ASLFeat: Learning Local Features of Accurate Shape and Localization

Zixin Luo\textsuperscript{1} Lei Zhou\textsuperscript{1} Xuyang Bai\textsuperscript{1} Hongkai Chen\textsuperscript{1} Jiahui Zhang\textsuperscript{2}
Yao Yao\textsuperscript{1} Shiwei Li\textsuperscript{3} Tian Fang\textsuperscript{3} Long Quan\textsuperscript{1}
\textsuperscript{1}Hong Kong University of Science and Technology
\textsuperscript{2}Tsinghua University \textsuperscript{3}Everest Innovation Technology
\{zluoag,lzhouai,xbaiad,hchencf,yaoag,quan\}@cse.ust.hk
jiahui-z15@mails.tsinghua.edu.cn \{sli,fangtian\}@altizure.com

Abstract

This work focuses on mitigating two limitations in the joint learning of local feature detectors and descriptors. First, the ability to estimate the local shape (scale, orientation, etc.) of feature points is often neglected during dense feature extraction, while the shape-awareness is crucial to acquire stronger geometric invariance. Second, the localization accuracy of detected keypoints is not sufficient to reliably recover camera geometry, which has become the bottleneck in tasks such as 3D reconstruction. In this paper, we present ASLFeat, with three light-weight yet effective modifications to mitigate above issues. First, we resort to deformable convolutional networks to densely estimate and apply local transformation. Second, we take advantage of the inherent feature hierarchy to restore spatial resolution and low-level details for accurate keypoint localization. Finally, we use a peakiness measurement to relate feature responses and derive more indicative detection scores. The effect of each modification is thoroughly studied, and the evaluation is extensively conducted across a variety of practical scenarios. State-of-the-art results are reported that demonstrate the superiority of our methods. [code release]

1. Introduction

Designing powerful local features is an essential basis for a broad range of computer vision tasks \cite{31, 43, 44, 30, 40, 15, 40}. During the past few years, the joint learning of local feature detectors and descriptors has gained increasing popularity, with promising results achieved in real applications. However, there are two limitations we consider that may have hinged further boost in performance: 1) the lack of shape-awareness of feature points for acquiring stronger geometric invariance, and 2) the lack of keypoint localization accuracy for solving camera geometry robustly.

Traditionally, the local shape is parameterized by hand-crafted scale/rotation estimation \cite{17, 29} or affine shape adaptation \cite{20}, while more recently, data-driven approaches \cite{23, 22, 39} have emerged that build a separate network to regress the shape parameters, then transform the patch inputs before feature descriptions. Due to the increasing prevalence of the joint learning with keypoint detectors \cite{6, 25, 27, 7, 4}, recent research focus has shifted to frameworks that densely extract features from image inputs, whereas no pre-defined keypoint is given and thus previous patch-wise shape estimation becomes inapplicable. As an alternative, LF-Net \cite{25} extracts dense features and transforms intermediate feature maps via Spatial Transformer Networks (STN) \cite{12}, whereas multiple forward passes are needed and only sparse predictions of shape parameters are practically feasible. In this view, there still lacks a solution that enables efficient local shape estimation in a dense prediction framework.

Besides, the localization accuracy of learned keypoints is still concerned in solving geometry-sensitive problems. For instance, LF-Net \cite{25} and D2-Net \cite{7} empirically yield low precision in two-view matching or introduce large reprojection error in Structure-from-Motion (SfM) tasks, which in essence can be ascribed to the lack of spatial accuracy as the detections are derived from low-resolution feature maps (e.g., 1/4 times the original size). To restore the spatial resolution, SuperPoint \cite{6} learns to upsample the feature maps with pixel-wise supervision from artificial points, while R2D2 \cite{27} employs dilated convolutions to maintain the spatial resolution but trades off excessive GPU computation and memory usage. Moreover, it is questionable that if the detections from the deepest layer are capable of identifying low-level structures (corners, edges, etc.) where keypoints are often located. Although widely discussed in dense prediction tasks \cite{28, 10, 16}, in our context, neither the keypoint localization accuracy, nor the low-level nature of keypoint detection has received adequate attention.

To mitigate above limitations, we present ASLFeat, with three light-weight yet effective modifications. First, we em-
ploy deformable convolutional networks (DCN) [5, 45] in
the dense prediction framework, which allows for not only
pixel-wise estimation of local transformation, but also pro-
gressive shape modelling by stacking multiple DCNs. Sec-
tion, we leverage the inherent feature hierarchy, and propose
a multi-level detection mechanism that restores not only the
spatial resolution without extra learning weights, but also
low-level details for accurate keypoint localization. Finally,
we base our methods on an improved D2-Net [7] that is
trained from scratch, and further propose a peakiness mea-
surement for more selective keypoint detection.

