FC-GNN: Recovering Reliable and Accurate Correspondences from Interferences

Haobo Xu[1,2*] Jun Zhou[1,2*] Hua Yang[1,2†] Renjie Pan[1,2] Cunyan Li[1,2]

[1]Institute of Image Communication and Network Engineering, Shanghai Jiao Tong University
[2]Shanghai Key Lab of Digital Media Processing and Transmission

Abstract

Finding correspondences between images is essential for many computer vision tasks and sparse matching pipelines have been popular for decades. However, matching noise within and between images, along with inconsistent keypoint detection, frequently degrades the matching performance. We review these problems and thus propose: 1) a novel and unified Filtering and Calibrating (FC) approach that jointly rejects outliers and optimizes inliers, and 2) leveraging both the matching context and the underlying image texture to remove matching uncertainties. Under the guidance of the above innovations, we construct Filtering and Calibrating Graph Neural Network (FC-GNN), which follows the FC approach to recover reliable and accurate correspondences from various interferences. FC-GNN conducts an effectively combined inference of contextual and local information through careful embedding and multiple information aggregations, predicting confidence scores and calibration offsets for the input correspondences to jointly filter out outliers and improve pixel-level matching accuracy. Moreover, we exploit the local coherence of matches to perform inference on local graphs, thereby reducing computational complexity. Overall, FC-GNN operates at lightning speed and can greatly boost the performance of diverse matching pipelines across various tasks, showcasing the immense potential of such approaches to become standard and pivotal components of image matching. Code is avaiable at https://github.com/xuy123456/fcgnn.

1. Introduction

Finding correspondences between images is a critical component of various computer vision tasks [9, 19, 51].A typical sparse matching pipeline usually detects and describes keypoints first, then performs matching and rejects the outliers. Despite numerous efforts to establish reliable and accurate correspondences, various factors continue to pose significant challenges.

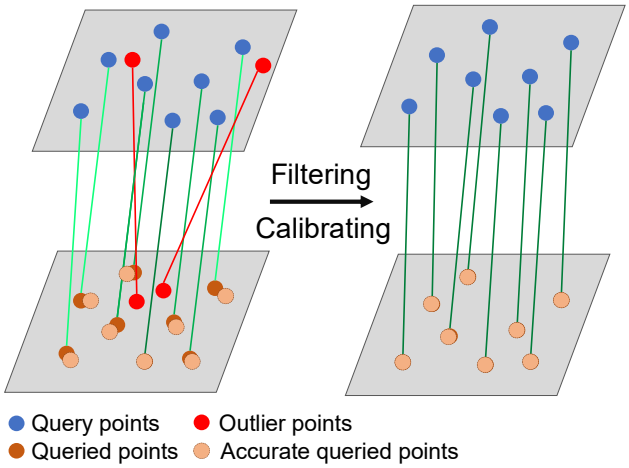


Figure 1. **Joint filtering and calibrating.** FC-GNN divides the optimization of correspondence set into two parts, namely 1) filtering out correspondences with large errors and 2) utilizing calibration offsets to correct the remaining correspondences. Such approch ensures the reliability of the correspondence set and greatly improves the pixel-level matching accuracy.

Matching noise within and between images is a critical factor that can impair matching performance. Simple matching methods, such as nearest neighbor (NN) search, relying solely on the performance of descriptors, are easy to be disturbed and often not reliable. To address this issue, some methods [4, 48, 49] are proposed to filter out outliers in correspondence sets or predict element weights to guide geometric estimation. While helpful in noisy matches, these methods cannot change the intrinsic quality of individual elements within the set, and thus still heavily rely on the quality of the initial matches. To overcome this limitation, some methods [5, 36, 39] try to build a reliable matching set starting from keypoints and descriptors, utilizing attention-based graph neural networks to jointly find correspondences and reject non-matchable points. The attention-based in-

*equal contribution
†corresponding author

ference is also used for matching between dense features [6, 40, 45] and also proven effective. Although useful, computational cost should be considered carefully in such methods, since full-attention complexity grows quadratically with the number of input elements [43].

The accuracy of keypoint detection is another important factor in sparse matching pipelines. In fact, numerous works have been devoted to improving the accuracy of keypoint detection [2, 10, 20, 27, 29, 31, 35, 42], but they are still limited by noise interference within and between images. Another approach is to refine the keypoints. Some two-view dense matching pipelines [6, 34, 40, 45, 52] have refinement modules because they follow a coarse-to-fine matching process, but these modules mostly depend on the matching pipelines themselves, making them not suitable to optimize various matching sets. Among them, Patch2Pix [52] is relatively flexible, which can regress pixel-level matches from given local regions and reject outliers with confidence scores. However, its refinement and outlier rejection are both based on local regions, which limits its performance.

In this work, we propose a novel **F**iltering and **C**alibrating approach and try to leverage both local image texture and matching context to recover reliable and accurate correspondences from interferences. We emphasize that, essentially, the matching process can be viewed as a procedure that is affected by different types of disturbances, resulting in varying degrees of errors. Therefore, for given correspondences, we propose to jointly reject outliers with large errors and calibrate inaccurate inliers with relatively small errors. We achieve this by designing FC-GNN, an attention-based graph neural network, which takes correspondences and their associated image patch pairs as inputs, leveraging both contextual and local information through self-attention layers. The network outputs confidence scores and calibration offsets, where the former helps reject outliers and the latter improves pixel-level matching accuracy. To avoid the high computational cost of global attention, we focus only on k-neighbors of each match during attention operating and gradually expand the receptive field through multiple cascaded attention layers.

The main contributions can be summarized as follows:

1. We present a novel approach for optimizing correspondences through joint filtering and calibrating, which shows the potential to serve as a standard component for matching optimization.
2. We design FC-GNN, an attention-based graph neural network that can fastly utilize both contextual and local information to optimize the correspondences.
3. Extensive experimental findings demonstrate the significant enhancement of our model in various matching pipelines across diverse tasks.

