Detector-Free Structure from Motion

Xingyi He[1], Jiaming Sun[2] Yifan Wang[1] Sida Peng[1]
Qixing Huang[3] Hujun Bao [1] Xiaowei Zhou[1†]

[1]Zhejiang University [2]Image Derivative Inc. [3]The University of Texas at Austin

Abstract

We propose a new structure-from-motion framework to recover accurate camera poses and point clouds from unordered images. Traditional SfM systems typically rely on the successful detection of repeatable keypoints across multiple views as the first step, which is difficult for texture-poor scenes, and poor keypoint detection may break down the whole SfM system. We propose a new detector-free SfM framework to draw benefits from the recent success of detector-free matchers to avoid the early determination of keypoints, while solving the multi-view inconsistency issue of detector-free matchers. Specifically, our framework first reconstructs a coarse SfM model from quantized detector-free matches. Then, it refines the model by a novel iterative refinement pipeline, which iterates between an attention-based multi-view matching module to refine feature tracks and a geometry refinement module to improve the reconstruction accuracy. Experiments demonstrate that the proposed framework outperforms existing detector-based SfM systems on common benchmark datasets. We also collect a texture-poor SfM dataset to demonstrate the capability of our framework to reconstruct texture-poor scenes. Based on this framework, we take the ***first place*** *in Image Matching Challenge 2023 [11]. Project page:* https://zju3dv.github.io/DetectorFreeSfM/.

1. Introduction

Structure-from-Motion (SfM) is a fundamental task in computer vision, which aims to recover camera poses, intrinsic parameters, and point clouds from multi-view images of a scene. The estimated camera poses and optional point clouds benefit downstream tasks, such as visual localization, multi-view stereo, and novel view synthesis.

SfM has been studied for decades, with many well-established methods [4, 12, 62, 13], open-source systems such as Bundler [47] and COLMAP [44], and commercial software [3, 2] that are accurate and scalable to large-scale scenes. As a routine, they require to detect and match sparse feature points across multiple views [31, 14, 41] at the beginning of the pipeline to build multi-view point-to-point correspondences. This requirement could not be fulfilled in many cases. For example, in texture-poor regions, it is hard to robustly detect repeatable keypoints across multiple views for matching. The poor feature detection and matching become the bottleneck of the whole SfM pipeline, which leads to missing image registration or even failed reconstruction

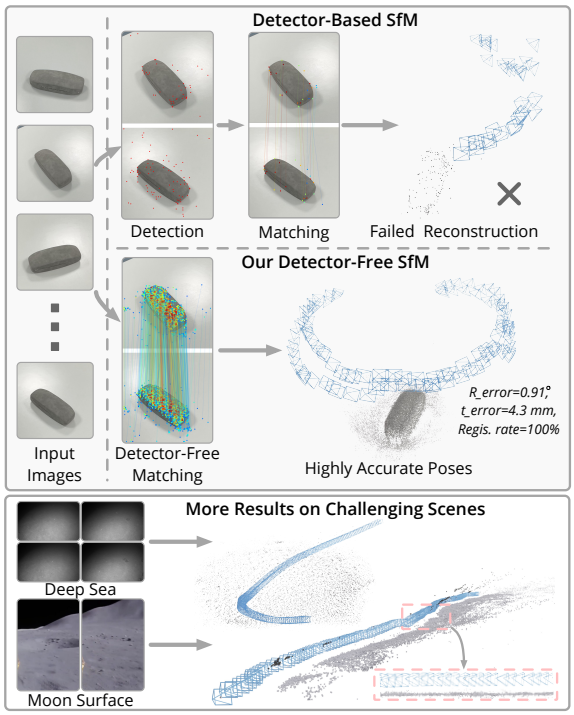

Figure 1. **Comparison between traditional detector-based SfM and the proposed detector-free SfM.** For the texture-poor scene, detector-based SfM fails due to the poor repeatability of detected keypoints at the beginning, while our detector-free SfM framework leverages detector-free matching and achieves complete reconstruction with highly accurate camera poses. Our framework is applicable to real-world challenging scenes such as the deep sea and the moon surface.

The authors from Zhejiang University are affiliated with the State Key Lab of CAD&CG. †Corresponding author: Xiaowei Zhou.

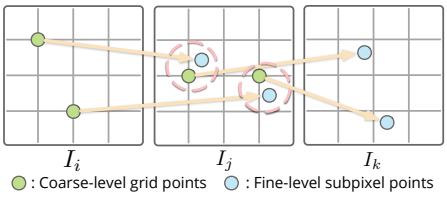

Figure 2. **Multi-view Inconsistency Issue of Detector-Free Matching.** The resulting feature locations of I_j are varied when I_j is matched to I_i and I_k, yielding fragmentary feature tracks.

of the entire model. Fig. 1 presents an example.

Recently, detector-free matchers [48, 10, 58] achieve state-of-the-art performance on the image-matching task. They have shown a strong capability for matching low-textured regions with the help of the detector-free design and the attention mechanism [54]. They often use a coarse-to-fine matching strategy for efficiency. The dense matching on a coarse grid is first performed between downsampled feature maps of two images. Then, the feature locations of coarse matches on one image are fixed, while their sub-pixel correspondences are searched on the other image with fine-level feature maps. Therefore, the resulting feature locations in an image depend on the other image, as shown in Fig. 2. This pair-dependent nature leads to fragmentary feature tracks when running pair-wise matching over multiple views, which makes detector-free matchers not directly applicable to existing SfM systems.

In this paper, we propose a new SfM framework that is able to leverage the recent success of detector-free matching and recover highly-accurate camera poses even for texture-poor scenes. An overview of our pipeline is depicted in Fig. 3. To solve the inconsistency issue of detector-free matching, our SfM framework reconstructs the scene in a coarse-to-fine manner, which first builds a coarse SfM model with the quantized matches, and then iteratively refines the model towards higher accuracy.

Specifically, our framework first matches image pairs with a detector-free feature matcher, e.g., LoFTR [48]. Then, in the coarse reconstruction phase, we quantize the feature locations by rounding them into a coarse grid to improve consistency and reconstruct a coarse SfM model. This coarse model provides initial camera poses and scene structures for the later refinement phase. Next, we propose an iterative refinement pipeline that alternates between a feature track refinement phase and a geometry refinement phase to improve pose and point cloud accuracy. The feature track refinement module is built on a novel transformer-based multi-view matching network, which enhances the discrimitiveness of extracted features by encoding positional and multi-view context with self- and cross-attention mechanisms. Based on refined feature tracks, the geometry refinement module uses bundle adjustment and track topology adjustment to improve the accuracy of camera poses and point clouds.

Experiments on the public ETH3D dataset [45] and Image Matching Challenge (IMC) [25] dataset demonstrate that our detector-free SfM framework outperforms state-of-the-art detector-based SfM systems with respect to various metrics. To further evaluate and demonstrate the capability of our SfM framework on challenging scenes, we also collect a texture-poor SfM dataset which is composed of 17 scenes with 1020 image bags. Thanks to the detector-free design and the iterative refinement pipeline, our framework can recover accurate camera poses with high registration rates even for challenging texture-poor scenes. Fig. 1 presents some examples.

Contributions:

- A new detector-free SfM framework built upon detector-free matchers to handle texture-poor scenes.
- An iterative refinement pipeline with a transformer-based multi-view matching network to efficiently refine both feature tracks and reconstruction results.
- A new texture-poor SfM dataset with ground-truth pose annotations.

2. Related Work

Structure-from-Motion. Feature correspondence-based SfM methods have long been investigated [33, 7, 18, 35]. Many previous works focus on improving the efficiency and robustness of large-scale scene reconstruction [4, 5, 12, 62, 13, 44]. Some methods try to disambiguate matches when applied to scenes with highly repetitive or symmetric structures [38, 61]. As discussed in the introduction, these methods require feature detection and matching at the beginning of the pipeline. In challenging scenes, especially in texture-poor regions, poor keypoint detection will affect the overall SfM pipeline.

More recent end-to-end SfM methods propose to directly regress poses [55, 69, 34, 67] or solve poses using differential bundle adjustment (BA) [49, 20]. These methods avoid explicit feature matching and thus don't suffer from poor feature matching. However, they have limited scalability and generalizability on real-world settings. With the success of recent neural scene representations, some methods [29, 24] try to optimize poses with differentiable rendering. However, they often rely on using previous correspondence-based methods, e.g., COLMAP [44], to provide initial poses, as joint pose and scene optimization from scratch is difficult to converge and prone to local minima, c.f. [29, 32].

Different from these previous methods, our detector-free SfM framework eliminates the requirement of sparse feature detection at the beginning of the pipeline, which is more robust in challenging scenarios such as low-textured regions and repetitive patterns. Moreover, our framework is scalable to large-scale scenes and can handle in-the-wild data with extreme view-point and illumination changes. [60, 22] are

relevant to our framework which also eliminates feature detection by performing coarse grid-level matching first and then refining 2D points for sub-pixel accuracy. Different from their refinement that is single- or two-view based, our framework is capable of leveraging multi-view information to refine a feature track.

Feature Matching. Feature Matching is often a prerequisite for SfM and SLAM. A typical feature matching pipeline [31, 40, 14, 15, 37] is to detect and describe keypoints on each image, and then match them by nearest neighbor search or learning-based matchers [41, 9]. The merit of these methods is the high matching efficiency based on the sparse points. However, for challenging scenarios, especially low-textured regions, poor feature detection at the beginning is the bottleneck and affects the overall SfM system.

