Learning Soft Estimator of Keypoint Scale and Orientation with Probabilistic Covariant Loss

Pei Yan[1], Yihua Tan[1*], Shengzhou Xiong[1], Yuan Tai[1], and Yansheng Li[2*]
[1]National Key Laboratory of Science and Technology on Multi-spectral Information Processing,
School of Artificial Intelligence and Automation, Huazhong University of Science and Technology, China
[2]School of Remote Sensing and Information Engineering, Wuhan University, China
{yanpei, yhtan}@hust.edu.cn, xiongshengzhou@126.com, t_y_@hust.edu.cn, yansheng.li@whu.edu.cn

Abstract

Estimating keypoint scale and orientation is crucial to extracting invariant features under significant geometric changes. Recently, the estimators based on self-supervised learning have been designed to adapt to complex imaging conditions. Such learning-based estimators generally predict a single scalar for the keypoint scale or orientation, called hard estimators. However, hard estimators are difficult to handle the local patches containing structures of different objects or multiple edges. In this paper, a Soft Self-Supervised Estimator (S3Esti) is proposed to overcome this problem by learning to predict multiple scales and orientations. S3Esti involves three core factors. First, the estimator is constructed to predict the discrete distributions of scales and orientations. The elements with high confidence will be kept as the final scales and orientations. Second, a probabilistic covariant loss is proposed to improve the consistency of the scale and orientation distributions under different transformations. Third, an optimization algorithm is designed to minimize the loss function, whose convergence is proved in theory. When combined with different keypoint extraction models, S3Esti generally improves over 50% accuracy in image matching tasks under significant viewpoint changes. In the 3D reconstruction task, S3Esti decreases more than 10% reprojection error and improves the number of registered images. [code release]

1. Introduction

Keypoint-based image matching is one of the fundamental problems in many applications such as image mosaic [13], camera pose estimation [6], 3D reconstruction [15] and visual localization [14]. High matching accuracy requires the keypoint feature invariant to different imaging conditions [24]. However, it is challenging to maintain the invariance under significant geometric changes [21, 35].

An intuitive solution is to estimate the geometric change parameters and rectify the local image patches. Many existing works [3, 30, 41] demonstrate that keypoint scale and orientation can effectively represent the local geometric changes because the scaling and rotation transformations generally dominate the geometric changes in a local region. Moreover, more accurate scales and orientations generally induce higher keypoint matching accuracy [28, 38][1]. The hand-crafted methods typically estimate the scale and orientation with the analyses of gradients in the local region [5]. Some predict only one scale/orientation for a patch [32], termed as *hard estimators* in this paper.

The existing works [8, 22] demonstrate that predicting a single scale/orientation is not robust for some patches. Such patches generally contain structures of different objects or multiple edges, termed as *composite-pattern patches* in this paper. Fig. 1 shows the example patches containing different objects (a solar panel and a wall), involving at least two significant edges. A hard estimator is difficult to provide robust results for the composite-pattern patch because the most significant orientation (or scale) may be switched after an image transformation. Concerning this problem, some hand-crafted models [1, 3, 22] are constructed to predict multiple scales or orientations, which are termed as *soft estimators*. For example, SIFT first measures the confidences of different orientations based on the histogram of oriented gradients, and then keeps at most two orientations with high confidences. Experiments demonstrate that soft estimators can generally provide more robust predictions [4, 22].

However, the existing hand-crafted estimators are not robust to the complex scenes involving illumination changes or inessential patterns because the image gradients are sensitive to these interferences [41]. A failed result is shown in Fig. 1 (a). Recently, some learning-based models have

*Yihua Tan and Yansheng Li are corresponding authors.
The code is available at https://github.com/elvintanhust/S3Esti.

[1] The experimental evidence is shown in Supplementary Section 8.

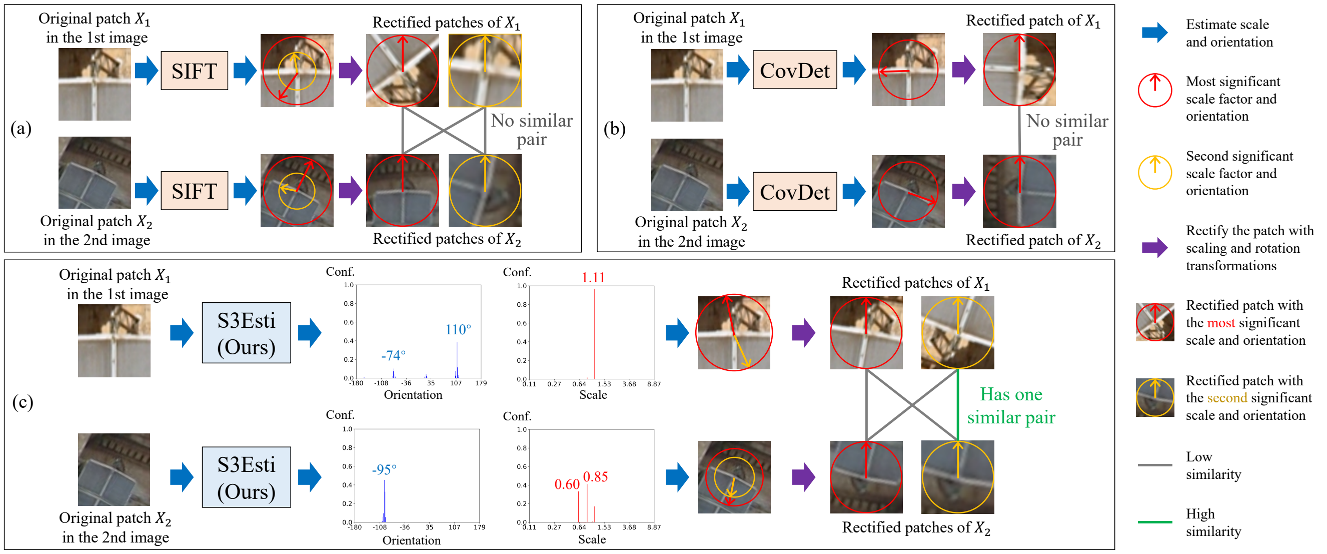
(a)
Original patch X_1 in the 1st image
SIFT
Rectified patches of X_1
No similar pair
Original patch X_2 in the 2nd image
SIFT
Rectified patches of X_2
(b)
Original patch X_1 in the 1st image
CovDet
Rectified patch of X_1
No similar pair
Original patch X_2 in the 2nd image
CovDet
Rectified patch of X_2
(c)
Original patch X_1 in the 1st image
S3Esti (Ours)
Conf.
1.0
0.8
0.6
0.4
0.2
0.0
-180
-108
-36
35
107
179
Orientation
-74°
110°
Conf.
1.11
0.11
0.27
0.64
1.53
3.68
8.87
Scale
Rectified patches of X_1
Has one similar pair
Original patch X_2 in the 2nd image
S3Esti (Ours)
-95°
0.60
0.85
Rectified patches of X_2
Estimate scale and orientation
Most significant scale factor and orientation
Second significant scale factor and orientation
Rectify the patch with scaling and rotation transformations
Rectified patch with the most significant scale and orientation
Rectified patch with the second significant scale and orientation
Low similarity
High similarity

