A APPENDIX

In this section, we describe the followings:

- Detailed Discussion on Ray Guidance.
- Details of baseline implementation
- Details of model implementation.
- Additional Results.

A.1 DETAILED DISCUSSION ON RAY GUIDANCE

Most conventional methods in Multi-view Stereo (MVS) utilize cameras to establish geometrical relationships across the input views. However, the relationship becomes unreliable when given poses are noisy. We argue that it is important for the image features to have an awareness of camera poses to mitigate the influence of unreliable relationships during the 3D reconstruction. To this end, we combine predicted Plücker rays with image features to construct the cost volume, leveraging the advantages of using a generic camera representation. The intuition behind our design choice is to inject awareness of camera pose in multi-view space to each image feature.

Specifically, we project features from different viewpoints to compute correlation among input images by converting rays into camera poses and performing homography warping. While this allows some pose error, which leads to misalignment in feature space, we rectify the cost volumes by pose-aware cost aggregation process described in Section 4.2 of the main paper. As shown in Figure 9, eliminating pose embedding leads to large discrepancies in geometry estimation, leading to blurry images or introducing artifacts. This highlights the importance of pose embedding in our fusion process.

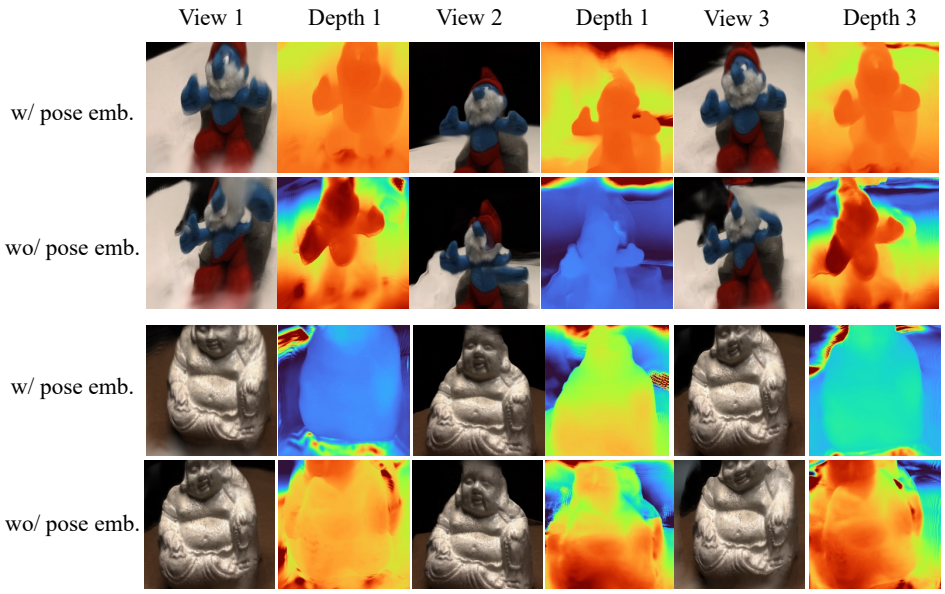

Figure 9: **Additional Qualitative Ablation Results on Pose Embedding.** Estimating geometry without pose embedding results in significant failures, producing blurry artifacts and misaligned structures in the 3D reconstruction. With pose embeddings, SHARE demonstrates the importance of geometric bias, achieving more accurate and sharper reconstructions. This highlights the effectiveness of pose-aware fusion in handling pose errors during the multi-view reconstruction process.