In recent years, many methods directly match image pairs in a dense [52] or semi-dense manner [39, 27, 48, 50, 10, 58], avoiding feature detection. With the help of Transformer [54], some semi-dense matching methods [48, 10, 58] achieve higher accuracy compared with detector-based baselines and show strong capabilities in building correspondences on low-textured regions. However, due to their inconsistency problem when matching multiple views (shown in Fig. 2), it is hard to directly apply them to the current SfM systems, as discussed in the introduction. While rounding [10] or merging strategies [46] could be used to produce long feature tracks for SfM, these strategies sacrifice the matching accuracy, which will significantly reduce the accuracy of the reconstructed SfM models. Unlike them, our detector-free SfM framework with a coarse-to-fine manner can recover highly accurate poses and point clouds.

Multi-View Refinement. Accurate multi-view correspondences are crucial for recovering accurate point clouds and camera poses in SfM. The technical challenge is that per-view detection of feature points cannot guarantee their geometric consistency among multiple views. To solve this problem, some previous methods perform multi-view refinement with flow [16] or dense features [30], which bring significant accuracy improvement for SfM. PatchFlow [16] first estimates the dense flow field within the local patch of each tentative pair and then refines multi-view 2D locations by minimizing the energy function based on the estimated flow. PixSfM [30] performs feature-metric keypoint adjustment and bundle adjustment to refine 2D feature locations before SfM and the entire scene after SfM, respectively. Our detector-free SfM framework may adopt these two methods to refine the quantized matches and SfM models. However, PatchFlow suffers from high computation due to pair-wise flow estimations. PixSfM needs to preserve feature patches or cost maps of all 2D observations in the memory for the feature-metric BA. Given that detector-free matchers produce significantly more correspondences than sparse matchers, the memory footprint of adapting PixSfM to our detector-free SfM pipeline is inevitably large, especially on large-scale scenes. Different from them, we devise a transformer-based multi-view refinement matching module, which can efficiently and accurately refine a feature track with a single forward pass. Moreover, thanks to the design of our refinement phase that separately refines feature tracks and performs geometry refinement, the geometric BA can be leveraged for efficiency both in terms of speed and memory. Experimental comparisons are provided in Sec. 4.6.

3. Method

An overview of our detector-free SfM framework is shown in Fig. 3. Given a set of unordered images $\{\mathbf{I}_i\}$, our objective is to recover camera poses $\{\boldsymbol{\xi}_i \in \mathbb{SE}(3)\}$, intrinsic parameters $\{\mathbf{C}_i\}$ and a scene point cloud $\{\mathbf{P}_j\}$. The recovered camera poses are in a global coordinate system. To achieve this goal, we propose a two-stage pipeline, in which we first establish correspondences between image pairs with a detector-free matcher and reconstruct an initial coarse SfM model (Sec. 3.1). Then, we perform iterative refinement to improve the accuracy of poses and point clouds (Sec. 3.2).

3.1. Detector-Free Matching and Coarse SfM

For a set of unordered images, our framework directly performs detector-free semi-dense feature matching between image pairs instead of first detecting sparse keypoints as in traditional SfM pipeline [44]. Eliminating the keypoint detection phase can help avoid poor detection affecting the overall SfM system and benefit the reconstruction of challenging texture-poor scenes.

Match Quantization. Directly adapting the correspondences of semi-dense matchers for SfM is not straightforward, due to the inconsistent multi-view matches as depicted in Fig. 2 and discussed in the introduction. Our idea is to strive for match consistency by sacrificing accuracy in the coarse SfM phase. Concretely, we quantize the 2D locations of matches into a grid: $\lfloor \mathbf{x}/r \rceil * r$, where $\lfloor \cdot \rceil$ is the rounding operator and r is the grid cell size. This quantization step forces multiple subpixel matches that are close to each other to merge into a single grid node, which improves consistency. Note that the coarse-level correspondences output by some detector-free [48, 58, 10] matchers are typically at $1/8$ image resolution, which can directly be used as quantized matches. The ablation analysis of r is given in Sec. 4.5.

After the match quantization, we utilize these coarse matches for incremental mapping [44] to obtain a coarse SfM model. The accuracy of recovered camera poses and point clouds are limited due to the match quantization, which serves as the initialization of our refinement framework introduced in the next section.

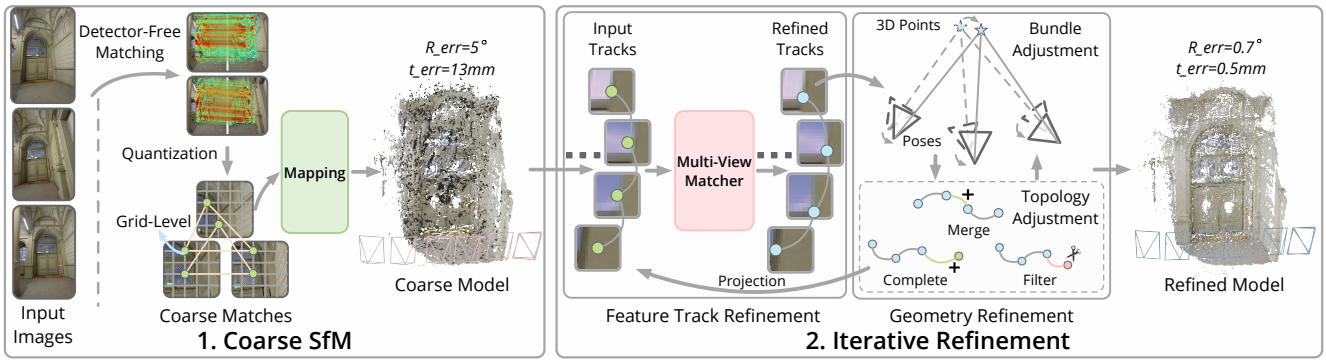

Figure 3. **Pipeline Overview.** Beginning with a collection of unordered images, the **Coarse SfM** stage generates an initial SfM model based on multi-view matches from a detector-free matcher. Then, the **Iterative Refinement** stage improves the accuracy of the SfM model by alternating between the feature track refinement module and the geometry refinement module.

3.2. Iterative SfM Refinement

We proceed to refine the initial SfM model to obtain improved camera poses and point clouds. To this end, we propose an iterative refinement pipeline. Within each iteration, we first enhance the accuracy of feature tracks with a multi-view matching module. These refined feature tracks are then fed into a geometry refinement phase which optimizes camera poses and point clouds jointly. The refinement process can be performed multiple times for higher accuracy. An overview is shown in Fig. 3.

3.2.1 Feature Track Refinement

A feature track $\mathcal{T}_j = \{\mathbf{x}_k \in \mathbb{R}^2 | k = 1 : N_j\}$ is a set of 2D keypoint locations in multi-view images corresponding to a 3D scene point $\mathbf{P}_j$. We devise a multi-view matching module to efficiently refine feature tracks $\{\mathcal{T}_j\}$ for high accuracy, which is illustrated in Fig. 4. The basic idea is to locally adjust the keypoint locations in all views so that the correlation among their features is maximized.

As exhaustively correlating all pairs of views is computationally intractable, we select a reference view, extract the feature at the keypoint in the reference view, and correlate it with the local feature maps with a size of $p \times p$ around the keypoints in other views (called query views), yielding a set of $p \times p$ heatmaps that can be viewed as distributions of the keypoint locations. In each query view, we compute the expectation and variance over each heatmap as the refined keypoint location and its uncertainty, respectively. This process gives us a candidate feature track with refined keypoint locations in all query views as well as the uncertainty of this candidate track, i.e., the sum of variance over all the heatmaps. To also refine the keypoint location in the reference view, we sample a $w \times w$ grid of reference locations around the original keypoint in the reference view. Then, we repeat the above feature correlation procedure to produce a candidate feature track for each sampled reference location. Finally, the candidate track with the smallest uncertainty is selected as the refined feature track $\mathcal{T}_j^*$.

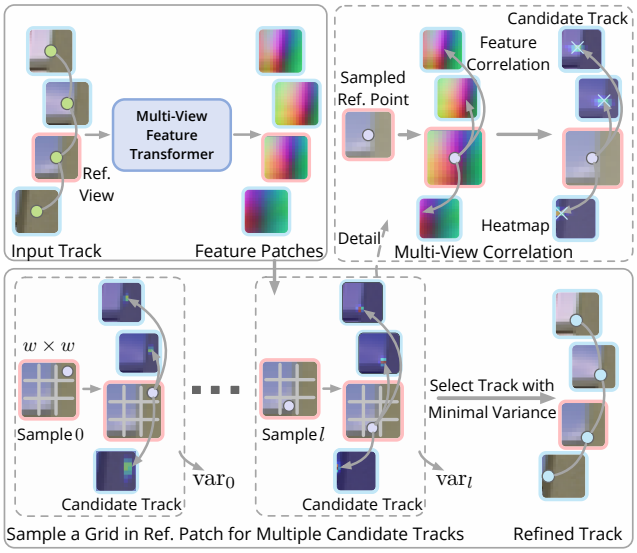

Figure 4. **Multi-View Matching Module.** Given an input feature track with a selected reference view (□), the local patches centered at the keypoints are fed into a multi-view feature transformer to extract feature patches. A $w \times w$ grid of reference locations is sampled in the reference view. For each reference location (●), its feature is correlated with the feature patches of query views (□) to obtain heatmaps that indicate the expected keypoint locations and their variances in the query views, yielding a candidate feature track. This process is repeated for all reference locations. Finally, the candidate track with the smallest variance is selected as the refined track.