Figure 1. Rectify image patches with the scales and orientations predicted by different estimators. In this example, each estimator predicts one or two pairs of scale and orientation. Every scale&orientation pair is used to resize and rotate the original patch, and the obtained patch is termed as a rectified patch. A good estimator should make the rectified patches as similar as possible if the original patches are centered at the same scene point. **(a)** The scales and orientations of SIFT are sensitive to illumination changes. Even though multiple predictions are kept, the rectified patches are not similar enough. **(b)** The learning-based CovDet can predict only one scale and orientation. It is difficult to provide robust predictions for the composite-pattern patch, making the rectified patches dissimilar. **(c)** The proposed S3Esti is a soft learning-based estimator that can predict multiple scales&orientations robust to illumination changes. S3Esti is more likely to get consistent scales and orientations. So it can still provide similar rectified patches under significant geometric changes.

been constructed to improve the adaptiveness for complex scenes. The existing learning-based estimators [18] are generally regression models that output one *scalar prediction* for a keypoint scale or orientation. The scalar formulation can hardly provide the confidences for multiple potential scales/orientations [11, 27], making the soft estimation strategy hard to be applied. Therefore, the existing learning-based models are hard estimators that are not robust to composite-pattern patches. The failed result in Fig. 1 (b) also demonstrates this problem.

This paper is motivated to design a soft learning-based estimator to integrate twofold advantages: the ability to predict multiple scales/orientations, and the data-driven adaptiveness for complex scenes. The proposed Soft Self-Supervised Estimator (S3Esti) is implemented as a convolutional neural network (CNN) [36] that is fed a local patch and outputs two *confidence vectors*. Each element of the vectors represents the confidence of a discretized scale or orientation. Some outputs of S3Esti are shown in Fig. 1 (c). The existing loss functions are not suitable for such vectorized outputs because they are designed for scalar prediction.

There are two difficulties in designing the loss function and optimization algorithm for S3Esti. First, S3Esti is a classification model for the discretized scales/orientations, but the scale/orientation labels are hard to be determined [18, 41]. Therefore, this paper formulates the labels as latent discrete distributions and integrates them into a novel probabilistic covariant loss. This loss can be considered as a probabilistic variant of the loss in [18].

Second, it is inefficient to update the latent scale and orientation labels with a pure gradient descent (GD) algorithm. Intuitively, there will be M independent scale and orientation labels for a training set containing M patches. As labels are unknown, GD algorithms initialize them randomly and update them with gradients. Every latent label is updated *only once per epoch* because each patch appears in only one mini-batch. This update frequency is low because the randomly initialized labels are inaccurate and the mini-batch gradients are noise. This paper designs an alternate optimization algorithm to search the optimal latent labels for the current parameters of CNN estimators.

Overall, the contributions of this paper are threefold:

(1) A soft self-supervised estimator is proposed to predict multiple scales and orientations for composite-pattern patches. Experiments demonstrate that S3Esti can provide more accurate results than the existing estimators.

(2) A loss function named probabilistic covariant loss is designed for S3Esti, making the scale/orientation predictions consistent under different geometric changes.

(3) An alternate optimization algorithm is designed by iteratively searching the latent scale&orientation labels and updating the neural network parameters.

2. Related Works

Keypoint scale and orientation estimation approaches can be divided into hand-crafted and learning-based methods. The former defines computation criteria to determine the scale and orientation. The latter designs optimization models to maximize the matching accuracy or minimize the covariant loss.

Hand-crafted estimators. Hand-crafted estimators typically determine the keypoint scale by locating the point at a scale space. SIFT [22] constructs scale space by performing multiple difference-of-Gaussian functions. Then the scale is determined by which layer this point locates on. SURF [3] and KAZE [1] design different scale spaces. Harris-Affine [25] approximately extends the space to an affine Gaussian scale space. To determine the orientation, the hand-crafted estimators generally analyze the gradient directions in a local region. SIFT [22] selects the orientation corresponding to high frequency in the histogram of oriented gradients (HOG). SURF [3] obtains the orientation according to the distribution of Haar-wavelet responses. ORB [32] defines the orientation as the direction from the keypoint location to the intensity centroid in the local region.

Learning-based estimators. The existing learning-based methods can be divided into descriptor-guiding and covariant estimators. Descriptor-guiding estimators learn to adapt to some specific keypoint descriptors. The orientation assignment model [41] is trained to maximize the matching accuracy of a descriptor. The approximate numerical gradient is used to update the model. LIFT [40], LFNet [30] and HesAffNet [27] predict scale and orientation with CNNs and rectify the patch with Spatial Transformer Networks (STNs) [16]. Then the gradients are calculated from the descriptor to the estimator. The metric-learning losses [7, 34] are used in such models. ASLFeat [23] improves the invariance of dense features by estimating the transformation with a deformable convolutional network (DCN) [9].

Covariant estimators are optimized with the covariant losses, which are independent of keypoint descriptors. CovDet [18] introduces the covariant loss function and learns an orientation detector to adapt to image translation and rotation. The subsequent work [42] introduces the standard patches into the covariant loss to improve the robustness of optimization. Another more strict covariant loss [11] is defined on a triplet of patches and an affine warped patch.

Existing learning-based methods are hard estimators. The existing learning-based estimators generally predict a scalar to represent the scale or orientation. So they are hard estimators. It is not straightforward to extend them as soft estimators. Specifically, the optimizations of descriptor-guiding estimators typically rely on the STN or DCN module [23, 40], which is difficult to be conducted with multiple scales and orientations. The loss functions of covariant estimators are defined on continuous scalar variables [18, 42], which cannot be directly defined on multiple scales and orientations. Another related work named AEU [8] uses a classification CNN to predict the relative rotation angle between two patches. However, this method relies on the ground-truth label, which is inapplicable to learn the scale/orientation whose ground-truth labels are unknown.

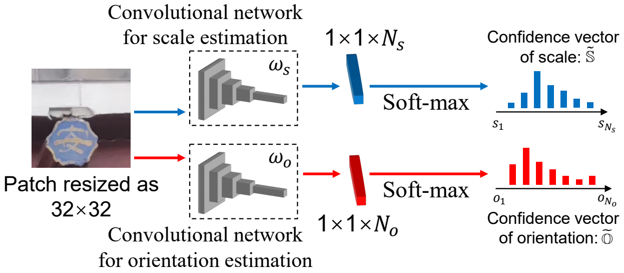

Figure 2. The architecture of S3Esti. The notations are consistent with the definitions in Sec. 3.

