FilterGNN: Image Feature Matching with Cascaded Outlier Filters and Linear Attention

Jun-Xiong Cai
Tsinghua University
Being, China
caijunxiogn000@136.com

Tai-Jiang Mu*
Tsinghua University
Beijing, China
taijiang@tsinghua.edu.cn

Yu-Kun Lai
Cardiff University
Wales, United Kingdom
yukun.lai@cs.cardiff.ac.uk

Abstract

Cross-view matching of image local features is a fundamental task for visual localization and 3D reconstruction. In this paper, we propose FilterGNN, a Transformer-based graph neural network (GNN), aiming to improve matching efficiency and accuracy for visual descriptors. Based on high matching sparseness and coarse-to-fine co-visible area detection, FilterGNN utilizes cascaded optimal graph matching filter modules to dynamically reject outlier matches. Moreover, we have successfully adapted linear attention for FilterGNN with post instance normalization support, which significantly reduces the complexity of complete graph learning from $O(N^2)$ to $O(N)$. Experiments show that FilterGNN only takes 6% time cost and 33.3% memory cost against SuperGlue under large-scale input size, and achieves competitive performance for various tasks, such as pose estimation, visual localization and sparse 3D reconstruction.

Keywords: Image Matching, Transformer, Linear Attention, Visual Localization, Sparse Reconstruction

1. Introduction

Finding pixel-wise correspondences from image pair is an essential step for camera pose estimation which has been widely used in Structure-from-Motion (SfM) [33], Simultaneous Localization and Mapping (SLAM) [22, 13], Visual Localization [30], etc. Most existing methods require two phases: local feature extraction and feature matching. Great efforts [9, 6] have been devoted to feature extraction using deep convolutional neural networks (DCNNs) over the past decade. Recently, some Transformer [44] based methods [31, 3, 36, 38, 39, 12] have been proposed to greatly improve the matching ability against traditional strategy of nearest neighbor (NN) search. Meanwhile, the extra computational cost remains as the main challenge for practical use in real-time applications.

Attention-based graph nerual networks, such as SuperGlue [31], mainly benefit from the Transformer's support for irregular data and the aggregated global context through the pair-wised attention mechanism. Specifically: a) Self-attention, exhaustively calculating correlation of any two keypoints extracted from the same image, is used to aggregate the inner-view global context. b) Correspondingly, cross attention is applied on the complete bipartite graph composed of two keypoint sets grouped by source images to learn cross-view information. c) Slightly different from bipartite graph matching, local feature matching task has a large number of non-matchable keypoints. Therefore, reasonable rejection mechanism is required for an optimal matching layer to detect them. With the refinement of local descriptors by complete graph based attention, SuperGlue has achieved significant performance gains on many pose estimation [41, 2] or visual localization benchmarks [32, 47, 40]. However, the fully-connected attention mechanism (as shown in Fig. 1(c)) brings in a computational complexity of $O(N^2d)$[1], which is much higher than NN search.

Efforts to reduce attention computation go in two main ways. The first is to build a sparse graph from the inputs. Due to the discrete and unordered nature of image local features, traditional methods such as spatial neighbor attention [25, 24, 15], which relies on ordered sliding windows, cannot be directly applied. More appropriate solutions focus on building subgraphs by sampling [3], projection [45, 5] or clustering [36], reducing the computational complexity to $O(kNd)$, with k being a small constant. Usually, these methods would inevitably bring about the loss of information at the size dimension, and the ratio between k and N should be carefully chosen. The other way is to design linear kernel functions to make approximation to fully-connected attention [5, 14, 10]. However, the previous work ClusterGNN [36] has reported the incompatibility of linear approximation on cross-attention, resulting in a significant drop in matching accuracy.

In this paper, we propose FilterGNN, which effectively

*Corresponding author.

[1] N represents the input size; d is the number of feature dimensions.

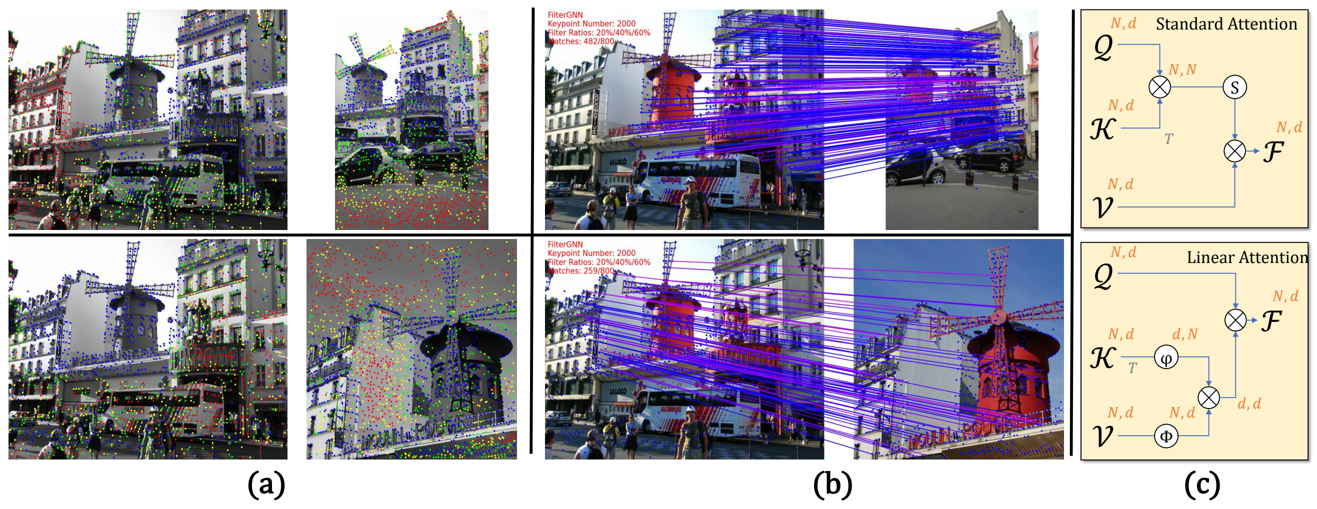


Figure 1. (a) Visualization of our proposed hierarchical filtering. Keypoints in red, yellow and green are discarded layer-by-layer, and (b) the remaining keypoints (blue) are used for the final feature matching with our FilterGNN. (c) Standard and linear attention. $\mathcal{Q} \otimes \mathcal{K}^T$ yields a computational complexity of $O(N^2 d)$, where $\otimes$ means matrix multiplication and (ϕ, φ) represents kernel functions with linear complexity. The dimensions of matrices are shown in orange.

combines the above two ways for comprehensive optimization of feature matching. Previous works [31, 36] detected non-matchable points only at the final optimal matching layer. Our FilterGNN, instead, exploits hierarchical outlier filters to dynamically reject invalid outliers, which are interspersed between attentional aggregation blocks. As shown in Figure 1, the low-level filter can quickly reject the isolated keypoints outside the co-visible area according to the basic descriptors; and the high-level filter is responsible for accurately removing outlier matches according to refined descriptors considering sparsity, visibility and geometric distribution consistency.