2. Related Work

Local feature matching. The goal of local feature matching is to establish correspondences between images. Sparse matching (detector-based methods) has long dominated the field of image matching. Traditional methods such as SIFT [27], SURF [3], and ORB [35] are still widely used today. In recent years, Learning-based keypoint detectors and descriptors, such as [2, 10, 12, 17, 20, 28–32, 37, 42, 44, 50], strive to find more robust and accurately localized keypoints and stronger feature representations. Different approaches also focus on filtering noisy matches [48, 49]. However, feature detection can be seen as pre-sampling the feature matching space, which may result in information loss. Therefore, some methods such as [6, 14, 21, 33, 34, 40, 45] directly establish matches on dense features. This leads to better pixel-level matching performance but also means increased computational complexity.

Matching and outlier rejection. For given keypoints and descriptors, simple matching methods like nearest neighbor (NN) search or NN + ratio test are susceptible to noise interference. Establishing and filtering matches based on context information is a useful approach. [4] predict the probability to weight matches, which can be used in RANSAC [15]. Some works [41, 48, 49] use networks to predict whether a match is an outlier. Some methods aim to learn a whole matching function [5, 16, 36, 39], in which S2DNet [16] matches between sparse features of one image and dense features of another, and [5, 24, 36, 39, 47] utilize graph neural networks, taking keypoints and corresponding descriptors as inputs to output high-confidence matches.

Matching refinement. Matching refinement is commonly used in dense matching and exists as part of the matching pipeline. In Sparse-NCNet [34], a relocalization module is used to obtain higher-precision matching through higher-resolution feature maps. DRC-Net [21] improves the performance of [34] by weighting fine-resolution scores with filtered coarse-resolution scores. LoFTR [40] uses a correlation-based approach to get fine matches, which is also used in [6, 45]. The problem with the above refinement methods is that they are part of the dense matching pipeline and it is not suitable to optimize any given match set. At the same time, their refinements are usually local. For sparse matching, in SfM pipelines, [13, 23] uses dense features to adjust feature points before geometric estimation, but dense features require a larger amount of calculation and memory consumption, and it has no filtering function. Patch2Pix [52] uses the given patch-level matching proposal to perform regression to obtain the final pixel-level proposal, and uses different confidence scores for filter matching. It can also be used for sparse matching. However, this way of regression and filtering is based on local features, which limits its performance.

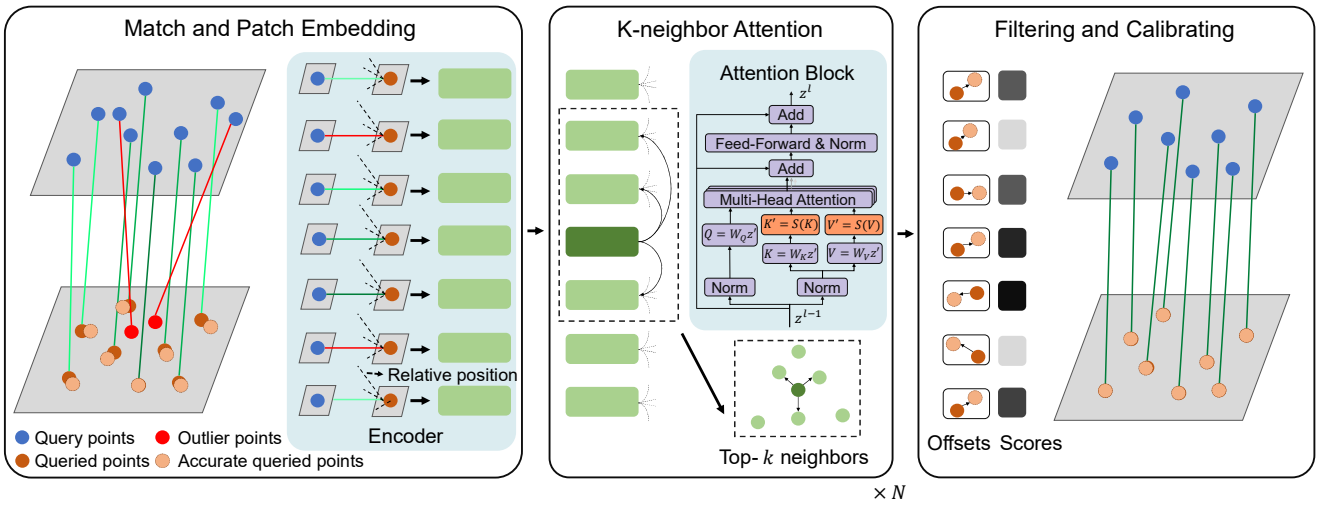


Figure 2. **Overview of FC-GNN.** For a given image pair and a set of matches, we first extract the patch pairs associated with the matches. The matches are represented in the form of relative positions, and then the relative positions and patch pairs are projected to a high-dimensional feature for self-attention. In the attention network, a match only focuses on its k-neighbors in the matching space (viewed as a pre-sampling function $S(\cdot)$), so that its receptive field is gradually expanded from local to global. After N layers of attention, two decoder heads output the matching confidence and offsets respectively. We directly reject matches with low confidence and use the generated offsets to perform calibration for the other matches.

3. Method

3.1. Problem Definition

Given an image pair (I_A, I_B), along with a set of assumed correspondences $\mathbf{C}$, our objective is to improve the reliability and accuracy of this set. This problem can be decomposed into two sub-problems: outlier rejection and inaccurate match calibration:

Outlier rejection involves the identification of mismatched correspondences with big errors. A match can be denoted as $m = (p_1, p_2)$, where $p_1 = (x_1, y_1)$ is a point in image I_A and $p_2 = (x_2, y_2)$ is the corresponding point in image I_B. We define the error of it as:

$$E(m) = ||\kappa(p_1) - p_2||_2, \tag{1}$$

where $\kappa(\cdot)$ represents the transformation between the two images and $p_2^{gt} = \kappa(p_1)$ represents the ground-truth position in image I_B corresponding to p_1 in I_A. $||\cdot||_2$ denotes the Euclidean norm. A match is considered an outlier if $E(m)$ exceeds the threshold ϵ. Our goal is to estimate a reliable set of correspondences $\mathbf{C}_f$, where

$$\mathbf{C}_f = \{(p_1, p_2) \mid ||\kappa(p_1) - p_2||_2 < \epsilon, \forall (p_1, p_2) \text{ in } \mathbf{C}\}. \tag{2}$$

Inaccurate match calibration aims to optimize the set $\mathbf{C}_f$ to improve its pixel-level accuracy, and we seek to estimate the calibration offset $r_{(p_1,p_2)}$ for each match in set $\mathbf{C}_f$. A new set of correspondences $\mathbf{C}_r$ can be generated by:

$$\mathbf{C}_r = \{(p_1, p_2 + r_{(p_1,p_2)}) \mid \forall (p_1, p_2) \text{ in } \mathbf{C}_f\}. \tag{3}$$

Our optimization objective is to minimize the value of $||\kappa(p_1) - p_2 - r_{(p_1,p_2)}||_2$.