3. S3Esti: Optimize Probabilistic Covariant Loss of Keypoint Scale and Orientation

Discretization of scale and orientation. As a soft estimator, S3Esti first predicts the discrete distributions of scale and orientation, and then keep the scale(s) and orientation(s) with high confidence. Therefore, the discrete formulations of scale and orientation are first introduced.

The keypoint scale is a value indicating the scaling factor to rectify (resize) the local image patch around the keypoint (shown as circles in Fig. 1). In this paper, only the scale in $[A^{-1}, A]$ is concerned, where $A \geq 1$ is a hyperparameter. Then the scale is discretized as N_s values:

$$\left\{s_i = A^{-1} \cdot {\delta_s}^{i-1} \middle| i = 1, ..., N_s\right\}, \delta_s = A^{\frac{2}{N_s - 1}}. \quad (1)$$

Here δ_s is the interval to discretize the scale.

The keypoint orientation is a value indicating the angle to rectify (rotate) the local image patch around the keypoint (shown as arrows in Fig. 1). Its range is $[-\pi, \pi)$. The orientation is discretized as N_o values:

$$\left\{o_i = -\pi + (i-1) \cdot \delta_o \middle| i = 1, ..., N_o\right\}, \delta_o = \frac{2\pi}{N_o}. \quad (2)$$

Here δ_o is the interval to discretize the orientation.

Model structure. Fig. 2 shows the structure of S3Esti. The scale and orientation estimators are implemented as two independent fully convolutional networks (FCN). Their network parameters are denoted as ω_s and ω_o. Every FCN follows a Soft-max layer. Taking the patch around a keypoint as input, S3Esti outputs the scale confidence vector $\tilde{\mathbb{S}}$ and orientation confidence vector $\tilde{\mathbb{O}}$. The lengths of $\tilde{\mathbb{S}}$ and $\tilde{\mathbb{O}}$ are N_s and N_o respectively. The i-th element of $\tilde{\mathbb{S}}$ is denoted as $\tilde{\mathbb{S}}[i]$. The meaning of $\tilde{\mathbb{S}}[i]$ is stated as:

$$\tilde{\mathbb{S}}[i] \text{ is the confidence that the scale is } s_i. \quad (3)$$

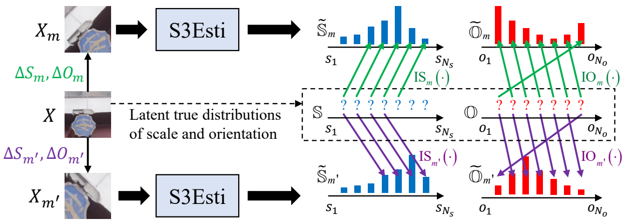

Figure 3. Visualization of the variables related to the probabilistic covariant loss. $\mathbb{S}$, $\mathbb{O}$ are the true distributions of the scale and orientation of the original patch X. "?" represents that they are unknown latent variables. $\tilde{\mathbb{S}}_m$, $\tilde{\mathbb{O}}_m$ are the predicting confidences for the transformed patch X_m. The green arrows represent the correspondence between $\mathbb{S}$, $\tilde{\mathbb{O}}_m$ and $\tilde{\mathbb{S}}_m$, $\tilde{\mathbb{O}}_m$. Another transformed patch is shown with the notation "′" and purple arrows.

Similarly, the i-th element of $\tilde{\mathbb{O}}$ is denoted as $\tilde{\mathbb{O}}[i]$:

$$\tilde{\mathbb{O}}[i] \text{ is the confidence that the orientation is } o_i. \tag{4}$$

As shown in Fig. 2, both $\tilde{\mathbb{S}}$ and $\tilde{\mathbb{O}}$ maintain the probability property with the Soft-max operation, namely,

$$\begin{aligned} &\tilde{\mathbb{S}}[i] \geq 0, i = 1, 2, ..., N_s, \quad \sum_i^{N_s} \tilde{\mathbb{S}}[i] = 1, \\ &\tilde{\mathbb{O}}[i] \geq 0, i = 1, 2, ..., N_o, \quad \sum_i^{N_o} \tilde{\mathbb{O}}[i] = 1. \end{aligned} \tag{5}$$

Probabilistic covariant loss. To mitigate the impact of geometric changes, the loss function of S3Esti should make the predicting scale and orientation confidences consistent for different transformations. Therefore, an original patch X is transformed into multiple patches to construct the loss function. Some related notations are defined as below.

As shown in Fig. 3, a set of transformed patches $\{X_m | m = 1, 2, ..., M\}$ are obtained by randomly scaling and rotating X, in which X_m is obtained with the scaling factor $\triangle S_m$ and rotation angle $\triangle O_m$. Every X_m is fed into S3Esti, and the predicting scale and orientation confidence vectors are denoted as $\tilde{\mathbb{S}}_m$ and $\tilde{\mathbb{O}}_m$.

Then two auxiliary variables $\mathbb{S}$ and $\mathbb{O}$ are introduced to complete the loss function. $\mathbb{S}$ and $\mathbb{O}$ are the true values of $\tilde{\mathbb{S}}$ and $\tilde{\mathbb{O}}$, representing the ideal distributions of the scale and orientation of X. Both $\mathbb{S}$ and $\mathbb{O}$ are unknown. They are considered as *latent variables* that will be jointly optimized with the network parameters ω_s and ω_o. In the ideal situation, every $\tilde{\mathbb{S}}_m$, $\tilde{\mathbb{O}}_m$ should be consistent with $\mathbb{S}$, $\mathbb{O}$.

As shown in Fig. 3, there may be a shift between $\tilde{\mathbb{S}}_m$ and $\mathbb{S}$ because of the transformation $\triangle S_m$. Such a shift can be directly computed with a function:

$$\mathrm{IS}_m(i) = \begin{cases} i + \triangle i_m, & 1 \leq i + \triangle i_m \leq N_s \\ \mathrm{NaN}, & \text{otherwise} \end{cases} \tag{6}$$
$$\text{where } \triangle i_m = \text{round}\left(\log_{\delta_s} \triangle S_m\right), i = 1, 2, ..., N_s.$$

Namely, the i-th index in $\mathbb{S}$ corresponds to the $\mathrm{IS}_m(i)$-th index in $\tilde{\mathbb{S}}_m$. NaN (Not a Number) indicates there is no corresponding index in $\tilde{\mathbb{S}}_m$. Eq. (6) is derived from Eq. (1). With the definition of Eq. (1), the increase of scale index is 1 when the scale value increases by δ_s times. Therefore, the index increase will be approximately equal to round $\left(\log_{\delta_s} \triangle S_m\right)$ when the scale increases by $\triangle S$ times.

