PER-VIEW GAUSSIAN PREDICTIONS ENABLE TRAINING-FREE DISTRACTOR FILTERING IN FEED-FORWARD 3DGS

Anonymous authors
Paper under double-blind review

ABSTRACT

Feed-forward 3D Gaussian Splatting reconstructs an explicit Gaussian representation from multiple input images, making 3D reconstruction increasingly accessible for casual captures. However, such captures frequently contain transient content that is inconsistent across the input views. Such content can persist in the reconstructed Gaussian representation despite conflicting with observations from other views. As a result, it may produce blurred, duplicated, or floating artifacts in novel views. We introduce a training-free filtering procedure that exploits the association between predicted Gaussians and their corresponding input views. Self-view filtering compares each input with a rendering that excludes its associated Gaussians, exposing content unsupported by the remaining Gaussians. Cross-view filtering measures how those Gaussians interfere with reconstruction of other input views. Gaussians selected by either filtering path are removed together. The procedure operates on a single frozen prediction without retraining or scene-specific optimization. Evaluations across recent feed-forward 3DGS models on novel view synthesis benchmarks with distractors in the input images show improved rendering quality with varying numbers of input views. On clean inputs, the original reconstructions are largely preserved.

1 INTRODUCTION

Recent advances in neural radiance fields (Mildenhall et al., 2021) and 3D Gaussian Splatting (Kerbl et al., 2023) have substantially improved novel view synthesis. More recently, feed-forward 3DGS models directly predict a Gaussian representation from a set of input images (Charatan et al., 2024; Chen et al., 2024; Xu et al., 2025b; Ye et al., 2025b; Jiang et al., 2025), enabling fast and generalizable reconstruction.

These methods commonly assume that the input images depict a static scene observed consistently across views. Casual captures, however, often contain pedestrians, vehicles, and other transient content whose observations are inconsistent with a shared static scene. When incorporated into the predicted representation, such distractors can produce blurry regions, duplicated appearance, and floating artifacts in novel views, as illustrated in Figure 1.

Distractor handling has primarily been studied in optimization-based NeRF and 3DGS pipelines. Existing methods model transient appearance (Martin-Brualla et al., 2021), use robust objectives or uncertainty estimation (Sabour et al., 2023; Ren et al., 2024), leverage pretrained features or residual cues (Kulhanek et al., 2024; Sabour et al., 2025), separate static and transient content (Wang et al., 2025b; Lin et al., 2025; Park et al., 2026), or use consistency and multistage filtering (Li et al., 2025; Seo et al., 2026; Wang et al., 2026).

With the emergence of feed-forward reconstruction, recent work has introduced distractor-aware variants through dedicated training, learned mask prediction, reconstruction input selection, or Gaussian pruning (Bao et al., 2024; Gupta et al., 2026; Pan et al., 2026). These approaches address distractors by adapting reconstruction pipelines, using components such as learned mask prediction heads or pretrained segmentation models. Consequently, transferring distractor robustness to a different reconstruction backbone may require model-specific adaptation. In contrast, we ask whether such robustness can be obtained from the reconstruction itself, without modifying the model.

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

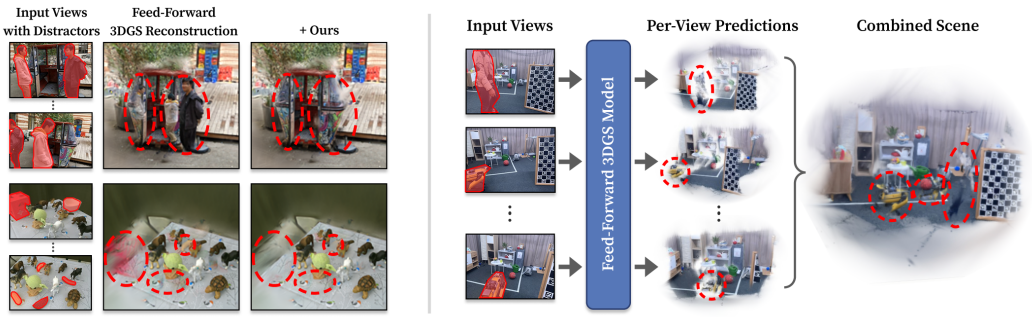


Figure 1: Distractor artifacts and the association between input views and Gaussian predictions. **Left:** Novel view renderings before and after filtering the same frozen prediction, with examples of transient content in the input images. **Right:** Gaussians predicted by a feed-forward 3DGS model can retain distractors observed in individual input views. These distractors can persist when the Gaussians are combined into a shared scene representation.

A natural starting point is the discrepancy between renderings and input images, as used in optimization-based pipelines. In the feed-forward setting, however, this cue can be obscured. The reconstruction models considered in this work retain an association between predicted Gaussians and their input views (Xu et al., 2025b;a; Ye et al., 2025a; Jiang et al., 2025). A model can therefore preserve content from an individual input even when that content is inconsistent with the remaining observations. The associated Gaussians may reproduce this content when rendered from the corresponding camera, yielding a rendering similar to the input even in distractor regions. The same Gaussians remain in the shared 3D representation and may nevertheless produce artifacts from other viewpoints.

Our key insight is to repurpose this per-view prediction structure as a detection cue from two perspectives. The association with input views allows us to selectively exclude or combine the Gaussians associated with each view. For each input view, excluding its associated Gaussians exposes content unsupported by the remaining representation. In other input views, adding those Gaussians to that view's own subset reveals whether they interfere with reproducing its input. Thus, the structure that can preserve inconsistent distractors also provides the means to identify and filter them.

Based on this insight, we propose a training-free procedure for filtering Gaussians associated with distractors from frozen feed-forward reconstruction models. Self-view and cross-view filtering assign discrepancy and interference scores, respectively, to Gaussians associated with each input. Each path produces a removal mask from its scores under observation and depth conditions. Gaussians selected by either mask are removed together from the original prediction.

Filtering operates on the final Gaussian output of the frozen reconstruction pipeline, without rerunning or modifying the model. Our method requires no additional training, learned distractor masks, or predefined distractor categories, and operates on the given inputs alone. Distractor filtering is thus added to already trained feed-forward 3DGS models as a post-processing step.

Our evaluation considers recent feed-forward 3DGS models, including DepthSplat (Xu et al., 2025b), ReSplat (Xu et al., 2025a), and YoNoSplat (Ye et al., 2025a). We evaluate novel view synthesis on RobustNeRF (Sabour et al., 2023) and NeRF On-the-go (Ren et al., 2024) using input images containing distractors, with varying numbers of input views. Each filtered reconstruction starts from the same frozen Gaussian prediction as its corresponding baseline, isolating the effect of filtering. Our method improves average reconstruction quality across these models and input counts. We also compare filtered AnySplat (Jiang et al., 2025) with GenWildSplat (Gupta et al., 2026), evaluate preservation on clean scenes, and examine the contribution and computational cost of filtering.