A.2 DETAILS OF BASELINE IMPLEMENTATION

For the small-scale DTU dataset (Jensen et al., 2014), we compared and validated our method against the pose-free baseline LEAP (Jiang et al., 2023). The LEAP model was trained on the DTU 3-

Table 4: **Comparison on baselines with different pose prediction methods on DTU dataset.**

Method	Pose	Rot. ↓	Trans. ↓	PSNR ↑	SSIM ↑	LPIPS ↓
PixelSplat	GT	–	–	20.96	0.65	0.31
	COLMAP	7.10	31.62	13.49	0.34	0.66
	MASt3R	2.40	**3.52**	15.69	0.40	0.50
	DUSt3R	**1.77**	13.66	15.98	0.42	0.47
	Ours	2.74	6.28	13.29	0.31	0.66
MVSplat	GT	–	–	21.00	0.69	0.24
	COLMAP	7.10	31.62	14.69	0.44	0.46
	MASt3R	2.40	**3.52**	13.31	0.31	0.58
	DUSt3R	**1.77**	13.66	13.22	0.32	0.58
	Ours	2.74	6.28	14.08	0.33	0.51
Ours	–	2.74	6.28	**19.94**	**0.63**	**0.28**

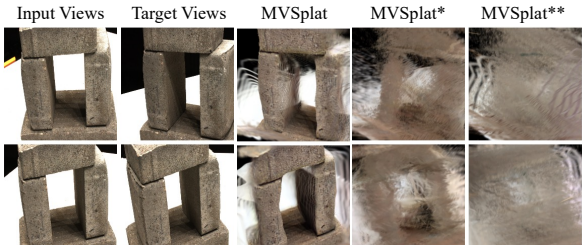

Figure 10: **Rendering of MVSplat Trained with Predicted and Noisy Poses.** The row labeled MVSplat shows the results of training with ground-truth poses, while MVSplat* and MVSplat** refer to the MVSplat model trained with predicted poses from DUSt3R and noisy poses with minor errors, respectively.

Table 5: **Quantitative results of pose estimation performance.** We evaluate the pose estimation performance on DTU dataset with small baselines, given three input views. The lowest error is marked as bold.

Method	Rot. ↓	Trans. ↓
DUSt3R	1.77	13.66
MASt3R	2.40	3.52
COLMAP	7.10	31.62
Relpose++	19.56	44.18
RayRegression	3.10	6.57
Ours	**2.74**	**6.28**

view dataset for 140K iterations. Since our evaluation on DTU uses three input views, we also trained pose-dependent state-of-the-art generalizable 3D reconstruction methods, including PixelSplat (Charatan et al., 2024) and MVSplat (Chen et al., 2024b), with a batch size of 1 for 140K iterations.

For the large-scale RealEstate10K dataset (Zhou et al., 2018), we compared our method against pose-free baselines CoPoNeRF (Hong et al., 2024) and FlowCam (Smith et al., 2023). Since CoPoNeRF and FlowCam use the same train-test split as our method, we directly compared our results with the reported values. Additionally, PixelSplat and MVSplat were evaluated using their pretrained checkpoints on the same 2-view train-test split settings.

We evaluated pose-dependent baselines under two conditions: using predicted poses and poses perturbed by random noise. For predicted poses, we used one of the state-of-the-art pose estimator, DUSt3R (Wang et al., 2024), to estimate poses from the input images. To ensure fair comparisons, we also evaluated the baselines with various pose estimators, including COLMAP (Schonberger & Frahm, 2016), DUSt3R (Wang et al., 2024), MASt3R (Leroy et al., 2024) and SHARE. For DUSt3R and MASt3R, we utilized pre-trained model weights provided in their official GitHub repositories. As shown in Table 4, our method consistently outperformed these combinations. Furthermore, the results for noisy poses (Table 1 and Table 2) highlight that even minor errors—currently unavoidable by state-of-the-art pose estimators—can introduce significant instability in reconstruction quality.

We trained the baseline models using ground-truth (GT) poses, as training with noisy poses lacking specific noise patterns often resulted in instability, divergence, or failure to converge. Figure 10 illustrates a comparison of MVSplat models trained on DTU with GT poses versus those trained with predicted poses from DUSt3R (Wang et al., 2024) and slightly perturbed poses ($\sigma = 0.01$, rotation error 0.95°, translation error 1.05°). These findings demonstrate that even small amounts of noise during training can destabilize models by introducing subtle misalignments between views, leading to a decline in reconstruction quality.

