Selfi: Self-improving Reconstruction Engine via 3D Geometric Feature Alignment

Supplementary Material

In our supplementary material we provide additional details on training and evaluation strategies in Sec. 1. We follow with experiments on alternative loss objectives, more evaluations, and visualizations in Sec. 2, and a discussion on limitations in Sec. 3. We additionally include a webpage with animated video results and visualizations.

1. Supplementary Methods

1.1. Additional Implementation Details

Feature Correspondence Training. For pseudo-ground-truth supervision, we obtain 2D query points from a source frame, and use depth and camera parameters to project the query point onto corresponding target points in the target frames. During training, the query points are selected uniformly at random from the source frames, and we use a batch size of 4096 query points per frame. We select the middle frame in the sequence as the source frame and project all other frames as target frames. During inference, we select 2048 points per source frame, and we use every fifth frame in the sequence as source frames. We use a radius of five frames around each source frame as the target frames for feature matching. To filter out unreliable matches, we adopt three criteria to remove matches: (1) if the query point falls outside of the target field-of-view using predicted depth and relative camera pose, (2) if the query point is projected behind the target camera (with negative depth), and (3) if the confidence of either point is less than 1.2 using the VGGT confidence maps.

Gaussian Head Training. For Gaussian head training, we sample a random stride between 2 and 6 for DL3DV and between 5 and 15 for RealEstate10K, then take a sequence of 11 frames – 6 inputs and 5 targets in between them. In addition, as the view-dependent density term results in low-confidence Gaussians becoming transparent, we prune these Gaussians for rendering efficiency. For a given render camera, we sort the Gaussians by predicted opacity, and remove 30% of the Gaussians with lowest opacity. This reduces the number of primitives that need to be rasterized, thus improving memory efficiency during training.

1.2. Baseline Evaluations

Two-View RealEstate10K. We evaluate on the two-view split proposed by PixelSplat [1], which consists of three target images randomly sampled between two context images at 256x256 resolution. We adopt the metrics from ReSplat [9] for evaluation on methods that require posed inputs, and we similarly evaluate NoPoSplat [10], Flare [12], and our model on the same images. For NoPoSplat and Flare, we perform post-hoc camera optimization to maximize the similarity of the rendered image to the target image using the ground-truth cameras as initialization, following the official codebases for each method. We do not perform any per-scene optimization with our method.

Multi-view Evaluation. We use the official codebases and pretrained checkpoints for Anysplat [3] and WorldMirror [6]. We evaluate these baselines and our model at 294x518 resolution on the same images. We first provide inputs and targets together to obtain poses for all images, then we run the Gaussian U-Net only on the input images and render the input-aligned primitives to the target images. For Flare [12], we find that with the released checkpoint we obtain better performance using 256x256 resolution and three images rather than the full image set, so we evaluate using a sliding window of one target and the two adjacent input frames using PnP to produce the poses; both the reduced resolution and sliding evaluation are advantageous to the Flare baseline relative to our evaluation of our model and the remaining baseline methods.

Varying Sequence Length and Varying Stride. In our main paper Tab. 1, we investigate the impact of changing the number of inputs, whereas in main paper Tab. 2 ,we evaluate the impact of changing the overlap between views. We note that these are two separate challenges for our method. We control overlap by changing the sampling stride of the sequence. Decreasing the overlap between views (*i.e.,* increase the camera baseline) makes reasoning about 3D structure more difficult. Therefore reconstruction better for large overlap than small overlap. On the other hand, we separately hold the sampling stride fixed and change the number of inputs. As our Gaussian primitives are pixel-aligned, having more inputs becomes harder because the model must produce consistent geometry for pixels among all views to avoid duplication or occlusion artifacts. Thus, we find that reconstruction metrics are better for short sequences compared to longer ones.

2. Supplementary Experiments

2.1. Alternative Training Designs

Contrastive Training Objective. The goal of our feature alignment is to train the feature adapter such that features