Reference View Selection. For each feature track, our criteria to select the reference view is to minimize the keypoint scale differences between the reference view and query views to improve the matchability. Specifically, we compute the depth values of keypoints based on the currently recovered poses and point clouds, which indicate the scale information. Then, the view with a medium scale across the track is selected as the reference view whereas the rest views

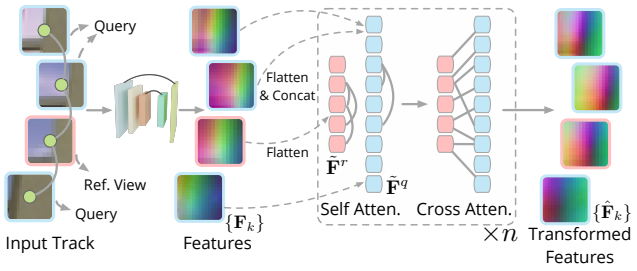

Figure 5. **Multi-View Feature Transformer.** The local patches centered at the keypoints of an input feature track are fed into a CNN to extract features and then flattened and concatenated to perform multiple self- and cross-attentions.

are query views. More details about scale estimation can be found in the supplementary material.

Multi-View Feature Transformer. The multi-view matching needs to extract local feature patches centered at 2D keypoints of each $\mathcal{T}_j$. Instead of using a CNN, we design a multi-view feature transformer to enhance the discrimitiveness of extracted features by encoding multi-view context with attention mechanisms. As shown in Fig. 5, we feed the $p \times p$ image patches centered at each keypoint into a CNN backbone to obtain a set of feature patches $\{\mathbf{F}_k \in \mathbb{R}^{p \times p \times c}\}$, where c is the number of channels. Then, $\{\mathbf{F}_k\}$ are flattened to $\{\tilde{\mathbf{F}}_k \in \mathbb{R}^{m \times c}\}$, where $m = p \times p$. The flattened features of query views are concatenated into a single query feature $\tilde{\mathbf{F}}^q$ along the first dimension. Then, we perform self- and cross-attention by n times between flattened reference feature $\tilde{\mathbf{F}}^r$ and query feature $\tilde{\mathbf{F}}^q$ to obtain the transformed multi-view features $\{\hat{\mathbf{F}}_k\}$, which are used for feature correlation to refine the feature track.

Training. Besides the detector-free matcher, the only learned module in our framework is the multi-view feature transformer. It is trained on MegaDepth [28] by minimizing the average ℓ_2 loss on keypoint locations between the refined tracks and the ground-truth tracks. We construct training data by sampling image bags on each scene with a maximum of six images in each bag. Image bags are sampled by the co-visibility extracted from the provided scene SfM model. Then, the ground-truth feature tracks in each bag are built by randomly selecting a reference image and projecting its grid points to other views by depth maps. The 2D locations of tracks in the query views are perturbed randomly by a maximum of seven pixels to generate the coarse feature tracks, which are the input of our multi-view matching module. More details are provided in the supplementary material.

3.2.2 Geometry Refinement

Based on the previous refined feature tracks $\{\mathcal{T}_j^*\}$, our geometry refinement pipeline iteratively refines the poses, intrinsics, point clouds, as well as topology of feature tracks. Track topology means the graph structure of a set of connected 2D keypoints.

Unlike PixSfM [30] that needs to preserve feature patches or cost maps of all 2D observations in memory to perform feature-metric BA, we can directly perform the efficient geometric BA [51] to optimize poses and point clouds based on the refined feature tracks. Formally, we minimize the reprojection error to optimize intrinsic parameters $\{\mathbf{C}_i\}$, poses $\{\boldsymbol{\xi}_i\}$, and 3D points $\{\mathbf{P}_j\}$:

$$E = \sum_j \sum_{\mathbf{x}_k^* \in \mathcal{T}_j^*} \rho\left(\left\|\pi\left(\boldsymbol{\xi}_i \cdot \mathbf{P}_j, \mathbf{C}_i\right) - \mathbf{x}_k^*\right\|_2^2\right) \ ,$$

where $\pi(\cdot)$ project points in the camera coordinate to image plane by $\mathbf{C}_i$, $\rho(\cdot)$ is a robust loss function [21].

After BA, we perform the feature track topology adjustment (TA) based on the refined model, which benefits further BA and multi-view matching. Since the overall scene is more accurate after the multi-view refinement and BA, we adjust the topology of feature tracks by adding 2D keypoints that previously failed to be registered into feature tracks and merging the tracks that can meet the reprojection criteria at this time, following [63, 44]. The outlier filtering [47, 63, 44] is also performed to further reject points that cannot meet the maximum reprojection threshold ϵ after the refinement.

We alternate BA and TA multiple times to obtain the refined poses and point clouds. Then, we project the refined point clouds to images with the current poses to update their 2D locations, which will serve as the initialization of the multi-view matching in the next refinement iteration.

3.3. Texture-Poor SfM Dataset

We collect an SfM dataset composed of 17 object-centric texture-poor scenes with accurate ground-truth poses. In our dataset, low-textured objects are placed on a textureless plane. For each object, we record a video sequence of around 30 seconds surrounding the object. The per-frame ground-truth poses are estimated by ARKit [1] and BA post-processing, with the assistance of textured markers, which are cropped out in test images. To impose larger viewpoint changes, we sample 60 subset image bags for each scene, similar to the IMC dataset [25]. Example images are shown in Fig. 6 and more details are in the supplementary material.

4. Experiments

4.1. Baselines and Datasets

Baselines. We compare our method with a few baseline methods in two categories: 1) Detector-based SfM pipeline [44] with different features, including SIFT [31], D2-Net [15], R2D2 [37] and SuperPoint (SP) [14], and matchers, including Nearest Neighbor (NN) and SuperGlue (SG) [41]. All these detector-based baselines are cou-

Type	Method	ETH3D Dataset			IMC Dataset			Texture-Poor SfM Dataset		
		AUC@1°	AUC@3°	AUC@5°	AUC@3°	AUC@5°	AUC@10°	AUC@3°	AUC@5°	AUC@10°
Detector-Based	COLMAP (SIFT+NN)	26.71	38.86	42.14	23.58	32.66	44.79	2.87	3.85	4.95
	SIFT + NN + PixSfM	26.94	39.01	42.19	25.54	34.80	46.73	3.13	4.08	5.09
	D2Net + NN + PixSfM	34.50	49.77	53.58	8.91	12.26	16.79	1.54	2.63	4.54
	R2D2 + NN + PixSfM	43.58	62.09	66.89	31.41	41.80	54.65	3.79	5.51	7.84
	SP + SG + PixSfM	50.82	68.52	72.86	45.19	57.22	70.47	14.00	19.23	24.55
Detector-Free	LoFTR + PixSfM	54.35	73.97	78.86	44.06	56.16	69.61	20.66	30.49	42.01
	Ours (LoFTR)	**59.12**	**75.59**	**79.53**	46.55	58.74	72.19	26.07	35.77	45.43
	Ours (AspanTrans.)	57.23	73.71	77.70	**46.79**	**59.01**	**72.50**	25.78	35.69	45.64
	Ours (MatchFormer)	56.70	73.00	76.84	45.83	57.88	71.22	**26.90**	**37.57**	**48.55**

Table 1. **Results of Multi-View Camera Pose Estimation.** Our framework is compared with detector-based and detector-free baselines on multiple datasets by the AUC of pose error at different thresholds. **Bold** and underline indicate the best and second-best results.

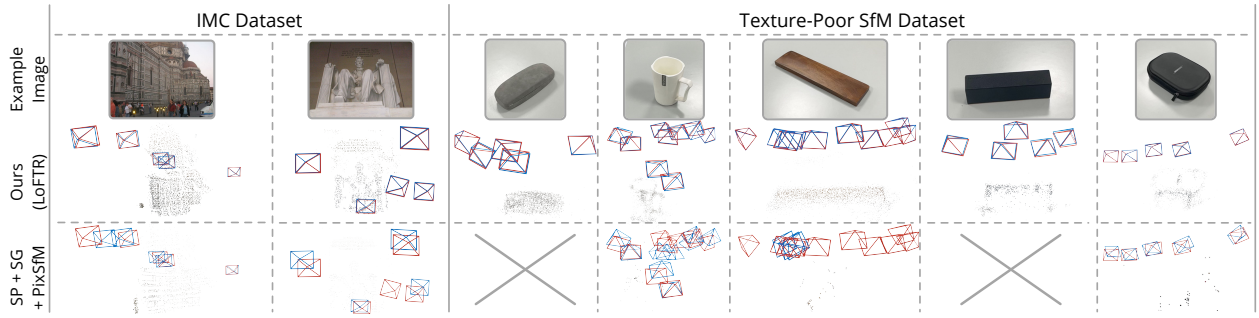

Figure 6. **Qualitative Results.** Our method with detector-free matcher LoFTR [48] is qualitatively compared with the detector-based baseline *SP + SG + PixSfM* on multiple scenes. The red cameras (♢) are ground-truth poses while the blue cameras (♢) are recovered poses.

pled with PixSfM [30], which is the state-of-the-art SfM refinement method. 2) Detector-free SfM baseline LoFTR [48] + PixSfM [30], where PixSfM is fed with LoFTR's matches, which are quantized by the same strategy as in our pipeline.

Datasets. Datasets used for evaluation include the Image Matching Challenge (IMC) 2021 dataset [25], the ETH3D dataset [45], and the proposed Texture-Poor SfM dataset. These datasets cover multiple types of scenes with different challenges. The IMC Phototourism dataset contains large-scale outdoor scenes. All eight test scenes with 1400 subsampled image bags are used for evaluation. The key challenge of this dataset is the sparse views with large viewpoint and illumination changes. The ETH3D dataset contains 25 indoor and outdoor scenes with sparsely captured high-resolution images and accurately calibrated poses by Lidar as ground truth. The proposed Texture-Poor SfM dataset contains low-textured object-centric scenes with 1020 subsampled image bags in total. On all datasets, images are considered unordered for all methods.