To further accelerate matching, we approximate the fully-connected attention with linear attention, by considering the following aspects: i) Residual attention block of SuperGlue [31] may lead to excessive variance amplification which affects the convergence speed and performance. Therefore, a reasonable normalization layer is recommended, especially for training from scratch. ii) Considering the potential huge domain gaps between the input image pair, such as orientation, scale and lighting, we adopt instance normalization [43] instead of layer normalization [1] in the vanilla Transformer [44]. iii) We found that the training gradient of linear attention is not as good as that of standard attention. Since the attention layer does not require extra learnable parameters, rather than training from scratch, we finetune linear attention by using the pretrained weights from standard attention. This not only preserves high matching performance, but also makes the training converge faster.

The main contributions are summarized as follows:

1. We propose a cascaded optimal graph matching filter module which can dynamically reject non-matchable keypoints. It not only reduces the computational cost but also provides a better feature distribution space for highly correlated keypoints.

2. We further propose an efficient and effective linear GNN architecture with post instance normalization, significantly reducing the computational complexity from $O(N^2 d)$ to $O(Nd)$.

3. Extensive experiments on various computer vision tasks demonstrate the applicability of our method, which achieves competitive results compared to state-of-the-art (SOTA) methods while being significantly more efficient.

2. Related Work

Image Feature Matching. Traditional pipelines mainly focus on robust interest point detection and visual descriptor computation. SIFT [19] is a scale-invariant handcrafted feature, which is widely used in the pose estimation task of SfM and Multi-View Stereo (MVS). ORB [28] focuses on efficiency which is mainly applied in SLAM. In recent years, deep convolutional neural networks (CNNs) have inspired many learning-based image feature extraction works, such as D2Net [9], SuperPoint [6] and ASLFeat [20], etc., whose matching ability has significantly surpassed that of handcrafted ones. They describe the content of local regions, and often perform data augmentation on scale, rotation, and imaging perturbations, to achieve orientation invariance [46] or affine invariance [21].

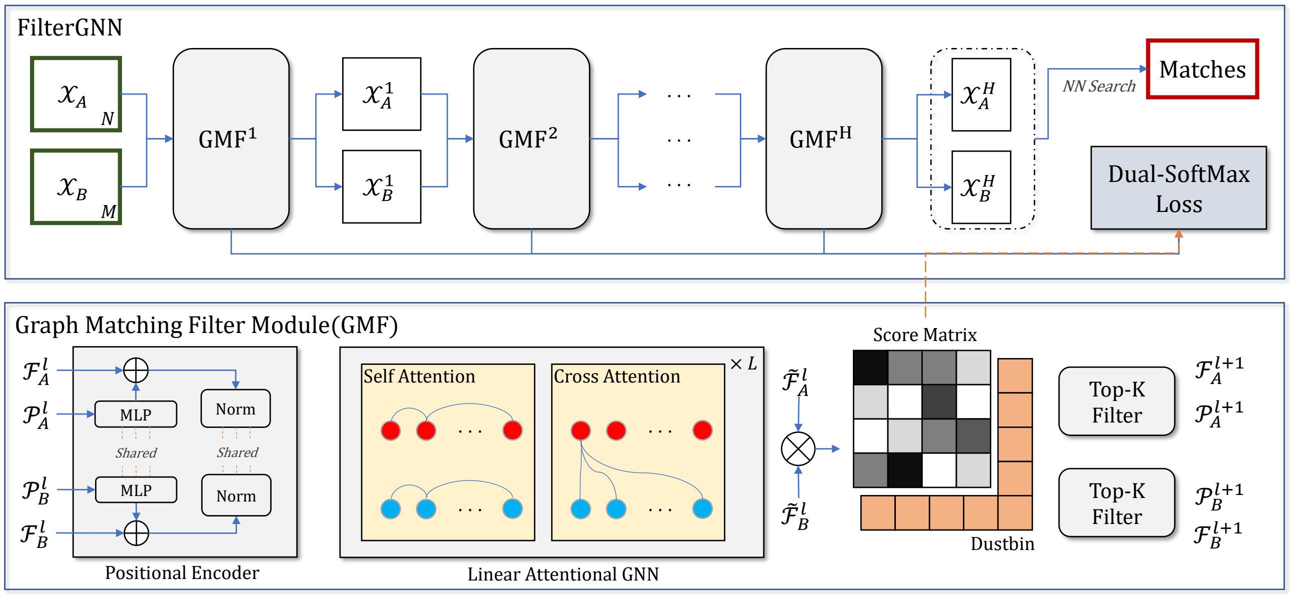


Figure 2. **The FilterGNN architecture.** FilterGNN uses cascaded optimal Graph Matching Filter (GMF) modules (Section 3.2) to efficiently filter out isolated keypoints for better feature matching. Each GMF module, acting as a mini SuperGlue [31] Block, refines the descriptors $\mathcal{F}$ of the input keypoints set $\mathcal{X} = (\mathcal{F}, \mathcal{P})$ using a shared Positional Encoder (Section 3.2.1) and L stacked Linear Attentional GNN layers (Section 3.2.2). A Top-K Filter is adopted to remove the k keypoints with lowest matching probabilities, gradually reducing the size of the keypoint set. The score matrix of each GMF is then expanded with an extra dustbin dimension to detect outliers, and it generates matching loss through the Dual-Softmax operator [26] for training. Final correspondences are generated by performing traditional Nearest Neighbor search on the remaining, distilled keypoint sets $(\mathcal{X}_A^H, \mathcal{X}_B^H)$ after H GMFs.

Recently, Sarlin et al. [31] proposed SuperGlue, which exploited a Transformer based graph neural network to aggregate inner-global and cross-view information from keypoint sets of image pairs. Since the goal of the feature matching network is to refine the descriptors, the source and target image features can be used directly as inputs. SuperGlue improves the Transformer architecture of encoder (stacked self attention modules) and decoder (stacked cross attention modules) with alternately stacked self- and cross- attention modules. However, SuperGlue suffers from a quadratic computational complexity of $O(N^2d)$, making it impractical to be directly applied to real-time systems.

Efficient Transformer-based Architecture. The Transformer architecture has achieved success both in natural language processing [7] tasks and computer vision tasks [8]. The inputs (texts or images) for these tasks have regular structures, from which sparse graph patterns can be easily built. Sparse Transformer [25, 4] performs the attention computation only in text subsequences within a shift local block. Similarly, SwinTransformer [18] adopts a sliding window mechanism to compute the multi-level local attention efficiently by making use of high sparsity of local window. Linformer [45] performs linear projection on the dimension of input size, which also requires the ordering of the input elements. Obviously, such methods cannot be directly generalized to cross-image matching tasks, because there is no reasonable way to predefine the order and spatial adjacency of input sparse keypoints.

