Learning of low-level feature keypoints for accurate and robust detection

Suwichaya Suwanwimolkul Satoshi Komorita Kazuyuki Tasaka
KDDI Research, Inc.
{su-suwanwimolkul, sa-komorita, ka-tasaka}@kdd-research.jp

Abstract

Joint learning of feature descriptor and detector has offered promising 3D reconstruction results; however, they often lack the low-level feature awareness, which causes low accuracy in matched keypoint locations. The others employed fixed operations to select the keypoints, but the selected keypoints may not correspond to the descriptor matching. To address these problems, we propose the supervised learning of keypoint detection with low-level features. Our detector is a single CNN layer extended from the descriptor backbone, which can be jointly learned with the descriptor for maximizing the descriptor matching. This results in a state-of-the-art 3D reconstruction, especially on improving reprojection error, and the highest accuracy in keypoint detection and matching on benchmark datasets. We also present a dedicated study on evaluation metrics to measure the accuracy of keypoint detection and matching.

1. Introduction

Accurately detecting and describing the interest points in images is crucial to many applications such as localization [29], Structure from Motion (SfM) [32], and 3D reconstruction [11]. Over the past few years, the joint learning of the keypoint description and detection has shown promising results on real applications. Nevertheless, these learning-based approaches often lack low-level feature keypoints. The low-level feature keypoints mark the essential information, i.e., shape, lines, and corners in the scene, which are *robust* to the geometric changes [10, 21, 19, 7]. The lack of robustness can cause the shift between matched keypoints, leading to the low *accuracy* in matched keypoint locations.

Traditionally, the process of keypoint detection is handcrafted. The keypoint detectors are designed to detect keypoints along the corners [10] and edges [15, 2, 20, 21]. These low-level features are robust against viewpoint and illumination changes. However, the performance of handcrafted approaches is limited to the prior knowledge that developers have in hand. Many researchers turn to use deep learning for the abundant descriptiveness and distinctiveness power of the semantic features from neural networks. The learned features for keypoint description [34, 38, 22, 17] have outperformed many handcrafted features, including SIFT [15]. However, a few learning-based approaches [26, 7, 12] focused only on learning of keypoint detector. To capture low-level features, [7] and [12] employed handcrafted keypoints [10, 15] in training. However, these works can fail when the handcrafted keypoints fail.

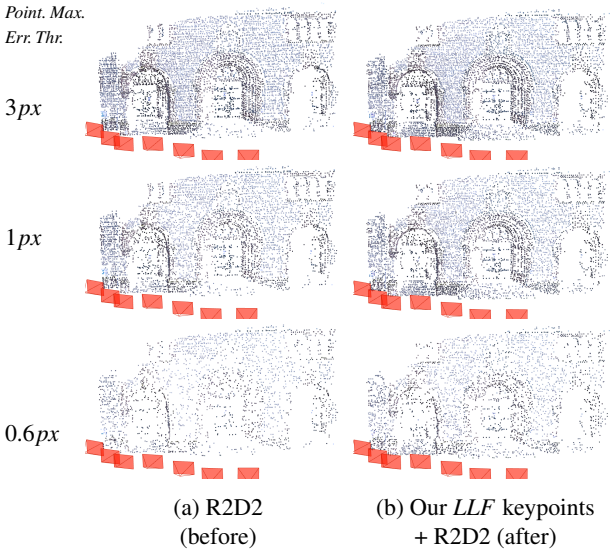


Figure 1: Comparison between (a) R2D2 (before) versus (b) our *LLF* keypoints + R2D2 (after) . Three point maximum error thresholds, $3px$ (top), $1px$, and $0.6px$ (bottom), are used to filter out erroneous points. At the lowest threshold, our approach provides more accurate shape and 3D points.

During the past few years, more studies focus on joint learning keypoint detection and description. Many algorithms encourage a certain level of low-level feature awareness in keypoint detector using unsupervised learning [6, 14, 31, 23, 40, 25]. The robustness against geometric changes was improved either by bootstrapping with multi-round adaptations on a base, pre-trained network [6, 14] or enforcing grid-wise peakiness to capture low-level features [25, 31, 40]. However, they do not employ any low-

level features to supervise the learning, which often leads to low accuracy in matched keypoint locations. To address the lack of low-level features, recent works [8, 19, 36] abandoned the learned detector. These methods derived the low-level feature keypoints from its neural network backbone and selected the keypoints with fixed softmax operations [8] or local peaks extraction [19, 36]. Although the keypoints are selected from the descriptor backbone, the fixed operations cannot guarantee to select the keypoints where descriptor matching is maximized. Thus, the accuracy of matched keypoint locations is low. Moreover, their keypoints densely cluster along edges and corners in images [19, 36].

To solve these problems, we propose to learn the low-level feature keypoint detector (*LLF* keypoints) with the multi-level features extracted from the neural network backbone. Our learning loss ensures the low-level feature awareness and sparse keypoint detection, which improves the accuracy of matched keypoint locations. We also propose a learning approach to jointly learn both our keypoint detector and descriptor to enforce *LLF* keypoints where the descriptor matching is maximized. Our learning can be applied on *a single CNN layer*, namely *LLF* detector. The *LLF* detector is extended from a descriptor backbone and can be trained with the descriptor backbone from scratch. In this paper, we present our application to improve the robustness and accuracy of repeatable keypoints of R2D2 [25], which results in reduced reprojection error while achieving a higher or similar number of registered images, sparse points, and dense points. Figure 1 demonstrates the effect of the *LLF* keypoints which provide sparse points with higher sub-pixel accuracy in 3D reconstruction (see section. 4.1.1 for full report). We also present a dedicated study to evaluate the improved accuracy of keypoints matching and detection. The architecture and our learning approach are shown in Figure 2 and 3. Our contributions are as follows:

- New learning approach and learning loss to capture low-level features;
- Light-weight keypoint detection extension;
- A dedicated study on evaluation metric to measure the accuracy of keypoints detection and matching.

The rest of the paper is organized as follows. We review the related work in section 2. Our proposed method in Section 3. Experiments and results are discussed in Section 4.

2. Related works

Handcrafted keypoints. Most of the existing keypoint detectors are handcrafted to select particular low-level features such as blob, corners or edges [10, 15, 2, 20, 21]. Harris [10] and Hessian [5] detectors used first and second order derivatives to find corners or blobs in images, which have been extended for handling multi-scales and affine transformation [21] and acceleration [4]. While the corner detectors

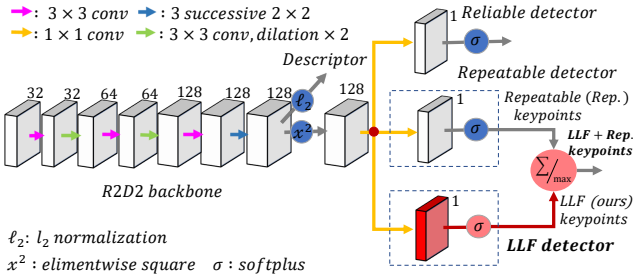


Figure 2: The proposed *LLF* detector.

are robust and efficient, SIFT looks for blobs that capture regional information of the keypoints [15].