Similarly, the shift between $\tilde{\mathbb{O}}_m$ and $\mathbb{O}$ can also be directly computed with a function:

$$\mathrm{IO}_m(i) = \begin{cases} i + \triangle i_m + N_o, & i + \triangle i_m < 1 \\ i + \triangle i_m, & 1 \leq i + \triangle i_m \leq N_o \\ i + \triangle i_m - N_o, & i + \triangle i_m > N_o \end{cases} \tag{7}$$
$$\text{where } \triangle i_m = \text{round}\left(\frac{\triangle O_m}{\delta_o}\right), i = 1, 2, ..., N_s.$$

Here the orientation index $\mathrm{IO}_m(i)$ is restricted in $[1, N_o]$ based on its periodicity. Eq. (7) is derived from Eq. (2), and the explanation is similar to that of Eq. (6).

The probabilistic covariant loss can be formulated with the above definitions. This loss aims to maximize the consistency between the predicting confidences $\tilde{\mathbb{S}}_m$, $\tilde{\mathbb{O}}_m$ and the true distributions $\mathbb{S}$, $\mathbb{O}$. Such consistency between two discrete distributions can be measured with their cross entropy. Therefore, the loss function of scale is:

$$\begin{aligned} &\min_{\omega_s, \mathbb{S}} - \sum_i \left(\frac{\mathbb{S}[i]}{Z_i} \cdot \sum_{m | \mathrm{IS}_m(i) \neq \mathrm{NaN}} \log\left(\tilde{\mathbb{S}}_m\left[\mathrm{IS}_m(i)\right]\right)\right) \\ &\text{s.t. } \mathbb{S}[i] \geq 0, i = 1, 2, ..., N_s, \quad \sum_i^{N_s} \mathbb{S}[i] = 1. \end{aligned} \tag{8}$$

$Z_i = \sum_m \mathbb{I}(\mathrm{IS}_m(i) \neq \mathrm{NaN})$ where $\mathbb{I}(\cdot)$ is the indicator function. Z_i is used to normalize the cross entropy because the number of legal $\mathrm{IS}_m(i)$ may be varied for different i. The network parameter ω_s determines the scale confidence $\tilde{\mathbb{S}}_m$. $\tilde{\mathbb{S}}_m$ requires no constraint because it naturally maintains the probability property with the Soft-max layer.

Similarly, the loss function of orientation is:

$$\begin{aligned} &\min_{\omega_o, \mathbb{O}} - \sum_i \left(\mathbb{O}[i] \cdot \sum_m \log\left(\tilde{\mathbb{O}}_m\left[\mathrm{IO}_m(i)\right]\right)\right) \\ &\text{s.t. } \mathbb{O}[i] \geq 0, i = 1, 2, ..., N_o, \quad \sum_i^{N_o} \mathbb{O}[i] = 1. \end{aligned} \tag{9}$$

This formulation is slightly different from Eq. (8) because no $\mathrm{IO}_m(i)$ is illegal.

Discussion for probabilistic covariant loss. In Eqs. (8) and (9), the true distributions $\mathbb{S}$, $\mathbb{O}$ are hard to be labelled with human supervision. So they are latent variables jointly optimized with ω_s, ω_o. Taking Eq. (8) as an example, minimizing such a loss leads to twofold effects:

(1) After the mapping of $\text{IS}_m(\cdot)$, the aligned $\tilde{\mathbb{S}}_m$ will be as similar as possible to $\mathbb{S}$ because the minimum of cross entropy corresponds to two identical distributions [29].
(2) $\tilde{\mathbb{S}}_m$ will be as sparse as possible. Namely, most elements of $\tilde{\mathbb{S}}_m$ will be close to 0. Based on effect (1), the cross entropy between $\tilde{\mathbb{S}}_m$ and $\mathbb{S}$ is approximately equal to the entropy of $\tilde{\mathbb{S}}_m$, which is smaller with a sparser $\tilde{\mathbb{S}}_m$ [29].

The above two effects make $\tilde{\mathbb{S}}_m$, $m = 1, 2, ..., M$ consistent with each other while maintaining sparse. Benefiting from this, only a few scales/orientations need to be kept for keypoint matching, as shown in Fig. 1 (c).

Optimization algorithm. This section only introduces the optimization of Eq. (8). A similar algorithm for Eq. (9) is in Supplementary Section 1. As discussed in Sec. 1, it is inefficient to update the latent scale and orientation labels with the pure gradient descent algorithm. Therefore, an alternate optimization algorithm (Tab. 1) is designed to divide Eq. (8) into two subproblems that can be optimized efficiently.

In the first subproblem, the optimal scale labels are searched by fixing the current network parameter ω_s. In the second subproblem, ω_s is updated to decrease the loss with the fixed scale labels. The convergence is proved in Supplementary Section 2.

Table 1. Alternate Optimization Algorithm for Scale Estimator. That for orientation estimator is in Supplementary Section 1.

```
Input: image dataset $D$, maximum iterations $T$,
initial value of network parameters $\omega_s^0$, the number of
transformed patches $M$, the largest concerned scale $A$.
Output: optimized parameter $\omega_s$.
Process:
 for $t$ from 1 to $T$:
 1 Randomly sample a training image $I$ from $D$;
 2 Randomly sample a coordinate $c$ from $I$ as the keypoint;
 3 Randomly sample the scaling factors $\triangle S_m \in [A^{-1}, A]$
     and rotation angles $\triangle O_m \in [-\pi, \pi]$, $m = 1, 2, ..., M$;
 4 Taking $c$ as the center, crop transformed patches $X_m$
     with parameters $\triangle S_m$ and $\triangle O_m$, $m = 1, 2, ..., M$;
 5 Feed $X_m$ into the estimator whose parameter is $\omega_s^{t-1}$,
     and obtain confidence vectors $\tilde{\mathbb{S}}_m$, $m = 1, 2, ..., M$;
 6 // fix $\omega_s^{t-1}$ and optimize $\mathbb{S}$, getting the solution $\mathbb{S}^*$
    $i^* = \arg\min_i -\frac{1}{Z_i} \sum_{m|\text{IS}_m(i) \neq \text{NaN}} \log \left( \tilde{\mathbb{S}}_m \left[ \text{IS}_m(i) \right] \right)$,
     and then $\mathbb{S}^*[i] = \begin{cases} 1, i = i^* \\ 0, \text{otherwise} \end{cases}$ ;
 7 // set $\mathbb{S} = \mathbb{S}^*$, and optimize $\omega_s$ with the gradient $\frac{\partial L}{\partial \omega_s}$
    $\frac{\partial L}{\partial \omega_s} = \frac{\partial}{\partial \omega_s} - \sum_i \frac{\mathbb{S}^*[i]}{Z_i} \cdot \sum_{m|\text{IS}_m(i) \neq \text{NaN}} \log \left( \tilde{\mathbb{S}}_m \left[ \text{IS}_m(i) \right] \right)$
    and update $\omega_s^{t-1}$ to $\omega_s^t$ with a gradient descent algorithm;
 end for
 $\omega_s \leftarrow \omega_s^T$;
```

4. Experiments

4.1. Model Implementation

Architecture details. As shown in Fig. 2, S3Esti implement the scale and orientation estimators as two independent FCNs. The FCN uses the VGG-A block [36] as the backbone. Then two 1×1 convolutional layers and a Softmax operation map the backbone feature to the confidence vector. The largest concerned scale $A = 9$, and therefore the range of keypoint scale is $[\frac{1}{9}, 9]$. With the definitions in Eq. (1) and Eq. (2), scale and orientation are discretized into 300 and 360 values respectively ($N_s = 300$, $N_o = 360$).

