Coarse-to-Fine Deep Orientation Estimator for Local Image Matching

Yasuaki Mori[1], Tsubasa Hirakawa[1], Takayoshi Yamashita[1], and Hironobu Fujiyoshi[1]

Chubu University, 1200 Matsumotocho, Kasugai, Aichi, Japan
{yasu071021@mprg.cs , hirakawa@mprg.cs , yamashita@isc , fujiyoshi@isc}.chubu.ac.jp

Abstract. Convolutional neural networks (CNNs) have become a mainstream method for keypoint matching in addition to image recognition, object detection, and semantic segmentation. Learned Invariant Feature Transform (LIFT) is pioneering method based on CNN. It performs keypoint detection, orientation estimation, and feature description in a single network. Among these processes, the orientation estimation is needed to obtain invariance for rotation changes. However, unlike the feature point detector and feature descriptor, the orientation estimator has not been considered important for accurate keypoint matching or been well researched even after LIFT is proposed. In this paper, we propose a novel coarse-to-fine orientation estimator that improves matching accuracy. First, the coarse orientation estimator estimates orientations to make the rotation error as small as possible even if large rotation changes exist between an image pair. Second, the fine orientation estimator further improves matching accuracy with the orientation estimated by the coarse orientation estimator. By using the proposed two-stage CNNs, we can accurately estimate orientations improving matching performance. The experimental results with the HPatches benchmark show that our method can achieve a more accurate precision-recall curve than single CNN-based orientation estimators.

Keywords: Keypoint Matching · Local Image Descriptor · Orientation Estimation · HPatches.

1 Introduction

Keypoint matching, which is the one of major problems in computer vision, is widely used for various applications such as large-scale image retrieval [13], image mosaicking [6], and simultaneous localization and mapping (SLAM) [2, 7, 12]. In these applications, the same points must be matched across images taken under different conditions (e.g., illumination, viewpoint, or rotation changes). Because of the development of convolutional neural networks (CNNs), several keypoint matching methods based on CNNs have been proposed [4, 16, 19, 20] and have

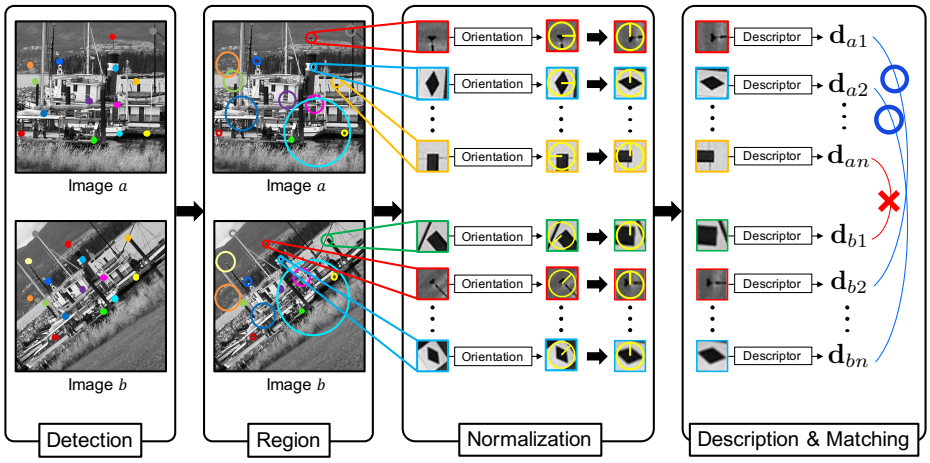

Fig. 1. Overview of keypoint matching. Keypoint matching is done by keypoint detection, region estimation, normalization, and feature description. The rotation change is dealt with by estimating orientation.

achieved higher performances than hand-crafted feature-based approaches [1, 5, 10, 15].

Typical keypoint matching methods, e.g., Scale Invariant Feature Transform (SIFT) [10] or Learned Invariant Feature Transform (LIFT) [19], individually process detection, normalization, and description as shown in Fig. 1. If any of these processes have low accuracy, the final matching accuracy will deteriorate. The process of orientation estimation is only focusing on rotation differences. Although keypoint detection and feature description have been widely studied, orientation estimation has not despite its importance in keypoint matching.

