This CVPR paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

Dense-SfM: Structure from Motion with Dense Consistent Matching

JongMin Lee
Seoul National University
jongmin.cv@gmail.com

Sungjoo Yoo
Seoul National University
sungjoo.yoo@gmail.com

Abstract

We present Dense-SfM, a novel Structure from Motion (SfM) framework designed for dense and accurate 3D reconstruction from multi-view images. Sparse keypoint matching, which traditional SfM methods often rely on, limits both accuracy and point density, especially in texture-less areas. Dense-SfM addresses this limitation by integrating dense matching with a Gaussian Splatting (GS) based track extension which gives more consistent, longer feature tracks. To further improve reconstruction accuracy, Dense-SfM is equipped with a multi-view kernelized matching module leveraging transformer and Gaussian Process architectures, for robust track refinement across multi-views. Evaluations on the ETH3D and Texture-Poor SfM datasets show that Dense-SfM offers significant improvements in accuracy and density over state-of-the-art methods. Project page: https://icetea-cv.github.io/densesfm/.

1. Introduction

Structure from Motion (SfM) is a fundamental task in computer vision that enables the reconstruction of 3D structures from multi-view images of a scene. It has been a widely researched problem, focusing on reconstructing accurate camera poses and 3D point cloud of a scene, benefiting a variety of downstream tasks such as visual localization [19, 43, 46, 48, 58], AR/VR [39, 70], multi-view stereo [20, 50, 52] and neural rendering [27, 34, 72]. Over the years, several well-established methods have been developed to address the SfM pipeline [36, 40, 49, 55, 68].

Recently, there has been an increasing demand for dense and accurate 3D reconstructions in various 3D vision tasks, including dense reconstruction [17, 65], object pose estimation [5, 23, 57] and neural rendering [27, 72]. However, existing detector-based methods have limitations in achieving sufficient accuracy and density of 3D points, especially in texture-less regions where obtaining matchable keypoints is challenging, limiting the performance of downstream tasks. With the advent of deep learning and attention mechanism [62], semi-dense and dense matching methods [6, 14, 15, 56, 66] have been introduced, enabling the reliable matches even in low-texture areas and producing denser match results than classical detector-based methods. This advancement has led to improved performance across many tasks, including homography and relative pose estimation [2, 7, 29]. However, due to their pairwise matching approach, the matching methods usually yield fragmented

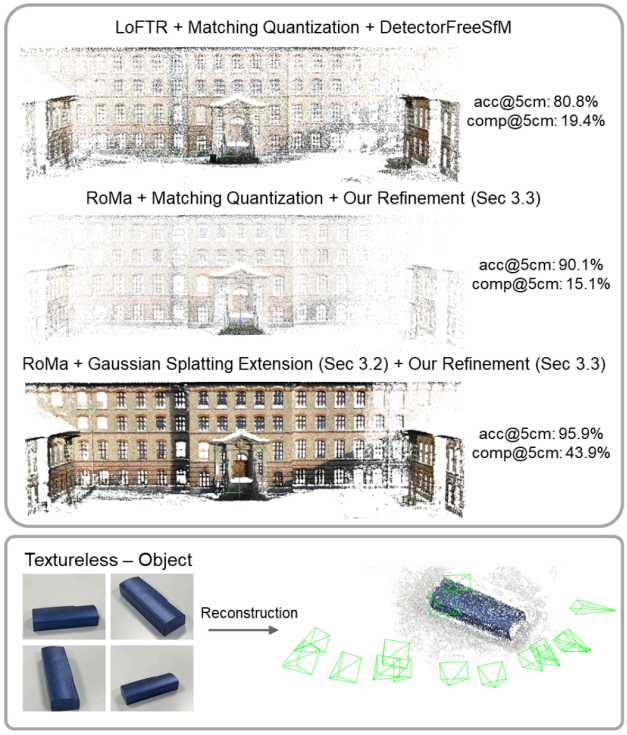


Figure 1. **Comparison of SfM results between quantized-matching and our matching.** To solve the fragmentary track problem, DFSfM [24] uses quantized matching, which can compromise the accuracy and completeness of 3D points due to quantization. In contrast, our Dense-SfM pipeline leverages Gaussian Splatting to solve the problem, offering more accurate and dense 3D points. Thanks to dense matching, our pipeline can also be applied to texture-less objects, further improving the quality of 3D points and pose estimation.

tracks[1] as mentioned in Detector-Free SfM (DFSfM) [24], making direct integration with existing SfM pipelines challenging. To solve this inconsistency issue, DFSfM applied quantization to the matched keypoints, and then refined the multi-view tracks to obtain accurate camera pose and point cloud. While the quantization strategy can increase track length and consistency, enhancing the multi-view refinement results, it can significantly limit the number of matches and accuracy, reducing the quality of the resulting SfM model, as exemplified in Fig. 1.

In this paper, we propose a novel SfM framework that leverages dense matching while addressing the limitations of pairwise matching methods. As shown in Fig. 1, our approach reconstructs a dense and accurate 3D point cloud, avoiding the inherent limitations of quantized matching. Specifically, our framework first performs dense matching using a dense matcher, *e.g.*, DKM [14] or RoMa [15], to construct an initial SfM model, where most tracks are initially fragmentary, *i.e.*, of short (mostly a pair of images) lengths. Next, we initialize small 3D Gaussians based on the initial SfM's point cloud, integrating attributes like color and position. These Gaussians are then optimized and densified through Gaussian Splatting (GS) [27, 72], filling in the scene while keeping the initial Gaussians' (SfM points') positions fixed. We then infer the visibility of each initial SfM point per image by rendering through the ray from camera poses. In case that the 3D point is visible on the image, we include the image into the track of the 3D point thereby increasing the track length—a process we refer to as *Track Extension*. Note that *Track Extension* enables us to resolve the problem of short tracks in pairwise matching. Lastly, we perform our proposed SfM refinement pipeline, which includes a multi-view kernelized matching module and geometric bundle adjustment, to further optimize the camera poses and 3D points. Our track refinement module is mainly built on two architectures, transformer and Gaussian Process [41], which incorporate both feature-based and positional information, resulting in notable performance gains over existing refinement modules as demonstrated in our ablation study (Tab. 3).

We evaluate our Dense-SfM framework on the ETH3D, Texture-Poor SfM and Image Matching Challenge (IMC) 2021 datasets, where it outperforms state-of-the-art SfM systems across multiple metrics. Thanks to the dense matching coupled with track extension via Gaussian Splatting, as well as our novel track refinement module, our framework achieves accurate camera pose estimations and dense 3D reconstructions.

Contributions:

- A novel SfM framework optimized for dense matchers, capable of reconstructing highly accurate and dense 3D models, even in texture-less regions.
- A track extension method that, based on Gaussian Splatting, helps find new images on which 3D point is visible, thereby increasing track length of the 3D point.
- A new track refinement pipeline that leverages transformer and Gaussian Process architectures to enhance the consistency of feature tracks and refine the overall reconstruction quality.
- Extensive evaluations across diverse metrics demonstrate that our framework achieves state-of-the-art performance.

