FREESPLATTER: POSE-FREE GAUSSIAN SPLATTING FOR SPARSE-VIEW 3D RECONSTRUCTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Existing sparse-view reconstruction models heavily rely on accurate known camera poses. However, deriving camera extrinsics and intrinsics from sparse-view images presents significant challenges. In this work, we present **FreeSplatter**, a highly scalable, feed-forward reconstruction framework capable of generating high-quality 3D Gaussians from **uncalibrated** sparse-view images and recovering their camera parameters in mere seconds. FreeSplatter is built upon a streamlined transformer architecture, comprising sequential self-attention blocks that facilitate information exchange among multi-view image tokens and decode them into pixel-wise 3D Gaussian primitives. The predicted Gaussian primitives are situated in a unified reference frame, allowing for high-fidelity 3D modeling and instant camera parameter estimation using off-the-shelf solvers. To cater to both **object-centric** and **scene-level** reconstruction, we train two model variants of FreeSplatter on extensive datasets. In both scenarios, FreeSplatter outperforms state-of-the-art baselines in terms of reconstruction quality and pose estimation accuracy. Furthermore, we showcase FreeSplatter's potential in enhancing the productivity of downstream applications, such as text/image-to-3D content creation.

1 INTRODUCTION

Recent breakthroughs in neural scene representation and differentiable rendering, *e.g.*, Neural Radiance Fields (NeRF) (Mildenhall et al., 2021) and Gaussian Splatting (GS) (Kerbl et al., 2023), have shown unprecedented multi-view reconstruction quality for densely-captured images with calibrated camera poses, employing a per-scene optimization approach. However, they are not applicable to sparse-view scenarios, where classical camera calibration techniques like Structure-from-Motion (SfM) (Schonberger & Frahm, 2016) tend to fail due to insufficient image overlaps. Generalizable reconstruction models (Hong et al., 2024b; Xu et al., 2024a; Charatan et al., 2024) attempt to address sparse-view reconstruction in a feed-forward manner by utilizing learned data priors. Despite their efficiency and generalization capabilities, these models predominantly assume access to accurate camera poses and intrinsics, or imply that they are obtained in a pre-processing step, thereby circumventing the difficulty of deriving camera parameters in real application scenarios. Liberating sparse-view reconstruction from known camera poses remains a significant challenge.

Prior works have explored training *pose-free* reconstruction models. PF-LRM (Wang et al., 2024a) and LEAP (Jiang et al., 2024b) share a similar framework that maps multi-view image tokens to a NeRF representation with a transformer. Despite the milestone they have set, their NeRF representation suffers from inefficient volume rendering and low resolution, limiting the training efficiency and scalability to complex scenes. Besides, inferring the input camera poses from such implicit representations is not trivial. PF-LRM employs an additional branch to predict per-view coarse point clouds for camera pose prediction, which introduces extra training cost and difficulties. DUSt3R (Wang et al., 2024b) presents a novel paradigm for joint 3D reconstruction and pose estimation by modeling SfM and Multi-view Stereo (MVS) as an end-to-end point-regression task. By regressing the "point maps", *i.e.*, the 3D un-projection of depth maps, from a stereo image pair in a unified reference frame, it can recover the relative camera pose efficiently with a PnP (Fischler & Bolles, 1981; Hartley & Zisserman, 2003) solver. DUSt3R demonstrates impressive zero-shot 3D reconstruction and pose prediction capability by training on a broad range of datasets.

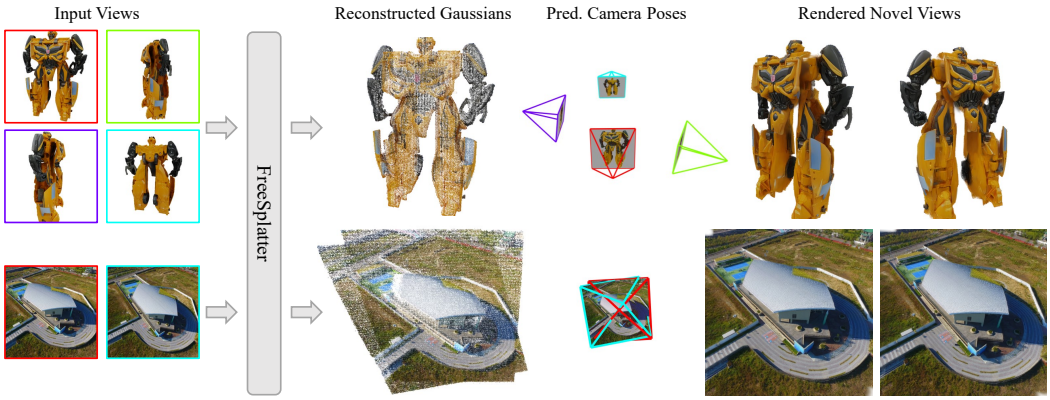

Figure 1: Given **uncalibrated** sparse-view images, our FreeSplatter can reconstruct pixel-wise 3D Gaussians, enabling both high-fidelity novel view rendering and instant camera pose estimation in mere seconds. FreeSplatter can deal with both object-centric (up) and scene-level (down) scenarios.

However, the sparsity of point cloud makes it a suboptimal representation for many downstream applications, *e.g.*, novel view synthesis. In comparison, 3D Gaussian Splats (3DGS) can encode high-fidelity radiance fields and enable efficient view synthesis by augmenting point clouds with additional Gaussian attributes. We then ask, can we directly predict the "Gaussian maps" from multi-view images to achieve both high-quality 3D modeling and instant camera pose estimation? In this work, we presents **FreeSplatter**, a feed-forward reconstruction framework that predicts per-pixel Gaussians from **uncalibrated** sparse-view images and estimates their camera parameters jointly, to say "yes" to this question. The core of FreeSplatter is a scalable single-stream transformer architecture that maps multi-view image tokens into pixel-aligned Gaussian maps with simple self-attention layers, requiring no input camera poses, intrinsics, nor post-alignment steps. The predicted Gaussian maps serve as a high-fidelity and efficient 3D scene representation, while the Gaussian locations enable ultra-fast camera extrinsics and intrinsics estimation in mere seconds using off-the-shelf solvers.

The training and rendering efficiency of 3D Gaussians makes it possible to consider more complex scene-level reconstruction. We thus train two FreeSplatter variants on Objaverse (Deitke et al., 2023) and a mixture of multiple scene datasets (Yao et al., 2020; Yeshwanth et al., 2023; Reizenstein et al., 2021) to handle object-centric and scene-level reconstruction scenarios, respectively. The two variants, denoted as **FreeSplatter-O** (object) and **FreeSplatter-S** (scene), share a common model architecture with $\sim$306 million parameters while making essential adjustments for task adaption. We conduct extensive experiments to demonstrate the superiority of FreeSplatter compared to prior sparse-view reconstruction methods in terms of reconstruction quality and pose estimation accuracy. To be highlighted, FreeSplatter-O outperforms previous *pose-dependent* large reconstruction models by a large margin, while FreeSplatter-S achieves a pose-estimation accuracy on par with MASt3R (Leroy et al., 2024), the upgraded version of DUSt3R, on ScanNet++ and CO3D benchmarks. Furthermore, we showcase the potential of FreeSplatter in enhancing the productivity of downstream applications, such as text/image-to-3D content creation.

2 Related Work

Large Reconstruction Models. The availability of large-scale 3D datasets (Deitke et al., 2023; 2024) unlocks the possibility of training highly generalizable models for open-category image-to-3D reconstruction. Large Reconstruction Models (LRMs) (Hong et al., 2024b; Xu et al., 2024c; Li et al., 2024a) leverage a scalable feed-forward transformer architecture to map sparse-view image tokens into a 3D triplane NeRF (Mildenhall et al., 2021; Chan et al., 2022) and supervise it with multi-view rendering loss. Subsequent works have made efforts on adopting mesh (Xu et al., 2024a; Wei et al., 2024; Wang et al., 2024c) or 3D Gaussians representation (Tang et al., 2024; Xu et al., 2024b; Zhang et al., 2024b) for real-time rendering speed, exploring more efficient network architectures (Zheng et al., 2024; Zhang et al., 2024a; Chen et al., 2024a; Li et al., 2024b; Cao et al., 2024), improving the texture quality (Boss et al., 2024; Siddiqui et al., 2024; Yang et al., 2024a), and exploiting explicit 3D supervision for better geometry (Liu et al., 2024a). Despite the superior reconstruction quality

and generalization capability, LRMs require *posed* images as input and are highly sensitive to the pose accuracy, limiting their applicable scenarios.

Pose-free Reconstruction. Classical pose-free reconstruction algorithms like Structure from Motion (SfM) (Hartley & Zisserman, 2003; Ullman, 1979; Schonberger & Frahm, 2016) begins with finding pixel correspondences across multiple views first, and then perform 3D points triangulation and bundle adjustment to optimize 3D coordinates and camera parameters jointly. Recent efforts on enhancing the SfM pipeline encompass leveraging learning-based feature descriptors (DeTone et al., 2018; Dusmanu et al., 2019; Revaud et al., 2019; Wang et al., 2023c) and image matchers (Edstedt et al., 2023; 2024; Sarlin et al., 2020; Lindenberger et al., 2023), as well as differentiable bundle adjustment (Lin et al., 2021; Wang et al., 2023a; Wei et al., 2020). SfM-based methods works well with sufficient image overlaps between nearby views, but face challenges in matching correspondences from sparse views. On the other hand, learning-based methods (Jiang et al., 2024a;b; Hong et al., 2024a; Fan et al., 2023) rely on learned data priors to recover the 3D geometry from input views. PF-LRM (Wang et al., 2024a) extends the LRM framework by predicting a per-view coarse point cloud for camera pose estimation with a differentiable PnP solver (Chen et al., 2022). DUSt3R (Wang et al., 2024b) presents a novel framework for Multi-view stereo (MVS) by modeling it as a pointmap-regression problem. Subsequent works further improve it in local representation capability (Leroy et al., 2024) and reconstruction efficiency (Wang & Agapito, 2024).

Generalizable Gaussian Splatting. Compared to NeRF's (Mildenhall et al., 2021) MLP-based implicit representation, 3DGS represents a scene explicitly with a point cloud, which strikes the balance between high-fidelity and real-time rendering speed. However, the per-scene optimization of 3DGS requires densely-captured images and sparse point cloud generated by SfM for initialization. Recent works (Charatan et al., 2024; Chen et al., 2024b; Szymanowicz et al., 2024; Liu et al., 2024b; Wewer et al., 2024) have explored feed-forward models for sparse-view Gaussian reconstruction by capitalizing on large-scale datasets and scalable model architectures. These models typically assume the access to accurate camera poses and utilize 3D-to-2D geometric projection for feature aggregation, adopting model designs like epipolar lines (Charatan et al., 2024) or plane-swept cost-volume (Chen et al., 2024b; Liu et al., 2024b). InstantSplat (Fan et al., 2024) and Splatt3R (Smart et al., 2024) leverage the pose-free reconstruction capability of DUSt3R/MASt3R for sparse-view reconstruction. The former initializes the Gaussian positions using the DUSt3R point cloud and then optimizes other Gaussian paremters, while the latter trains a Gaussian head in a frozen MASt3R model. Despite the impressive results, their reconstruction quality highly depends on the point cloud quality generated by DUSt3R.

3 Method

Given a sparse set of N input images $\{\boldsymbol{I}^n \mid n = 1, \dots, N\}$ with no known camera extrinsics nor intrinsics, FreeSplatter aims to reconstruct the scene as a set of Gaussians and estimate the camera parameters of the N images simultaneously. The pipeline can be formulated as:

$$\boldsymbol{G}, \boldsymbol{P}^1, \dots, \boldsymbol{P}^N, f = \text{FreeSplatter}\left(\boldsymbol{I}^1, \dots, \boldsymbol{I}^N\right), \tag{1}$$

where $\boldsymbol{G}$ denotes the reconstructed Gaussians which is the union of N pixel-wise Gaussian maps, *i.e.*, $\boldsymbol{G} = \{\boldsymbol{G}^n \mid n = 1, \dots, N\}$, $\boldsymbol{P}^n$ denotes the estimated camera pose for $\boldsymbol{I}^n$, and f denotes the focal length. We assume a common focal length for all input images which is reasonable in most scenarios.

We implement FreeSplatter as a feed-forward transformer that takes multi-view image as input and predict N Gaussian maps $\{\boldsymbol{G}^n \mid n = 1, \dots, N\}$ in a unified reference frame. The Gaussian maps enable both high-fidelity scene modeling and ultra-fast camera parameter estimation using off-the-shelf solvers (Fischler & Bolles, 1981; Hartley & Zisserman, 2003; Plastria, 2011), since the Gaussian centers have located the surface point cloud explicitly. In this section, we first briefly review the background knowledge of Gaussian Splatting (Section 3.1), and then introduce the model architecture (Section 3.2) and training details (Section 3.3) of our approach.