Therefore, we focus on orientation estimation. SIFT [10], the representative method of keypoint matching, estimates the orientation by using the gradient information of the image. LIFT [19], a typical CNN-based method of keypoint matching, estimates the orientation on local image patches by inputting local image patches into a CNN. In this method, positive pairs are only used to training of the orientation estimator. The positive pairs mean a couple of local image patches of the same point taken under different conditions. The orientation estimator on LIFT estimates the angle of similar local image patches into the same orientation. However, when LIFT estimates the orientation between similar but different keypoint images, these are normalized into the same orientation, which increases the number of incorrect matching results. This problem is caused by the network training that focuses on only similarities. A practical approach to overcome this problem is triplet loss with negative pairs: a couple of local image patches of different points. Triplet loss is often used when training the distance between features such as in descriptor training. However, in orientation estimation, triplet loss may decrease orientation estimation accuracy.

In this paper, we propose a novel CNN-based orientation estimation method that improves matching accuracy and retains accurate orientation estimation. The proposed method estimates the orientation of local image patches by using two coarse-to-fine structured CNNs. These CNNs use different loss functions for training and estimate different orientations. The first CNN (i.e., coarse orientation estimator) is trained using positive pairs and accurately estimates the orientation. This training process focuses on the similarities of image pairs. The second CNN (i.e., fine orientation estimator) is input a local image normalized by the coarse orientation estimator and estimates the orientation improving the matching accuracy, which uses negative pairs for network training and focuses on the differences in image pairs. Then, the final orientation of the input image is obtained by adding the orientations estimated by the coarse and fine orientation estimators.

To summarize, our key contributions are as follows:

- The proposed two-stage structure estimates orientation that achieves both accurate orientation estimation and matching results.
- The fine orientation estimator can estimate the orientation improving the final matching accuracy by using the negative pairs for orientation training. Because the fine orientation estimator focuses on dissimilar points during training, our method can reduce incorrect matching results.

2 Related work

Orientation estimation in keypoint matching has been considered less important than keypoint detection and feature description. Hence, the orientation estimation method of SIFT [10] is merely used for various local descriptors. However, the accuracy of orientation estimation greatly affects the performance of the descriptor. In this section, we introduce the orientation estimation in conventional keypoint matching.

2.1 Hand-crafted keypoint detectors

SIFT [10] is the most common method in keypoint matching. SIFT uses the histograms of pixel gradients at estimates orientation and descriptions feature by voting. Speeded-up robust features (SURF) [5] computes Haar-wavelet response of sample points to extract the local image orientation. Oriented FAST and rotated BRIEF (ORB) [15] computes orientation by using image moments and the center of mass computed from pixel values in local image patches. Histogrammed intensity patches (HIPs) [17] estimate orientation by using intensity over a 16-pixel circle centered around the keypoint detected by FAST. These methods are design in a hand-crafted manner.

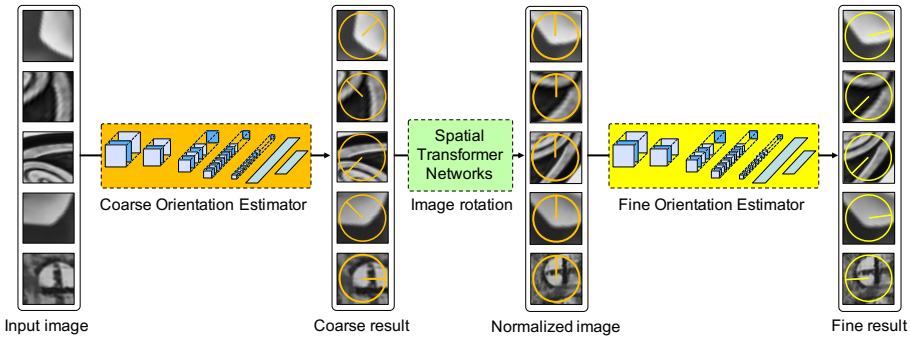

Fig. 2. Overview of the proposed method. Image patches are input to the coarse orientation estimator and then normalized by using the estimated orientation and the spatial transformer networks. The normalized image is input to the fine orientation estimator, and we obtain final results.

2.2 CNN-based methods

Although a number of CNN-based methods for keypoint matching have been proposed, few researches have focused on the CNN-based orientation estimation. In LIFT [19], different CNNs are used for keypoint detection, orientation estimation, and feature description. For orientation estimation, LIFT uses a CNN-based method proposed by Yi et al. [20]. This method estimates orientation of local image using a single CNN. LF-Net [14] is a CNN-based method that computes from keypoint detection to feature description in an end-to-end manner. This method outputs 4-channel feature maps from the CNN used for a keypoint detector. Then, the location, scale, and orientation of the keypoint are estimated simultaneously.