[1] A keypoint is assigned a set of 2D coordinates, called feature track or track in short, on neighbor images where the keypoint is matched.

2. Related Work

Image Matching. Establishing 2D correspondences through feature matching is a fundamental component for SfM and visual SLAM. Traditional feature matching typically involves i) detecting keypoints [8, 11, 31, 42, 44, 45], ii) computing descriptors [13, 32, 33, 37, 59, 61] and iii) performing descriptor matching [18, 47, 71]. Despite its efficiency, it struggles in challenging scenarios such as low-textured regions, where keypoint detection is unreliable, and repetitive patterns, where matching often fails—leading to suboptimal SfM results.

To address these limitations, more recent methods bypass the need for keypoint detection and instead directly perform semi-dense [6, 56, 66] or dense [14, 15] matching between image pairs. These approaches benefit from global receptive fields provided by transformers [62] or correlations computed from CNN or ViT features [9, 38], which allows them to achieve better performance in texture-poor areas where traditional methods often fail.

However, semi-dense and dense matching often results in fragmentary feature tracks, making it difficult to apply on SfM pipeline [53]. To solve this, DFSfM [24] introduced quantization matching that merges nearby subpixel matches into grid nodes, improving track consistency. Yet, this approach can compromise the accuracy of the initial camera poses and point clouds, heavily relying on refinement modules and reducing point cloud density.

In contrast, our framework leverages Gaussian Splatting to assess visibility of 3D points. By projecting 3D points onto new images on which the points are visible, we obtain longer and consistent tracks without sacrificing the initial matching accuracy, which leads to improved refinement performance and, finally, better SfM results.

Structure From Motion. Feature correspondence-based SfM methods have been extensively studied [3, 16, 35]. Incremental SfM [1, 55, 68] has become a widely adopted approach, with COLMAP [49] recognized as one of the most popular frameworks.

Recently, differentiable SfM methods [4, 10, 54, 63, 64, 67] have aimed to simplify or entirely replace the traditional

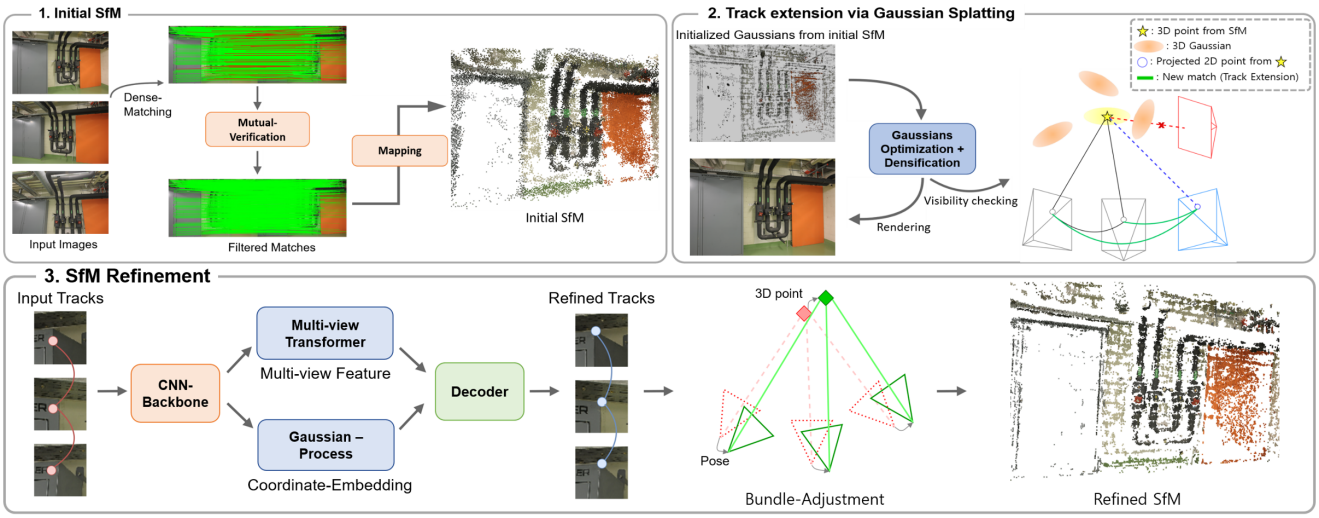


Figure 2. **Pipeline Overview.** From a set of images, we construct an initial SfM model using dense two-view matching, filtering unreliable matches through mutual verification. To extend track length, we project 3D points onto additional images, using a visibility filter based on Gaussian Splatting. We then refine these extended tracks with our track refinement module and perform geometric bundle adjustment to improve the accuracy of SfM model.

SfM pipeline. VGGSfM [64] manages all SfM components as fully differentiable, enabling end-to-end training. DUSt3R [67] predicts pairwise pointmaps and refines them through optimization-based global alignment. Its successors, MASt3R [28] and MASt3R-SfM [10], improve pixel matching for localization by predicting additional per-pixel features and leveraging its encoder for fast image retrieval, leading to a newly proposed SfM pipeline.

Multi-view Refinement. Inconsistent feature points across multiple views can impact the accuracy of reconstructed point cloud and camera poses in SfM. Hence, precise multi-view correspondences are essential for accurately recovering 3D structures and camera parameters in SfM pipelines.

To address this, several previous methods have proposed multi-view keypoint refinement models based on flow [12] or dense features [30], which lead to significant improvements in SfM accuracy. PatchFlow [12] refines keypoint locations using a geometric cost constrained by local flow between two views, followed by global optimization using a multi-view graph of relative feature geometry constraints. Similarly, PixSfM [30] refines keypoint locations through feature-metric keypoint adjustment by optimizing a direct cost over discriminative dense feature maps along the tracks. DFSfM [24], introduces a multi-view matching module that iteratively refines the tracks obtained from the COLMAP framework. The module extracts local patches from each image and feeds them into a transformer-based network to refine the keypoint's location by enforcing multi-view consistency.

Building on these recent advancements, we propose a novel refinement module that leverages both feature and positional information to refine the keypoint coordinates within the tracks. Instead of relying on statistical methods to select the refined tracks, our module fully regresses and learns the confidence (certainty) per track, resulting in more accurate refinement. As demonstrated in our ablation study (Tab. 3), our approach outperforms the existing refinement methods.

3. Method

An overview of our Dense-SfM framework is shown in Fig. 2. Given N images $\{I_i\}$ of a scene, our goal is to estimate the camera poses and reconstruct the 3D point cloud representing the scene.

We design our framework as a three-stage pipeline. First, we construct an initial SfM model using correspondences between image pairs from a two-view dense matcher (*e.g.*, DKM [14] and RoMa [15]). Next, we extend the track length of each 3D point by projecting it onto new images where it is visible, verifying visibility through Gaussian Splatting [27]. Finally, we perform iterative SfM refinement to enhance the accuracy of poses and point cloud.

3.1. Initial SfM via Dense Matching

Our framework employs dense feature matching between image pairs, unlike traditional SfM pipelines [49] which rely on sparse keypoints [8, 31]. By employing dense matching, our approach improves the reconstruction of texture-poor scenes and enables the registration of dense 3D points, which is challenging for conventional methods.