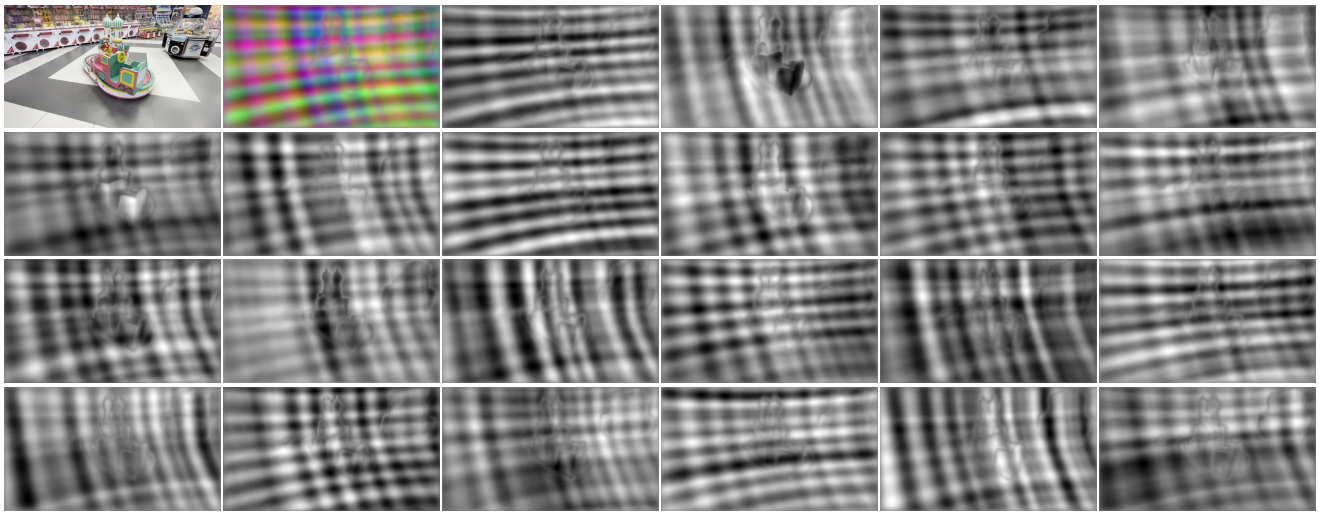


Figure 1. **Feature Visualization.** For the top-left image, we produce a 24-dimensional feature map to compute corresponding points. First, we average across three groups of eight channels to produce a color visualization. While we observe a structured spatial pattern in this image, visualizing the channels independently (showing 22 channels in gray) reveals that each channel captures a slightly different pattern.

Figure 2. **Contrastive Loss Experiment.** Using a CLIP-style contrastive training objective encourages features of correct 2D-2D matching points to be more similar than those of incorrect matches. In our experiments, we found that this objective simply resulted in the features converging to the same value, so that all queries match to the same target point. Thus we adopted the alternative strategy that encourages features of the correct match to be more similar than all other pixels in the target image.

corresponding to spatially proximal 3D locations exhibit high similarity. One straightforward approach is to follow a CLIP-style contrastive learning scheme [7], as used in [5]. Given a set of 2D-2D correspondences, the training objective encourages the correct match to have more similar features than the incorrect matches. However, in our experiments we find that this objective causes our features to collapse to the same value, as shown in Fig. 2. We hypothesize that this may be due to insufficient supervision, where the contrastive loss only supervises over matched points, whereas our final objective supervises over all pixels in the target frame.

Nearest Neighbor as Pseudo-Ground-Truth. For our pseudo-ground-truth we use the predicted depth from the source frame $\mathbf{D}_s$ and the relative camera pose from source to target frame $\mathbf{R}_{t\leftarrow s}, \mathbf{t}_{t\leftarrow s}$ to determine 2D-2D matches for the query pixels in the target frames. An alternative strategy is to compare the 3D points produced from source and target frames and assign correspondences when 2D points coincide in 3D. For this, we compute the 3D target-frame coordinates of the query point $\mathbf{P}_t^n$ following Eq. 5 in the main text,and also unproject all points in the target frame using $\mathbf{D}_t \pi_{\mathbf{K}}^{-1} \mathbf{p}_t$. We assign the correpondence in the target frame as the pixel that is closest in 3D space to $\mathbf{P}_t^n$. However, we observed that this pseudo-ground-truth is prone to noise, and produces matches that jitter as the target frame moves. A visualization of this effect is presented in the pseudo-GT videos on the last section of our project page. The pseudo-GT matching points derived from projection remain consistent even across long sequences, whereas points matched using K-Nearest Neighbors appear noisy. Even with a large K (*e.g.,* 10), significant jitter is observed. This pseudo-GT negatively impacts the training of our feature adapter.