Because more accurate CNN-based orientation estimation has not been researched after these methods are proposed. In the latest keypoint matching methods based on CNN, orientation is often estimated by using either of these algorithms.

3 Proposed method

The proposed method estimates orientation by using two CNNs. Figure 2 shows the overview of the proposed orientation estimation method. First, we input local images to the coarse orientation estimator to estimate the orientation. Second, the spatial transformer networks (STN) [8] normalizes the input image with result of coarse orientation estimator. Third, the normalized image is input to the fine orientation estimator, and we estimate the orientation. Finally, the orientation of the input image can be obtained by adding the orientations estimated by both the coarse and fine orientation estimators. Both CNNs have the same network structures but are trained independently by using different loss functions.

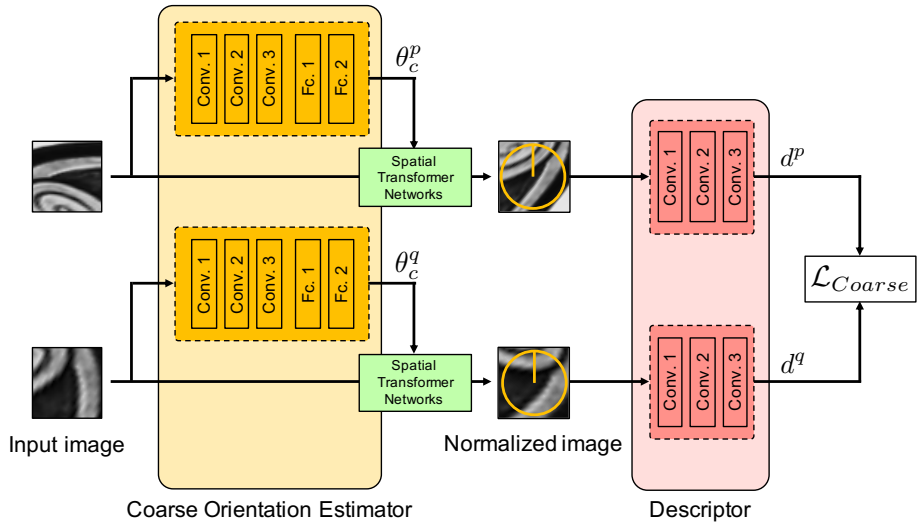

Fig. 3. The training process of the coarse orientation estimator. We train this network with positive pairs. The two coarse orientation estimators shown in this figure share parameters.

Hereafter, we introduce the details of our proposed coarse-to-fine orientation estimator.

3.1 Coarse orientation estimator

For accurate keypoint matching, orientation estimation is also an important process. If keypoint matching has inaccurate orientation estimation, it might obtain indiscriminate features even for the same image pairs in feature descriptor. Therefore, orientation must be accurately estimated to improve keypoint matching. The coarse orientation estimator is the first process of our proposed method, as shown in Fig. 2. The coarse orientation estimator aims to estimate orientation exactly even if rotation changes exist between two local image patches. To achieve such orientation estimation, we train the coarse orientation estimator by using only positive pairs to make it focus on the similarities in patches.

Figure 3 shows the training process of the coarse orientation estimator. We use the distance between the described features of image patches to train the coarse orientation estimator. The reason is that it is difficult to prepare ground truth with respect to orientation as described by Yi et al. [20]. Although SIFT orientation could be used as ground truth, it may not necessarily be correct. Therefore, the distance between described features of normalized image patches leads to estimate orientation accurately in the coarse orientation estimator. In training of coarse orientation estimator, we use only positive pairs to focus on the similar points in the input image pair. Moreover, we rotate the input images