Despite the key insights of above modifications being
familiar, we address their importance in our specific con-
text, fully optimize the implementation in a non-trivial way,
and thoroughly study the effect by comparing with differ-
ent design choices. To summarize, we aim to provide an-
wswers to two critical questions: 1) what deformation param-
ereterization is needed for local descriptors (geometrically
constrained [23, 22, 39] or free-form modelling [5, 45]),
2) what feature fusion is effective for keypoint detectors
(multi-scale input [27, 7], in-network multi-scale inference [25], or multi-level fusion [28]). Finally, we exten-
sively evaluate our methods across various practical scenar-
is, including image matching [1, 2], 3D reconstruction [32]
and visual localization [30]. We demonstrate drastic im-
provements upon the backbone architecture, D2-Net, and
report state-of-the-art results on popular benchmarks.

2. Related works

Hand-crafted local features have been widely evaluated
in [1, 32], we here focus mainly on the learning approaches.
Local shape estimation. Most existing descriptor learning
methods [19, 18, 21, 37, 36, 40] do not explicitly model
the local shape, but rely on geometric data augmentation
(scaling/rotational perturbation) or hand-crafted shape es-
timation (scale/rotation estimation [17, 29]) to acquire
gometric invariance. Instead, OriNet [23] and LIFT [39]
propose to learn a canonical orientation of feature points,
AffNet [22] predicts more affine parameters to improve
the modelling power, and the log-polar representation [8]
is used to handle in particular scale changes. Despite the
promising results, these methods are limited to take im-
image patches as input, and introduce a considerable amount
of computation since two independent networks are con-
structed for predicting patch shape and patch description
separately. As an alternative, LF-Net [25] takes images
as input and performs STN [12] on intermediate features,
while multiple forward passes are needed to transform in-
dividual “feature patch”, and thus only prediction on sparse
locations is practically applicable.

Meanwhile, the modelling of local shape has been shown
crucial in image recognition tasks, which inspires works
such as scale-adaptive convolution (SAC) for flexible-size
dilations [42] and deformable convolution networks (DCN)
for tunable grid sampling locations [5, 45]. In this paper, we
adopt the similar idea in our context, and propose to equip
DCN for dense local transformation prediction, of which
the inference requires only a single forward pass and is thus
of high efficiency.

Joint local feature learning. The joint learning of feature
detectors and descriptors has received increasing attention,
where a unified network is constructed to share most com-
putations of the two tasks for fast inference. In terms of
descriptor learning, the ranking loss [25, 7, 6, 4, 27] has
been primarily used as a de-facto standard. However, due
to the difficulty of acquiring unbiased ground-truth data, no
general consensus has yet been reached regarding an effec-
tive loss design for keypoint detector learning. For instance,
LF-Net [25] warps the detection map and minimizes the dif-
ference at selected pixels in two views, while SuperPoint [6]
operates a self-supervised paradigm with a bootstrap train-
ing on synthetic data and multi-round adaptations on real
data. More recent R2D2 [27] enforces grid-wise peak-
ness in conjunction with reliability prediction for descriptor,
offsets to localize keypoints.

By contrast, D2-Net [7] eschews learning extra weights
for a keypoint detector, but hand-crafts a selection rule to
derive keypoints from the same feature maps that are used
for extracting feature descriptors. This design essentially
couples the capability of the feature detector and descriptor,
and results in a clean framework without complex heuristics
in loss formulation. However, it is a known issue that D2-
Net lacks of accuracy of keypoint localization, as keypoints
are derived from low-resolution feature maps. In this paper,
we base our methods on D2-Net, and mitigate above limit-
ation by a light-weight modification that cheaply restores
both the spatial resolution and low-level details.

3. Methods

3.1. Prerequisites

The backbone architecture in this work is built upon
1) deformable convolutional networks (DCN) [5, 45] that
predict and apply dense spatial transformation, and 2) D2-

Deformable convolutional networks (DCN) [5, 45] target
to learn dynamic receptive filed to accommodate the ability
of modelling geometric variations. Formally, given a regu-
lar grid \( R \) that samples values over the input feature maps
\( x \), the output features \( y \) of a standard convolution for each
spatial position \( p \) can be written as:

\[
y(p) = \sum_{p_n \in R} w(p_n) \cdot x(p + p_n),
\]

DCN augments the regular convolution by additionally
learning both sampling offsets \([\{\Delta p_n|n = 1, ..., N\}\) and feature amplitudes \([\{\Delta m_n|n = 1, ..., N\}\), where \(N = |\mathcal{R}|\), and rewrites Eq. 1 as:

\[
y(p) = \sum_{p_n \in \mathcal{R}} w(p_n) \cdot \mathbf{x}(p + p_n + \Delta p_n) \cdot \Delta m_n.
\]

As the offset \(\Delta p_n\) is typically fractional, Eq. 2 is implemented via bilinear interpolation, while the feature amplitude \(\Delta m_n\) is limited to \((0, 1)\). During training, the initial values of \(\Delta p_n\) and \(\Delta m_n\) are respectively set to 0 and 0.5, following the settings in [45].

