Beyond Cartesian Representations for Local Descriptors

Patrick Ebel[1], Anastasiia Mishchuk[1], Kwang Moo Yi[2], Pascal Fua[1], Eduard Trulls[3]
[1]Computer Vision Lab, École Polytechnique Fédérale de Lausanne
[2]Visual Computing Group, University of Victoria [3]Google Switzerland
{firstname.lastname}@epfl.ch, kyi@uvic.ca, trulls@google.com

Abstract

The dominant approach for learning local patch descriptors relies on small image regions whose scale must be properly estimated a priori by a keypoint detector. In other words, if two patches are not in correspondence, their descriptors will not match. A strategy often used to alleviate this problem is to "pool" the pixel-wise features over log-polar regions, rather than regularly spaced ones.

By contrast, we propose to extract the "support region" directly with a log-polar sampling scheme. We show that this provides us with a better representation by simultaneously oversampling the immediate neighbourhood of the point and undersampling regions far away from it. We demonstrate that this representation is particularly amenable to learning descriptors with deep networks. Our models can match descriptors across a much wider range of scales than was possible before, and also leverage much larger support regions without suffering from occlusions. We report state-of-the-art results on three different datasets.

1. Introduction

Keypoint matching has played a pivotal role in computer vision for well over a decade. This is clearly demonstrated by the fact that SIFT [23] remains the most cited paper in computer vision history. While many areas of computer vision are currently dominated by dense deep networks, that is, methods that take entire images as input, some problems remain best approached using sparse features. For example, despite recent attempts at tackling 6DOF pose estimation using dense networks, the top-performing models for wide-baseline stereo and large-scale Structure-from-Motion (SfM) still rely on sparse features [49, 51, 33].

As a result, the quest for ever-improving local feature descriptors goes on [23, 5, 46, 42, 39, 12, 50, 38, 41, 28, 45, 19, 25, 15, 24, 10, 31]. These methods all seek to achieve invariance to small changes in location, orientation, scale, perspective, and illumination, along with imaging artefacts and partial occlusions. Most descriptors, however, whether learned or hand-crafted, operate on SIFT-like keypoints and thus rely on simple heuristics to estimate the scale. If the scales for two keypoints do not correspond, neither will the support regions used to extract their descriptors, which is widely accepted as an unrecoverable situation. This is damaging because scale detection is often unreliable.

In this paper we demonstrate that this does not *need* to be the case. To this end, we go beyond the current paradigm for local descriptors, which we call the *cartesian* approach. This paradigm confines local descriptors to small, regularly sampled regions and relies on accurate scale estimates. By contrast, we posit that extracting the *support region* with a log-polar sampling scheme allows us to generate a better local representation by oversampling the immediate neighborhood of the point. We show that this approach is conducive to learning scale-invariant descriptors with off-the-shelf deep networks, enabling us to match keypoints across mismatched scales; see Fig. 4. Furthermore, we demonstrate that this representation is far less sensitive to occlusions or background motion than its cartesian counterpart, which allows us to exploit much larger image regions than was possible before to further boost performance.

Note that while log-polar representations have been used extensively by local features, this has typically involved log-polar aggregation of local statistics that are still computed on the cartesian image grid. By contrast, we propose to *warp* the patch using a log-polar sampling scheme and learn an optimal descriptor on this data. Fig. 1 illustrates the difference between these two approaches.

In short, we propose a new approach to represent local patches and show how to leverage it to achieve scale invariance. In the remainder of the paper, we first briefly review how scale has been handled in the vast body of literature pertaining to matching descriptors, whether learned or designed. We then describe our method and show that it outperforms the state of the art on several challenging datasets.

This research was partially funded by Google's Visual Positioning System, the Swiss National Science Foundation, the Natural Sciences and Engineering Research Council of Canada, and by Compute Canada.

2. Related works

In this section we first review techniques representative of the many that have been proposed to achieve scale invariance for local feature matching, with and without explicit scale detection. Next, we discuss approaches to learning models for patch descriptors. Finally, we study the use of log-polar representations in local features. For a thorough, up-to-date survey on local features please refer to [8].

Scale Invariance via Scale Detection. The vast majority of work in the literature assumes that scale estimation is handled by the keypoint detector and that keypoints can be put in correspondence only if their scales match. This includes classical hand-crafted pipelines such as SIFT [23] or SURF [5]. Image measurements are then aggregated over a correspondingly-sized support region to extract the descriptor. As a result, errors in this *a priori* scale estimation cannot be recovered from, and the affected keypoints are simply written off as potential correspondences.

Two-stage pipelines. Special strategies can be used for rigid matching under large zoom. Zhou *et al*. [52] propose a two-stage approach to first coarsely register the image in scale-space and then narrow down the search scope to matches of commensurate scale. Shan *et al*. [36] assume that dense SfM models are available, along with an approximate pose, and synthesize ground views from aerial viewpoints using the 3D model, for aerial-to-ground matching. Both methods rely on SIFT features and would directly benefit from improved, scale-invariant descriptors such as ours.

Scale Invariance without Scale Detection. A simple way to achieve scale invariance is to concatenate multi-scale descriptors and find the best match among them. This was done in [47] to improve robustness against scale changes with ORB features [32]. Scale-Less SIFT (SLS) [14] goes beyond that and exploits the observation that SIFT descriptors do not change drastically over close, contiguous scales, which suggests that they are embedded in a low-dimensional space. This observation can be used to find a representation more compact than their concatenation. The resulting feature vectors are still high-dimensional (8k) but can be reduced by PCA to a 512-dimensional vector. However, this requires a singular value decomposition for each keypoint to find its subspace, which is very costly.