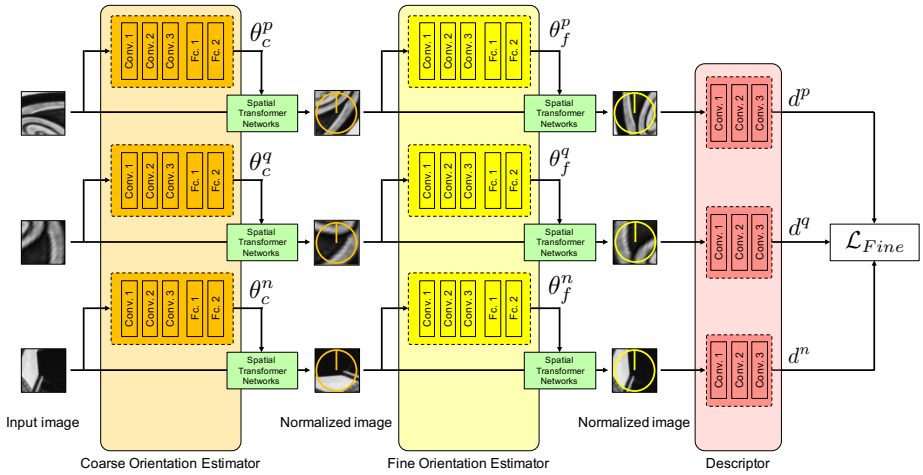

Fig. 4. The training process of the fine orientation estimator. The input images for the fine orientation estimator are normalized by the coarse orientation estimator. During the training, we update only the network parameter of the fine orientation estimator and not the other networks.

randomly to achieve robust orientation estimation. Here, let $\mathbf{p}^q$ and $\mathbf{p}^p$ be image patches of a positive pairs and $O_c(\mathbf{p})$ be the estimated orientation of patch $\mathbf{p}$ by coarse orientation estimator. We define the loss function of the coarse orientation estimator $L_{coarse}\left(\mathbf{p}^q, \mathbf{p}^p\right)$ as:

$$\mathcal{L}_{coarse}\left(\mathbf{p}^q, \mathbf{p}^p\right) = ||d(I_c(\mathbf{p}^q)) - d(I_c(\mathbf{p}^p))||_2^2, \tag{1}$$

where $d(\cdot)$ is a feature vector extracted by a descriptor. $I_c(\mathbf{p})$ is an image patch rotated by estimated orientation $O_c(\mathbf{p})$, which is formulated as follows:

$$I_c = t(\mathbf{p}, O_c(\mathbf{p})), \tag{2}$$

where $t(\mathbf{p}, O_c(\mathbf{p}))$ is computed by STN. The parameters of the descriptor are fixed and we do not train the descriptor during the training of the coarse orientation estimator. It mean that the network is attention to the similarities between positive pairs.

3.2 Fine orientation estimator

The coarse orientation estimator has a problem that the same orientation is estimated for different image patches having repetitive patterns or simple textures. This is caused that the coarse orientation estimator trains using only positive pairs. Therefore, we proposed auxiliary network as the fine orientation estimator. The fine orientation estimator is trained by using negative pairs to focus on

the different appearance points between image patches, which can estimate the orientation contributing to keypoint matching accuracy.

Figure 4 shows the training process of the fine orientation estimator. The fine orientation estimator is trained by using the distance between the extracted features of normalized image patches by a descriptor. We train the fine orientation estimator after training the coarse orientation estimator. We use three local images, which consist of a positive pairs and a negative image, as a training sample pair. Because the fine orientation estimator is necessary to focus on the differences between images. Let $\mathbf{p}_n$ be a negative image in a training sample pair and $O_f(\mathbf{p})$ be the orientation of patch $\mathbf{p}$ estimated by the fine orientation estimator. The loss function of the fine orientation estimator $L_{fine}(\mathbf{p}^q, \mathbf{p}^p, \mathbf{p}_n)$ is defined as:

$$\begin{aligned} &\mathcal{L}_{fine}\left(\mathbf{p}^q, \mathbf{p}^p, \mathbf{p}_n\right) = \\ &\max\left(0, ||d(I_f(\mathbf{p}^q)) - d(I_f(\mathbf{p}^p))||_2^2 - ||d(I_f(\mathbf{p}^q)) - d(I_f(\mathbf{p}^n))||_2^2 + M\right), \end{aligned} \tag{3}$$

where M is a margin of triple loss. We set $M = 8.0$ in the experiments described below. $I_f(\mathbf{p})$ is an image patch rotated by estimated orientation $O_f(\mathbf{p})$, which is formulated as follows:

$$I_f = t(\mathbf{p}, O_f(I_c(\mathbf{p}))). \tag{4}$$

During the training of the fine orientation estimator, the parameters of the coarse orientation estimator and the descriptor are fixed to estimate orientation to accurately match keypoints.