A variant named S3Esti_Joint is also implemented, whose scale and orientation estimators share the backbone. The overall accuracy of S3Esti_Joint and S3Esti is similar. However, S3Esti_Joint usually converges to a poor local minimum, requiring a carefully designed training strategy. The details are introduced in Supplementary Section 8.

Training details. S3Esti is trained on MS COCO 2014 training set [20]. In every mini-batch, 512 original patches are randomly cropped[2]. Every original patch is obtained with two steps. First, the 16×16, 32×32, and 64×64 local images are cropped around the same random center point. Second, all three local images are resized to 32×32 and concatenated as a tensor. This multi-scale cropping strategy aims to provide more information for a patch. Then every original patch is randomly transformed into two patches ($M = 2$). The transformations consist of scaling, rotation and the gray-scale augmentations used in [39]. In the optimization step 7 of Tab. 1, the stochastic gradient descent (SGD) algorithm is performed by fixing the learning rate and momentum factor as 0.001 and 0.9 respectively. The training is stopped after 30 epochs.

Inference of S3Esti. In inference, each patch centered at a keypoint is fed into S3Esti. The outputs are the confidences of different scales and orientations. The results are first filtered with a non-Maximum suppression (NMS). The NMS windows of the scale and orientation are set as $A^{\frac{1}{5}}$ and 45° respectively. Then each patch keeps at most $K = 3$ significant scales and orientations whose confidences are larger than $\text{Conf}_{\text{thre}} = 0.001$. Every patch is rectified with the pairs of scale and orientation[3], and then the similarities between different keypoints are computed based on both the original and rectified patches. Finally, the matching result is

[2] The training patches for S3Esti are randomly cropped rather than centered at some kinds of keypoints. This is different from most of the existing methods (like LFNet, CovDet) that are trained for the specific detectors.

[3] For a keypoint keeping three scales S_1, S_2, S_3 and three orientations O_1, O_2, O_3, the original patch will be rectified to five new patches. Supposing S_1 and O_1 correspond to the largest confidences, the parameter pairs are (S_1, O_1), (S_1, O_2), (S_1, O_3), (S_2, O_1), (S_3, O_1) respectively. Therefore, any original patch is rectified to at most $2K - 1$ patches. Finally, the original patch and at most $2K - 1$ rectified patches are used to represent the keypoint when computing the similarity.

determined as the keypoint inducing the highest similarity.

Moreover, a variable estimator named *S3Esti-S* is also evaluated. S3Esti-S sets $K = 1$ and ignore the restriction of $\text{Conf}_{\text{thre}}$. All other configurations are the same as S3Esti. Therefore, S3Esti-S is a hard estimator. The variant model with $K = 2$ is slightly inferior to the original S3Esti. This ablation experiment is in Supplementary Section 8.

4.2. Evaluation Dataset

HPatches dataset [2]. HPatches is used to evaluate the estimation error of scale/orientation and the matching accuracy between image pairs. HPatches has 116 sequences. Every sequence contains 6 images. Any two images in the same sequence have enough overlap, while the true homography matrix between them is known. Therefore, there are $C_6^2 = 15$ image pairs in every sequence, and HPatches consists of a total of 1740 image pairs. In this paper, HPatches is divided into three subsets, namely, HPatches-illu, HPatches-view-small, and HPatches-view-large. *HPatches-illu* has 855 pairs containing only illumination changes. *HPatches-view-small* has 770 image pairs containing relatively slight viewpoint changes. Between each pair of images, the relative global scaling factor is in $[0.5, 2]$ and the rotation angle is in $[-20°, 20°]$[4]. *HPatches-view-large* has 115 image pairs containing significant viewpoint changes, namely, the relative global scaling factor is out of $[0.5, 2]$, or the rotation angle is out of $[-20°, 20°]$.

Following the evaluation configurations in [10,39], every image is resized to 640 × 480 before extracting keypoints. All keypoint extraction methods use their recommended hyperparameters, and then keep at most 1000 keypoints with highest detection score for an image.

ETH dataset [33]. ETH is a dataset containing non-planar scenes and illumination&viewpoint changes. In Tab. 3, three sequences are used to evaluate the performance on 3D reconstruction tasks. Following the implementation [33], keypoints are extracted from the original resolution and the number of keypoints has no additional restriction.

MegaDepth dataset [19]. MegaDepth is a relative pose estimation dataset containing significant viewpoint changes. Following the existing work [37], the "Sacre Coeur" and "St. Peter's Square" scenes containing 1500 selected image pairs are used to evaluate the pose estimation accuracy. As discussed in Sec. 4.6, the results of this dataset can explain the superiority of S3Esti on the 3D reconstruction task.

4.3. Estimation Error of Scale and Orientation

Evaluation process and metrics. Our S3Esti is compared with two hand-crafted and three learning-based estimators, namely SURF [3], SIFT [22], LFNet [30], HesAffNet [27] and CovDet [18][5]. The evaluation for every estimator contains four steps. First, the *combined keypoint set* containing more than 1.9M points is extracted from HP-view-small and HP-view-large datasets with six keypoint models, namely, HAN_HN (HesAffNet+HardNet [26]), SuperPoint [10], Key.Net_HN (Key.Net [17]+HardNet), R2D2 [31], POP [39] and LFNet [30][6]. Second, the corresponding keypoint pairs are recovered according to the ground-truth homography matrices. Third, for any two keypoints in a pair, the scales and orientations are predicted with an estimator. Then the estimation error is measured by comparing the ground-truth relative scale/orientation with the predicting value[7]. Fourth, the estimation errors of scale and orientation are averaged over the combined keypoint set. The obtained mean values are denoted as *Scale Error* (*S. Err.*) and *Orientation Error* (*O. Err.*) respectively.

In the third step, the soft estimators perform a scale and orientation selection strategy to guarantee fairness. Taking Fig. 1 (c) as the example, the patch X_1 in the first image will keep only one scale and orientation that induce the highest similarity to the patches in the second image, even though the soft estimator predicts multiple orientations for X_1. The intuitive visualization of this strategy is shown in Supplementary Section 4. With the above strategy, the estimation error is evaluated for the scales and orientations paired by the nearest keypoint matching.

Moreover, some extra configurations are performed for different estimators. First, HesAffNet and LFNet originally contain keypoint detectors and scale/orientation estimators. Therefore, their estimators are evaluated on the keypoints extracted by their own detectors rather than the combined keypoint set. Second, the scale is computed on scale-pyramids rather than local patches when we estimate the scale of any keypoint with the SIFT and SURF estimators. Specifically, all local extrema in the scale pyramid are first obtained, and then every keypoint is assigned with the scale of its spatial nearest local extremum.