SGMNet [3] simulates seed downsampling through attentional pooling, with a computational complexity of $O(kNd)$, which is affected by the the number of seeds k, especially for large-scale inputs. Following SGMNet, Suwanwimolkul et al. [39] proposed neighborhood attention with an additional pairwise neighbor layer. ClusterGNN [36] and RoutingTransformer [27] divide the complete graph into multiple subgraphs in terms of semantic similarity by clustering, and then only perform self-attention within each subgraph. Their ideal time complexity of GNNs is $O(N^{1.5}d)$. These methods focus on building appropriate sparse graphs from the input complete graph to simplify the attention computation.

Katharopoulos et al. [14] propose a linear approximation for the attention layer using a kernel function. As shown in Figure 1, the computational complexity is reduced by changing the order of matrix multiplication. Many follow-up works [5, 10, 35] have designed different kinds of kernel functions for different tasks. We draw inspiration from these kernel function approximation based methods to make significant progress on the image matching task.

3. FilterGNN Architecture

3.1. Overview

Given two keypoint sets $(\mathcal{X}_A, \mathcal{X}_B)$ from a pair of images (A, B), the feature matching is to find the correspondences $\mathcal{M} = \{(i, j)\}$ that make the 3D positions of features $\mathcal{X}_A^{(i)}$ and $\mathcal{X}_B^{(j)}$ as close as possible. The input $\mathcal{X}$ is composed of keypoint descriptors $\mathcal{D}$ and positions $\mathcal{P} = \{(u, v, c)\}$, where (u, v) is image coordinates and c represents the corresponding keypoint detection score.

Typically, we can build the similarity score matrix directly using the cosine distances of the visual descriptors, and generate the correspondences through nearest neighbor search. Currently, most visual descriptors, such as SIFT [19], SuperPoint [6], D2net [9], etc., only encode local context. However, in long-term visual localization systems, there are many interference factors, such as lighting conditions, camera poses, repetitive structures (buildings, checkerboards, etc.), that cannot be effectively handled through local descriptors.

Our proposed FilterGNN thus aims to refine the local visual descriptors for feature matching. As shown in Figure 2, our FilterGNN applies optimal Graph Matching Filters in a cascaded manner to the keypoint sets to filter out non-matchable keypoints step-by-step, along with refining the descriptors by aggregating global context through attention cascaded like SuperGlue. Each GMF is dedicatedly designed with a novel linear attention GNN. In this way, the final correspondences can be efficiently and accurately generated by performing traditional nearest neighbor search on the remaining, distilled keypoint sets.

3.2. Graph Matching Filter (GMF)

Our Graph Matching Filter, similar to a lightweight SuperGlue, consists of a shared Positional Encoder, a Linear Attentional GNN and a Top-K Filter. Among them, Positional Encoder is the operator to extract geometric content, and Linear Attentional GNN is designed to aggregate global information. After refining the input feature descriptors, we remove a fixed proportion of non-matchable points according to their cross-image matching scores.

3.2.1 Positional Encoder

The definition of the Positional Encoder module is as follows:

$$\widetilde{\mathcal{F}} = \sigma(\mathcal{D} \oplus \text{mlp}_{pe}(\mathcal{P})), \tag{1}$$

where σ represents the instance normalization [43] and $\oplus$ means matrix addition. $\mathcal{D}$ contains the feature descriptors. mlp_{pe} is the position encoding MLP that performs high-dimensional embedding of the input image coordinates $\mathcal{P}$.

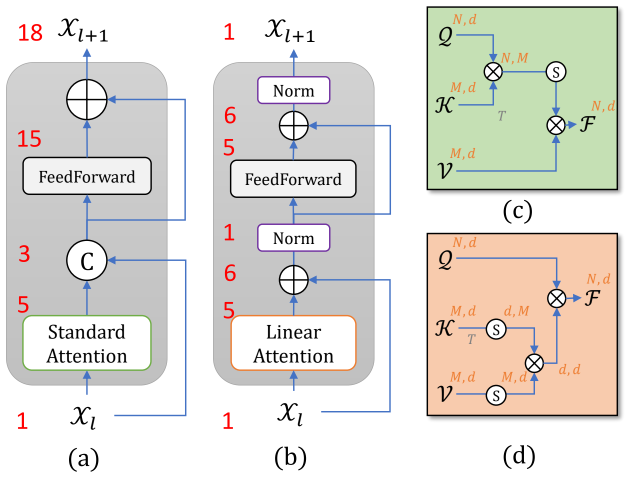


Figure 3. **(a)** Original attention block in SuperGlue. **(b)** Our proposed linear attention block with post-norm. **(c)** Standard attention. **(d)** Linear attention. The red numbers indicate the variance during the operation. The variance multipliers of both the Feed-Forward layer and Attention layer are assumed to be 5.

Here, instance normalization is equipped to control the variance of the output features.

3.2.2 Linear Attentional GNN

Linear Attentional GNN consists of L alternately placed self attention blocks and cross attention blocks. As shown in Figures 3(a)(b), each attention block consists of an linear attention layer and a feed-forward layer. And different from SuperGlue, we place additional normalization layers after the residual computing, respectively. Inspired by Swin TransformerV2 [17], we demonstrate the variance change in Figure 3.

The feed-forward layer is simply defined as a residual MLP mlp_{ff} as follows:

$$FF(X) = \sigma(X \oplus \text{mlp}_{ff}(X)). \tag{2}$$

For self/cross attention block $\widetilde{Att}$, it takes source features $\mathcal{F}_{src} \in \mathbb{R}^{N,d}$ and target feature $\mathcal{F}_{tgt} \in \mathbb{R}^{M,d}$ as inputs and outputs refined aggregated features by adding the attention output to $\mathcal{F}_{src}$. Here, M and N represent the numbers of remaining keypoints of the two images, which are not required to be equal. The detailed definitions of $\widetilde{Att}$ is as follows:

$$\begin{aligned} \widetilde{Att}(\mathcal{F}_{src}, \mathcal{F}_{tgt}) &= \sigma(\mathcal{F}_{src} \oplus Att(\mathcal{Q}, \mathcal{K}, \mathcal{V}) \otimes W_{att}), \\ \mathcal{Q} &= \mathcal{F}_{src} \otimes W_Q \\ \mathcal{K} &= \mathcal{F}_{tgt} \otimes W_K \\ \mathcal{V} &= \mathcal{F}_{tgt} \otimes W_V, \end{aligned} \tag{3}$$

where $W_{att,Q,K,V} \in \mathbb{R}^{d,d}$ represents linear projection weights and $\otimes$ means matrix multiplication. $\mathcal{Q}$, $\mathcal{K}$ and $\mathcal{V}$

correspond to queries, keys and values in the Transformer architecture. When the source and target are the same (e.g., from the same image), $\widetilde{Att}$ performs the self attention; otherwise, it represents the cross-attention.

