


Article

AdaSG: A Lightweight Feature Point Matching Method Using Adaptive Descriptor with GNN for VSLAM

Ye Liu ¹, Kun Huang ¹, Jingyuan Li ¹, Xiangting Li ¹, Zeng Zeng ², Liang Chang ¹  and Jun Zhou ^{1,*}

¹ School of Information and Communication Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China

² School of Microelectronics, Shanghai University, Shanghai 200444, China

* Correspondence: zhouj@uestc.edu.cn

Abstract: Feature point matching is a key component in visual simultaneous localization and mapping (VSLAM). Recently, the neural network has been employed in the feature point matching to improve matching performance. Among the state-of-the-art feature point matching methods, the SuperGlue is one of the top methods and ranked the first in the CVPR 2020 workshop on image matching. However, this method utilizes graph neural network (GNN), resulting in large computational complexity, which makes it unsuitable for resource-constrained devices, such as robots and mobile phones. In this work, we propose a lightweight feature point matching method based on the SuperGlue (named as AdaSG). Compared to the SuperGlue, the AdaSG adaptively adjusts its operating architecture according to the similarity of input image pair to reduce the computational complexity while achieving high matching performance. The proposed method has been evaluated through the commonly used datasets, including indoor and outdoor environments. Compared with several state-of-the-art feature point matching methods, the proposed method achieves significantly less runtime (up to 43× for indoor and up to 6× for outdoor) with similar or better matching performance. It is suitable for feature point matching in resource constrained devices.

Keywords: feature point matching; GNN; VSLAM



Citation: Liu, Y.; Huang, K.; Li, J.; Li, X.; Zeng, Z.; Chang, L.; Zhou, J. AdaSG: A Lightweight Feature Point Matching Method Using Adaptive Descriptor with GNN for VSLAM. *Sensors* **2022**, *22*, 5992. <https://doi.org/10.3390/s22165992>

Academic Editors: Fengwei An, Hong-Ning Dai, Po Yang and Ping Li

Received: 6 July 2022

Accepted: 8 August 2022

Published: 11 August 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Feature point matching [1,2] is a key component for pose estimation in VSLAM, which is widely used in applications, such as autonomous driving, robots, and wearable augmented reality (AR) [3–6]. Pose estimation based on feature points mainly consists of two parts: feature point extraction and feature point matching. The former extracts the feature points from two stereo or consecutive images and represents them using certain descriptors [7–9]. The latter computes the distance of the feature points based on the descriptors [10] and performs feature point matching to estimate the pose (as shown in Figure 1). Most of the conventional feature point matching methods use the nearest neighbor method [11] to compute the distance between the feature points and perform the matching.

One of the major challenges for the feature point matching is that it is prone to errors due to numerous factors, such as viewpoint change, lack of texture, illumination variation, and blur. To address this challenge, various methods have been proposed, such as Lowe's ratio test [12], mutual checks, and neighborhood consensus [13,14], which combines the nearest neighbor with outlier rejection methods.

In recent years, neural networks [15,16] have been utilized to enhance the quality of the descriptors by learning from a large number of datasets to improve matching performance. One of the most state-of-the-art methods is the SuperGlue [17], which is inspired by Transformer [18], and utilizes the GNN with attention mechanism to improve the matching performance. It ranked the first in the CVPR 2020 workshop on image matching [19].

Despite its superior performance, the SuperGlue brings in large computational complexity, making it unsuitable for resource-constrained devices, such as robots and mobile phones.

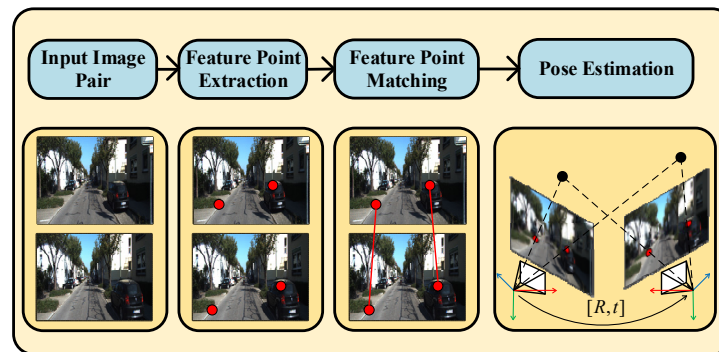


Figure 1. The operating flow of pose estimation in VSLAM. The red points represent the feature points in the images and the black points represent the corresponding 3D points in the real world.

There are some lightweight methods proposed to improve the heavy-weight algorithm. Adjusting the neural network, which is possible for many neural networks, is usually used to reduce parameters [20–22]. Moreover, taking advantage of the characteristics of some specific application scene can always reduce computational complexity [23,24].

Inspired by these methods, in this work, a lightweight feature point matching method (named AdaSG) based on SuperGlue has been proposed in order to address the above issue. It adaptively adjusts the descriptor and the outlier rejection mechanism according to the similarity of the input image pair so as to reduce the computational complexity while maintaining good matching performance.

The remaining part of this paper is organized as follows: Section 2 reviews the existing works on feature point matching. Section 3 presents the proposed AdaSG method. Section 4 discusses the experimental results, and Section 5 concludes the paper.

2. Related Work

The feature point matching method mainly consists of two steps: distance calculation and outlier rejection [12]. The former is responsible for calculating the distance between different feature points based on their descriptors by using certain distance metrics, such as Euclidean distance [25], inner product [26], or Hamming distance [27]. Based on the distance, the outlier rejection filters out the feature points with large distances, which are defined as outliers. The most common outlier rejection method is the nearest neighbor, which sets a threshold to select the closest feature points. However, this method does not work well in certain scenarios, such as viewpoint change, lack of texture, illumination variation, and blur. To address these challenges, the revised outlier rejection methods have been proposed. For example, Lowe’s ratio test was proposed in [12]. In this work, the outlier rejection takes the distance ratio of the closest neighbor and the second closest neighbor into account. The correct matchings are determined by comparing this ratio with a certain threshold. In [13], the neighborhood consensus was proposed, in which the invariant regions are extracted to facilitate the matching when the viewpoint changes. However, in some challenging scenarios with significant viewpoint changes, it is still difficult to obtain correct matching using these methods.

To deal with this challenge, some works started using neural networks to enhance the descriptors of feature points to obtain better matching results [28–30]. These studies mainly use the convolutional neural networks (CNN) to obtain better descriptors than the handcrafted descriptors from feature point extraction. In [31], LF-Net, which is composed of two stages networks, was proposed to extract feature points location and descriptors. The former detects feature points and the latter computes descriptors. In [32], an encoder–decoder architecture was trained by using homographic adaption to learn multi-scale features. This work not only enhances the descriptors but also boosts feature point de-

tection. In [33], D2-Net was proposed to extract the feature points distribution and their descriptors by using a shared CNN. This describe-and-detect strategy performs well in the aforementioned challenging scenarios. In [34], R2D2 was proposed to jointly learn reliable detectors and descriptors. This work discovered that descriptors learned in discriminative regions help improve the matching performance.