Results. The overall estimation errors of different estimators are shown in Tab. 2. "No-esti" represents that no estimator is performed. Namely, the scale and orientation are assigned as 1.0 and 0° respectively. The notation † indicates that the estimator is evaluated on the keypoint set extracted

[4] Generally, the image pair in HPatches contains homography transformations that cannot be precisely represented with the scaling factor and rotation angle. Therefore, the relative scaling factor and rotation angle between two images are approximately derived according to the central circle deformation caused by homography transformation. The details are introduced in Supplementary Section 4.

[5] The official CovDet does not estimate scale. We re-implement CovDet to predict scale and orientation. The backbone and training configurations are identical to those of S3Esti. Details are in Supplementary Section 7.

[6] It is not straightforward to combine S3Esti with SuperPoint and other models taking a full image as input. The strategy of Patch-R2D2 [8] is used to achieve that.

[7] This evaluation step is detailed in Supplementary Section 4.

Table 2. Estimation Error of Keypoint Scale and Orientation.

	HP-view-small		HP-view-large	
	S. Err.	O. Err. (°)	S. Err.	O. Err. (°)
No-esti	0.237	2.687	0.382	36.884
SURF	0.260	17.985	0.482	43.894
SIFT	0.254	17.640	0.431	42.203
LFNet†	0.221	22.246	0.365	33.716
HesAffNet†	0.229	**3.376**	0.288	40.559
CovDet	0.325	19.019	0.491	33.738
S3Esti-S	0.283	17.638	0.440	36.209
S3Esti	**0.214**	19.763	**0.282**	**26.794**

by its own detector rather than the combined keypoint set.

The proposed S3Esti outperforms other estimators on HP-view-large containing large viewpoint changes, while being competitive on HP-view-small. The error of S3Esti-S is higher than S3Esti. Therefore, it is crucial to keep multiple significant scales/orientations with soft prediction. Supplementary Section 5 gives the cumulative frequency histograms of the estimation errors, demonstrating that S3Esti provides a larger number of accurate estimations.

4.4. Results on Image Matching Task

Metrics. Image matching accuracy is evaluated with Matching Score (MScore)[8] and Homography Accuracy (HA) [10]. MScore is the average ratio between the correctly recovered matches and the total number of keypoints in the overlapping area. A match is correct if its reprojection error is smaller than the threshold ϵ. HA is the ratio of the image corner points whose homography estimation errors (H-error) are smaller than the threshold γ. H-error indicates the average reprojection error of the four image corners.

Results. Fig. 4 shows that S3Esti leads to better matching results than the original methods. In Fig. 5, three methods, LIFT [40], D2Net [12] and LoFTR [37][9], are added as comparison methods. The results demonstrate that HAN_HN+S3Esti and POP+S3Esti outperform the state-of-the-art methods on HP-view-large while maintaining the accuracy on the other two image sets.

Fig. 6 visualizes several patch rectification results based on the scales and orientations provided by S3Esti. The original patches are cropped from the HPatches dataset. With multiple predictions, S3Esti provides robust rectification results even for the composite-pattern patches.

The evaluation results of more combinations of S3Esti and the existing keypoint extraction models are shown in Supplementary Section 6, further verifying that S3Esti is more effective than the existing estimators. The ablation experiments and more visualization results are introduced in Supplementary Section 8 and 11. In Supplementary Section 9, S3Esti is compared with AEU [8] that estimates the relative transformation between two images.

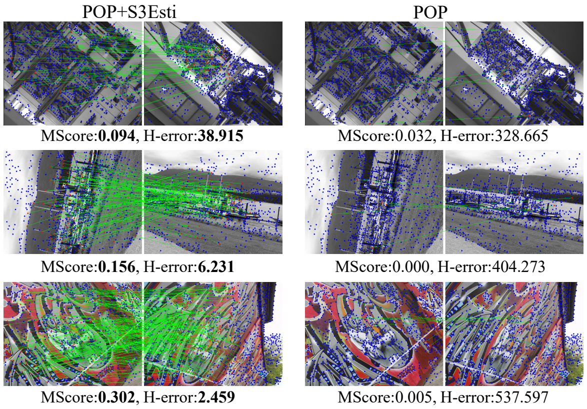
POP+S3Esti
POP
MScore:0.094, H-error:38.915
MScore:0.032, H-error:328.665
MScore:0.156, H-error:6.231
MScore:0.000, H-error:404.273
MScore:0.302, H-error:2.459
MScore:0.005, H-error:537.597

Figure 4. The visualization for the keypoint matching results with or without S3Esti. The green lines represent the matched points kept by the RANSAC process. MScore denotes the matching score with the error threshold $\epsilon = 3$, while H-error indicates the homography estimation error. The two metrics are introduced in Sec. 4.4. The results in the third row indicate that S3Esti also benefits the matching under more complex homography transformations. More results are in Supplementary Section 11.

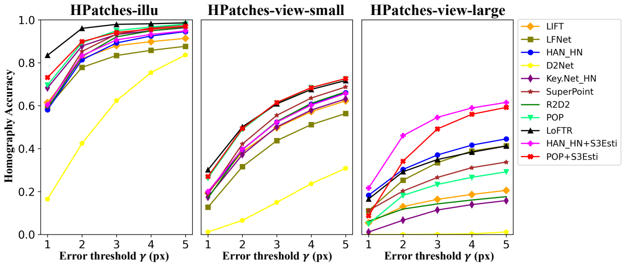

Figure 5. Homography Accuracy of different methods. The models integrated with S3Esti outperform the state-of-the-art methods under significant viewpoint changes. Moreover, S3Esti maintains the matching accuracy under small viewpoint changes.

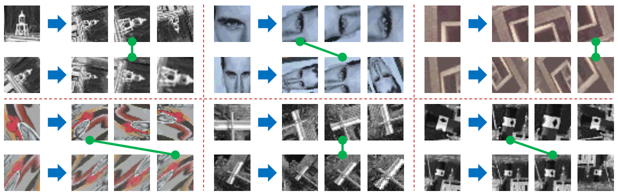

Figure 6. Patch rectification results of S3Esti. The patches on the left and right of an arrow are the original and rectified patches, respectively. Here the rectified patches with top-3 confidences are kept. The final matched patches are linked with a green line.

Inference speed. Taking HAN_HN+S3Esti as an exam-

[8] As introduced in [31], MScore and MMA are similar but different metrics. Here MScore is used following the configurations in [10, 39].

[9] LoFTR is a recent detector-free method. At present, it is inefficient to combine the proposed S3Esti with this kind of method.

ple, the extra time consumption caused by S3Esti is about 0.2 seconds for 1000 keypoints. The extra time involves scale&orientation estimation, patch rectifying and keypoint description. More details are in Supplementary Section 10.