Learned detectors. The success of learning feature descriptions for general image classification has inspired the recent applications in keypoint detection. FAST [26] is the first approach that used machine learning for fusing point and line features for keypoint detection. Recent works [31, 40] focused on learning keypoint detection for repeatability by increasing the keypoints in repeatable areas between image pairs. QuadNet [31] employed ranking loss. Then, [40] added the grid-wise peakiness for the sparse detection. The keypoint repeatability of [31, 40] is high, but their matched keypoints have low accuracy since the ground truth of keypoint location is not well defined [12]. Meanwhile, [7, 12] resorted to using handcrafted keypoints such as [10, 15], in training, due to the consistent representation of handcrafted keypoints to the low-level features. However, these works can fail if handcrafted keypoints fail.

Jointly learned descriptor and detector. The joint learning description and detection ensures the keypoint detection where the descriptor matching is maximized, by optimizing the learning losses related to both terms [39, 37, 6, 14, 17, 23, 25]. Many works employed unsupervised learning [39, 6, 14, 25] and/or relied solely on high-level features for keypoint detection [25, 23], which enable the keypoints to be matched as much as possible, but at the cost of the low matched keypoint accuracy. DELF [23] proposed to supervise learning of an attention network with image-level class labels, which captures the relevance scores of high-level information in descriptors. SuperPoint [6] steps further by training a base network with annotated corners. Then, the base network is further trained by bootstrapping with multi-round homography adaptations. Without the pre-trained network, R2D2 [25] proposed to learn the repeatable and reliable keypoints by enforcing grid-wise peakiness associated with the descriptor matching area. However, these approaches do not use the low-level features in supervision, leading to the lack of robustness against viewpoint changes.

Unlike previous methods, recent methods [8, 19, 36] abandoned the learned detector and derived the low-level feature keypoints directly from descriptor backbones. The

keypoints are selected with fixed operations. D2-Net [8] applied detection scores similar to [10] and a softmax operation to the select keypoints. UR2KiD [36] derived keypoints with a set of operations, i.e., activation norms, channel grouping, and self-distillation. ASLFeat [19] derived keypoints identified by local peakiness measure [40]. Because the keypoint selection is rather handcrafted, there is no guarantee that the selected keypoints are associated with matching the learned descriptors. Therefore, the matched keypoints do not always have high accuracy.

Keypoint refinement. Recently, [9] proposed a multiple views keypoint refinement method that solves the problem of low keypoint accuracy, similar to ours. [9] takes the matched keypoints and refines the geometry of keypoint detection and description from multiple views. However, our method improves the keypoint detection before matching, enabling other performance improvements aside from the keypoint accuracy. Also, [9] takes more computation to optimize the keypoint location under geometrical constraints. Meanwhile, our work shows the possibility of a lightweight extension that improves the matched keypoint accuracy. The comparison is provided in Section 4.2.2.

We take the inspiration from these works, especially [25, 19]. Our proposed method solves the low matched keypoint accuracy by learning with low-level features extracted from the descriptor backbone. In this paper, we apply our proposed method on R2D2 [25] as an example. In principle, our proposed scheme can be applied to other architectures, such as LIFT [39] or SuperPoint [6], that separate networks for feature detection and description. While it is possible to apply our concept to improve 3D keypoint, e.g. [35, 27], this work focuses on extracting keypoints in 2D coordinates.

3. Methods

Let us consider a setting where I and I' are the two images of the same scene, and $U \in \mathcal{R}^{H\times W\times 2}$ denotes the ground truth correspondence between them. That is, if (x,y) is a pixel coordinate in image I that corresponds to (x',y') in image I', then $U_{x,y} = (x',y')$. Let the Y and Y' denote the keypoint map associated with image I and I'. Let Y'_U denote Y' warped according to U. While it is natural to enforce the similarity between Y and Y'_U such as [24] with a least square loss, our proposal is to also learn for the keypoints that capture low-level features, which are stronger for geometric invariance. Thus, we aim to minimize

$$\begin{aligned}\mathcal{L}_{LLF} =& ||Y - Y'_U||_2^2 + ||Y - \bar{F}||_2^2 + ||Y'_U - \bar{F}'_U||_2^2 \\ &+ \mathcal{L}_{peak}(Y) + \mathcal{L}_{peak}(Y')\end{aligned} \tag{1}$$

where $\mathcal{L}_{peak}(\cdot)$ is the peakiness loss [25] for enforcing sparsity in keypoint detection, and $\bar{F}$ and $\bar{F}' \in \mathcal{R}^{H\times W}$ denote the multi-level features extracted from the descriptor backbone. The multi-level features capture the low-level information that is robust against geometric changes as proven by [8, 19, 36], but unlike these methods [8, 19, 36], we learn the detector with the multi-level features $\bar{F}$. The resulting detector is the low-level feature (*LLF*) detector that predicts a set of sparse keypoint map $Y \in [0,1]^{H\times W}$ with an emphasis on the low-level feature in an image. We present the architecture of our *LLF* detector in next. Then, we present the multi-level feature extraction for $\bar{F}$ and our modification of Eq. (1) for joint learning with the descriptor of R2D2.

3.1. Low-level feature detector

To enable our *LLF* detection with minimum extra runtime, we propose to attach the *LLF* detector after the network backbone and is in parallel with the existing detectors. The example of our application with R2D2 [25] is shown in Figure 2. Given an image I of size $H \times W$, R2D2 backbone provides the dense descriptors $X \in \mathcal{R}^{H\times W\times 128}$, and it is branched into two identical layers: (1) repeatable and (2) reliable detectors. The repeatable detector outputs the repeatability keypoint map $S \in [0,1]^{H\times W}$ which indicates repeatable keypoint locations. The reliable detector outputs the reliability map $R \in [0,1]^{H\times W}$ which estimates the reliability of the descriptor $X_{x,y}$ for good matching.

To realize the lightweight computation, the *LLF* detector is a single CNN layer. Here, we choose the same architecture as the repeatable detector, i.e., an element-wise square operation followed by a 1×1 convolutional layer and a softmax [25]. Unlike the repeatable detector, the *LLF* detector will be trained in supervised learning with the multi-level features $\bar{F}$ for more robustness and accuracy. The *LLF* detector outputs the *LLF* keypoint map, which can be used in combination with R2D2 keypoints. Nevertheless, our study in Section 4.1.1 shows that the best performance is achieved by using 100% *LLF* keypoints without any repeatable keypoints. Thus, R2D2's repeatable detector can be removed when running inference on the neural network. Our *LLF* detector can be used without costing extra runtime.