Our contributions are as follows:

- We show that the native association between input views and predicted Gaussians, which can allow distractors to persist in the reconstruction, can be repurposed as a cue for training-free distractor filtering.
- We propose a training-free distractor filtering method that exploits the per-view prediction structure, using discrepancies exposed by excluding each view's Gaussians and their effects on reconstruction of other input views.
- We demonstrate improvements in reconstruction quality across three models and two distractor benchmarks with varying input counts. On distractor-free inputs from four Robust-NeRF scenes, reconstruction quality remains close to the original predictions across four models.

2 Related Work

2.1 Feed-Forward 3D Gaussian Splatting

Following the development of neural radiance fields (Mildenhall et al., 2021; Barron et al., 2022), 3D Gaussian Splatting (Kerbl et al., 2023) has driven rapid advances in novel view synthesis across diverse scene settings and applications (Lu et al., 2024; Matsuki et al., 2024; Kheradmand et al., 2024; Wu et al., 2024; Qin et al., 2024; Liu et al., 2024; Xie et al., 2024). These approaches typically optimize a separate representation for each scene. A recent direction instead develops generalizable feed-forward models that directly predict Gaussian representations from input images, avoiding per-scene optimization.

pixelSplat (Charatan et al., 2024) introduced feed-forward Gaussian reconstruction from image pairs, while MVSplat (Chen et al., 2024) extended this formulation to sparse posed views. DepthSplat (Xu et al., 2025b) further combines Gaussian reconstruction with learned depth estimation. Subsequent methods support unposed or otherwise less constrained inputs (Ye et al., 2025b; Jiang et al., 2025; Ye et al., 2025a), recurrent refinement (Xu et al., 2025a), token-aligned prediction (Li et al., 2026), voxel-aligned prediction (Wang et al., 2025a), and adaptive subpixel primitive prediction (Moreau et al., 2026).

The reconstruction models considered in this work preserve an association between predicted Gaussians and their input views. Their explicit Gaussian outputs also allow selected subsets to be removed temporarily and rendered again. We use these properties for self-view filtering based on content unsupported by the remaining subsets and cross-view filtering based on a subset's effect on other input views.

2.2 Ignoring Distractors in 3D Reconstruction

Distractor-free novel view synthesis has primarily been studied through optimization-based NeRF and 3DGS pipelines. NeRF-based approaches model transient appearance (Martin-Brualla et al., 2021), suppress inconsistent observations with robust objectives (Sabour et al., 2023), or exploit uncertainty in casually captured scenes (Ren et al., 2024). Related 3D Gaussian Splatting methods leverage pretrained features to handle inconsistent observations (Kulhanek et al., 2024; Sabour et al., 2025), explicitly separate static and transient representations (Wang et al., 2025b; Lin et al., 2025; Park et al., 2026), or suppress unstable artifacts through consistency between two 3DGS models trained in parallel (Li et al., 2025). Other methods derive cleaner supervision through progressive or two-stage reconstruction (Seo et al., 2026; Wang et al., 2026). These approaches identify distractors while optimizing a scene-specific representation against its input images.

Recent work has also introduced feed-forward reconstruction models trained specifically for distractor handling. VGTW (Pan et al., 2026) builds on VGGT and introduces distractor-aware training together with an auxiliary mask head supervised by pixel-level annotations. DGGS (Bao et al., 2024) builds on MVSplat (Chen et al., 2024) and uses reference-based mask prediction and refinement during training. Its reported inference procedure uses the predicted masks to score and reselect references before pruning Gaussians associated with distractors. GenWildSplat (Gupta et al., 2026), based on AnySplat (Jiang et al., 2025), uses pretrained semantic segmentation for transient-object masking together with curriculum learning on synthetic and real data.

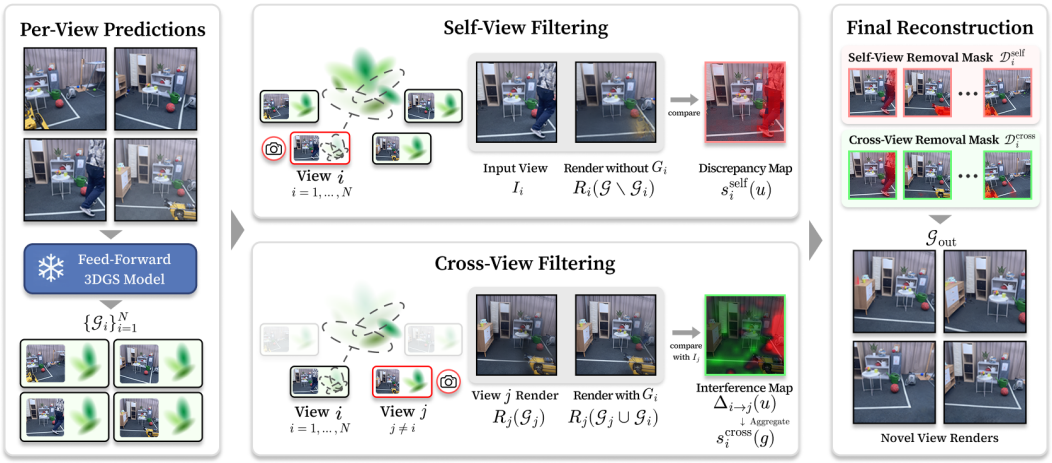


Figure 2: Overview of our method. For each input view i, we apply self-view and cross-view filtering to its associated Gaussians $\mathcal{G}_i$. Self-view filtering compares I_i with a rendering that excludes $\mathcal{G}_i$. Cross-view filtering compares renderings at another input view j using $\mathcal{G}_j$ alone and together with $\mathcal{G}_i$. Red regions in the interference map indicate degraded reconstruction after including $\mathcal{G}_i$, measured by lower similarity to I_j. Map values are assigned to Gaussians in $\mathcal{G}_i$, with cross-view values aggregated across views. Each filtering path selects Gaussians for removal using its scores and validity conditions. Gaussians selected by either path are removed from the original prediction to obtain the final representation for novel view rendering.

These approaches incorporate distractor handling into the reconstruction pipeline itself, relying on additional trained components or backbone-specific adaptation. Our method instead decouples distractor handling from the reconstruction model: it operates on one Gaussian prediction produced from a fixed set of inputs, with no additional training, predefined distractor categories, or learned mask prediction. Under its reported evaluation protocol, DGGS assumes access to a scene image pool around each query view, from which reconstruction inputs are scored and reselected before reconstruction is executed again. Our method assumes only a fixed input set: filtering operates on the resulting Gaussian representation without rerunning the reconstruction model.