The Scale and rotation-Invariant Descriptor (SID) [20] samples axis-aligned derivatives over a log-polar grid, along with incremental smoothing over image regions further away from the keypoint. Thus, scale changes and rotations result in translations on the measurement matrix. Using the Fourier Transform Modulus of this signal, which is translation-invariant, makes the descriptors scale- and rotation-invariant. However, SID requires fine sampling over large support regions, which fails in real-world scenarios with viewpoint changes and occlusions. Seg-SID [43] addresses this shortcoming by leveraging segmentation cues to suppress image measurements from image regions not associated to the keypoint, but this requires image-level segmentation and is failure-prone. SID also suffers from high dimensionality ($\sim$3k).

More importantly, both SID and SLS were designed for dense matching with SIFT Flow [22] as a back-end and are not suitable for large-scale reconstruction due to their computational cost. Finally, they both rely on hand-crafted features and cannot compete with the machine learning models that currently dominate the field. We now turn to these.

Learned Descriptors. Early works applied PCA to SIFT [18], learned comparison metrics [40], or learned descriptors with convex optimization [39]. Current research on patch descriptors is dominated by convolutional neural networks. MatchNet [12] and DeepCompare [50] train descriptor extraction and distance metric networks using a Siamese architecture. DeepDesc [38] uses hard positive and negative mining to learn discriminative features. A triplet-based loss is introduced in [4]. L2-Net [41] improves the loss function by enforcing similarity in the intermediate feature maps and penalizing highly correlated descriptor bins. HardNet [28] extends the formulation of [38] to mine over all the samples in the batch. In [15], mining heuristics are replaced by a differentiable approximation of the average precision metric that is then used for optimization. Spectral pooling is introduced in [45] to deal with geometric transformations. An alternative to siamese- and triplet-based loss functions is proposed in [19] to address their shortcomings. GeoDesc [25] uses geometry constraints for optimization. ContextDesc [24] incorporates global context, and geometric context from the keypoint distribution.

All of the deep methods, except [25, 24], are trained on the same dataset [7], which consists of patches pre-extracted on keypoints using Difference of Gaussians (DoG) [23] or multi-scale Harris corners [13]. Only keypoints that survive a 3D reconstruction by Structure from Motion (SfM) are considered, and similarly to the traditional approach, the learned models are simply expected to fail if the detector fails first. To the best of our knowledge there is no learning-based method that explicitly addresses scale invariance.

Another line of works comprises those that use deep architectures to learn keypoints and descriptors jointly. LIFT [48] is trained on patches extracted around SIFT keypoints with corresponding scales, similarly to the previous methods. LF-Net [29] learns to detect the scale with self-supervision, but in practice seems to perform best over a very narrow set of scales. SuperPoint [9] learns scale invariance at the descriptor level, which works for visual odometry but breaks in more generalized problems. D2-Net [10] focuses on difficult imaging conditions and relies on a single network for detection and description. R2D2 [31] ap-

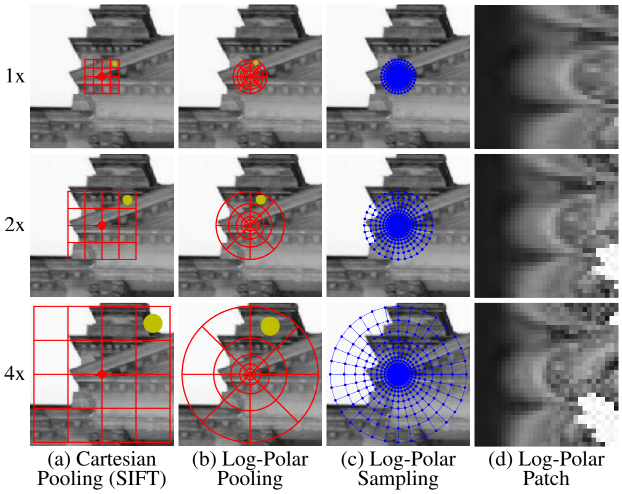


Figure 1: **Pooling vs Sampling.** (a,b) The red patterns depict the regions that most descriptors use to *pool* features computed on a cartesian pixel-grid. The size of the pattern depends on the local scale, and we show three versions. Under large scale changes, many regions of the cartesian and log-polar grids, such as the ones highlighted by the yellow dots, can no longer be put in correspondence. (c) By contrast, we first resample the patch according to the patterns shown in blue (32×32). Their size also depends on the scale. (d) Even though the scales vary from 1 to 4, the resulting log-polar patches are all fairly similar, notably near the center of the feature location, depicted by the left side of the patch.

plies L2-Net convolutionally while penalizing repeatable but non-discriminative patches.

Leveraging Polar representations. Polar and log-polar representations have been extensively used in computer vision to aggregate local information, because they are robust to small changes in scale and rotation. Traditional handcrafted patch descriptors typically consist of two stages: feature *extraction* and feature *pooling*. First, image measurements such as gradients are computed for every pixel. Then, they are aggregated over small regions around the point given its location, orientation, and scale. SIFT, for instance, aggregates features (histograms of gradient orientations) over 4×4 cells around the keypoint; see Fig. 1.

Several descriptors aggregate features over polar or log-polar regions. GLOH [26] computes SIFT over a log-polar grid and then reduces the dimensionality by PCA. Daisy [42] aggregates oriented image gradients over a polar grid using a Gaussian kernel with a size proportional to the distance between the keypoint and the grid point, to bypass aliasing effects. The seminal Shape Contexts paper [6] introduces a descriptor for object recognition by picking points on the contour of a shape and histogramming the location of each point relative to every other point over log-polar bins. Local Self-Similarities (LSS) [37] proposes a technique to match different image modalities by measuring internal self-similarities over the regions determined by a log-polar grid. Winder and Brown [46] study many pooling configurations within a framework similar to Daisy and find log-polar to be optimal among their choices. Several binary descriptors, such as BRISK [21] or FREAK [2], rely on sampling patterns over similarly-defined grids to compute intensity differences and extract the features.