3.1 Preliminary

3D Gaussian Splatting (3DGS) (Kerbl et al., 2023) represents a scene as a set of 3D Gaussian primitives $\{\boldsymbol{g}_k = (\boldsymbol{\mu}_k, \boldsymbol{r}_k, \boldsymbol{s}_k, o_k, \boldsymbol{c}_k) \mid k = 1, \dots, K\}$, each primitive is parameterized by location

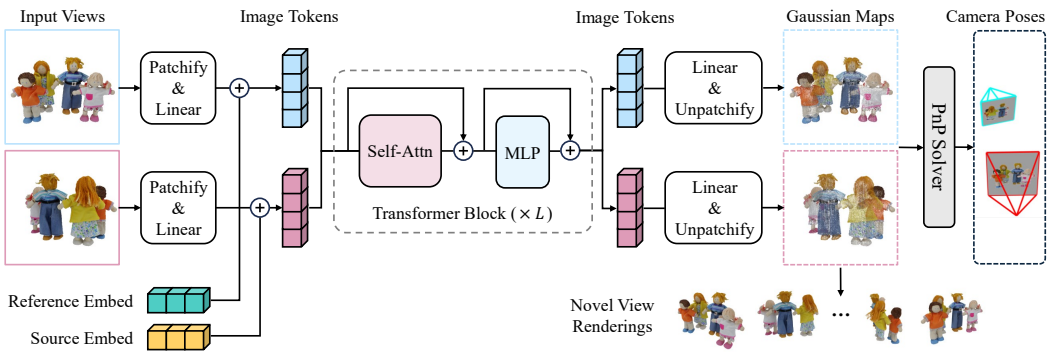

Figure 2: **FreeSplatter pipeline.** Given input views $\{\boldsymbol{I}^n \mid n = 1, \dots, N\}$ without any known camera extrinsics or intrinsics, we first patchify them into image tokens, and then feed all tokens into a sequence of self-attention blocks to exchange information among multiple views. Finally, we decode the output image tokens into N Gaussian maps $\{\boldsymbol{G}^n \mid n = 1, \dots, N\}$, from which we can render novel views, as well as recovering camera focal length f and poses $\{\boldsymbol{P}^n \mid n = 1, \dots, N\}$ with simple iterative solvers.

$\boldsymbol{\mu}_k \in \mathbb{R}^3$, rotation quaternion $\boldsymbol{r}_k \in \mathbb{R}^4$, scale $\boldsymbol{s}_k \in \mathbb{R}^3$, opacity $o_k \in \mathbb{R}$, and Spherical Harmonic (SH) coefficients $\boldsymbol{c}_k \in \mathbb{R}^{3\times d}$ for computing view–dependent color, with d denoting the degree of SH. These primitives parameterize the scene's radiance field with an explicit point cloud and can be rendered into novel views with an efficient rasterizer. Compared to the expensive volume rendering process of NeRF, 3DGS achieves comparable rendering quality while being much more efficient in terms of time and memory.

3.2 Model Architecture

As Figure 2 shows, FreeSplatter employs a transformer architecture similar to GS-LRM (Zhang et al., 2024b). Given a set of uncalibrated images $\{\boldsymbol{I}^n \mid n = 1, \dots, N\}$, we first tokenize them into image tokens $\{\boldsymbol{e}^{n,m} \mid n = 1, \dots, N, m = 1, \dots, M\}$, where M denotes the patch number of each image. Then we feed all image tokens into a sequence of self-attention blocks to exchange information among multiple views, and finally decode them into N Gaussian maps $\{\boldsymbol{G}^n \mid n = 1, \dots, N\}$ composed of per-pixel Gaussian primitives, from which we can render novel views and recover camera parameters with simple iterative solvers.

Image Tokenization. The only input of our model is N images $\left\{\boldsymbol{I}^n \in \mathbb{R}^{H\times W\times 3} \mid n = 1, \dots, N\right\}$. Following ViT (Dosovitskiy et al., 2021), we split the images into $p \times p$ patches ($p = 8$) and flatten them into 1D vectors of dimension $p^2 \cdot 3$. Then a linear layer is applied to map the vectors into d-dim image tokens (d equals to the transformer width). Denoting the m-th token of the n-th image as $\boldsymbol{e}^{n,m}$, we add it with two additional embeddings before fedding it into the transformer:

$$\boldsymbol{e}^{n,m} = \boldsymbol{e}^{n,m} + \boldsymbol{e}^{m}_{\text{pos}} + \boldsymbol{e}^{n}_{\text{view}}, \tag{2}$$

where $\boldsymbol{e}^{m}_{\text{pos}}$ denotes the position embedding for the m-th patch, and $\boldsymbol{e}^{n}_{\text{view}}$ is a view embedding for distinguishing the reference view and other views. Specifically, **we take the first image as the reference view and predict all Gaussian in its camera frame**. We use a reference embedding $\boldsymbol{e}^{\text{ref}}$ for the first view ($n = 1$) and another source embedding $\boldsymbol{e}^{\text{src}}$ for other views ($n = 2, \dots, N$), both of which are learnable during training.

Feed-forward Transformer. The image tokens added with position and view embeddings are then fed into a sequence of L self-attention blocks to exchange information across multiple views. Each block consists of a self-attention layer and an MLP layer, equipping with pre-normalization and residual connection (He et al., 2016). The forward process of each block can be formulated as:

$$\begin{aligned} \boldsymbol{e}^{n,m}_{\text{attn}} &= \text{SelfAttn}\left(\text{LayerNorm}\left(\boldsymbol{e}^{n,m}_{\text{in}}\right), \{\text{LayerNorm}\left(\boldsymbol{e}^{n,m}_{\text{in}}\right)\}\right) + \boldsymbol{e}^{n,m}_{\text{in}}, \\ \boldsymbol{e}^{n,m}_{\text{out}} &= \text{MLP}\left(\text{LayerNorm}\left(\boldsymbol{e}^{n,m}_{\text{attn}}\right)\right) + \boldsymbol{e}^{n,m}_{\text{attn}}. \end{aligned} \tag{3}$$

Gaussian Map Prediction. For each output token $\boldsymbol{e}^{n,m}_{\text{out}}$ of the last (L-th) block, we decode it back into p^2 Gaussians with a simple linear layer, leading to a 1D vector of dimension $p^2 \cdot q$, where q

Figure 3: **Object-centric Sparse-view Reconstruction.** We show 6 samples from the Google Scanned Objects dataset. To be noted, the results of LGM and InstantMesh (2nd and 3rd rows) are generated with ground truth camera poses (and intrinsics), while our results (4th row) are generated in a completely pose-free manner.

is the parameter number of each Gaussian primitive. Then we unpatchify the predicted Gaussian parameters into a Gaussian patch $\boldsymbol{G}^{n,m} \in \mathbb{R}^{p \times p \times q}$. Finally, we spatially concatenate the Gaussian patches $\boldsymbol{G}^{n,m}$ for each view, leading to N Gaussian maps $\left\{\boldsymbol{G}^{n} \in \mathbb{R}^{H \times W \times q} \mid n = 1, \ldots, N\right\}$.

Each pixel in the Gaussian maps contains a q-dim 3D Gaussian primitive. Prior pose-dependent Gaussian LRMs (Zhang et al., 2024b; Xu et al., 2024b; Tang et al., 2024) parameterize each Gaussian's location with a single depth value. However, our model assumes no available camera poses nor instrinsics, making it impossible to un-project the Gaussian locations from depths. Therefore, we directly predict the actual Gaussian locations in the reference frame and adopt a pixel-alignment loss to restrict Gaussians to lie on camera rays (see Section 3.3 for details).

Camera Pose Estimation. To recover the camera poses, we first estimate the focal length f from predicted Gaussian maps (see Section A.1 for details). Different from DUSt3R that can only feed a pair of images into the model to predict pair-wise point maps and requires an additional alignment to unify the pair-wise reconstruction results into a global point cloud, FreeSplatter predicts the Gaussian maps of all views in a unified reference frame, thus can recover the camera poses $\{\boldsymbol{P}^n \mid n = 1, \ldots, N\}$ in a feed-forward manner without global alignment. For each Gaussian location map $\boldsymbol{X}^n \in \mathbb{R}^{H \times W \times 3}$ (the first 3 channels of $\boldsymbol{G}^n$), its pixel coordinate map $\boldsymbol{Y}^n \in \mathbb{R}^{H \times W \times 2}$ and a valid mask $\boldsymbol{M}^n \in \mathbb{R}^{H \times W}$, we apply a PnP-RANSAC solver (Hartley & Zisserman, 2003; Bradski, 2000) to estimate $\boldsymbol{P}^n \in \mathbb{R}^{4 \times 4}$:

$$\boldsymbol{P}^n = \mathrm{PnP}\left(\boldsymbol{X}^n, \boldsymbol{Y}^n, \boldsymbol{M}^n, \boldsymbol{K}\right), \tag{4}$$

where $\boldsymbol{K} = [[f, 0, \frac{W}{2}], [0, f, \frac{H}{2}], [0, 0, 1]]$ denotes the estimated intrinsic matrix. The mask $\boldsymbol{M}^n$ aims to mark the pixel locations that contribute to the pose optimization, which differs in object-centric and scene-level reconstruction scenarios, please see Section A.1 for more details.

3.3 Training Details

We train two FreeSplatter variants targeting for object-centric and scene-level pose-free reconstruction. They share the same model architecture and parameter size, but we made essential adjustments in training objectives and strategies to adapt them to different scenarios.

Staged Training Strategy. Prior pose-dependent LRMs (Hong et al., 2024b; Li et al., 2024a; Xu et al., 2024b; Zhang et al., 2024b) leverage pure rendering loss for supervision. However, our model assumes no known camera poses nor intrinsics and the Gaussian positions are free in 3D space, making it extremely challenging to predict correct Gaussian positions. Gaussian-based reconstruction approaches heavily rely on the initialization of Gaussian positions, *e.g.*, 3DGS (Kerbl et al., 2023) initializes the Gaussian positions with the sparse point cloud generated by SfM, while the parameters of our model are randomly initialized at the beginning. In practice, we found it essential to supervise the Gaussian positions at the beginning:

$$\mathcal{L}_{\text{pos}} = \sum_{n=1}^{N} \left\| \boldsymbol{M}^n \odot \hat{\boldsymbol{X}}^n - \boldsymbol{M}^n \odot \boldsymbol{X}^n \right\|, \tag{5}$$

where $\hat{\boldsymbol{X}}^n \in \mathbb{R}^{H \times W \times 3}$ is the predicted Gaussian position map, $\boldsymbol{X}^n$ is the ground truth point map unprojected from depths. $\boldsymbol{M}^n \in \mathbb{R}^{H \times W}$ is a mask indicating the pixels storing valid depth values, which is exactly the foreground object mask for object-centric reconstruction. For scene-level reconstruction, $\boldsymbol{M}^n$ depends on where the depth values are defined in different datasets.

We apply $\mathcal{L}_{\text{pos}}$ in the pre-training stage, so that the model learns to predict approximately correct Gaussian positions. In our experiments, this pre-training is **essential** to model's convergence. However, $\mathcal{L}_{\text{pos}}$ can only supervise the pixels with valid depths, while the Gaussian positions predicted at other pixels remain unconstrained. Besides, the ground truth depths are noisy in some datasets, and applying $\mathcal{L}_{\text{pos}}$ throughout the training leads to degraded rendering quality. To provide a more stable geometric supervision, we adopt a pixel-alignment loss to enforce each predicted Gaussian to be aligned with its corresponding pixel, which maximizes the ray cosine similarities:

$$\mathcal{L}_{\text{align}} = \sum_{n=1}^{N} \sum_{i=0}^{H} \sum_{j=0}^{W} \left(1 - \frac{\hat{\boldsymbol{r}}_{i,j}^n \cdot \boldsymbol{r}_{i,j}^n}{\|\hat{\boldsymbol{r}}_{i,j}^n\| \|\boldsymbol{r}_{i,j}^n\|} \right), \tag{6}$$

where $\boldsymbol{r}_{i,j}^n$ denotes the ray from the camera origin $\boldsymbol{t}^n$ to point $\boldsymbol{X}_{i,j}^n$. $\mathcal{L}_{\text{align}}$ restricts the predicted Gaussians to be distributed on the camera rays, which improves the rendering quality. Besides, enforcing the pixel-alignment is important to camera parameter estimation, since the iterative solvers minimize the pixel-projection errors.