4.2. Implementation Details

Our detector-free SfM framework is implemented with multiple detector-free matchers, including LoFTR [48], MatchFormer [58] and AspanTransformer [10], to demonstrate the compatibility of our pipeline. In the coarse SfM phase, we use their coarse-level matches ($r = 8$) as quantized matches for SfM [44]. Then, the refinement is performed twice. A maximum of 16 views are used for multi-view refinement matching, where longer tracks will be divided into segments and processed separately. The local patch size for feature extraction $p = 15$ and the region size for reference location search $w = 7$. The backbone from S2DNet [19] is used as the CNN feature extractor, and the number of attention groups $n = 2$. The linear attention [26] is used in all attention layers for efficiency. In geometry refinement, the BA and topology adjustment are alternated five times, and the outlier filter threshold $\epsilon = 3px$. The running time reported in the experiments is measured using four NVIDIA-V100 GPUs for parallelized matching and 16 CPU cores for BA.

4.3. Multi-View Camera Pose Estimation

Camera pose estimation is a central goal of SfM. This section evaluates the recovered multi-view poses.

Evaluation Protocols. On all datasets, matches are built exhaustively between all tentative image pairs, and the same image resizing strategy is used for all methods. For all the baselines, the default hyperparameters in their original implementations are used. The AUC of pose error at different thresholds is used as the metric to evaluate the accuracy of estimated multi-view poses, following the IMC benchmark [25] and PixSfM [30]. More details are provided in the supplementary material.

Results. As shown in Tab. 1, our detector-free SfM framework outperforms existing baselines over all datasets. On the

	Method	Accuracy (%)			Completeness (%)		
		1cm	2cm	5cm	1cm	2cm	5cm
Detector-Based	SIFT + NN + PixSfM	76.18	85.60	93.16	0.17	0.71	3.29
	D2Net + NN + PixSfM	74.75	83.81	91.98	0.83	2.69	8.95
	R2D2 + NN + PixSfM	74.12	84.49	91.98	0.43	1.58	6.71
	SP + SG + PixSfM	79.01	87.04	93.80	0.75	2.77	11.28
Detector-Free	LoFTR + PatchFlow	66.73	78.73	89.93	3.48	11.34	30.96
	LoFTR + PixSfM	74.42	84.08	92.63	2.91	9.39	27.31
	Ours (LoFTR)	**80.38**	**89.01**	**95.83**	3.73	11.07	29.54
	Ours (AspanTrans.)	77.63	87.40	95.02	**3.97**	**12.18**	**32.42**
	Ours (MatchFormer)	79.86	88.51	95.48	3.76	11.06	29.05

Table 2. **Results of 3D Triangulation.** Our method is compared with the baselines on the ETH3D [45] dataset using accuracy and completeness metrics with different thresholds.

ETH3D dataset with high-resolution images, our framework with LoFTR achieves the highest multi-view pose accuracy. Even when detector-based methods are further refined with PixSfM for multi-view consistency, our framework still surpasses them by a large margin. On the IMC dataset with large viewpoint and illumination changes, the detector-based baseline SP+SG+PixSfM achieves remarkable performance, while our detector-free framework consistently performs better on all metrics. The results demonstrate the robustness and effectiveness of our framework on large-scale outdoor scenes with internet images. Due to the severe low-textured scenario and viewpoint changes in the Texture-Poor SfM dataset, detector-based methods struggle with poor keypoint detection, as shown in Fig. 6. Thanks to the detector-free design, our framework achieves significantly higher accuracy.

Compared with LoFTR+PixSfM, the detector-free baseline that uses the same LoFTR coarse matches as ours, our framework is more accurate on all datasets and metrics, especially on the AUC@1° metric with a strict error threshold, which demonstrates the effectiveness of our iterative refinement pipeline with the multi-view matching module.

4.4. 3D Triangulation

With known camera poses and intrinsics, triangulating accurate scene point clouds based on image correspondences is another important task in SfM. This section evaluates the accuracy and completeness of triangulated point clouds.

Evaluation Protocols. The training set of ETH3D is used for evaluation, which is composed of 13 indoor and outdoor scenes with millimeter-accurate scanned dense point clouds as ground truth. We follow the protocol used in [16, 30], which triangulates the scene point clouds with fixed camera poses and intrinsics. Then, we use the ETH3D benchmark [45] to evaluate the triangulated point clouds in terms of accuracy and completeness. The metrics are reported with different distance thresholds including $(1cm, 2cm, 5cm)$, which are averaged across all scenes. The results of SIFT, D2Net, and R2D2 descriptors are from the PixSfM [30] paper, while the results of other baselines are obtained by running their open-source code.

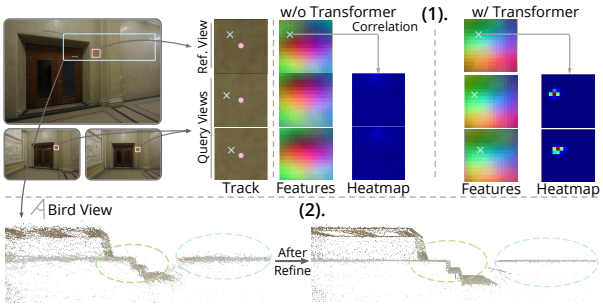

Figure 7. **Effects of Transformer and Refinement. 1.** For a feature track in the texture-poor region, its feature patches (visualized by PCA) become more discriminative after the multi-view transformer. ○ and × represent coarse and refined keypoint locations, respectively. **2.** The point cloud after refinement becomes more accurate.

Results. The results are presented in Tab. 2. Despite the trade-off between accuracy and completeness, our detector-free SfM framework achieves better performances on both metrics. Compared to the state-of-the-art detector-based baseline SP+SG+PixSfM, our framework with LoFTR coarse matches achieves higher accuracy with $\sim 3\times$ reconstruction completeness, thanks to the iterative refinement module. Our framework with AspanTransformer coarse matches achieves higher completeness while sacrificing a little accuracy compared to using the LoFTR matches. Compared with the detector-free LoFTR+PixSfM, our method using the same input matches achieves higher performances both in terms of accuracy and completeness, which further demonstrates the effectiveness of our refinement module.

4.5. Ablation Studies

We conduct several experiments to validate the efficacy of our design choices on the ETH3D dataset with triangulation metrics. More ablation studies with pose metrics are in the supplementary material.

Coarse Match Quantization. Tab. 3 (1) shows the impact of the match quantization rounding ratio r. Our framework achieves satisfying accuracy and completeness directly using the coarse-level matches output by LoFTR ($r = 8$). Using a smaller quantization ratio yields better matching accuracy but significantly more 2D and 3D points, thus decreasing running efficiency.

Number of Refinement Iterations. Tab. 3 (2) reports the results after each refinement iteration. Without refinement, the coarse SfM point cloud is inaccurate due to the match quantization. After the first iteration, the accuracy improves significantly, especially on the *1cm* distance threshold. Increasing the number of iterations can improve accuracy, with a slight decrease in completeness due to the track merge.

		Accu. (%)		Complete. (%)		Time (s)
		1cm	2cm	1cm	2cm	
(1) Quantization ratio	$r = 8$	80.38	89.01	3.73	11.07	557
	$r = 4$	**81.58**	**89.82**	4.41	12.27	718
	$r = 2$	81.18	89.78	**5.41**	**14.15**	791
(2) Number of iterations	No refine.	42.13	59.92	2.21	8.45	296
	1 iter	77.62	87.04	**3.83**	**11.44**	430
	2 iter	80.38	89.01	3.73	11.07	557
	3 iter	**81.26**	**89.59**	3.57	10.64	678
(3) Number of views in multi-view matching	2 views	69.77	81.69	2.02	7.10	438
	4 views	72.30	83.42	2.48	8.35	435
	8 views	74.68	85.02	3.09	9.84	431
	16 views	**77.62**	**87.04**	**3.83**	**11.44**	430
(4) Refinement designs	Full model	**80.38**	**89.01**	3.73	11.07	557
	w/o transformer	71.85	82.66	2.72	8.79	541
	w/o ref. location search	76.66	86.79	**4.16**	**12.56**	554
	w/o topology adjustment	75.58	85.47	4.07	12.17	552

Table 3. **Ablation Studies.** On the ETH3D dataset, we quantitatively evaluate the impact of the quantization ratio, the number of iterations of refinement, the number of views used for multi-view matching, and other designs in refinement. The reported triangulation accuracy and completeness are averaged across all scenes, while the running time is evaluated on a single scene *Kicker*.

Refining more than twice brings little accuracy improvement while spending more time. Therefore, we only perform refinement twice for both efficiency and accuracy.

Maximum Number of Views in Multi-view Matching. Tab. 3 (3) shows the effect of the number of views used for multi-view matching in a single iteration of refinement. It is shown that using more views for multi-view matching consistently improves both accuracy and completeness without significantly affecting running time.

Refinement Designs. Tab. 3 (4) shows the benefits of the feature transformer and reference location search in multi-view matching and the track topology adjustment in the geometry refinement. Compared with multi-view matching that directly uses backbone CNN features for matching, using multi-view transformed features can significantly improve accuracy and completeness. The result demonstrates the effectiveness of the proposed transformer module, which considers feature relations among multiple views and helps disambiguate features for more accurate matching, as visualized in Fig. 7 (1). Reference location search in the reference view brings a 3.7% improvement on the *1cm* metric. Without the track topology adjustment in geometry refinement, the point clouds' accuracy drops by 4.6% on the strict threshold (*1cm*), which demonstrates the benefits of topology adjustment on accuracy.