Bundle Adjustment with Different Sampling Strides. In Tab. 5 of the main text, we evaluated our feature correspondences for improving poses via bundle adjustment on sequences of different lengths with a fixed stride of 2. Here, we additionally investigate changing the sampling stride between 2, 4, and 8, holding the sequence length fixed to 10 images. This effectively increases the baseline between adjacent images. We find that our correspondences consistently improve the pose estimation after bundle adjustment under these various settings in Tab. 2.

2.2. Additional Evaluations and Visualizations

Evaluation with the RayZer [2] Setting. We evaluate our model with the RayZer convention using their random sample split, consisting of 16 input images and 8 target images at 256x256 resolution. While RayZer uses a neural renderer, requiring a transformer decoder to be run for novel

Method	Training Supervision	Inference w. COLMAP Cam.	Random Sample PSNR↑	SSIM↑	LPIPS↓
GS-LRM [11]	2D + Camera	✓	23.02	0.705	0.266
LVSM [4]	2D + Camera	✓	23.10	0.703	0.257
RayZer [2]	2D	✗	23.72	0.733	0.222
RayZer* [2]	2D	✗	**25.47**	0.795	0.181
Ours	2D + VGGT [8]	✗	23.75	**0.850**	**0.129**

Table 1. **Quantitative Comparisons in RayZer [2] Setup**. We evaluate renderings from our model using the evaluation index proposed in RayZer, consisting of 16 inputs and 8 targets at 256×256 resolution. We attain competitive performance to RayZer, outperforming on SSIM and LPIPS, and our scene representation can be directed rasterized to novel views without the need for an additional network pass. Rayzer* is the external re-implementation.

Figure 3. **Qualitative Comparisons of Our Method and Rayzer [2]**. Compared to RayZer [2] (bottom row), our method (top row) preserves sharper high-frequency details and avoids patch artifacts in complex regions.

Stride	BA	AUC@3	AUC@5	AUC@15
2	✗	0.8130	0.8767	0.9550
	✓	0.8669	0.9140	0.9690
4	✗	0.8439	0.8973	0.9623
	✓	0.8672	0.9105	0.9657
8	✗	0.8443	0.9008	0.9648
	✓	0.8501	0.9049	0.9662

Table 2. **Bundle Adjustment with Different Strides**. We increasing the sampling stride of the input sequence, where larger stride corresponds to larger camera baseline. As the stride increases, the initial pose estimate from VGGT improves, but a subsequent bundle adjustment using our features offers further improvements.

views, our model produces a 3D representation that can be directly rendered to novel views with 3DGS. We achieve competitive results compared to RayZer, outperforming it on SSIM and LPIPS Tab. 1. Note that RayZer reports two versions of the implementation: internal and external. We report both. We also provide our qualitative comparisons in Fig. 3. Our method's lower PSNR in Tab. 1 may be due to its sensitivity to lighting fluctuations, which can cause a systematic color shift. Our approach preserves high-frequency textures and leads to higher SSIM and LPIPS, while RayZer produces patch artifacts, especially in the leafy regions of the second and third scenes. We also include a failure case in the last column—our model struggles with directly viewed light sources like the sun.

Visualization of Geometrically Aligned Features. We visualize the per-channel values of our trained features in Fig. 1. We observe that the channels capture both image structure and learned spatial encoding values. Although we observe a repeating pattern, when we plot individual channel values we find that the learned patten each channel is slightly different.

Visualization of Feature Matching Results. We plot

Figure 4. **Visualization of Matching on Re10K**. We similarly point query points (left image) and their matching pixels (right image) for the RealEstate10K dataset.

Feat Upsampling	Feat Processing	PSNR ↑	SSIM ↑	LPIPS ↓
DeConv	w/o Alignment	20.07	0.625	0.213
DPT	KNN	22.40	0.723	0.199
DPT	Geometric Alignment	**23.65**	**0.802**	**0.153**

Table 3. **Ablation of Feature Upsampling and Processing.** Our DPT-based geometric alignment is critical for high-fidelity reconstruction, significantly outperforming both deconvolution-based upsampling and KNN-based feature aggregation baselines.

our computed correspondences from the aligned features in Fig. 5 and Fig. 4. Our feature matching successfully handles large changes in camera viewpoint.

More Qualitative Comparisons. We provide additional qualitative comparisons of our model renderings compared to baselines AnySplat [3] and WorldMirror [6] in Fig. 6 and Fig. 7. We include additional video examples on our webpage and recommend that reviewers check them.