For efficiency, we first sample matches from the dense matching using non-max suppression [8], based on per-

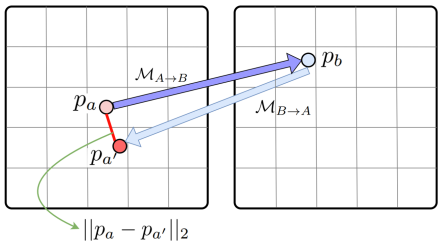


Figure 3. **Bidirectional verification on two-view dense matching.** The match result p_b from $\mathcal{M}_{A\to B}$ is re-used as input to $\mathcal{M}_{B\to A}$ to estimate $p_{a'}$. We then compute the distance between p_a and $p_{a'}$, where a smaller distance indicates higher reliability of the match between p_a and p_b.

pixel confidence scores from the dense matching model's output. Then, we sort out reliable matches by a method called mutual verification using bi-directional dense matching, similar to the mutual-nearest neighbor approach commonly used in detector-based methods.

As shown in Fig. 3, from the bi-directional dense matching results $M_{A\to B}$ and $M_{B\to A}$ between images A and B, we define the pixel coordinates p_b and $p_{a'}$ as

$$p_b = \mathcal{M}_{A\to B}[p_a], \quad p_{a'} = \mathcal{M}_{B\to A}[p_b] \tag{1}$$

where $p_{a'}$ is sampled using bilinear-interpolation.

To ensure more reliable matches, we filter out correspondences using a binary mask, which is obtained with the distance from p_a to $p_{a'}$ as follows:

$$B = [||p_a - p_{a'}||_2 \leq \epsilon_p] \tag{2}$$

Using these filtered matches, we construct the initial SfM by triangulating 3D points through dense two-view fine matching. If camera poses are not provided for the reconstruction task, we first perform match quantization [24] and incremental mapping [49] to obtain coarse poses. We then triangulate 3D points using raw matching with the coarse pose and refine both the point cloud and camera poses through the refinement process, as explained in Sec. 3.3.

3.2. Track Extension via Gaussian Splatting

In the initial SfM, most of the tracks are short, *i.e.* consisting of mostly two images, due to the fragmentary nature of feature tracks in pair-wise matching, which can limit the performance of track refinement, as shown in DFSfM [24].

Longer tracks, which offer more information on 3D geometry, could improve the track refinement performance. Thus, we generate new matching pairs on the existing matches for consistent multi-view matches, extending the length of track. We achieve this by projecting 3D points (triangulated by two-view matches) onto additional images on which, if successfully projected, the point is considered as visible from the camera pose as illustrated (by the blue dashed line) in Fig. 2.

To determine the visibility of each 3D point from a camera pose, we apply a visibility filter using Gaussian Splatting (GS) [27, 72], which supports fast training and rendering. We initialize a GS model with the 3D points obtained in the initial SfM. Specifically, we treat them as small 3D Gaussians setting each mean $\boldsymbol{\mu} \in \mathbb{R}^3$ to the 3D point coordinates, rotation R to the identity matrix, opacity α to 1, and scale S based on the policy from Splatam [26], which uses focal length and depth information. These Gaussians are optimized and densified to cover the entire scene, while keeping the initial Gaussian's parameters fixed. The details of initialization and training process is described in supplementary material.

After training the GS model, we utilize it for evaluating the visibility M of all SfM points (represented as small Gaussians) from each camera pose using the following formula:

$$M = \Big[\max_{r\in R}\{\alpha_{\text{SfM}} \prod_{j=1}^{N_{SfM}-1} (1-\alpha_j)\} > \epsilon_v\Big] \tag{3}$$

where R is the set of rays from the camera center (of a camera pose) for rendering the image, α_{SfM} is the opacity of Gaussian representing the SfM point (set to 1), N_{SfM} the number of Gaussians along the ray r from the camera center to the SfM point, and α_j the opacity of j-th intermediate Gaussian. If the highest visibility score among the pixels rendered from the Gaussian representing the SfM point is larger than ϵ_v, we consider the SfM point, *i.e.* N_{SfM}-th Gaussian, visible from the view of the camera pose. For each 3D point, we extend its track by adding its projected 2D point(s) onto the visible view(s). To do that, we project a 3D point P_j onto the visible image using camera pose $\{R_i, t_i\}$ and intrinsic matrix C_i as follows:

$$p_{ij} = \Pi\left(R_i P_j + t_i, C_i\right) \tag{4}$$

Then, we verify the correspondence between the newly obtained 2D point p_{ij} and its associated original 2D keypoint with geometric verification [49] using initial SfM's camera poses. After verified, we add the new 2D point to the track of the 3D point P_j, which extends the length of P_j's track.

Note that we project the 3D point onto the images that were not involved in the initial triangulation (*i.e.*, establishing a new one-to-one correspondence of 2D-3D matches per image as illustrated with green lines in Fig. 2 (Stage 2). As will be demonstrated in our experiments (Tab. 3), the proposed *Track extension* enables the subsequent track refinement to produce more refined tracks, which leads to more accurate 3D points in the SfM model.

3.3. Iterative SfM Refinement

Using the extended tracks, we refine the initial SfM model to obtain improved camera poses and point cloud. Specifically, we iteratively improve the accuracy of feature tracks

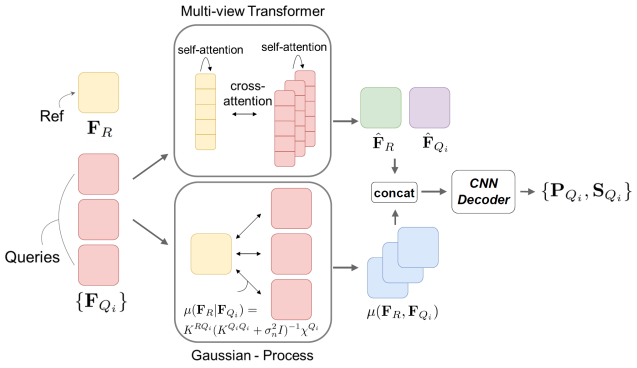


Figure 4. **Multi-view kernelized matching architecture.** The feature maps from the reference and query views are processed in two ways: first, by a multi-view transformer, applying self and cross-attention and secondly, by a Gaussian Process that computes the posterior mean conditioned on the feature of each query view. All results are concatenated and fed into a CNN decoder, yielding probability distribution of coordinates for refinement $\mathbf{P}_{Q_i}$ and corresponding confidence score $\mathbf{S}_{Q_i}$ for each query view.

by employing our multi-view kernelized matching module alongside bundle adjustment, jointly optimizing camera poses and point cloud.

3.3.1. Feature Track Refinement

A feature track (or track) $\mathcal{T}_j = \{\mathbf{x}_k \in \mathbb{R}^2 | k = 1 : N_j\}$ represents a set of 2D keypoint locations on multi-view (N_j) images that correspond to a 3D point P_j. In our proposed method of feature track refinement, each track goes through a multi-view kernelized matching architecture (Fig. 4) and the regression of refined coordinates (Fig. 5). The matching architecture produces two outputs, a probability distribution of coordinate per pixel P_{Q_i} and confidence score S_{Q_i} per view. Then, the regression from these outputs produces a refined track by adjusting the keypoint locations.