Note that all of these methods define polar or log-polar regions for feature *pooling*, that is, the pixel-wise features are always computed in cartesian space, and it is only their aggregation that takes place in log-polar space. As shown in Fig. 1, this is drastically different to our approach, which consists in warping the raw pixel data and use that representation to learn scale-invariant models.

3. Method

First, we describe our sampling scheme in Section 3.1 and, then, our network architecture and training strategy in Section 3.2. For the purposes of this section, we assume that the training data consists of pairs of keypoints across two images that are in correspondence in terms of location and orientation, but not necessarily scale. The actual procedure used to generate the training data is described in Section 4.1.

3.1. Log-Polar Sampling

As in most papers about learning descriptors [12, 50, 38, 4, 41, 28], we use SIFT keypoints [23]. Given an image $\mathbf{I}$ of size $H \times W$, a keypoint $\mathbf{p}_i$ on $\mathbf{I}$ is fully described by its center coordinates (x_i, y_i), its scale $\sigma_i \in \mathbb{R}_+$, and its orientation $\theta_i \in [0, 2\pi)$. We use a Polar Transformer Network (PTN) [11] to extract a $L \times L$ patch around keypoint $\mathbf{p}_i$. To this end, we rely on the following coordinate transform:

$$\begin{aligned} x_i^s &= x_i + e^{log(r_i) x_i^t / W} cos\left(\varphi_i\right) \ , \\ y_i^s &= y_i + e^{log(r_i) x_i^t / W} sin\left(\varphi_i\right) \ . \end{aligned} \tag{1}$$

Variables (x_i^s, y_i^s) denote source coordinates and (x_i^t, y_i^t) target coordinates, after the transform. The coordinate origin is centered on (x_i, y_i), the angle is $\varphi_i = \frac{\theta_i + 2\pi y_i^t}{H}$, and the radius r_i is given by $\frac{\lambda}{2}\sigma_i$, where λ is a factor that converts the SIFT scale to image pixels[1]. Finally, we construct the warped patches by looking up the intensity values in image $\mathbf{I}$ at coordinates (x_i^t, y_i^t) with bilinear interpolation, as done in [11]. This process is illustrated in Fig. 1.

We denote patches extracted in this way as **LogPol**. For comparison purposes, we also consider the standard cartesian approach, using Spatial Transformer Networks (STN) [17] on a regularly spaced sampling grid, defined by

$$\begin{aligned} x_i^t &= x_i + x_i^s cos\left(\theta_i\right) \sigma_i / W - y_i^s sin\left(\theta_i\right) \sigma_i / H \ , \\ y_i^t &= y_i + x_i^s sin\left(\theta_i\right) \sigma_i / W + y_i^s sin\left(\theta_i\right) \sigma_i / H \ . \end{aligned} \tag{2}$$

[1]Given the convention followed by OpenCV, $\lambda = 12$ denotes the scale multiplier of SIFT. One can extract larger image regions setting $\lambda > 12$.

(a) (b) (c) (d)

Figure 2: **Cartesian vs Log-Polar.** (a,c) Two images taken from different viewpoints with four pairs of corresponding keypoints, denoted by their color. (b,d) Patches around these keypoints extracted with their estimated scale and orientation, with λ=16, similarly color-coded. On each column, we show cartesian patches on the left and log-polar patches on the right. While cartesian patches can look very different, log-polar ones remain similar. This is particularly visible for the red keypoint, whose scale estimates are very different in the two images.

We denote these patches as **Cart**. Note that STN and PTN were designed to facilitate whole-image classification by allowing deep networks to manipulate data spatially, thus removing the burden of learning spatial invariance from the classifier. This is not applicable here: we only use their respective samplers, which allow us to efficiently sample the images with in-line data augmentation at a negligible computational cost by applying small perturbations when extracting the patches.

The following properties of log-polar patches distinguish them from cartesian ones:

- Rotations in cartesian space correspond to shifts on the polar axis in log-polar space (rotation equivariance).
- Points close to the origin of the log-polar representation are oversampled, which helps discriminate between neighbouring keypoints.
- Peripheral regions are undersampled, which means that paired patches look similar to the eye even under drastic scale changes (scale equivariance).

This phenomena is illustrated in Fig. 2. Note how the log-polar representation facilitates visual matching even when scales are mismatched. Our approach is predicated on leveraging this information effectively with the deep networks and training framework introduced in the next section.

3.2. Network Architecture and Training

To extract patch descriptors, we use a HardNet [28] architecture. As shown in Fig. 3, our network has seven convolutional layers and takes as input grayscale patches of size 32×32. Input patches are pre-processed with Instance Normalization [44]. Feature maps are zero-padded after each convolutional layer, and we use strided convolutions instead of pooling layers. Each convolution is followed by a ReLU and Batch Normalization, but the last convolution layer omits the ReLU. We apply dropout regularization with a rate of 0.1 after the last ReLU. The final convolutional layer is followed by Batch Normalization and l_2 normalization. The output of the network is a descriptor of unit length and size 128. We found this to be a good compromise between descriptor size and performance.