More Ablations on Feature Alignment. While VGGT is trained to produce accurate, 3D-consistent outputs (e.g., cameras and depth), its backbone features are not explicitly aligned for 3D consistency. We employ a feature adapter to surface the backbone's implicit 3D knowledge without interfering with other VGGT prediction heads, while simultaneously bridging the $14\times$ resolution gap between backbone tokens and pixel-level outputs. Our experiments demonstrate that these 3D-aligned features improve NVS quality and enable more efficient camera pose refinement compared to VGGT's CoTracker. We attribute these gains to supe-

rior initial 3D consistency when predicting input-aligned primitives and denser, pixel-level self-supervised training. In Tab. 3, we further evaluate a simpler deconvolution upsampler, which underperforms compared to DPT. Additionally, we report a baseline multi-view feature aggregation alternative using KNN to merge points in 3D space and average their features. This approach also underperforms relative to our aligned features, confirming the necessity of our proposed alignment strategy.

3. Limitations

While our experimental results are encouraging, there are limitations that remain. Firstly, because we rely on a pretrained VGGT backbone, We find that depth predictions in certain regions may not be accurate, for instance, in the sky and far-away regions. This is largely because the VGGT is trained with a normalized scale to eliminate ambiguity. As a result, a background with an extremely large depth difference from the foreground cannot be represented in a normalized scene field. While a view-dependent density term can mitigate this effect by reducing low-confidence Gaussians to be transparent when rendering to a novel view, the fundamental artifact stems from incorrect depth and pose estimates. In addition, we find that exposure differences between the input images and ground-truth targets can impact rendering metrics, as our produced rendering will mimic the exposure of the input images which may differ from that of the ground-truth target. Our model can exhibit failures on dynamic scenes as both the VGGT and our feature alignment head train exclusively on static scenes; we observe that dynamic regions may be incorrectly matched with the parts of the static background they occlude, and improving the featuring matching on dynamic scenes is a promising future direction to investigate.

References

[1] David Charatan, Sizhe Lester Li, Andrea Tagliasacchi, and Vincent Sitzmann. pixelsplat: 3d gaussian splats from image pairs for scalable generalizable 3d reconstruction. In *CVPR*, 2024. 1

[2] Hanwen Jiang, Hao Tan, Peng Wang, Haian Jin, Yue Zhao, Sai Bi, Kai Zhang, Fujun Luan, Kalyan Sunkavalli, Qixing Huang, et al. Rayzer: A self-supervised large view synthesis model. In *ICCV*, 2025. 2, 3

[3] Lihan Jiang, Yucheng Mao, Linning Xu, Tao Lu, Kerui Ren, Yichen Jin, Xudong Xu, Mulin Yu, Jiangmiao Pang, Feng Zhao, et al. Anysplat: Feed-forward 3d gaussian splatting from unconstrained views. *ACM TOG*, 2025. 1, 4, 7, 8

[4] Haian Jin, Hanwen Jiang, Hao Tan, Kai Zhang, Sai Bi, Tianyuan Zhang, Fujun Luan, Noah Snavely, and Zexiang Xu. Lvsm: A large view synthesis model with minimal 3d inductive bias. In *ICLR*, 2025. 3

[5] Vincent Leroy, Yohann Cabon, and Jerome Revaud. Grounding image matching in 3d with mast3r. In *ECCV*, 2024. 2

[6] Yifan Liu, Zhiyuan Min, Zhenwei Wang, Junta Wu, Tengfei Wang, Yixuan Yuan, Yawei Luo, and Chunchao Guo. Worldmirror: Universal 3d world reconstruction with any-prior prompting. *arXiv preprint arXiv:2510.10726*, 2025. 1, 4, 7, 8

[7] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *ICML*, 2021. 2

[8] Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. Vggt: Visual geometry grounded transformer. In *CVPR*, 2025. 3