4.6. Efficiency on Large-Scale Scenes

We conduct experiments on the Aachen v1.1 dataset [42, 68, 43] to demonstrate the efficiency of our framework in handling large-scale scenes. The time and memory costs for refinement are shown in Tab. 4. We compare our method with PixSfM that uses the same LoFTR [48] coarse matches and the same number of CPU cores as ours, where its cost

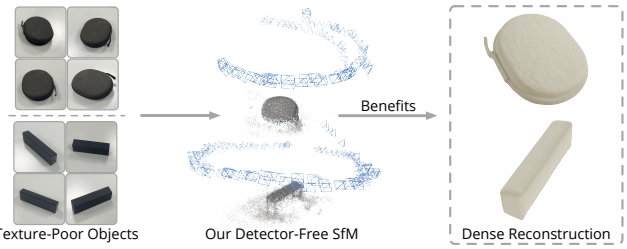

Figure 8. **Applications.** The recovered poses on texture-poor scenes by our detector-free SfM framework benefit substream tasks, e.g., dense reconstruction using neural implicit fields [57].

		500 Images	1000 Images	2000 Images
Number of 3D Points		553k	1525k	3235k
Ours Refinement Time (s)		312	969	2319
BA Memory (GB)	PixSfM (Feature Map)	161.7	393.8	904.5
	PixSfM (Cost Map)	3.79	9.23	21.2
	Ours	**0.37**	**1.21**	**2.63**

Table 4. **Efficiency on Large-Scale Scenes.** Our method is compared with PixSfM. Both of them use LoFTR coarse matches as input and share the same coarse SfM initialization. Only refinement time and peak memory footprint during BA are reported.

map approximation is used to reduce the memory footprint and improve efficiency. Our pipeline achieves competitive efficiency even on the scene with 2000 images and 3.2 million 3D points. Moreover, since we perform multi-view matching first and then refine geometry, we do not need to store the feature patch or cost map of each 2D point in memory for BA as PixSfM does. Therefore, the geometric BA in our pipeline can be very efficient with a small memory footprint on large-scale scenes, which significantly outperforms PixSfM in memory efficiency.

On the scene with 2000 images, the detector-free matching [48] and coarse SfM takes 4.2 hours in total, due to a large number of semi-dense matches and 3D points. Thus, the overall speed of our framework is slower than detector-based systems that are based on sparse features. More running time comparisons on large-scale scenes in the 1DSfM [62] dataset are shown in the supplementary material.

5. Conclusions

We propose a new detector-free SfM framework to recover camera poses and point clouds from unordered images. In contrast to traditional SfM systems that depend on keypoint detection at the beginning, our framework leverages the recent success of detector-free matchers to avoid early determination of keypoints which may break down the whole SfM system if the detected keypoints are not repeatable, which often occur in challenging texture-poor scenes. Extensive experiments demonstrate that our framework outperforms detector-based SfM baselines across all datasets and metrics. We believe the proposed SfM framework opens up the possibility to reconstruct texture-poor scenes from unordered images as shown in Fig. 1 and Fig. 6 and benefits

downstream tasks such as dense reconstruction and view synthesis as shown in Fig. 8, as it can recover accurate poses and relatively dense point clouds.

Limitations and future works. The main limitation of our framework is efficiency. Due to the significant number of matches produced by detector-free matches, the overall mapping phase will be inevitably slower than the previous detector-based pipelines, especially on large-scale scenes.

As future work, our framework can be extended with more advanced parallelized BA methods [23, 36] for better efficiency and integration with multi-modality data such as depth maps and IMUs if available in real applications.

References

[1] ARKit. https://developer.apple.com/augmented-reality/. 5, 12

[2] Metashape. https://www.agisoft.com/. 1

[3] Reality capture. https://www.capturingreality.com/. 1

[4] Sameer Agarwal, Yasutaka Furukawa, Noah Snavely, Ian Simon, Brian Curless, Steven M. Seitz, and Richard Szeliski. Building rome in a day. *ICCV*, 2009. 1, 2

[5] Sameer Agarwal, Noah Snavely, Steven M. Seitz, and Richard Szeliski. Bundle adjustment in the large. In *ECCV*, 2010. 2, 11

[6] Relja Arandjelović, Petr Gronát, Akihiko Torii, Tomás Pajdla, and Josef Sivic. Netvlad: Cnn architecture for weakly supervised place recognition. *TPAMI*, 2015. 13, 14

[7] Paul A. Beardsley, Philip H. S. Torr, and Andrew Zisserman. 3d model acquisition from extended image sequences. In *ECCV*, 1996. 2

[8] Martin Byröd and Kalle Åström. Conjugate gradient bundle adjustment. In *ECCV*, 2010. 11

[9] Hongkai Chen, Zixin Luo, Jiahui Zhang, Lei Zhou, Xuyang Bai, Zeyu Hu, Chiew-Lan Tai, and Long Quan. Learning to match features with seeded graph matching network. *ICCV*, 2021. 3

[10] Hongkai Chen, Zixin Luo, Lei Zhou, Yurun Tian, Mingmin Zhen, Tian Fang, David N. R. McKinnon, Yanghai Tsin, and Long Quan. Aspanformer: Detector-free image matching with adaptive span transformer. In *ECCV*, 2022. 2, 3, 6, 13

[11] Ashley Chow, Eduard Trulls, HCL-Jevster, Kwang Moo Yi, lcmrll, old ufo, Sohier Dane, tanjigou, WastedCode, and Weiwei Sun. Image matching challenge 2023, 2023. 1

[12] David J. Crandall, Andrew Owens, Noah Snavely, and Daniel P. Huttenlocher. Discrete-continuous optimization for large-scale structure from motion. *CVPR*, 2011. 1, 2

[13] Zhaopeng Cui and Ping Tan. Global structure-from-motion by similarity averaging. *ICCV*, 2015. 1, 2, 14

[14] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superpoint: Self-supervised interest point detection and description. *CVPRW*, 2018. 1, 3, 5, 12

[15] Mihai Dusmanu, Ignacio Rocco, Tomas Pajdla, Marc Pollefeys, Josef Sivic, Akihiko Torii, and Torsten Sattler. D2-Net: A Trainable CNN for Joint Detection and Description of Local Features. In *CVPR*, 2019. 3, 5

[16] Mihai Dusmanu, Johannes L. Schönberger, and Marc Pollefeys. Multi-View Optimization of Local Feature Geometry. In *ECCV*, 2020. 3, 7, 11, 12

[17] Maxime Ferrera, Vincent Creuze, Julien Moras, and Pauline Trouvé-Peloux. Aqualoc: An underwater dataset for visual–inertial–pressure localization. *The International Journal of Robotics Research*, 2019. 14

[18] Andrew William Fitzgibbon and Andrew Zisserman. Automatic camera recovery for closed or open image sequences. In *ECCV*, 1998. 2

[19] Hugo Germain, Guillaume Bourmaud, and Vincent Lepetit. S2dnet: Learning image features for accurate sparse-to-dense matching. In *ECCV*, 2020. 6, 11

[20] Xiaodong Gu, Weihao Yuan, Zuozhuo Dai, Siyu Zhu, Chengzhou Tang, and Ping Tan. Dro: Deep recurrent optimizer for structure-from-motion. *ArXiv:2103.13201*, 2021. 2

[21] Frank R Hampel, Elvezio M Ronchetti, Peter J Rousseeuw, and Werner A Stahel. *Robust statistics: the approach based on influence functions*. Wiley, 1986. 5

[22] Xingyi He, Jiaming Sun, Yuang Wang, Di Huang, Hujun Bao, and Xiaowei Zhou. Onepose++: Keypoint-free one-shot object pose estimation without CAD models. In *NeurIPS*, 2022. 2

[23] Jingwei Huang, Shan Huang, and Mingwei Sun. Deeplm: Large-scale nonlinear least squares on deep learning frameworks using stochastic domain decomposition. *CVPR*, 2021. 9

[24] Yoonwoo Jeong, Seokjun Ahn, Christopher Bongsoo Choy, Anima Anandkumar, Minsu Cho, and Jaesik Park. Self-calibrating neural radiance fields. *ICCV*, 2021. 2

[25] Yuhe Jin, Dmytro Mishkin, Anastasiia Mishchuk, Jiri Matas, Pascal Fua, Kwang Moo Yi, and Eduard Trulls. Image matching across wide baselines: From paper to practice. *IJCV*, 2021. 2, 5, 6, 13

[26] Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are RNNs: Fast autoregressive transformers with linear attention. In *ICML*, 2020. 6, 11

[27] Xinghui Li, Kai Han, Shuda Li, and Victor Prisacariu. Dual-resolution correspondence networks. In *NeurIPS*, 2020. 3

[28] Zhengqi Li and Noah Snavely. Megadepth: Learning single-view depth prediction from internet photos. *CVPR*, 2018. 5, 12

[29] Chen-Hsuan Lin, Wei-Chiu Ma, Antonio Torralba, and Simon Lucey. Barf: Bundle-adjusting neural radiance fields. *ICCV*, 2021. 2

[30] Philipp Lindenberger, Paul-Edouard Sarlin, Viktor Larsson, and Marc Pollefeys. Pixel-perfect structure-from-motion with featuremetric refinement. *ICCV*, 2021. 3, 5, 6, 7, 13

[31] G LoweDavid. Distinctive image features from scale-invariant keypoints. *IJCV*, 2004. 1, 3, 5