4.5. Results on 3D Reconstruction Task

Metrics. Three metrics of 3D reconstruction are evaluated in this section. *#Dense Points*[10] is the number of reconstructed 3D points. The large number indicates a successful reconstruction and an appropriately dense point cloud. *Registered Images* (*#Reg. Images*) is the number of registered images that contribute to the reconstruction. A higher value generally means that the 3D model covers more images and therefore achieves better completeness. *Reprojection Error* (*Reproj. Error*) is the averaged reprojection error of all 3D points, which indicates the accuracy of the 3D model.

Results. In Tab. 3, POP+S3 and SP+S3 represent POP+S3Esti and SuperPoint+S3Esti respectively. Generally, POP+S3 and SP+S3 can maintain #Dense Points, indicating they complete 3D reconstruction successfully. POP+S3 outperforms the others on Reproj. Error, while POP+S3 and SP+S3 obtain a higher #Reg. Images than the original methods. Some visualization results of 3D reconstruction are shown in Supplementary Section 11. Overall, S3Esti can improve the number of registered images and the accuracy of 3D reconstruction.

4.6. Discussion

How S3Esti benefits 3D reconstruction. As demonstrated in Secs. 4.3 and 4.4, S3Esti can benefit image matching because it provides more accurate scales and orientations that lower the matching difficulty under significant geometric changes. This section discusses how S3Esti improves 3D reconstruction. The main reason may be the improvement in wide-baseline image matching. According to the results of the MegaDepth dataset in Supplementary Section 3, S3Esti improves the matching accuracy under significant viewpoint changes that generally correspond to wide-baseline image pairs. The matching of wide-baseline pairs is helpful to register more images and improve the triangulation precision in 3D reconstruction [33].

Adaptiveness on homography transformation. As shown in the third row of Fig. 4, S3Esti also benefits the image matching involving homography transformations. The reason may be that the scaling and rotation transformations still dominate the geometric changes in a local region [22, 41]. Furthermore, most of the sequences in HP-view-small and HP-view-large involve homography transformations [2]. So the quantitative results in Fig. 5 also verify the adaptiveness of S3Esti on homography transformation.

Limitations and improvement directions. The main limitations of S3Esti are twofold. First, S3Esti only predicts scale and orientation without concerning other parameters. Intuitively, the accurate estimation of more parameters like shear and squeeze may further improve the matching accuracy. Second, the current S3Esti is an independent model, which is inefficient because the intermediate features have to be computed with its own CNN backbone.

Table 3. 3D Reconstruction Results of Different Methods. “Madrid Metropolis”, “Gendarmenmarkt” and “Tower of London” are three scenes in the ETH dataset. The green texts indicate that S3Esti improves a metric compared with the baseline, while red means S3Esti worsens it. **Bord** or **green bord** texts indicate the best performance on a metric.

	Madrid Metropolis			Gendarmenmarkt			Tower of London		
Method	#Dense Points	#Reg. Images	Reproj. Error	#Dense Points	#Reg. Images	Reproj. Error	#Dense Points	#Reg. Images	Reproj. Error
D2Net	1.46M	501	1.28	3.49M	**1053**	1.19	2.73M	785	1.24
R2D2	0.17M	344	1.19	3.63M	917	1.15	0.96M	652	1.27
HAN_HN	1.74M	520	0.89	4.33M	1028	0.96	2.96M	**789**	0.85
SP	1.76M	518	1.11	3.90M	963	1.16	2.79M	730	1.12
SP+S3	1.62M	524	0.90	3.73M	1011	0.97	2.83M	775	0.89
POP	1.91M	**554**	0.84	4.08M	993	0.91	2.95M	727	0.82
POP+S3	1.73M	550	**0.70**	3.93M	1037	**0.80**	2.81M	761	**0.70**

The corresponding improvement directions are as below. First, the discretization formulations of more transformation parameters should be explored. Then S3Esti can be modified to handle such new parameters. Predicting more parameters with S3Esti will not lead to a “combination explosion” as introduced in Sec. 4.1. Second, S3Esti can be integrated into the existing local feature extraction models to re-use the intermediate features. Therefore, the overall time consumption can be significantly decreased.

5. Conclusion

In this paper, a soft self-supervised estimator named S3Esti is proposed for keypoint scale and orientation estimation. S3Esti can provide more accurate predictions compared with the existing estimators. S3Esti achieves overall 50% accuracy improvements in the image matching task under significant geometric changes, while maintaining the accuracy for small viewpoint changes. It also has good generalization on homography transformations. In the 3D reconstruction task, S3Esti improves the accuracy of the 3D point cloud. S3Esti is suitable to be a pluggable module for the existing systems. Its limitations and improvement directions are also discussed.

Acknowledgement

This work was partially supported by the National R&D Program for Major Research Instruments of Natural Science Foundation of China under grant 62027808; Shenzhen Basic Research Program, grant number JCYJ20180305180708546. We sincerely thank our reviewers and ACs for providing insightful suggestions.

[10] Following the notation in [33], “#” indicates that this metric is a count.

References

[1] Pablo Fernández Alcantarilla, Adrien Bartoli, and Andrew J Davison. Kaze features. In *Proceedings of the European Conference on Computer Vision*, pages 214–227. Springer, 2012. 1, 3

[2] Vassileios Balntas, Karel Lenc, Andrea Vedaldi, and Krystian Mikolajczyk. Hpatches: A benchmark and evaluation of handcrafted and learned local descriptors. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 5173–5182, 2017. 6, 8

[3] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In *Proceedings of the European conference on computer vision*, pages 404–417. Springer, 2006. 1, 3, 6

[4] Fabio Bellavia, Domenico Tegolo, and Emanuele Trucco. Improving sift-based descriptors stability to rotations. In *Proceedings of the International Conference on Pattern Recognition*, pages 3460–3463, 2010. 1

[5] Matthew A. Brown, Richard Szeliski, and Simon A. J. Winder. Multi-image matching using multi-scale oriented patches. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 510–517, 2005. 1

[6] Kefan Chen, Noah Snavely, and Ameesh Makadia. Wide-baseline relative camera pose estimation with directional learning. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 3258–3268, 2021. 1

[7] Christopher Choy, Jaesik Park, and Vladlen Koltun. Fully convolutional geometric features. In *Proceedings of the International Conference on Computer Vision*, 2019. 3

[8] Ji Dai, Shiwei Jin, Junkang Zhang, and Truong Q. Nguyen. Boosting feature matching accuracy with pairwise affine estimation. *IEEE Trans. Image Process.*, 29:8278–8291, 2020. 1, 3, 6, 7

[9] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *Proceedings of the International Conference on Computer Vision*, pages 764–773, 2017. 3

[10] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superpoint: Self-supervised interest point detection and description. In *Proceedings of the Conference on Computer Vision and Pattern Recognition Workshops*, pages 224–236, 2018. 6, 7