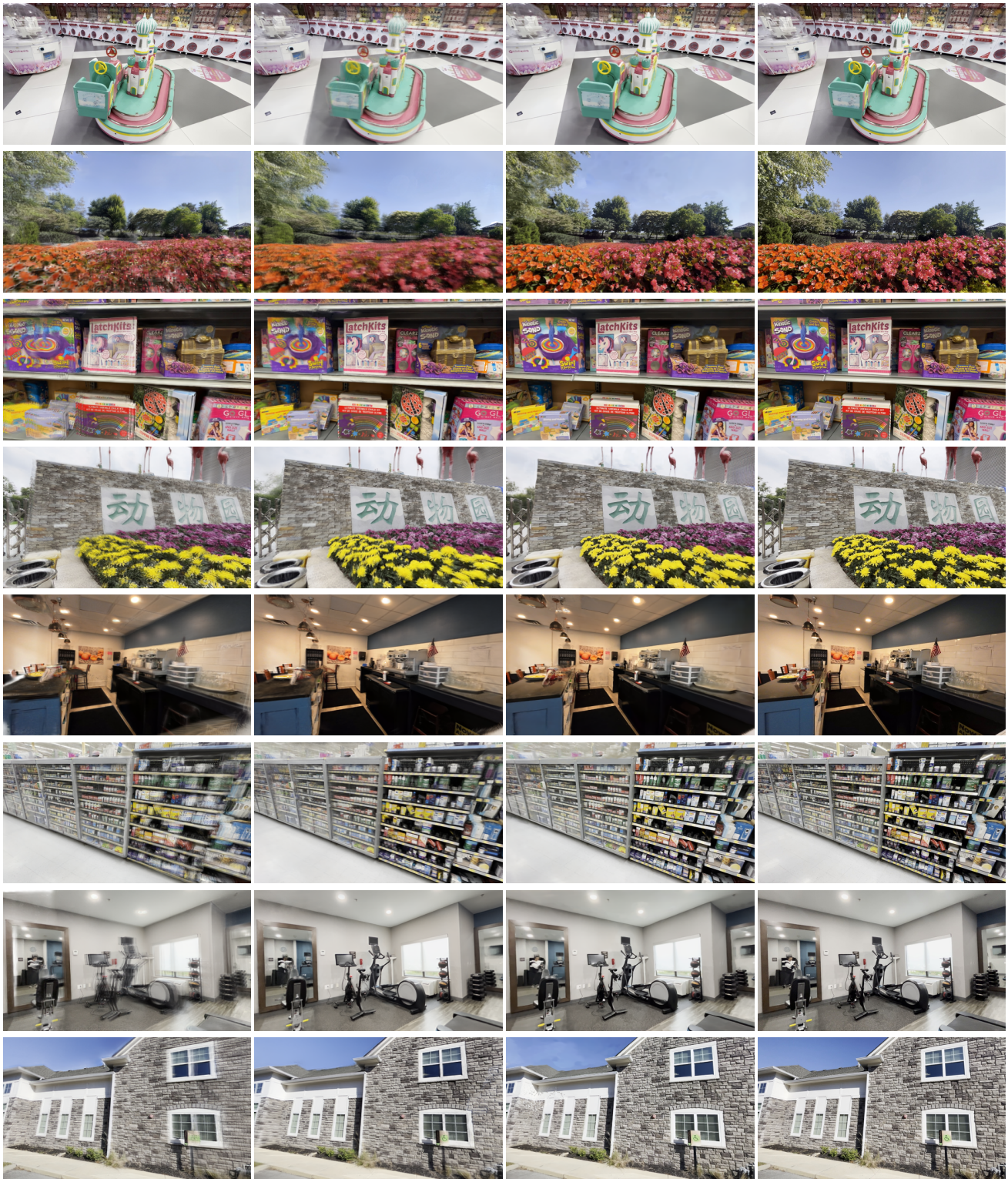
[9] Haofei Xu, Daniel Barath, Andreas Geiger, and Marc Pollefeys. Resplat: Learning recurrent gaussian splats. *arXiv preprint arXiv:2510.08575*, 2025. 1

[10] Botao Ye, Sifei Liu, Haofei Xu, Xueting Li, Marc Pollefeys, Ming-Hsuan Yang, and Songyou Peng. No pose, no problem: Surprisingly simple 3d gaussian splats from sparse unposed images. In *ICLR*, 2025. 1

[11] Kai Zhang, Sai Bi, Hao Tan, Yuanbo Xiangli, Nanxuan Zhao, Kalyan Sunkavalli, and Zexiang Xu. Gs-lrm: Large reconstruction model for 3d gaussian splatting. In *ECCV*, 2024. 3