3.2. Multi-level feature extraction

To extract the multi-level features $\bar{F} \in \mathcal{R}^{H\times W}$ for supervising the *LLF* detector, we propose to use the weighted average of the dominating features from multi-channel, multi-level features for our learning. Given $I \in \mathcal{R}^{H\times W}$ as the input image to the descriptor backbone, the multi-channel features $F \in \mathcal{R}^{H\times W\times D}$ is extracted from ℓth level of the descriptor backbone with D feature channels. We extract the dominating features $\hat{F}^{(\ell)}$ from multi-channel features by

$$\hat{F}^{\ell} = \sum_{c\in[D]} \text{softmax}(F_{x,y,c}) \tag{2}$$

where $c \in [D]$ sorts each feature channel. $\text{softmax}(F_{x,y,c}) = \frac{\exp(F_{x,y,c})}{\sum_{x,y,c}\exp(F_{x,y,c})}$. We, then, normalize the dominating features to maintain reasonable scales with spatial mean $\mu_{x,y}$ and

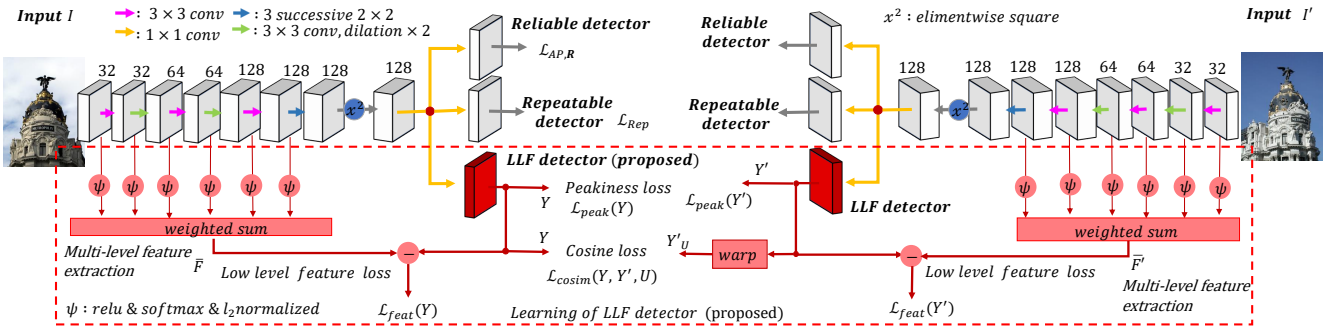


Figure 3: The proposed learning framework.

variant $\sigma^2_{x,y}$, i.e., $\tilde{F}^\ell = \frac{\hat{F}^\ell - \mu_{x,y}}{\sigma^2_{x,y}}$. To summarize the features extracted from multiple layers, we calculate the weighted sum [19] of the dominating features from multiple layers:

$$\bar{F} = \frac{1}{\sum_{\ell \in \mathbb{L}} w_\ell} \sum_{\ell \in \mathbb{L}} w_\ell \tilde{F}^{(\ell)} \tag{3}$$

where w_ℓ weights the important of each layer from $\mathbb{L}$, the set of chosen layers from a descriptor backbone. For R2D2, we choose $\mathbb{L} = \{2, 5, 8, 11, 14, 17\}$ which are the output from the *relu* of R2D2 backbone. The extracted features are used in supervised learning of *LLF* detector.

3.3. Low-level feature loss

Although using the loss function Eq. (1) is possible, here we propose to modify Eq. (1) such that this loss function can be employed in the joint learning of R2D2 which is our example application. The learning losses in R2D2 are patch-based and are normalized with patch size. Thus, we modify the following terms for the joint learning. Given the image I, one can extract both keypoint map Y and the low-level features $\bar{F}$ from the backbone, where $\bar{F}$ is calculated with Eq. (3). The low-level feature loss $\mathcal{L}_{feat}$ is defined as mean square distance between Y and $\bar{F}$:

$$\mathcal{L}_{feat}(Y) = \frac{1}{HW} \sum_{x,y} ||Y_{x,y} - \bar{F}_{x,y}||^2 \tag{4}$$

where $\bar{F}$ is gradient detached. Using this loss alone results in dense keypoints. Thus, we employed the peakiness losses $\mathcal{L}_{peak}$ [40, 25] to increase the gap between a local peak and the average value for sparsity. Let a patch of size $N \times N$ define the neighborhood of the local peak, and $\mathcal{P}$ be the set of all overlapping patches. The peakiness loss is as follows:

$$\mathcal{L}_{peak}(Y) = 1 - \frac{1}{|\mathcal{P}|} \sum_{p \in \mathcal{P}} \left(\max_{(x,y) \in p} Y_{x,y} - \operatorname*{mean}_{(x,y) \in p} Y_{x,y} \right) \tag{5}$$

Instead of using the least square loss as in Eq. (1), we resort to the cosine loss $\mathcal{L}_{cosim}$ similar to [25] for its bounded value $\in (0, 1)$. To ensure the consistent result to the peakiness loss, we calculate the cosine similarity of each patch $p \in \mathcal{P}$. The cosine loss between keypoint maps defined as

$$\mathcal{L}_{cosine}(Y, Y', U) = 1 - \frac{1}{|\mathcal{P}|} \sum_{p \in \mathcal{P}} \left(\text{cosim}\left(Y[p], Y'_U[p]\right) \right) \tag{6}$$

where $Y[\cdot]$ is the flattened $N \times N$ patch extracted from Y. Finally, the learning loss of *LLF* detector is defined as:

$$\begin{aligned} \mathcal{L}_{LLF} =& \mathcal{L}_{cosine}(Y, Y', U) + \frac{1}{2}(\mathcal{L}_{feat}(Y) + \mathcal{L}_{feat}(Y')) \\ &+ \frac{1}{2}(\mathcal{L}_{peak}(Y) + \mathcal{L}_{peak}(Y')) \end{aligned} \tag{7}$$

Next, we discuss the learning framework where the *LLF* detector is jointly learned with the R2D2 descriptor.

3.4. Learning framework

We propose to enforce the *LLF* keypoints where R2D2 descriptor matching is maximized, by optimizing the learning losses. We jointly learn our *LLF* detector and R2D2 descriptor with separate losses, i.e., learning our *LLF* detector with Eq. (7) and learning R2D2 descriptor with its own loss. R2D2 descriptor is learned by minimizing the loss at the reliable detector [25]. In addition, we found that joint learning of our *LLF* detector and R2D2's repeatable detector improves the quality of detection. The repeatable detector is learned in an unsupervised learning, which keeps the balance between discovering potential keypoints [39, 6], against our supervised learning that enforces similarity to the low-level features. Figure 3 demonstrates the joint learning of our *LLF* detector, repeatable detector, and reliable detector in our learning framework.