**D2-Net** [7] proposes a describe-and-detect strategy to jointly extract feature descriptions and detections. Over the last feature maps \(y \in \mathbb{R}^H \times W \times C\), D2-Net applies channel-wise L2-normalization to obtain dense feature descriptors, while the feature detections are derived from 1) the local score and 2) the channel-wise score. Specifically, for each location \((i, j)\) in \(y^c\) \((c = 1, 2, ..., C)\), the local score is obtained by:

\[
\alpha_{ij}^c = \frac{\exp(y_{ij}^c)}{\sum_{(i', j') \in \mathcal{N}(i, j)} \exp(y_{ij'}^c)},
\]

where \(\mathcal{N}(i, j)\) is neighboring pixels around \((i, j)\), e.g., 9 neighbours defined by a \(3 \times 3\) kernel. Next, the channel-wise score is obtained by:

\[
\beta_{ij}^c = \frac{y_{ij}^c}{\max_j y_{ij}^c}.
\]

The final detection score is combined as:

\[
s_{ij} = \max_i (\alpha_{ij}^c \beta_{ij}^c).
\]

The detection score will be later used as a weighting term in loss formulation (Sec. 3.4), and will allow for top-K selection of keypoints during testing.

### 3.2. DCN with Geometric Constraints

The original free-form DCN predicts local transformation of high degrees of freedom (DOF), e.g., \(9 \times 2\) offsets for a \(3 \times 3\) kernel. On the one hand, it enables the potential to model complex deformation such as non-planarity, while on the other hand, it takes a risk of over-parametrizing the local shape, where simpler affine or perspective transformation are often considered to serve as a good approximation [20, 23, 22]. To find out what deformation is needed in our context, we compare three shape modellings via enforcing different geometric constraints in DCN, including 1) similarity, 2) affine and 3) homography. The shape properties of the investigated variants are summarized in Tab. 1.

#### Affine-constrained DCN

Traditionally, the local shape is often modelled by similarity transformation with estimates of rotation and scale [17, 29]. In a learning framework such as [23, 25], this transformation is decomposed as:

\[
\mathbf{S} = \lambda R(\theta) = \lambda \begin{pmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{pmatrix}.
\]

Moreover, a few works such as HesAff [20] further includes an estimate of shearing, which is cast as a learnable problem by AffNet [22]. Here, we follow AffNet and decompose the affine transformation as:

\[
\mathbf{A} = \mathbf{S} \mathbf{A}' = \lambda R(\theta) \mathbf{A}'
\]

\[
= \lambda \begin{pmatrix} a_{11} & a_{12} & 0 \\ a_{21} & a_{22} & 0 \\ 0 & 0 & 1 \end{pmatrix},
\]

where \(\det \mathbf{A}' = 1\). The network is implemented to predict one scalar for scaling \((\lambda)\), another two for rotation \((\cos(\theta), \sin(\theta))\), while the other three for shearing \((A')\).
Homography-constrained DCN. Virtually, the local deformation can be better approximated by a homography (perspective) transformation $\mathbf{H}$, and we here adopt the Tensor Direct Linear Transform (Tensor DLT) [24] to solve the 4-point parameterization of $\mathbf{H}$ in a differentiable manner.

Formally, a linear system can be created that solves $\hat{\mathbf{M}} \mathbf{h} = \mathbf{0}$, where $\hat{\mathbf{M}} \in \mathbb{R}^{9 \times 9}$ and $\mathbf{h}$ is a vector with 9 elements consisting of the entries of $\mathbf{H}$, and each correspondence provides two equations in $\mathbf{M}$. By enforcing the last element of $\mathbf{h}$ to equal $1$ [11] and omitting the translation, we set $H_{33} = 1$ and $H_{13} = H_{23} = 0$, then rewrite the above system of equations as $\hat{\mathbf{M}}(\mathbf{h}) = \mathbf{b}(\mathbf{i})$, where $\hat{\mathbf{M}}(\mathbf{i}) \in \mathbb{R}^{2 \times 6}$ and for each correspondence,

$$
\hat{\mathbf{M}}(\mathbf{i}) = \begin{bmatrix}
0 & 0 & -u_i & -v_i & u'_i u_i & v'_i v_i \\
u_i & v_i & 0 & 0 & -u'_i u_i & -v'_i v_i
\end{bmatrix},
$$

(8)

$$
\hat{\mathbf{b}}(\mathbf{i}) = [-u'_i, u'_i]^T \in \mathbb{R}^{2 \times 1} \text{ and } \hat{\mathbf{h}} \text{ consists of 6 elements from the first two columns of } \mathbf{H} \text{. By stacking the equations of 4 correspondences, we derive the final linear system:}
$$

$$
\hat{\mathbf{M}} \mathbf{h} = \hat{\mathbf{b}}.
$$

(9)

Suppose that correspondence points are not collinear, $\hat{\mathbf{h}}$ can be then efficiently and uniquely solved by using the differentiable pseudo-inverse of $\hat{\mathbf{A}}$$^1$. In practice, we initialize 4 corner points at $\{(1,1), (-1,1), (1,-1), (-1,-1)\}$, and implement the network to predict 8 corresponding offsets lying in $(-1,1)$ so as to avoid collinearity.

After forming the above transformation $\mathbf{T} \in \{\mathbf{S}, \mathbf{A}, \mathbf{H}\}$, the offset values in Eq. 2 are now obtained by:

$$
\Delta \mathbf{p}_n = \mathbf{T} \mathbf{p}_n - \mathbf{p}_n, \text{ where } \mathbf{p}_n \in \mathcal{R},
$$

(10)

so that geometry constraints are enforced in DCN. More implementation details can be found in the Appendix.