Attention mechanism is the core of our GNN. As shown in Figure 3(c), for the standard attention mechanism of SuperGlue [31] from vanilla Transformer [44] $Att(\mathcal{Q}, \mathcal{K}, \mathcal{V})$ is defined as:

$$Att_{std}(\mathcal{Q}, \mathcal{K}, \mathcal{V}) = Softmax(\frac{\mathcal{Q} \otimes \mathcal{K}^T}{\sqrt{d}}) \otimes \mathcal{V}, \quad (4)$$

where the computational complexity of $\mathcal{Q} \otimes \mathcal{K}^T$ is $O(N^2 d)$, which is the bottleneck of the whole method.

And the linear attention [14] through kernel function approximation is defined as follows:

$$\begin{aligned} Att_{linear}(\mathcal{Q}, \mathcal{K}, \mathcal{V}) &= \phi(\mathcal{Q}) \otimes (\varphi(\mathcal{K}^T) \otimes \mathcal{V}) \\ \phi(x) = \varphi(x) &= Softmax(x). \end{aligned} \quad (5)$$

(ϕ, φ) also have other choices. The definition in the equation above is borrowed from Efficient Attention [35]. Obviously, the entire computational complexity of Att_{linear} is $O(Nd^2)$. Usually, N is much larger than d. Therefore, $O(N^2 d)$ and $O(Nd^2)$ can also be represented by $O(N^2)$ and $O(N)$, respectively.

As reported by Shi et al. [36], directly training FilterGNN with Att_{linear} from scratch does not make the network converge well. We notice that neither Att_{std} nor Att_{linear} requires extra learning parameters to the network. Therefore, we can adopt a two-step training approach: firstly, we use Att_{std} for pre-training until convergence; secondly, we replace Att_{std} with Att_{linear} to fine-tune the network parameters. In this way, our FilterGNN converges quickly without significant performance degradation. For a more detailed discussion, please refer to Section 4.3.

3.2.3 Top-K Filter

For each GMF module, the score matrix $\mathcal{S} \in \mathbb{R}^{N,M}$ is defined as the dot-product between the refined features as follows:

$$\mathcal{S} = \widetilde{\mathcal{F}}_A \otimes \widetilde{\mathcal{F}}_B^T. \quad (6)$$

We expand $\mathcal{S}$ to $\widetilde{\mathcal{S}} \in \mathbb{R}^{N+1,M+1}$ by adding an extra learnable dustbin dimension for unmatched keypoint detection as SuperGlue [31]. Then, We adopt the Dual-softmax [26] operator to produce optimized matching confidence matrix $\mathcal{C}$ as follows:

$$\mathcal{C} = \log Softmax(\widetilde{\mathcal{S}}) + \log Softmax(\widetilde{\mathcal{S}}^T)^T. \quad (7)$$

Finally, we define the matching probability of each keypoint as row-wise/column wise maximum value of $\mathcal{C}$ (excluding the dustbin dimension), and filter out the lowest k keypoints. k is set to γN in this work. It is worth noting that the NN Search of the last layer in Figure 2 is also a similar process. This step no longer considers the dustbin, and directly determines the predicted matches based on the score matrix. This process is also quite efficient since the large number of outliers have been filtered out in previous layers. As shown in Figures 1 and 4, the detected outliers from GMF are highly related with the co-visibility. A source image would output different outlier detection results with different target images

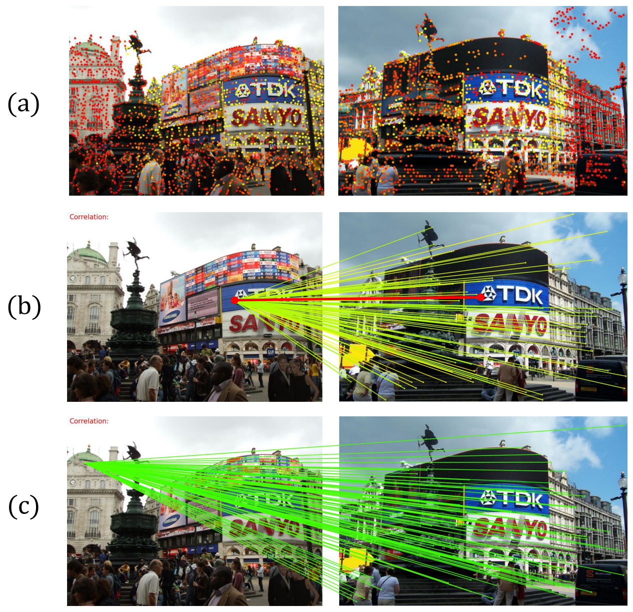


Figure 4. (a) Visualization of matching probabilities: the color of the keypoints goes from yellow to red as the probability goes from high to low. (b) Visualization of matching confidence vector of a matchable keypoint. (c) Visualization of matching confidence vector of a non-matchable keypoint. The color of the lines from red to green indicates the confidence from high to low.

3.3. Loss Function

We adopt a multi-level weighted loss function for H GMF modules, as

$$\mathcal{L} = \sum_h w_h \mathcal{L}_h, w_h = 1 - \gamma h, \quad (8)$$

to achieve fast convergence and ensure stability outlier filtering. γ represents the rejection ratio of each layer, which is set to 0.1 in this work. The matching loss L_h of the h^{th} GMF module is defined as:

$$\begin{aligned} \mathcal{L}_h = &- \frac{1}{|\mathcal{M}|} \sum_{(i,j)\in\mathcal{M}} \mathcal{C}_{ij} - \\ &\frac{1}{|\mathcal{U}_A|} \sum_{i\in\mathcal{U}_A} \mathcal{C}_{i,m+1} - \frac{1}{|\mathcal{U}_B|} \sum_{j\in\mathcal{U}_B} \mathcal{C}_{n+1,j}, \end{aligned} \quad (9)$$

where $\mathcal{M}$ represents the groundtruth matching set, and $(\mathcal{U}_A, \mathcal{U}_B)$ represents the non-matchable keypoint sets. Combined with the definition of $\mathcal{C}$ in Equation 7, we hope that the matching confidence vector of any matchable keypoint is as similar as possible to a one-hot vector (as shown in Figure 4(b)). That is, the cosine similarity between unmatched keypoints should be as low as possible. However, the dimension of features is limited, and as the input size increases, the angle between non-matching points will be squeezed smaller, increasing the possibility of mismatching. Note that our proposed outlier filtering mechanism can provide a relatively wider feature distribution space for deeper layers. Therefore, the validity of FilterGNN is theoretically guaranteed.

4. Experiments and Discussions

4.1. Implementation Details

Training Dataset. FilterGNN was trained with the MegaDepth dataset [16], including 195 outdoor scenes with reconstructed camera poses and depth. We adopted the same training/validation split of 153/36 as reported in [31].