The learning loss for each detector is as follows:
(1) Our *LLF* detector is learned with $\mathcal{L}_{LLF}$ Eq. (7);
(2) R2D2's repeatable detector is learned with (Eq.(3) in [25]):

$$\mathcal{L}_{Rep} = \frac{1}{2}(\mathcal{L}_{peak}(S) + \mathcal{L}_{peak}(S')) + \mathcal{L}_{cosine}(S, S', U) \tag{8}$$

where S and S' denotes the repeatability map of I and I'.
(3) R2D2 descriptor is learned with (Eq.(4) in [25]):

$$\mathcal{L}_{AP,\mathbf{R}} = \frac{1}{B} \sum_{(x,y)} 1 - AP(p_{x,y}) \tag{9}$$

Methods	Feature Matching				Keypoint Detection		
	MMA	*MME*	*#Inlie.*	*#Matches*	ϵ_{IoU}	$\#Corr_{IoU}$	*#Corr.*
R2D2 [25]	0.710	1.265	**311**	**398**	0.096	**590**	559
Our 0%*LLF*+100%*Rep.*+ R2D2	0.662	1.194	226	306	0.094	**591**	**572**
Our 25%*LLF*+75%*Rep.*+ R2D2	0.722	1.083	259	323	**0.089**	552	566
Our 50%*LLF*+50%*Rep.*+ R2D2	**0.723**	**1.083**	262	326	**0.088**	556	**570**
Our 100%*LLF*+ R2D2	**0.740**	**1.070**	**269**	**328**	0.092	**569**	562
Our max(*LLF*, *Rep.*)+R2D2	**0.723**	**1.083**	**262**	**326**	**0.088**	556	**570**

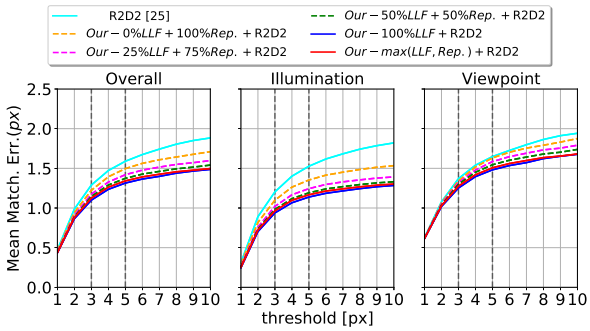


Table 1: The impact of the proposed modification, where our *LLF* detector is integrated with R2D2. The learning of *LLF* detector impacts the overall performance. As *LLF* keypoints increase ($\%LLF > 0$), *MMA* increases while the error, ${\epsilon_{IoU}}^2$ and *MME*, decreases. On the right, *MME* decreases as *LLF* keypoints increase. Our 100%*LLF*+R2D2 gives the lowest error.

Datasets	Methods	*#Reg. Imges*	*#Sparse Points*	*Track. Len.*	*Reproj. Error*	*#Obs. Points*
Herzjesu	R2D2 [25]	**8**	**13.6K**	**5.91**	1.02	**80K**
8 images	Our 0%*LLF*+100%*Rep.*+ R2D2	**8**	13.0K	5.05	0.95	66K
	Our 25%*LLF*+75%*Rep.*+ R2D2	**8**	11.4K	5.33	0.96	61K
	Our 50%*LLF*+50%*Rep.*+ R2D2	**8**	13.0K	5.56	**0.93**	72K
	Our 100%*LLF*+ R2D2	**8**	**13.0K**	**5.69**	**0.88**	**74K**
	Our max(*LLF*, *Rep.*)+R2D2	**8**	**13.1K**	**5.61**	**0.92**	**73K**
Fountain	R2D2 [25]	**11**	16.6K	**7.53**	1.04	**125K**
11 images	Our 0%*LLF*+100%*Rep.*+ R2D2	**11**	**18.0K**	5.87	0.93	106K
	Our 25%*LLF*+75%*Rep.*+ R2D2	**11**	15.3K	6.37	0.95	98K
	Our 50%*LLF*+50%*Rep.*+ R2D2	**11**	**17.3K**	6.85	**0.92**	118K
	Our 100%*LLF*+ R2D2	**11**	16.3K	**7.31**	**0.88**	**119K**
	Our max(*LLF*, *Rep.*)+R2D2	**11**	**17.2K**	**6.96**	**0.91**	**120K**

Table 2: Impact on 3D reconstruction.

where $AP(\cdot)$ is computed for each of the B patches $\{p_{x,y}\}$ in the batch. $AP(\cdot)$ is defined in [25]. These learning losses are used together when training the entire network.

4. Experimental results

In the following section, our method is evaluated across several practical scenarios, including keypoint detection and matching, 3D reconstruction, and visual localization. Visual results on keypoint detection and matching and 3D reconstruction are provided in *Supplementary*.

4.1. Keypoint detection and matching

Evaluation metrics. We evaluate the accuracy of the keypoint detection and matching using the following metrics from [8, 12, 13, 21]:

Intersection-over-Union error (ϵ_{IoU}). To evaluate the error of keypoint detection, we measure ϵ_{IoU} [12, 13] which is the compliment of the intersection over union areas between two candidates. We report the results of ϵ_{IoU} using the keypoint scales and locations (*SL*), and using only the locations (L) where scales are detected by ground truth. Radius of size 30 is used to normalize the different keypoint scales, and we choose top 1000 keypoints in evaluation.

Repeatability. We measure the proportion of correct correspondences identified by ϵ_{IoU}. The correspondences between an image pair are corrected, if $\epsilon_{IoU} < 0.4$ [12]. We also report the number of correspondences where $\epsilon_{IoU} < 0.4$ denoted as $\#Corr_{IoU}$. Following [12, 21], the repeatability is the ratio between the correspondences ($\#Corr_{IoU}$) divided by the lower number of detected keypoints.

Mean Matching Accuracy (MMA). To demonstrate the performance of feature matching, we employed *MMA* [8], which is the average percentage of correct matches in an image pair across multiple error thresholds.

Mean Matched Error (MME). We extend *MMA* [8] to evaluate the matched keypoint sub-pixel error, similar to [16]. That is, we measure the average distance of the matched keypoints to the projected locations using the ground truth homography, thus, termed mean matched error (*MME*). We also report *MME* across the error thresholds.

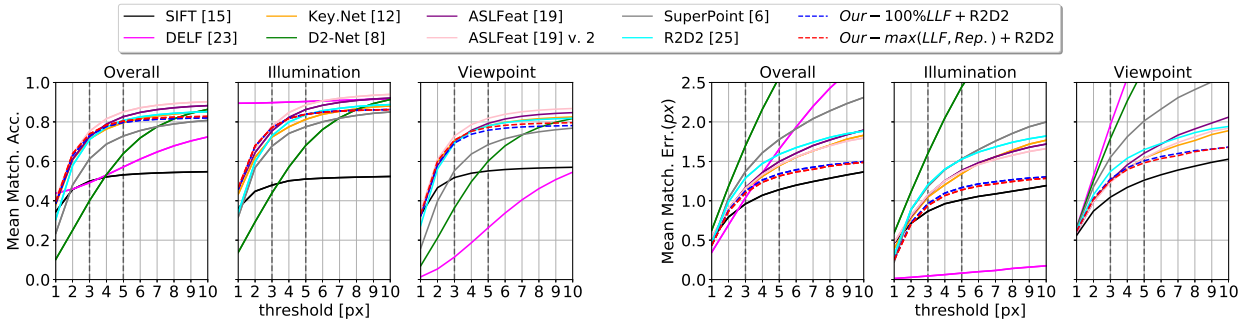
These evaluation metrics are presented in Table 1 and 3. We highlight the *top two* or *top three* and *underline* the best result. In addition, we report the average number of correct matches (*#Inlie.*), total matches (*#Matches*), and correct correspondences (*#Corr.*) whose pixel error is $< 3px$.