Loss Functions. The overall training objective of our model is:

$$\mathcal{L} = \mathcal{L}_{\text{render}} + \lambda_{\text{a}} \cdot \mathcal{L}_{\text{align}} + \mathbf{1}_{\text{t} \leq \text{T}_{\text{max}}} \lambda_{\text{p}} \cdot \mathcal{L}_{\text{pos}}, \tag{7}$$

where the rendering loss $\mathcal{L}_{\text{render}}$ is a combination of MSE and LPIPS loss. t and T_{max} denote the current training step and maximum pre-training step, respectively. In our implementation, we set $\lambda_{\text{a}} = 1.0, \lambda_{\text{p}} = 10.0$.

Occlusion in Pixel-aligned Gaussians. Prior pose-dependent Gaussian-basd LRMs (Tang et al., 2024; Zhang et al., 2024b; Xu et al., 2024b) parameterize the Gaussian positions with a single depth value to ensure they are pixel-aligned. Despite the simplicity, pixel-aligned Gaussians can only represent the areas captured by input images. It is challenging for sparse input views to cover the whole scene, resulting in missing reconstruction results in the occluded areas. Our model also suffers from this problem due to the usage of $\mathcal{L}_{\text{align}}$. For the object-centric model, we adopt a simple strategy to alleviate this problem by only calculate Equation 6 in the foreground area, while allowing other Gaussians to move freely and model the occluded areas.

For the scene-level model trained on real-world images, all predicted Gaussians have be pixel-aligned to model the complex background. We instead focus on reconstructing the observed areas. To avoid the areas in the target views that are invisible in input views provide negative guidance for training, we adopt the target-view masking strategy proposed by Splatt3R and only calculate the rendering loss in visible areas. Please refer to Smart et al. (2024) for more details.

4 Experiments

4.1 Experimental Settings

Training Datasets. FreeSplatter-O assumes the input images are white-background with an object in the center, which is trained on Objaverse (Deitke et al., 2023), a large-scale dataset containing around 800K 3D assets. For each 3D asset, we normalize it into a $[-1, 1]^3$ cube, and then

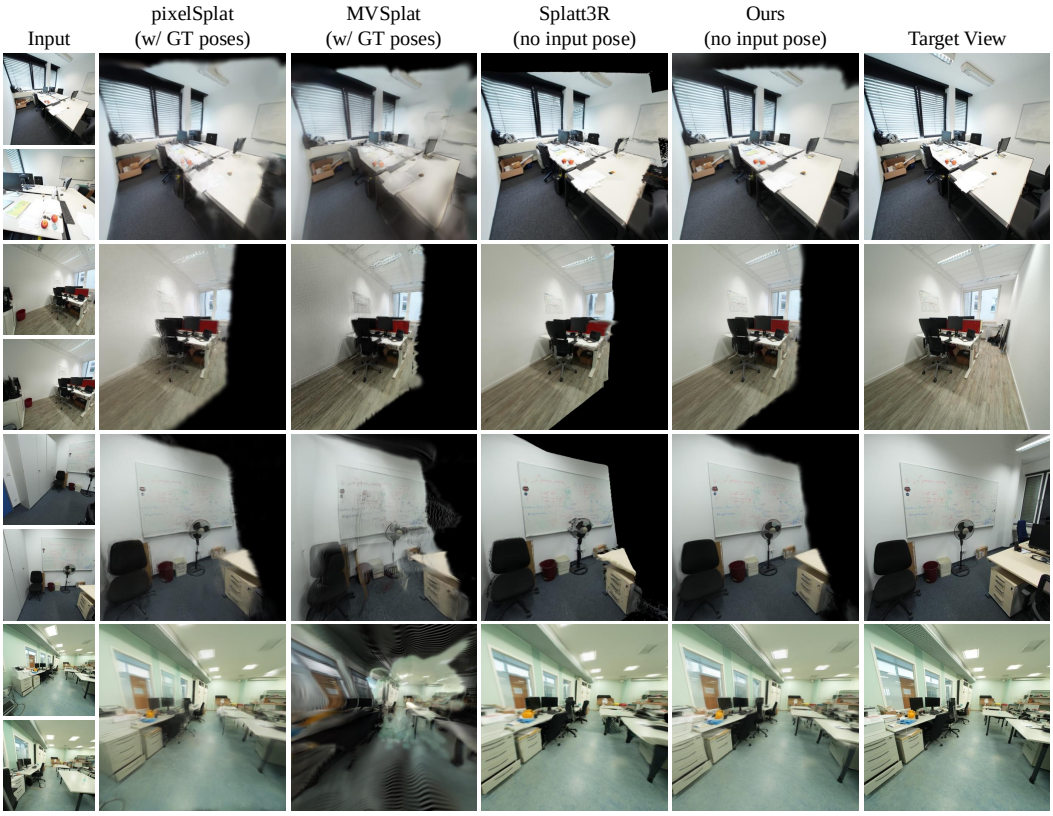

Figure 4: **Scene-level Reconstruction on ScanNet++.** The results of pixelSplat and MVSplat are obtained with ground truth input poses, while the results of Splat3R and ours are pose-free.

render 32 random views associated with depth maps around it, all in a resolution of 512×512. FreeSplatter-S is trained on a mixture of multiple datasets, including BlendedMVS (Yao et al., 2020), ScanNet++ (Yeshwanth et al., 2023) and CO3Dv2 (Reizenstein et al., 2021), which is a subset of DUSt3R's (Wang et al., 2024b) training datasets covering the most representative scene types: outdoor, indoor, and realistic objects.

Evaluation Datasets. For object-centric reconstruction and pose-estimation, we evaluate our model on OmniObject3D (Wu et al., 2023) and Google Scanned Objects (GSO) (Downs et al., 2022) datasets, containing around 6K and 1K scanned 3D assets, respectively. We use the whole GSO dataset and pick out 300 OmniObject3D objects in 30 categories as the evaluation set. For each object, we render 20 random views and 4 additional structured input views for the sparse-view reconstruction task, which are evenly placed around the objects with an elevation of 20° to ensure a good coverage. For scene-level reconstruction and pose estimation, we mainly evaluate on the test splits of ScanNet++ (Yeshwanth et al., 2023) and CO3Dv2 (Reizenstein et al., 2021).

4.2 Sparse-view Reconstruction

Baselines. Prior pose-free object reconstruction models like LEAP (Jiang et al., 2024b) were only trained on small datasets, showing very limited generalization capability in open-domain datasets. PF-LRM (Wang et al., 2024a) is the most comparable work to our FreeSplatter-O, but its code has not been open-sourced yet. Therefore, we choose to compare with two pose-dependent large reconstruction models, *i.e.*, LGM (Tang et al., 2024) and InstantMesh (Xu et al., 2024a), which are base on 3D Gaussians and tri-plane NeRF, respectively. We directly feed ground truth camera poses into them, while our model knows nothing about the cameras. For scene-level reconstruction, we compare with two state-of-the-art works on generalizable Gaussians, *i.e.*, pixelSplat (Charatan et al., 2024) and MVSplat (Chen et al., 2024b), both of which are pose-dependent. We fine-tune their checkpoints trained on RealEstate10K (Zhou et al., 2018) on ScanNet++ for more fair comparison,

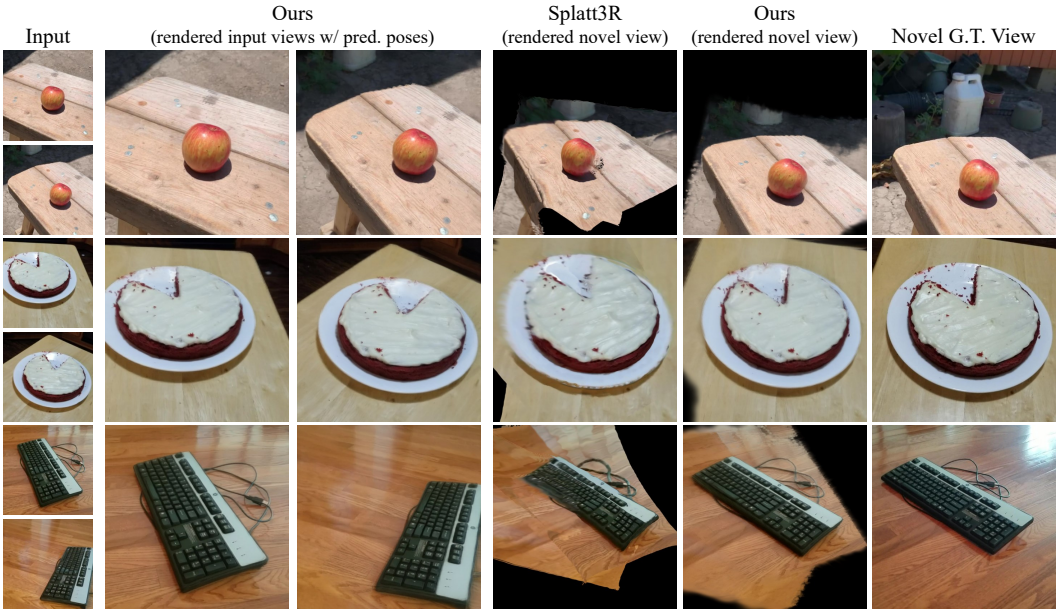

Figure 5: **Scene-level Reconstruction on CO3D.**

and do not apply them on the CO3Dv2 dataset due to the significant domain gap. Besides, we also compare with an up-to-date pose-free Gaussian reconstruction method, *i.e.*, Splatt3R (Smart et al., 2024), which utilizes the frozen MASt3R (Leroy et al., 2024) model for Gaussian location prediction, and adds an additional Gaussian head for predicting other Gaussian attributes.

Metrics. For both object-centric and scene-level reconstruction, We evaluate the rendering quality of reconstructed Gaussians and report the standard metrics for novel view synthesis, *i.e.*, PSNR, SSIM and LPIPS. All metrics are evaluated at the resolution of 512×512.

As Table 2 shows, although our FreeSplatter models utilizes no camera information as input, they outperforms their respective competitors on both obejct-centric and scene-level reconstruction tasks significantly. Specifically, our FreeSplatter-O model outperforms the two pose-dependent LRMs in all metrics and datasets by a large margin, with a PSNR gap > 5 on the GSO dataset and > 7 on the OmniObject3D dataset, both of which are unseen during training. The visualization results in Figure 3 also reveal the performance gap between the baselines and our model. InstantMesh and LGM show blurry artifacts (see the alphabets in the 4th column), while our rendering results are clear and preserve the details in the ground truth views.

Prior works on Large Reconstruction Models take it granted that accurate camera poses are required for high-quality 3D reconstruction, and inject the pose information (and usually intrinsics too) into the model through LayerNorm modulation layers (Li et al., 2024a) or plucker ray embeddings (Xu et al., 2024c; Tang et al., 2024; Xu et al., 2024b). The superior performance of FreeSplatter-O indicates that, **camera poses may not be essential to training high-quality and scalable large reconstruction models**.

In scene-level reconstruction, our FreeSplatter-S outperforms both pose-dependent models in almost all metrics, demonstrating its superior reconstruction quality and generalization capability. As for the pose-free competitor Splatt3R, we notice that although it benefits from the powerful 3D point regression capability of MASt3R (Leroy et al., 2024) learned from massive datasets, it is also influenced by MASt3R's wrong predictions, since it keeps the pre-trained MASt3R frozen and only trains the Gaussian head, leading to static Gaussian locations. In comparison, our model is trained end-to-end with direct rendering guidance, thus both the Gaussian locations and other attributes get updated. We visualize the view synthesis results on ScanNet++ and CO3D in Figure 4 and Figure 5 respectively, demonstrating the visual quality of our method. To be noted, the images of our FreeSplatter and Splatt3R are rendered by aligning the target cameras with the predicted cameras.

Table 1: **Quantitative results on camera pose estimation.** We highlight the best metric as red.