Other studies leverage regional information to improve the discrimination of descriptors, such as regional descriptors [35] or log-polar patches [36]. In [35], the generation of descriptors utilizes not only visual context from high-level image representation but also geometric context from 2D feature point distribution. In [36], the local region information is extracted with a log-polar sampling scheme to acquire better descriptors, which oversamples the adjacent regions of the feature points and undersamples the regions far away from it.

Although the neural network-based methods have improved the matching performance to some extent, they have not achieved satisfactory results. In these works, there is an issue limiting their performance. These works focus on pixel-level information but lack higher-level information which reflects the correspondences between the feature points. In fact, feature point matching can be seen as an instance of graph matching. A graph can be constructed by associating each feature point to a node and defining special edges. How to leverage the knowledge in the graph domain has become an interesting topic.

Recently, GNN [17,37–39] has been utilized to address this issue. In [37], the PointNet was proposed to learn the information of both global and local feature points for 3D recognition tasks. PointNet architecture is a special case of GNN but without definition of edges. Inspired by [37], PointCN [38] was proposed to deal with feature point matching by classifying inliers and outliers. In this work, a novel normalization technique is proposed to process each feature point separately with embedded global information. The global information is generated by taking the camera motion into account. However, PointNet-like architecture cannot capture the local context information. Based on [38], OANet [39] was proposed to capture the local context information by exploiting differentiable pooling which is used to process an entire graph. In the meanwhile, the global context information is also captured by an Order-Aware Filtering block with spatial association. Based on these works, the SuperGlue was recently proposed, which applies an attentional GNN to aggregate information based on self-cross attention mechanism [17]. This method learned complex priors and elegantly solved occlusion and non-repeatable feature points. The SuperGlue ranked first in the CVPR 2020 workshop on image matching [19].

The methods based on neural network outperform conventional methods by enhancing the descriptors. However, this significantly increases the computational complexity, making them unsuitable for resource-constrained devices, such as robots and mobile phones. Therefore, how to reduce the computational complexity while maintaining good matching performance needs to be investigated.

Some work has proposed lightweight methods to reduce computational complexity. Exploring more lightweight neural network is usually used to reduce parameters. In [20], MobileNet is proposed to build a lightweight deep neural network. Moreover, leveraging the characteristics of certain specific scenarios can always reduce computational complexity. [23] uses a small, annotated dataset to build a prediction model, which avoids using complex neural networks to complete tasks. In [24], ORBSLAM2 uses a constant velocity movement model to reduce matching complexity during tracking threads.

3. Proposed AdaSG

We have chosen the SuperGlue as the baseline of our proposed method due to its excellent performance. The SuperGlue consists of two major blocks: an attentional GNN and an optimal matching layer, as shown in Figure 2. The attentional GNN carries out two strategies to obtain better descriptors. One is to integrate more contextual cues (e.g., the position of feature points) to increase the distinctiveness of original descriptors. The other is to use alternating self- and cross-attention layers to resolve ambiguities. This strategy,

inspired by humans, integrates more knowledge between both images during matching. When asked to match feature points in two images, humans usually look back-and-forth between the two images until finding a difference or similarity.

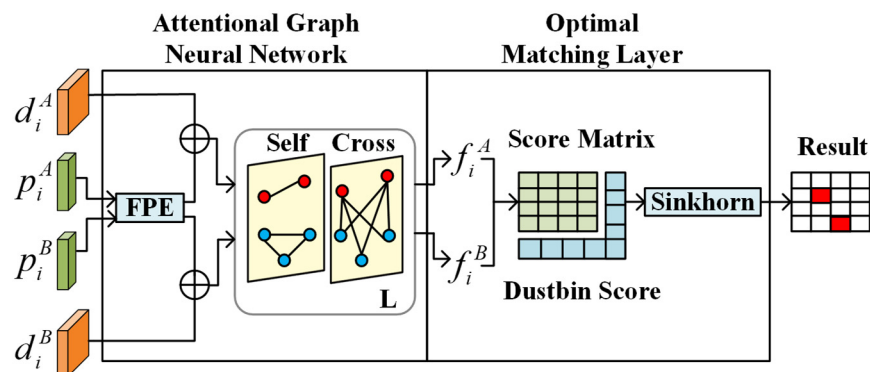


Figure 2. The architecture of SuperGlue. SuperGlue uses a keypoint encoder, which is also called a feature point encoder (FPE), to fuse contextual cues (keypoint position p and descriptor d) and then uses alternating self- and cross-attention layers (repeated L times) to obtain matching descriptors f .

After that, the optimal matching layer performs iterative optimization to obtain the matching results, which is represented as an assignment matrix. The assignment matrix is obtained by maximizing the score function as in normal optimization problems through iterative procedure. A dustbin mechanism is used to assign unmatched feature points to reduce incorrect matching. This optimization problem can be regarded as an optimal transport problem. Therefore, the solution is a Sinkhorn algorithm, which is a commonly used efficient algorithm for optimal transport problem.

Despite its excellent matching performance, the SuperGlue brings in large computational complexity in two aspects. Firstly, the SuperGlue treats all the input images equally. In order to achieve good matching performance, the images are always processed using GNN regardless of the similarity of the input image pair. However, not all images are difficult to match and require a complex GNN to enhance the quality of the descriptors. For the input images that are easy to match, a simplified method can significantly reduce the computational complexity while achieving a similar performance to the SuperGlue.

Secondly, the optimal matching layer brings in additional computational complexity due to the iteration of Sinkhorn algorithm. The runtime of Sinkhorn with iteration almost accounts for half of the overall runtime. How to reduce the complexity of the optimal matching layer is worth investigating.

To address the above issue, in this work, we propose Adaptive SuperGlue (AdaSG in short), which is a lightweight feature point matching method. In this method, the descriptor and the outlier rejection mechanism are adaptively adjusted according to the similarity of the input image pairs so as to reduce the computational complexity, while achieving good matching performance.

AdaSG is mainly composed of three modules: a similarity evaluation module, an adaptive aggregation module and an adaptive outlier rejection module, as shown in Figure 3.

3.1. Similarity Evaluation Module

The similarity evaluation module is responsible for evaluating the similarity of the input image pair. If the similarity of the two images is low (e.g., as shown in Figure 4), meaning that a large viewpoint change or illumination change is present, then the matching difficulty is considered high. In this case, the AdaSG will turn on the D-Mode. In this mode, the adaptive aggregation module will activate the GNN for the descriptor enhancement, and the outlier rejection module will increase the optimization strength to improve the matching performance.

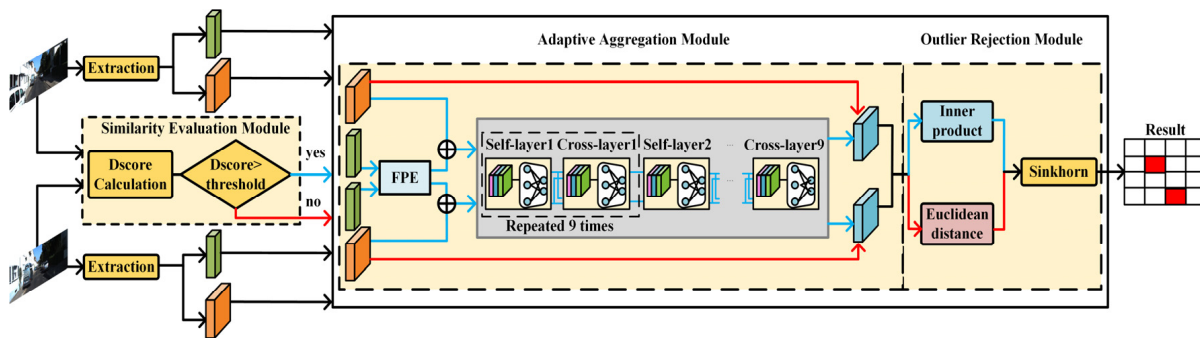


Figure 3. The architecture of the proposed AdaSG. Red lines represent the E-mode dataflow and blue lines represent the D-mode dataflow. Self-layering and cross-layering were repeated nine times alternatively.