Datasets. The full HPatches dataset [3] with ground-truth homography are used. The dataset contains 116 image sequence where 57 sequences have large illumination change, and 59 sequences with large viewpoint changes.

Baseline and training data. Our work is compared with the reproduced results of R2D2 [25]. We use the trained R2D2-*WAF*-*N*16 and R2D2-*WASF*-*N*16 released from the official site of [25]. We use the same training data and settings of *WAF* and *WASF* [25] for training our *LLF* detector and R2D2's backbone from scratch, with batch size $(B) = 4$, number of epoch $= 25$, and patch size $(N) = 16$ (details provided in *Supplementary*). We use R2D2-*WAF*-*N*16-based development for every experiment. Except for visual localization, we use R2D2-*WASF*-*N*16-based development for the consistency with reported results in [25, 9].

Methods	Overall								Illumination				Viewpoints			
	Feat. Matching			ϵ_{IoU}		Repeatability			Feat. Matching		ϵ_{IoU}		Feat. Matching		ϵ_{IoU}	
	MMA	*MME*	*#Inlie.*	*SL*	*L*	*SL*	*L*	*#Corr.*	*MMA*	*MME*	*SL*	*L*	*MMA*	*MME*	*SL*	*L*
SIFT [15]	0.51	**1.014**	232	0.178	0.120	37.8	59.0	402	0.48	0.897	0.118	0.120	0.55	**1.127**	0.237	0.119
SURF [4]	0.47	1.211	213	0.173	0.120	44.2	**62.1**	451	0.47	1.040	0.109	0.113	0.48	1.378	0.235	0.127
Key.Net [12]	**0.72**	1.186	**408**	0.138	0.093	**60.3**	**68.2**	**591**	0.72	1.010	**0.090**	0.092	**0.71**	1.360	0.185	0.094
D2-Net [8]	0.30	1.725	141	0.219	0.183	37.2	54.7	210	0.39	1.607	0.179	0.168	0.22	1.843	0.257	0.197
ASLFeat [19]	0.69	1.178	**358**	0.142	**0.089**	49.8	61.3	**573**	0.72	1.024	**0.088**	**0.088**	0.66	1.330	0.195	**0.091**
DELF [23]	0.47	**1.016**	280	0.151	0.128	47.7	60.3	369	**0.89**	**0.043**	**0.005**	**0.011**	0.07	1.986	0.293	0.242
SuperPoint [6]	0.59	1.381	273	0.153	0.110	**57.7**	**79.1**	320	0.65	1.135	0.101	0.101	0.53	1.623	0.202	0.119
R2D2 [25]	0.71	1.265	**311**	**0.118**	0.096	**51.9**	59.2	559	0.73	1.100	0.099	0.097	0.69	1.428	**0.136**	0.096
Our max(*LLF*, *Rep.*)+R2D2	**0.72**	1.083	262	**0.124**	**0.088**	46.6	55.8	**570**	**0.75**	**0.835**	0.099	**0.087**	**0.70**	**1.327**	**0.148**	**0.089**
Our 100%*LLF*+R2D2	**0.74**	**1.070**	269	**0.126**	**0.092**	47.3	57.1	562	**0.77**	**0.819**	0.102	0.092	**0.71**	**1.318**	**0.148**	**0.092**

Table 3: Comparison to the state-of-the-art methods on the full HPatches dataset with mean matching accuracy (*MMA*), mean matched keypoint error (*MME*), average intersection over union error (ϵ_{IoU}), and repeatability (%). The error threshold is set to 3*px*. In overall, our 100%*LLF*+R2D2 is the best in *MMA* and achieves the top three in *MME* and ϵ_{IoU} (*L*) and (*SL*).



(a) *MMA* (b) *MME*

Figure 4: Comparison with the state-of-the-art in (a) *MMA* [8] and (b) *MME* at different error threshold (1-10*px*).

Comparative methods. Our work is compared with 1) the standard handcrafted: SIFT [15] and SURF [15]; 2) learning-based detector: Key.Net [12]; 3) joint local feature learning methods: R2D2 [25], SuperPoint [6], DELF [23], D2-Net [8], and ASLFeat [19]. We compare with the *multi-scaled features*, except SuperPoint that has single scale. The maximum scale detection size[1](*max. scale size*) of our work and R2D2 is set to 1600*px*. We report either results from the original papers, or derived from authors public implementations with default settings, unless otherwise specified. The number of keypoints (*#kpts*[1]) is limited to 1K for the full HPatches in Table 1 and 3, and 5K for *MMA*[8] in Figure 4.

4.1.1 Impact of low-level feature keypoints.

Table 1 shows the impact of our low-level feature (*LLF*) keypoints. We report the proportion of *LLF* keypoints as the percent to the total keypoints from *LLF* and R2D2's repeatable detector. Generally, the higher percent of *LLF* keypoints leads to the higher accuracy in keypoint detection and matching. At first, without any *LLF* keypoints (0%*LLF*), the rest of keypoints provide higher #*Corr.* As the *LLF* keypoints increases (%*LLF* $>$ 0), the error *MME* and ϵ_{IoU}[2] notably decrease, and *MMA* increases, which indicate the improved matched keypoint accuracy. We also report our max(*LLF*, *Rep.*) that chooses higher scored keypoints. On the right shows the *MME* across multiple error thresholds. The more *LLF* keypoints lead the lower *MME*, but this also cause the decrease in #*Matches* and #$Corr_{IoU}$. The lower #*Matches*, but higher *MMA*, indicate that only the low error keypoints are matched. Meanwhile, the decrease in #$Corr_{IoU}$, but higher #*Corr.*, suggest the less clustered keypoints in repeatable area, leading to lower repeatability. Nevertheless, the impact is insignificant to the 3D reconstruction as in Table 2. As *LLF* keypoints increase, the reprojection error decreases, while the others are on par with R2D2. Examples of 3D reconstruction is in Figure 1.[3]

4.1.2 Comparisons with the state-of-the-art methods.

Table 3 presents the comparison to state-of-the-art local feature extractors on the full HPatches dataset. The results are based on setting the error threshold to 3*px*. Overall,

[1] Impact of *max. scale size* and *#kpts* are studied in *Supplementary*.

