Dense-SfM: Structure from Motion with Dense Consistent Matching

Supplementary Material

1. Experiment

1.1. Sparse feature matching with our refinement

Sparse Det. & Matcher	Refinement	Accuracy (%)			Completeness (%)		
		1cm	2cm	5cm	1cm	2cm	5cm
SIFT + NN	Raw	66.4	78.62	89.38	0.12	0.60	3.02
	PixSfM	76.81	86.72	94.28	**0.15**	**0.66**	**3.12**
	DFSfM	**82.66**	90.41	**96.3**	0.14	0.59	2.78
	Ours	82.37	**90.42**	**96.3**	**0.15**	0.63	3.01
R2D2 + NN	Raw	59.7	73.92	87.25	0.34	1.4	6.42
	PixSfM	76.31	85.62	93.37	0.42	1.52	6.43
	DFSfM	81.51	89.81	95.81	**0.47**	**1.63**	**6.57**
	Ours	**83.1**	**90.5**	**96.03**	0.45	1.58	6.54
SP + SG	Raw	67.04	78.85	89.42	0.31	1.35	6.82
	PixSfM	79.98	87.84	94.52	**0.49**	**1.85**	**8.31**
	DFSfM	81.4	89.23	95.74	0.37	1.42	6.69
	Ours	**83.37**	**90.74**	**96.58**	0.38	1.49	7.01

Table 1. **Comparison of sparse local features accompanied with our refinement and PixSfM, DetectorFree-SfM (DFSfM)**. Our method is compared with the baselines on the ETH3D dataset using accuracy and completeness metrics with different thresholds.

Our track refinement module (multi-view kernelized matching described in Sec.3.3) can also be applied to refine SfM models reconstructed using sparse detection and matching to further improve the robustness of SfM. We evaluate each SfM result on the ETH3D dataset, assessing accuracy and completeness following the ETH3D benchmark [9].

As shown in Tab. 1, our refinement module consistently outperforms PixSfM and DFSfM in accuracy when accompanied by the same sparse detectors and matchers. Furthermore, our refinement module does not compromise the completeness much for higher accuracy.

1.2. Semi-dense matching with our pipeline

Matcher	Matching Preprocess	Refinement	Accuracy (%)			Completeness (%)		
			1cm	2cm	5cm	1cm	2cm	5cm
LoFTR	Quantization	DFSfM	80.38	89.01	95.83	3.73	11.07	29.54
	Quanzization	Ours	82.93	90.74	96.56	4.28	12.49	32.09
	Ours(GS-Extension)	Ours	**85.47**	**92.63**	**97.44**	**6.05**	**15.01**	**33.45**

Table 2. **Comparison of LoFTR(semi-dense matching) accompanied with matching process for track consistency and track refinement.** Our method is compared with the baselines on the ETH3D dataset using accuracy and completeness metrics with different thresholds.

We also apply our pipeline with semi-dense matcher, LoFTR, by replacing dense matchers (*e.g*. DKM or RoMa) with LoFTR for building an initial SfM (without mutual verification). Specifically, to address pairwise matching's inconsistency, we compare our proposed Gaussian Splatting (GS) based track extension with quantization, and also evaluate our refinement module against DFSfM's refinement module.

As shown in Tab. 2, our GS based track extension not only provides superior accuracy but also improves completeness compared to quantization, highlighting its effectiveness. Furthermore, our refinement module consistently outperforms the one proposed in DFSfM, demonstrating the overall superiority of our pipeline when paired with a semi-dense matcher.

1.3. Implementation Details

As in PixSfM and DFSfM, we obtain match results from exhausitive pairs within a set of images. To obtain dense matching from image pairs, we resize images to 1162×768 for the ETH3D dataset and IMC 2021 dataset, while maintaining the original size of 840×840 for the Texture-Poor SfM dataset.

Since dense matching methods (*e.g*. DKM and RoMa) are computationally heavier than detector-based or semi-dense approaches due to their pixel-wise matching process, we improve efficiency by obtaining coarse matches at a lower resolution (560×560 for the ETH3D and IMC 2021, 420×420 for the Texture-Poor SfM dataset) in the global matching stage of RoMa, which dominates the runtime. We then refine the matches to the target resolution. Under this configuration, on the ETH3D dataset, RoMa achieves a runtime of 0.08 seconds per image pair on a single A6000 GPU. In comparison, DFSfM with LoFTR operates at a higher resolution (1600×1200) and requires 0.2 seconds per pair. Our matching pipeline, which includes bidirectional matching with verification, runs in 0.17 seconds per pair, outperforming the LoFTR-based pipeline while maintaining a comparable matching speed.