3.3. Selective and Accurate Keypoint Detection

Keypoint peakiness measurement. As introduced in Sec. 3.1, D2-Net scores a keypoint regarding both spatial and channel-wise responses. Among many possible metrics, D2-Net implements a ratio-to-max (Eq. 4) to evaluate channel-wise extremeness, whereas one possible limitation lies on that it only weakly relates to the actual distribution of all responses along the channel.

To study this effect, we first trivially modify Eq. 4 with a channel-wise softmax, whereas this modification deteriorates the performance in our experiments. Instead, inspired by [27, 41], we propose to use peakiness as a keypoint measurement in D2-Net, which rewrites Eq. 4 as:

$$
\beta_{ij}^c = \text{softplus}(y_{ij}^c - \frac{1}{C} \sum_t y_{ij}^t),
$$

(11)

$^1$Implemented via function tf.matrix_solve in TensorFlow.
by having different sizes of receptive field for localizing keypoints. Secondly, compared with U-Net architecture, it cheaply restores the spatial resolution without introducing extra learning weights to achieve pixel-wise accuracy. Thirdly, different from U-Net that directly fuses low-level and high-level features, it keeps the low-level features untouched, but fuses the detections of multi-level semantics, which helps to better preserve the low-level structures such as corners or edges. The implementation details of above variants can be found in the Appendix.

3.4. Learning Framework

Network architecture. The network architecture is illustrated in Fig. 1. To reduce computations, we replace the VGG backbone [34] used in D2-Net with more light-weight L2-Net [36]. Similar to R2D2 [27], we further replace the last 8 × 8 convolution of L2-Net by three 3 × 3 convolutions, resulting in feature maps of 128 dimension and 1/4 times resolution of the input. Finally, the last three convolutions, conv6, conv7 and conv8, are substituted with DCN (Sec. 3.1). Three levels, conv1, conv3 and conv8, are selected to perform the proposed MulDet (Sec. 3.3). The combination weights in Eq. 13 are empirically set to $w_i = 1, 2, 3$, and the dilation rate to find neighboring pixels $N(i, j)$ in Eq. 3 is set to 3, 2, 1, respectively, which we find to deliver best trade-offs to balance the attention on low-level and abstracted features.

Loss design. We identify a set of correspondences $C$ for an image pair $(I, I')$ via densely warping $I$ to $I'$ regarding ground-truth depths and camera parameters. To derive the training loss for both detector and descriptor, we adopt the formulation in D2-Net [7], written as:

$$
\mathcal{L}(I, I') = \frac{1}{|C|} \sum_{c \in C} \hat{s}_c \hat{s}_c' \mathcal{M}(f_c, f_c'),
$$

where $\hat{s}_c$ and $\hat{s}_c'$ are combined detection scores in Eq. 13 for image $I$ and $I'$, $f_c$ and $f_c'$ are their corresponding descriptors, and $\mathcal{M}(\cdot, \cdot)$ is the ranking loss for representation learning. Instead of using the hardest-triplet loss in D2-Net [7], we adopt the hardest-contrastive form in FCGF [3], which we find guarantee better convergence when training from scratch and equipping DCN, written as:

$$
\mathcal{M}(f_c, f_c') = [D(f_c, f_c') - m_p]_+ + [m_n - \min_{k \neq c}(\min_{k \neq c} D(f_c, f_c'))]_+,
$$

where $D(\cdot, \cdot)$ denotes the Euclidean distance measured between two descriptors, and $m_p, m_n$ are respectively set to 0.2, 1.0 for positives and negatives. Similar to D2-Net [7], a safe radius sized 3 is set to avoid taking spatially too close feature points as negatives.

3.5. Implementations

Training. In contrast to D2-Net [7] which starts from an ImageNet pretrained model with only the last convolution fine-tuned, we train our model from scratch with ground-truth cameras and depths obtained from [33, 26] (the same data used in [19, 18]). The training consumes 800K image pairs sized 480 × 480 and batched 2. Learning gradients are computed for image pairs that have at least 32 matches, while the maximum match number is limited to 512. Each input image is standardized to have zero mean and unit norm, and independently applied with random photometric augmentation including brightness, contrast and blurriness. The SGD optimizer is used with momentum of 0.9, and the base learning rate is set to 0.1.

Although an end-to-end learning with DCN is feasible, we find that a two-stage training yields better results in practice. Specifically, in the first round we train the model with all regular convolutions for 400K iterations. In the second round, we tune only the DCNs with the base learning rate divided by 10 for another 400K iterations. Our implementation is made in TensorFlow with single NVIDIA RTX 2080Ti card, and the training finishes within 42 hours.

Testing. A non-maximum suppression (NMS) sized 3 is applied to remove detections that are spatially too close. Similar to D2-Net, we postprocess the keypoints with the SIFT-like edge elimination (with threshold set to 10) and sub-pixel refinement, the descriptors are then bilinearly interpolated at the refined locations. We select top-K keypoints regarding detection scores obtained in Eq. 13, and empirically discard those whose scores are lower than 0.50.