[32] Quan Meng, Anpei Chen, Haimin Luo, Minye Wu, Hao Su, Lan Xu, Xuming He, and Jingyi Yu. Gnerf: Gan-based neural radiance field without posed camera. *ICCV*, 2021. 2

[33] Roger Mohr, Long Quan, and Francoise Veillon. Relative 3d reconstruction using multiple uncalibrated images. *The International Journal of Robotics Research*, 1993. 2

[34] Chethan Parameshwara, Gokul Hari, Cornelia Fermuller, Nitin J. Sanket, and Yiannis Aloimonos. Diffposenet: Direct differentiable camera pose estimation. *CVPR*, 2022. 2

[35] Marc Pollefeys, Luc Van Gool, Maarten Vergauwen, Frank Verbiest, Kurt Cornelis, Jan Tops, and Reinhard Koch. Visual modeling with a hand-held camera. *IJCV*, 2004. 2

[36] Jie Ren, Wenteng Liang, Ran Yan, Luo Mai, Shiwen Liu, and Xiao Liu. Megba: A gpu-based distributed library for large-scale bundle adjustment. In *ECCV*, 2022. 9

[37] Jerome Revaud, Philippe Weinzaepfel, César Roberto de Souza, and Martin Humenberger. R2D2: repeatable and reliable detector and descriptor. In *NeurIPS*, 2019. 3, 5

[38] Richard Roberts, Sudipta N. Sinha, Richard Szeliski, and Drew Steedly. Structure from motion for scenes with large duplicate structures. *CVPR*, 2011. 2, 14

[39] I. Rocco, M. Cimpoi, R. Arandjelović, A. Torii, T. Pajdla, and J. Sivic. Neighbourhood consensus networks. *NeurIPS*, 2018. 3

[40] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary R. Bradski. Orb: An efficient alternative to sift or surf. *ICCV*, 2011. 3

[41] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. SuperGlue: Learning feature matching with graph neural networks. In *CVPR*, 2020. 1, 3, 5, 12

[42] Torsten Sattler, William P. Maddern, Carl Toft, Akihiko Torii, Lars Hammarstrand, Erik Stenborg, Daniel Safari, M. Okutomi, Marc Pollefeys, Josef Sivic, Fredrik Kahl, and Tomás Pajdla. Benchmarking 6dof outdoor visual localization in changing conditions. *CVPR*, 2018. 8, 13

[43] Torsten Sattler, Tobias Weyand, B. Leibe, and Leif P. Kobbelt. Image retrieval for image-based localization revisited. In *British Machine Vision Conference*, 2012. 8, 13

[44] Johannes L. Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *CVPR*, 2016. 1, 2, 3, 5, 6, 11, 12

[45] Thomas Schöps, Johannes L. Schönberger, S. Galliani, Torsten Sattler, Konrad Schindler, Marc Pollefeys, and Andreas Geiger. A multi-view stereo benchmark with high-resolution images and multi-camera videos. *CVPR*, 2017. 2, 6, 7

[46] Zehong Shen, Jiaming Sun, Yuang Wang, Xinying He, Hujun Bao, and Xiaowei Zhou. Semi-dense feature matching with transformers and its applications in multiple-view geometry. *TPAMI*, 2022. 3

[47] Noah Snavely, Steven M. Seitz, and Richard Szeliski. Photo tourism: exploring photo collections in 3d. *TOG*, 2006. 1, 5

[48] Jiaming Sun, Zehong Shen, Yuang Wang, Hujun Bao, and Xiaowei Zhou. Loftr: Detector-free local feature matching with transformers. *CVPR*, 2021. 2, 3, 6, 8, 12, 13

[49] Chengzhou Tang and Ping Tan. Ba-net: Dense bundle adjustment networks. In *ICLR*, 2019. 2

[50] Shitao Tang, Jiahui Zhang, Siyu Zhu, and Ping Tan. Quadtree attention for vision transformers. *ICLR*, 2022. 3

[51] Bill Triggs, Philip F. McLauchlan, Richard I. Hartley, and Andrew William Fitzgibbon. Bundle adjustment - a modern synthesis. In *Workshop on Vision Algorithms*, 1999. 5

[52] Prune Truong, Martin Danelljan, Luc Van Gool, and Radu Timofte. Learning accurate dense correspondences and when to trust them. *CVPR*, 2021. 3

[53] Michał Tyszkiewicz, Pascal Fua, and Eduard Trulls. Disk: Learning local features with policy gradient. *NeurIPS*, 2020. 12

[54] Ashish Vaswani, Noam M. Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017. 2, 3, 11

[55] Sudheendra Vijayanarasimhan, Susanna Ricco, Cordelia Schmid, Rahul Sukthankar, and Katerina Fragkiadaki. Sfm-net: Learning of structure and motion from video. *ArXiv:1704.07804*, 2017. 2

[56] Lei Wang, Lin-Lin Ge, Shan Luo, Zi Jun Yan, Zhaopeng Cui, and Jieqing Feng. Tc-sfm: Robust track-community-based structure-from-motion. *ArXiv*, abs/2206.05866, 2022. 14

[57] Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. In *NeurIPS*, 2021. 8, 14

[58] Qing Wang, Jiaming Zhang, Kailun Yang, Kunyu Peng, and Rainer Stiefelhagen. Matchformer: Interleaving attention in transformers for feature matching. In *ACCV*, 2022. 2, 3, 6, 13

[59] Qianqian Wang, Xiaowei Zhou, Bharath Hariharan, and Noah Snavely. Learning feature descriptors using camera pose supervision. In *ECCV*, 2020. 12

[60] Aji Resindra Widya, Akihiko Torii, and M. Okutomi. Structure from motion using dense cnn features with keypoint relocalization. *IPSJ Transactions on Computer Vision and Applications*, 2018. 2

[61] Kyle Wilson and Noah Snavely. Network principles for sfm: Disambiguating repeated structures with local context. *ICCV*, 2013. 2

[62] Kyle Wilson and Noah Snavely. Robust global translations with 1dsfm. In *ECCV*, 2014. 1, 2, 8, 14

[63] Changchang Wu. Towards linear-time incremental structure from motion. *3DV*, 2013. 5

[64] Changchang Wu, Sameer Agarwal, Brian Curless, and Steven M. Seitz. Multicore bundle adjustment. *CVPR*, 2011. 13

[65] Qingan Yan, Long Yang, Ling Zhang, and Chunxia Xiao. Distinguishing the indistinguishable: Exploring structural ambiguities via geodesic context. *CVPR*, pages 152–160, 2017. 14

[66] Christopher Zach, Arnold Irschara, and Horst Bischof. What can missing correspondences tell us about 3d structure and motion? *CVPR*, pages 1–8, 2008. 14

[67] Jason Y. Zhang, Deva Ramanan, and Shubham Tulsiani. RelPose: Predicting probabilistic relative rotation for single objects in the wild. In *ECCV*, 2022. 2

[68] Zichao Zhang, Torsten Sattler, and Davide Scaramuzza. Reference pose generation for long-term visual localization via learned features and view synthesis. *IJCV*, 2020. 8, 13

[69] Tinghui Zhou, Matthew A. Brown, Noah Snavely, and David G. Lowe. Unsupervised learning of depth and ego-motion from video. *CVPR*, 2017. 2

Supplementary Material

A. Discussion about Coarse SfM Accuracy

The coarse SfM phase is NOT to guarantee accuracy. Instead, it sacrifices accuracy for better completeness (registration rate). To this end, we use *quantized* detector-free matches, where the pixel threshold is relatively large in this phase (i.e., 4 pixels by default and 8 pixels for high-resolution images in ETH3D dataset) in both of the matching RANSAC and mapping. Therefore, a sufficient number of images can be registered with an acceptable pose error, which serves as the initialization of the refinement phase for higher pose accuracy. Nevertheless, Tab. 5 shows that the coarse SfM alone can achieve competitive accuracy compared with the state-of-the-art detector-based method on the IMC dataset, resulting in consistently superior results after refinement.

Type	Method	AUC@3°	AUC@5°	AUC@10°	AUC@20°
Detector-Based (Reference)	R2D2 + NN + PixSfM	31.41	41.80	54.65	64.90
	SP + SG + PixSfM	45.19	57.22	70.47	79.86
Detector-Free	Ours Coarse SfM (Quant. to 8×8)	39.88	52.42	67.16	78.14
	Ours full	**46.55**	**58.74**	**72.19**	**81.62**

Table 5. Results of coarse SfM's pose accuracy on IMC 2021 dataset.

B. Method Details

B.1. Reference View Selection in Feature Track Refinement

A track $\mathcal{T}_j = \{\mathbf{x}_k \in \mathbb{R}^2 | k = 1 : N_j\}$ is divided into reference and query views and features of the reference view are correlated on query views to search for multi-view correspondences. This strategy avoids exhaustively searching correspondences between every pair within a feature track, which is a complex topology [16] and is inefficient for refinement. Our criteria for selecting the reference view is to minimize the keypoint scale differences between the reference view and query views to improve the matchability. Concretely, we define the scale s_k of a 2D observation $\mathbf{x}_k$ in a feature track $\mathcal{T}_j$ as $s_k = {}^{d_k}/_{f_i}$, where f_i is focal length in intrinsic parameter $\mathbf{C}_i$ and d_k is the depth of $\mathbf{x}_k$ that is obtained by projecting its corresponding 3D point $\mathbf{P}_j$ with the current estimated pose $\boldsymbol{\xi}_i$. Then, the view with a medium scale across the track is selected as the reference view, whereas the rest views are query views.

B.2. Multi-View Feature Transformer

The backbone from S2DNet [19] is used as the CNN feature extractor. We interpolate and fuse the output features of adaption layers in S2DNet, which are at original image resolution and $1/8$ resolution respectively, to create a single feature map.