3 Method

3.1 Preliminaries

3D Gaussian Splatting. 3D Gaussian Splatting (Kerbl et al., 2023) represents a scene as a set of N_g Gaussian primitives $\mathcal{G} = \{g_n\}_{n=1}^{N_g}$, each carrying a position, covariance, opacity, and view-dependent color. We denote by $\mathcal{R}(\mathcal{G}, P)$ the image rendered from $\mathcal{G}$ under camera parameters P via depth-ordered alpha compositing. For a Gaussian set $\mathcal{A}$, we write $R_i(\mathcal{A}) := \mathcal{R}(\mathcal{A}, P_i)$ for its rendering at input camera P_i. The explicit primitive-based representation allows selected Gaussian subsets to be excluded or combined and the resulting representation rendered directly.

Feed-Forward 3DGS. Given N input images and their camera parameters,

$$\mathcal{X} = \{(I_i, P_i)\}_{i=1}^{N}, \tag{1}$$

a feed-forward reconstruction pipeline F_θ predicts a Gaussian representation (Charatan et al., 2024; Chen et al., 2024; Xu et al., 2025b). For the models considered here, its final output preserves an association between Gaussians and input views:

$$\{\mathcal{G}_i\}_{i=1}^{N} = F_\theta(\mathcal{X}), \qquad \mathcal{G} = \bigcup_{i=1}^{N} \mathcal{G}_i. \tag{2}$$

Here, $\mathcal{G}_i$ is the subset of Gaussians associated with input view i. We use the final output of the frozen baseline.

Table 1: Reconstruction quality on RobustNeRF and NeRF On-the-go with 4, 8, and 16 input views. Results are averaged over scenes within each dataset. Each + Ours entry filters the corresponding baseline prediction using identical inputs and model weights.

	4 views			8 views			16 views		
Method	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
RobustNeRF dataset									
DepthSplat	20.10	0.738	0.267	19.91	0.706	0.288	18.44	0.644	0.369
+ Ours	22.00	0.768	0.219	21.52	0.738	0.238	20.15	0.697	0.282
YoNoSplat	21.70	0.722	0.260	23.12	0.776	0.230	22.71	0.733	0.249
+ Ours	22.98	0.735	0.235	24.04	0.794	0.199	23.15	0.741	0.231
ReSplat	22.25	0.778	0.270	22.73	0.763	0.280	24.71	0.797	0.238
+ Ours	22.93	0.783	0.264	23.11	0.768	0.273	24.83	0.799	0.235
NeRF On-the-go dataset									
DepthSplat	17.86	0.549	0.306	17.76	0.524	0.336	16.32	0.447	0.416
+ Ours	18.52	0.568	0.279	19.00	0.554	0.287	17.95	0.503	0.332
YoNoSplat	18.35	0.547	0.330	18.66	0.561	0.326	17.95	0.532	0.350
+ Ours	19.44	0.573	0.296	20.05	0.588	0.287	19.08	0.562	0.305
ReSplat	18.87	0.618	0.341	19.13	0.610	0.341	19.45	0.625	0.319
+ Ours	19.32	0.629	0.330	20.11	0.627	0.324	20.05	0.638	0.306

3.2 Problem Setup

Our goal is to identify and remove Gaussians associated with distractors, defined as transient content whose observations across input views are inconsistent with a shared static scene. Filtering uses only the fixed inputs and the frozen baseline's prediction, without clean reference images, distractor masks, held-out evaluation images, or additional images for input selection. It neither reruns the reconstruction pipeline nor optimizes its parameters or the predicted representation.

Figure 2 outlines the procedure. Grouping Gaussians by their associated input view lets us examine each subset from that camera and from other input cameras. Self-view filtering measures how well the remaining Gaussians explain the source input when its associated subset is excluded. Cross-view filtering measures how adding that subset to another view's subset changes rendering similarity to that view's input. Each filtering path produces a removal mask using its scores and validity conditions, detailed in Section 4. We remove Gaussians selected by either mask.

3.3 Self-View Filtering

Content supported primarily by $\mathcal{G}_i$ may be inconsistent with the rendering from the remaining Gaussians. We expose this discrepancy by excluding $\mathcal{G}_i$ and comparing the resulting rendering with input I_i. Let $\mathcal{S}(I, R)$ denote a similarity map between images I and R, where u indexes image patches. Our implementation computes $\mathcal{S}(I, R)(u)$ as the cosine similarity between their frozen DINOv3 features (Siméoni et al., 2025) at patch u, yielding values in $[-1, 1]$. The discrepancy map is defined as:

$$s_i^{\text{self}}(u) = 1 - \mathcal{S}\big(I_i, R_i(\mathcal{G} \setminus \mathcal{G}_i)\big)(u). \tag{3}$$

A high score indicates content that Gaussians associated with other inputs fail to reproduce. To evaluate discrepancies in regions rendered by the remaining Gaussians, we select patches with sufficient accumulated opacity. To focus on content whose removal reveals a surface farther from the camera, we require the rendering without $\mathcal{G}_i$ to have greater depth on average than the full rendering. We transfer the patch scores to source Gaussians through their corresponding image locations, obtaining $s_i^{\text{self}}(g)$ for $g \in \mathcal{G}_i$. The score is valid only for patches satisfying the opacity and depth conditions detailed in Section 4.

3.4 CROSS-VIEW FILTERING

For each other input view $j \in \{1, \dots, N\} \setminus \{i\}$, we render $\mathcal{G}_j$ alone and together with the subset $\mathcal{G}_i$, both at camera P_j. Their similarities to the same input I_j define an interference map:

$$\Delta_{i \to j}(u) = \mathcal{S}\big(I_j, R_j(\mathcal{G}_j)\big)(u) - \mathcal{S}\big(I_j, R_j(\mathcal{G}_j \cup \mathcal{G}_i)\big)(u). \tag{4}$$

Positive values indicate that adding $\mathcal{G}_i$ reduces similarity to input I_j; negative values indicate an increase. For each Gaussian $g \in \mathcal{G}_i$, we project its center into view j and sample the interference map at that location, obtaining $\Delta_{i \to j}(g)$. This value represents the effect of $\mathcal{G}_i$ at the projected location of g.

To assess whether g can interfere with reconstruction in view j, we require it to project within the image and lie no farther from camera P_j than the corresponding Gaussian in $\mathcal{G}_j$. Let $\mathcal{V}_i(g)$ contain the indices of input views $j \neq i$ satisfying the geometric conditions in Section 4, and let $\mathcal{F}_i(g) \subseteq \mathcal{V}_i(g)$ contain those that also satisfy the depth condition. For nonempty $\mathcal{V}_i(g)$, we aggregate the sampled interference values as

$$s_i^{\text{cross}}(g) = \frac{\sum_{j \in \mathcal{F}_i(g)} \Delta_{i \to j}(g)}{|\mathcal{V}_i(g)|}. \tag{5}$$

Otherwise, we set $s_i^{\text{cross}}(g) = 0$. Normalizing by $|\mathcal{V}_i(g)|$ makes the score reflect both the measured interference and the fraction of geometrically valid views satisfying the depth condition. Views that fail this condition contribute zero to the average. The score is valid only if at least one view satisfies the depth condition.