(a)



(b)

Figure 4. The image pair with low similarity. (a,b) are pictures of a landmark location, but the camera angle and image content are very different.

If the similarity of the two images is high (e.g., as shown in Figure 5), meaning that the viewpoint change or illumination change is small, then the matching difficulty is considered low. In this case, the AdaSG will turn on the E-Mode. In this mode, the aggregation module will de-activate the GNN to skip the descriptor enhancement and the outlier rejection module will lower the optimization strength to reduce the computational complexity.

The similarity of the two images is calculated using the sum of absolute differences (SAD), as shown in (1).

$$Dscore = \sum_{i=1}^W \sum_{j=1}^H |I_A(i, j) - I_B(i, j)| \tag{1}$$

where $Dscore$ represents the similarity and $I_A(i, j)$ and $I_B(i, j)$ represent the pixel grayscale value of image A and image B at (i, j) .



Figure 5. The image pair with high similarity. (a,b) are two images extracted from the video stream, which look almost identical.

3.2. Adaptive Aggregation Module

The adaptive aggregation module is used to enhance the input descriptors by activating the GNN to improve the matching performance according to the work modes (i.e., D-mode and E-mode). The architecture of the adaptive aggregation module is shown in Figure 6 and its detailed operation is described below.

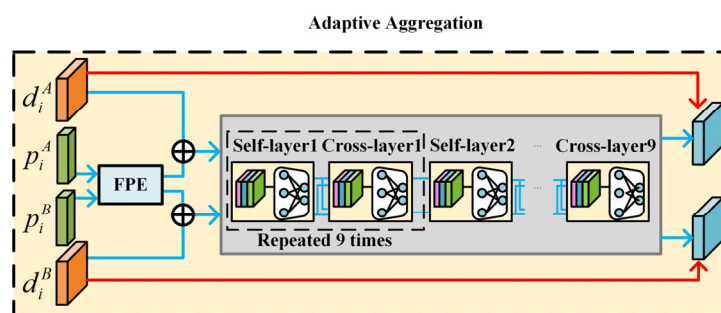


Figure 6. The architecture of adaptive aggregation module. The red lines represent E-mode dataflow and the blue lines represent D-mode dataflow. Self-layering and cross-layering are repeated nine times alternatively.

The inputs of the adaptive aggregation module include the descriptors and the location information of the feature points from the image pair. A feature point encoder (FPE) is used to fuse the descriptors and the location information according to (2).

$${}^{(0)}x_i = d_i + MLP(p_i) \quad (2)$$

where ${}^{(0)}x_i$ represents the new descriptor; d_i represents the original descriptor; p_i represents the location information of the feature point, which is composed of the coordinates and confidence of the feature point; and Multilayer Perceptron (*MLP*) represents a neural network composed of fully connected layers.

After the FPE, a *GNN* is used to enhance the descriptor of feature points by aggregating the information of the descriptors based on the graph structure with special edges defined by attention mechanism as in the SuperGlue [17]. The architecture of the *GNN* is shown in Figure 7. The *GNN* is mainly composed of a self-layer and cross-layer alternatively, which repeats nine times. The architecture of the self-layer and cross-layer are the same, which leverages the attention mechanism to obtain an aggregation message. However, their inputs are different. The inputs of the self-layers are from one of the input images while the inputs of the cross-layer are from both input images. Through the self-cross mechanism, both self-image and cross-image information are aggregated to improve the discriminativeness and matching performance.

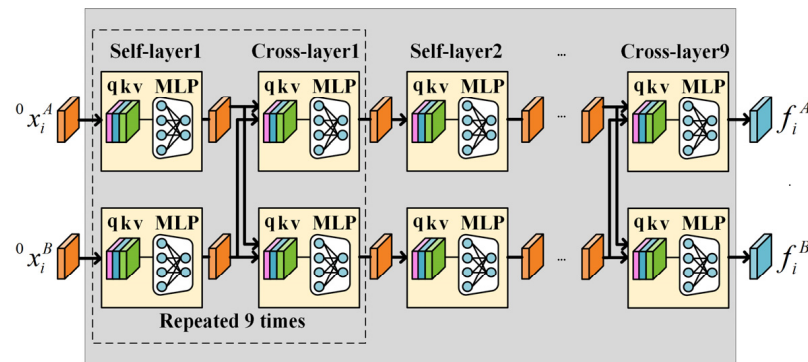


Figure 7. The architecture of *GNN*. The *q*, *k*, and *v* represent the query, key, and value in the attention mechanism. *x* is obtained by FPE. The self-layer and cross-layer are repeated nine times alternatively.

Different from [17], which uses *GNN* to enhance the descriptors for all the input images, in the proposed adaptive aggregation method, the descriptor is adaptively generated according to the work mode, which is determined by the similarity evaluation module from the input images. For the D-mode, the *GNN* and FPE are activated to enhance the descriptors. In this case, the descriptors are the output of the *GNN*, as shown in (3).

$$\begin{aligned} f_i^A &= GNN\left({}^{(0)}x_A, {}^{(0)}x_B\right), \forall i \in A \\ f_j^B &= GNN\left({}^{(0)}x_B, {}^{(0)}x_A\right), \forall j \in B \end{aligned} \quad (3)$$

where ${}^{(0)}x_A$ and ${}^{(0)}x_B$ indicate the output of FPE for the input image pair (A, B), respectively, and f_i^A and f_j^B are descriptors generated by the *GNN* from the image A and B.

For the E-mode, the *GNN* and FPE will be de-activated for skipping the descriptor enhancement. In this case, the descriptors are the original descriptors, which are good enough for matching as the two images are very similar, as shown in (4).

$$\begin{aligned} f_i^A &= d_i, \forall i \in A \\ f_j^B &= d_j, \forall j \in B \end{aligned} \quad (4)$$

where f_i^A and f_j^B are the descriptors of image A and image B. The (3) and (4) can be merged and represented using (5).

$$\begin{aligned} f_i^A &= \gamma * GNN\left({}^{(0)}x_A, {}^{(0)}x_B\right) + (1 - \gamma) * d_i, \forall i \in A \\ f_j^B &= \gamma * GNN\left({}^{(0)}x_B, {}^{(0)}x_A\right) + (1 - \gamma) * d_j, \forall j \in B \end{aligned} \quad (5)$$

where $\gamma \in \{0, 1\}$, $\gamma = 0$ indicates that E-mode is selected, while $\gamma = 1$ indicates that D-mode is selected.