3.2. Key Strategy

To recover reliable and accurate correspondences from interferences, we believe that both contextual and local information are effective and should be utilized. **For outlier rejection**, contextual information allows the model to perceive the geometric structure between images, while local information can help assess the reliability of matches from a local perspective, as similar patch pairs are more likely to represent correct matches. **For match calibration**, local information helps correct matches directly, while contextual information helps the calibration eliminate uncertainties and interferences based on the perception of correct geometric estimation.

To combine local and contextual information, we use attention-based graph neural networks. However, running on a complete graph (full attention) is too complex. We observe that the correlation of neighboring correspondences is usually stronger and also constrained by geometric relationships. Therefore, we start processing from neighboring correspondences and gradually elevate to a more global level (similar to convolution), which is more efficient.

3.3. Network Architecture

We use an attention-based graph neural network architecture like [5, 36, 39]. However, unlike these methods, our processing units are matches instead of keypoints,

which means that our attention only takes the form of self-attention. The architecture is shown in Fig. 2.

3.3.1 Match and Patch Embedding

For a given match and its corresponding patch pair, we first jointly embed them into a high-dimensional feature as the basic input. It includes both match information and local textures.

Match embedding. For a given match $m = (p_1, p_2)$, it represents a four-dimensional coordinate (x_1, y_1, x_2, y_2), where $p_1 = (x_1, y_1), p_2 = (x_2, y_2)$. There are several methods for position encoding, including absolute [11, 43], relative [25, 38] and implicit position encoding [7, 46], etc. We use relative positional encoding for the following reasons: (i) It makes position encoding invariant to changes in image size, and (ii) it directly expresses the geometric relationship between different matches. For each match, we first calculate its distance from all the other matches. For example, for matches $m = (x_1, y_1, x_2, y_2)$ and $m' = (x'_1, y'_1, x'_2, y'_2)$, the distance is defined as:

$$d(m, m') = ||m - m'||_2. \tag{4}$$

Then, for match m, we select its nearest k matches, and then calculate the relative positions between these matches and match m. For the top-i nearest match, $pos^i = (x_1^i - x_1, y_1^i - y_1, x_2^i - x_2, y_2^i - y_2)$. The overall relative position is $pos = (pos^0, pos^1, ..., pos^{k-1})$, we then project it into a d-dimensional feature $\mathbf{p}$ through Multilayer Perceptron (MLP).

Patch embedding. To enhance the optimization of correspondence sets, we incorporate local match information by directly embedding relevant patches. This approach offers two advantages: (i) Avoid deep dense features like [52], thereby reducing the number of parameters and computation time. (ii) Let the network learn the difference between two patches from the beginning. We extract two patch blocks of $S \times S$ size centered on the matching points. For each patch pair, we flatten and project it to a d-dimensional feature $\mathbf{f}$ through MLP.

Feature merging. We can directly utilize the added features $\mathbf{z} = \mathbf{p} + \mathbf{f}$ to obtain both match and patch information.

3.3.2 K-neighbor Self-attention

To reduce the computational complexity of attention, we expand the receptive field gradually from local to a large scale. We define the distance $d(m, m')$ between matches as the Euclidean distance in the matching space like match embedding. An obvious observation is that neighboring matches have higher correlation, and are also constrained by geometric relations. Therefore, for each match, we search for its k-nearest neighbor matches and generate an index, which can be formulated as a sampling function $S(\cdot)$. Our attention formulation can be expressed as:

$$\hat{\text{Attn}}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}(\frac{\mathbf{Q}S(\mathbf{K})^T}{\sqrt{dim}})S(\mathbf{V}), \tag{5}$$

in which $\mathbf{Q}$, $\mathbf{K}$, $\mathbf{V}$ denote query, key and value respectively, and dim is the feature dimension. In this way, each match only needs to attend to the k-nearest neighbor matches, which reduces the computational complexity from $O(n^2)$ to $O(kn)$, where n is the length of inputs. We can expand the receptive field of attention by stacking multiple attention layers. We imitate [43] and use the residual [18] architecture to build the attention layer. The l-th layer can be expressed as:

$$\begin{aligned} \hat{\mathbf{z}}^l &= \text{Proj}(\text{LN}(\mathbf{z}^{l-1})), \\ \widetilde{\mathbf{z}}^l &= \hat{\text{MHA}}(\hat{\mathbf{z}}^l) + \mathbf{z}^{l-1}, \\ \mathbf{z}^l &= \text{FFN}(\text{LN}(\widetilde{\mathbf{z}}^l)) + \mathbf{z}^{l-1}, \end{aligned} \tag{6}$$

in which $\mathbf{z}^{l-1}$ and $\mathbf{z}^l$ denote the input and output features, respectively. $\hat{\mathbf{z}}^l$ and $\widetilde{\mathbf{z}}^l$ are intermediate variables. Proj, LN, MHA and FFN denote Projection, LayerNorm, Multi-Head Attention, and Feed Forward Network, respectively.