Visual Descriptor. We used ASLfeat [20], a robust indoor/outdoor visual descriptor with dimensions of 128, throughout all experiments. We extract a maximum of 2048 keypoints for each image, and randomly select 1024 for data augmentation during training.

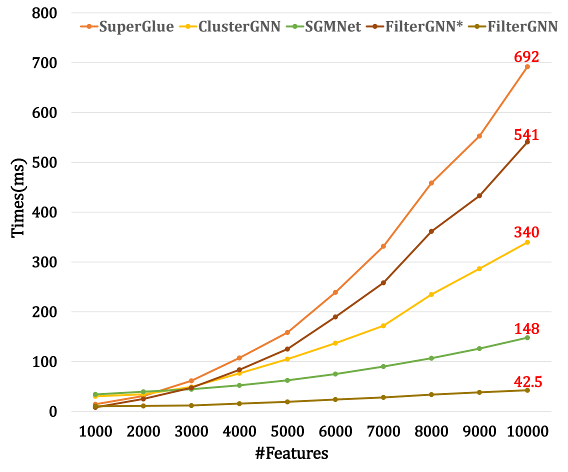
Architecture Details. All feature representations $(\mathcal{Q}, \mathcal{K}, \mathcal{V}, \mathcal{D}, \mathcal{F})$ share the same dimension of 128 as ASLfeat. Both H and L is set to 3. The channels of mlp_{pe} are set to (3, 64, 128, 256, 128), and the channels of mlp_{ff} are set to (128, 512, 128). Each linear layer (excluding the last layer) in both MLPs is followed by a batch normalization and a ReLU layer. All the attention layers mentioned are implemented with 4-heads multi-head attention. Our model is optimized using Adam with an initial learning rate of 1×10^{-4} for first 10 epoch, followed with an exponential decay of 0.9 for 20 epochs.

4.2. Results

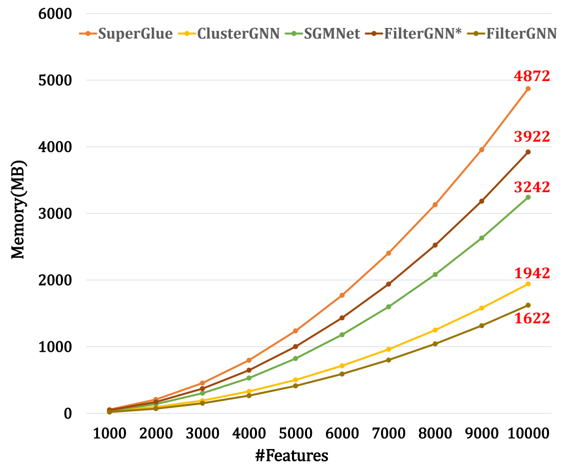
We evaluate the efficiency and performance of our method by making comparison with state-of-the-art methods, including SuperGlue [31], SGMNet [3] and ClusterGNN [36], on various computer vision tasks. ASLfeatV2 [20] is specified as the input image visual descriptor, which is currently one of the best descriptors applicable to both indoor and outdoor scenes.

4.2.1 Efficiency

All experiments were run on the same Nvidia GeForce RTX 3090 GPU. To clearly demonstrate the effect of the cascaded filter mechanism in FilterGNN, we adopt FilterGNN* to represent the corresponding results with standard attention for comparison.



(a) Running time



(b) Memory

Figure 5. Running time (a) and GPU memory consumption (b) w.r.t. the number of input keypoints for different methods.

We first compare the running time and memory usage of the inference phase on a single GPU for the basic task of feature matching. Most statistics are averaged with the same batch size (4 by default). For SuperGlue with 10k input size, batch size is set to 3 to avoid running out of memory. As shown in Figure 5, we report the running time and memory consumption w.r.t. different numbers of input keypoints (ranging from 1K to 10K). The time complexity of our FilterGNN is linear with the input feature size and our method has achieved a significant improvement both in time and memory consumption compared to the state-of-the-art (SOTA) methods. Especially, when the number of input keypoints is 10K, our method only takes 6% time cost and 33.3% memory consumption against SuperGlue. In the reports below, we will use 2K, 4K and 6k input points for different experiments.

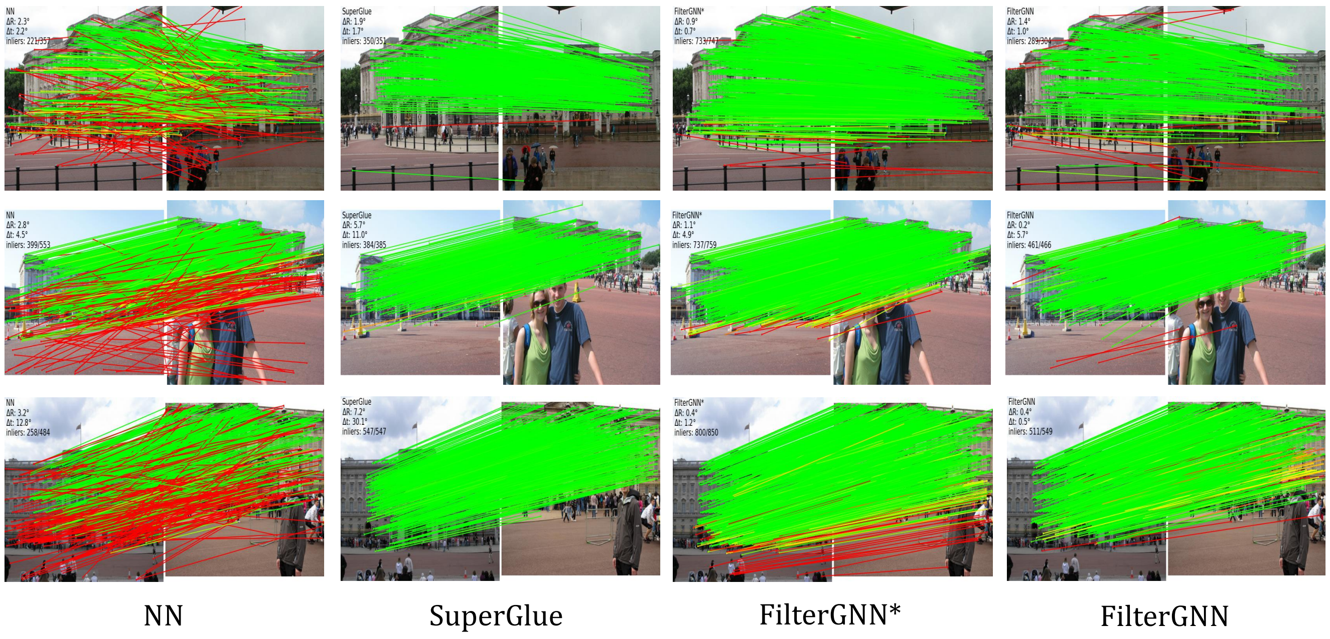


Figure 6. **Qualitative examples of YFCC100M.** The red and green lines indicate the outliers and inliers, respectively. Rotation error, translation error and the number of inliers/matches are shown in the upper left corner of each image.