[2] We report ϵ_{IoU} using only keypoint location (*L*).

[3] Full visual results are in *Supplementary*

our method shows the improved matched keypoints accuracy over the others. Our 100%*LLF*+R2D2 has the highest *MMA*, and max(*LLF*,*Rep.*)+R2D2 achieves the top three in *MMA*, ϵ_{IoU}, both *L* and *SL*, *MME*, and *#Corr.* The low ϵ_{IoU} in *L* and *SL* validates the robustness of keypoints location and scales estimation. Meanwhile, R2D2 is the best in ϵ_{IoU} (*SL*). Our method has a slightly higher ϵ_{IoU} (*SL*) since *LLF* keypoints may not well aligned with scale detection of R2D2 layout. However, our method surpassed R2D2 on *MME* and ϵ_{IoU} (*L*), which considers only the keypoint location. The repeatability is lower due to the less keypoint clustering, as discussed previously. SIFT and DELF have the best *MME*, but SIFT has low performance in the other areas, and DELF has poor results in viewpoint. KeyNet and SuperPoint are the best in repeatability *L* and *SL*. ASLFeat yields good results[4] in many areas. However, KeyNet, SuperPoint, and ASLFeat have a much higher error, *MME*, and ϵ_{IoU} (*SL*). The high *MME* indicates misalignment between matched keypoints, while ϵ_{IoU} (*SL*) indicates the lack of robustness against the changed scales in viewpoint.

Figure 4 offers the broader view of the comparison on *MMA* [8] and *MME* across multiple error thresholds (1-10 *px*). Following [8], we exclude eight high-resolution sequences and use the remaining 52 and 56 sequences with illumination or viewpoint changes, which causes a shift in the ranking. In this setting, our max(*LLF*,*Rep.*)+R2D2 is the second best after ASLFeat v.2[4], i.e, 0.74 vs. 0.75, at 3*px*. ASLFeat v.2 used more advanced training data than other methods. Many methods offer higher *MMA* when the error threshold $> 4px$. However, by increasing the error threshold, the mean matched error (*MME*) in keypoint locations also increases (Figure 4b). Our 100%*LLF*+R2D2 achieves low *MME* across error thresholds and is the second-best after SIFT. This confirms the impact of our work on improving the accuracy of matched keypoints.

4.2. 3D Reconstruction

Dataset. We employ the ETH benchmark [33] to evaluate the performance on 3D reconstruction. We follow most of the protocols in [33]. Except for our method and the baseline R2D2 [25], we use NetVLAD [1] to retrieve the top 20 nearby images of each image, and only match against them, for the images from Madrid Metropolis, Gendarmenmarkt, and Tower of London (instead of exhaustive matching). Then, we extract and match the local features with the matching ratio of 0.9. Sparse and dense reconstruction are performed by the SfM and MVS from COLMAP [32].

Evaluation protocols. We report the number of registered images (*#Reg. Imges*), sparse points (*#Sparse Points*), mean tracking length (*Track. Len.*), and reprojection error (*Reproj. Error*) for sparse reconstruction. The number of dense points (*#Dense Points*) is for dense reconstruction. Our method is compared against the state-of-the-art local features and keypoint refinement in Section 4.2.1 and 4.2.2. We limit *#kpts* of R2D2 and ours to 10K in Section 4.2.1 and 5K in 4.2.2; *max. scale size* is set to 1600*px*.

[4]Comparison with ASLFeat v.2 is discussed in *Supplementary*

Datasets	Methods	*#Reg. Imges*	*#Sparse Points*	*Track. Len.*	*Reproj. Error*	*#Dense Points*
Madrid Metropolis 1344 images	RootSIFT [2, 15]	**500**	**116K**	6.32	**0.60**	**1.82M**
	GeoDesc [18]	495	**114K**	5.97	**0.65**	1.56M
	D2-Net [8]	495	**114K**	6.39	1.35	1.46M
	ASLFeat [19]	**649**	**129K**	9.56	0.95	**1.92M**
	SuperPoint [6]	438	29K	9.03	1.02	1.55M
	R2D2 [25]	443	55K	**10.30**	0.90	1.63M
	Our max(*LLF*,*Rep.*)+R2D2	**504**	92K	**9.58**	0.86	**1.78M**
	Our 100%*LLF*+R2D2	499	78K	**9.79**	**0.85**	1.74M
Gendarmenmarkt 1463 images	RootSIFT [2, 15]	1035	**338K**	5.52	**0.69**	4.23M
	GeoDesc [18]	1004	**441K**	5.14	**0.73**	3.88M
	D2-Net [8]	965	310K	5.55	1.28	3.15M
	ASLFeat [19]	**1061**	**320K**	**8.98**	1.05	4.00M
	SuperPoint [6]	967	93K	7.22	1.03	3.81M
	R2D2 [25]	1000	183K	**9.82**	0.96	**4.35M**
	Our max(*LLF*,*Rep.*)+R2D2	**1053**	283K	8.58	0.95	**4.30M**
	Our 100%*LLF*+R2D2	**1044**	235K	**8.93**	**0.91**	**4.34M**
Tower of London 1576 images	RootSIFT [2, 15]	**804**	239K	7.76	**0.61**	3.05M
	GeoDesc [18]	776	**341K**	6.71	**0.63**	2.73M
	D2-Net [8]	708	**287K**	5.20	1.34	2.86M
	ASLFeat [19]	**846**	**252K**	**13.16**	0.95	3.08M
	SuperPoint [6]	681	52K	8.67	0.96	2.77M
	R2D2 [25]	763	102K	**13.35**	0.89	**3.11M**
	Our max(*LLF*,*Rep.*)+R2D2	**806**	160K	11.58	0.86	**3.19M**
	Our 100%*LLF*+R2D2	775	139K	**12.82**	**0.83**	**3.12M**

Table 4: Comparison to the state-of-the-art methods on ETH benchmark [33] for 3D reconstruction.

4.2.1 Comparison with state-of-the-art local features.

From Table 4, our method, i.e., 100%*LLF*+R2D2 and max(*LLF*,*Rep.*)+R2D2, provides the complete sparse and dense reconstructions according to *#Reg. Imges* and *#Dense Points*. Both of our works offer lower *Reproj. Error* than R2D2 [25]. Our 100%*LLF*+R2D2 has the second best lowest *Reproj. Error* among the learning methods after GeoDesc [18]. This validates the impact of the proposed *LLF* detector for improving the accuracy of 3D reconstruction. Though GeoDesc has a very low *Reproj. Error*, its *Track. Len.* is much shorter than the others. Our method also performs better than R2D2 in *#Reg. Imges* and *#Sparse Points*, while giving a shorter *Track. Len.* by 1.24 frame on average. This could be due to the mismatch between *LLF* keypoints and the scales detection from R2D2. However, our *Track. Len.* is still among the top three. Nevertheless, RootSIFT still has notably smaller reprojection error, which suggests the possible future improvement.