3.5 DISTRACTOR PRUNING AND RECONSTRUCTION

We select Gaussians for removal when, in either filtering path, they satisfy the corresponding conditions and their scores reach or exceed the corresponding threshold. Let $v_i^{\text{self}}(g)$ and $v_i^{\text{cross}}(g)$ indicate whether g satisfies the conditions for each filtering path, taking value 1 if satisfied and 0 otherwise. The self-view and cross-view removal masks select the following Gaussian sets, respectively:

$$\begin{aligned} \mathcal{D}_i^{\text{self}} &= \{g \in \mathcal{G}_i \mid v_i^{\text{self}}(g) = 1,\ s_i^{\text{self}}(g) \geq \tau_{\text{self}}\}, \\ \mathcal{D}_i^{\text{cross}} &= \{g \in \mathcal{G}_i \mid v_i^{\text{cross}}(g) = 1,\ s_i^{\text{cross}}(g) \geq \tau_{\text{cross}}\}. \end{aligned} \tag{6}$$

We remove Gaussians selected by either mask, yielding

$$\mathcal{G}_{\text{out}} = \mathcal{G} \setminus \bigcup_{i=1}^{N} \left(\mathcal{D}_i^{\text{self}} \cup \mathcal{D}_i^{\text{cross}}\right). \tag{7}$$

Gaussians selected by neither mask are retained. Both masks are computed from the original prediction, and selected Gaussians are removed without additional model inference, optimization, or iterative removal. Novel views are rendered directly from $\mathcal{G}_{\text{out}}$ using the baseline's native renderer.

4 EXPERIMENTS

Experimental Setup. Our evaluation protocol uses the RobustNeRF (Sabour et al., 2023) and NeRF On-the-go (Ren et al., 2024) benchmarks with 4, 8, and 16 input views containing distractors, covering sparse to moderately dense capture settings. We also evaluate reconstruction quality using distractor-free inputs from the same RobustNeRF scenes. All experiments use four fixed configurations of input and test views for each scene and input count. All reported quality metrics and runtimes are averaged over these configurations. Each configuration starts from an input view and a test view with high overlap in the visible COLMAP (Schonberger & Frahm, 2016) points, and views are added to increase the scene coverage jointly observed by both sets. Test images are used only for evaluation.

We report PSNR, SSIM (Wang et al., 2004), and LPIPS (Zhang et al., 2018) on the test images. Each reconstruction model is evaluated at its configured inference resolution. Experiments use NVIDIA RTX 3090 and RTX 4090 GPUs; runtime comparisons use a single RTX 4090.

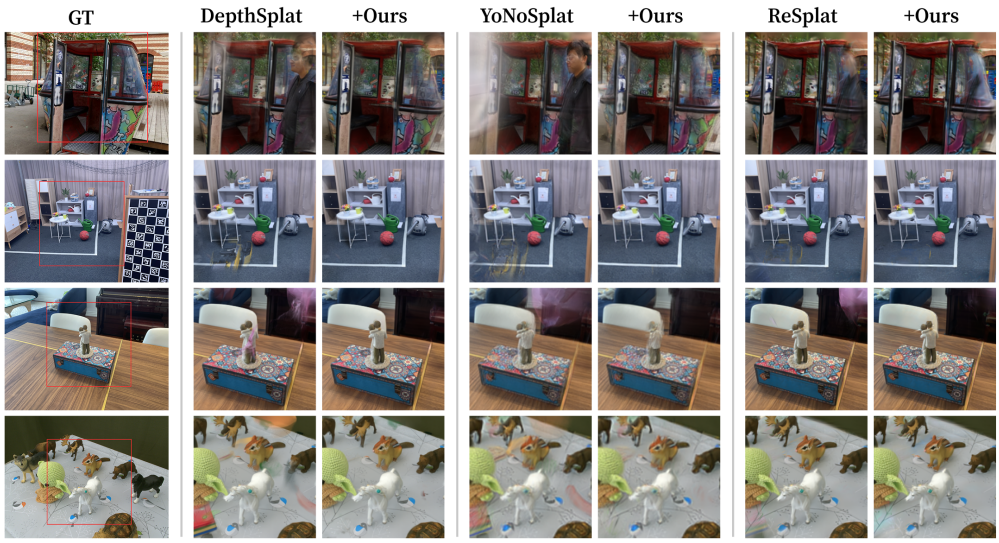


Figure 3: Qualitative comparisons of novel view renderings from DepthSplat, YoNoSplat, and ReSplat before and after filtering. Red boxes in the ground-truth images mark the regions shown in the corresponding renderings. Each **+Ours** result is obtained from the same Gaussian prediction as its baseline.

Implementation Details. We use frozen DINOv3 ViT-B/16 features, native baseline renderers, and fixed thresholds $\tau_{\text{self}} = 0.50$ and $\tau_{\text{cross}} = 0.05$ for all scenes and models. For self-view filtering, we require mean accumulated opacity > 0.50 in the rendering without $\mathcal{G}_i$. We compute the mean depth difference over pixels with finite, positive depths in both renderings and exclude patches without such pixels. Depth follows the convention of each renderer.

For cross-view filtering, geometric validity requires positive opacity for both the source Gaussian and its counterpart in $\mathcal{G}_j$, finite, positive depths at camera P_j, and a finite source projection within the image bounds. We sample interference maps bilinearly at projected source centers.

Both paths operate on the original prediction. Selected opacities are set to zero, leaving all other Gaussian attributes unchanged.

Reconstruction Models and Comparisons. We apply our method to the released DepthSplat (Xu et al., 2025b), ReSplat (Xu et al., 2025a), and YoNoSplat (Ye et al., 2025a) models. DepthSplat uses its DL3DV checkpoint trained with two to six views for all input counts, so the eight- and sixteen-view settings test transfer to larger input sets. ReSplat uses its 8-view checkpoint for up to 8 inputs and its 16-view checkpoint for 16.

Each paired comparison uses identical images, cameras, model weights, and the same final Gaussian prediction; only the filtering decision differs, isolating the effect of removal. For the comparison with GenWildSplat (Gupta et al., 2026), we use four and six input views with AnySplat (Jiang et al., 2025) and its estimated cameras. GenWildSplat targets unconstrained image collections and uses semantic masks for predefined transient object categories. This experiment compares its reconstruction quality with that of our filtered AnySplat predictions. DGGS (Bao et al., 2024) is excluded, as no implementation or weights were publicly available at submission and its inference requires additional scene images beyond the given inputs.