1.4. More Ablation Study

In this section, we validate the effectiveness of our refinement pipeline by evaluating its performance on the reconstruction task, which involves recovering both camera poses and 3D point cloud. For evaluating point cloud accuracy, we use the *Pipes* scene from the ETH3D dataset, as the scene provides ground-truth point clouds. We align reconstructed poses with the ground truth poses through COLMAP [8], aligning 3D point as well, and evaluate the accuracy of point cloud following the ETH3D benchmark [9].

As shown in Tab. 4, our pipeline improves the accuracy

Matching Design	Scene Name													Avg.
	courtyard	delivery area	electro	facade	kicker	meadow	office	pipes	playground	relief	relief2	terrace	terrains	
	Average Track Length													
Raw	2.19	2.06	2.12	2.41	2.09	2.08	2.07	2.03	2.04	2.08	2.05	2.12	2.08	2.11
Quantized Matching (r=4)	3.43	3.11	3.37	4.82	2.90	2.86	2.91	2.89	3.05	3.21	3.13	3.19	3.14	3.23
Track Extension via GS	5.12	4.59	5.11	9.48	4.18	3.58	4.23	3.97	4.25	5.75	5.15	4.61	4.64	**4.97**

Table 3. Comparison of average track length accompanied with matching strategies (Quantization, Track Extension via GS) to obtain consistent tracks.

Refinement	Camera Pose Estimation			Accu.(%) (*Pipes*)		
	AUC@1°	AUC@3°	AUC@5°	1cm	2cm	5cm
No Refine	39.20	66.51	74.12	55.02	78.91	93.90
Iter 1	60.83	78.26	82.54	66.35	85.71	95.95
Track Extension via GS + Iter 2	**60.92**	**78.41**	**82.63**	**74.75**	**92.06**	**97.98**

Table 4. **Ablation Study of Refinement Iterations.** On the ETH3D dataset, we quantitatively evaluate the impact of the number of refinement iterations. The AUC of pose error and accuracy of 3D points at different thresholds are reported.

of both pose and point cloud through refinement. Note that the point cloud of *No Refine* is triangulated from two-view non-quantized matching results (resulting in most tracks having a length of 2), based on coarse poses obtained from quantized matching as DFSfM. Thanks to track extension via Gaussian Splatting, we can feed more views into our multi-view kernelized refinement module, resulting in a slight improvement on camera pose accuracy, and a significant improvement on point cloud accuracy.

1.5. Runtime Comparison

We evaluate the runtime of our multi-view kernelized matching module against the multi-view matcher module in DFSfM[2] on the *Pipes* scene, using the same SfM model for refinement. Our proposed refinement module costs a runtime increase about 22%, from 35.3s to 43.2s on a single A6000 GPU, due to the added computational cost of the Gaussian Process and CNN Decoder.

2. Details of Track extension via Gaussian Splatting

We present here the detailed implementation of track extension via Gaussian Splatting.

2.1. Training Gaussians

Initialization. As described in Sec.3.2 in the main paper, 3D Gaussians are initialized based on the initial SfM point cloud. We set the initial Gaussians' position to the 3D point coordinates, opacity to 1, and rotation to the identity matrix. The scale parameter S is determined using the method in Splatem [3], where each Gaussian's radius is set such that it is projected as an one-pixel radius circle on the 2D image. Specifically, we compute depth values for each 3D point based on the camera poses of its keypoints. Then we select the maximum value across the depth values and divide it by the focal length as follows:

$$S = \frac{D_{max}}{f} \tag{1}$$

where D_{max} represents the maximum depth value for the 3D point, and f is the focal length.

Training details. Since the images of the dataset we used for evaluation consist of sparse views on a scene, we use training setup with Few-Shot Gaussian Splatting (FSGS) [12] where it uses Proximity-guided Gaussian Unpooling [12] for densifying Gaussians. For training loss, we only use photometric loss between rendered image and original image (excluding depth regularizer), because the estimated depth from a monocular depth estimation network can differ from the semi-dense depth provided by the initial SfM, potentially interfering with the visibility checks for the initialized Gaussians (SfM's 3D points).

All images used in the SfM reconstruction are leveraged for training 3D Gaussians. Gaussians are densified every 500 iterations from the 1,500 iterations to 5000 iterations, with total optimization steps set to 6,000. The training time requires approximately 5 minutes on *Pipes* scene in the ETH3D dataset on a single RTX 4090 GPU. Other training parameters follow those outlined in FSGS [12].