Datasets	Methods	*#Reg. Imges*	*Track. Len.*	*Reproj. Error*	*#Obs. Points*
Madrid Metropolis 1344 images	R2D2 *our rep.*	439	**10.56**	0.92	366K
	R2D2 *by [9]*	422	10.17	0.90	357K
	Ref [9]+R2D2	427	10.15	**0.76**	360K
	Our max(*LLF*,*Rep.*)+R2D2	**455**	9.79	0.84	**483K**
	Our 100%*LLF*+R2D2	**463**	**10.36**	**0.83**	**474K**
Gendarmenmarkt 1463 images	R2D2 *our rep.*	993	9.83	0.97	1.13M
	R2D2 *by [9]*	988	**9.94**	0.98	1.10M
	Ref [9]+R2D2	935	**10.04**	**0.89**	1.04M
	Our max(*LLF*,*Rep.*)+R2D2	**1030**	8.29	0.93	**1.33M**
	Our 100%*LLF*+R2D2	**1028**	8.78	**0.91**	**1.29M**
Tower of London 1576 images	R2D2 *our rep.*	700	13.38	0.91	757K
	R2D2 *by [9]*	693	**13.44**	0.92	758K
	Ref [9]+R2D2	700	**13.73**	**0.76**	760K
	Our max(*LLF*,*Rep.*)+R2D2	**746**	11.10	0.85	**964K**
	Our 100%*LLF*+R2D2	**744**	11.96	**0.82**	**971K**

Table 5: Comparison with the keypoint refinement [9].

4.2.2 Comparison with keypoint refinement.

Table 5 shows the comparison to the recent keypoint refinement method [9], which addresses the problem of refining the geometry of local features from multiple views. We denote this method as Ref [9]. For a fair comparison, we report the reproduced results of R2D2 by [9] as well as our reproduced results (*our rep.*). The similarity between R2D2 *by [9]* and *our rep.* indicates the two settings are similar. Firstly, from the Table 5, our method and Ref [9] provide less *Reproj. Error* than R2D2. The *Reproj. Error* of our 100%*LLF*+R2D2 is on par with Ref [9] on Gendarmenmarkt, but it does not achieve lower *Reproj. Error* than Ref [9] in Madrid Metropolis and Tower of London. Our *Track. Len.* is longer for Madrid Metropolis, but it is shorter in Gendarmenmarkt and Tower of London. Nevertheless, our method gives higher *#Reg. Imges* and *#Obs. Points* in all cases. Though the goal of our method and Ref [9] are similar, Ref [9] refines the matched keypoints, which improves the accuracy in 3D reconstruction, but not *#Reg. Imges* and *#Obs. Points*. Meanwhile, the *LLF* detector operates at the upstream of the 3D reconstruction pipeline; thus, it can improve the overall performance.

4.3. Visual Localization

Dataset. We employ the Aachen Day-Night dataset [30, 28] to demonstrate the effect on visual localization tasks, where the key challenge is to match images with extreme day-night changes for 98 queries.

Evaluation protocols. We use the evaluation protocols and tools provided by *The Visual Localization Benchmark*[5], which takes costumed features, performs image registration with COLMAP [32], and finally reports the percent of successfully localized images within 3 error tolerances: (0.25m, 2°) / (0.5m, 5°) / (5m, 10°). We compare with the benchmark from official site[5]. For our method and reproduced result of R2D2, we set *max. scale size* to 1600px and limit *#kpts* to 10K. We report results based on *default* image retrieval and feature matching supplied in the official site[5].

[5]https://www.visuallocalization.net/

Methods	*#kpts*	*#dim*	*#weights*	0.25m, 2°	0.5m, 5°	5m, 10°
RootSIFT [2, 15]	11K	128	-	54.1	66.3	75.5
D2-Net [8]	19K	512	15M	74.5	86.7	**100.0**
ASLFeat [19] v.2	10K	128	0.4M	**81.6**	87.8	**100.0**
SuperPoint [6]	7K	256	1.3M	73.5	79.6	88.8
R2D2 [25] $N=8$	40K	128	1.0M	76.5	**90.8**	**100.0**
R2D2 [25] $N=8$	10K	128	1.0M	74.5	83.7	**100.0**
R2D2 (*our rep.*) $N=16$	10K	128	0.5M	76.5	88.8	98.0
Our max(*LLF*,*Rep.*)+R2D2	10K	128	0.5M+129	75.5	87.8	98.0
Our 100%*LLF*+R2D2	10K	128	0.5M+129	72.4	**90.8**	99.0

Table 6: Evaluation on the Aachen Day-Night dataset [30, 28] for visual localization task.

Results. From Table 6, our 100%*LLF*+R2D2 provides better performance than R2D2 (*our rep.*) at the error tolerances (0.5m, 5°) and (5m, 10°). Our 100%*LLF*+R2D2 achieves the highest percentage of successfully localized images at (0.5m, 5°) which is close to R2D2 ($N=8$, 40K) that requires much more computational resources which are #$kpts$ (40K vs. 10K) and #$weights$ (1M vs. 0.5M+129). Because R2D2 ($N=8$, 40K) employed smaller patch size ($N=8$), the learned model is twice larger. ASLFeat v.2 also has competitive performance. It offers the highest percentages of successfully localized images at (0.25m, 2°) with lower #$weights$. Nevertheless, ASLFeat v.2 relies on using larger *max. scale size* (2048px) and an advanced database with depth information for training, i.e., blended images and rendered depths. However, our method improves the performance by new supervised learning technique, only.

5. Summary

Motivated by the role of low-level features in providing robustness against geometric changes in image scenes, we proposed a new method that supervises keypoint detection learning with the low-level features. Our learning ensures that the resulting keypoints will be sparse and agree with descriptor matching. We applied the supervised learning to a single CNN layer, namely *LLF* detector. We examined the role of the *LLF* detector applied on R2D2 and provided a dedicated study to measure the accuracy of keypoint detection. Extensive experiments showed that our *LLF* detector provides a significant improvement to achieve the highest accuracy in keypoint detection, state-of-the-art 3D reconstruction, and competitive visual localization.

References

[1] R. Arandjelovìc, P. Gronat, A. Torii, T. Pajdla, and J. Sivic. NetVLAD: CNN architecture for weakly supervised place recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(6):1437–1451, 2018.

[2] R. Arandjelović and A. Zisserman. Three things everyone should know to improve object retrieval. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2911–2918, 2012.

[3] V. Balntas, K. Lenc, A. Vedaldi, and K. Mikolajczyk. HPatches: A benchmark and evaluation of handcrafted and learned local descriptors. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[4] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool. Speeded-up robust features (SURF). *Computer Vision Image Understanding*, 110(3):346359, June 2008.

[5] P. R. Beaudet. Rotationally invariant image operators. *International Joint Conference on Pattern Recognition*, 1978.

[6] D. DeTone, T. Malisiewicz, and A. Rabinovich. SuperPoint: Self-supervised interest point detection and description. In *2018 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 337–33712, 2018.