3.3.3 Filtering and Calibrating

We employ two decoders, namely two MLPs, for filtering and calibrating matches. For each match in the given correspondence set $\mathbf{C}$, the filtering decoder assigns it a confidence score θ, which is scaled to a range of 0 to 1 using the sigmoid function. For outliers with large errors, we directly reject them, and obtain the filtered matching set $\mathbf{C}_f$:

$$\mathbf{C}_f = \{(p_1, p_2) \mid \theta_{(p_1,p_2)} > \theta_f, \forall (p_1, p_2) \text{ in } \mathbf{C}\}, \tag{7}$$

in which θ_f is the set threshold. The calibrating decoder computes the offset values for the two directions $r = (x, y)$ of each matching output. For matches marked as inliers (in $\mathbf{C}_f$), we obtain the calibrated matching set $\mathbf{C}_r$ based on the filtered set:

$$\mathbf{C}_r = \{(p_1, p_2 + r_{(p_1,p_2)}) \mid \forall (p_1, p_2) \text{ in } \mathbf{C}_f\}, \tag{8}$$

in which $\mathbf{C}_r$ represents the final calibrated correspondences set obtained after optimization.

3.3.4 Loss

Consistent with the problem definition, our loss includes two parts, namely, the matching classification loss and the calibration loss. A match with an error greater than ϵ pixels is considered incorrect, and the corresponding ground-truth

set is $\mathcal{M}_n$, the rest of the match set is expressed as $\mathcal{M}_p$. We use logarithmic loss and the classification loss $\mathcal{L}_f$ can be expressed as :

$$\mathcal{L}_f = -\frac{1}{|\mathcal{M}_p|}\sum_{m\in\mathcal{M}_p}\log\theta_m - \frac{1}{|\mathcal{M}_n|}\sum_{m\in\mathcal{M}_n}\log\left(1-\theta_m\right), \tag{9}$$

in which θ_m is the predicted score for match m. Unlike the classification loss, the calibration loss $\mathcal{L}_r$ is only applied to matches with errors smaller than ϵ pixels. We express it as the mean of Euclidean distance between the predicted offsets r_m and the ground-truth offsets r_m^{gt}:

$$\mathcal{L}_r = \frac{1}{|\mathbf{C}_f^{gt}|}\sum_{m\in\mathbf{C}_f^{gt}}||r_m - r_m^{gt}||_2. \tag{10}$$

The overall loss is:

$$\mathcal{L} = \mathcal{L}_f + \mathcal{L}_r. \tag{11}$$

3.4. Comparisons to Related Work

FC-GNN vs. Filtering-only methods. Filtering-only frameworks [41, 48, 49] find inliers from noisy matches and the effect of filtering is limited by the initial intrinsic of matches. FC-GNN proposes joint filtering and calibrating, and can even optimize well-filtered match sets.

FC-GNN vs. Patch2Pix [52]. Patch2Pix regresses matches from local patch proposals on top of deep features extracted by ResNet [18]. However, it suffers from two limitations: 1) Extracting dense feature maps reduces speed and increases memory consumption. 2) It lacks consideration for matching contextual information, limiting its optimization capabilities. FC-GNN incorporates local information from the image patches and aggregates context information, thus reducing complexity and achieving better performance.

FC-GNN vs. PixSfM [23]. Our method emphasizes different aspects from PixSfM. PixSfM directly optimize multi-view SfM tracks, and does not reject outliers. Our method is designed for better two-view optimization. For exchange, when used in SfM pipelines, it may cause the disconnecting of the tracks. Therefore, we do not use PixSfM as a baseline for comparison, nor do we test on SfM pipelines.

3.5. Implementation Details

Training data. We train our model on the widely used MegaDepth [22] dataset like Patch2Pix [52]. We hope our method to be able to handle a more general distribution. Therefore, we use synthetic data for training. A specialized data processing way is utilized for our newly proposed filtering and calibrating approach:

We introduce noises to the ground-truth correspondences step by step. The noises can be classified into outlier noise and inlier noise. Outlier noise corresponds to mismatches of large errors, while inlier noise arises from inaccurate keypoint detection or adjacent mismatches with small errors. To add outlier noise, we replace the original correct match with randomly selected two points from two images. The inlier noise offsets are added with queried points, which can be determined by the radius R and the angle α. The radius follows an absolute value Gaussian distribution with a standard deviation of δ, where $\delta \sim U(0, 10)$, and $\alpha \sim U(0, 2\pi)$ ($U(\cdot,\cdot)$ denotes uniform distribution). The offset is:

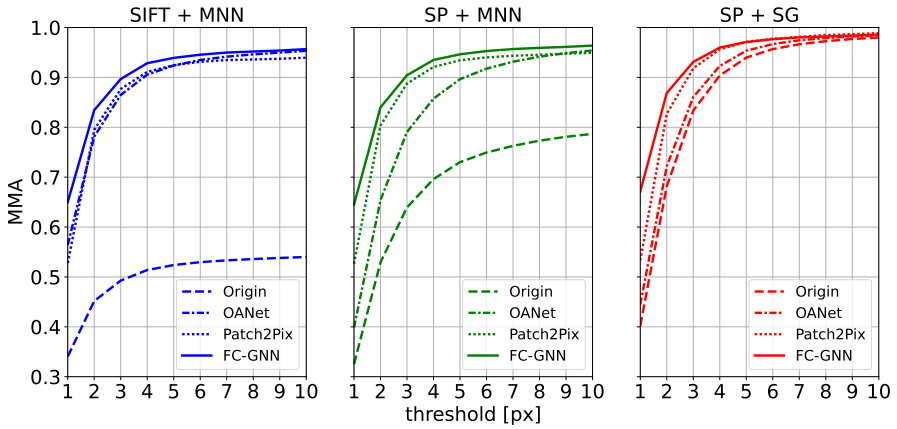
$$(x_n, y_n) = (R\cdot\sin(\alpha), R\cdot\cos(\alpha)). \tag{12}$$

A small uniform bias (e_1, e_2) (standard normal distribution) is then added to all points to avoid zero mean deviation. During training, we sample a total of 18k pairs from the MegaDepth dataset with overlap rate ranging from 0.1 to 0.3, and crop all training images to a size of 640 $\times$ 640. For each sample, we randomly select 3k matches from the dense annotations (pixel-to-subpixel) and add noise to them. Matches with an error exceeding 8 pixels will be considered as incorrect matches.