.

Table 1. **Pose estimation on the YFCC100M benchmark.** The best result is in boldface and the second best is underlined.

Matcher	AUC (↑)			P (↑)	MS (↑)
	5°	10°	20°		
NN	27.95	45.20	61.17	54.29	14.29
SuperGlue	39.92	59.93	76.03	**99.16**	15.55
SGMNet	32.22	52.53	70.16	-	-
ClusterGNN	35.31	56.13	73.56	-	-
FilterGNN	40.91	60.15	75.47	91.26	23.92
FilterGNN*	**44.25**	**63.81**	**78.65**	94.46	**32.59**

4.2.2 Pose Estimation

Camera pose estimation is one of the most important applications for local feature matching, on which the RANSAC post-processing is usually adopted to filter correspondences. Following SuperGlue, we evaluate the accuracy of location estimation on the **YFCC100M [41]** benchmark, which contains 4k test image pairs with groundtruth relative poses and known camera intrinsics. In addition to SuperGlue, SGMNet and ClusterGNN, we also take nearest neighbor search (NN) as the baseline to evaluate the performance of the raw input ASLfeatV2 descriptors. In this experiment the number of input keypoints is set to 2,048. As shown in Table 1, We report the success rate by Area Under Curve (AUC) metric [33, 23, 34] with three different thresholds (i.e, 5°, 10° and 20°), which combine both rotation error and translation error. We also report the precision of matches (P) and the ratio between the number of matches and the input size (MS). All indicators are better for bigger numbers. Our FilterGNN* significantly outperforms current SOTA methods, indicating that our cascaded filter mechanism indeed excludes non-matchable keypoints and increases the number of matches, thus contributing to a more accurate pose estimation. Our FilterGNN using linear attention also achieves very competitive performance compared to SOTA methods, but with less time and memory consumption, as previously demonstrated. It is worth noting that the high prediction accuracy of SuperGlue [31] is built on the fewer predicted matches. It can be infer that SuperGlue [31] is conservative, while FilterGNN is more aggressive and more capable of solving fine-grained problems. We also display some qualitative results in Figure 6.

4.2.3 Visual localization

Visual localization is also an important application of image local feature matching. The typical pipeline contains image retrieval, image matching, and a Perspective-n-Point pose solver. Both the number and accuracy of matches do affect the precision of localization. We integrate FilterGNN into the official HLoc [30] pipeline and run experiments on the Long-Term Visual Localization Benchmark [42]. Specifically, we selected two representative datasets: The Aachen Day-Night dataset [32, 47] (outdoor) and InLoc dataset [40] (indoor).

The Aachen Day-Night dataset has 922 query images, and 824/98 daytime/nighttime images. All images were

Table 2. Outdoor Localization Results on Aachen Day-Night Benchmark(v1.0). The best result is in bold.

Method	Day	Night
	(0.25m,2°) / (0.5m,5°) / (1.0m,10°)	
NN	82.3 / 89.2 / 92.7	67.3 / 79.6 / 85.7
Superglue	87.9 / 95.4 / 98.3	81.6 / 91.8 / 99.0
ClusterGNN	88.6 / **95.5** / 98.4	**85.7** / **93.9** / 99.0
FilterGNN*	**89.2** / 95.4 / 98.5	**85.7** / 92.9 / **100.0**
FilterGNN	88.7 / 95.4 / **98.7**	84.7 / 92.9 / **100.0**

Table 3. Indoor Localization Results on InLoc Dataset. The best result is in bold.

Method	DUC1	DUC2
	(0.25m,10°) / (0.5m,10°) / (1.0m,10°)	
NN	40.4 / 58.1 / 67.7	35.9 / 52.7 / 60.3
SuperGlue	51.5 / 66.7 / 75.8	53.4 / **76.3** / **84.0**
ClusterGNN	52.5 / 68.7 / 76.8	55.0 / 76.0 / 82.4
FilterGNN*	**55.6** / **69.7** / **78.8**	**59.5** / 75.6 / 77.1
FilterGNN	52.5 / 67.7 / 77.8	58.0 / 77.1 / 82.4

taken around the same street in Aachen, where the sparseness of views and day-night variation are the main challenges.

The InLoc dataset provides 329 query images and 9,972 database images. The main challenges include complex lighting conditions and common textureless objects (floor, ceiling, wall, etc.).

The number of input keypoint is set to 4,096. We report the percentage of correctly localized queries under different thresholds (referring to the leader board of Long-Term Visual Localization Benchmark [42]). The results are listed in Tables 2 and 3 for Aachen Day-Night dataset and InLoc dataset, respectively. Our method also achieves comparable results with other SOTA competitors.

4.2.4 Sparse Reconstruction

In order to demonstrate the robustness of FilterGNN more intuitively, we integrate it into the COLMAP [33] pipeline for sparse 3D reconstruction. We extracted 6000 keypoints per image in this task. **THU-MVS dataset [29]** provides multi-view images with 3D groundtruth, consisting of two cases: a) a cat model with 108 views and b) a dog model with 72 views. Animal images are usually weakly textured body surfaces, which brings great challenges to local feature matching. As shown in Figure 7, FilterGNN performs best on both the reconstruction density and accuracy. Compared with SuperGlue, our method increases the reconstruction density by 72% on average, and reduces the reconstruction error by 35%.

4.3. Ablation Study and Discussion

We divide the ablation study into two aspects: submodule composition and structure. First, we report the effect of

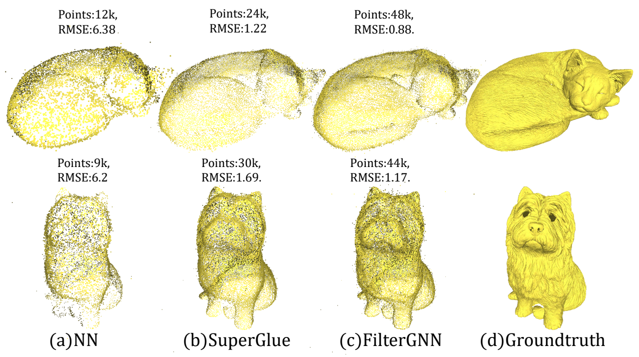


Figure 7. **Visualization of sparse Reconstruction on THU-MVS dataset [29].** The size of reconstruction point cloud and reconstruction error are displayed above corresponding model. RMSE refers to the root mean square error.

post-norm, attention methods and optimal matching function through matching accuracy (precision and recall) comparison on validation part of the MegaDepth dataset [16] (with all training epochs set to 20). Post-norm has two options, layer norm and instance norm. For optimal matching function (Opt.), as reported in previous works [38, 36], both Sinkhorn [37] and Dual-Softmax [26] we have tested work well in most cases. For attention computation, we tested 4 most representative methods:

- **Standard** scale product attention implemented in Transformer [44];
- **Linear Attention** (LA) [14] with both kernel functions (ϕ, φ) set to $1 + elu$;
- **Performer** [5], with an extra low-dimension linear projection and kernel functions (ϕ, φ) set to $(softmax, exponential)$ respectively;
- **Efficient Attention** (EA) [35], also adopted in our work, with both kernel functions (ϕ, φ) set to $softmax$.