4. Experiments

In the following sections, we evaluate our methods across several practical scenarios, including image matching, 3D reconstruction and visual localization. Further experiments on dense reconstruction and image retrieval can be found in the Appendix.

4.1. Image Matching

Datasets. First, we use the popular HPatches dataset [1], which includes 116 image sequences with ground-truth homography. Following D2-Net [7], we exclude 8 high-resolution sequences, leaving 52 and 56 sequences with illumination or viewpoint variations, respectively.

Though widely used, HPatches dataset exhibits only homography transformation, which may not comprehensively reflect the performance in real applications. Thus, we resort to the newly proposed FM-Bench [2], which comprises four datasets captured in practical scenarios: the TUM dataset [35] in indoor SLAM settings, the KITTI dataset [9] in driving scenes, the Tanks and Temples dataset
(T&T) [13] for wide-baseline reconstruction, and the Community Photo Collection (CPC) [38] for wild reconstruction from web images. For each datasets, 1000 overlapping image pairs are randomly chosen for evaluation, with ground-truth fundamental matrix pre-computed.

**Evaluation protocols.** On HPatches dataset [1], three standard metrics are used: 1) Keypoint repeatability (%Rep.), a.k.a. the ratio of possible matches and the minimum number of keypoints in the shared view. 2) Matching score (%M.S.), a.k.a. the ratio of correct matches and the minimum number of keypoints in the shared view. 3) Mean matching accuracy (%MMA), a.k.a. the ratio of correct matches and possible matches. Here, a match is defined to correspond if the point distance is below some error threshold after homography wrapping, and a correct match is further required to be a mutual nearest neighbor during brute-force searching. For above metrics, we report their average scores over all image pairs in the dataset.

In terms of FM-Bench [2], a full matching pipeline including outlier rejection (e.g., ratio test [17]) and geometric verification (e.g., RANSAC) is performed, and the final pose recovery accuracy is evaluated. To determine the correctness of a pose estimate, FM-Bench uses ground-truth pose to generate a set of virtual correspondences, then measures the average of normalized symmetric epipolar distance regarding a pose estimate, and finally computes %Recall as the ratio of estimates where the distance error is below a certain threshold (0.05 by default). At correspondence level, FM-Bench also reports intermediate results such as the inlier ratio (%Inlier/%Inlier-m) and correspondence number (%Corr/%Corr-m) after/before RANSAC.

<table>
<thead>
<tr>
<th>HPatches dataset (error threshold @ 3px)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Config.</strong></td>
</tr>
<tr>
<td>----------</td>
</tr>
<tr>
<td>D2-Net</td>
</tr>
<tr>
<td>orig.</td>
</tr>
<tr>
<td>our impl.</td>
</tr>
<tr>
<td>peakiness meas.</td>
</tr>
<tr>
<td>----------</td>
</tr>
<tr>
<td>+ MulDet</td>
</tr>
<tr>
<td>multi-scale (pyramid)</td>
</tr>
<tr>
<td>multi-scale (in-network)</td>
</tr>
<tr>
<td>multi-level (U-Net)</td>
</tr>
<tr>
<td>multi-level (U-Net) w/ peakiness meas.</td>
</tr>
<tr>
<td>----------</td>
</tr>
<tr>
<td>+ MulDet &amp; DCN</td>
</tr>
<tr>
<td>s.t. similarity</td>
</tr>
<tr>
<td>s.t. affine</td>
</tr>
<tr>
<td>s.t. homography</td>
</tr>
<tr>
<td>free-form, 7 keypoints</td>
</tr>
<tr>
<td>free-form</td>
</tr>
<tr>
<td>free-form, multi-scale</td>
</tr>
</tbody>
</table>

**Table 2.** Ablation experiments of the proposed modifications, where peakiness meas. improves the detection scoring upon D2-Net, + MulDet studies the effect of different feature fusion strategies, and + MulDet & DCN further compares the effect of different parameterization of deformation.

**Comparative methods.** We compare our methods with 1) patch descriptors, including HardNet++ [21] with SIFT [17] detector (SIFT + HN++), or plus a shape estimator HesAffNet [22] (HAN + HN++). Also, ContextDesc [18] with SIFT detector (SIFT + ContextDesc). 2) Joint local feature learning approaches including SuperPoint [6], LF-Net [25], D2-Net (fine-tuned) [7] and more recent R2D2 [27]. Unless otherwise specified, we report either results reported in original papers, or derived from authors’ public implementations with default parameters. We limit the maximum numbers of features of our methods to 5K and 20K on HPatches dataset and FM-Bench, respectively.

On FM-Bench, both the mutual check and ratio test [17] are applied to reject outliers before RANSAC. A ratio at 0.8 is used for all methods except for D2-Net and R2D2.

**Baseline.** To avoid overstatement, we first present our re-implementation of D2-Net (our impl.) as the baseline. As mentioned in Sec. 3.4 and Sec. 3.5, the new baseline differs from the original D2-Net (orig.) in three aspects: 1) Different backbone architecture (L2-Net [36] with 128-d output vs. VGG [34] with 512-d output). 2) Different loss formulation (hardest-contrastive [3] vs. hardest-triplet [7]). 3) Different training settings (trained from scratch vs. fine-tuned only the last convolution from a pre-trained model). As shown in Tab. 2 and Tab. 3, the new baseline outperforms original D2-Net in general, while being more parameter- and computation-efficient regarding model size.