Training details. In the process of match and patch embedding, we use 8 neighbors for relative position encoding, and the size of the patch is set to 41×41. Both are projected into 256-dimensional features through MLP. There are 9 attention layers, with each layer pays attention to 8 adjacent matches (including itself). During training, the batch size is set to 4, the initial learning rate is set to 1×10^{-4} and decaying 1% after every 20,000 samples. We use the AdamW [26] optimizer for optimization and the weight decay is set to 0.1. We train and evaluate our model on a V100 GPU, and the model converges after about 3 days of training. For evaluation, we use θ_f = 0.999 for image matching tasks and θ_f = 0.5 for geometric estimation tasks.

4. Experiments

To evaluate our model, we perform a detailed analysis on three tasks: image matching, homography estimation, and relative pose estimation. We employ three different matchers: SIFT [27] + mutual NN (SIFT + MNN), SuperPoint [10] + mutual NN (SP + MNN), and SuperPoint [10] + SuperGlue [36] (SP + SG), which represent handcrafted features + simple matching methods, trained features + simple matching methods, and trained features + contextual matching methods, respectively. Additionally, we include ASpanFormer [6] as a representative of dense matching for some tasks. For matching optimizers, we select OANet [49] and Patch2Pix [52] as baselines, in which the former can reject outliers and the latter can jointly filter and refine matches. We also present results on raw matching without any post-processing (Origin).



Method	#Matches
SIFT [27] + MNN	1.78k
∼ OANet [49]	0.83k
∼ Patch2Pix [52]	0.84k
∼ FC-GNN	1.02k
SP [27] + MNN	1.19k
∼ OANet [49]	0.61k
∼ Patch2Pix [52]	0.73k
∼ FC-GNN	0.90k
SP [27] + SG [36]	1.19k
∼ OANet [49]	0.30k
∼ Patch2Pix [52]	0.73k
∼ FC-GNN	1.10k

Figure 3. **Image Matching on HPatches [1].** The MMA for each method is presented as a function of the matching threshold (in pixels). We denote our methods with solid lines and others with dashed lines to make the results more clear.

Matcher	Refiner	Illumination	Viewpoint	Overall	#Matches	Time (ms)
		AUC (%, @3, 5, 10px)				
ASpanFormer [6]	Origin	**80.0 / 87.3 / 93.4**	49.2 / 62.1 / 75.4	64.3 / 74.4 / 84.2	2.76k	**163.6**
	FC-GNN	79.4 / 87.2 / **93.4**	**53.4 / 65.5 / 78.1**	**66.1 / 76.1 / 85.5**	2.75k	**12.8**
SIFT [27] + MNN	Origin	69.4 / 78.3 / 85.9	50.3 / 62.4 / 75.3	59.6 / 70.1 / 80.4	0.84k	**109.6**
	OANet [49]	68.3 / 77.9 / 85.7	41.9 / 56.8 / 72.8	54.8 / 67.1 / 79.1	0.39k	13.4
	Patch2Pix [52]	70.3 / 79.6 / 87.6	40.1 / 53.8 / 70.0	54.8 / 66.4 / 78.6	0.60k	128.7
	FC-GNN	**72.4 / 81.3 / 88.8**	**51.8 / 63.9 / 77.2**	**61.9 / 72.4 / 82.8**	0.56k	**7.6**
SP [10] + MNN	Origin	61.4 / 74.8 / 87.0	38.0 / 53.5 / 69.8	49.4 / 63.9 / 78.2	0.55k	**11.4**
	OANet [49]	46.6 / 64.0 / 80.7	24.2 / 38.2 / 56.7	35.1 / 50.8 / 68.4	0.25k	11.3
	Patch2Pix [52]	72.9 / 83.0 / 91.4	41.4 / 55.0 / 69.8	56.8 / 68.7 / 80.3	0.51k	102.5
	FC-GNN	**75.2 / 84.4 / 92.0**	**48.9 / 61.3 / 74.9**	**61.8 / 72.6 / 83.3**	0.49k	**6.2**
SP [10] + SG [36]	Origin	62.6 / 76.4 / 88.1	45.7 / 61.3 / 76.8	54.0 / 68.6 / 82.3	0.61k	**48.2**
	OANet [49]	37.4 / 56.5 / 76.2	20.1 / 35.6 / 56.8	28.5 / 45.8 / 66.3	0.14k	11.7
	Patch2Pix [52]	72.4 / 82.8 / 91.2	42.4 / 56.2 / 72.6	57.0 / 69.2 / 81.7	0.58k	104.4
	FC-GNN	**77.3 / 85.9 / 92.9**	**57.2 / 69.0 / 81.3**	**67.0 / 77.2 / 86.9**	0.60k	**6.4**

Table 1. **Homography estimation and runtime comparison on HPatches [1].** The error AUCs in percentage is reported. The matching runtime are colored **red** and the matching optimization processes are individually timed. We mark the best results in **bold**.

4.1. Image Matching

Our model jointly filters and calibrates matches, so the accuracy of matches is a direct metric of reflection. We first test our model on the HPatches [1] dataset, in which 57 sequences are mainly illumination changes and 59 sequences are mainly viewpoint changes. We use the ground-truth homography matrix $\mathcal{H}$ to obtain the mean matching accuracy (MMA) of the results.

Results. The results in Fig. 3 demonstrate FC-GNN's robust filtering and refinement capabilities across various matchers. Compared to OANet [49] and Patch2Pix [52], our method effectively eliminates outliers and improves the pixel-level matching accuracy. It is worth noting that our method also preserves more correspondences, indicating that our approach is more robust, thereby avoiding incorrect overfiltering. Specifically, for SP [10] + SG [36], in which outliers have already been well filtered out, our method achieves significant improvements in accuracy compared to OANet, which only filters but does not calibrate, and outperforms Patch2Pix, whose refinement capability is insufficient. As the threshold decreases, our method outperforms other methods by a significant margin, and boosts the accuracy of SP + SG from 40% to 67% at the 1-pixel threshold. Additional qualitative results can be found in Fig. 4.