3.3 The calculation of final orientation

The orientation estimated by the fine orientation estimator is not the orientation for the original input local image since the input image is normalized by the coarse orientation estimator. Therefore, we compute the final orientation θ^{input} for an input image by adding the orientations estimated by the coarse and fine orientation estimators, which are defined as follows:

$$\theta^{input} = O_c + O_f. \tag{5}$$

4 Experiments

We evaluate the proposed method in terms how the orientation estimation accuracy affects the final keypoint matching accuracy.

4.1 Experimental settings

Dataset We use the HPatches dataset [3] to evaluate the proposed method. The HPatches dataset consists of local image patches extracted from six original images of the same scene under different conditions such as viewpoint and

Table 1. Results on HPatches benchmark. Orientation indicates the loss function used for training the coarse and fine orientation estimators. "L2" is Eq. (1), and "Tri" is triplet loss shown in Eq. (3).

Orientation		Patch Verification	Image Matching	Patch Retrieval
Coarse	Fine			
L2	—	0.8332	0.2660	0.3734
Tri	—	0.8623	0.2754	0.3976
L2	L2	0.8366	0.2755	0.3859
Tri	L2	0.8299	0.2670	0.3745
Tri	Tri	0.8658	0.2976	0.4196
L2	Tri (Proposed)	**0.8680**	**0.3011**	**0.4232**

illumination changes. In total, the dataset contains 116 scenes. To detect image patches from the original images, the HPatches dataset used DoG, Hessian, and Harris detectors. On the basis of the difficulty of those changes, image patches are categorized into the three categories: easy, hard, and tough. We used 1,461,525 positive pairs for training and 892,980 positive pairs for evaluation. Additionally, we created negative pairs by selecting patches randomly and used them for training the fine orientation estimator. While training both the coarse and fine orientation estimators, we randomly rotated the image pairs and input them into the networks to obtain robustness to larger rotation changes.

Training details We input local image patches of 64×64 pixels into the proposed method. The coarse and fine orientation estimators have the same structure, which consists of three convolutional layers and two fully-connected layers. For the convolution layers, the first convolution layer use a filter size of 5×5 and 10 output channel with 3×3 max pooling, the second convolutional layer a filter size of 5×5 and 20 output channel with 4×4 max pooling, and third convolutional layer use a filter size of 3×3 and 50 output channel with 2×2 max pooling. The size of the output of the first fully connected layer is 100, with the second fully connected layer having two outputs. Then convert the output to radian. During the training, we used the Adam optimizer [9] to update network parameters. We set the learning rate as 0.001, mini-batch size as 128, and the number of training epochs is 300. The descriptor consists of three convolutional layers and extracts a 128-dimensional feature vector from 64×64 pixels patches. We train a descriptor by triplet loss with HPatches dataset. Also, we use a descriptor trained with easy samples on the HPatches dataset for training the fine orientation estimator. We train each network in the following order: descriptor, coarse orientation estimator, and fine orientation estimator.

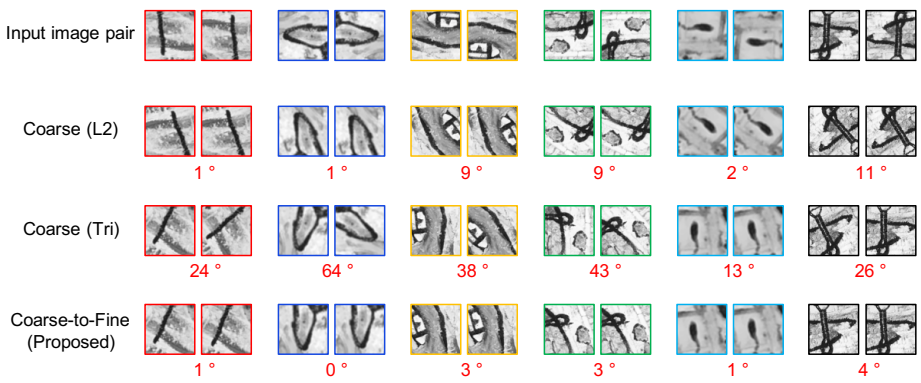

Fig. 5. Examples of orientation estimation accuracy. First row is input image pair, and the others are normalized images by orientation estimator. Input images are rotated by 180 degrees. The red numbers below the pair are estimation errors. From second to fourth rows are results to different loss functions. "L2" is the loss shown in Eq. (1) and "Tri" is triplet loss shown in Eq. (3).