Method	OmniObject3D				GSO			
	RRE ↓	RRA@15° ↑	RRA@30° ↑	TE↓	RRE ↓	RRA@15° ↑	RRA@30° ↑	TE↓
FORGE	76.822	0.081	0.257	0.430	97.814	0.022	0.083	0.898
MASt3R	96.670	0.052	0.112	0.524	61.820	0.244	0.445	0.353
FreeSplatter-S	83.795	0.076	0.167	0.631	72.914	0.212	0.396	0.426
FreeSplatter-O	11.550	0.909	0.937	0.081	3.851	0.962	0.978	0.030
	ScanNet++				CO3Dv2			
	RRE ↓	RRA@15° ↑	RRA@30° ↑	TE↓	RRE ↓	RRA@15° ↑	RRA@30° ↑	TE↓
PoseDiffusion	-	-	-	-	7.950	0.803	0.868	0.328
MASt3R	0.724	0.988	0.993	0.104	2.918	0.975	0.989	0.112
FreeSplatter-S	0.791	0.982	0.987	0.110	3.054	0.976	0.986	0.148

Table 2: **Quantitative results on sparse-view reconstruction.**

Method	OmniObject3D			GSO		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
LGM (w/ gt pose)	24.852	0.942	0.060	24.463	0.891	0.093
InstantMesh (w/ gt pose)	24.077	0.945	0.062	25.421	0.891	0.095
Ours (FreeSplatter-O)	31.929	0.973	0.027	30.443	0.945	0.055
Method	ScanNet++			CO3Dv2		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
pixelSplat (w/ gt pose)	24.974	0.889	0.180	-	-	-
MVSplat (w/ gt pose)	22.601	0.862	0.208	-	-	-
Splatt3R	21.013	0.830	0.209	18.074	0.740	0.197
Ours (FreeSplatter-S)	25.807	0.887	0.140	20.405	0.781	0.162

4.3 Camera Pose Estimation

Baselines. To validate the pose estimation ability of our method, we compare with the state-of-the-art work in zero-shot multi-view pose estimation, *i.e.*, MASt3R. Besides, we also make a comparison with FORGE (Jiang et al., 2024a) and PoseDiffusion (Fan et al., 2023) on the object-centric and scene-level datasets, respectively. Since PoseDiffusion was not trained on ScanNet++, we only adopt it on the CO3Dv2 dataset for a fair comparison. We do not include COLMAP-based frameworks (Schonberger & Frahm, 2016) as our baselines, as we found that they exhibit a very high failure rate in our sparse-view setting, which was also observed by prior works (Wang et al., 2024a).

Metrics. Following Wang et al. (2024a) and Wang et al. (2023b), we split the evaluation of pose estimation into a rotation part and a translation part individually. For the rotation part, we report the relative rotation errors (RRE) in degrees, as while as relative rotation accuracy (RRA), which denotes the portion of rotation errors below a certain threshold. Following prior works, we adopt 15° and 30° as the thresholds. For the translation part, we report the translation errors (TE) between the predicted and ground truth camera centers. For more than 2 input views, we compute the errors between each pair of cameras and average them.

As depicted in Table 1, our FreeSplatter-O model surpasses other baselines by a huge margin in object-centric datasets, while FreeSplatter-S behaves on par with MASt3R in scene-level datasets. We observe that MASt3R exhibits terrible performance on rendered object-centric datasets without backgrounds. This is possibly due to the significant domain gap between the rendered object-centric images with the training datasets of MASt3R. Different from captured realistic scene images with complex backgrounds and textures, the rendered object-centric images are relatively low-textured and contain less notable features, making the pose estimation extremely challenging without sufficient data priors. On scene-level ScanNet++ and CO3D, MASt3R demonstrates the best performance while our FreeSplatter-S behaves on par with it. We attribute the slight gap in metrics to the scale of training datasets. We only train our FreeSplatter-S model on 3 datasets, which are exactly the subset of DUSt3R/MASt3R's training dataset. Therefore, MASt3R has learned more data priors than ours. We leave scaling up our model on more datasets as a future work.

4.4 Ablation Studies

Number of Input Views. We make an experiment on a GSO sample to illustrate how the number of input views influences the reconstruction quality on the object-centric scenarios. Please refer to Figure 15 in the appendix for more details.

Impact of Pixel-Alignment Loss. We study the impact of pixel-alignment loss defined in Equation 6. We remove the pixel-alignment loss from the training of both models, and report the novel

Table 3: **Ablation on the pixel-alignment loss.**

Method	GSO			Method	ScanNet++		
	PSNR↑	SSIM↑	LPIPS↓		PSNR↑	SSIM↑	LPISP↓
FreeSplatter-O (w/o $\mathcal{L}_{\text{align}}$)	26.684	0.898	0.092	FreeSplatter-S (w/o $\mathcal{L}_{\text{align}}$)	21.330	0.832	0.201
FreeSplatter-O	30.443	0.945	0.055	FreeSplatter-S	25.807	0.887	0.140

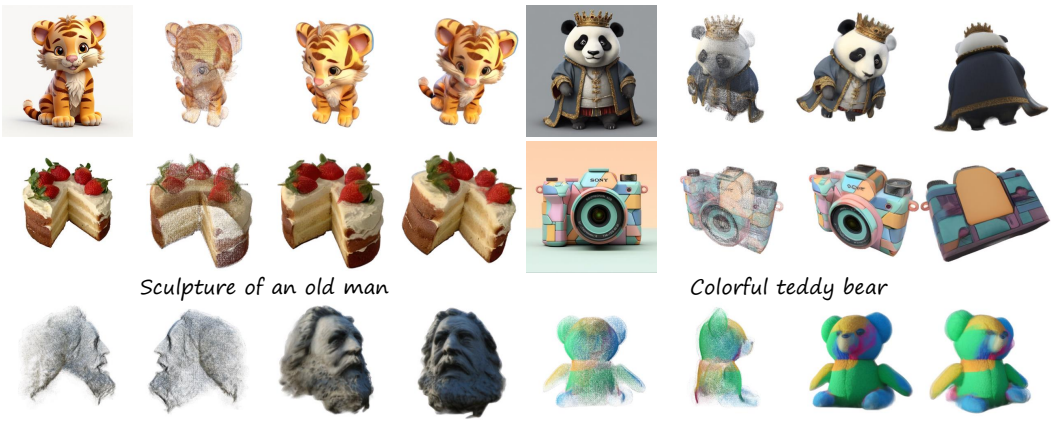

Figure 6: **3D content creation with FreeSplatter.** 1st and 2nd rows: image-to-3D generation results with Zero123++, we visulize the input image, reconstructed Gaussians, and two rendered novel views from left to right. 3rd row: text-to-3D generation results with MVDream, we annotate the text prompt on the top, and visualize two views of reconstructed Gaussians, and two rendered novel views from left to right.

view rendering metrics on GSO and ScanNet++ dataset. We can observe significant drops in all metrics without the pixel-alignment loss, which prove its effectiveness.

4.5 Applications

FreeSplatter can be seamlessly integrated into 3D content creation pipelines, and its pose-free nature can potentially bring great convenience to users and enhance the productivity by a large margin. In a classical 3D generation pipeline (Li et al., 2024a; Xu et al., 2024a; Tang et al., 2024; Wang et al., 2024c) that first generates multi-view images with a multi-view diffusion model and then feed them into an LRM for reconstruction, we have to figure out how the camera poses of the multi-view diffusion model are defined and carefully align them with the LRM. With FreeSplatter as the content creator, these tedious steps are not required any longer. We just feed the multi-view images into FreeSplatter and get the outputs instantly after several seconds. Figure 6 shows some results of text/image-to 3D generation with MVDream (Shi et al., 2024) and Zero123++ (Shi et al., 2023), respectively. We notice that our method can render significantly more clear views from the images than previous pose-dependent LRMs, showing stronger robustness to multi-view inconsistency.

5 Conclusion

In this work, we present FreeSplatter, a highly scalable framework for pose-free sparse-view reconstruction. By leveraging a single-stream transformer architecture and predicting multi-view Gaussian maps in a unified frame, FreeSplatter enables both high-fidelity 3D modeling and instant camera pose estimation. We provide two model variants for both object-centric and scene-level reconstruction, both showing superior recontruction qulity and pose estimation accuracy. FreeSplatter also exhibits great potential in enhancing the productivity of downstream applications like text/image-to-3D content creation, which can liberate users from dealing with complex camera poses.

Limitations. Despite the promising performance of FreeSplatter, its pre-training stage relies on depth data, making it non-trivial to be trained on datasets with no depth labels, *e.g.*, RealEstate10K and MVImgNet (Yu et al., 2023). Besides, FreeSplatter requires two distinct model variants to deal with object-centric and scene-level reconstructions, while a unified model for both tasks would be preferred. We leave this as the future work.

REFERENCES

Mark Boss, Zixuan Huang, Aaryaman Vasishta, and Varun Jampani. Sf3d: Stable fast 3d mesh reconstruction with uv-unwrapping and illumination disentanglement. *arXiv preprint arXiv:2408.00653*, 2024.

G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

Ang Cao, Justin Johnson, Andrea Vedaldi, and David Novotny. Lightplane: Highly-scalable components for neural 3d fields. *arXiv preprint arXiv:2404.19760*, 2024.

Eric R Chan, Connor Z Lin, Matthew A Chan, Koki Nagano, Boxiao Pan, Shalini De Mello, Orazio Gallo, Leonidas J Guibas, Jonathan Tremblay, Sameh Khamis, et al. Efficient geometry-aware 3d generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 16123–16133, 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.

Anpei Chen, Haofei Xu, Stefano Esposito, Siyu Tang, and Andreas Geiger. Lara: Efficient large-baseline radiance fields. *arXiv preprint arXiv:2407.04699*, 2024a.

Hansheng Chen, Pichao Wang, Fan Wang, Wei Tian, Lu Xiong, and Hao Li. Epro-pnp: Generalized end-to-end probabilistic perspective-n-points for monocular object pose estimation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 2781–2790, 2022.

Tianqi Chen, Bing Xu, Chiyuan Zhang, and Carlos Guestrin. Training deep nets with sublinear memory cost. *arXiv preprint arXiv:1604.06174*, 2016.

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. *arXiv preprint arXiv:2403.14627*, 2024b.

Jasmine Collins, Shubham Goel, Kenan Deng, Achleshwar Luthra, Leon Xu, Erhan Gundogdu, Xi Zhang, Tomas F Yago Vicente, Thomas Dideriksen, Himanshu Arora, et al. Abo: Dataset and benchmarks for real-world 3d object understanding. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 21126–21136, 2022.

Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. Objaverse: A universe of annotated 3d objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 13142–13153, 2023.

Matt Deitke, Ruoshi Liu, Matthew Wallingford, Huong Ngo, Oscar Michel, Aditya Kusupati, Alan Fan, Christian Laforte, Vikram Voleti, Samir Yitzhak Gadre, et al. Objaverse-xl: A universe of 10m+ 3d objects. *Advances in Neural Information Processing Systems*, 36, 2024.

Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superpoint: Self-supervised interest point detection and description. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pp. 224–236, 2018.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=YicbFdNTTy.

Laura Downs, Anthony Francis, Nate Koenig, Brandon Kinman, Ryan Hickman, Krista Reymann, Thomas B McHugh, and Vincent Vanhoucke. Google scanned objects: A high-quality dataset of 3d scanned household items. In *2022 International Conference on Robotics and Automation (ICRA)*, pp. 2553–2560. IEEE, 2022.

Mihai Dusmanu, Ignacio Rocco, Tomas Pajdla, Marc Pollefeys, Josef Sivic, Akihiko Torii, and Torsten Sattler. D2-net: A trainable cnn for joint description and detection of local features. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, pp. 8092–8101, 2019.

Johan Edstedt, Ioannis Athanasiadis, Mårten Wadenbäck, and Michael Felsberg. Dkm: Dense kernelized feature matching for geometry estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 17765–17775, 2023.

Johan Edstedt, Qiyu Sun, Georg Bökman, Mårten Wadenbäck, and Michael Felsberg. Roma: Robust dense feature matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 19790–19800, 2024.

Zhiwen Fan, Panwang Pan, Peihao Wang, Yifan Jiang, Hanwen Jiang, Dejia Xu, Zehao Zhu, Dilin Wang, and Zhangyang Wang. Pose-free generalizable rendering transformer. *arXiv e-prints*, pp. arXiv–2310, 2023.

Zhiwen Fan, Wenyan Cong, Kairun Wen, Kevin Wang, Jian Zhang, Xinghao Ding, Danfei Xu, Boris Ivanovic, Marco Pavone, Georgios Pavlakos, et al. Instantsplat: Unbounded sparse-view pose-free gaussian splatting in 40 seconds. *arXiv preprint arXiv:2403.20309*, 2024.

Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24 (6):381–395, 1981.

Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Sunghwan Hong, Jaewoo Jung, Heeseong Shin, Jiaolong Yang, Seungryong Kim, and Chong Luo. Unifying correspondence pose and nerf for generalized pose-free novel view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 20196–20206, 2024a.

Yicong Hong, Kai Zhang, Jiuxiang Gu, Sai Bi, Yang Zhou, Difan Liu, Feng Liu, Kalyan Sunkavalli, Trung Bui, and Hao Tan. LRM: Large reconstruction model for single image to 3d. In *The Twelfth International Conference on Learning Representations*, 2024b. URL https://openreview.net/forum?id=sllU8vvsFF.