**Ablations on peakiness measurement.** We first adopt the peakiness measurement for more indicative keypoint scoring (Sec. 3.3). As shown in Tab. 2, this modification (peakiness meas.) notably improves the results regarding all evaluation metrics on HPatches dataset. This effect is validated on FM-Bench, which is shown to apply for all different scenarios as shown in Tab. 3 (ASLFeat w/o peakiness meas.). Our later modifications will be thus based on this model.

**Ablations on MulDet.** As shown in Tab. 2, applying multi-scale detection solely does not take obvious effect, as spatial accuracy is still lacking. Instead, adopting multi-level detection, with spatial resolution restored, remarkably boosts the performance, which conforms the necessity of pixel-level accuracy especially when small pixel error is tolerated. It is also note-worthy that, despite less learning weights and computation, the proposed multi-level detection outperforms the U-Net variant, addressing the low-level nature of this task where a better preservation of low-level features is beneficial. Although the proposed multi-level detection also includes feature fusion of difference scales, we find that combining a more explicit multi-scale (pyramid) detection (free-form, multi-scale) is in particular advantageous in order to handle the scale changes. This combination will be denoted as ASLFeat (MS) in the following context.

**Ablations on DCN.** As shown in Tab. 2, all investigated
variants of DCN are valid and notably boost the performance. Among those designs, the free-form variant slightly outperforms the constrained version, despite the fact that HPatches datasets exhibit only homography transformation. This confirms that modelling non-planarity is feasible and useful for local features, and we thus opt for the free-form DCN to better handle geometric variations. Besides, we also implement a single-layer DCN (free-form, 1 layer) that replaces only the last regular convolution (i.e., conv8 in Fig. 1), showing that stacking more DCNs is beneficial and the shape estimation can be learned progressively.

Figure 3. Comparisons on HPatches dataset [1] with mean matching accuracy (MMA) evaluated at different error thresholds, where “MS” denotes that the multi-scale inference is enabled.

Comparisons with other methods. As illustrated in Fig. 3, both ASLFeat and its multi-scale (MS) variant achieve overall best results on HPatches dataset regarding both illumination and viewpoint variations at different error thresholds. Specifically, ASLFeat delivers remarkable improvements upon its backbone architecture, D2-Net, especially at low error thresholds, which in particular demonstrates that the keypoint localization error has been largely reduced. Besides, ASLFeat notably outperforms other joint learning approaches.

4.2. 3D Reconstruction

Datasets. We resort to ETH benchmark [32] to demonstrate the effect on 3D reconstruction tasks. Following [7], we
evaluate on three medium-scale datasets from [38].

**Evaluation protocols.** We exhaustively match all image pairs for each dataset with both ratio test at 0.8 and mutual check for outlier rejection, then run SFM and MVS algorithms by COLMAP [31]. For sparse reconstruction, we report the number of registered images (#Reg. Images), the number of sparse points (#Sparse Points), average track length (Track Length) and mean reprojection error (Reproj. Error). For dense reconstruction, we report the number of dense points (#Dense Points). We limit the maximum number of features of ASLFeat to 20K.

**Results.** As shown in Tab. 4, ASLFeat produces the most complete reconstructions regarding #Reg. Images and #Dense Points. Besides, ASLFeat results in Reproj. Error that is on par with SuperPoint and smaller than D2-Net, which again validates the effect of the proposed MulDet for restoring spatial information. However, the reprojection error produced by hand-crafted keypoints (e.g., RootSIFT) is still notably smaller than all learning methods, which implies that future effort can be spent to further improve the keypoint localization in a learning framework.

### 4.3. Visual Localization

**Datasets.** We resort to Aachen Day-Night dataset [30] to demonstrate the effect on visual localization tasks, where the key challenge lies on matching images with extreme day-night changes for 98 queries.

**Evaluation protocols.** We use the evaluation pipeline provided in The Visual Localization Benchmark3, which takes custom features as input, then relies on COLMAP [31] for image registration, and finally generates the percentages of successfully localized images within three error tolerances (0.5m, 2°) / (1m, 5°) / (5m, 10°). The maximum feature number of our methods are limited to 20K.