Multi-view kernelized matching. Following DFSfM, we split a track into the keypoint of a reference view and those of query views (views from the track except the reference view) with a policy of median depth value. Then, we feed image patches centered at each keypoint into a CNN feature extractor, obtaining feature maps $\mathbf{F}_R, \mathbf{F}_{Q_i} \in \mathbb{R}^{p\times p\times c}$, where $\mathbf{F}_R$ is from the reference view and $\mathbf{F}_{Q_i}$'s are from query views. As shown in Fig. 4, our multi-view kernelized matching architecture takes as input these feature maps of reference and query views.

Our proposed matching architecture consists of two paths, feature-based and coordinate-embedding-based ones. On the feature-based path (denoted with Multi-view Transformer in Fig. 4), we utilize a local feature transformer to obtain the transformed feature map $\hat{\mathbf{F}}_R$ and $\hat{\mathbf{F}}_{Q_i}$'s through self and cross-attention between $\mathbf{F}_R$ and $\mathbf{F}_Q$'s. On the coordinate-embedding-based path (denoted with Gaussian-

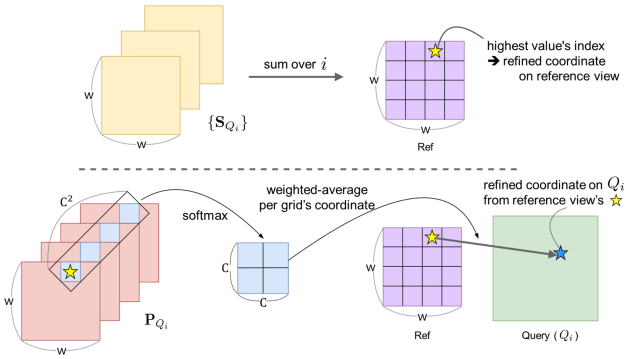


Figure 5. **Regression of refined coordinate. Top: Selection of refined coordinate on the reference view:** From the set of S_{Q_i} values, representing the correspondence scores from the reference view to each query view Q_i, we sum over S_{Q_i}'s and select the pixel with the highest score as the refined coordinate for the reference view. **Bottom: Regression of refined coordinate on each query view:** We apply a channel-wise softmax to obtain a probability distribution of coordinates for each pixel, then calculate a weighted average to determine the correspondence from the reference view to query view Q_i.

Process in Fig. 4), we employ a Gaussian Process [41] to utilize the positional information, deriving a coordinate embedding feature $\mu(\mathbf{F}_R|\mathbf{F}_{Q_i})$ for each query view. Following DKM [14], we use an exponential cosine similarity kernel:

$$k(f_A, f_B) = \exp(-\tau)\exp\left(\tau\frac{\langle f_A, f_B\rangle}{\sqrt{\langle f_A, f_A\rangle\langle f_B, f_B\rangle + \varepsilon}}\right), \tag{5}$$

where f_A and f_B each is a feature vector from $\mathbf{F}_A$ and $\mathbf{F}_B$, respectively. From this kernel function, we build kernel matrices $K^{\mathcal{R}\mathcal{Q}_i}, K^{\mathcal{Q}_i\mathcal{Q}_i} \in \mathbb{R}^{p^2\times p^2}$ to obtain posterior mean $\mu(\mathbf{F}_R|\mathbf{F}_{Q_i})$ as follows:

$$\mu(\mathbf{F}_R|\mathbf{F}_{Q_i}) = K^{RQ_i}(K^{Q_iQ_i} + \sigma_n^2 I)^{-1}\chi^{Q_i} \tag{6}$$

where $\chi^{\mathcal{Q}_i}$ is the Fourier positional encoding [34] of F_{Q_i}'s pixel coordinate. Finally, we concatenate the transformed feature maps $\hat{\mathbf{F}}_R$, $\hat{\mathbf{F}}_{Q_i}$ and coordinate embedding $\mu(\mathbf{F}_R|\mathbf{F}_{Q_i})$, and feed them into a CNN decoder D_θ:

$$P_{Q_i}, S_{Q_i} = D_\theta(\hat{\mathbf{F}}_R \oplus \hat{\mathbf{F}}_{Q_i} \oplus \mu(\mathbf{F}_R|\mathbf{F}_{Q_i})) \tag{7}$$

where we obtain a probability distribution of coordinate per pixel $P_{Q_i} \in \mathbb{R}^{C^2\times w\times w}$ and confidence score $S_{Q_i} \in \mathbb{R}^{w\times w}$, C^2 is the number of distribution's anchors, and $w \times w$ a search boundary for refined coordinates.

Regression of keypoint's location. Fig. 5 illustrates how the refined track is obtained. First, we obtain the refined coordinate of keypoint on the reference view. From the output of multi-view kernelized matching, $\{S_{Q_i}\}$, we estimate the confidence value of each pixel as the candidate of refined coordinate on the reference view. As shown in Fig. 5 (top),

we select, as the refined coordinate on the reference view, the pixel with the highest value, *i.e*. the highest confidence from $\sum_i S_{Q_i}$.

Then, as shown in Fig. 5 (bottom), from the refined coordinate on the reference view (calculated above), we estimate the refined coordinate of keypoint for each query view Q_i. For each patch of P_{Q_i} of size $w \times w$, we apply softmax across channels from the pixel of refined coordinate for the reference view, to obtain a probability distribution over a $C \times C$ grid. The refined coordinate for the query view Q_i is then computed as a weighted average, where each pixel's probability serves as its weight. By multiplying each probability by its corresponding pixel coordinate and summing these weighted values, we derive a refined coordinate for query view Q_i. Finally, we construct the final refined track $\mathcal{T}_j^*$, consisting of the refined coordinates for the reference view and all query views.

Training setup and loss function. We set up training as in DFSfM [24] using MegaDepth dataset [29]. We use image bags sampled from the SfM model provided in the dataset, in which ground truth tracks are derived from the given camera pose and MVS depth map. For training, we perturb the ground truth 2D locations of tracks in the query views with random noise, providing these as input to our refinement module. Unlike DFSfM, which relies on statistics (variance) as uncertainty measure, we directly learn confidence score through a neural network. Consequently, our loss function is not simply formulated as an L_2 loss to minimize the distance between the refined tracks and the ground truth tracks. Instead, we jointly predict and learn the confidence $S_{Q_i} > 0$ for each query view, and include it in the regression loss as follows:

$$\mathcal{L} = \frac{1}{N} \sum_{j \in n_t} \sum_{i \in n_j} s_{Q_i} \cdot \|p_{Q_i} - p_{gt}\|_2 - \alpha \log s_{Q_i}. \quad (8)$$

where n_t is the number of feature tracks, n_j the number of query views in a track j, N the total number of refined keypoints on all the query views, p_{Q_i} and p_{gt} the learned 2D coordinate from Q_i (as illustrated in Fig. 5) and the ground truth 2D coordinate, respectively, and s_{Q_i} the confidence score of prediction p_{Q_i}, which is obtained, from S_{Q_i}, at the pixel of the reference view's ground truth keypoint. The second term in the loss is used to avoid being underconfident, where we set α as 20.