Hanwen Jiang, Zhenyu Jiang, Kristen Grauman, and Yuke Zhu. Few-view object reconstruction with unknown categories and camera poses. In *2024 International Conference on 3D Vision (3DV)*, pp. 31–41. IEEE, 2024a.

Hanwen Jiang, Zhenyu Jiang, Yue Zhao, and Qixing Huang. LEAP: Liberate sparse-view 3d modeling from camera poses. In *The Twelfth International Conference on Learning Representations*, 2024b. URL https://openreview.net/forum?id=KPmajBxEaF.

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

Benjamin Lefaudeux, Francisco Massa, Diana Liskovich, Wenhan Xiong, Vittorio Caggiano, Sean Naren, Min Xu, Jieru Hu, Marta Tintore, Susan Zhang, Patrick Labatut, Daniel Haziza, Luca Wehrstedt, Jeremy Reizenstein, and Grigory Sizov. xformers: A modular and hackable transformer modelling library. https://github.com/facebookresearch/xformers, 2022.

Vincent Leroy, Yohann Cabon, and Jérôme Revaud. Grounding image matching in 3d with mast3r. *arXiv preprint arXiv:2406.09756*, 2024.

Jiahao Li, Hao Tan, Kai Zhang, Zexiang Xu, Fujun Luan, Yinghao Xu, Yicong Hong, Kalyan Sunkavalli, Greg Shakhnarovich, and Sai Bi. Instant3d: Fast text-to-3d with sparse-view generation and large reconstruction model. In *The Twelfth International Conference on Learning Representations*, 2024a. URL https://openreview.net/forum?id=2lDQLiH1W4.

Mengfei Li, Xiaoxiao Long, Yixun Liang, Weiyu Li, Yuan Liu, Peng Li, Xiaowei Chi, Xingqun Qi, Wei Xue, Wenhan Luo, et al. M-lrm: Multi-view large reconstruction model. *arXiv preprint arXiv:2406.07648*, 2024b.

Chen-Hsuan Lin, Wei-Chiu Ma, Antonio Torralba, and Simon Lucey. Barf: Bundle-adjusting neural radiance fields. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 5741–5751, 2021.

Philipp Lindenberger, Paul-Edouard Sarlin, and Marc Pollefeys. Lightglue: Local feature matching at light speed. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 17627–17638, 2023.

Minghua Liu, Chong Zeng, Xinyue Wei, Ruoxi Shi, Linghao Chen, Chao Xu, Mengqi Zhang, Zhaoning Wang, Xiaoshuai Zhang, Isabella Liu, et al. Meshformer: High-quality mesh generation with 3d-guided reconstruction model. *arXiv preprint arXiv:2408.10198*, 2024a.

Tianqi Liu, Guangcong Wang, Shoukang Hu, Liao Shen, Xinyi Ye, Yuhang Zang, Zhiguo Cao, Wei Li, and Ziwei Liu. Fast generalizable gaussian splatting reconstruction from multi-view stereo. *arXiv preprint arXiv:2405.12218*, 2024b.

I Loshchilov. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Paulius Micikevicius, Sharan Narang, Jonah Alben, Gregory Diamos, Erich Elsen, David Garcia, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, et al. Mixed precision training. *arXiv preprint arXiv:1710.03740*, 2017.

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.

Frank Plastria. The weiszfeld algorithm: proof, amendments, and extensions. *Foundations of location analysis*, pp. 357–389, 2011.

Jeremy Reizenstein, Roman Shapovalov, Philipp Henzler, Luca Sbordone, Patrick Labatut, and David Novotny. Common objects in 3d: Large-scale learning and evaluation of real-life 3d category reconstruction. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 10901–10911, 2021.

Jerome Revaud, Cesar De Souza, Martin Humenberger, and Philippe Weinzaepfel. R2d2: Reliable and repeatable detector and descriptor. *Advances in neural information processing systems*, 32, 2019.

Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superglue: Learning feature matching with graph neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 4938–4947, 2020.

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.

Ruoxi Shi, Hansheng Chen, Zhuoyang Zhang, Minghua Liu, Chao Xu, Xinyue Wei, Linghao Chen, Chong Zeng, and Hao Su. Zero123++: a single image to consistent multi-view diffusion base model. *arXiv preprint arXiv:2310.15110*, 2023.

Yichun Shi, Peng Wang, Jianglong Ye, Long Mai, Kejie Li, and Xiao Yang. MVDream: Multi-view diffusion for 3d generation. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=FUgrjq2pbB.

Yawar Siddiqui, Tom Monnier, Filippos Kokkinos, Mahendra Kariya, Yanir Kleiman, Emilien Garreau, Oran Gafni, Natalia Neverova, Andrea Vedaldi, Roman Shapovalov, et al. Meta 3d assetgen: Text-to-mesh generation with high-quality geometry, texture, and pbr materials. *arXiv preprint arXiv:2407.02445*, 2024.

Brandon Smart, Chuanxia Zheng, Iro Laina, and Victor Adrian Prisacariu. Splatt3r: Zero-shot gaussian splatting from uncalibarated image pairs. *arXiv preprint arXiv:2408.13912*, 2024.

Stanislaw Szymanowicz, Chrisitian Rupprecht, and Andrea Vedaldi. Splatter image: Ultra-fast single-view 3d reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10208–10217, 2024.

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

Shimon Ullman. The interpretation of structure from motion. *Proceedings of the Royal Society of London. Series B. Biological Sciences*, 203(1153):405–426, 1979.

Hengyi Wang and Lourdes Agapito. 3d reconstruction with spatial memory. *arXiv preprint arXiv:2408.16061*, 2024.

Jianyuan Wang, Nikita Karaev, Christian Rupprecht, and David Novotny. Visual geometry grounded deep structure from motion. *arXiv preprint arXiv:2312.04563*, 2023a.

Jianyuan Wang, Christian Rupprecht, and David Novotny. Posediffusion: Solving pose estimation via diffusion-aided bundle adjustment. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9773–9783, 2023b.

Peng Wang, Hao Tan, Sai Bi, Yinghao Xu, Fujun Luan, Kalyan Sunkavalli, Wenping Wang, Zexiang Xu, and Kai Zhang. PF-LRM: Pose-free large reconstruction model for joint pose and shape prediction. In *The Twelfth International Conference on Learning Representations*, 2024a. URL https://openreview.net/forum?id=noe76eRcPC.

Shuzhe Wang, Juho Kannala, Marc Pollefeys, and Daniel Barath. Guiding local feature matching with surface curvature. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 17981–17991, 2023c.

Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jerome Revaud. Dust3r: Geometric 3d vision made easy. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 20697–20709, 2024b.

Zhengyi Wang, Yikai Wang, Yifei Chen, Chendong Xiang, Shuo Chen, Dajiang Yu, Chongxuan Li, Hang Su, and Jun Zhu. Crm: Single image to 3d textured mesh with convolutional reconstruction model. *arXiv preprint arXiv:2403.05034*, 2024c.

Xingkui Wei, Yinda Zhang, Zhuwen Li, Yanwei Fu, and Xiangyang Xue. Deepsfm: Structure from motion via deep bundle adjustment. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I 16*, pp. 230–247. Springer, 2020.

Xinyue Wei, Kai Zhang, Sai Bi, Hao Tan, Fujun Luan, Valentin Deschaintre, Kalyan Sunkavalli, Hao Su, and Zexiang Xu. Meshlrm: Large reconstruction model for high-quality mesh. *arXiv preprint arXiv:2404.12385*, 2024.

Christopher Wewer, Kevin Raj, Eddy Ilg, Bernt Schiele, and Jan Eric Lenssen. latentsplat: Autoencoding variational gaussians for fast generalizable 3d reconstruction. *arXiv preprint arXiv:2403.16292*, 2024.

Tong Wu, Jiarui Zhang, Xiao Fu, Yuxin Wang, Jiawei Ren, Liang Pan, Wayne Wu, Lei Yang, Jiaqi Wang, Chen Qian, et al. Omniobject3d: Large-vocabulary 3d object dataset for realistic perception, reconstruction and generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 803–814, 2023.

Jiale Xu, Weihao Cheng, Yiming Gao, Xintao Wang, Shenghua Gao, and Ying Shan. Instantmesh: Efficient 3d mesh generation from a single image with sparse-view large reconstruction models. *arXiv preprint arXiv:2404.07191*, 2024a.

Yinghao Xu, Zifan Shi, Wang Yifan, Hansheng Chen, Ceyuan Yang, Sida Peng, Yujun Shen, and Gordon Wetzstein. Grm: Large gaussian reconstruction model for efficient 3d reconstruction and generation. *arXiv preprint arXiv:2403.14621*, 2024b.

Yinghao Xu, Hao Tan, Fujun Luan, Sai Bi, Peng Wang, Jiahao Li, Zifan Shi, Kalyan Sunkavalli, Gordon Wetzstein, Zexiang Xu, and Kai Zhang. DMV3d: Denoising multi-view diffusion using 3d large reconstruction model. In *The Twelfth International Conference on Learning Representations*, 2024c. URL https://openreview.net/forum?id=H4yQefeXhp.

Fan Yang, Jianfeng Zhang, Yichun Shi, Bowen Chen, Chenxu Zhang, Huichao Zhang, Xiaofeng Yang, Jiashi Feng, and Guosheng Lin. Magic-boost: Boost 3d generation with mutli-view conditioned diffusion. *arXiv preprint arXiv:2404.06429*, 2024a.

Xianghui Yang, Huiwen Shi, Bowen Zhang, Fan Yang, Jiacheng Wang, Hongxu Zhao, Xinhai Liu, Xinzhou Wang, Qingxiang Lin, Jiaao Yu, et al. Hunyuan3d-1.0: A unified framework for text-to-3d and image-to-3d generation. *arXiv preprint arXiv:2411.02293*, 2024b.

Yao Yao, Zixin Luo, Shiwei Li, Jingyang Zhang, Yufan Ren, Lei Zhou, Tian Fang, and Long Quan. Blendedmvs: A large-scale dataset for generalized multi-view stereo networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 1790–1799, 2020.

Chandan Yeshwanth, Yueh-Cheng Liu, Matthias Nießner, and Angela Dai. Scannet++: A high-fidelity dataset of 3d indoor scenes. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 12–22, 2023.

Xianggang Yu, Mutian Xu, Yidan Zhang, Haolin Liu, Chongjie Ye, Yushuang Wu, Zizheng Yan, Chenming Zhu, Zhangyang Xiong, Tianyou Liang, et al. Mvimgnet: A large-scale dataset of multi-view images. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 9150–9161, 2023.

Chubin Zhang, Hongliang Song, Yi Wei, Yu Chen, Jiwen Lu, and Yansong Tang. Geolrm: Geometry-aware large reconstruction model for high-quality 3d gaussian generation. *arXiv preprint arXiv:2406.15333*, 2024a.

Kai Zhang, Nick Kolkin, Sai Bi, Fujun Luan, Zexiang Xu, Eli Shechtman, and Noah Snavely. Arf: Artistic radiance fields, 2022.

Kai Zhang, Sai Bi, Hao Tan, Yuanbo Xiangli, Nanxuan Zhao, Kalyan Sunkavalli, and Zexiang Xu. Gs-lrm: Large reconstruction model for 3d gaussian splatting. *arXiv preprint arXiv:2404.19702*, 2024b.

Xin-Yang Zheng, Hao Pan, Yu-Xiao Guo, Xin Tong, and Yang Liu. Mvdˆ 2: Efficient multiview 3d reconstruction for multiview diffusion. In *ACM SIGGRAPH 2024 Conference Papers*, pp. 1–11, 2024.

Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: Learning view synthesis using multiplane images. *ACM Trans. Graph*, 37, 2018.

A Appendix

A.1 Implementation Details

Implementation Details. FreeSplatter-O and FreeSplatter-S share the same model architecture of a 24-layer transformer with a width of 1024, containing $\sim$ 306 million parameters. We do not use bias term throughout the model except for the output linear layer for Gaussian prediction. The model is trained on 256×256 input images for 500K steps and then fine-tuned on 512×512 input images for 100K steps on 16 NVIDIA H800 GPUs. We adopt an AdamW (Loshchilov, 2017) optimizer with $\beta_1 = 0.9, \beta_2 = 0.95$, the weight decay and learning rate are set to 0.01 and 4×10^{-5}, respectively. The maximum pre-training step $T_{\max} = 2 \times 10^5$, *i.e.*, we adopt the Gaussian position loss in Equation 5 for the first 200K training steps. To enhance the training efficiency, we utilize the xFormers (Lefaudeux et al., 2022) library for memory-efficient attention, gradient checkpointing (Chen et al., 2016), bfloat16 training (Micikevicius et al., 2017), and deferred back-propagation (Zhang et al., 2022) for GS rendering. In each batch, We sample 4 and 2 input views for FreeSplatter-O and FreeSplatter-S respectively and 4 novel views for supervision.