Table 4 shows the detailed comparison results. We can draw the following conclusions: i) Instance norm always has a positive effect, while layer norm leads to negative effect. ii) All linear attention methods except Efficient Attention [35] will cause a significant drop, which is much different from reported evaluation on vision tasks or natural language processing tasks. iii) Pretrained with standard attention significantly improves the performance of Efficient Attention [35], but has no significant effect on other methods. Directly adopting the traditional linear attention method would reduce the accuracy rate by 30%, and the pre-trained strategy would reduce the performance drop under 8%. The two-stage pre-training strategy also brings an improvement of about 10%. It is worth noting that in some

Table 4. Comparison among different post-norms, attention methods or optimal matching functions. *LN* and *IN* represent layer norm and instance norm. *sh* and *ds* mean Sinkhorn and Dual-Softmax. “any” means that the choice has no appreciable changes on the results.

Attention	Norm	Pretrain	Opt.	P/R
Standard	-	-	*sh*	78.1/88.5
Standard	-	-	*ds*	81.6/88.2
Standard	*LN*	-	*sh*	67.0/50.0
Standard	*LN*	-	*ds*	NAN
Standard	*IN*	-	any	86.8/87.8
Performer [5]	-	-	any	54.3/59.6
Performer [5]	*IN*	any	any	58.4/62.9
LA [14]	-	-	*sh*	55.9/56.2
LA [14]	-	-	*ds*	NAN
LA [14]	*IN*	any	any	59.3/57.6
EA [35]	-	-	any	62.6/60.9
EA [35]	*IN*	-	any	70.2/74.5
EA [35]	-	✓	any	78.3/79.0
EA [35]	*IN*	✓	any	80.3/80.1

Table 5. Ablation of FilterGNN structure. The bold row represents the default settings.

Methods	H	L	γ	P/R
FilterGNN	3	3	0.1	78.0/81.5
	3	**3**	**0.2**	**80.3/80.1**
	3	3	0.3	92.0/65.7
	3	2	0.2	74.2/78.3
	3	4	0.2	80.5/80.2
	2	3	0.2	77.9/75.7
	4	3	0.2	85.2/72.3

cases, Dual-Softmax will cause abnormal gradients. We speculate that it may be caused by the accumulation of data variance in the residual network, which can be effectively avoided by using instance norm.

Then, we report the effect of the network structure, including the number of GMFs H, the filtering ratio of GMFs γ, and the number of attention blocks L in each attentional GNN module of GPF. As shown in Table 5, increasing the value of H or L from defaults contributes limited improvement. Too large γ will reduce the recall, which is not good for other post-processing tasks.

5. Conclusion

In this paper, we present FilterGNN, an efficient and novel approach for image local feature matching. Observing the high sparsity property of long-term image feature matching, we design hierarchical filter blocks to remove non-matchable keypoints in a cascaded manner. This instant outlier removal mechanism adjusts the network's focus to the keypoints within co-visible regions, which is beneficial for solving fine-grained problems. The experimental results also verify that FilterGNN can indeed substantially increase the number of predicted matches, which is crucial for both accurate visual localization and high-quality sparse reconstruction. Moreover, we also strictly reduce the time complexity from $O(N^2)$ to $O(N)$ by introducing a two-stage pretraining-finetuning strategy without obvious performance degradation, which has been validated in many tasks. Since FilterGNN can reach a running speed of 50Hz with an input of 10K keypoints, in the future, we will apply it to more complex scenes, such as 3D point clouds [11] and super-resolution images, which require more input feature points.

Acknowledgement

This work was supported by the Natural Science Foundation of China (Grant No.: 62220106003).

References

[1] J. L. Ba, J. R. Kiros, and G. E. Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016. 2

[2] V. Balntas, K. Lenc, A. Vedaldi, and K. Mikolajczyk. HPatches: A benchmark and evaluation of handcrafted and learned local descriptors. In *CVPR*, pages 5173–5182, 2017. 1

[3] H. Chen, Z. Luo, J. Zhang, L. Zhou, X. Bai, Z. Hu, C.-L. Tai, and L. Quan. Learning to match features with seeded graph matching network. In *CVPR*, pages 6301–6310, 2021. 1, 3, 6

[4] R. Child, S. Gray, A. Radford, and I. Sutskever. Generating long sequences with sparse transformers. *CoRR*, abs/1904.10509, 2019. 3

[5] K. Choromanski, V. Likhosherstov, D. Dohan, X. Song, A. Gane, T. Sarlos, P. Hawkins, J. Davis, A. Mohiuddin, L. Kaiser, et al. Rethinking attention with performers. *arXiv preprint arXiv:2009.14794*, 2020. 1, 3, 8, 9

[6] D. DeTone, T. Malisiewicz, and A. Rabinovich. SuperPoint: Self-supervised interest point detection and description. In *CVPRW*, pages 224–236, 2018. 1, 2, 4

[7] J. Devlin, M. Chang, K. Lee, and K. Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, pages 4171–4186, 2019. 3

[8] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021. 3

[9] M. Dusmanu, I. Rocco, T. Pajdla, M. Pollefeys, J. Sivic, A. Torii, and T. Sattler. D2-net: A trainable CNN for joint detection and description of local features. In *CVPR*, pages 8092–8101, 2019. 1, 2, 4

[10] Y. Gu, X. Qin, Y. Peng, and L. Li. Content-augmented feature pyramid network with light linear spatial transformers for object detection. *arXiv preprint arXiv:2105.09464*, 2021. 1, 3

[11] J. Guo, H. Wang, Z. Cheng, X. Zhang, and D.-M. Yan. Learning local shape descriptors for computing non-rigid dense correspondence. *Computational Visual Media*, 6:95–112, 2020. 9

[12] M.-H. Guo, T.-X. Xu, J.-J. Liu, Z.-N. Liu, P.-T. Jiang, T.-J. Mu, S.-H. Zhang, R. R. Martin, M.-M. Cheng, and S.-M. Hu. Attention mechanisms in computer vision: A survey. *Computational Visual Media*, 8(3):331–368, 2022. 1

[13] J. Huang, S. Yang, Z. Zhao, Y.-K. Lai, and S.-M. Hu. Clusterslam: A slam backend for simultaneous rigid body clustering and motion estimation. *Computational Visual Media*, 7:87–101, 2021. 1