Quantitative Results. Table 1 compares each reconstruction before and after filtering. Since each pair shares the same images, model weights, and Gaussian prediction, the comparison isolates the effect of filtering. Filtering improves PSNR, SSIM, and LPIPS averaged over each dataset for all three models at every evaluated input count.

Qualitative Results. Figure 3 compares the original and filtered reconstructions across three reconstruction models. Filtering suppresses blurred, duplicated, or floating artifacts associated with

Table 2: Ablation of the filtering components and image similarity measure with four input views. Results are averaged over the three reconstruction models and scenes from both distractor benchmarks.

Variant	PSNR↑	SSIM↑	LPIPS↓
Vanilla	19.56	0.641	0.301
Component Ablation			
Cross-View Only	20.36	0.656	0.278
Self-View Only	20.17	0.651	0.287
Similarity Metric			
w/ CLIP	20.25	0.652	0.288
w/ L_1	20.32	0.652	0.288
Full Method	20.51	0.658	0.277

Table 3: Reconstruction quality of AnySplat, AnySplat + Ours, and the released GenWildSplat model with four and six input views. + Ours filters the frozen AnySplat prediction.

Method	Views	PSNR↑	SSIM↑	LPIPS↓
RobustNeRF dataset				
AnySplat	4	13.96	0.460	0.491
+ Ours	4	14.59	0.482	0.451
GenWildSplat	4	13.99	0.484	0.516
AnySplat	6	15.75	0.506	0.439
+ Ours	6	16.72	0.533	0.386
GenWildSplat	6	14.39	0.482	0.532
NeRF On-the-go dataset				
AnySplat	4	13.66	0.285	0.469
+ Ours	4	14.06	0.295	0.438
GenWildSplat	4	13.50	0.315	0.544
AnySplat	6	14.91	0.324	0.440
+ Ours	6	15.51	0.336	0.400
GenWildSplat	6	13.73	0.315	0.546

transient content while largely preserving static scene structure. These results show that filtering the frozen predictions of different models can improve the visual quality of novel view renderings.

Ablation Studies. Table 2 compares self-view filtering alone, cross-view filtering alone, and their combination (Full Method) against the unfiltered reconstruction. Results use four input views and are averaged over scenes from RobustNeRF and NeRF On-the-go and the three reconstruction models. Both filtering paths improve reconstruction quality independently, while combining them gives the best results across all three metrics. We also evaluate CLIP (Radford et al., 2021) features and a simple L_1 image error as alternatives to our default comparison using DINOv3. The default setting achieves the best results across all three metrics, while both alternatives improve over the unfiltered baseline. These results suggest that our filtering procedure can identify distractors using other pretrained features or even a simple photometric error. Appendix A.1 reports component ablations with additional input counts.

Comparison with In-the-Wild Feed-Forward Model. Table 3 compares AnySplat, its filtered prediction, and GenWildSplat on the two distractor benchmarks using four and six input views. + Ours filters the frozen AnySplat prediction; for GenWildSplat, we use its released model. The protocol uses the cameras estimated by AnySplat for reconstruction and filtering. GenWildSplat builds on AnySplat and uses semantic masks for predefined transient categories. We include its released model as a reference for reconstruction quality on inputs containing distractors. Filtering improves AnySplat across all three metrics on both datasets with four and six inputs. Compared with GenWildSplat, our method achieves higher PSNR and lower LPIPS in all four settings; SSIM is higher with six inputs on both datasets.

Preservation on Clean Scenes. Table 4 and Figure 4 evaluate preservation on distractor-free inputs from the four RobustNeRF scenes using either known or estimated camera poses. Our results remain close to the unfiltered baselines across all four reconstruction models. In Figure 4, red overlays mark regions selected for removal. GenWildSplat's semantic masks can cover static objects belonging to its predefined transient categories, whereas our filtering largely preserves content consistent across the input views.

Overhead Analysis. Table 5 reports baseline inference, filtering, and total runtime. Filtering includes rendering, feature extraction, and Gaussian removal. Across the evaluated models, filtering takes 0.105–0.179 seconds with four input views and 1.125–2.151 seconds with sixteen input views. Self-view filtering renders each input camera with and without its associated Gaussians. Cross-view

Table 4: Reconstruction quality on distractor-free inputs from four RobustNeRF scenes, using ground-truth or estimated camera poses.

	4 views			6 views		
Method	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
With GT Pose						
DepthSplat	23.14	0.804	0.169	22.04	0.769	0.197
+ Ours	23.08	0.804	0.171	22.10	0.770	0.198
YoNoSplat	24.98	0.802	0.167	25.23	0.812	0.163
+ Ours	24.97	0.801	0.169	25.18	0.811	0.165
ReSplat	24.88	0.809	0.224	24.47	0.798	0.232
+ Ours	24.85	0.809	0.225	24.41	0.798	0.233
With Estimated Pose						
AnySplat	16.39	0.505	0.411	17.68	0.557	0.349
+ Ours	16.34	0.504	0.412	17.58	0.555	0.351
GenWildSplat	14.58	0.507	0.483	14.81	0.506	0.487

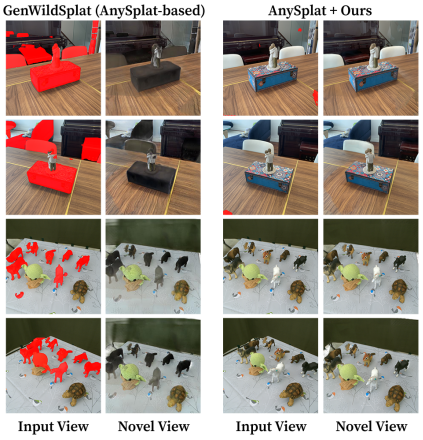


Figure 4: GenWildSplat and AnySplat + Ours on distractor-free inputs. Red indicates regions selected for removal.

Table 5: Mean processing time (s) on one RTX 4090, averaged over scenes from RobustNeRF and NeRF On-the-go datasets. Filtering includes both paths and Gaussian removal.

	4 views			8 views			16 views		
Method	Inference	Filtering	Total	Inference	Filtering	Total	Inference	Filtering	Total
ReSplat	0.167	0.105	0.272	0.290	0.316	0.606	0.585	1.125	1.711
DepthSplat	0.098	0.179	0.277	0.148	0.594	0.742	0.279	2.151	2.429
YoNoSplat	0.222	0.135	0.358	0.260	0.435	0.695	0.390	1.624	2.014

filtering renders $\mathcal{G}_j$ and $\mathcal{G}_j \cup \mathcal{G}_i$ at camera P_j for each pair of input views $i \neq j$. Filtering produces $\mathcal{G}_{\text{out}}$ once for each input set; subsequent novel views require no further filtering or reconstruction.