A.3 Details of model implementation

In this section, we'll discuss our framework in more detail. Given sparse-view unposed images, our goal is to build comprehensive Gaussians in a canonical space. The output of the multi-view feature extractor is $V \times C \times H \times W$, where we set C as 128 in all experiments. Given these features, we estimate the relative Plücker rays $V \times 6 \times H \times W$ with two additional transformer blocks following the U-Net structure of (Zhang et al., 2024). Then, we embed ray with a lightweight MLP to latent space and modulate multi-view features using AdaLN (Peebles & Xie, 2023), following LaRa (Chen et al., 2024a). In the ray-guided multi-view fusion process, we first build the cost volumes from all input views, where the depth candidates D are all set to 128. We warp all the features to the reference views with the estimated pose (converted from Plücker rays). Then, we build the geometry volume V_g as in 3. The geometry volume is used to estimate the anchor points $3 \times \frac{H}{4} \times \frac{W}{4}$. Simultaneously, we build the feature volume V_f in a similar manner, but with the upscaled multi-view features, to estimate the offset vectors and Gaussian parameters necessary for finer detail reconstruction.

We divide channels of $\mathbf{V}_f$ for displacement prediction of anchor points (32), and the remaining channels (96) encode texture-related Gaussian parameters. The geometry channels of $\mathbf{V}_f$ are passed through the offset prediction MLP head f_o, which predicts the offset vectors $\Delta\mathbf{p}_k = f_o(\mathbf{V}_f)$, for the Gaussian positions. We set $K = 3$ for all experiments. These offset vectors are then concatenated with the remaining channels of $\mathbf{V}_f$ Another MLP head, f_p, processes the concatenated features to estimate the remaining Gaussian parameters.

A.4 Additional Results

Results of pose estimation We evaluated our pose estimation performance in terms of rotation error (degrees) and translation error (degrees), as detailed in the main paper. Comparisons were made against state-of-the-art pose estimators, including DUSt3R (Wang et al., 2024), MASt3R (Leroy et al., 2024), and RayRegression from Cameras-as-Rays (Zhang et al., 2024). Additionally, we compared our method with COLMAP (Schonberger & Frahm, 2016) for primitive pose estimation and RelPose++ (Lin et al., 2024) as a direct 6D pose estimator. The evaluation used three small-baseline views from the DTU (Jensen et al., 2014) dataset as input images.

While our primary objective is high-quality novel view synthesis rather than pose estimation, our method achieves pose estimation performance comparable to state-of-the-art methods, further demonstrating its robustness and versatility.

Cross-dataset generalization Table 6 and Figure 11 present the cross-dataset generalization results, comparing our proposed method, SHARE, with baseline approaches. Models trained on the RealEstate10K (Zhou et al., 2018) dataset were evaluated on the ACID (Liu et al., 2021) dataset, while those trained on the DTU (Jensen et al., 2014) dataset were tested on BlendedMVS (Yao et al., 2020). The ACID dataset comprises natural large-scaled scenes captured using aerial drones, divided into 11,075 scenes for training and 1,972 scenes for testing, with accompanying camera extrinsic and intrinsic parameters. The BlendedMVS dataset consists of 3D models of diverse scenes, including outdoor and indoor environments. In our experiments, we utilize a subset of BlendedMVS as a cross-dataset evaluation benchmark to assess the generalization ability of our method.

Notably, under the challenging conditions of pose error $\sigma = 0.01$, which remains difficult even for state-of-the-art pose estimators, SHARE consistently outperforms all baseline methods across all metrics. These findings underscore the robustness of SHARE, particularly in realistic scenarios where pose estimation inaccuracies are inevitable.