<table>
<thead>
<tr>
<th>Datasets</th>
<th>Methods</th>
<th>#Reg. Images</th>
<th>#Sparse Points</th>
<th>Track Length</th>
<th>Reproj. Error</th>
<th>#Dense Points</th>
</tr>
</thead>
<tbody>
<tr>
<td>Madrid</td>
<td>RootSIFT</td>
<td>500</td>
<td>116K</td>
<td>6.32</td>
<td>0.60px</td>
<td>1.83M</td>
</tr>
<tr>
<td>Metropolis</td>
<td>GeoDesc</td>
<td>495</td>
<td>144K</td>
<td>5.97</td>
<td>0.65px</td>
<td>1.56M</td>
</tr>
<tr>
<td>1344 images</td>
<td>SuperPoint</td>
<td>438</td>
<td>29K</td>
<td>9.03</td>
<td>0.70px</td>
<td>1.55M</td>
</tr>
<tr>
<td></td>
<td>D2-Net (MS)</td>
<td>495</td>
<td>144K</td>
<td>6.39</td>
<td>1.35px</td>
<td>1.46M</td>
</tr>
<tr>
<td></td>
<td>ASLFeat (MS)</td>
<td>649</td>
<td>129K</td>
<td>9.56</td>
<td>0.95px</td>
<td>1.92M</td>
</tr>
<tr>
<td>Gendarmen-</td>
<td>RootSIFT</td>
<td>1035</td>
<td>338K</td>
<td>5.52</td>
<td>0.69px</td>
<td>4.23M</td>
</tr>
<tr>
<td>markt</td>
<td>GeoDesc</td>
<td>1004</td>
<td>444K</td>
<td>5.14</td>
<td>0.73px</td>
<td>3.88M</td>
</tr>
<tr>
<td>1463 images</td>
<td>SuperPoint</td>
<td>965</td>
<td>310K</td>
<td>5.55</td>
<td>1.28px</td>
<td>3.15M</td>
</tr>
<tr>
<td></td>
<td>D2-Net (MS)</td>
<td>1061</td>
<td>320K</td>
<td>8.98</td>
<td>1.05px</td>
<td>4.00M</td>
</tr>
<tr>
<td></td>
<td>ASLFeat (MS)</td>
<td>846</td>
<td>252K</td>
<td>13.16</td>
<td>0.95px</td>
<td>3.08M</td>
</tr>
<tr>
<td>Tower of</td>
<td>RootSIFT</td>
<td>804</td>
<td>239K</td>
<td>7.76</td>
<td>0.61px</td>
<td>3.03M</td>
</tr>
<tr>
<td>London</td>
<td>GeoDesc</td>
<td>776</td>
<td>341K</td>
<td>6.71</td>
<td>0.63px</td>
<td>2.73M</td>
</tr>
<tr>
<td>1576 images</td>
<td>SuperPoint</td>
<td>667</td>
<td>222K</td>
<td>8.72</td>
<td>1.06px</td>
<td>4.01M</td>
</tr>
<tr>
<td></td>
<td>D2-Net (MS)</td>
<td>1004</td>
<td>287K</td>
<td>5.20</td>
<td>1.34px</td>
<td>2.86M</td>
</tr>
<tr>
<td></td>
<td>ASLFeat (MS)</td>
<td>1040</td>
<td>221K</td>
<td>8.72</td>
<td>1.00px</td>
<td>4.00M</td>
</tr>
<tr>
<td></td>
<td>ASLFeat</td>
<td>613</td>
<td>821</td>
<td>12.52</td>
<td>0.92px</td>
<td>3.06M</td>
</tr>
<tr>
<td></td>
<td>RootSIFT</td>
<td>252K</td>
<td>846</td>
<td>6.32</td>
<td>1.05px</td>
<td>4.00M</td>
</tr>
<tr>
<td></td>
<td>GeoDesc</td>
<td>1061</td>
<td>252K</td>
<td>13.16</td>
<td>0.95px</td>
<td>3.08M</td>
</tr>
<tr>
<td></td>
<td>SuperPoint</td>
<td>649</td>
<td>252K</td>
<td>8.72</td>
<td>1.00px</td>
<td>4.00M</td>
</tr>
<tr>
<td></td>
<td>D2-Net (MS)</td>
<td>1004</td>
<td>252K</td>
<td>13.16</td>
<td>0.95px</td>
<td>3.08M</td>
</tr>
<tr>
<td></td>
<td>ASLFeat (MS)</td>
<td>1040</td>
<td>221K</td>
<td>8.72</td>
<td>1.00px</td>
<td>4.00M</td>
</tr>
<tr>
<td></td>
<td>ASLFeat</td>
<td>613</td>
<td>821</td>
<td>12.52</td>
<td>0.92px</td>
<td>3.06M</td>
</tr>
<tr>
<td></td>
<td>RootSIFT</td>
<td>252K</td>
<td>846</td>
<td>6.32</td>
<td>1.05px</td>
<td>4.00M</td>
</tr>
<tr>
<td></td>
<td>GeoDesc</td>
<td>1061</td>
<td>252K</td>
<td>13.16</td>
<td>0.95px</td>
<td>3.08M</td>
</tr>
<tr>
<td></td>
<td>SuperPoint</td>
<td>649</td>
<td>252K</td>
<td>8.72</td>
<td>1.00px</td>
<td>4.00M</td>
</tr>
<tr>
<td></td>
<td>D2-Net (MS)</td>
<td>1004</td>
<td>252K</td>
<td>13.16</td>
<td>0.95px</td>
<td>3.08M</td>
</tr>
<tr>
<td></td>
<td>ASLFeat (MS)</td>
<td>1040</td>
<td>221K</td>
<td>8.72</td>
<td>1.00px</td>
<td>4.00M</td>
</tr>
<tr>
<td></td>
<td>ASLFeat</td>
<td>613</td>
<td>821</td>
<td>12.52</td>
<td>0.92px</td>
<td>3.06M</td>
</tr>
<tr>
<td></td>
<td>RootSIFT</td>
<td>252K</td>
<td>846</td>
<td>6.32</td>
<td>1.05px</td>
<td>4.00M</td>
</tr>
<tr>
<td></td>
<td>GeoDesc</td>
<td>1061</td>
<td>252K</td>
<td>13.16</td>
<td>0.95px</td>
<td>3.08M</td>
</tr>
<tr>
<td></td>
<td>SuperPoint</td>
<td>649</td>
<td>252K</td>
<td>8.72</td>
<td>1.00px</td>
<td>4.00M</td>
</tr>
<tr>
<td></td>
<td>D2-Net (MS)</td>
<td>1004</td>
<td>252K</td>
<td>13.16</td>
<td>0.95px</td>
<td>3.08M</td>
</tr>
<tr>
<td></td>
<td>ASLFeat (MS)</td>
<td>1040</td>
<td>221K</td>
<td>8.72</td>
<td>1.00px</td>
<td>4.00M</td>
</tr>
<tr>
<td></td>
<td>ASLFeat</td>
<td>613</td>
<td>821</td>
<td>12.52</td>
<td>0.92px</td>
<td>3.06M</td>
</tr>
</tbody>
</table>