[7] P. Di Febbo, C. Dal Mutto, K. Tieu, and S. Mattoccia. KCNN: Extremely-efficient hardware keypoint detection with a compact convolutional neural network. In *2018 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 795–7958, 2018.

[8] M. Dusmanu, I. Rocco, T. Pajdla, M. Pollefeys, J. Sivic, A. Torii, and T. Sattler. D2-Net: A trainable CNN for joint description and detection of local features. In *2019 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8084–8093, 2019.

[9] M. Dusmanu, J. L. Schönberger, and M. Pollefeys. Multi-View optimization of local feature geometry. In *Proceedings of the European Conference on Computer Vision*, 2020.

[10] C. Harris and M. Stephens. A combined corner and edge detector. In *Procedings of the Alvey Vision Conference 1988*. Alvey Vision Club, 1988.

[11] J. Heinly, J. L. Schönberger, E. Dunn, and J. Frahm. Reconstructing the world* in six days. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3287–3295, 2015.

[12] A. B. Laguna, E. Riba, D. Ponsa, and K. Mikolajczyk. Key.net: Keypoint detection by handcrafted and learned cnn filters. In *2019 IEEE International Conference on Computer Vision (ICCV)*, pages 5835–5843, 2019.

[13] K. Lenc and A. Vedaldi. Large scale evaluation of local image feature detectors on homography datasets. In *Proceedings of the British Machine Vision Conference (BMVC)*. BMVA Press, 2018.

[14] J. Li and G. H. Lee. USIP: Unsupervised Stable interest point detection from 3D point clouds. *arXiv preprint arXiv:1904.00229*, 2019.

[15] D. Lowe. Distinctive image Features from scale-invariant keypoints. *International Journal of Computer Vision*, 60:91–, 11 2004.

[16] V. Lui, J. Geeves, W. Yii, and T. Drummond. Efficient sub-pixel refinement with symbolic linear predictors. In *2018 IEEE Conference on Computer Vision and Pattern Recognition*, pages 8165–8173, 2018.

[17] Z. Luo, T. Shen, L. Zhou, J. Zhang, Y. Yao, S. Li, T. Fang, and L. Quan. ContextDesc: Local descriptor augmentation with cross-modality context. *2019 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[18] Z. Luo, T. Shen, L. Zhou, S. Zhu, R. Zhang, Y. Yao, T. Fang, and L. Quan. GeoDesc: Learning local descriptors by integrating geometry constraints. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.

[19] Z. Luo, L. Zhou, X. Bai, H. Chen, J. Zhang, Y. Yao, S. Li, T. Fang, and L. Quan. ASLFeat: Learning local features of accurate shape and localization. *2020 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[20] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide-baseline stereo from maximally stable extremal regions. *Image and Vision Computing*, 22(10):761 – 767, 2004. British Machine Vision Computing 2002.

[21] K. Mikolajczyk and C. Schmid. Scale & affine invariant interest point detectors. *International Journal of Computer Vision*, 60(1):6386, Oct. 2004.

[22] A. Mishchuk, D. Mishkin, F. Radenovic, and J. Matas. Working hard to know your neighbor's margins: Local descriptor learning loss. In *Advances in Neural Information Processing Systems 30*, pages 4826–4837. 2017.

[23] H. Noh, A. Araujo, J. Sim, T. Weyand, and B. Han. Large-Scale image retrieval With attentive deep local features. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[24] Y. Ono, E. Trulls, P. Fua, and K. M. Yi. LF-Net: Learning local features from images. In *Advances in Neural Information Processing Systems 31*, pages 6234–6244. Curran Associates, Inc., 2018.

[25] J. Revaud, P. Weinzaepfel, C. Roberto de Souza, and M. Humenberger. R2D2: Repeatable and reliable detector and descriptor. In *Advances in Neural Information Processing Systems*, 2019.

[26] E. Rosten and T. Drummond. Fusing points and lines for high performance tracking. In *IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, volume 2, pages 1508–1515 Vol. 2, 2005.

[27] S. Salti, F. Tombari, R. Spezialetti, and L. D. Stefano. Learning a descriptor-specific 3d keypoint detector. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 2318–2326, 2015.

[28] T. Sattler, W. Maddern, C. Toft, A. Torii, L. Hammarstrand, E. Stenborg, D. Safari, M. Okutomi, M. Pollefeys, J. Sivic, F. Kahl, and T. Pajdla. Benchmarking 6DOF outdoor visual localization in changing conditions. In *2018 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[29] T. Sattler, A. Torii, J. Sivic, M. Pollefeys, H. Taira, M. Okutomi, and T. Pajdla. Are Large-Scale 3D models really necessary for accurate visual localization? In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6175–6184, 2017.

[30] T. Sattler, T. Weyand, B. Leibe, and L. Kobbelt. Image Retrieval for image-Based localization revisited. In *Proceedings of the British Machine Vision Conference*, pages 76.1–76.12. BMVA Press, 2012.

[31] N. Savinov, A. Seki, L. Ladicky, T. Sattler, and M. Pollefeys. Quad-Networks: Unsupervised learning to rank for interest point detection. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[32] J. L. Schönberger and J. Frahm. Structure-from-Motion revisited. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4104–4113, 2016.

[33] J. L. Schönberger, H. Hardmeier, T. Sattler, and M. Pollefeys. Comparative evaluation of hand-crafted and learned local Features. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6959–6968, 2017.

[34] E. Simo-Serra, E. Trulls, L. Ferraz, I. Kokkinos, P. Fua, and F. Moreno-Noguer. Discriminative learning of deep convolutional feature point descriptors. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 118–126, 2015.

[35] S. Suwajanakorn, N. Snavely, J. J. Tompson, and M. Norouzi. Discovery of latent 3D keypoints via end-to-end geometric reasoning. In *Advances in Neural Information Processing Systems 31*, pages 2059–2070. 2018.

[36] H. Heijnen T. Yang, D. Nguyen and V. Balntas. UR2KiD: Unifying retrieval, keypoint detection, and keypoint description without local correspondence supervision. *arXiv preprint arXiv:2001.07252*, 2020.

[37] Y. Tian, B. Fan, and F. Wu. L2-Net: Deep learning of discriminative patch descriptor in Euclidean space. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6128–6136, 2017.

[38] D. Ponsa V. Balntas, E. Riba and K. Mikolajczyk. Learning local feature descriptors with triplets and shallow convolutional neural networks. In *Proceedings of the British Machine Vision Conference (BMVC)*, pages 119.1–119.11. BMVA Press, 2016.

[39] K. M. Yi, E. Trulls, V. Lepetit, and P. Fua. LIFT: Learned invariant feature transform. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 467–483, 2016.

[40] L. Zhang and S. Rusinkiewicz. Learning to detect features in texture images. In *2018 IEEE Conference on Computer Vision and Pattern Recognition*, pages 6325–6333, 2018.