Comparision with the Concurrent Work. We compare SHARE with our concurrent work, Splatt3R (Smart et al., 2024) which utilizes pretrained MASt3R (Leroy et al., 2024) weights for geometry estimation. Since Splatt3R requires ground-truth dense depths map during training, it is not directly applicable to our used datasets (RealEstate10K (Zhou et al., 2018) doesn't contain gt depths, and DTU (Jensen et al., 2014) contains masked depths, which we found that it is not directly applicable without method modifications because of Splatt3R's pixel-aligned dense prediction mechanism). Instead, we directly compare with the pretrained Splatt3R model trained on ScanNet++ (Yeshwanth et al., 2023). We note that Splatt3R employs a "masking loss" (refer to Section 3.4 in their paper) to render only valid pixels for the target view based on input images. To avoid this issue, we measure PSNR and other metrics only for the valid pixels produced by Splatt3R (pixels with > 0 values).

Table 6: **Quantitative comparison of cross-dataset generalization.** The best-performing values across all metrics are highlighted in bold.

Method	Pose	RealEstate10K → ACID			DTU → BlendedMVS		
		PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
pixelSplat	GT	26.84	0.81	0.18	11.64	0.20	0.67
	$\sigma = 0.01$	21.73	0.57	0.28	11.65	0.20	0.68
MVSplat	GT	28.18	0.84	0.15	12.04	0.19	0.56
	$\sigma = 0.01$	21.65	0.57	0.27	11.92	0.20	0.59
Ours	-	**23.47**	**0.69**	**0.26**	**12.19**	**0.26**	**0.61**

Table 7: **Quantitative Comparison with Concurrent Work.** We compare our method with the concurrent work Splatt3R on the DTU and RealEstate10K datasets, using two input views for both datasets. Splatt3R results are obtained using pretrained weights trained on the ScanNet++ dataset, while our method is trained on each respective dataset. The best results are highlighted in bold.

Method	DTU (2-views)			RealEstate10K		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
Splatt3R	11.78	0.28	0.57	15.80	0.53	0.30
Ours	**17.50**	**0.34**	**0.48**	**21.23**	**0.71**	**0.26**

Including entire regions would lead to significant drops in PSNR and thus would not reflect the method's intended performance.

In Table 7 and Figure 12, we present comparisons both on the DTU and RealEstate10K datasets, where SHARE outperforms Splatt3R. To ensure fairness, as comparing Splatt3R trained on ScanNet++ with SHARE trained on each dataset may introduce biases, we conducted additional evaluations in a cross-dataset setting. Specifically, we compared Splatt3R trained on ScanNet++ and SHARE trained on RealEstate10K in the ACID (Liu et al., 2021) dataset. As illustrated in Table 8 and Figure 13, SHARE demonstrates superior rendering quality compared to Splatt3R. We measure metrics only for the valid pixels produced by Splatt3R (pixels with > 0 values). Including entire regions would lead to significant drops in PSNR and thus would not reflect the method's intended performance. Splatt3R exhibits scale ambiguity in its predicted scenes, which can lead to a substantial drop in performance when applied to datasets with unseen scale distributions.

Discussion on large baseline inputs We visualized large-baseline camera scenarios (Figure 14). We compare our method with PixelSplat (Charatan et al., 2024) and MVSplat (Chen et al., 2024b) using both our predicted poses and perturbed poses with Gaussian noise, which exhibit similar or lower pose errors compared to predicted poses.

Discussion on Efficiency. We evaluated and compared the inference time (in seconds) and GPU memory usage (in MB) of our method against baseline approaches on the RealEstate10K dataset, as detailed in Table 9. Inference time is measured as the end-to-end duration required for novel view synthesis using two unposed input images, while GPU memory usage includes both static and dynamic memory allocations during inference. Our method achieves superior efficiency in both inference time and GPU memory usage compared to the pose-free, generalizable NVS baseline CoPoNeRF (Hong et al., 2024) and the concurrent method Splatt3R. Furthermore, our approach

Table 8: **Quantitative Comparison with Concurrent Work: Cross-Dataset Generalization.** We evaluate and compare the cross-dataset generalization performance of our method and Splatt3R. The best results are highlighted in bold.