Camera Intrinsic Estimation. Following DUSt3R (Wang et al., 2024b), we assume centered principle point and square pixels in this work. Thanks to the point-cloud-based Gaussian representation, we can extract the Gaussian locations as "point maps" $\left\{\boldsymbol{X}^n \in \mathbb{R}^{H \times W \times 3} \mid n = 1, \dots, N\right\}$ from the predicted Gaussian maps. Since all points are predicted in the first view's camera frame, we can estimate the focal length of the first view by minimizing the pixel-projection errors:

$$f^1_* = \arg\min_{f^1} \sum_{i=0}^{W} \sum_{j=0}^{H} \left\| (i', j') - f^1 \frac{\left(X^1_{i,j,0}, X^1_{i,j,1}\right)}{X^1_{i,j,2}} \right\|, \tag{8}$$

where $(i', j') = (i - \frac{W}{2}, j - \frac{H}{2})$ denotes pixel coordinates relative to the principle point $\left(\frac{H}{2}, \frac{W}{2}\right)$. This optimization problem can be solved by Weiszfeld algorithm (Plastria, 2011) efficiently.

Different from DUSt3R that only accepts binocular images as input, our model architecture naturally supports single-view reconstruction. Thus we feed each image $\boldsymbol{I}^n$ into the model individually to predict its monocular Gaussian map $\boldsymbol{G}^n_{\text{mono}}$, and then estimate its focal length f^n from $\boldsymbol{X}^n_{\text{mono}}$. Finally, we compute the average focal length and adopt it for all views: $f = \frac{1}{N} \sum_{n=1}^{N} f^n$.

Masks for PnP-RANSAC solver. We adopt different masks $\boldsymbol{M}^n$ in Equation 4 for object-centric and scene-level reconstruction when estimating the input camera poses:

- For object-centric reconstruction, the model is trained on white-background images rendered from 3D assets, and the predicted Gaussians at the background area are not restricted to be pixel-aligned (See Section 3.3 for details), which may influence the camera pose solving. Therefore, we assume white-background input images at inference time, and adopt the foreground mask segmented by an off-the-shelf background-removal tool[1] as $\boldsymbol{M}^n$. The segmentation masks are not always perfect but are precise enough for the PnP-RANSAC solver to estimate the poses robustly.
- For scene-level reconstruction, we do not distinguish the foreground and background areas during training, and all the predicted Gaussians are restricted to be pixel-aligned by applying the pixel-alignment loss universally. Therefore, all visible Gaussians should contribute to pose solving and we use the Gaussian opacity map $\boldsymbol{O}^n \in \mathbb{R}^{H \times W}$ to compute $\boldsymbol{M}^n$ with a minimal visibility threshold τ: $\boldsymbol{M}^n = (\boldsymbol{O}^n > \tau)$.

Camera Normalization. The definition of reconstruction frame plays an important role in model training. We take the camera frame of the first view as the reference frame for reconstruction, and normalize all cameras in a training batch into this frame. Besides, a scaling operation is required to deal with various scene scales. Denoting the original input cameras as $\{\boldsymbol{P}^n_{\text{in}} = [\boldsymbol{R}^n_{\text{in}} \mid \boldsymbol{t}^n_{\text{in}}] \mid n = 1, \dots, N\}$, we adopt different camera normalization strategies for object-centric and scene-level reconstruction.

For object-centric reconstruction, the training cameras are sampled orbiting the object center. We first scale all input cameras to make the distance from the first camera to the object center equal to

[1] https://github.com/danielgatis/rembg

2, and then transform all cameras into the reference frame so that the first view's camera pose is an identity matrix. With this strategy, we fix the object center at $(0, 0, 2)$ in the reference frame. For scene-level reconstruction, the camera distributions are more complex. Thus we fist transform all cameras into the reference frame, and then scale them using a factor $s = 1/d$, d is the mean distance of all valid points to the origin:

$$d = \frac{1}{\sum_{n=1}^{N} |\boldsymbol{M}^n|} \sum_{n=1}^{N} \sum_{i=0} \sum_{j=0} M_{i,j}^n \cdot \left\| \boldsymbol{X}_{i,j}^n \right\|, \tag{9}$$

where $\boldsymbol{X}^n \in \mathbb{R}^{H \times W \times 3}$ denotes the ground truth point map unprojected from depth map, $\boldsymbol{M}^n \in \mathbb{R}^{H \times W}$ is the valid depth mask. To be noted, we also scale the ground truth depths when scaling the cameras.

Scene Scale Ambiguity. Given a set of scene images, we can not infer the actual scale of the scene, since we can obtain identical rendered images when scaling the scene and cameras simultaneously, *i.e.*, the scene scale ambiguity problem. For object-centric reconstruction, we have fixed the object center at $(0, 0, 2)$ in the reference frame, so there is no scale ambiguity problem. But for scene images, it is hard to define a canonical center and it is reasonable for our model to reconstruct the scene in arbitrary scale. Besides, the training data is a mixture of multiple datasets that vary greatly in scale, which may make the model confused on scale prediction. Then a question arises, how can we align the scale of the model's prediction with the scale of the training cameras so that we can render target views for supervision?

As mentioned above, we scale the camera locations in a training batch using $s = 1/d$ (d is defined in Equation 9), which can bound the scene into a range reasonable for the model to predict. However, the scale factor is related to the choice of valid points, leading to different scene scales with a minimal difference in selected points. The model would get confused if we train it to reconstruct exactly in this scale since it has no idea which points we utilized to compute this scale. Inspired by DUSt3R (Wang et al., 2024b), we choose to rescale the reconstructed Gaussians before rendering them to deal with scene ambiguity. We first compute the scale factor $\hat{s} = 1/\hat{d}$ using Equation 9 with predicted Gaussian positions $\hat{\boldsymbol{X}}^n$ and the same mask $\boldsymbol{M}^n$. Then we adjust the *position* and *scale* of each Gaussian as $\hat{\boldsymbol{\mu}}_k = \hat{s} \cdot \boldsymbol{\mu}_k, \hat{\boldsymbol{s}}_k = \hat{s} \cdot \boldsymbol{s}_k$. With this operation, we do not care about the absolute scale of the reconstructed scene, but we expect it to align with the ground truth scene after scaling them using the factors $\hat{s}$ and s respectively. In the pre-training stage, we compute $\mathcal{L}_{\text{pos}}$ using the scaled predicted positions and ground truth positions too.

A.2 Additional Experiments

A.2.1 Comparison with PF-LRM

PF-LRM is a highly-relevant work to our FreeSplatter-O model. We make a comparison with it on its GSO and OmniObject3D evaluation datasets, both quantitatively and qualitatively.

Quantitative comparison. We show the quantitative results in Table 4 and Table 5. On both sparse-view reconstruction and pose estimation tasks, our model achieves lower metrics than PF-LRM on GSO dataset and higher metrics on Omni3D dataset. However, we point out that the images in PF-LRM's GSO evaluation set are rendered with the same rendering engine and hyper-parameters (e.g., light intensity) as PF-LRM's training dataset, which is beneficial for PF-LRM's evaluaton metrics due to the smaller gap between its training and evaluation datasets. On the contrary, the images in the Omni3D evaluation set are not manually rendered but are from the original dataset itself. Therefore, we argue that Omni3D is a more fair benchmark for comparison, and our model consistently outperforms PF-LRM on this benchmark.

Qualitative comparison. We also show qualitative comparison results on both datasets in Figure 7, which demonstrate that our FreeSplatter synthesizes significantly better visual details than PF-LRM.

A.2.2 More Results on Image-to-3D Generation

In this section, we provide **a large set** of visualizations on image-to-3D generation by combining our FreeSplatter-O with different multi-view diffusion models.

Table 4: **Quantitative comparison with PF-LRM on sparse-view reconstruction.**

Method	GSO			OmniObject3D		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
	Evaluate renderings at G.T. novel-view poses					
PF-LRM	25.08	0.877	0.095	21.77	0.866	0.097
FreeSplatter-O	23.54	0.864	0.100	22.83	0.876	0.088
	Evaluate renderings at predicted input poses					
PF-LRM	27.10	0.905	0.065	25.86	0.901	0.062
FreeSplatter-O	25.50	0.897	0.076	26.49	0.926	0.050

Table 5: **Quantitative comparison with PF-LRM on pose estimation.**

Method	GSO				OmniObject3D			
	RRE ↓	RRA@15° ↑	RRA@30° ↑	TE↓	RRE ↓	RRA@15° ↑	RRA@30° ↑	TE↓
PF-LRM	3.99	0.956	0.976	0.041	8.013	0.889	0.954	0.089
FreeSplatter-O	8.96	0.909	0.936	0.090	3.446	0.982	0.996	0.039

Comparison with pose-dependent LRMs. In Figure 8 and Figure 9, we compare FreeSplatter's image-to-3D generation results with pose-dependent LRMs, *i.e.*, InstantMesh and LGM, using Zero123++ v1.2 (Shi et al., 2023) and a recent model Hunyuan3D Std (Yang et al., 2024b) as the multi-view generator, respectively. For each input image, we fix the random seed to generate multi-view images, and visualize the novel view synthesis results of each reconstruction model. From the results, we can observe that FreeSplatter generates significantly more clear views and preserves the geometry and texture details better than other baselines.

Using input image to enhance the results. In Figure 10, we show another interesting use case of FreeSplatter, i.e., using the input image to enhance the image-to-3D generation results. For multi-view diffusion models like Zero123++ v1.1 (Shi et al., 2023), it generates 6 views from an input image at pre-defined poses, but **the pose of the input image is unknown** (its azimuth is known to be 0° but elevation is unknown). In this case, classical pose-dependent LRMs cannot leverage the input image for reconstruction, but FreeSplatter is able to do this! As Figure 10 shows, using the input image apart from generated views can significantly enhance the reconstruction results in many cases, especially for contents that Zero123++ struggles to generate, *e.g.*, human faces. Compared to generated views, the input image is often more high-quality and contains richer visual details. The pose-free nature of FreeSplatter make it capable of exploiting these precious information in the reconstruction process.

Recovering the predefined camera poses of multi-view diffusion models. In Figure 11, we show that FreeSplatter can faithfully recover the pre-defined camera poses of existing multi-view diffusion models from its reconstructed Gaussian maps. This demonstrates its robustness to generated multi-view images which may contain inconsistent contents.