4.2. Homography Estimation

In this section, we assess our model's homography estimation and runtime performance with the HPatches dataset [1] on a V100 GPU. For interpretable experimental results, we crop images to a $4:3$ aspect ratio and scale them to 640×480. Homography matrix $\hat{\mathcal{H}}$ is estimated using RANSAC

Matcher	Refiner	Pose estimation AUC @5°	@10°	@20°
ASpanFormer [6]	Origin	55.36	71.31	82.90
	FC-GNN	**56.64**	**72.43**	**83.76**
SIFT [27] + MNN	Origin	16.67	28.54	42.75
	OANet [49]	40.28	57.07	71.00
	Patch2Pix [52]	33.54	48.66	62.07
	FC-GNN	**44.57**	**60.32**	**72.78**
SP [10] + MNN	Origin	30.00	45.24	59.29
	OANet [49]	31.59	49.30	64.32
	Patch2Pix [52]	39.29	54.83	67.27
	FC-GNN	**45.58**	**60.92**	**72.19**
SP [10] + SG [36]	Origin	49.13	66.16	79.23
	OANet [49]	23.40	40.31	58.36
	Patch2Pix [52]	47.32	63.98	77.23
	FC-GNN	**54.67**	**71.03**	**82.65**

Table 2. **Outdoor pose estimation results.** The AUC of the pose error in percentage is reported. We mark the best results in **bold**.

Matcher	Refiner	Pose estimation AUC @5°	@10°	@20°
ASpanFormer [6]	Origin	25.69	45.85	63.31
	FC-GNN	**26.01**	**46.43**	**63.90**
SIFT [27] + MNN	Origin	4.26	10.10	18.11
	OANet [49]	5.88	13.58	23.22
	Patch2Pix [52]	6.09	14.07	24.62
	FC-GNN	**7.88**	**16.87**	**27.59**
SP [10] + MNN	Origin	8.79	19.51	32.51
	OANet [49]	7.15	16.96	29.48
	Patch2Pix [52]	12.01	25.83	40.91
	FC-GNN	**14.47**	**29.45**	**44.50**
SP [10] + SG [36]	Origin	15.31	31.64	48.00
	OANet [49]	3.56	9.66	19.83
	Patch2Pix [52]	15.33	31.74	48.10
	FC-GNN	**18.46**	**36.47**	**52.98**

Table 3. **Indoor pose estimation results.** The AUC of the pose error in percentage is reported. We mark the best results in **bold**.

[15] as the robust estimator, with default parameters. AUCs of corner errors [40] are reported with thresholds of 3, 5, and 10. We report the results for illumination, viewpoint, and overall, respectively.

Results. As shown in Tab. 1, our method achieves the best overall homography estimation results on each matcher. Specifically, on SP [10] + SG [36] overall, our method achieves optimal performance with AUC improvements of 13.0%, 8.6%, and 4.6% at 3, 5, and 10 pixel thresholds. Our model enhances ASpanFormer's performance in dense matching, albeit to a lesser degree than sparse methods. This is likely due to its initial high performance and the less distinctive matching points as dense matcher. We no longer report the results of OANet and Patch2Pix on ASpanFormer as they both result in negative optimization.

Tab. 1 also reports the runtime. It can be easily observed that: 1) With FC-GNN, SP + SG outperforms ASpanFormer with much less time. 2) FC-GNN enables SP + MNN to surpass SP + SG with much less time. 3) FC-GNN exhibits superior efficiency and performance compared to OANet and PatchPix. These findings highlight the excellent optimization achieved by FC-GNN at lightning speed, making it highly valuable for real-world applications.

4.3. Outdoor Pose Estimation

We evaluate our model's outdoor pose estimation using the MegaDepth1500 dataset [22, 40], which includes challenging lighting and viewpoint changes, as well as repetitive patterns. Scenes of the dataset are excluded during our training. We estimate relative pose using RANSAC and evaluate the pose error's AUC with thresholds of 5°, 10°, and 20° following [6, 36, 40].

Results. As shown in Tab. 2, our method again achieves the best results. This proves that our method can perform matching filtering and calibrating in more complex situations. For SP + SG , its error AUC under 5° is increased by 5.5%, making it comparable to dense matcher ASpanFormer. OANet [49] and Patch2Pix [52] do not improve the performance of SP + SG. Our network can also improve the performance of AspanFormer, demonstrating strong optimization capabilities.

4.4. Indoor Pose Estimation

We then evaluate our model on indoor scenes. An important challenge for indoor scenes is the lack of textures. We test on ScanNet [8, 40] dataset. Similar to outdoor, we report the error AUCs for thresholds 5°, 10°, and 20° respectively.

Results. As shown in Tab. 3, our method is able to optimize the matching performance even in challenging scenarios with a lot of flat areas. Such ability of our method is partly due to the calibration ability of our network, as well as our training approach that does not assume the position of keypoints. As a result, matches even on relatively flat regions are trained and optimized, ensuring the adaptability of our algorithm to different datasets.

4.5. Understanding FC-GNN

Ablation study. We perform ablation experiments on HPatches [1] for image matching and on MegaDepth [22] for pose estimation, using SP [10] + MNN as our matcher. We report the results of reserving different components, including context information (T, matches only), local information (L, patches only and no attention neighbors), filtering (F), and calibration (C). Results in Tab. 4 show significant performance improvements with each FC-GNN block. Take a closer look, utilizing contextual information does a better job of filtering out outliers, while local information