5 Conclusion

We presented a training-free filtering procedure that uses the per-view prediction structure of feed-forward 3DGS models for self-view and cross-view filtering. Self-view filtering identifies content inconsistent with renderings from the remaining Gaussians, while cross-view filtering measures how a Gaussian subset affects reconstruction of other input views. Gaussians selected by either filtering path are removed from the original prediction. The procedure requires no retraining or scene-specific optimization and uses no predefined semantic distractor categories. Experiments show improved average reconstruction quality across models and input counts, while largely preserving reconstruction quality on distractor-free inputs.

Our filtering relies on inconsistency across input views rather than semantic identity. Objects commonly treated as distractors, such as people, may therefore be retained when observed consistently across the input views. Our method also assumes an association between predicted Gaussians and input views, which the reconstruction models considered here natively provide; extending the procedure to architectures without this structure is an open direction. Future work could explore object-level reasoning for distractor filtering and geometry-guided generative completion of regions exposed by removal.

AI USE STATEMENT

We used a large language model to support manuscript writing and editing and to search for related work. Its use included improving the manuscript's grammar, clarity, and consistency, reformatting LaTeX content such as tables and figures, and assisting with the writing of code used in this research.

REPRODUCIBILITY STATEMENT

The main paper and appendix describe our method, experimental settings, and quantitative results. The datasets used in our experiments are openly available, and we plan to make our method's source code public upon acceptance of the paper.

REFERENCES

Yanqi Bao, Jing Liao, Jing Huo, and Yang Gao. Distractor-free generalizable 3d gaussian splatting. *arXiv preprint arXiv:2411.17605*, 2024.

Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 5470–5479, 2022.

David Charatan, Sizhe Lester 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*, pp. 19457–19467, 2024.

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*, pp. 370–386. Springer, 2024.

Vinayak Gupta, Chih-Hao Lin, Shenlong Wang, Anand Bhattad, and Jia-Bin Huang. Generalizable sparse-view 3d reconstruction from unconstrained images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 33217–33226, 2026.

Lihan Jiang, Yucheng Mao, Linning Xu, Tao Lu, Kerui Ren, Yichen Jin, Xudong Xu, Mulin Yu, Jiangmiao Pang, Feng Zhao, et al. Anysplat: Feed-forward 3d gaussian splatting from unconstrained views. *ACM Transactions on Graphics (TOG)*, 44(6):1–16, 2025.

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, George Drettakis, et al. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4):139–1, 2023.

Shakiba Kheradmand, Daniel Rebain, Gopal Sharma, Weiwei Sun, Yang-Che Tseng, Hossam Isack, Abhishek Kar, Andrea Tagliasacchi, and Kwang Moo Yi. 3d gaussian splatting as markov chain monte carlo. *Advances in Neural Information Processing Systems*, 37:80965–80986, 2024.

Jonas Kulhanek, Songyou Peng, Zuzana Kukelova, Marc Pollefeys, and Torsten Sattler. Wildgaussians: 3d gaussian splatting in the wild. *arXiv preprint arXiv:2407.08447*, 2024.

Chengqi Li, Zhihao Shi, Yangdi Lu, Wenbo He, and Xiangyu Xu. Robust neural rendering in the wild with asymmetric dual 3d gaussian splatting. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=jPaM3AiFLq.

Yihui Li, Chengxin Lv, Zichen Tang, Hongyu Yang, and Di Huang. Tokensplat: Token-aligned 3d gaussian splatting for feed-forward pose-free reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 40886–40895, 2026.

Jingyu Lin, Jiaqi Gu, Lubin Fan, Bojian Wu, Yujing Lou, Renjie Chen, Ligang Liu, and Jieping Ye. Hybridgs: Decoupling transients and statics with 2d and 3d gaussian splatting. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 788–797, 2025.

Yang Liu, Chuanchen Luo, Lue Fan, Naiyan Wang, Junran Peng, and Zhaoxiang Zhang. Citygaussian: Real-time high-quality large-scale scene rendering with gaussians. In *European Conference on Computer Vision*, pp. 265–282. Springer, 2024.

Tao Lu, Mulin Yu, Linning Xu, Yuanbo Xiangli, Limin Wang, Dahua Lin, and Bo Dai. Scaffold-gs: Structured 3d gaussians for view-adaptive rendering. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 20654–20664, 2024.

Ricardo Martin-Brualla, Noha Radwan, Mehdi SM 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*, pp. 7210–7219, 2021.

Hidenobu Matsuki, Riku Murai, Paul HJ Kelly, and Andrew J Davison. Gaussian splatting slam. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 18039–18048, 2024.

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. *Communications of the ACM*, 65(1):99–106, 2021.

Arthur Moreau, Richard Shaw, Michal Nazarczuk, Jisu Shin, Thomas Tanay, Zhensong Zhang, Songcen Xu, and Eduardo Pérez-Pellitero. Off the grid: Detection of primitives for feed-forward 3d gaussian splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11756–11766, 2026.

Tianbo Pan, Xingyi Yang, Shizun Wang, and Xinchao Wang. Visual geometry transformer in the wild: Distractor-free 3d reconstruction. *arXiv preprint arXiv:2606.22787*, 2026.

Wongi Park, Myeongseok Nam, Siwon Kim, Sangwoo Jo, and Soomok Lee. Forestsplats: Deformable transient field for gaussian splatting in the wild. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 6978–6987, 2026.

Minghan Qin, Wanhua Li, Jiawei Zhou, Haoqian Wang, and Hanspeter Pfister. Langsplat: 3d language gaussian splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 20051–20060, 2024.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PmLR, 2021.

Weining Ren, Zihan Zhu, Boyang Sun, Jiaqi Chen, Marc Pollefeys, and Songyou Peng. Nerf on-the-go: Exploiting uncertainty for distractor-free nerfs in the wild. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8931–8940, 2024.

Sara Sabour, Suhani Vora, Daniel Duckworth, Ivan Krasin, David J Fleet, and Andrea Tagliasacchi. Robustnerf: Ignoring distractors with robust losses. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 20626–20636, 2023.

Sara Sabour, Lily Goli, George Kopanas, Mark Matthews, Dmitry Lagun, Leonidas Guibas, Alec Jacobson, David Fleet, and Andrea Tagliasacchi. Spotlesssplats: Ignoring distractors in 3d gaussian splatting. *ACM Transactions on Graphics*, 44(2):1–11, 2025.

Johannes L Schonberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4104–4113, 2016.

Kangmin Seo, MinKyu Lee, Tae-Young Kim, ByeongCheol Lee, JoonSeoung An, and Jae-Pil Heo. Pdf-gs: Progressive distractor filtering for robust 3d gaussian splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Findings*, pp. 468–477, June 2026.