Method	Training data	ACID		
		PSNR ↑	SSIM ↑	LPIPS ↓
Splatt3R	ScanNet++	17.49	0.63	**0.26**
Ours	RealEstate10K	**23.47**	**0.69**	**0.26**

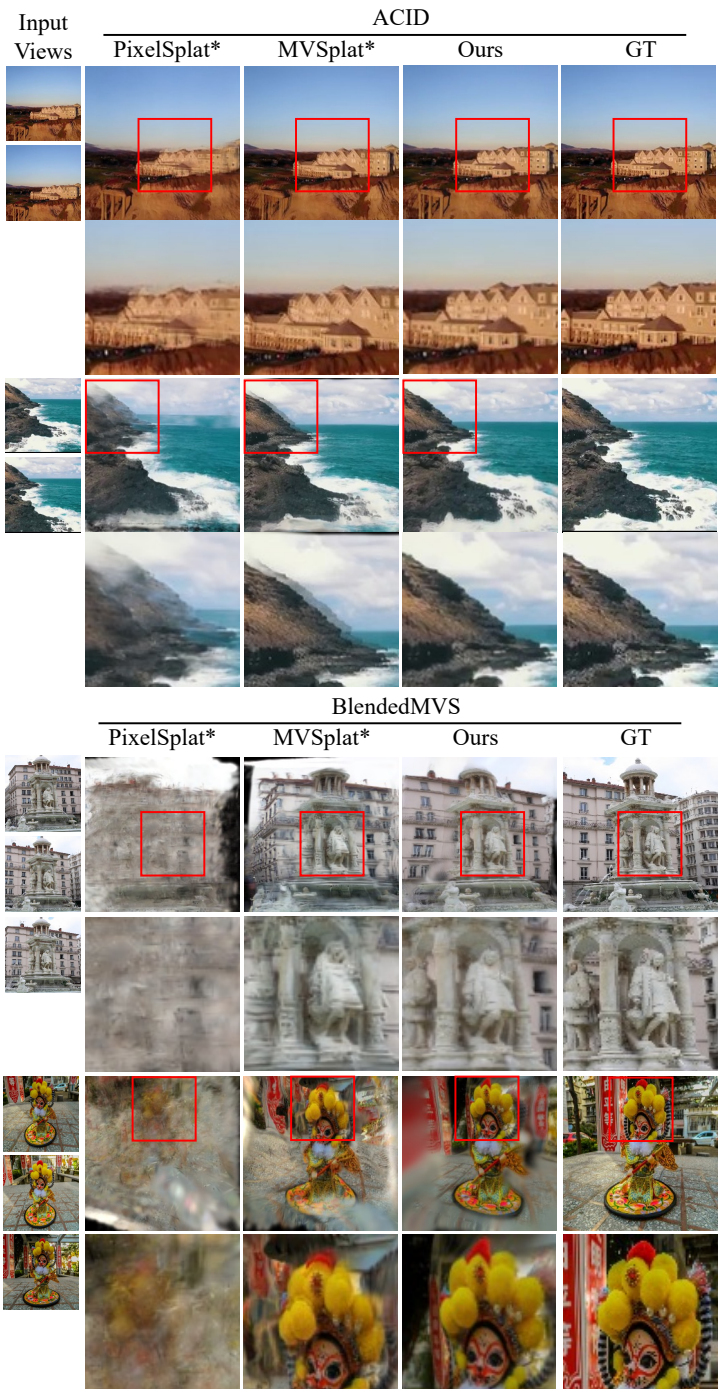

Figure 11: **Qualitative Results for Novel View Synthesis in Cross-Dataset Generalization.** PixelSplat* and MVSplat* denote methods combined with noisy camera settings ($\sigma = 0.01$). To aid visibility, we highlight the regions of interest with red boxes and provide close-up visualizations of these areas for detailed comparison.

delivers the highest rendering quality among the compared methods, underscoring its effectiveness. All experiments were conducted on an NVIDIA RTX 4080 GPU.