[14] A. Katharopoulos, A. Vyas, N. Pappas, and F. Fleuret. Transformers are RNNs: Fast autoregressive transformers with linear attention. In *ICML*, pages 5156–5165. PMLR, 2020. 1, 3, 5, 8, 9

[15] N. Kitaev, Ł. Kaiser, and A. Levskaya. Reformer: The efficient transformer. In *ICLR*, 2020. 1

[16] Z. Li and N. Snavely. MegaDepth: Learning single-view depth prediction from internet photos. In *CVPR*, pages 2041–2050, 2018. 6, 8

[17] Z. Liu, H. Hu, Y. Lin, Z. Yao, Z. Xie, Y. Wei, J. Ning, Y. Cao, Z. Zhang, L. Dong, et al. Swin transformer v2: Scaling up capacity and resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12009–12019, 2022. 4

[18] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10012–10022, 2021. 3

[19] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004. 2, 4

[20] Z. Luo, L. Zhou, X. Bai, H. Chen, J. Zhang, Y. Yao, S. Li, T. Fang, and L. Quan. ASLFeat: Learning local features of accurate shape and localization. In *CVPR*, pages 6589–6598, 2020. 2, 6

[21] D. Mishkin, F. Radenovic, and J. Matas. Repeatability is not enough: Learning affine regions via discriminability. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 284–300, 2018. 2

[22] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardós. ORB-SLAM: A versatile and accurate monocular SLAM system. *IEEE Transactions on Robotics*, 31(5):1147–1163, 2015. 1

[23] Y. Ono, E. Trulls, P. Fua, and K. M. Yi. LF-Net: Learning local features from images. In *NIPS*, pages 6237–6247, 2018. 7

[24] N. Parmar, A. Vaswani, J. Uszkoreit, L. Kaiser, N. Shazeer, A. Ku, and D. Tran. Image transformer. In *ICML*, pages 4055–4064. PMLR, 2018. 1

[25] J. Qiu, H. Ma, O. Levy, S. W.-t. Yih, S. Wang, and J. Tang. Blockwise self-attention for long document understanding. In *EMNLP*, pages 2555–2565, 2019. 1, 3

[26] I. Rocco, M. Cimpoi, R. Arandjelović, A. Torii, T. Pajdla, and J. Sivic. Neighbourhood consensus networks. In *NIPS*, pages 1658–1669, 2018. 3, 5, 8

[27] A. Roy, M. Saffar, A. Vaswani, and D. Grangier. Efficient content-based sparse attention with routing transformers. *Transactions of the Association for Computational Linguistics*, 9:53–68, 2021. 3

[28] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. ORB: An efficient alternative to SIFT or SURF. In *ICCV*, pages 2564–2571. Ieee, 2011. 2

[29] S. Sakai, K. Ito, T. Aoki, T. Watanabe, and H. Unten. Phase-based window matching with geometric correction for multi-view stereo. *IEICE TRANSACTIONS on Information and Systems*, 98(10):1818–1828, 2015. 8

[30] P.-E. Sarlin, C. Cadena, R. Siegwart, and M. Dymczyk. From coarse to fine: Robust hierarchical localization at large scale. In *CVPR*, pages 12716–12725, 2019. 1, 7

[31] P.-E. Sarlin, D. DeTone, T. Malisiewicz, and A. Rabinovich. SuperGlue: Learning feature matching with graph neural networks. In *CVPR*, pages 4938–4947, 2020. 1, 2, 3, 5, 6, 7

[32] T. Sattler, T. Weyand, B. Leibe, and L. Kobbelt. Image retrieval for image-based localization revisited. In *BMVC*, volume 1, page 4, 2012. 1, 7

[33] J. L. Schonberger and J.-M. Frahm. Structure-from-motion revisited. In *CVPR*, pages 4104–4113, 2016. 1, 7, 8

[34] J. L. Schönberger, E. Zheng, J.-M. Frahm, and M. Pollefeys. Pixelwise view selection for unstructured multi-view stereo. In *ECCV*, pages 501–518. Springer, 2016. 7

[35] Z. Shen, M. Zhang, H. Zhao, S. Yi, and H. Li. Efficient attention: Attention with linear complexities. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 3531–3539, 2021. 3, 5, 8, 9

[36] Y. Shi, J.-X. Cai, Y. Shavit, T.-J. Mu, W. Feng, and K. Zhang. ClusterGNN: Cluster-based coarse-to-fine graph neural network for efficient feature matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12517–12526, 2022. 1, 2, 3, 5, 6, 8

[37] R. Sinkhorn and P. Knopp. Concerning nonnegative matrices and doubly stochastic matrices. *Pacific Journal of Mathematics*, 21(2):343–348, 1967. 8

[38] J. Sun, Z. Shen, Y. Wang, H. Bao, and X. Zhou. LoFTR: Detector-free local feature matching with transformers. In *CVPR*, pages 8922–8931, 2021. 1, 8

[39] S. Suwanwimolkul and S. Komorita. Efficient linear attention for fast and accurate keypoint matching. In *Proceedings of the 2022 International Conference on Multimedia Retrieval*, pages 330–341, 2022. 1, 3

[40] H. Taira, M. Okutomi, T. Sattler, M. Cimpoi, M. Pollefeys, J. Sivic, T. Pajdla, and A. Torii. InLoc: Indoor visual localization with dense matching and view synthesis. In *CVPR*, pages 7199–7209, 2018. 1, 7

[41] B. Thomee, D. A. Shamma, G. Friedland, B. Elizalde, K. Ni, D. Poland, D. Borth, and L.-J. Li. YFCC100M: The new data in multimedia research. *Communications of the ACM*, 59(2):64–73, 2016. 1, 7

[42] C. Toft, W. Maddern, A. Torii, L. Hammarstrand, E. Stenborg, D. Safari, M. Okutomi, M. Pollefeys, J. Sivic, T. Pajdla, et al. Long-term visual localization revisited. *TPAMI*, 2020. 7, 8

[43] D. Ulyanov, A. Vedaldi, and V. Lempitsky. Instance normalization: The missing ingredient for fast stylization. *arXiv preprint arXiv:1607.08022*, 2016. 2, 4

[44] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *NeurIPS*, pages 5998–6008, 2017. 1, 2, 5, 8

[45] S. Wang, B. Z. Li, M. Khabsa, H. Fang, and H. Ma. Linformer: Self-attention with linear complexity. *CoRR*, abs/2006.04768, 2020. 1, 3

[46] K. M. Yi, E. Trulls, V. Lepetit, and P. Fua. LIFT: Learned invariant feature transform. In *ECCV*, pages 467–483. Springer, 2016. 2

[47] Z. Zhang, T. Sattler, and D. Scaramuzza. Reference pose generation for visual localization via learned features and view synthesis. *IJCV*, 129(4):821–844, 2021. 1, 7