4.2 Accuracies over different loss functions

We compare the performance over different loss functions used for each orientation estimator. The HPatches benchmark evaluates the performance with respect to three metrics: patch verification, image matching, and patch retrieval. For more details about these metrics, please refer to Balntas et al. [3]. In this paper, due to we focus on the effect and performance of orientation estimation, we first randomly rotate image patches and normalize the patches by using estimated orientation. Then, we extract feature vectors from the normalized patches. The distance between extracted features is large even if we use positive pairs when the accuracy of orientation estimation is low. Therefore, accurate orientation estimator is required to obtain the highly accuracy in keypoint matching. In addition, we perform orientation normalization of patch pairs containing a rotation change of 180 degrees and investigate estimation errors. For comparative methods, we use a single CNN, i.e., only coarse orientation estimator, and the proposed coarse-to-fine structured orientation estimator. Furthermore, we change the loss function used for each network training to investigate the effect on matching accuracy.

Table 1 shows results of each orientation estimation method on HPatches benchmark and Fig. 5 shows examples of the estimated orientation accuracies. From Tab. 1, when only the coarse orientation estimator is used, the triplet loss achieved higher accuracy than L2 loss. However, triplet loss has higher estimation errors than L2 loss as shown in Fig. 5. This result indicates that the estimated orientation affects the accuracy, and triplet loss function is robust to different patch pairs. And it can improves accuracy even when the estimation error is larger than L2 loss. In the proposed coarse-to-fine orientation estimation method, we employ triplet loss for the fine orientation estimator. Because of

Table 2. Results for area under the precision-recall curve

Method	HP	Rot. 90	Rot. 180
LIFT	0.535	0.154	0.437
Coarse	0.540	0.182	0.511
Coarse-to-Fine (Proposed)	0.536	**0.232**	**0.513**

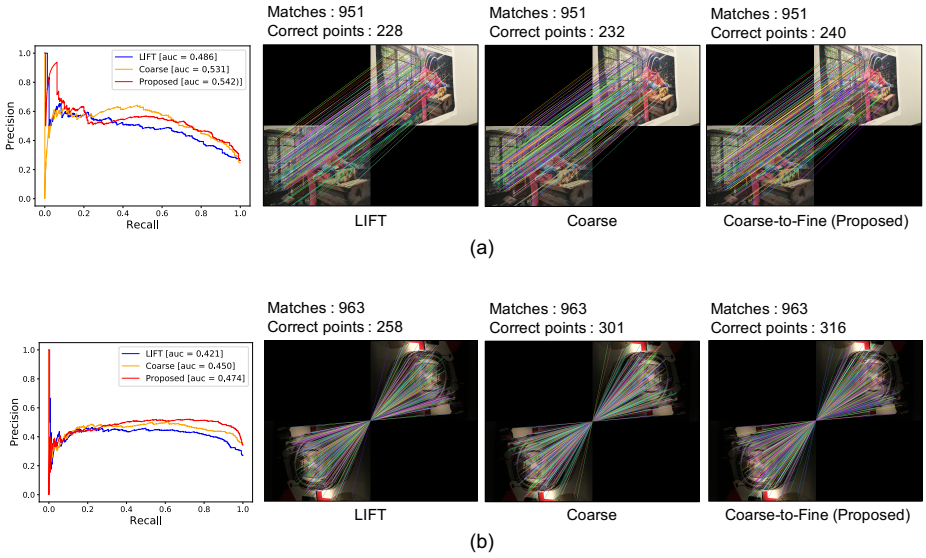

Fig. 6. Examples of matching results and precision-recall curve for (a) HP and (b) Rot. 180 conditions. From left to right: precision-recall curve, matching results of LIFT, only coarse orientation estimator, and the proposed method.

it achieved the highest accuracy and smallest estimation error. Therefore, the proposed coarse-to-fine structured method effectively improves orientation estimation accuracy.

4.3 Evaluation of area under the precision-recall curve

In this section, we evaluate the performance on keypoint matching. In this experiment, we use HPatches full sequence [3], which consists of 116 sequences and 580 image pairs. We use the LIFT detector for keypoint detection and the scale estimation. We evaluate the performance under three different conditions. The first condition is keypoint matching with original image pairs (denoted as HP). In the second condition, we rotate reference images 90 degrees (denoted as

Rot. 90) and use them to conduct keypoint matching. In the third condition, we rotate the reference images 180 degrees (denoted as Rot. 180).