After the feature extraction, flattening, and concatenation, we use the Linear Transformer [26] to efficiently transform the reference feature $\tilde{\mathbf{F}}^r$ and query feature $\tilde{\mathbf{F}}^q$. Linear Transformer reduces the computational complexity of the Transformer [54] from $O(N^2)$ to $O(N)$ by substituting the exponential kernel with an alternative kernel function $\mathrm{sim}(Q, K) = \phi(Q) \cdot \phi(K)^{\mathrm{T}}$, where $\phi(\cdot) = \mathrm{elu}(\cdot) + 1$. Please refer to the original paper [54] for more details.

We denote a set of self- and cross-attention layers as an attention block:

$$\begin{cases} \tilde{\mathbf{F}}'^{r}_{(l+1)} = \mathrm{SelfAtten}(\tilde{\mathbf{F}}^{r}_{(l)}, \tilde{\mathbf{F}}^{r}_{(l)}) \ , \\ \tilde{\mathbf{F}}'^{q}_{(l+1)} = \mathrm{SelfAtten}(\tilde{\mathbf{F}}^{q}_{(l)}, \tilde{\mathbf{F}}^{q}_{(l)}) \ , \\ \tilde{\mathbf{F}}^{r}_{l+1}, \tilde{\mathbf{F}}^{q}_{l+1} = \mathrm{CrossAtten}(\tilde{\mathbf{F}}'^{r}_{(l+1)}, \tilde{\mathbf{F}}'^{q}_{(l+1)}) \ . \end{cases}$$

The indices of intermediate features are indicated by $\cdot_{(l)}$. $\tilde{\mathbf{F}}'$ represents an intermediate feature processed by a self-attention layer. Our attention module sequentially performs the attention block $n = 2$ times to transform the reference and query features.

The transformed features are reshaped into feature patches $\{\hat{\mathbf{F}}_k \in \mathbb{R}^{p \times p \times c}\}$ for multi-view feature correlation.

B.3. Geometry Refinement

With refined feature tracks, we perform bundle adjustment (BA) to optimize the scene geometry by reprojection error. The Cauchy function is used as the robust loss function $\rho(\cdot)$. For efficiency, we form the reduced camera system by the Schur Complement and then solve it by dense or sparse decomposition for small- or medium-scale scenes (number of images smaller than 500), respectively. On large-scale scenes, the reduced camera system is solved by Preconditioned Conjugate Gradients algorithm (PCG) [5, 8]. Moreover, to reduce the drift during BA, we select the farthest two views in the coarse model and fix the pose of one image and one translation DoF of the other image during BA, following [44].

B.4. Camera Parameter Estimation Details

Like COLMAP, our method does not require known intrinsic parameters, which can be inferred from image information (EXIF if available, otherwise, using max image edge size as initialization) and refined during BA, both in coarse SfM and refinement phase. All methods are not provided with intrinsics when evaluated on the IMC and ETH3D datasets. In the image registration phase, image poses are solved by the PnP algorithm first, followed by no-linear optimization. Then the poses will be optimized simultaneously with the point cloud in BA.

C. Training of Multi-View Feature Transformer

C.1. Ground Truth Generation

Our multi-view feature transformation module is trained on the MegaDepth [28], which is a large-scale outdoor dataset with 196 different scenes. To construct ground truth feature tracks for training, we first sample image bags for each scene and then project the grid-level points of a randomly selected reference view to other query views by depth maps.

Specifically, we sample 2000 image bags for each scene with a maximum of six images in each bag. The co-visibility extracted from the provided scene SfM model is used to sample image bags. We define the co-visibility ratio v of a sampled image bag as:

$$v = \frac{|\{\mathbf{P}\}_0 \cap \{\mathbf{P}\}_1 \cap \cdots \cap \{\mathbf{P}\}_i|}{\min(|\{\mathbf{P}\}_0|, |\{\mathbf{P}\}_1|, \cdots, |\{\mathbf{P}\}_i|)} \ ,$$

where $\{\mathbf{P}\}_i$ is the set of 3D points observed by image $\mathbf{I}_i$, and $|\cdot|$ is the operator that calculates the number of elements in a set. The image bags with a co-visibility ratio $0.02 < v < 0.6$ are kept for training. Moreover, the low-quality scenes reported by [53, 16] ('0000', '0002', '0011', '0020', '0033', '0050', '0103', '0105', '0143', '0176', '0177', '0265', '0366', '0474', '0860', '4541') and scenes that overlap with IMC test set ('0024', '0021', '0025', '1589', '0019', '0008', '0032', '0063') are removed from training.

After the image bag sampling, to construct ground truth feature tracks, we randomly select a reference image in the bag and project its grid-level points to other query views by the depth map, intrinsic parameters, and poses. Since the depth maps in MegaDepth are obtained by the MVS algorithm, inaccurate depth values exist. For accurate ground-truth multi-view matches, projection depth error and cycle projection error with strict thresholds are checked after the projection to filter inaccurate 2D observations in a feature track. The projection depth error e_d and cycle projection error e_c are defined as follows:

$$\begin{cases} e_d = \frac{\|\mathbf{D}_q(\mathbf{x}_{proj}) - d_{proj}\|}{\mathbf{D}_q(\mathbf{x}_{proj})} \ , \\ e_c = \|\mathbf{x}_r - \boldsymbol{\pi}_r \cdot \boldsymbol{\xi}_{r\to q}^{-1} \cdot \mathbf{D}_q(\mathbf{x}_{proj}) \cdot \boldsymbol{\pi}_q^{-1}(\mathbf{x}_{proj})\| \ , \end{cases}$$

$$\text{where } \ \mathbf{x}_{proj} = \boldsymbol{\pi}_q \cdot \boldsymbol{\xi}_{r\to q} \cdot \mathbf{D}_r(\mathbf{x}_r) \cdot \boldsymbol{\pi}_r^{-1}(\mathbf{x}_r) \ .$$

$\mathbf{x}_r$ is a sampled 2D point in reference view, $\mathbf{D}_{(\cdot)}$ is the depth map of reference or query view, $\boldsymbol{\pi}$ is the projection determined by intrinsic parameters, and $\boldsymbol{\xi}_{r\to q} = \boldsymbol{\xi}_q \cdot \boldsymbol{\xi}_r^{-1}$ is the relative pose between reference view and a query view. d_{proj} is the z value of 3D points in query view corresponding to $\mathbf{x}_{proj}$. A point in the query view is kept in the ground-truth feature track when projection depth error $e_d < 0.005$ and cycle projection error $e_c < 1px$.

C.2. Loss

The multi-view transformer module is trained by minimizing the average ℓ_2 loss on keypoint locations between the refined tracks and the ground-truth tracks. Following [59, 48], we make our loss uncertainty weighted with a variance term $\sigma^2(\mathbf{x})$:

$$\mathcal{L} = \frac{1}{N} \sum_{j\in n_t} \sum_{k\in n_v} \frac{1}{\sigma^2(\mathbf{x})} \|\mathbf{x} - \mathbf{x}_{gt}\|_2 \ ,$$

where n_t is the number of feature tracks, n_v is the number of query views in a track, and N is the total number of refined keypoints. $\sigma^2(\mathbf{x})$ is calculated by the trace of the heatmap's covariance matrix, which is detached during training to prevent the network from decreasing the loss by increasing the variance.

C.3. Training Details

The images are resized to have the longest edge of 840. The feature backbone is initialized by the pretrained weighted from S2DNet, and the attention blocks are randomly initialized. We use the AdamW optimizer to train the entire network, where the initial learning rate of backbone and attention blocks are 2×10^{-4} and 4×10^{-4}, respectively. The network training takes about 30 hours with a batch size of 8 on 8 NVIDIA V100 GPUs.

D. Texture-Poor SfM Dataset

In the proposed Texture-Poor SfM dataset, low-textured objects are placed on a texture-less plane, and video is captured surrounding each object. Each video is recorded at 30 fps for about 30 seconds in 1920×1440 resolution with per-frame poses and intrinsic parameters estimated by ARKit [1]. To stabilize the feature tracking and pose estimation in ARKit, textured markers are elaborately placed on the plane but far from the object. Then we annotate the 3D foreground region for later filter backgrounds that are discriminative and can reduce the difficulty of the dataset.

After the data capture, we perform a global BA [44] to further optimize camera poses estimated by ARKit and reduce the potential drift. We extract features [14], match [41] them, and then perform triangulation using the currently estimated poses. Then the global BA is performed to optimize poses. The placed discriminative markers can also facilitate feature extraction and matching in this phase. After the pose refinement, we crop out the background with salient features, and images after crop are resized to 840×840. With the refined poses, we project the annotated foreground regions to each image to filter backgrounds with salient features, where only cropped foreground images without markers are used for evaluation.

To impose larger viewpoint changes, we sample 60 subset image bags for each scene based on co-visibility, similar to

Sparse Det. & Matcher	Refinement	ETH3D Dataset			IMC2021 Dataset		
		AUC@3°	AUC@5°	AUC@10°	AUC@3°	AUC@5°	AUC@10°
SIFT + NN	PixSfM	26.94	39.01	42.19	25.54	34.80	46.73
	Ours	**29.28**	**41.76**	**45.12**	**26.77**	**36.18**	**48.32**
R2D2 + NN	PixSfM	43.58	62.09	66.89	31.41	41.80	54.65
	Ours	**46.84**	**64.31**	**68.75**	**32.35**	**42.83**	**55.65**
SP + SG	PixSfM	50.82	68.52	72.86	45.19	57.22	70.47
	Ours	**52.66**	**70.15**	**74.85**	**45.43**	**57.75**	**71.57**

Table 6. Comparison of sparse local features accompanied with our refinement and PixSfM on ETH3D dataset and IMC2021 dataset.

the IMC 2021 dataset [25]. Each bag contains either 5, 10, or 20 images.