Qualitative results of novel view synthesis We present our additional results on the DTU (Jensen et al., 2014) dataset (Figure 15) and RealEstate10K (Zhou et al., 2018) dataset (Figure 16).

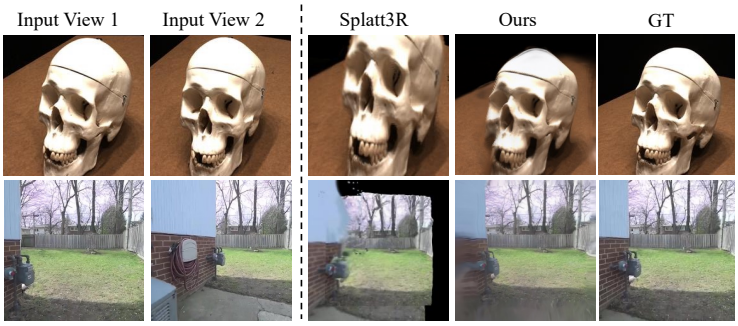

Figure 12: **Qualitative Comparision with the Concurrent Work.**

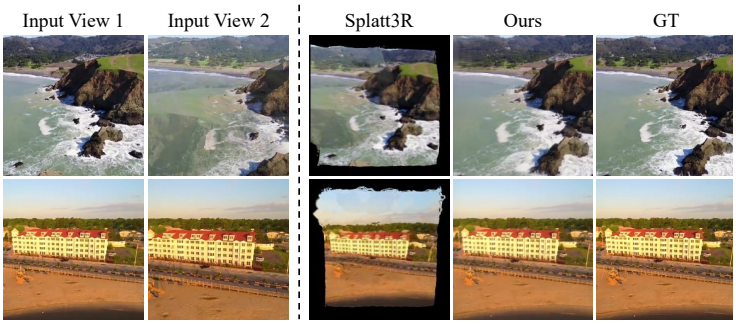

Figure 13: **Qualitative Comparision with the Concurrent Work: Cross-dataset Generalization.**

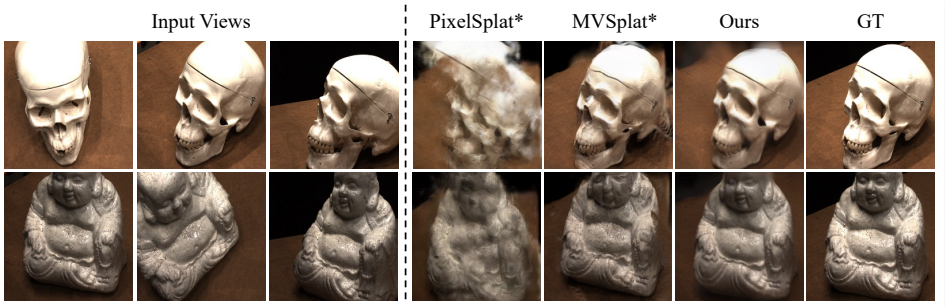

Figure 14: **Qualitative Results of Novel View Synthesis with Large-Baseline View Sets.** PixelSplat and MVSplat denote methods combined with a noisy camera setup, incorporating Gaussian noise with a standard deviation of 0.01.

Table 9: **Model Efficiency Measurements.** Each metric is evaluated across models using the same dataset configuration and averaged for consistency.