We compare the performances of the proposed method, LIFT, and the coarse orientation estimator trained with positive pairs. Note that all methods use the same descriptor. For the evaluation metric, we use the area under the precision-recall curve (AUC).

Table 1 shows AUCs over different three conditions. The results of HP show that the proposed method can maintain the same matching accuracies as the other methods. Moreover, the proposed method achieved the highest accuracies for Rot. 90 and Rot. 180. Therefore, the proposed method can improve the matching when there are larger rotation changes. Figure 6 shows examples of matching results and precision-recall curves. The proposed method achieved a larger number of correct matches and a more accurate precision-recall curve than the other methods. Therefore, the orientation estimated by the proposed method contributes to keypoint matching.

4.4 Comparison with other local feature descriptors

We compare the performances of the proposed method and other local feature descriptors provided in the HPatches benchmark. For comparative methods, we use SIFT [10], RootSIFT [1], BRIEF [11], ORB [15], BinBoost [18], Siamese variants of DeepCompare [21], DeepDesc [16], TFeat margin*, and TFeat ratio* [4]. Patch pairs prepared in HPatches have at most a 30-degree rotation difference. The results of comparative methods are obtained by evaluating with this dataset. The proposed method first normalizes local image patches with estimated orientations and describes features from the normalized patches. Then, we evaluate for each task.

Figure 7 shows the evaluation results on the HPatches benchmark. The proposed method achieved the highest accuracies in all tasks. Moreover, for tough samples, the proposed method largely improves the performance. Therefore, our method efficiently normalizes rotation and describes features for image viewpoint changes.

5 Conclusion

In this paper, we proposed an orientation estimator using two convolutional neural networks (CNNs) that has a coarse-to-fine structure. The coarse orientation estimator is trained by using positive pairs and can estimate the orientations accurately. The fine orientation estimator is trained by triplet loss with negative pairs and can estimate orientation for accurate keypoint matching. We train each network with different loss functions, which enables us to achieve both accurate orientation estimation and accurate keypoint matching. The evaluation results demonstrate that our method achieved higher accuracy in all tasks on the HPatches benchmark than conventional hand-crafted approaches and

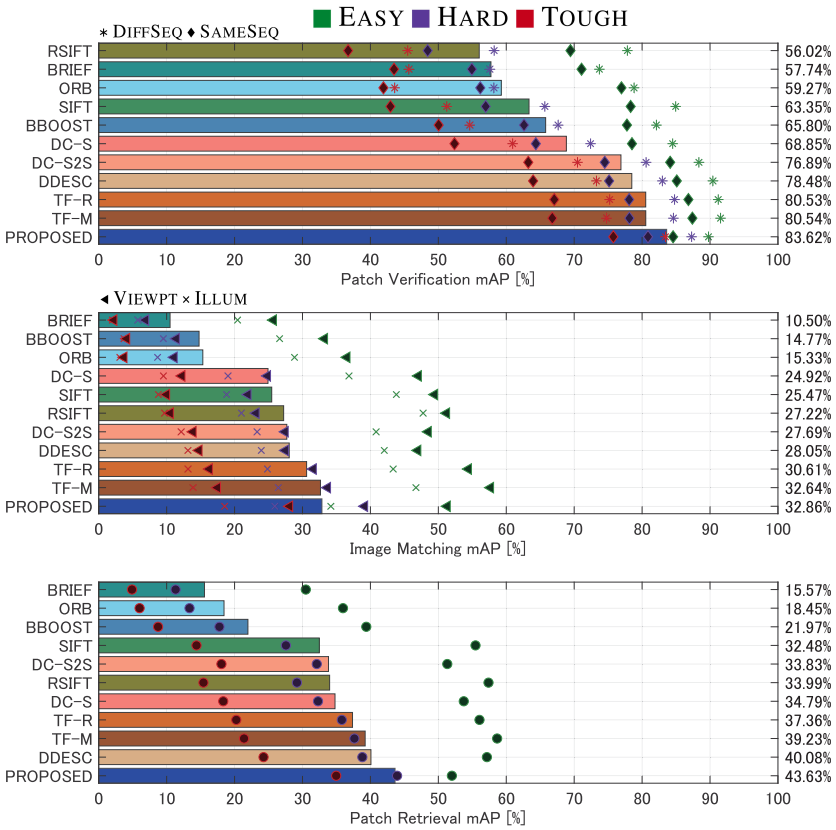

Fig. 7. The evaluation results on HPatches benchmark. Top to bottom: verification, matching and retrieval results. Marker color indicates the level of geometrical noise: easy, hard, and tough. DIFFSEQ and SAMESEQ show the scores of negative examples for the verification task. VIEWPT and ILLUM indicate the type of differences in image pairs in matching task.