The standard way to train such networks is in a siamese configuration, with two copies of the network, sharing weights. Among the many loss formulations that have have been proposed [38, 4, 19, 15], we use the triplet loss of [4], as in [28]. To build the required triplets, we consider a collection of patch pairs $\{\mathbf{P}^a_k, \mathbf{P}^b_k\}$ which contain two different views of a 3D point, where $k = 1 \ldots K$, with K denoting the batch size. We systematically check that the 3D points in a given batch are unique, so that $\mathbf{P}^a_i$ and $\mathbf{P}^b_j$ only correspond if $i = j$. We denote their respective descriptors as $\{\mathbf{f}^a_k, \mathbf{f}^b_k\}$. We then mine negative samples with the 'hardest-in-batch' procedure of [28]. Specifically, we build a pairwise distance matrix $\mathbf{D}_{i,j} = d(\mathbf{f}^a_i, \mathbf{f}^b_j)$, $i, j \in [1, K]$, where $d(\mathbf{f}^a_i, \mathbf{f}^b_j)$ is the Euclidean distance between descriptors $\mathbf{f}^a_i$ and $\mathbf{f}^b_j$ if $i \neq j$, and an arbitrarily large value otherwise. We denote the hardest negative sample for $\mathbf{P}^a_k$, *i.e.*, the one with the smallest distance, as $\mathbf{P}^b_{k_{min}}$, and the hardest negative sample for $\mathbf{P}^b_k$ as $\mathbf{P}^a_{k_{min}}$. We consider both $\mathbf{P}^a_k$ and $\mathbf{P}^b_k$ as possible anchors, for all k. Denoting a triplet with anchor (A), positive $(+)$ and negative $(-)$ patches as $(A, +, -)$, we form triplet k taking the hardest negative example, *i.e.* $\{\mathbf{P}^a_k, \mathbf{P}^b_k, \mathbf{P}^b_{k_{min}}\}$ if $d(\mathbf{P}^a_k, \mathbf{P}^b_{k_{min}}) < d(\mathbf{P}^b_k, \mathbf{P}^a_{k_{min}})$ and $\{\mathbf{P}^b_k, \mathbf{P}^a_k, \mathbf{P}^a_{k_{min}}\}$ otherwise. We then take the loss to be

$$\mathcal{L}(\mathbf{f}^A, \mathbf{f}^+, \mathbf{f}^-) = \sum_{k=1}^{K} \max\left(0, 1 + |\mathbf{f}^A_k - \mathbf{f}^+_k|^2 - |\mathbf{f}^A_k - \mathbf{f}^-_k|^2\right).$$

We set the batch size K to 1000. For optimization we use Stochastic Gradient Descent (SGD) with a learning rate of 10, momentum of 0.9, weight decay 10^{-4}, and decay the learning rate linearly to zero within a set number of training epochs [28]. Sampling the patches in-line allows us to apply data augmentation at training time, jittering the orientation of each anchor keypoint by $\Delta\theta \sim \mathcal{N}(0, 25)$ degrees. Our

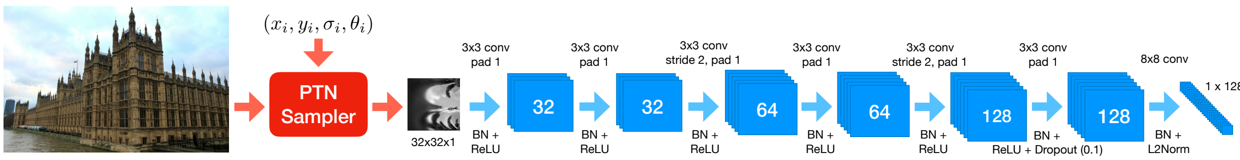


Figure 3: **Network architecture.** We extract patches size 32×32 in-line with a sampler (pictured: PTN) on the desired keypoints. This enables data augmentation techniques. The patches are then given to a network which produces descriptors size 128.

implementation uses Pytorch as a back-end. Code, models and training data are all available.[2]

4. Experiments

In Section 4.1, we introduce the dataset we built to train scale-invariant descriptors, because there is currently none available for this purpose. We then compare ourselves to the state of the art on it. In Sections 4.2, 4.3 and 4.4, we benchmark our models on three publicly available datasets: HPatches [3], AMOS patches [30], and the CVPR'19 Photo Tourism image matching challenge [1]. As baselines, we consider: SIFT [23], TFeat [4], L2-Net [41], HardNet [28], and GeoDesc [25].[3] For our own method we consider descriptors learned with either cartesian or log-polar patches.

4.1. Results on the New Dataset

Nearly all learned descriptors rely on the dataset of [7] for training, which provides patches extracted over different viewpoints for three different scenes. Correspondences were established from SfM reconstructions and SIFT. They are thus biased towards keypoints that can be matched with SIFT, *i.e*., commensurate in terms of scale. In order to learn scale-invariant descriptors under real-world conditions, we require patches extracted at non-corresponding scales, for which we need the original images, which are not provided by [7]. Other datasets, such as [27] or [3], provide images along with homographies for correspondence, but focus on affine transformations and are much too small to train deep networks effectively. Therefore we collected a new dataset for training purposes. In the remainder of this section, we detail how we created it and then report our results on it.

4.1.1 Creating the Dataset

We applied COLMAP [34], a state-of-the-art SfM framework, over large collections of photo-tourism images originally collected by [16]. These images show drastic changes in terms of viewpoint, illumination, and other imaging properties, which is crucial to learn invariance [48]. In addition to sparse reconstructions, COLMAP provides dense correspondences in the form of depth maps. We used them to generate training data by randomly selecting a pair of images $\mathbf{I}_i$ and $\mathbf{I}_j$, extracting SIFT keypoints for both, and using the depth maps to build ground truth correspondences. To do this we projected each keypoint from one image to the other using the estimated poses and depth maps. We took a correspondence (m, n) to be valid if the projection of keypoint m in image i falls within 1.5 pixels of keypoint n in image j. We performed a bijective check to ensure one-to-one correspondences. We applied this projection in a cycle, from i to j and back to i, to ensure that the depth estimates are consistent across both views, and discarded the putative correspondence otherwise. Points which fall in occluded areas were likewise discarded. Note that we only check for corresponding *locations*, but not scales: in this manner we are collecting SIFT keypoints with non-matching scales whose distribution comes from real-world data.

We also require the orientations to be compatible across views. To guarantee this we use the ground truth camera poses to compute the difference between orientation estimates and filter out keypoint matches outside 25^o, as in [7]. Finally, we suppress pairs of keypoints closer than 7 pixels to each other, to exclude patches with large overlaps, which would be particularly problematic for cartesian patches.