Method	Inference time (s)	GPU Memory (MB)
CoPoNeRF	3.37	9587.22
MVSplat + MASt3R	0.22	**4376.94**
Splatt3R	0.26	6198.00
Ours	**0.17**	5887.18

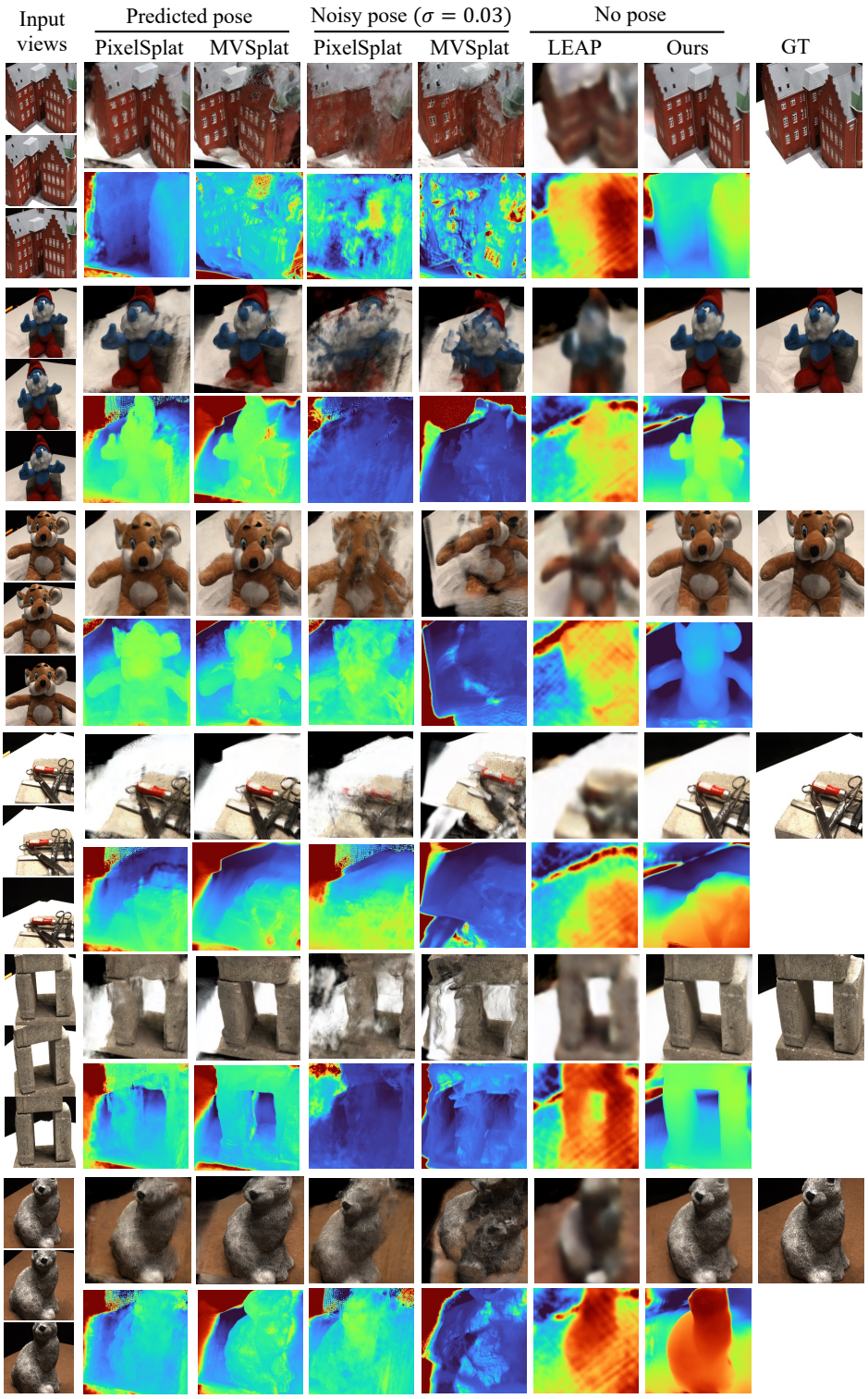

Figure 15: **Additional Qualitative Results on the DTU Dataset.** Rendered target images are shown based on three input views. The predicted pose indicates poses predicted using DUSt3R (Wang et al., 2024).