Oriane Siméoni, Huy V. Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seungeun Yi, Michaël Ramamonjisoa, Francisco Massa, Daniel Haziza, Luca Wehrstedt, Jianyuan Wang, Timothée Darcet, Théo Moutakanni, Leonel Sentana, Claire Roberts, Andrea Vedaldi, Jamie Tolan, John Brandt, Camille Couprie, Julien Mairal, Hervé Jégou, Patrick Labatut, and Piotr Bojanowski. DINOv3, 2025. URL https://arxiv.org/abs/2508.10104.

Weijie Wang, Yeqing Chen, Zeyu Zhang, Hengyu Liu, Haoxiao Wang, Zhiyuan Feng, Wenkang Qin, Feng Chen, Zheng Zhu, Donny Y Chen, et al. Volsplat: Rethinking feed-forward 3d gaussian splatting with voxel-aligned prediction. *arXiv preprint arXiv:2509.19297*, 2025a.

Xu Wang, Zhiru Wang, Shiyun Xie, Chengwei Pan, and Yisong Chen. Dualsplat: Robust 3d gaussian splatting via pseudo-mask bootstrapping from reconstruction failures. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4912–4921, 2026.

Yihao Wang, Marcus Klasson, Matias Turkulainen, Shuzhe Wang, Juho Kannala, and Arno Solin. Desplat: Decomposed gaussian splatting for distractor-free rendering. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 722–732, 2025b.

Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

Guanjun Wu, Taoran Yi, Jiemin Fang, Lingxi Xie, Xiaopeng Zhang, Wei Wei, Wenyu Liu, Qi Tian, and Xinggang Wang. 4d gaussian splatting for real-time dynamic scene rendering. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 20310–20320, 2024.

Tianyi Xie, Zeshun Zong, Yuxing Qiu, Xuan Li, Yutao Feng, Yin Yang, and Chenfanfu Jiang. Physgaussian: Physics-integrated 3d gaussians for generative dynamics. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4389–4398, 2024.

Haofei Xu, Daniel Barath, Andreas Geiger, and Marc Pollefeys. Resplat: Learning recurrent gaussian splats. *arXiv preprint arXiv:2510.08575*, 2025a.

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 Computer Vision and Pattern Recognition Conference*, pp. 16453–16463, 2025b.

Botao Ye, Boqi Chen, Haofei Xu, Daniel Barath, and Marc Pollefeys. Yonosplat: You only need one model for feedforward 3d gaussian splatting. *arXiv preprint arXiv:2511.07321*, 2025a.

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 *International Conference on Learning Representations*, volume 2025, pp. 54009–54033, 2025b.

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 conference on computer vision and pattern recognition*, pp. 586–595, 2018.

A Additional Experiments

A.1 Component Ablations with Different Numbers of Views

Table 6 compares the unfiltered prediction, self-view filtering alone, cross-view filtering alone, and their combination with 4, 8, and 16 input views. All variants share the same initial prediction, and filtering variants retain the same thresholds and validity conditions. We average each metric over scenes from RobustNeRF and NeRF On-the-go and the three main reconstruction models.

Table 6: Component ablations with 4, 8, and 16 input views. Full Method uses the union of the two removal masks.

	4 views			8 views			16 views		
Variant	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
Vanilla	19.56	0.641	0.301	19.88	0.638	0.307	19.53	0.611	0.331
Self-View Only	20.17	0.651	0.287	20.40	0.648	0.292	20.00	0.623	0.313
Cross-View Only	20.36	0.656	0.278	20.83	0.659	0.278	20.32	0.635	0.294
Full Method	20.51	0.658	0.277	20.98	0.661	0.275	20.50	0.639	0.288

A.2 Clean Scenes with Different Numbers of Views

Table 7 extends the clean scene comparison to 4, 8, and 16 input views for DepthSplat, YoNoSplat, and ReSplat. Each entry averages results on Android, Crab, Statue, and Yoda. Baseline and filtered predictions use identical inputs and evaluation cameras within each configuration.

Table 7: Reconstruction quality on the four clean RobustNeRF scenes with 4, 8, and 16 input views. Each baseline and its + Ours result share the same Gaussian prediction and ground-truth camera inputs. The four-view results for DepthSplat, YoNoSplat, and ReSplat match those in Table 4.

	4 views			8 views			16 views		
Method	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
DepthSplat	23.14	0.804	0.169	21.63	0.755	0.209	19.80	0.701	0.261
+ Ours	23.08	0.804	0.171	21.69	0.756	0.211	20.02	0.708	0.254
YoNoSplat	24.98	0.802	0.167	25.39	0.819	0.157	24.38	0.793	0.182
+ Ours	24.97	0.801	0.169	25.37	0.818	0.158	24.36	0.792	0.182
ReSplat	24.88	0.809	0.224	24.56	0.798	0.230	26.32	0.835	0.194
+ Ours	24.85	0.809	0.225	24.49	0.797	0.232	26.30	0.834	0.195

B Per-Scene Reconstruction Quality

Tables 8 and 9 report the scene-level results on NeRF On-the-go and RobustNeRF underlying Table 1 for all three reconstruction models and input counts.

Table 8: Per-scene results on NeRF On-the-go for all backbones and input counts.