Unlike the standard training of GS [4] that requires over 10,000 iterations for high-quality rendering, our process focuses only on verifying the visibility of the initial SfM's 3D points. Thus, we do not need as many iterations as the general training for high quality image rendering [4, 12].

2.2. Track length analysis

Tab. 3 compared average track length. The results show that our method significantly increases track length, which enables our refinement module to leverage more views, contributing to the higher accuracy reported in Tab.3 (1) of the main paper.

3. Applying Gaussian Splatting

To demonstrate the utility of our framework, we use it to initialize 3D Gaussians for novel view synthesis in a few-shot setting. Starting from SfM points, we apply

SfM Initialization	Metrics			
	L1↓	PSNR↑	SSIM↑	LPIPS↓
SIFT	0.0784	18.41	0.612	0.377
LoFTR+DFSfM	0.0750	18.63	0.633	0.358
RoMa+Ours	**0.0697**	**19.27**	**0.660**	**0.332**

Table 5. **Quantitative Comparison in the LLFF Dataset, with 3 Training Views.** Initialization with our method achieves the best performance in terms of rendering accuracy on all metrics.

Few-Shot Gaussian Splatting (FSGS) [12] on the LLFF dataset[7]. We compare the performance of 3D Gaussian models initialized using SIFT [6] with COLMAP, DetectorFree-SfM (DFSfM) with LoFTR, and our framework with RoMa, using fixed camera poses and intrinsics provided by FSGS [12] pipeline. For Gaussian initialization and training, we use only three views and render the original image size provided in the dataset.

For evaluation, test images are selected from the same scene in the LLFF dataset, excluding training images used for initialization and training. As shown in Tab. 5, our method demonstrates significantly superior performance, thanks to dense and accurate initialized point cloud generated by our framework.

4. Limitations and Future works

A key limitation of our framework is that track extension via Gaussian Splatting struggles in scenes with high photometric variation or transient occlusions. As future work, our pipeline can be extended with more advanced Gaussian Splatting models [1, 5, 10, 11], to better handle occlusions and appearance changes, allowing our framework to perform robustly in a wider range of scenarios.

References

[1] Hiba Dahmani, Moussab Bennehar, Nathan Piasco, Luis Roldao, and Dzmitry Tsishkou. Swag: Splatting in the wild images with appearance-conditioned gaussians, 2024. 3

[2] Xingyi He, Jiaming Sun, Yifan Wang, Sida Peng, Qixing Huang, Hujun Bao, and Xiaowei Zhou. Detector-free structure from motion. *CVPR*, 2024. 2

[3] Nikhil Keetha, Jay Karhade, Krishna Murthy Jatavallabhula, Gengshan Yang, Sebastian Scherer, Deva Ramanan, and Jonathon Luiten. Splatam: Splat, track & map 3d gaussians for dense rgb-d slam. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024. 2

[4] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42 (4), 2023. 2

[5] Jonas Kulhanek, Songyou Peng, Zuzana Kukelova, Marc Pollefeys, and Torsten Sattler. WildGaussians: 3D gaussian splatting in the wild. *arXiv*, 2024. 3

[6] David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60:91–110, 2004. 3

[7] Ben Mildenhall, Pratul P. Srinivasan, Rodrigo Ortiz-Cayon, Nima Khademi Kalantari, Ravi Ramamoorthi, Ren Ng, and Abhishek Kar. Local light field fusion: Practical view synthesis with prescriptive sampling guidelines. *ACM Transactions on Graphics (TOG)*, 2019. 3

[8] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 1

[9] Thomas Schöps, Johannes L. Schönberger, Silvano Galliani, Torsten Sattler, Konrad Schindler, Marc Pollefeys, and Andreas Geiger. A multi-view stereo benchmark with high-resolution images and multi-camera videos. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2538–2547, 2017. 1

[10] Yuze Wang, Junyi Wang, and Yue Qi. We-gs: An in-the-wild efficient 3d gaussian representation for unconstrained photo collections, 2024. 3

[11] Dongbin Zhang, Chuming Wang, Weitao Wang, Peihao Li, Minghan Qin, and Haoqian Wang. Gaussian in the wild: 3d gaussian splatting for unconstrained image collections, 2024. 3

[12] Zehao Zhu, Zhiwen Fan, Yifan Jiang, and Zhangyang Wang. Fsgs: Real-time few-shot view synthesis using gaussian splatting, 2024. 2, 3