3.3. Outlier Rejection Module

In the SuperGlue, the outlier rejection module employs Sinkhorn and mutual-check to filter out incorrect matching based on the distance matrix. Different from the SuperGlue, in this work, we proposed adaptive Sinkhorn iterations to reduce complexity while maintaining the performance. Through experiments we found that in the E-mode, the Sinkhorn, with a few iterations (e.g., 3), can achieve high accuracy, while in the D-mode, many more iterations (e.g., 100) are needed for achieving high accuracy. Therefore, the number of iterations can also be adapted according to the work mode. In the E-mode, the Sinkhorn iteration is set to a smaller number, while it is set to a larger number in the D-mode. Additionally, the mutual-check mechanism, which is used to calculate the distance of feature points and select similar feature points, can be adapted according to the work mode.

In the D-mode, the original descriptor vector has been transformed by the GNN, therefore the inner product is used to compute the distance matrix S , as shown in (6). The i th feature point in image A and the j th feature point in image B can be matched if $S_{i,j}$ is the maximum both in the i th row and j th column of S and larger than the point selection threshold.

$$S_{i,j} = \langle f_i^A, f_j^B \rangle, \forall (i, j) \in A \times B \quad (6)$$

where $\langle \rangle$ represents the inner product. f_i^A represents the i th descriptor of image A and f_j^B represents the j th descriptor of image B. $S_{i,j}$ represents the element in the S at (i, j) .

In the E-mode, without GNN the Euclidean distance can be used to compute the distance of original descriptor vectors, which is also shown in (7). The i th feature point in image A and the j th feature point in image B can be matched if $S_{i,j}$ is the minimum both in i th row and j th column of S and less than the point selection threshold.

$$S_{i,j} = \sqrt{(f_i^A - f_j^B)^T (f_i^A - f_j^B)}, \forall (i, j) \in A \times B \quad (7)$$

4. Experimental Results and Analysis

4.1. Experimental Setup

The proposed AdaSG is implemented using PyTorch and run on an NVIDIA Tesla v100 GPU. As in the SuperGlue paper [17], two datasets are used to evaluate the performance of the proposed method, including the indoor dataset Scannet [40] and the outdoor dataset YFCC100M [41]. The Scannet is a large-scale indoor dataset composed of sequence images with ground truth poses. YFCC100M is a large-scale outdoor dataset composed of non-sequence images with ground truth poses. These two datasets can be used to evaluate the performance of the matching method for indoor and outdoor scenarios. However, YFCC100M does not contain sequence images. Therefore, we also used KITTI [42], which is also a large-scale outdoor dataset with sequence images. Both YFCC100M and KITTI have been used to evaluate the proposed AdaSG for outdoor scenarios. For the Scannet and the YFCC100M datasets, all the sequences tested in the SuperGlue paper [17] are used to evaluate the performance. For the KITTI dataset, all the images are used for evaluation to investigate the performance on sequence images.

Like the previous work [17,38,39], the area under the cumulative error curve (AUC) of the pose error at different threshold is used to evaluate the performance of the proposed method. The pose error is the maximum of angle error between rotation and translation. The cumulative error curve is generated by calculating the precision. Then $AUC@m$ indicates the area under this curve up to a maximum threshold m . In this experiment, m is set to 5° , 10° , and 20° .

The matching precision (P) and the matching score (MS) are also presented to measure the matching performance based on its epipolar distance. The matching is correct if its epipolar distance is less than 10^{-4} . The matching precision is the percentage of the correct matched feature point pairs in all the matched feature point pairs, and the matching score

is the percentage of the correct matched feature point pairs in all the feature point pairs, including matched and unmatched pairs.

As mentioned previously, in the proposed AdaSG method, three threshold parameters need to be set, including the threshold of similarity of the input image pair, T_S ; the point selection threshold in D-mode, T_D ; and the point selection threshold in E-mode, T_E . For the experiments, these parameters are set to 0.12, 0.2, and 0.8, respectively. In the E-mode the Sinkhorn iteration T_i is set to 0, and in the D-mode the Sinkhorn iteration T_i is set to 100.

The parameters of SuperGlue and AdaSG are shown in Table 1. SuperGlue contains a Keypoint Encoder, which has 100 k parameters and an attentional GNN with 12 M parameters. Compared to SuperGlue, AdaSG has zero parameters in E-mode and has the same parameters as SuperGlue in D-mode.

Table 1. Parameter of SuperGlue and AdaSG.

Method	Parameters
SuperGlue	12.1 M
AdaSG (E-mode)	0
AdaSG (D-mode)	12.1 M

For benchmarking, the proposed AdaSG is compared with the state-of-the-art matching methods, including the SuperGlue [17], the PointCN [38], the OANet [39], and the GlusterGNN [43]. The OANet and the PointCN are both combined with the nearest neighbor method for testing. The GlusterGNN is only compared on the YFCC100M, which is the same as in [43], for fair comparison. Firstly, as the compared methods use different datasets and evaluation metrics in their papers, for fair comparison, we obtained the source codes of the compared methods and re-ran them using the same datasets and evaluation metrics, as in [17]. Secondly, as the compared methods are matching methods, for the feature point extraction before the matching we used the SuperPoint, which is a state-of-the-art feature point extraction method [32] for all the compared methods. In addition, for fair comparison with the SuperGlue, the weights of the GNN in the proposed AdaSG use the same weights from the SuperGlue paper [17].

Furthermore, to investigate and compare the performance of the AdaSG and the SuperGlue on resource-constrained devices, we implemented the AdaSG and the SuperGlue on an embedded system board (i.e., RK3399Pro [44]), which has four ARM-A53 cores at 1.4 GHz with 4 Gb DDR3 for running the algorithms, as shown in Figure 8. In the experiments, the images are sent from the laptop to the board for feature point extraction and matching, and the matching results are sent to the monitor for display.

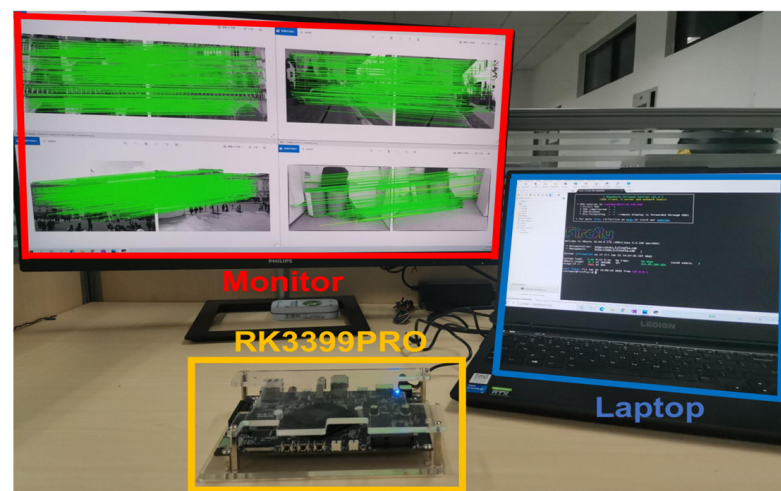


Figure 8. The whole system.