We can similarly use the ground truth to warp the scale across images, which we do in order to estimate the frequency of inaccurate scale estimates. Given a correspondence (m, n) comprised of two keypoints with scales (s_i^m, s_j^n), we warp the scale from image i to image j to obtain $\hat{s}_i^m$, and compute the scale difference ratio as $r = \frac{max(\hat{s}_i^m, s_j^n)}{min(\hat{s}_i^m, s_j^n)}$, so that $r \geq 1$, with 1 encoding perfect scale correspondence. We histogram this ratio and use it to evaluate each method under scale changes, as depicted in Fig. 4.

We select 11 sequences for training, and 9 for testing. Please refer to the supplementary material for details. We split the training sequences into training and validation sets in a per-image basis, with a 70:30 ratio. Images are downsampled to a maximum height or width of 1024 pixels, which is the resolution that we extract keypoints at, and mirror-padded to 1500×1500 to alleviate boundary effects. To obtain patches in cartesian space, we sample the image at the desired keypoints with STN. For log-polar patches we use PTN over a support region commensurate with STN; see Section 3.1. We also consider larger patches, increasing λ. We generate up to 1000 correspondences for each image pair, and extract the patches from the images on the fly.

Training requires negative samples, that is, points *not* in correspondence. Finding negatives is easy when a SfM reconstruction is available, as done in [7], ensuring that key-

[2] https://github.com/cvlab-epfl/log-polar-descriptors

[3] We use OpenCV for SIFT, and public implementations for the rest.

Sequence	SIFT	TFeat	L2-Net	Geodesc	HardNet	Ours ($\lambda = 12$) Cart	Ours ($\lambda = 12$) LogPol	Ours ($\lambda = 96$) LogPol
‘british_museum’	5.91	3.53	3.52	4.30	3.21	2.17	2.18	**0.96**
‘florence_cathedral_side’	4.36	1.30	0.51	2.13	0.40	0.23	0.23	**0.20**
‘lincoln_memorial_statue’	2.89	4.32	2.28	2.61	1.65	1.30	1.31	**0.91**
‘milan_cathedral’	7.08	1.98	1.48	1.86	0.35	0.19	0.12	**0.07**
‘mount_rushmore’	18.71	11.94	2.52	2.27	0.43	0.42	0.32	**0.22**
‘reichstag’	2.22	0.44	0.30	0.42	0.21	0.19	0.19	**0.09**
‘sagrada_familia’	9.01	2.41	0.85	1.08	0.27	0.21	0.19	**0.03**
‘st_pauls_cathedral’	8.64	2.01	1.48	2.45	0.68	0.42	0.46	**0.20**
‘united_states_capitol’	8.67	3.90	2.64	5.43	1.60	1.33	0.98	**0.53**
Average	7.50	3.54	1.73	2.51	0.98	0.72	0.67	**0.36**

Table 1: **FPR95 on our new dataset.** We benchmark our models against the baselines with patches extracted at the SIFT scale, $\lambda = 12$. We also show that log-polar models are able to leverage much larger support regions, using $\lambda = 96$. By contrast, with cartesian patches performance degrades as we increase the support region, as we demonstrate in the ablation study of Table 2.

points are stable across all views. This not feasible in our case. Instead, we generate training samples from a single image pair at a time. Specifically, we take one image pair from each of the 11 sequences and use it to fill roughly $1/11$th of each training batch. We can then perform negative mining over the entire batch, as outlined in Section 3.2.

4.1.2 Patch Correspondence Verification

In this section we evaluate performance in terms of patch matching over the test sequences. We extract descriptors for SIFT keypoints with corresponding locations, but using their original scales, which are not always in correspondence. We train our networks with cartesian and log-polar patches, keeping all other settings identical. We use the standard metric in patch matching benchmarks, FPR95, *i.e.*, the False Positive Rate at 95% True Positive recall. For the baselines, we extract patches at the SIFT scale, *i.e.*, $\lambda = 12$. We also consider $\lambda > 12$ for log-polar patches. We report the results in Table 1 and discuss them below.

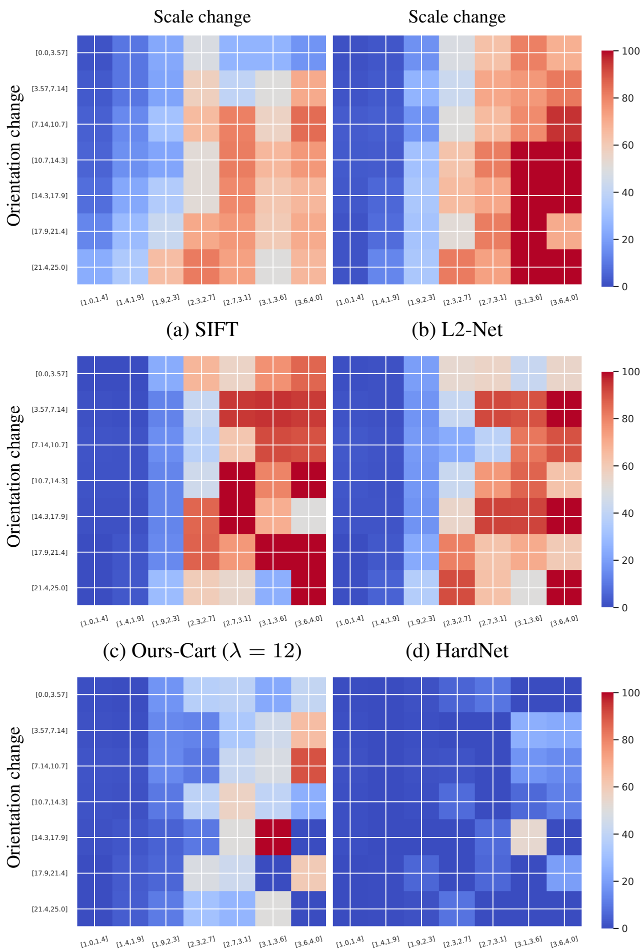
Comparison to the state of the art. Our models trained with log-polar patches deliver the best performance on each sequence, followed by our models trained on cartesian patches, and then HardNet. Remarkably, we achieve our best results with $\lambda = 96$, which corresponds to patches much larger than those best-suited for traditional descriptors, extracted with $\lambda = 12$, a fact that we will examine more closely below. Note the small gap between HardNet and Ours-Cartesian, which is due to the innate differences between datasets and training the latter with mismatched scales. The other baselines perform significantly worse.