3.3.2. Bundle Adjustment

Using the refined feature tracks, we iteratively perform geometric bundle adjustment (BA) [60] to refine the intrinsic parameters $\{C_i\}$, camera poses $\{\boldsymbol{\xi}_i\}$, and 3D points' coordinates $\{P_j\}$. Formally, we minimize the reprojection error E defined as:

$$E = \sum_j \sum_{x_k^* \in \mathcal{T}_j^*} \rho\left(\|\pi\left(\boldsymbol{\xi}_i \cdot P_j, C_i\right) - x_k^*\|_2^2\right) \ ,$$

where $\pi(\cdot)$ projects points to image plane using intrinsic parameter C_i, and $\rho(\cdot)$ is a robust loss function [21]. We then filter outliers [49, 68] by rejecting points that exceed the maximum reprojection error threshold ϵ_f. After performing BA and outlier filtering, the refined SfM can be passed to the next iteration of feature track refinement and BA.

4. Experiments

4.1. Datasets

We use the ETH3D [51], Texture-Poor SfM datasets [24], and Image Matching Challenge (IMC) 2021 dataset [25], each presenting unique challenges across different types of scenes. The ETH3D dataset includes indoor and outdoor scenes with sparsely captured, high-resolution images, and ground truth poses calibrated using Lidar. The Texture-Poor SfM dataset focuses on object-centric scenes with low texture, comprising subsampled image sets. We use all test scenes from the IMC Phototourism dataset, evaluating them with subsampled image bags. The main challenge of this dataset lies in its sparse views, accompanied by significant variations in viewpoint and illumination.

4.2. Implementation Details

For efficiency, we apply non-max suppression [8] with a radius of 4 pixels for ETH3D and 2 pixels for Texture-Poor SfM and IMC dataset. For mutual verification, we filter out matches with a distance exceeding $\epsilon_p = 3$px in Eq. 2.

To determine the visibility of Gaussians (SfM points) from a camera pose, we set the visibility threshold ϵ_v to 0.5 in Eq. 3. Details of training and implementation of Gaussian Splatting are provided in the supplementary material.

We use S2Dnet [18] as a CNN backbone for our refinement architecture, and adopt the multi-view transformer architecture from DFSfM [24]. To ensure a fair comparison, we apply the same refinement parameters as in DFSfM, including a local patch size ($p = 15$) from the feature maps ($\mathbf{F}_R$ and $\mathbf{F}_Q$) obtained from the CNN backbone, and a search boundary for reference view's refined keypoints set to $w = 7$. The number of distribution anchors C^2 is set to 7^2. For the CNN decoder in Fig. 4, we use a series of residual blocks [22] and channel attention blocks [69]. During bundle adjustment, matches with reprojection errors exceeding $\epsilon_f = 3$px are discarded. The total SfM refinement process is iterated twice. More implementation details are reported in the supplementary material.

4.3. 3D Triangulation

Accurately triangulating scene point clouds, given known camera poses and intrinsics, is an important task in SfM. In this section, we evaluate the accuracy and completeness of the reconstructed point clouds.

Type	Method	Accuracy (%)			Completeness (%)		
		1cm	2cm	5cm	1cm	2cm	5cm
Detector-Based	SIFT [31] + NN + PixSfM [30]	76.18	85.60	93.16	0.17	0.71	3.29
	D2Net [11] + NN + PixSfM	74.75	83.81	91.98	0.83	2.69	8.95
	R2D2 [42] + NN + PixSfM	74.12	84.49	91.98	0.43	1.58	6.71
	SP [8] + SG [47] + PixSfM	79.01	87.04	93.80	0.75	2.77	11.28
Semi-Dense Matching	LoFTR [56] + PixSfM	74.42	84.08	92.63	2.91	9.39	27.31
	LoFTR + DFSfM [24]	80.38	89.01	95.83	3.73	11.07	29.54
	AspanTrans. [6] + DFSfM	77.63	87.40	95.02	3.97	12.18	32.42
	MatchFormer [66] + DFSfM	79.86	88.51	95.48	3.76	11.06	29.05
Dense-Matching	RoMa + DFSfM	79.32	88.42	95.82	3.13	9.79	29.10
	DKM + Ours	84.05	91.96	97.39	5.95	13.95	31.03
	RoMa + Ours	**84.79**	**92.62**	**97.77**	**7.38**	**17.06**	**36.35**
Deep	VGGSfM [64]	80.62	89.49	96.52	4.52	13.11	33.96
Point-Based	Mast3r-SfM [10]	42.02	59.74	80.58	2.05	8.76	31.73
	Mast3r-SfM + Our Refinement (Sec. 3.3)	77.22	86.47	94.05	2.40	8.26	27.17

Table 1. **Results of 3D Triangulation.** Our method is compared with the baselines on the ETH3D [51] dataset using accuracy and completeness metrics with different thresholds.

Evaluation. We use the ETH3D training set, comprising 13 indoor and outdoor scenes with millimeter-accurate dense point clouds as ground truth. Following the protocol in PixSfM [30] and DFSfM [24], we triangulate 3D points using fixed camera poses and intrinsics, then evaluate the resulting point clouds for accuracy and completeness using the ETH3D benchmark [51]. The results are reported at varying distance thresholds (1cm, 2cm, 5cm), averaged across all scenes. Consistent with the benchmark setup, accuracy and completeness are measured by the proportion of triangulated points that fall within the specified distances from the ground truth points.

Results. The results are shown in Tab. 1. While there is a trade-off between accuracy and completeness, our Dense-SfM framework achieves superior performances on both metrics. Compared to the state-of-the-art baseline LoFTR + DFSfM, our pipeline with RoMa achieves higher accuracy and completeness, thanks to employing dense matching and our proposed pipeline. Also, compared to the RoMa + DFSfM, where we apply quantization on dense matching and refine the tracks as in DFSfM, our method achieves higher performance in terms of both accuracy and completeness, highlighting the effectiveness of our SfM framework. More results with detector-based and semi-dense matching are reported in the supplementary material.

4.4. Multi-View Camera Pose Estimation

In this section, we evaluate the recovered multi-view poses.

Evaluation protocols. For the evaluation, we use a subset of ETH3D training and test dataset, which includes 22 indoor and outdoor scenes, the Texture-Poor SfM dataset, and the test scenes of IMC 2021 dataset. We evaluate the accuracy of the estimated multi-view poses using the AUC of pose error at various thresholds. For the evaluation of IMC 2021 dataset, due to the high occlusion and appearance changes in many test scenes, directly applying track extension via GS is challenging. Instead, we adopt a quantization method [24] on the matching results, focusing on comparing our multi-view kernelized matching module with DFSfM's refinement module. Specifically, we apply quantization to the matched keypoints with a radius of 4, and then use our refinement module to refine the quantized keypoints.