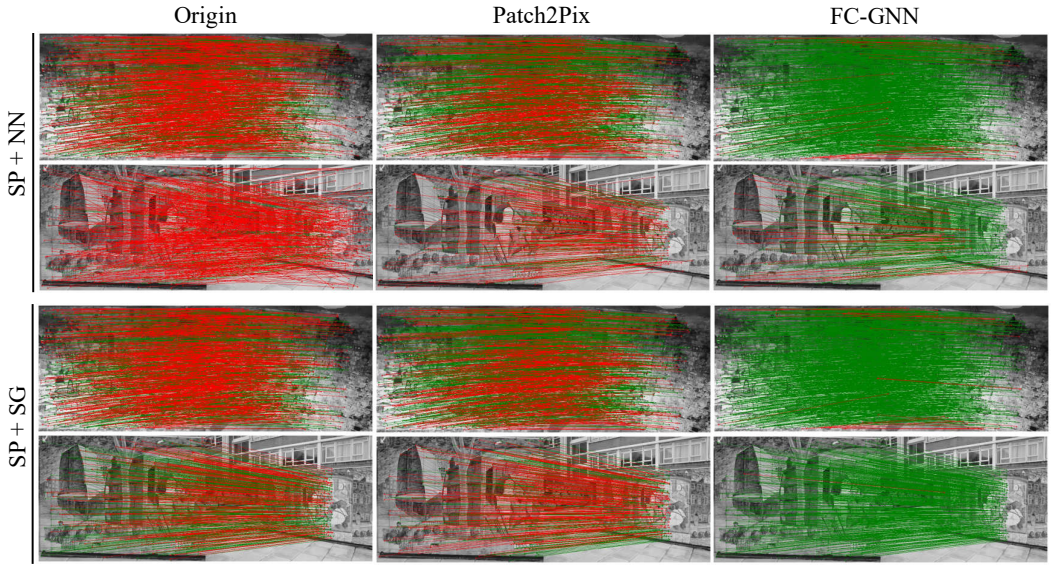


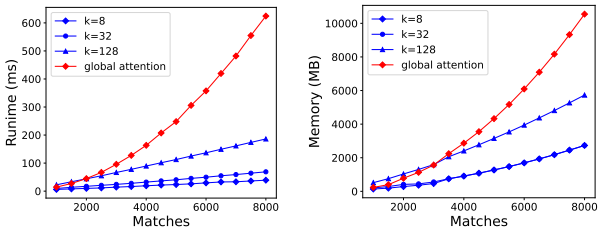
Figure 4. **Qualitative image matches on HPatches [1].** We mark matches with an error ≤ 1 pixel as green, and the rest as red. It can be seen that FC-GNN greatly improves the accuracy of matching and effectively filtering out outliers.

method	MMA @1px	MMA @8px	Pose. est @20°
SP [10] + MNN (Origin)	32.5	77.3	59.3
T + F	39.9	95.5	67.2
T + C	43.3	78.5	61.5
T + FC	52.3	95.7	66.8
L + F	45.3	91.9	61.0
L + C	46.3	77.9	64.6
L + FC	59.6	92.0	66.7
TL + F	42.1	95.8	68.3
TL + C	51.4	78.3	65.6
TL + FC (3 layers)	61.1	95.7	71.8
TL + FC (9 layers)	**64.5**	**95.9**	**72.2**

Table 4. **Ablation of FC-GNN.** We show the significance of incorporating all elements of context information (T), local information (L), filtering (F), and calibration (C).

holds an advantage in enhancing match calibration. Filtering and calibration also both greatly enhance matching performance. We also evaluate a model with fewer layers and it can be observed that even the relatively small model demonstrates strong optimization capabilities, indicating the effectiveness of our design. Ultimately, the full model achieves the best image matching and geometry estimation results, indicating high-quality matching.

Efficiency analysis. We show the runtime and memory usage of our model with k-neighbor attention and global attention on a V100 GPU. As shown in Fig. 5, the k-neighbor attention model increases much slower in runtime and memory usage with more matches. For instance, at $k = 8$, processing 5k matches only takes 23 ms.



(a) Comparison of runtime. (b) Comparison of memory.

Figure 5. **Efficiency analysis.** We report the runtime and memory consumption with an increasing number of input matches.

5. Conclusion

We propose FC-GNN, an attention-based graph neural network that jointly filters out outliers and improves pixel-level matching accuracy for correspondence sets. Our approach follows a filtering and calibrating process, leveraging contextual and local information through cascaded attention layers. FC-GNN optimizes the results of various matching pipelines, leading to improved various tasks. Furthermore, we adopt k-nearest neighbor attention to reduce the model's complexity, making it more suitable for real-world tasks.

Acknowledgements. This research was partly supported by grants of National Natural Science Foundation of China (NSFC, Grant No. 62171281), Science and Technology Commission of Shanghai Municipality (STCSM, Grant Nos. 20DZ1200203, 2021SHZDZX0102, 22DZ2229005).

References

[1] Vassileios Balntas, Karel Lenc, Andrea Vedaldi, and Krystian Mikolajczyk. Hpatches: A benchmark and evaluation of handcrafted and learned local descriptors. In *CVPR*, 2017. 6, 7, 8

[2] Axel Barroso-Laguna, Edgar Riba, Daniel Ponsa, and Krystian Mikolajczyk. Key. net: Keypoint detection by handcrafted and learned cnn filters. In *CVPR*, 2019. 2

[3] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In *ECCV*, 2006. 2

[4] Eric Brachmann and Carsten Rother. Neural-guided ransac: Learning where to sample model hypotheses. In *CVPR*, 2019. 1, 2

[5] 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. In *ICCV*, 2021. 1, 2, 3

[6] Hongkai Chen, Zixin Luo, Lei Zhou, Yurun Tian, Mingmin Zhen, Tian Fang, David McKinnon, Yanghai Tsin, and Long Quan. Aspanformer: Detector-free image matching with adaptive span transformer. In *ECCV*, 2022. 2, 5, 6, 7

[7] Xiangxiang Chu, Bo Zhang, Zhi Tian, Xiaolin Wei, and Huaxia Xia. Do we really need explicit position encodings for vision transformers. *arXiv preprint arXiv:2102.10882*, 3 (8), 2021. 4

[8] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *CVPR*, 2017. 7

[9] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Deep image homography estimation. *arXiv preprint arXiv:1606.03798*, 2016. 1

[10] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superpoint: Self-supervised interest point detection and description. In *CVPR*, 2018. 2, 5, 6, 7, 8

[11] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 4

[12] Mihai Dusmanu, Ignacio Rocco, Tomas Pajdla, Marc Pollefeys, Josef Sivic, Akihiko Torii, and Torsten Sattler. D2-net: A trainable cnn for joint description and detection of local features. In *CVPR*, 2019. 2

[13] Mihai Dusmanu, Johannes L Schönberger, and Marc Pollefeys. Multi-view optimization of local feature geometry. In *ECCV*, 2020. 2

[14] Ufuk Efe, Kutalmis Gokalp Ince, and Aydin Alatan. Dfm: A performance baseline for deep feature matching. In *CVPR*, 2021. 2

[15] Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 1981. 2, 7

[16] Hugo Germain, Guillaume Bourmaud, and Vincent Lepetit. S2dnet: Learning accurate correspondences for sparse-to-dense feature matching. *arXiv preprint arXiv:2004.01673*, 2020. 2