Performance under large scale mismatches. In Fig. 4 we break down the results of Table 1 in terms of orientation and scale mismatches. Note how models trained on log-polar representations can tolerate a wide range of scale mismatches. Our results show a negligible drop in performance under scale changes up to 2-3x, and remain useful even at 3-4x. All baselines degrade significantly under scale changes of 2x and become essentially useless beyond that. Note that this invariance is made possible by leveraging log-polar representations and cannot be achieved by simply exposing the models to cartesian patches exhibiting scale changes, as evidenced by the performance of Ours-Cartesian shown in Fig. 4-(c). Finally, remember that this data has been collected from real-world settings with unreliable scale detection. In other words, our models allow us to retrieve *more correspondences* without changing the detector.

Increasing the size of the support region. As shown in Fig. 2, patches extracted with log-polar sampling are remarkably similar across different scales, because scale changes correspond to shifts in the horizontal dimension. This representation is not only easier to interpret visually, but also easier to learn invariant models with. Moreover, oversampling the immediate neighbourhood of the point allows us to leverage larger support regions, because the effect of occlusions and background motion in log-polar patches is smaller than in their cartesian counterparts. We demonstrate this by training models for different values of λ, and report the results in Table 2. Our models are able to exploit support regions much larger than cartesian-based approaches. We see performance flatten out at $\lambda = 96$, and observe boundary issues beyond that point, so we use this value for all experiments in the paper. Note how the radius of the circle determining the support region is *8 times larger* than the optimal value for cartesian patches, and its area *64 times larger*. Note that we use an identical architecture, which can only leverage this information effectively thanks to the log-polar representation.

4.1.3 Image-level Patch Retrieval

Next, we evaluate our performance in terms of patch retrieval. For every image pair in the test sequence, we ex-



(a) SIFT (b) L2-Net

(c) Ours-Cart ($\lambda = 12$) (d) HardNet

(e) Ours-LogPol ($\lambda = 12$) (f) Ours-LogPol ($\lambda = 96$)

Figure 4: **FPR95 vs Scale and orientation changes.** We break down the results of Table 1, histogramming them by the error in the keypoint detection stage. Orientation misdetections increase top to bottom, up to 25^o. Scale misdetections increase left to right, up to 4x. (a,b,d) All baselines degrade quickly under scale changes. (c) Training deep networks with cartesian patches with scale changes is not sufficient. (e,f) By contrast, our log-polar representation enables them to learn scale invariance. Note that some bins are sparsely populated, which explains sudden discontinuities.

tract SIFT keypoints on each image and establish ground truth correspondences using the procedure outlined in Section 4.1. Matches with a difference of up to 25 degrees in orientation are considered positives. Typically, a large percentage of the image pixels are occluded, so that it is not possible to generate a large number of matches. Instead, for every pair of images, we extract up to $N_m = 500$ matches and then generate $N_d = 3000$ distractors, defined as keypoints further than 3 pixels away from a keypoint. The task is thus to find the needle in the haystack, where every keypoint has one positive match and $N_m + N_d - 1$ negatives. We compute the distance between descriptors, extract the rank of each match, and accumulate it over all keypoints and images pairs. The results are summarized in Fig. 5. Our models with log-polar patches obtain the best results, retrieving the correct match about 97% of the time for our best model, for $\lambda = 96$. They are followed by our models with cartesian patches, and HardNet. Notice that contrary to the previous experiment, we evaluate on a realistic patch retrieval scenario with a large number of distractors, which indicates that our performance holds even when sampling keypoints densely, and that it does so regardless of λ.

λ	12	16	32	64	96	128
Ours, Cart	**0.72**	0.77	1.36	4.79	7.03	8.43
Ours, LogPol	0.67	0.61	0.47	0.40	**0.36**	**0.36**

Table 2: **FPR95 vs λ.** We evaluate models trained with differently-sized support regions. Performance increases with λ for log-polar patches, but quickly degrades for cartesian ones.

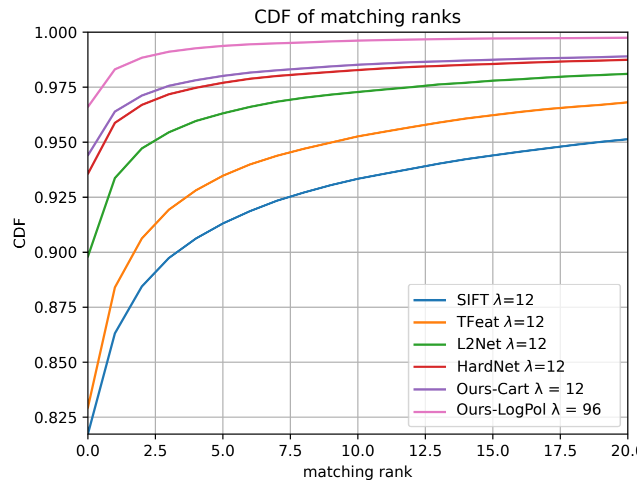


Figure 5: **Patch retrieval on the new dataset.** We plot the cumulative distribution function of the rank in a patch retrieval scenario with a large number of distractors. Our models outperform all the baselines. Log-polar models (**pink**) are significantly better than cartesian ones (**purple**) and baselines based on cartesian patches, such as HardNet (**red**).