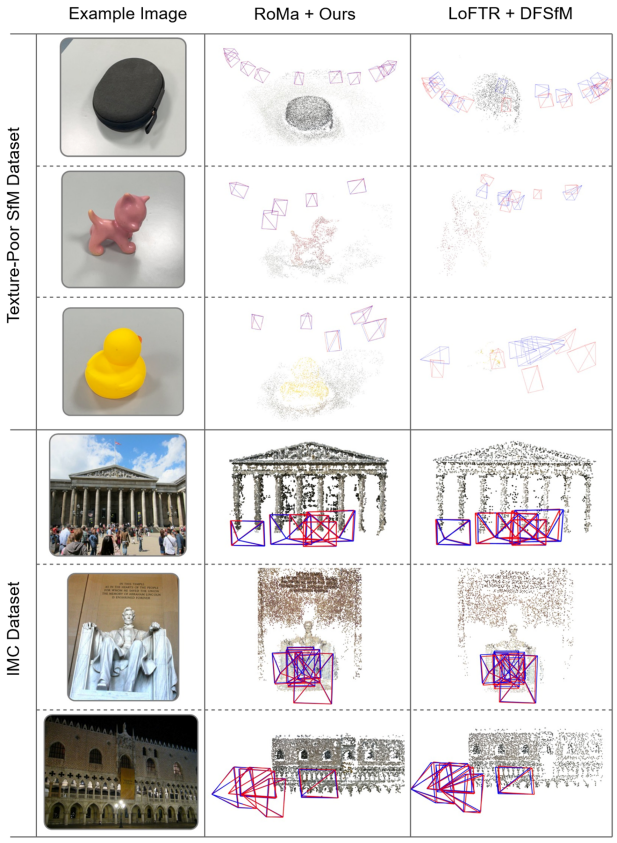


Figure 6. **Qualitative Results on Texture-Poor SfM and IMC dataset.** Our method with the dense matcher RoMa is qualitatively compared with the baseline LoFTR + DFSfM. The red cameras are ground truth poses while the blue cameras are recovered poses.

Results. As shown in Tab. 2, our Dense-SfM framework outperforms existing baselines over all datasets. On the ETH3D dataset, our framework with dense matching achieves the highest multi-view pose accuracy, with a particularly large improvement at the highest threshold.

As shown in Fig. 6, our framework successfully reconstructs dense and accurate 3D points thanks to dense matching and our pipeline, even in challenging low-textured regions with viewpoint changes, leading to much higher pose estimation accuracy.

4.5. Ablation Studies

We conduct several experiments to evaluate the key components of our pipeline, including the matching design used as input for SfM and the architectural choices of the refinement module, on the ETH3D dataset using triangulation metrics.

Type	Method	ETH3D Dataset			Texture-Poor SfM Dataset			IMC Dataset		
		AUC@1°	AUC@3°	AUC@5°	AUC@3°	AUC@5°	AUC@10°	AUC@3°	AUC@5°	AUC@10°
Detector-Based	COLMAP (SIFT+NN) [49]	26.71	38.86	42.14	2.87	3.85	4.95	23.58	23.66	44.79
	SIFT [31] + NN + PixSfM [30]	26.94	39.01	42.19	3.13	4.08	5.09	25.54	34.80	46.73
	D2Net [11] + NN + PixSfM	34.50	49.77	53.58	1.54	2.63	4.54	8.91	12.26	16.79
	R2D2 [42] + NN + PixSfM	43.58	62.09	66.89	3.79	5.51	7.84	31.41	41.80	54.65
	SP [8] + SG [47] + PixSfM	50.82	68.52	72.86	14.00	19.23	24.55	45.19	57.22	70.47
Semi-Dense Matching	LoFTR [56] + PixSfM	54.35	73.97	78.86	20.66	30.49	42.01	44.06	56.16	69.61
	LoFTR + DFSfM [24]	59.12	75.59	79.53	26.07	35.77	45.43	46.55	58.74	72.19
	AspanTrans. [6] + DFSfM	57.23	73.71	77.70	25.78	35.69	45.64	46.79	59.01	72.50
	MatchFormer [66] + DFSfM	56.70	73.00	76.84	26.90	37.57	48.55	45.83	57.88	71.22
Dense Matching	RoMa + DFSfM	57.97	77.02	81.59	34.28	50.01	69.01	47.43	59.84	73.19
	DKM + Ours	59.04	77.73	82.20	41.19	57.26	72.95	**48.65**	**61.09**	**74.37**
	RoMa + Ours	**60.92**	**78.41**	**82.63**	**49.94**	**66.23**	**81.41**	48.48	60.79	73.90
Deep	VGGSfM [64]	28.09	44.41	49.91	1.86	6.26	21.10	45.23	58.89	73.92
Point-Based	Mast3r-SfM [10]	35.85	58.46	65.03	43.46	61.51	78.97	42.26	54.53	67.97
	Mast3r-SfM + Our Refinement	37.50	59.18	65.48	48.03	64.48	80.31	44.98	57.09	70.09

Table 2. **Results of Multi-View Camera Pose Estimation.** Our framework is compared with detector-based, detector-free and dense-matching baselines by the AUC of pose error at different thresholds. **Bold** and underline indicate the best and second-best results.

	Design Choice	Accu. (%)		Complete. (%)	
		1cm	2cm	1cm	2cm
(1) Matching designs	**Full model**	84.79	92.62	7.38	17.06
	w/o Mutual Verification	84.03	92.06	8.14	18.68
	w/o GS Extension	77.45	87.80	**9.45**	**21.64**
	Quantized Matching (r=4)	82.82	90.75	3.53	10.62
(2) Refinement designs	**Full model**	84.79	92.62	7.38	17.06
	Multi-view Trans. [24]	80.59	90.04	6.04	15.2
	w/o Gaussian Process	72.47	84.72	4.78	12.59
	w/o Multi-view Trans.	83.39	91.63	7.04	16.49
	w/o ref. confidence selection	83.81	92.15	8.83	19.85
(3) Number of iterations	No refine.	69.77	82.59	8.61	20.4
	1 iter	81.12	90.52	7.62	17.86
	2 iter	84.79	92.6	7.38	17.06
	3 iter	**85.01**	**92.75**	6.74	15.91

Table 3. **Ablation Studies.** On the ETH3D dataset, we quantitatively evaluate the impact of design choice of matching and refinement module, and the number of iterations of refinement. The reported triangulation accuracy and completeness are averaged across all scenes.

Matching Designs. Tab. 3 (1) demonstrates the benefits of our proposed matching filtering (mutual verification) and track extension via Gaussian Splatting (GS). Compared to the configurations without mutual verification, where all matches are used for reconstructing the initial SfM, incorporating mutual verification improves accuracy, with a trade-off in completeness due to filtering. Moreover, extending match pairs through GS significantly improves accuracy, which shows that using more views for multi-view kernelized matching contribute to higher performance of accuracy.

