinement

Figure 11 compares novel views before and after camera refinement for our model, ReSplat$_{\text{re}}$, AdaptSplat, and MVP. Following Section 4.4, the same frozen refiner updates source cameras using rendering feedback from each method's own reconstruction. Each method then reconstructs the scene with the refined cameras. In these examples, refinement reduces blur and ghosting around object boundaries and fine structures.

(a)


GT
Ours Before
Ours After
Before
After
ReSplat$_{re}$
AdaptSplat
MVP

(b)


GT
Ours Before
Ours After
Before
After
ReSplat$_{re}$
AdaptSplat
MVP

(c)

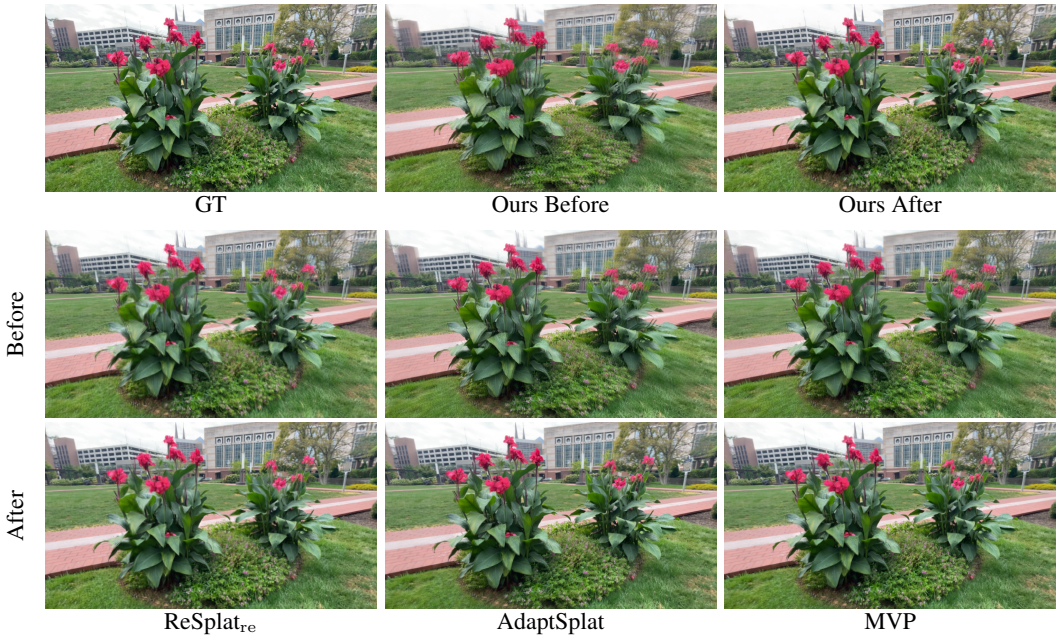


Figure 11: Camera refinement on three DL3DV scenes at 960×540 with 16 input views. Before and After use initial cameras and cameras refined by our refiner from each method's own rendering feedback, respectively. GT shows the target image.