4.2. Results on HPatches

The HPatches dataset [3] contains 116 sequences with 6 images each, with either viewpoint or illumination changes. As in [7], HPatches provides pre-extracted patches sampled at corresponding scales, which are not useful for our purposes. However, it also provides the original images and ground truth homographies. We thus define the following protocol. We use SIFT to find keypoints and determine correspondences among them using the ground truth homographies. We consider sequences with viewpoint and illumination changes separately. This provides us with 20733 correspondences for the illumination split and 22079 correspon-

Method	Viewpoint split	Illumination split
SIFT, $\lambda = 12$	0.740	0.607
HardNet, $\lambda = 12$	0.813	0.707
GeoDesc, $\lambda = 12$	**0.879**	0.727
Ours-Cart, $\lambda = 12$	0.828	0.722
Ours-Cart, $\lambda = 16$	0.831	0.732
Ours-Cart, $\lambda = 32$	0.825	0.736
Ours-Cart, $\lambda = 64$	0.752	0.666
Ours-Cart, $\lambda = 96$	0.681	0.616
Ours, LogPol, $\lambda = 12$	0.833	0.729
Ours, LogPol, $\lambda = 16$	0.838	0.743
Ours, LogPol, $\lambda = 32$	0.849	0.764
Ours, LogPol, $\lambda = 64$	0.849	0.774
Ours, LogPol, $\lambda = 96$	0.847	**0.774**

Table 3: **Results on HPatches.** Rank-1 performance on the viewpoint and illumination splits of the HPatches dataset. Our log-polar sampling approach performs better on average than all the baselines, and performance increases with λ, until it saturates.

dences for the viewpoint split. For every match, we compute the distance between a pair of corresponding descriptors and also to all the negatives in the dataset, and evaluate our models in terms of the rank-1 metric, *i.e.*, the percentage of samples for which we can retrieve the correct match with rank 1. We show the results in Table 3. As expected, our log-polar models outperform most of the baselines, and perform better as λ increases. For this experiment we use the models trained on our dataset, without fine-tuning.

4.3. Results on AMOS patches

We also consider AMOS patches [30], a dataset released recently featuring pairs of images captured by webcams and carefully curated in order to provide correspondences. We evaluate our method on the training split, which consists of 27 sequences, each with 50 images, and which also provides keypoints with scales and orientations for every image. We use unique matching keypoint pairs across all images, obtaining a split of 13268 unique keypoint pairs. We use the same metric as for HPatches, and summarize the results in Table 4. As before, we do not re-train the models in any way. Again, our models outperform the state of the art and our results improve with the size of the support region, unlike for methods based on cartesian patches.

4.4. Results on the Phototourism Challenge

Patch matching performance does not always translate to upstream applications, as evidenced by [48, 35]. We thus also evaluate our method on the public Phototourism Image Matching challenge [1]. This benchmark features two tracks: stereo and multi-view matching, and evaluates local features in terms of the quality of the *reconstructed poses*.

Method	Rank-1					
λ	6	12	16	32	64	96
SIFT	0.551	0.518	0.516	0.510	0.480	0.436
GeoDesc	0.434	0.396	0.389	0.416	0.438	0.417
HardNet	0.529	0.464	0.450	0.451	0.470	0.449
Ours, Cart	0.554	0.507	0.530	0.549	0.524	0.481
Ours, LogPol	0.607	0.604	0.625	0.641	0.648	**0.651**

Table 4: **Results on AMOS patches.** Rank-1 performance on the AMOS patches dataset. We noticed that for this dataset, extracting descriptors with smaller patches produces better results for most baselines, so we also consider $\lambda = 6$. Our models trained on log-polar patches outperform the state of the art, and performance increases with λ.

Type	Method	Stereo task mAP$^{15^o}$	Stereo task Rank†	Multi-view task mAP$^{15^o}$	Multi-view task Rank†
DoG	SIFT (IJCV'04)	0.0277	9	0.4146	8
	TFeat (BMVC'16)	0.0357	8	0.4643	7
	L2-Net (CVPR'17)	0.0400	6	0.5087	5
	HardNet (NIPS'17)	0.0425	4	**0.5481**	1
	GeoDesc (ECCV'18)	0.0368	7	0.5298	4
	ContextDesc (CVPR'19)	0.0439	3	0.5399	3
e2e	SuperPoint (CVPR'18)	0.0415	5	0.4778	6
	D2-Net (CVPR'19)	**0.0490**	1	0.3967	9
Ours (DoG)	Ours-Cartesian, $\lambda = 16$	0.0405	—	0.5208	—
	Ours-LogPol, $\lambda = 32$	0.0420	—	0.5389	—
	Ours-LogPol, $\lambda = 64$	0.0432	—	0.5396	—
	Ours-LogPol, $\lambda = 96$	**0.0448**	2	**0.5427**	2

Table 5: **PhotoTourism challenge.** Mean average precision in pose estimation with an error threshold of 15^o. Top method (†among comparable submissions) in **red**, runner-up in **green**. We rank 2nd on both tracks, and 1st on average.

Features are submitted to the organizers, who compute the results. We provide them in Table 5, including comparable baselines (up to 8k features per image, matched by brute-force nearest-neighbour) extracted from the public leaderboards. Our method ranks second on both tracks, and first in terms of average rank. Note that our observations from Section 4.1.2 carry over – models trained on log-polar patches improve with patch size, and outperform cartesian models.

5. Conclusions and Future Work

We have introduced a novel approach to learn local descriptors that goes beyond the current paradigm, which relies on image measurements sampled in cartesian space. We show that we can learn richer and more scale-invariant representations by coupling log-polar sampling with state-of-the-art deep networks. This allows us to match local descriptors across a wider range of scales, virtually for free.

Our approach could be used to learn invariance to arbitrary scale changes. This can be, however, counterproductive when used alongside SIFT, as the majority of its detections are accurate enough. Instead, we intend to bypass scale detection and learn end-to-end pipelines as in [48, 29].