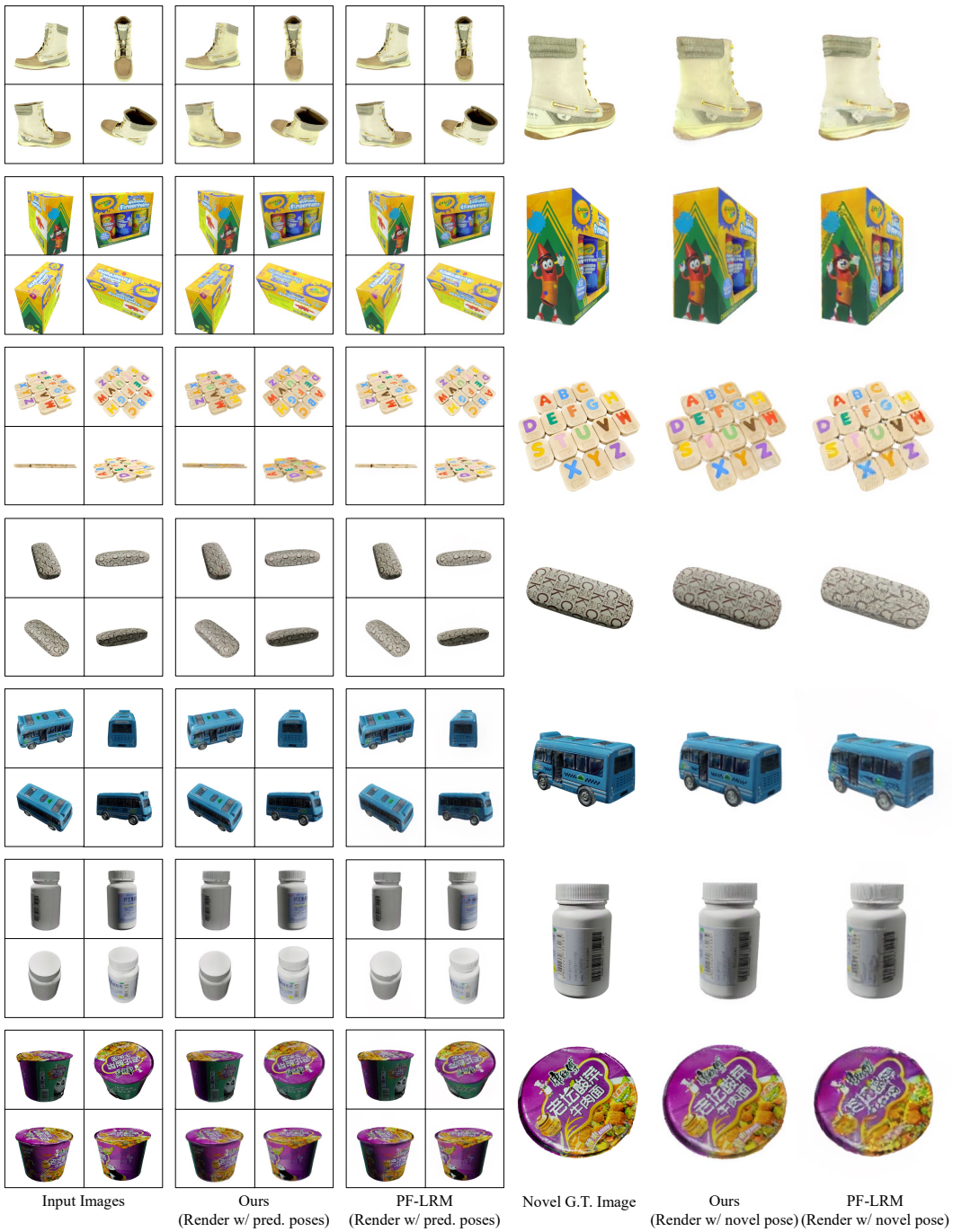

Figure 7: **Comparison with PF-LRM on its evaluation datasets.** The test samples in the first 3 rows are from the GSO evaluation set of PF-LRM, while the samples in the last 4 rows are from the OmniObject3D evaluation set of PF-LRM.

Figure 8: **Comparison on image-to-3D generation with Zero123++ v1.2.**

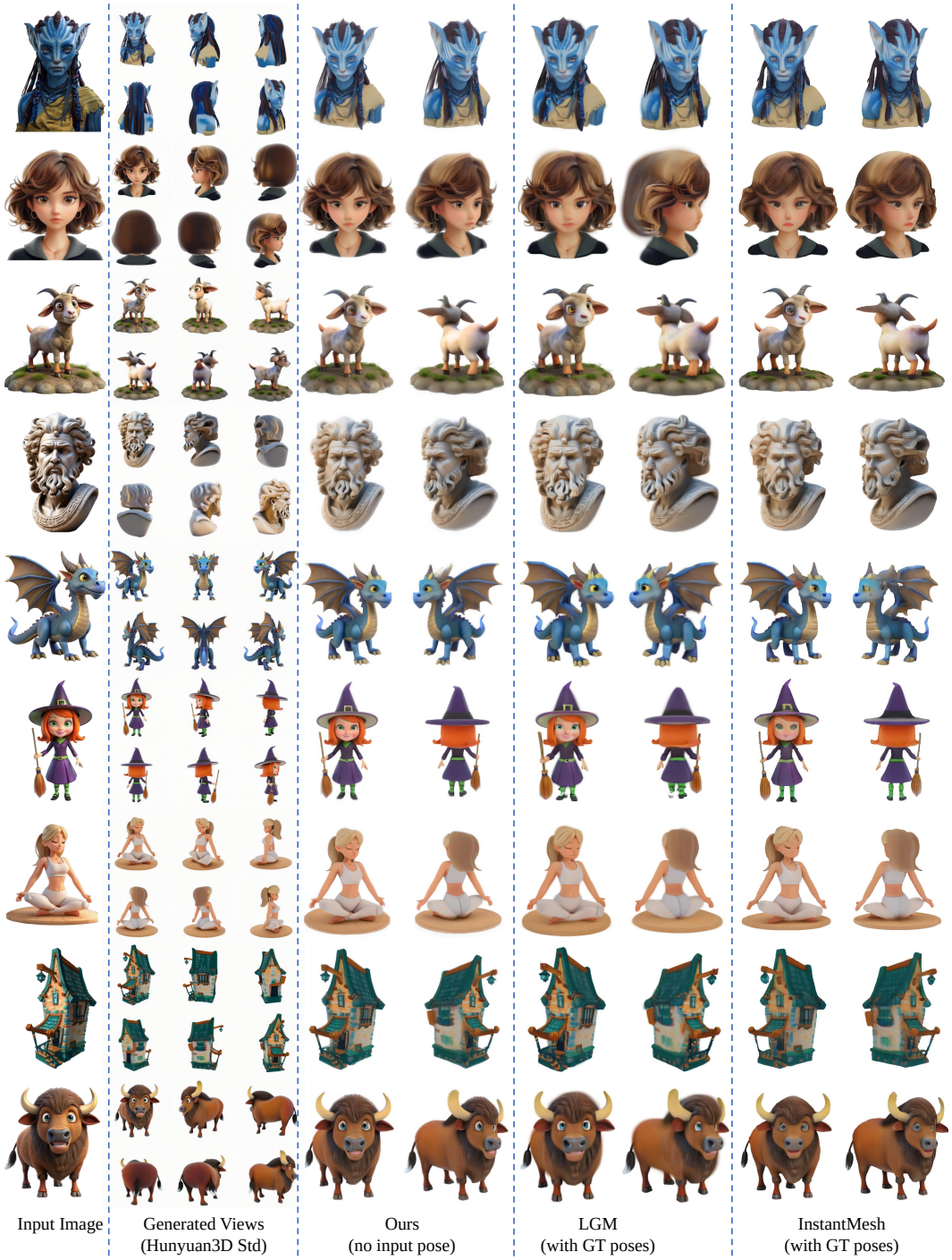

Figure 9: **Comparison on image-to-3D generation with Hunyuan3D Std.**

Figure 10: **Use the input image for image-to-3D generation with Zero123++ v1.1.**

Figure 11: **Our FreeSplatter can faithfully recover the pre-defined camera poses of existing multi-view diffusion models.** We use gray pyramids to visualize the ground truth pre-defined camera poses of the diffusion models, and colorful pyramids to visualize the estimated poses.

A.2.3 CROSS-DATASET GENERALIZATION

FreeSplatter-O generalizes to other object datasets. To demonstrate the cross-dataset generalization ability of our FreeSplatter-O, we visualize the object-centric reconstruction results on OmniObject3D and ABO (Collins et al., 2022) datasets in Figure 12. Both datasets are outside the training scope of FreeSplatter-O. We can observe that FreeSplatter-O generates very high-quality renderings.

FreeSplatter-S generalizes to RealEstate10K. Although our FreeSplatter-S model was not trained on RealEstate10K (Zhou et al., 2018), we directly utilize it for zero-shot view synthesis on this dataset. We visualize the results in Figure 13. We can observe that our model can faithfully reconstruct the input views at the estimated input poses, and the rendered novel views align well with the ground truth. We also report the quantitative results of both sparse-view reconstruction and pose-estimation tasks in Table 6 and Table 7, respectively. In summary, pixelSplat and MVSplat achieve better metrics on the sparse-view reconstruction task, which is foreseeable since they utilize RealEstate10K as training dataset and ground truth camera poses as input, while our model is not trained on this dataset and leverages no camera information. However, our model significantly outperforms another pose-free baseline Splatt3R which is built upon MASt3R trained on a larger dataset, demonstrating the effectiveness of our framework. For camera pose estimation on RealEstate10K, MASt3R achieves the best Relative Rotation Error (RRE) while our FreeSplatter-S performs the best in all other metrics.

Table 6: **Quantitative sparse-view reconstruction results on RealEstate10K.**

Method	PSNR ↑	SSIM ↑	LPIPS ↓
pixelSplat (w/ GT poses)	24.469	0.829	0.224
MVSplat (w/ GT poses)	20.033	0.789	0.280
Splatt3R (no pose)	16.634	0.604	0.422
FreeSplatter-S (no pose)	18.851	0.659	0.369

Table 7: **Quantitative pose estimation results on RealEstate10K.**

Method	RRE ↓	RRA@15° ↑	RRA@30° ↑	TE↓
PoseDiffsion	14.387	0.732	0.780	0.466
RayDiffsion	12.023	0.767	0.814	0.439
RoMa	5.663	0.918	0.947	0.402
MASt3R	2.341	0.972	0.994	0.374
FreeSplatter-S	3.513	0.982	0.992	0.293

FreeSplatter-S generalizes to various other datasets. We demonstrate FreeSplatter-S's broad generalization capabilities through extensive evaluation across diverse datasets in Figure 14. Our results span scanned objects (DTU dataset), large-scale outdoor scenes (Tanks & Temples), multi-view object collections (MVImgNet), and AI-generated synthetic contents (Sora-generated videos). These results validate our framework's robustness across varying scene types and capture conditions.

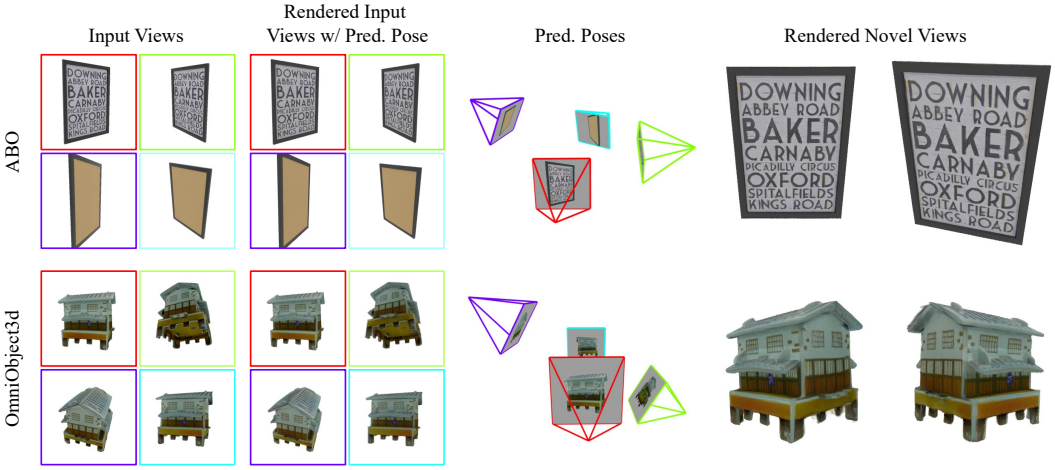

Figure 12: **Zero-shot pose-free reconstruction results on ABO and OmniObject3D.** Both datasets are unseen for FreeSplatter-O. Our model can faithfully estimate the input camera poses and render high-fidelity novel views.

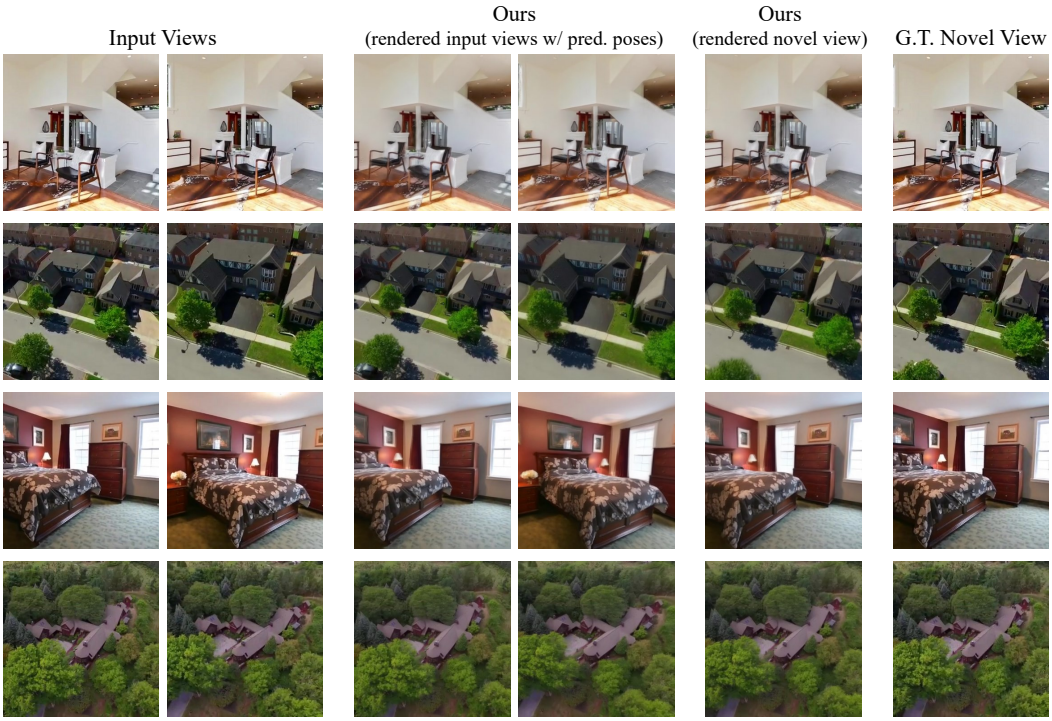

Figure 13: **Zero-shot pose-free reconstruction and view synthesis results on RealEstate10K.** Our FreeSplatter-S model was not trained on RealEstate10K, we directly utilize it for zero-shot view synthesis on this dataset. We can observe that our model can faithfully reconstruct the input views at the estimated input poses, and the rendered novel views align well with the ground truth.