			DepthSplat			YoNoSplat			ReSplat		
Scene	Views	Variant	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
Corner	4	Vanilla	19.06	0.713	0.231	21.35	0.747	0.222	19.97	0.759	0.241
		+ Ours	19.87	0.742	0.185	23.20	0.798	0.149	20.77	0.773	0.226
	8	Vanilla	20.59	0.738	0.225	21.75	0.769	0.219	23.20	0.837	0.182
		+ Ours	22.61	0.785	0.154	23.96	0.817	0.159	25.10	0.855	0.162
	16	Vanilla	18.72	0.641	0.323	20.54	0.733	0.259	23.89	0.851	0.158
		+ Ours	20.43	0.705	0.225	22.29	0.780	0.201	24.16	0.854	0.154
Fountain	4	Vanilla	19.63	0.582	0.262	20.98	0.641	0.235	21.26	0.643	0.325
		+ Ours	19.67	0.583	0.260	21.09	0.641	0.235	21.23	0.641	0.327
	8	Vanilla	18.27	0.494	0.327	19.50	0.563	0.304	19.84	0.568	0.368
		+ Ours	18.30	0.495	0.325	19.99	0.569	0.295	19.95	0.569	0.368
	16	Vanilla	15.89	0.392	0.397	17.56	0.469	0.354	17.86	0.510	0.388
		+ Ours	16.05	0.403	0.383	17.71	0.477	0.345	17.97	0.511	0.390
Mountain	4	Vanilla	18.44	0.517	0.318	19.12	0.559	0.284	19.30	0.606	0.336
		+ Ours	18.57	0.520	0.318	19.05	0.554	0.293	19.35	0.606	0.338
	8	Vanilla	17.18	0.444	0.370	18.95	0.513	0.300	18.65	0.539	0.377
		+ Ours	17.40	0.450	0.364	18.89	0.511	0.302	18.72	0.540	0.378
	16	Vanilla	15.91	0.377	0.423	18.50	0.498	0.306	18.48	0.524	0.367
		+ Ours	16.33	0.396	0.399	18.53	0.498	0.305	18.53	0.525	0.367
Patio	4	Vanilla	14.52	0.505	0.349	13.26	0.337	0.453	15.89	0.586	0.344
		+ Ours	16.74	0.567	0.278	16.01	0.397	0.370	17.12	0.626	0.304
	8	Vanilla	14.25	0.515	0.365	13.54	0.415	0.418	14.63	0.593	0.343
		+ Ours	17.13	0.596	0.269	16.96	0.476	0.326	17.23	0.649	0.294
	16	Vanilla	14.03	0.432	0.456	14.38	0.422	0.419	16.01	0.605	0.329
		+ Ours	17.06	0.539	0.331	16.17	0.465	0.361	17.18	0.643	0.296
Patio High	4	Vanilla	17.75	0.538	0.261	17.05	0.548	0.316	18.60	0.648	0.297
		+ Ours	18.15	0.548	0.244	18.37	0.588	0.258	18.80	0.655	0.290
	8	Vanilla	17.62	0.520	0.309	18.46	0.586	0.269	19.37	0.662	0.294
		+ Ours	18.76	0.549	0.259	19.74	0.618	0.229	20.16	0.679	0.277
	16	Vanilla	16.20	0.424	0.410	17.87	0.535	0.316	20.25	0.701	0.271
		+ Ours	18.22	0.496	0.302	19.02	0.573	0.255	21.05	0.722	0.247
Spot	4	Vanilla	17.77	0.438	0.412	18.33	0.447	0.468	18.21	0.468	0.502
		+ Ours	18.12	0.448	0.388	18.94	0.457	0.470	18.68	0.471	0.495
	8	Vanilla	18.67	0.431	0.422	19.75	0.521	0.445	19.10	0.464	0.481
		+ Ours	19.77	0.452	0.353	20.74	0.540	0.414	19.52	0.470	0.467
	16	Vanilla	17.19	0.417	0.487	18.83	0.536	0.447	20.23	0.560	0.403
		+ Ours	19.63	0.478	0.351	20.78	0.575	0.365	21.41	0.574	0.380
Mean	4	Vanilla	17.86	0.549	0.306	18.35	0.547	0.330	18.87	0.618	0.341
		+ Ours	18.52	0.568	0.279	19.44	0.573	0.296	19.32	0.629	0.330
	8	Vanilla	17.76	0.524	0.336	18.66	0.561	0.326	19.13	0.610	0.341
		+ Ours	19.00	0.554	0.287	20.05	0.588	0.287	20.11	0.627	0.324
	16	Vanilla	16.32	0.447	0.416	17.95	0.532	0.350	19.45	0.625	0.319
		+ Ours	17.95	0.503	0.332	19.08	0.562	0.305	20.05	0.638	0.306

Table 9: Per-scene results on RobustNeRF for all baselines and input counts.

			DepthSplat			YoNoSplat			ReSplat		
Scene	Views	Variant	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
Android	4	Vanilla	18.42	0.650	0.343	18.14	0.501	0.397	19.27	0.674	0.352
		+ Ours	19.52	0.686	0.286	18.61	0.500	0.382	19.47	0.677	0.349
	8	Vanilla	18.18	0.588	0.363	19.13	0.597	0.356	18.13	0.612	0.391
		+ Ours	19.21	0.629	0.300	20.10	0.632	0.306	18.32	0.619	0.383
	16	Vanilla	17.26	0.517	0.429	18.99	0.568	0.358	21.20	0.682	0.302
		+ Ours	19.69	0.610	0.292	19.75	0.590	0.318	21.37	0.685	0.296
Crab	4	Vanilla	21.47	0.794	0.227	25.28	0.866	0.182	25.98	0.867	0.228
		+ Ours	23.70	0.819	0.189	25.91	0.875	0.164	26.17	0.868	0.227
	8	Vanilla	20.48	0.757	0.259	26.78	0.898	0.146	26.38	0.863	0.226
		+ Ours	22.76	0.787	0.213	27.00	0.900	0.140	26.43	0.863	0.225
	16	Vanilla	19.25	0.705	0.344	25.76	0.861	0.182	28.73	0.901	0.179
		+ Ours	20.59	0.737	0.287	25.82	0.862	0.178	28.78	0.902	0.178
Statue	4	Vanilla	19.01	0.717	0.251	19.83	0.688	0.240	19.79	0.745	0.258
		+ Ours	20.59	0.742	0.213	21.34	0.706	0.215	21.39	0.757	0.243
	8	Vanilla	18.98	0.694	0.278	20.56	0.727	0.230	21.06	0.746	0.262
		+ Ours	19.74	0.718	0.239	21.73	0.746	0.190	21.91	0.756	0.246
	16	Vanilla	17.06	0.638	0.363	19.76	0.633	0.261	21.36	0.745	0.260
		+ Ours	18.48	0.685	0.279	20.09	0.640	0.245	21.49	0.747	0.254
Yoda	4	Vanilla	21.49	0.791	0.245	23.57	0.834	0.223	23.96	0.825	0.242
		+ Ours	24.21	0.824	0.188	26.07	0.860	0.178	24.70	0.829	0.236
	8	Vanilla	22.00	0.786	0.253	26.01	0.884	0.189	25.33	0.831	0.243
		+ Ours	24.37	0.819	0.198	27.33	0.896	0.162	25.76	0.833	0.239
	16	Vanilla	20.18	0.714	0.338	26.33	0.868	0.196	27.55	0.861	0.213
		+ Ours	21.82	0.757	0.271	26.95	0.874	0.182	27.67	0.862	0.211
Mean	4	Vanilla	20.10	0.738	0.267	21.70	0.722	0.260	22.25	0.778	0.270
		+ Ours	22.00	0.768	0.219	22.98	0.735	0.235	22.93	0.783	0.264
	8	Vanilla	19.91	0.706	0.288	23.12	0.776	0.230	22.73	0.763	0.280
		+ Ours	21.52	0.738	0.238	24.04	0.794	0.199	23.11	0.768	0.273
	16	Vanilla	18.44	0.644	0.369	22.71	0.733	0.249	24.71	0.797	0.238
		+ Ours	20.15	0.697	0.282	23.15	0.741	0.231	24.83	0.799	0.235