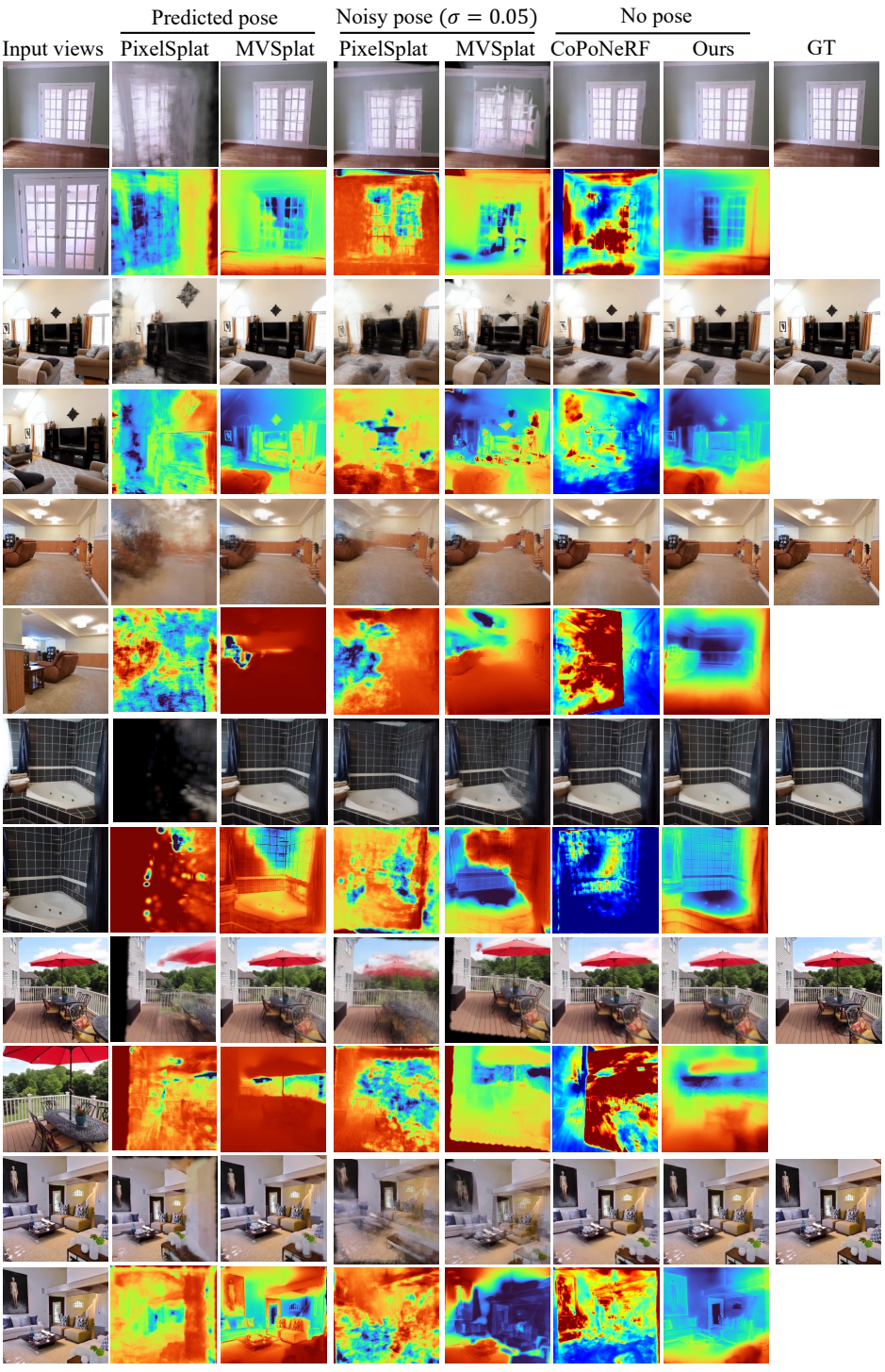

Figure 16: **Rendering and Depth comparison on RealEstate10K** The visualized images are rendered target images given 2 input views. The predicted pose indicates poses predicted using DUSt3R (Wang et al., 2024).