[17] Pierre Gleize, Weiyao Wang, and Matt Feiszli. Silk–simple learned keypoints. *arXiv preprint arXiv:2304.06194*, 2023. 2

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 4, 5

[19] Jogendra Nath Kundu, MV Rahul, Aditya Ganeshan, and R Venkatesh Babu. Object pose estimation from monocular image using multi-view keypoint correspondence. In *ECCV*, 2018. 1

[20] Jongmin Lee, Byungjin Kim, and Minsu Cho. Self-supervised equivariant learning for oriented keypoint detection. In *CVPR*, 2022. 2

[21] Xinghui Li, Kai Han, Shuda Li, and Victor Prisacariu. Dual-resolution correspondence networks. In *NeurIPS*, 2020. 2

[22] Zhengqi Li and Noah Snavely. Megadepth: Learning single-view depth prediction from internet photos. In *CVPR*, 2018. 5, 7

[23] Philipp Lindenberger, Paul-Edouard Sarlin, Viktor Larsson, and Marc Pollefeys. Pixel-perfect structure-from-motion with featuremetric refinement. In *CVPR*, 2021. 2, 5

[24] Philipp Lindenberger, Paul-Edouard Sarlin, and Marc Pollefeys. Lightglue: Local feature matching at light speed. *arXiv preprint arXiv:2306.13643*, 2023. 2

[25] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *CVPR*, 2021. 4

[26] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. 5

[27] David G Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 2004. 2, 5, 6, 7

[28] Zixin Luo, Tianwei Shen, Lei Zhou, Jiahui Zhang, Yao Yao, Shiwei Li, Tian Fang, and Long Quan. Contextdesc: Local descriptor augmentation with cross-modality context. In *CVPR*, 2019. 2

[29] 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. In *CVPR*, 2020. 2

[30] Anastasiia Mishchuk, Dmytro Mishkin, Filip Radenovic, and Jiri Matas. Working hard to know your neighbor's margins: Local descriptor learning loss. *NeurIPS*, 30, 2017.

[31] Jerome Revaud, Philippe Weinzaepfel, César De Souza, Noe Pion, Gabriela Csurka, Yohann Cabon, and Martin Humenberger. R2d2: repeatable and reliable detector and descriptor. *arXiv preprint arXiv:1906.06195*, 2019. 2

[32] Jérome Revaud, Vincent Leroy, Philippe Weinzaepfel, and Boris Chidlovskii. Pump: Pyramidal and uniqueness matching priors for unsupervised learning of local descriptors. In *CVPR*, 2022. 2

[33] Ignacio Rocco, Mircea Cimpoi, Relja Arandjelović, Akihiko Torii, Tomas Pajdla, and Josef Sivic. Neighbourhood consensus networks. In *NeurIPS*, 2018. 2

[34] Ignacio Rocco, Relja Arandjelović, and Josef Sivic. Efficient neighbourhood consensus networks via submanifold sparse convolutions. In *ECCV*, 2020. 2

[35] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. Orb: An efficient alternative to sift or surf. In *ICCV*, 2011. 2

[36] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superglue: Learning feature matching with graph neural networks. In *CVPR*, 2020. 1, 2, 3, 5, 6, 7

[37] Nikolay Savinov, Akihito Seki, Lubor Ladicky, Torsten Sattler, and Marc Pollefeys. Quad-networks: unsupervised learning to rank for interest point detection. In *CVPR*, 2017. 2

[38] Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. Self-attention with relative position representations. *arXiv preprint arXiv:1803.02155*, 2018. 4

[39] Yan Shi, Jun-Xiong Cai, Yoli Shavit, Tai-Jiang Mu, Wensen Feng, and Kai Zhang. Clustergnn: Cluster-based coarse-to-fine graph neural network for efficient feature matching. In *CVPR*, 2022. 1, 2, 3

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

[41] Weiwei Sun, Wei Jiang, Eduard Trulls, Andrea Tagliasacchi, and Kwang Moo Yi. Acne: Attentive context normalization for robust permutation-equivariant learning. In *CVPR*, 2020. 2, 5

[42] Michał Tyszkiewicz, Pascal Fua, and Eduard Trulls. Disk: Learning local features with policy gradient. In *NeurIPS*, 2020. 2

[43] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017. 2, 4

[44] Qianqian Wang, Xiaowei Zhou, Bharath Hariharan, and Noah Snavely. Learning feature descriptors using camera pose supervision. In *ECCV*, 2020. 2

[45] Qing Wang, Jiaming Zhang, Kailun Yang, Kunyu Peng, and Rainer Stiefelhagen. Matchformer: Interleaving attention in transformers for feature matching. *arXiv preprint arXiv:2203.09645*, 2022. 2

[46] Haiping Wu, Bin Xiao, Noel Codella, Mengchen Liu, Xiyang Dai, Lu Yuan, and Lei Zhang. Cvt: Introducing convolutions to vision transformers. In *CVPR*, 2021. 4

[47] Fei Xue, Ignas Budvytis, and Roberto Cipolla. Imp: Iterative matching and pose estimation with adaptive pooling. In *CVPR*, 2023. 2

[48] Kwang Moo Yi, Eduard Trulls, Yuki Ono, Vincent Lepetit, Mathieu Salzmann, and Pascal Fua. Learning to find good correspondences. In *CVPR*, 2018. 1, 2, 5

[49] 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. In *ICCV*, 2019. 1, 2, 5, 6, 7

[50] Xiaoming Zhao, Xingming Wu, Weihai Chen, Peter CY Chen, Qingsong Xu, and Zhengguo Li. Aliked: A lighter keypoint and descriptor extraction network via deformable transformation. *IEEE Transactions on Instrumentation and Measurement*, 2023. 2

[51] Qunjie Zhou, Torsten Sattler, Marc Pollefeys, and Laura Leal-Taixe. To learn or not to learn: Visual localization from essential matrices. In *Proceedings of IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2020. 1

[52] Qunjie Zhou, Torsten Sattler, and Laura Leal-Taixe. Patch2pix: Epipolar-guided pixel-level correspondences. In *CVPR*, 2021. 2, 4, 5, 6, 7