For a comparison with DFSfM [24], which applies quantization to matched keypoints to address the fragmented track problem, we replaced our GS track extension with their quantization approach, using a rounding ratio of $r = 4$. Results indicate that our GS track extension not only maintains completeness by avoiding the limitations of quantization but also achieves superior accuracy.

Refinement designs. Table 3 (2) shows the benefit of our proposed multi-view kernelized matching's components in Fig. 4 and confidence-based track selection in Fig. 5.

We first compare our refinement module to the multi-view refinement module in DFSfM [24], by substituting our refinement module with it. The result shows that our refinement module consistently outperforms the existing approach, achieving higher accuracy and completeness.

Next, we evaluate the contributions of each component in our refinement architecture (Fig. 4) by testing configurations where either the multi-view transformer or Gaussian Process is removed. The results show that our full model, which integrates both components, achieves the highest performance in accuracy and completeness, confirming that combining feature-based and positional information yields more precise refinement. Furthermore, our strategy of selecting a final refined track based on predicted confidence in the reference view further enhances accuracy.

Impact of iterations. Table 3 (3) reports the results after each refinement iteration. The initial SfM point cloud is less accurate due to reliance solely on initial two-view matching. As the iterations progress, the accuracy of the SfM point cloud gets improved. However, further refinement beyond two iterations gives little gains. So, we limit the refinement to two iterations for efficiency.

5. Conclusion

This paper introduces a novel Dense-SfM framework aimed at reconstructing dense point cloud and estimating camera poses from a set of images. Unlike conventional SfM systems that often rely on initial 2D matches, our framework leverages Gaussian Splatting to generate new matching pairs, producing longer tracks that significantly enhance both the accuracy and density of the reconstructed point cloud. Extensive evaluations on diverse datasets show that our framework consistently outperforms existing SfM baselines across multiple metrics.

6. Acknowledgement

This research was supported in part by Samsung Advanced Institute of Technology (SAIT), the NAVER-Intel Co-Lab, Inter-university Semiconductor Research Center (ISRC), and Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) [NO.RS-2021-II211343, Artificial Intelligence Graduate School Program (Seoul National University)].

References

[1] Sameer Agarwal, Noah Snavely, Ian Simon, Steven M. Seitz, and Richard Szeliski. Building rome in a day. In *2009 IEEE 12th International Conference on Computer Vision*, pages 72–79, 2009. 2

[2] Vassileios Balntas, Karel Lenc, Andrea Vedaldi, and Krystian Mikolajczyk. Hpatches: A benchmark and evaluation of handcrafted and learned local descriptors, 2017. 1

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

[4] Eric Brachmann, Jamie Wynn, Shuai Chen, Tommaso Cavallari, Áron Monszpart, Daniyar Turmukhambetov, and Victor Adrian Prisacariu. Scene coordinate reconstruction: Posing of image collections via incremental learning of a relocalizer. In *ECCV*, 2024. 2

[5] Pedro Castro and Tae-Kyun Kim. Posematcher: One-shot 6d object pose estimation by deep feature matching. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*, pages 2148–2157, 2023. 1

[6] 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. 1, 2, 7, 8

[7] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas A. Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. *CoRR*, abs/1702.04405, 2017. 1

[8] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superpoint: Self-supervised interest point detection and description, 2018. 2, 3, 6, 7, 8

[9] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *CoRR*, abs/2010.11929, 2020. 2

[10] Bardienus Duisterhof, Lojze Zust, Philippe Weinzaepfel, Vincent Leroy, Yohann Cabon, and Jerome Revaud. Mast3r-sfm: a fully-integrated solution for unconstrained structure-from-motion, 2024. 2, 3, 7, 8

[11] 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, 2019. 2, 7, 8

[12] Mihai Dusmanu, Johannes L. Schönberger, and Marc Pollefeys. Multi-view optimization of local feature geometry. *CoRR*, abs/2003.08348, 2020. 3

[13] Patrick Ebel, Anastasiia Mishchuk, Kwang Moo Yi, Pascal Fua, and Eduard Trulls. Beyond cartesian representations for local descriptors. *CoRR*, abs/1908.05547, 2019. 2

[14] Johan Edstedt, Mårten Wadenbäck, and Michael Felsberg. Deep kernelized dense geometric matching. *CoRR*, abs/2202.00667, 2022. 1, 2, 3, 5

[15] Johan Edstedt, Qiyu Sun, Georg Bökman, Mårten Wadenbäck, and Michael Felsberg. RoMa: Robust Dense Feature Matching. *IEEE Conference on Computer Vision and Pattern Recognition*, 2024. 1, 2, 3

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

[17] Qiancheng Fu, Qingshan Xu, Yew-Soon Ong, and Wenbing Tao. Geo-neus: Geometry-consistent neural implicit surfaces learning for multi-view reconstruction. *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. 1

[18] Hugo Germain, Guillaume Bourmaud, and Vincent Lepetit. S2dnet: Learning accurate correspondences for sparse-to-dense feature matching. *CoRR*, abs/2004.01673, 2020. 2, 6

[19] Hugo Germain, Vincent Lepetit, and Guillaume Bourmaud. Neural reprojection error: Merging feature learning and camera pose estimation. *CoRR*, abs/2103.07153, 2021. 1

[20] Michael Goesele, Brian Curless, and Steven M. Seitz. Multi-view stereo revisited. *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, 2:2402–2409, 2006. 1

[21] Frank R Hampel, Elvezio M Ronchetti, Peter J Rousseeuw, and Werner A Stahel. *Robust statistics: the approach based on influence functions*. Wiley, 1986. 6

[22] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015. 6

[23] 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 *Advances in Neural Information Processing Systems*, 2022. 1

[24] Xingyi He, Jiaming Sun, Yifan Wang, Sida Peng, Qixing Huang, Hujun Bao, and Xiaowei Zhou. Detector-free structure from motion. *CVPR*, 2024. 1, 2, 3, 4, 6, 7, 8

[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. *CoRR*, abs/2003.01587, 2020. 6

[26] 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. 4

[27] 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. 1, 2, 3, 4

[28] Vincent Leroy, Yohann Cabon, and Jerome Revaud. Grounding image matching in 3d with mast3r, 2024. 3

[29] Zhengqi Li and Noah Snavely. Megadepth: Learning single-view depth prediction from internet photos. In *Computer Vision and Pattern Recognition (CVPR)*, 2018. 1, 6

[30] Philipp Lindenberger, Paul-Edouard Sarlin, Viktor Larsson, and Marc Pollefeys. Pixel-Perfect Structure-from-Motion with Featuremetric Refinement. In *ICCV*, 2021. 3, 7, 8

[31] David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60:91–110, 2004. 2, 3, 7, 8

[32] Zixin Luo, Tianwei Shen, Lei Zhou, Siyu Zhu, Runze Zhang, Yao Yao, Tian Fang, and Long Quan. Geodesc: Learning local descriptors by integrating geometry constraints. *CoRR*, abs/1807.06294, 2018. 2

[33] Zixin Luo, Lei Zhou, Xuyang Bai, Hongkai Chen, Jiahui Zhang, Yao Yao, Shiwei Li, Tian Fang, and Long Quan. Aslfeat: Learning local features of accurate shape and localization. *CoRR*, abs/2003.10071, 2020. 2