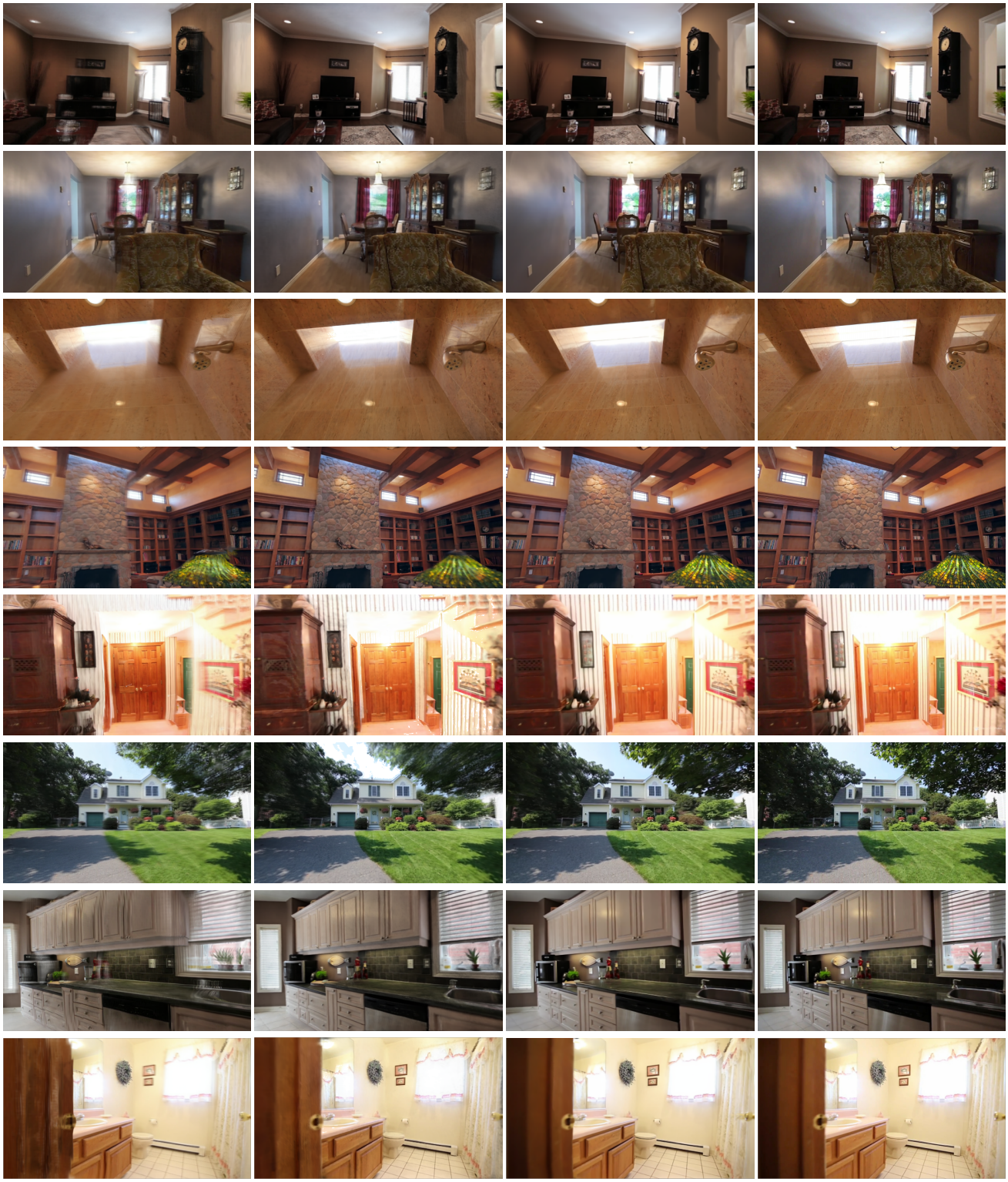
[12] Shangzhan Zhang, Jianyuan Wang, Yinghao Xu, Nan Xue, Christian Rupprecht, Xiaowei Zhou, Yujun Shen, and Gordon Wetzstein. Flare: Feed-forward geometry, appearance and camera estimation from uncalibrated sparse views. In *CVPR*, 2025. 1

Figure 5. **Visualization of Matching on DL3DV**. Given query points in the left image, we visualize the matched pixels in the right image. Our matching is robust to large changes in camera viewpoint.



(a) AnySplat (b) WorldMirror (c) **Ours** (d) GT

Figure 6. **Qualitative Comparisons on DL3DV**. We visualize additional examples of renderings from AnySplat [3], WorldMirror [6], and our method on DL3DV.



(a) AnySplat (b) WorldMirror (c) **Ours** (d) GT

Figure 7. **Qualitative Comparisons on RE10K**. We visualize additional examples of renderings from AnySplat [3], WorldMirror [6], and our method on RealEstate10K.