4.2. Indoor Dataset Experimental Results

Indoor matching is challenging because there are many walls, floors, and other areas where the texture information is not rich. Additionally, there are many objects with high similarity, such as doors and rooms, which can easily cause mismatch. Figure 9 and Table 2 show indoor dataset experimental results using the Scannet dataset. As can be seen in the table, the performance of the proposed AdaSG is similar to that of the SuperGlue and is better than the other methods. However, its average runtime is significantly shorter than that of the SuperGlue ($43\times$). This is mainly due to the fact that AdaSG adaptively adjusts its architecture according to the similarity of the input image pair which significantly reduces the computational complexity.

Table 2. Indoor Experiment on the Scannet dataset.

Matcher	AUC (%)			P (%)	MS (%)	Average Runtime (s)
	@5°	@10°	@20°			
OANet	0.15	0.71	2.35	96.28	44.56	0.016
PointCN	0.28	0.64	2.18	97.38	41.67	0.006
SuperGlue	0.37	1.36	4.82	96.62	76.30	0.043
AdaSG	0.38	1.38	4.79	96.48	75.59	0.001

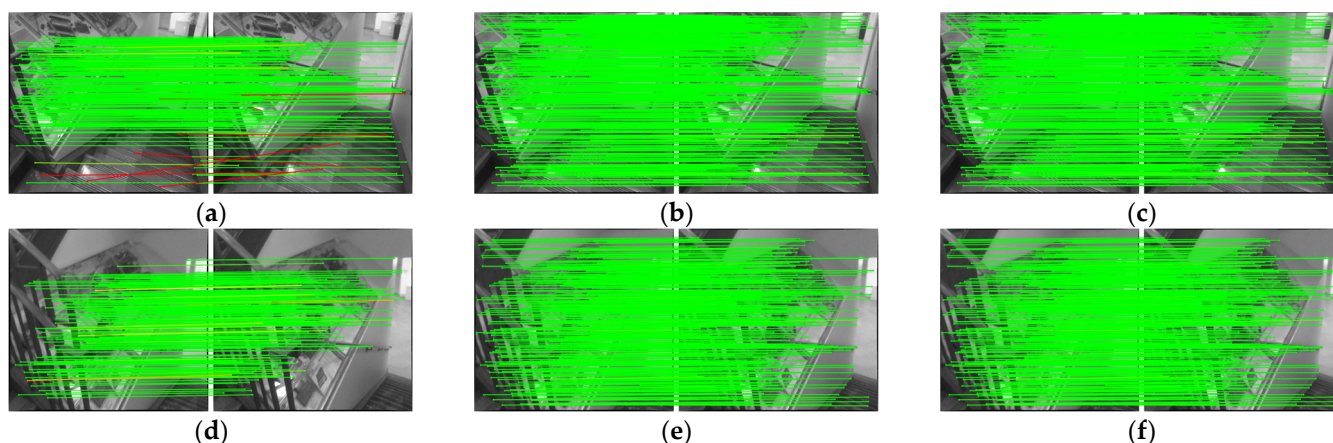


Figure 9. The visualized matching result on Scannet. The red lines represent mismatches, and the green lines represent correct matching. (a) The results of OANet. (b) The results of SuperGlue. (c) The results of AdaSG. (d) The other results of OANet. (e) The other results of SuperGlue. (f) The other results of AdaSG.

We also investigated the impact of threshold of similarity on the matching precision and average runtime by varying the threshold value. The results are shown in Figure 10. It can be seen from the figure that as the threshold value increases, the matching precision decreases and the runtime decreases. When the threshold increases from 0 to 0.12, which is the selected threshold value, the average runtime decreases by 97.67%, while the matching precision only decreases slightly from 96.62% to 96.48%. The average runtime and matching precision saturate at 0.001 s and 96.48%, respectively, when the threshold exceeds 0.1, because with such large threshold all the input image pairs are consider similar and the E-mode is activated.

4.3. Outdoor Dataset Experimental Results

Figure 11 and Table 3 show the outdoor dataset experimental results using YFCC100M. As can be seen from the table, the performance of the AdaSG is similar to that of the SuperGlue and is significantly better than the other methods. However, its average runtime is also similar to that of the SuperGlue, which is longer than the other methods. This is mainly due to the fact that the YFCC100M dataset mainly contains non-sequence images

and in this case D-mode is activated much more frequently to improve the matching performance at the cost of runtime.

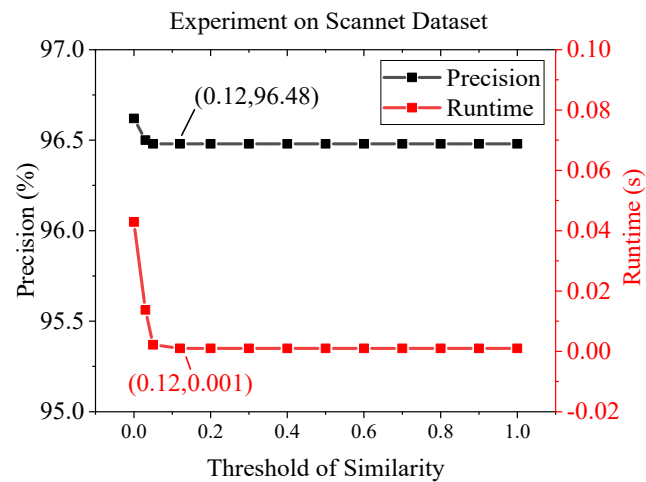


Figure 10. The impact of threshold of similarity on matching precision and average runtime on the Scannet dataset.

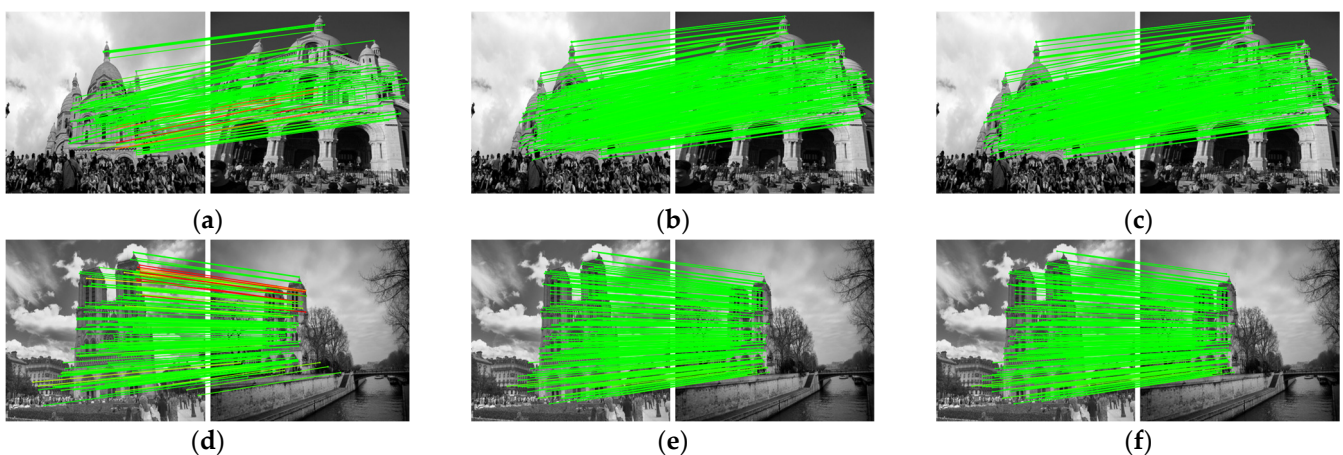


Figure 11. The visualized matching result on YFCC100M. The red lines represent mismatches, and the green lines represent correct matching. (a) The results of OANet. (b) The results of SuperGlue. (c) The results of AdaSG. (d) The other results of OANet. (e) The other results of SuperGlue. (f) The other results of AdaSG.