Table 4. Evaluation results on ETH benchmark [32] for 3D reconstruction.

**Results.** As shown in Tab. 5, although only mediocre results are obtained in previous evaluations, D2-Net performs surprisingly well under challenging illumination variations. This can be probably ascribed to the superior robustness of low-level features pre-trained on ImageNet. On the other hand, our method outperforms the plain implementation of R2D2, while a specialized R2D2 model (R2D2 (fine-tuned)) achieves the state-of-the-art results with doubled model size, training on day image from Aachen dataset and using photo-realistic style transfer to generate night images.

<table>
<thead>
<tr>
<th>Methods</th>
<th>#Features</th>
<th>Dim</th>
<th>0.5m</th>
<th>2°</th>
<th>Im. 5°</th>
<th>5m, 10°</th>
</tr>
</thead>
<tbody>
<tr>
<td>RootSIFT</td>
<td>11K</td>
<td>128</td>
<td>33.7</td>
<td>52.0</td>
<td>65.3</td>
<td></td>
</tr>
<tr>
<td>HAN + HN++</td>
<td>11K</td>
<td>128</td>
<td>37.8</td>
<td>54.1</td>
<td>75.5</td>
<td></td>
</tr>
<tr>
<td>SIFT + ContextDesc</td>
<td>11K</td>
<td>128</td>
<td>40.8</td>
<td>55.1</td>
<td>80.6</td>
<td></td>
</tr>
<tr>
<td>SuperPoint</td>
<td>79K</td>
<td>256</td>
<td>8.96</td>
<td>57.1</td>
<td>75.3</td>
<td></td>
</tr>
<tr>
<td>D2-Net (MS)</td>
<td>19K</td>
<td>512</td>
<td>44.9</td>
<td>64.3</td>
<td>88.8</td>
<td></td>
</tr>
<tr>
<td>R2D2 (MS)</td>
<td>10K</td>
<td>128</td>
<td>43.9</td>
<td>61.2</td>
<td>77.6</td>
<td></td>
</tr>
<tr>
<td>R2D2 (MS, fine-tuned)</td>
<td>10K</td>
<td>128</td>
<td>45.0</td>
<td>66.3</td>
<td>88.8</td>
<td></td>
</tr>
<tr>
<td>D2-Net (our impl.)</td>
<td>10K</td>
<td>128</td>
<td>40.8</td>
<td>79.8</td>
<td>77.8</td>
<td></td>
</tr>
<tr>
<td>ASLFeat</td>
<td>11K</td>
<td>256</td>
<td>52.0</td>
<td>80.6</td>
<td>111.0</td>
<td>192.9</td>
</tr>
<tr>
<td>ASLFeat (MS)</td>
<td>10K</td>
<td>128</td>
<td>44.9</td>
<td>64.3</td>
<td>85.7</td>
<td></td>
</tr>
</tbody>
</table>


### 5. Conclusions

In this paper, we have used D2-Net as the backbone architecture to jointly learn the local feature detector and descriptor. Three light-weight yet effective modifications have been proposed that drastically boost the performance in two aspects: the ability to model the local shape for stronger geometric invariance, and the ability to localize keypoints accurately for solving robust camera geometry. We have conducted extensive experiments to study the effect of each modification, and demonstrated the superiority and practicability of our methods across various applications.

**Acknowledgments.** This work is supported by Hong Kong RGC GRF 16206819, 16203518 and T22-603/15N.

---

3https://www.visuallocalization.net/
References


[22] D. Mishkin, F. Radenovic, and J. Matas. Repeatability is not enough: Learning affine regions via discriminability. In ECCV, 2018. 1, 2, 3, 6, 7

[23] K. Moo Yi, Y. Verdie, P. Fua, and V. Lepetit. Learning to assign orientations to feature points. In CVPR, 2016. 1, 2, 3


[38] K. Wilson and N. Snavely. Robust global translations with ioufn. In ECCV, 2014. 6, 7, 8