[34] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis, 2020. 1, 5

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

[36] Pierre Moulon, Pascal Monasse, Romuald Perrot, and Renaud Marlet. OpenMVG: Open multiple view geometry. In *International Workshop on Reproducible Research in Pattern Recognition*, pages 60–74. Springer, 2016. 1

[37] Yuki Ono, Eduard Trulls, Pascal Fua, and Kwang Moo Yi. Lf-net: Learning local features from images. *CoRR*, abs/1805.09662, 2018. 2

[38] Maxime Oquab, Timothée Darcet, Theo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Russell Howes, Po-Yao Huang, Hu Xu, Vasu Sharma, Shang-Wen Li, Wojciech Galuba, Mike Rabbat, Mido Assran, Nicolas Ballas, Gabriel Synnaeve, Ishan Misra, Herve Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. Dinov2: Learning robust visual features without supervision, 2023. 2

[39] Felix Ott, Lucas Heublein, David Rügamer, Bernd Bischl, and Christopher Mutschler. Fusing structure from motion and simulation-augmented pose regression from optical flow for challenging indoor environments. *Journal of Visual Communication and Image Representation*, 103:104256, 2024. 1

[40] Linfei Pan, Daniel Barath, Marc Pollefeys, and Johannes Lutz Schönberger. Global Structure-from-Motion Revisited. In *European Conference on Computer Vision (ECCV)*, 2024. 1

[41] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2005. 2, 5

[42] Jerome Revaud, Philippe Weinzaepfel, César De Souza, Noe Pion, Gabriela Csurka, Yohann Cabon, and Martin Humenberger. R2d2: Repeatable and reliable detector and descriptor, 2019. 2, 7, 8

[43] Jerome Revaud, Yohann Cabon, Romain Brégier, JongMin Lee, and Philippe Weinzaepfel. Sacreg: Scene-agnostic coordinate regression for visual localization, 2023. 1

[44] Edward Rosten and Tom Drummond. Machine learning for high-speed corner detection. In *Computer Vision – ECCV 2006*, pages 430–443, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg. 2

[45] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. Orb: An efficient alternative to sift or surf. In *2011 International Conference on Computer Vision*, pages 2564–2571, 2011. 2

[46] Paul-Edouard Sarlin, Cesar Cadena, Roland Siegwart, and Marcin Dymczyk. From coarse to fine: Robust hierarchical localization at large scale. *CoRR*, abs/1812.03506, 2018. 1

[47] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superglue: Learning feature matching with graph neural networks, 2020. 2, 7, 8

[48] Torsten Sattler, Akihiko Torii, Josef Sivic, Marc Pollefeys, Hajime Taira, Masatoshi Okutomi, and Tomas Pajdla. Are large-scale 3d models really necessary for accurate visual localization? In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6175–6184, 2017. 1

[49] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 1, 2, 3, 4, 6, 8

[50] Johannes Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise view selection for unstructured multi-view stereo. 2016. 1

[51] 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. 6, 7

[52] Shuhan Shen. Accurate multiple view 3d reconstruction using patch-based stereo for large-scale scenes. *IEEE Transactions on Image Processing*, 22(5):1901–1914, 2013. 1

[53] 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. 2

[54] Cameron Smith, David Charatan, Ayush Tewari, and Vincent Sitzmann. Flowmap: High-quality camera poses, intrinsics, and depth via gradient descent. In *arXiv*, 2024. 2

[55] Noah Snavely, Steven M. Seitz, and Richard Szeliski. Photo tourism: exploring photo collections in 3d. *Seminal Graphics Papers: Pushing the Boundaries, Volume 2*, 2006. 1, 2

[56] Jiaming Sun, Zehong Shen, Yuang Wang, Hujun Bao, and Xiaowei Zhou. LoFTR: Detector-free local feature matching with transformers. *CVPR*, 2021. 1, 2, 7, 8

[57] Jiaming Sun, Zihao Wang, Siyu Zhang, Xingyi He, Hongcheng Zhao, Guofeng Zhang, and Xiaowei Zhou. OnePose: One-shot object pose estimation without CAD models. *CVPR*, 2022. 1

[58] Linus Svärm, Olof Enqvist, Magnus Oskarsson, and Fredrik Kahl. Accurate localization and pose estimation for large 3d models. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 532–539, 2014. 1

[59] Yurun Tian, Xin Yu, Bin Fan, Fuchao Wu, Huub Heijnen, and Vassileios Balntas. Sosnet: Second order similarity regularization for local descriptor learning. *CoRR*, abs/1904.05019, 2019. 2

[60] Bill Triggs, Philip F. McLauchlan, Richard I. Hartley, and Andrew W. Fitzgibbon. Bundle adjustment — a modern synthesis. In *Vision Algorithms: Theory and Practice*, pages 298–372, Berlin, Heidelberg, 2000. Springer Berlin Heidelberg. 6

[61] Michal J. Tyszkiewicz, Pascal Fua, and Eduard Trulls. DISK: learning local features with policy gradient. *CoRR*, abs/2006.13566, 2020. 2

[62] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017. 1, 2

[63] Hengyi Wang and Lourdes Agapito. 3d reconstruction with spatial memory. *arXiv preprint arXiv:2408.16061*, 2024. 2

[64] Jianyuan Wang, Nikita Karaev, Christian Rupprecht, and David Novotny. Visual geometry grounded deep structure from motion, 2023. 2, 3, 7, 8

[65] 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. *NeurIPS*, 2021. 1

[66] Qing Wang, Jiaming Zhang, Kailun Yang, Kunyu Peng, and Rainer Stiefelhagen. Matchformer: Interleaving attention in transformers for feature matching. In *ACCV*, 2022. 1, 2, 7, 8

[67] Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jerome Revaud. Dust3r: Geometric 3d vision made easy. In *CVPR*, 2024. 2, 3

[68] Changchang Wu. Towards linear-time incremental structure from motion. In *2013 International Conference on 3D Vision - 3DV 2013*, pages 127–134, 2013. 1, 2, 6

[69] Changqian Yu, Jingbo Wang, Chao Peng, Changxin Gao, Gang Yu, and Nong Sang. Learning a discriminative feature network for semantic segmentation. *CoRR*, abs/1804.09337, 2018. 6

[70] Harry Zhang, Jeffrey Ichnowski, Yahav Avigal, Joseph Gonzales, Ion Stoica, and Ken Goldberg. Dex-net ar: Distributed deep grasp planning using a commodity cellphone and augmented reality app. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 552–558, 2020. 1

[71] Jiahui Zhang, Dawei Sun, Zixin Luo, Anbang Yao, Lei Zhou, Tianwei Shen, Yurong Chen, Long Quan, and Hongen Liao. Learning two-view correspondences and geometry using order-aware network. *International Conference on Computer Vision (ICCV)*, 2019. 2

[72] Zehao Zhu, Zhiwen Fan, Yifan Jiang, and Zhangyang Wang. Fsgs: Real-time few-shot view synthesis using gaussian splatting, 2024. 1, 2, 4