CNN-based approaches. Moreover, our method achieved higher matching accuracy than single CNN-based orientation estimators. Consequently, the proposed method can estimate orientation that contributes to final keypoint matching accuracy. Our future work includes obtaining the invariance for projective changes.

References

1. Arandjelović, R., Zisserman, A.: Three things everyone should know to improve object retrieval. In: Conference on Computer Vision and Pattern Recognition. pp. 2911–2918 (2012)
2. Bailey, T., Durrant-Whyte, H.: Simultaneous localization and mapping (SLAM): Part II. Robotics & Automation Magazine (2006)

3. Balntas, V., Lenc, K., Vedaldi, A., Mikolajczyk, K.: HPatches: A benchmark and evaluation of handcrafted and learned local descriptors. In: Computer Vision and Pattern Recognition. pp. 5173–5182 (2017)
4. Balntas, V., Riba, E., Ponsa, D., Mikolajczyk, K.: Learning local feature descriptors with triplets and shallow convolutional neural networks. In: British Machine Vision Conference. p. 119 (2016)
5. Bay, H., Tuytelaars, T., Gool, L.V.: SURF: Speeded-Up Robust Features. Computer Vision and Image Understanding **110**(3), 346–359 (2008)
6. Brown, M., Lowe, D.G.: Automatic panoramic image stitching using invariant features. International Journal of Computer Vision **74**(1), 59–73 (2007)
7. Durrant-Whyte, H., Bailey, T.: Simultaneous localization and mapping: part I. Robotics & Automation Magazine (2006)
8. Jaderberg, M., Simonyan, K., Zisserman, A., Kavukcuoglu, K.: Spatial Transformer Networks. In: Neural Information Processing Systems. pp. 2017–2025 (2015)
9. Kingma, D.P., Ba, J.L.: Adam : A Method For Stochasic Optimization. In: International Conference on Learning Representation (2015)
10. Lowe, D.G.: Object Recognition from Local Scale-Invariant Features. In: International Conference on Computer Vision (1999)
11. Michael, C., Vincent, L., Christoph, S., Pascal, F.: BRIEF: binary robust independent elementary features. In: European Conference on Computer Vision. pp. 778–792 (2010)
12. Mur-Artal, R., Montiel, J.M.M., Tardos, J.D.: ORB-SLAM: a versatile and accurate monocular SLAM system. IEEE Transactions on Robotics **31**(5), 1147–1163 (2015)
13. Nister, D., Stewenius, H.: Scalable recognition with a vocabulary tree. In: Computer Vision and Pattern Recognition. vol. 2, pp. 2161–2168 (2006)
14. Ono, Y., Trulls, E., Fua, P., Yi, K.M.: LF-Net: Learning Local Features from Images. In: Neural Information Processing Systems (2018)
15. Rublee, E., Rabaud, V., Konolige, K., Bradski, G.: ORB: An Efficient Alternative to SIFT or SURF. In: International Conference on Computer Vision. pp. 2564–2571 (2011)
16. Simo-Serra, E., Trulls, E., Ferraz, L., Kokkinos, I., Fua, P., Moreno-Noguer, F.: Discriminative learning of deep convolutional feature point descriptors. In: International Conference on Computer Vision. pp. 118–126 (2015)
17. S.Taylor, T.Drummond: Binary Histogrammed Intensity Patches for Effcient and Robust Matching. International Journal of Computer Vision pp. 241–265 (2011)
18. Trzcinski, T., Christoudias, M., Lepetit, V.: Learning image descriptors with boosting. Pattern Analysis and Machine Intelligence pp. 597–610 (2015)
19. Yi, K.M., Trulls, E., Lepetit, V., Fua, P.: LIFT: Learned Invariant Feature Transform. In: European Conference on Computer Vision. pp. 467–483 (2016)
20. Yi, K.M., Verdie, Y., Fua, P., Lepetit, V.: Learning to Assign Orientations to Feature Points. In: Computer Vision and Pattern Recognition. pp. 107–116 (2016)
21. Zagoruyko, S., Komodakis, N.: Learning to compare image patches via convolutional neural networks. In: Computer Vision and Pattern Recognition. pp. 359–366 (2011)