Table 3. Outdoor experiment on the YFCC100M dataset.

Matcher	AUC (%)			P (%)	MS (%)	Average Runtime (s)
	@5°	@10°	@20°			
OANet	23.92	41.47	58.65	84.24	15.58	0.021
PointCN	22.04	38.54	55.67	73.13	17.06	0.004
SuperGlue	37.94	58.3	74.59	97.74	23.00	0.092
ClusterGNN	35.31	56.13	73.56	N/A	N/A	N/A
AdaSG	37.90	58.23	74.52	97.64	22.91	0.088

Figure 12 and Table 4 show the outdoor dataset experimental results using KITTI. The KITTI contains sequence images, but the moving speed is quite high as the images are taken in a moving car. Therefore, it is a mixture of similar and non-similar image pairs. This causes the D-mode and E-mode to be activated in a hybrid fashion. As can be seen in the table, the performance of the AdaSG is similar to that of SuperGlue, which is much better than the other methods, and its average runtime is almost 6× better than the

SuperGlue. Figures 13 and 14 show the impact of the threshold of similarity on the matching precision and average runtime for the YFCC100M and KITTI datasets, respectively. As the YFCC100M dataset mainly contains non-sequence images (with low similarity), the average run time only decreases by 0.94% when the threshold increases from 0 to 0.12. The situation becomes different for the KITTI dataset which contains sequence images. The average run time decreases by 83.04% with a small decrease on the matching precision (from 99.79% to 99.76%) when the threshold increases from 0 to 0.12.

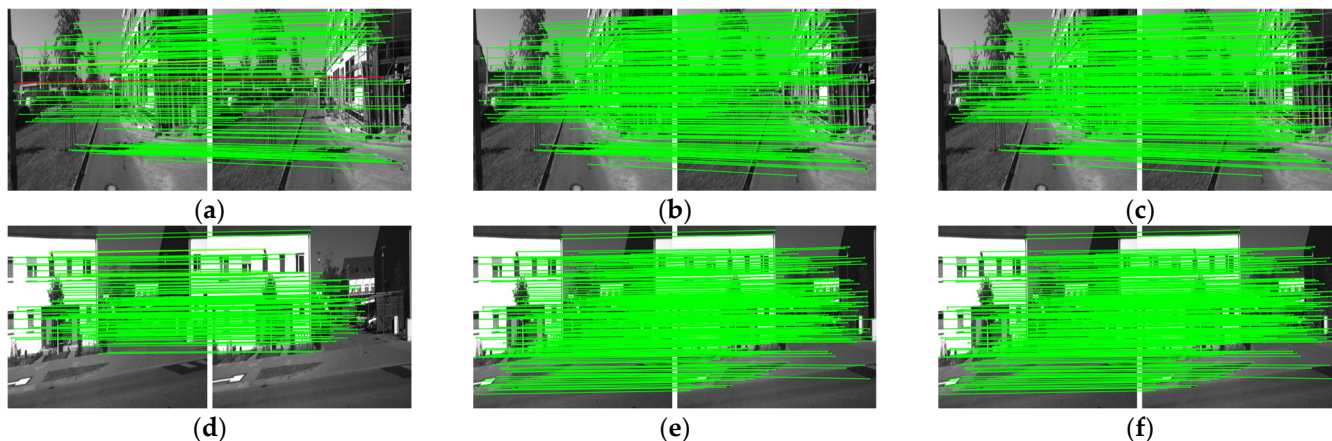


Figure 12. The visualized matching result on KITTI. The red lines represent mismatches, and the green lines represent correct matching. (a) The results of OANet. (b) The results of SuperGlue. (c) The results of AdaSG. (d) The other results of OANet. (e) The other results of SuperGlue. (f) The other results of AdaSG.

Table 4. Outdoor Experiment on KITTI Dataset.

Matcher	AUC (%)			P (%)	MS (%)	Average Runtime (s)
	@5°	@10°	@20°			
OANet	49.83	68.32	80.32	99.30	45.88	0.010
PointCN	38.58	68.32	80.32	99.70	38.59	0.003
SuperGlue	61.75	78.60	88.20	99.79	69.09	0.041
AdaSG	61.83	78.56	88.12	99.75	69.05	0.007

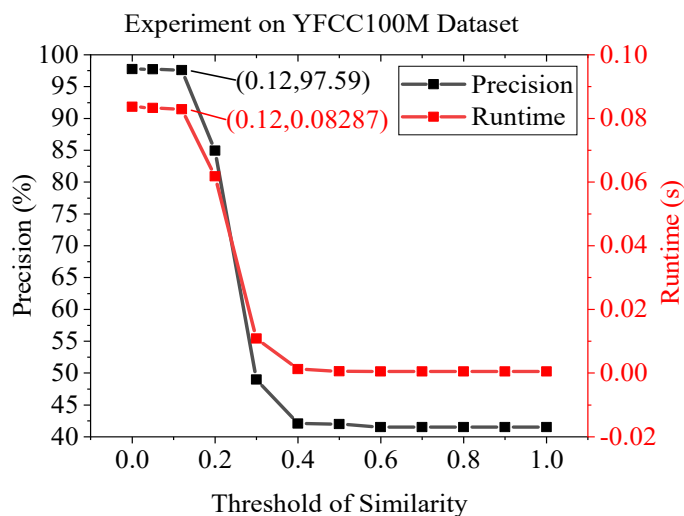


Figure 13. The impact of threshold of similarity on matching precision and average runtime on YFCC100M dataset.

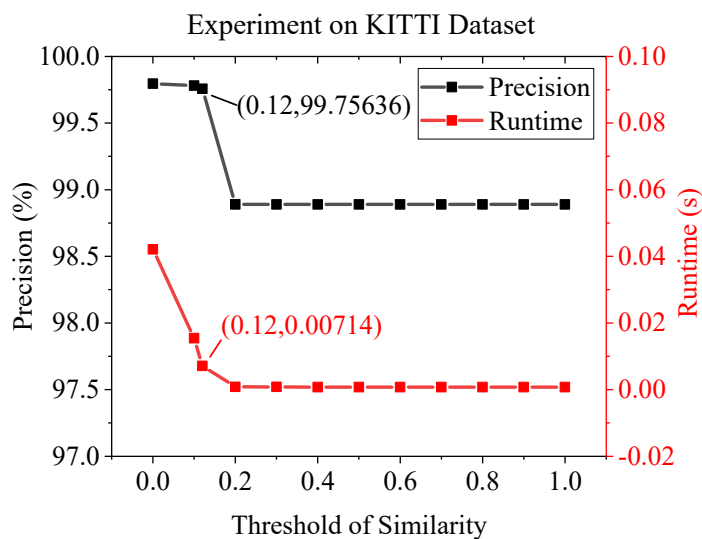


Figure 14. The impact of threshold of similarity on matching precision and average runtime on KITTI dataset.