E. Experiments

E.1. Datasets

On the IMC dataset, the validation set is already separated. We follow their protocol and use all eight test scenes for evaluation, and use validation scenes *Sacre Coeur*, *Saint Peter's Square*, and *Reichstag* for tuning hyperparameters. Images are resized so that the longest edge dimension is equal to 1200 pixels for all methods. As for the ETH3D dataset, the images are resized to have a maximum edge dimension of 1600 pixels for all methods. On the proposed Texture-Poor SfM dataset, we use randomly selected three scenes as validation sets for tuning hyper-parameters and the remaining scenes for evaluation. Note that due to the image crop in the post-process of the Texture-Poor SfM dataset, the principle points of intrinsic parameters are not in the image center. To avoid the degeneration of estimating principle points in SfM, all of the methods are provided with known intrinsic parameters, which are kept fixed during SfM.

E.2. Metric of Multi-View Camera Pose Estimation

The AUC of pose error at different thresholds is used as the metric to evaluate the accuracy of estimated multi-view poses, following the IMC benchmark [25] and PixSfM [30]. This metric converts N multi-view poses to C_N^2 pair-wise relative transformation, which is invariant to the difference of coordinate system between reconstructed and ground-truth poses. The pose error is defined as the maximum angular error in rotation and translation.

On the IMC dataset and Texture-Poor SfM dataset, we use pose error at $(3°, 5°, 10°)$ thresholds. Since the ETH3D dataset has high-resolution images and accurate ground truth calibration, we further report a more strict $1°$ threshold to evaluate the capability of highly accurate pose estimation.

E.3. Sparse Features with Our Refinement.

Our refinement module can also be used to refine SfM models reconstructed by sparse feature detecting and matching to further bring pose improvement. Pose accuracy is evaluated on the ETH3D dataset and IMC 2021 dataset. Results shown in Tab. 6 demonstrate that our framework can consistently outperform PixSfM when accompanied by the same sparse detectors and matchers.

		ETH3D Dataset			IMC (*Mount Rushmore*)		
		AUC@1°	AUC@3°	AUC@5°	AUC@3°	AUC@5°	AUC@10°
LoFTR [48]	No Refine	30.88	58.90	68.06	21.26	32.09	47.96
	Iter 1	57.20	74.61	78.85	29.69	41.42	56.61
	Iter 2	**59.12**	**75.59**	**79.53**	**32.35**	**43.92**	**58.91**
AspanTrans. [10]	No Refine	28.41	55.87	65.40	19.04	29.02	44.26
	Iter 1	55.48	72.84	77.07	29.06	40.29	55.11
	Iter 2	**57.23**	**73.71**	**77.70**	**31.77**	**43.23**	**57.79**
MatchFormer [58]	No Refine	26.30	53.95	63.48	8.48	15.46	28.47
	Iter 1	54.30	71.29	75.52	24.25	35.10	49.80
	Iter 2	**56.70**	**73.00**	**76.84**	**29.31**	**39.66**	**53.32**

Table 7. **Ablation Study of Refinement Iterations.** On the ETH3D dataset and scene *Mount Rushmore* in the IMC dataset, we quantitatively evaluate the impact of the number of refinement iterations. The AUC of pose error at different thresholds is reported.

	IMC (*Mount Rushmore*)		
	AUC@3°	AUC@5°	AUC@10°
Medium Scale	**32.35**	**43.92**	**58.91**
Smallest Scale	30.63	42.11	56.99
Largest Scale	31.94	42.98	57.54
Random Selection	31.21	42.45	57.24

Table 8. **Ablation Study of Reference View Selection.** On the scene *Mount Rushmore* in the IMC dataset, we evaluate the impact of the reference view selection strategies. The AUC of pose error at different thresholds is reported.

E.4. More Ablation Studies

In this part, we validate the effectiveness of our refinement pipeline by the multi-view pose metric on multiple datasets.

The results in Tab. 7 indicate that our iterative refinement pipeline consistently improves pose accuracy for various detector-free matchers across different datasets. As shown in Tab 8, using other reference view selection strategies, including using a view with the smallest or largest scale, and randomly selecting a reference view for each track, will reduce the final pose accuracy.

E.5. Efficiency on Large-Scale Scenes

Experiments of the large-scale scenes are conducted on a server using 16 CPU cores (Intel Xeon Gold 6146) and four NVIDIA V100 GPUs. The subset images are uniformly sampled from the Aachen v1.1 dataset [42, 68, 43]. For each image, the top 20 most covisible images determined by image retrieval [6] are used for matching, where images are resized so that the longest edge equals 1200. To refine the large-scale scene with a large number of 3D points caused by semi-dense matchers, we perform refinement only once and use four GPUs for parallelized multi-view matching. As for geometry refinement, we use multi-core bundle adjustment [64] to leverage multiple CPU cores.

Compare with detector-based pipeline. We show the overall running time comparison with detector-based

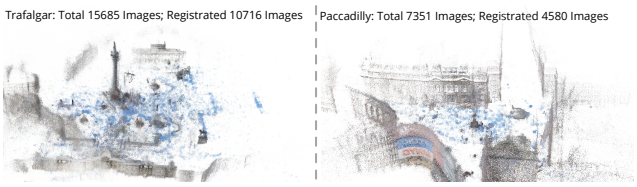

Figure 9. Reconstruction of scenes in the 1DSfM dataset.

Method	Trafalgar (15685 Img)		Piccadilly (7351)		Vienna Cathe. (6288)		Union Squere (5960)	
	Num. Img.	Time	Num. Img.	Time	Num. Img.	Time	Num. Img.	Time
COLMAP	7001	**3.2h**	2950	**1.8h**	1139	**1.1h**	1026	**0.9h**
COLMAP (SP + SG)	9482	6.8h	3488	3.4h	1764	2.7h	1835	2.5h
Ours	**10716**	19.7h	**4580**	11.2h	**2436**	8.8h	**2026**	8.2h

Table 9. Comparsion with detector-based methods on the 1DSfM dataset.

pipelines on the four largest scenes in the 1DSfM [62] dataset in Tab. 9. On large-scale scenes, our framework is slower than detector-based pipeline with sparse features. On the one hand, detector-free matching is inherently slower than sparse matching. Moreover, due to a significant number of matches produced by detector-free matchers, the incremental mapping phase is also slower. However, on the scene with images collected from the internet and with large viewpoint and illumination changes, our framework can registrate significantly more images compared with sparse methods. These results also show that our framework applies to large-scale scenes (with more than 15000 images). Visualizations of reconstruction are shown in Fig. 9

F. Failure Cases

As shown in Fig. 10, on the scenes [38] with strong duplicated structures, our framework may yield error registrations, which may come with the side effects of detector-free matchers that are capable of matching texture-poor scenes. Many previous methods [66, 38, 13, 65, 56] have focused on solving scene disambiguations, which can be further integrated into our framework to alleviate this problem. With the Missing Correspondence Disambiguation [66, 13], our framework can successfully reconstruct the ambiguous scenes, as shown in Fig. 10 (row 3).

G. Real-World Scenes "Deep Sea" and "Moon Surface"

In this section, we introduce the data collection and running of challenging real-world scenes shown in the main paper's Fig. **1** and the demo video. The *Deep Sea* scene is from sequence 5 of the Aqualoc dataset [17]. This sequence was chosen because it contains a texture-poor section that presents significant challenges. The *Moon Surface* sequence is taken from an internet video that has low-texture and repetitive patterns, as well as severe motion blur.

For the scene *Deep Sea*, we use the image retrieval [6]

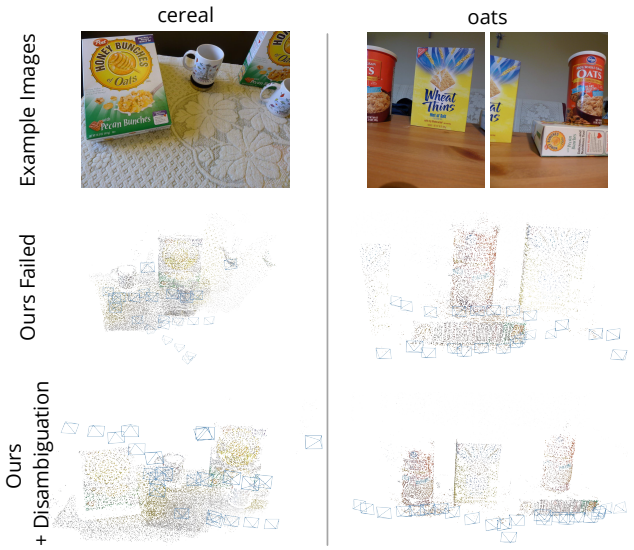

Figure 10. Failure cases of our framework on scenes [38] with strong duplicated structures. With the help of disambiguation method [13], our framework can correctly reconstruct ambiguous scenes.

to select the top 30 most covisible images of each image for matching. As for the *Moon Surface*, we use sequential matching to match an image with its nearest 20 frames and run our framework.

H. Dense Reconstruction

To demonstrate the application of our framework that can provide accurate poses for dense reconstruction on texture-poor scenes, we run our framework on the scene *Headphone Box* and *Eyeglass Box*. The sequences are downsampled to 6fps for running our detector-free SfM framework, which provides the recovered poses for neural surface reconstruction method NeuS [57] to reconstruct the scene. We manually filter the background reconstruction and only keep the object of interest for visualization.