[11] Nehal Doiphode, Rahul Mitra, Shuaib Ahmed, and Arjun Jain. An improved learning framework for covariant local feature detection. In *Proceedings of the Asian Conference on Computer Vision*, volume 11366, pages 262–276, 2018. 2, 3

[12] Mihai Dusmanu, Ignacio Rocco, Tomas Pajdla, Marc Pollefeys, Josef Sivic, Akihiko Torii, and Torsten Sattler. D2-net: A trainable cnn for joint description and detection of local features. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 8092–8101, 2019. 7

[13] Kai Feng, Yongqiang Zhao, Jonathan Cheung-Wai Chan, Seong G. Kong, Xun Zhang, and Binglu Wang. Mosaic convolution-attention network for demosaicing multispectral filter array images. *IEEE Transactions on Computational Imaging*, 7:864–878, 2021. 1

[14] Janghun Hyeon, Joohyung Kim, and Nakju Doh. Pose correction for highly accurate visual localization in large-scale indoor spaces. In *Proceedings of the International Conference on Computer Vision*, pages 15974–15983, October 2021. 1

[15] Anupama K. Ingale and J. Divya Udayan. Real-time 3d reconstruction techniques applied in dynamic scenes: A systematic literature review. *Comput. Sci. Rev.*, 39:100338, 2021. 1

[16] Max Jaderberg, Karen Simonyan, Andrew Zisserman, and Koray Kavukcuoglu. Spatial transformer networks. In *Proceedings of the Advances in Neural Information Processing Systems*, pages 2017–2025, 2015. 3

[17] Axel Barroso Laguna, Edgar Riba, Daniel Ponsa, and Krystian Mikolajczyk. Key.net: Keypoint detection by handcrafted and learned CNN filters. In *Proceedings of the International Conference on Computer Vision*, pages 5835–5843. IEEE, 2019. 6

[18] Karel Lenc and Andrea Vedaldi. Learning covariant feature detectors. In *Proceedings of the European conference on computer vision*, volume 9915, pages 100–117, 2016. 2, 3, 6

[19] Zhengqi Li and Noah Snavely. Megadepth: Learning single-view depth prediction from internet photos. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 2041–2050, 2018. 6

[20] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Proceedings of the European conference on computer vision*, pages 740–755. Springer, 2014. 5

[21] Yuan Liu, Zehong Shen, Zhixuan Lin, Sida Peng, Hujun Bao, and Xiaowei Zhou. GIFT: learning transformation-invariant dense visual descriptors via group cnns. In *Proceedings of the Advances in Neural Information Processing Systems*, pages 6990–7001, 2019. 1

[22] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004. 1, 3, 6, 8

[23] Zixin Luo, Lei Zhou, Xuyang Bai, Hongkai Chen, Jiahui Zhang, Yao Yao, Shiwei Li, Tian Fang, and Long Quan. Aslfeat: Learning local features of accurate shape and localization. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 6588–6597, 2020. 3

[24] Jiayi Ma, Xingyu Jiang, Aoxiang Fan, Junjun Jiang, and Junchi Yan. Image matching from handcrafted to deep features: A survey. *Int. J. Comput. Vis.*, 129(1):23–79, 2021. 1

[25] Krystian Mikolajczyk and Cordelia Schmid. An affine invariant interest point detector. In *Proceedings of the European conference on computer vision*, pages 128–142, 2002. 3

[26] Anastasya Mishchuk, Dmytro Mishkin, Filip Radenovic, and Jiri Matas. Working hard to know your neighbor's margins: Local descriptor learning loss. In *Proceedings of the Advances in Neural Information Processing Systems*, pages 4826–4837, 2017. 6

[27] Dmytro Mishkin, Filip Radenović, and Jiři Matas. Repeatability is not enough: Learning affine regions via discriminability. In *Proceedings of the European Conference on Computer Vision*, 2018. 2, 3, 6

[28] Dibyendu Mukherjee, Q. M. Jonathan Wu, and Guanghui Wang. A comparative experimental study of image feature detectors and descriptors. *Mach. Vis. Appl.*, 26(4):443–466, 2015. 1

[29] Michael A. Nielsen. *Neural networks and deep learning*. San Francisco, CA: Determination press, 2015. 5

[30] Yuki Ono, Eduard Trulls, Pascal Fua, and Kwang Moo Yi. Lf-net: learning local features from images. In *Proceedings of the Advances in Neural Information Processing Systems*, pages 6237–6247, 2018. 1, 3, 6

[31] Jérôme Revaud, César Roberto de Souza, Martin Humenberger, and Philippe Weinzaepfel. R2D2: reliable and repeatable detector and descriptor. In *Proceedings of the Advances in Neural Information Processing Systems*, pages 12405–12415, 2019. 6, 7

[32] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary R. Bradski. ORB: an efficient alternative to SIFT or SURF. In Dimitris N. Metaxas, Long Quan, Alberto Sanfeliu, and Luc Van Gool, editors, *Proceedings of the International Conference on Computer Vision*, pages 2564–2571, 2011. 1, 3

[33] Johannes L. Schönberger, Hans Hardmeier, Torsten Sattler, and Marc Pollefeys. Comparative evaluation of hand-crafted and learned local features. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 6959–6968, 2017. 6, 8

[34] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 815–823, 2015. 3

[35] Xuelun Shen, Cheng Wang, Xin Li, Yifan Peng, Zijian He, Chenglu Wen, and Ming Cheng. Learning scale awareness in keypoint extraction and description. *Pattern Recognit.*, 121:108221, 2022. 1

[36] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *Proceedings of the International Conference on Learning Representations*, 2015. 2, 5

[37] Jiaming Sun, Zehong Shen, Yuang Wang, Hujun Bao, and Xiaowei Zhou. Loftr: Detector-free local feature matching with transformers. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 8922–8931, 2021. 6, 7

[38] Simon Taylor and Tom Drummond. Binary histogrammed intensity patches for efficient and robust matching. *Int. J. Comput. Vis.*, 94(2):241–265, 2011. 1

[39] Pei Yan, Yihua Tan, Yuan Tai, Dongrui Wu, Hanbin Luo, and Xiaolong Hao. Unsupervised learning framework for interest point detection and description via properties optimization. *Pattern Recognition*, 112:107808, 2021. 5, 6, 7

[40] Kwang Moo Yi, Eduard Trulls, Vincent Lepetit, and Pascal Fua. Lift: Learned invariant feature transform. In *Proceedings of the European Conference on Computer Vision*, pages 467–483. Springer, 2016. 3, 7

[41] Kwang Moo Yi, Yannick Verdie, Pascal Fua, and Vincent Lepetit. Learning to assign orientations to feature points. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 107–116, 2016. 1, 2, 3, 8

[42] Xu Zhang, Felix X. Yu, Svebor Karaman, and Shih-Fu Chang. Learning discriminative and transformation covariant local feature detectors. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 4923–4931, 2017. 3