4.4. Experimental Results on Embedded System

Figures 15 and 16 show the experimental results on the embedded system RK3399Pro for the KITTI dataset. As can be seen from the Figure 16, the matching performance of the AdaSG is similar to that of the SuperGlue on different sequences. Figure 15 shows the runtime comparison of the AdaSG and the SuperGlue. It can be seen that the average runtime of AdaSG is around 10× less than SuperGlue on the sequences 01, 03, 04, 06, 09, and 10, and is around 3× less than SuperGlue on the sequence 00, 02, 05, 07, and 08. Compared with the SuperGlue, the proposed AdaSG is much faster when running on resource-constrained devices.

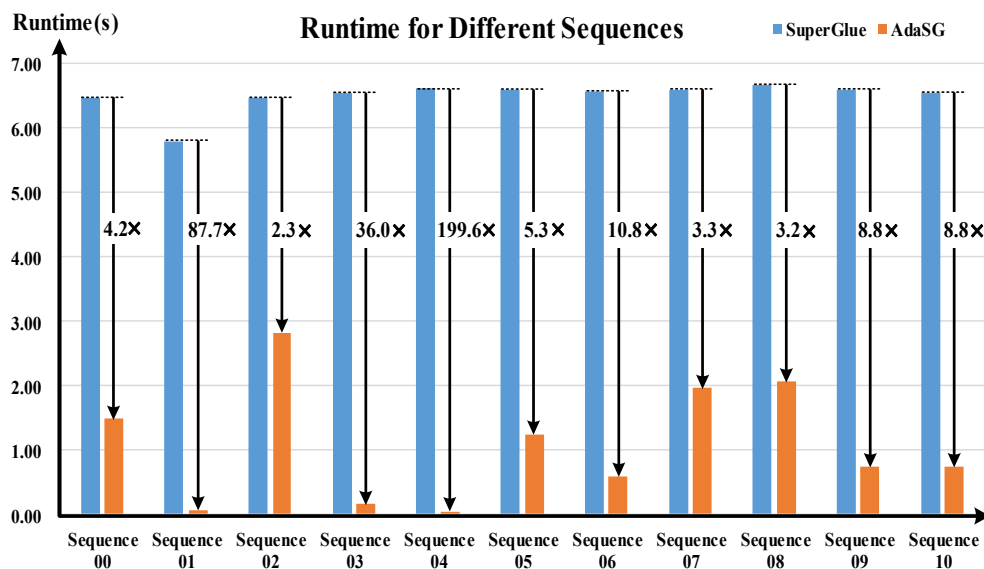


Figure 15. The runtime for different sequences of KITTI.



Figure 16. The matching performance for different sequences of KITTI. (a–h) is the sequence 00–10 of KITTI, respectively.

5. Conclusions

The SuperGlue is one of the top feature point matching methods (ranked the first in the CVPR 2020 workshop on image matching). This method uses GNN to improve the matching performance. However, this also brings in large computational complexity mak-

ing it unsuitable for resource-constrained devices. In this work, we propose a lightweight feature point matching method based on the SuperGlue (named AdaSG). It adaptively adjusts its operating architecture according to the similarity of the input image pair to reduce the computational complexity while achieving high matching performance. When running on the GPU, the proposed method achieves up to a $43\times$ and $6\times$ average runtime reduction for the indoor and outdoor datasets, respectively, with similar or higher matching performance compared with the SuperGlue. When running on an embedded system with constrained computing resources, the proposed method achieves up to $10\times$ performance improvement compared with the SuperGlue.

Author Contributions: Conceptualization, Y.L., K.H., J.L. and J.Z.; methodology, Y.L. and J.Z.; validation, J.L. and K.H.; formal analysis, Y.L. and L.C.; investigation, J.L., K.H. and X.L.; writing—original draft preparation, Y.L. and K.H.; writing—review and editing, Z.Z., L.C. and J.Z.; supervision, J.Z.; funding acquisition, J.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by NSAF (Grant No. U2030204).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Ma, J.; Jiang, X.; Fan, A.; Jiang, J.; Yan, J. Image matching from handcrafted to deep features: A survey. *Int. J. Comput. Vis.* **2021**, *129*, 23–79. [[CrossRef](#)]
2. Wang, J.; Zhong, S.; Yan, L.; Cao, Z. An Embedded System-on-Chip Architecture for Real-time Visual Detection and Matching. *IEEE Trans. Circuits Syst. Video Technol.* **2014**, *24*, 525–538. [[CrossRef](#)]
3. Cadena, C.; Carlone, L.; Carrillo, H.; Latif, Y.; Scaramuzza, D.; Neira, J.; Reid, I.; Leonard, J.J. Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. *IEEE Trans. Robot.* **2016**, *32*, 1309–1332. [[CrossRef](#)]
4. Fuentes-Pacheco, J.; Ruiz-Ascencio, J.; Rendón-Mancha, J.M. Visual simultaneous localization and mapping: A survey. *Artif. Intell. Rev.* **2015**, *43*, 55–81. [[CrossRef](#)]
5. Yuan, Z.; Song, X.; Bai, L.; Wang, Z.; Ouyang, W. Temporal-Channel Transformer for 3D Lidar-Based Video Object Detection for Autonomous Driving. *IEEE Trans. Circuits Syst. Video Technol.* **2022**, *32*, 2068–2078. [[CrossRef](#)]
6. Tang, F.; Wu, Y.; Hou, X.; Ling, H. 3D Mapping and 6D Pose Computation for Real Time Augmented Reality on Cylindrical Objects. *IEEE Trans. Circuits Syst. Video Technol.* **2020**, *30*, 2887–2899. [[CrossRef](#)]
7. Huang, Z.; Wei, Z.; Zhang, G. RWBD: Learning Robust Weighted Binary Descriptor for Image Matching. *IEEE Trans. Circuits Syst. Video Technol.* **2018**, *28*, 1553–1564. [[CrossRef](#)]
8. Huang, F.; Huang, S.; Ker, J.; Chen, Y. High-Performance SIFT Hardware Accelerator for Real-Time Image Feature Extraction. *IEEE Trans. Circuits Syst. Video Technol.* **2012**, *22*, 340–351. [[CrossRef](#)]
9. Yi, K.M.; Trulls, E.; Lepetit, V.; Fua, P. Lift: Learned invariant feature transform. In *European Conference on Computer Vision*; Springer: Berlin/Heidelberg, Germany, 2016; pp. 467–483.
10. Pan, H.; Chen, Y.; He, Z.; Meng, F.; Fan, N. TCDesc: Learning Topology Consistent Descriptors for Image Matching. *IEEE Trans. Circuits Syst. Video Technol.* **2021**, *32*, 2845–2855. [[CrossRef](#)]
11. Le, V.P.; De Tran, C. Key-point matching with post-filter using sift and brief in logo spotting. In Proceedings of the 2015 IEEE RIVF International Conference on Computing & Communication Technologies-Research, Innovation, and Vision for Future (RIVF), Can Tho, Vietnam, 25–28 January 2015; pp. 89–93.
12. Lowe, D.G. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vis.* **2004**, *60*, 91–110. [[CrossRef](#)]
13. Tuytelaars, T.; van Gool, L. Wide baseline stereo matching based on local, affinity invariant regions. In Proceedings of the British Machine Conference, Bristol, UK, 11–14 September 2000; pp. 38.1–38.14.
14. Cech, J.; Matas, J.; Perdoch, M. Efficient sequential correspondence selection by cosegmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* **2010**, *32*, 1568–1581. [[CrossRef](#)] [[PubMed](#)]
15. Zhu, H.; Jiao, L.; Ma, W.; Liu, F.; Zhao, W. A novel neural network for remote sensing image matching. *IEEE Trans. Neural Netw. Learn. Syst.* **2019**, *30*, 2853–2865. [[CrossRef](#)] [[PubMed](#)]
16. Li, W.; Zhu, X.; Gong, S. Harmonious attention network for person re-identification. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–23 June 2018; pp. 2285–2294.
17. Sarlin, P.-E.; DeTone, D.; Malisiewicz, T.; Rabinovich, A. Superglue: Learning feature matching with graph neural networks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 4938–4947.

18. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. In Proceedings of the 31st International Conference on Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; pp. 6000–6010.
19. CVPR. CVPR 2020 Workshop on Image Matching: Local Features and Beyond. 10 February 2020. Available online: <https://www.cs.ubc.ca/research/image-matching-challenge/2020/> (accessed on 9 November 2020).
20. Howard, A.G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; Adam, H. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv* **2017**, arXiv:1704.04861.
21. Zhou, Y.; Chen, S.; Wang, Y.; Huan, W. Review of research on lightweight convolutional neural networks. In Proceedings of the 2020 IEEE 5th Information Technology and Mechatronics Engineering Conference (ITOEC), Chongqing, China, 12–14 June 2020.
22. Bouguettaya, A.; Kechida, A.; Taberkit, A.M. A survey on lightweight CNN-based object detection algorithms for platforms with limited computational resources. *Int. J. Inform. Appl. Math.* **2019**, *2*, 28–44.
23. Nguyen, H.H.; Ho, B.H.; Lai, H.P.; Tran, H.T.; Le, H.T.; Banuls, A.-L.; Prudhomme, J. A Lightweight Keypoint Matching Framework for Morphometric Landmark Detection. *Ecol. Inform.* **2022**, *70*, 101694. [[CrossRef](#)]
24. Mur-Artal, R.; Tardós, J.D. Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. *IEEE Trans. Robot.* **2017**, *33*, 1255–1262. [[CrossRef](#)]
25. Danielsson, P.-E. Euclidean distance mapping. *Comput. Graph. Image Process.* **1980**, *14*, 227–248. [[CrossRef](#)]
26. Winograd, S. A new algorithm for inner product. *IEEE Trans. Comput.* **1968**, *100*, 693–694. [[CrossRef](#)]
27. Rublee, E.; Rabaud, V.; Konolige, K.; Bradski, G. ORB: An efficient alternative to SIFT or SURF. In Proceedings of the 2011 International Conference on Computer Vision, Barcelona, Spain, 6–13 November 2011; pp. 2564–2571.
28. Mishchuk, A.; Mishkin, D.; Radenovic, F.; Matas, J. Working hard to know your neighbor’s margins: Local descriptor learning loss. *arXiv* **2017**, arXiv:1705.10872.
29. Shen, X.; Wang, C.; Li, X.; Yu, Z.; Li, J.; Wen, C.; Cheng, M.; He, Z. Rf-net: An end-to-end image matching network based on receptive field. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 8132–8140.
30. Christiansen, P.H.; Kragh, M.F.; Brodskiy, Y.; Karstoft, H. Unsuperpoint: End-to-end unsupervised interest point detector and descriptor. *arXiv* **2019**, arXiv:1907.04011.
31. Ono, Y.; Trulls, E.; Fua, P.; Yi, K.M. LF-Net: Learning local features from images. In Proceedings of the 32nd International Conference on Neural Information Processing Systems, Montréal, QC, Canada, 3–8 December 2018; pp. 6237–6247.
32. DeTone, D.; Malisiewicz, T.; Rabinovich, A. Superpoint: Self-supervised interest point detection and description. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, Salt Lake City, UT, USA, 18–22 June 2018; pp. 224–236.
33. Dusmanu, M.; Rocco, I.; Pajdla, T.; Pollefeys, M.; Sivic, J.; Torii, A.; Sattler, T. D2-Net: A Trainable CNN for Joint Detection and Description of Local Features. In Proceedings of the CVPR 2019-IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019.
34. Revaud, J.; Weinzaepfel, P.; de Souza, C.; Pion, N.; Csurka, G.; Cabon, Y.; Humenberger, M. R2D2: Repeatable and reliable detector and descriptor. *arXiv* **2019**, arXiv:1906.06195.
35. Luo, Z.; Shen, T.; Zhou, L.; Zhang, J.; Yao, Y.; Li, S.; Fang, T.; Quan, L. Contextdesc: Local descriptor augmentation with cross-modality context. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 2527–2536.
36. Ebel, P.; Mishchuk, A.; Yi, K.M.; Fua, P.; Trulls, E. Beyond cartesian representations for local descriptors. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Korea, 27 October–2 November 2019; pp. 253–262.
37. Qi, C.R.; Su, H.; Mo, K.; Guibas, L.J. Pointnet: Deep learning on point sets for 3d classification and segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 652–660.
38. Yi, K.M.; Trulls, E.; Ono, Y.; Lepetit, V.; Salzmann, M.; Fua, P. Learning to find good correspondences. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 2666–2674.
39. Zhang, J.; Sun, D.; Luo, Z.; Yao, A.; Zhou, L.; Shen, T.; Chen, Y.; Quan, L.; Liao, H. Learning two-view correspondences and geometry using order-aware network. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Korea, 27 October–2 November 2019; pp. 5845–5854.
40. Dai, A.; Chang, A.X.; Savva, M.; Halber, M.; Funkhouser, T.; Nießner, M. ScanNet: Richly-Annotated 3D Reconstructions of Indoor Scenes. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 2432–2443.
41. Thomee, B.; Shamma, D.A.; Friedland, G.; Elizalde, B.; Nj, K.; Poland, D.; Borth, D.; Li, L.-J. YFCC100M: The new data in multimedia research. *Commun. ACM* **2016**, *59*, 64–73. [[CrossRef](#)]
42. Geiger, A.; Lenz, P.; Stiller, C.; Urtasun, R. Vision meets robotics: The kitti dataset. *Int. J. Robot. Res.* **2013**, *32*, 1231–1237. [[CrossRef](#)]
43. Shi, Y.; Cai, J.; Shavit, Y.; Mu, T.; Feng, W.; Zhang, K. 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, New Orleans, LA, USA, 21–24 June 2022.
44. RockChip. RK3399Pro. December 2018. Available online: <https://rockchip.fr/RK3399Pro%20datasheet%20V1.1.pdf> (accessed on 23 March 2022).