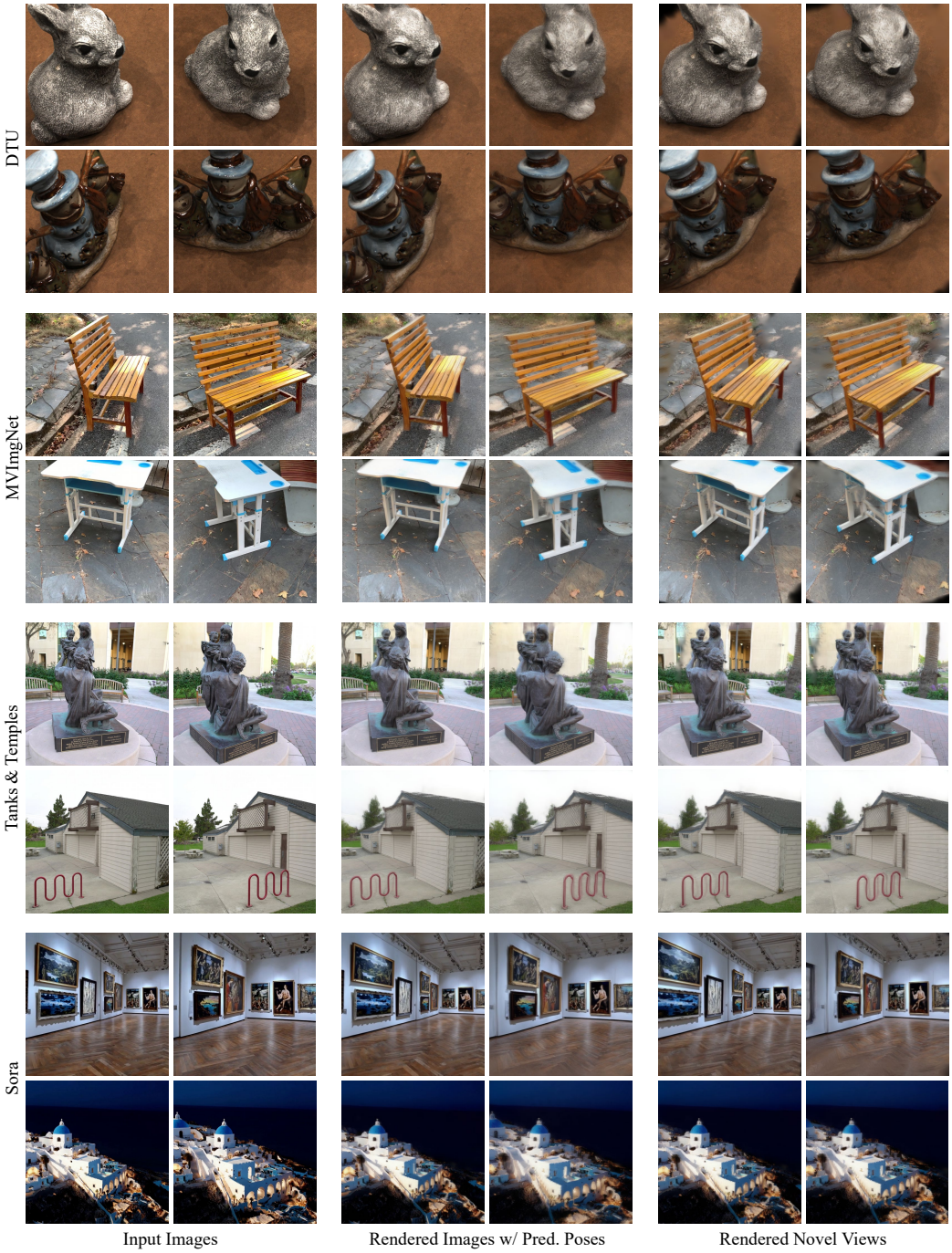

Figure 14: **Zero-shot generalization of FreeSplatter-S on various datasets.** We show 2 examples for each dataset.

A.3 Ablation Studies

Model architecture. Our FreeSplatter model is a 24-layer transformer with a patch size of 8. In our initial experiments of this project, we have trained object-level reconstruction models with different layers and patch size. We compare these models with our final model to evaluate the influence of model architecture on the performance. Specifically, we evaluate on the Google Scanned Objects dataset and report the results in Table 8. The results demonstrate that, using more layers and smaller patch size can consistently improve the model's performance. This is reasonable since more transformer layers leads to more parameters, while smaller patch size leads to less memory usage (less image tokens for attention computation) but higher information compression ratio.

Table 8: **Ablation study on model architectures. The results are evaluated on GSO dataset. L and P denote number of transformer layers and patch size, respectively.**

Architecture	PSNR ↑	SSIM ↑	LPIPS ↓
$L = 16, P = 16$	25.417	0.896	0.088
$L = 16, P = 8$	28.945	0.934	0.064
$L = 24, P = 16$	28.622	0.927	0.063
$L = 24, P = 8$	30.443	0.945	0.055

View embedding addition. As suggested by the reviewer, we also evaluate the effectiveness of adding view embeddings to images tokens. In FreeSplatter, we add the multi-view image tokens with a reference view embedding $\mathbf{e}_{\text{ref}}$ or a source view embedding $\mathbf{e}_{\text{src}}$ before feeding them into the transformer to make the model identify the reference view, so that it can reconstruct Gaussians in the reference view's camera frame. We present the results in Figure 16 of the revised paper.

Specifically, for 4 input views V_i ($i = 1, 2, 3, 4$), we try different combinations of $\mathbf{e}_{\text{ref}}$ and $\mathbf{e}_{\text{src}}$ when adding them to the image tokens, and then render the reconstructed Gaussians with an "identity camera pose" $C = [[1, 0, 0, 0], [0, 1, 0, 0], [0, 0, 1, 0], [0, 0, 0, 1]]$ to see the results. We can observe that, when adding the j-th view's tokens with $\mathbf{e}_{\text{ref}}$ and other views' tokens with $\mathbf{e}_{\text{src}}$, the rendered image is exactly the j-th view. This means that the model successfully identify the j-th view as the reference view and reconstruct Gaussians in its camera frame. In comparison, all other view embedding combinations leads to degraded reconstruction.

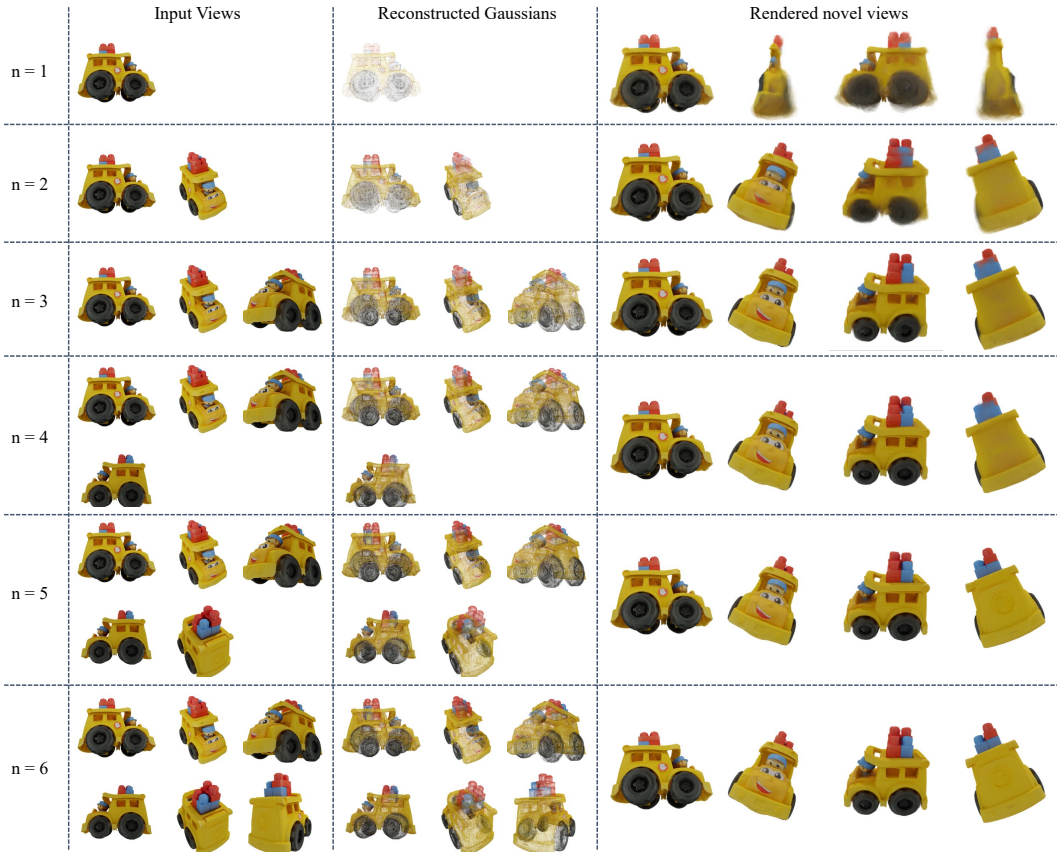

Figure 15: **Illustration on the influence of input view number.** We show the visual comparison of FreeSplatter-O results with varying numbers of input views ($n = 1 - 6$). From left to right: input views, reconstructed Gaussians, and rendered target views at 4 fixed viewpoints. Additional input views increase Gaussian density and improve previously uncovered regions, with diminishing returns beyond n=4 when object coverage becomes sufficient.

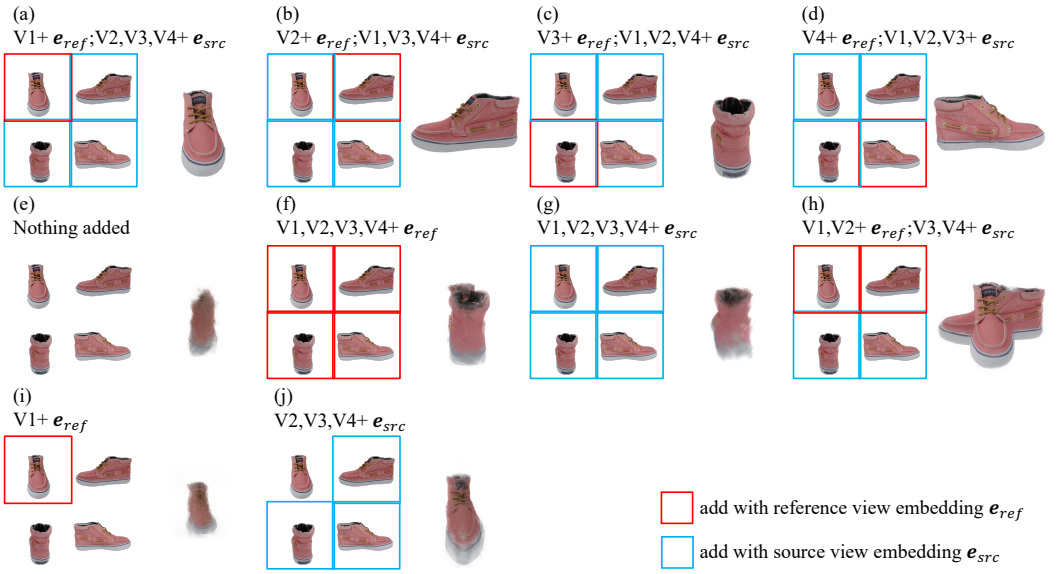

Figure 16: **Ablation study on view embedding addition.** Red/blue boxes indicate views added with reference/source view embeddings respectively. For each case, we visualize the image rendered with identity camera (*i.e., reference pose*) on the right.