References

[1] Phototourism Challenge, CVPR 2019 Image Matching Workshop. https://image-matching-workshop.github.io. Accessed August 1, 2019. 5, 8

[2] Alexandre Alahi, Raphael Ortiz, and Pierre Vandergheynst. FREAK: Fast Retina Keypoint. In *CVPR*, 2012. 3

[3] Vassileios Balntas, Karel Lenc, Andrea Vedaldi, and Krystian Mikolajczyk. Hpatches: A Benchmark and Evaluation of Handcrafted and Learned Local Descriptors. In *CVPR*, 2017. 5, 7

[4] Vassileios Balntas, Edgar Riba, Daniel Ponsa, and Krystian Mikolajczyk. Learning Local Feature Descriptors with Triplets and Shallow Convolutional Neural Networks. In *BMVC*, 2016. 2, 3, 4, 5

[5] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. SURF: Speeded Up Robust Features. In *ECCV*, 2006. 1, 2

[6] Serge Belongie, Jitendra Malik, and Jan Puzicha. Shape Matching and Object Recognition Using Shape Contexts. *PAMI*, 24(24):509–522, April 2002. 3

[7] Matthew Brown, Gang Hua, and Simon Winder. Discriminative Learning of Local Image Descriptors. *PAMI*, 2011. 2, 5, 7

[8] Gabriela Csurka and Martin Humenberger. From handcrafted to deep local invariant features. In *arXiv preprint arXiv:1807.10254*, 2018. 2

[9] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superpoint: Self-Supervised Interest Point Detection and Description. *CVPR Workshop on Deep Learning for Visual SLAM*, 2018. 2

[10] Mihai Dusmanu, Ignacio Rocco, Tomas Pajdla, Marc Pollefeys, Josef Sivic, Akihiko Torii, and Torsten Sattler. D2-Net: A Trainable CNN for Joint Detection and Description of Local Features. In *CVPR*, 2019. 1, 2

[11] Carlos Esteves, Christine Allen-Blanchette, Xiaowei Zhou, and Kostas Daniilidis. Polar Transformer Networks. In *ICLR*, 2018. 3

[12] Xufeng Han, Thomas Leung, Yangqing Jia, Rahul Sukthankar, and Alexander C. Berg. MatchNet: Unifying Feature and Metric Learning for Patch-Based Matching. In *CVPR*, 2015. 1, 2, 3

[13] Christopher G. Harris and Mike .J. Stephens. A Combined Corner and Edge Detector. In *Fourth Alvey Vision Conference*, 1988. 2

[14] Tal Hassner, Viki Mayzels, and Lihi Zelnik-Manor. On SIFTs and Their Scales. In *CVPR*, 2012. 2

[15] Kun He, Yan Lu, and Stan Sclaroff. Local descriptors optimized for average precision. In *CVPR*, 2018. 1, 2, 4

[16] Jared Heinly, Johannes L. Schoenberger, Enrique Dunn, and Jan-Michael Frahm. Reconstructing the World in Six Days. In *CVPR*, 2015. 5

[17] Max Jaderberg, Karen Simonyan, Andrew Zisserman, and Koray Kavukcuoglu. Spatial Transformer Networks. In *NIPS*, pages 2017–2025, 2015. 3

[18] Yan Ke and Rahul Sukthankar. PCA-SIFT: A More Distinctive Representation for Local Image Descriptors. In *CVPR*, pages 111–119, 2000. 2

[19] Michel Keller, Zetao Chen, Fabiola Maffra, Patrik Schmuck, and Margarita Chli. Learning Deep Descriptors with Scale-Aware Triplet Networks. In *CVPR*, 2018. 1, 2, 4

[20] Iasonas Kokkinos, Michael Bronstein, and Alan Yuille. Dense Scale Invariant Descriptors for Images and Surfaces. Technical report, INRIA, 2012. 2

[21] Stefan Leutenegger, Margarita Chli, and Roland Siegwart. BRISK: Binary Robust Invariant Scalable Keypoints. In *ICCV*, 2011. 3

[22] Ce Liu, Jenny Yuen, and Antonio Torralba. SIFT Flow: Dense Correspondence Across Scenes and Its Applications. In *ECCV*, 2008. 2

[23] David Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *IJCV*, 20(2):91–110, Nov 2004. 1, 2, 3, 5

[24] Zixin Luo, Tianwei Shen, Lei Zhou, Jiahui Zhang, Yao Yao, Shiwei Li, Tian Fang, and Long Quan. ContextDesc: Local Descriptor Augmentation with Cross-Modality Context. In *CVPR*, 2019. 1, 2

[25] Zixin Luo, Tianwei Shen, Lei Zhou, Siyu Zhu, Runze Zhang, Yao Yao, Tian Fang, and Long Quan. Geodesc: Learning Local Descriptors by Integrating Geometry Constraints. In *ECCV*, 2018. 1, 2, 5

[26] Krystian Mikolajczyk and Cordelia Schmid. A Performance Evaluation of Local Descriptors. *PAMI*, 27(10):1615–1630, 2004. 3

[27] Krystian Mikolajczyk, Tinne Tuytelaars, Cordelia Schmid, Andrew Zisserman, Jiri Matas, Frederik Schaffalitzky, Timor Kadir, and Luc Van Gool. A Comparison of Affine Region Detectors. *IJCV*, 65(1/2):43–72, 2005. 5

[28] Anastasiia Mishchuk, Dmytro Mishkin, Filip Radenovic, and Jiri Matas. Working Hard to Know Your Neighbor's Margins: Local Descriptor Learning Loss. In *NIPS*, 2017. 1, 2, 3, 4, 5

[29] Yuki Ono, Eduard Trulls, Pascal Fua, and Kwang Moo Yi. Lf-Net: Learning Local Features from Images. In *NIPS*, 2018. 2, 8

[30] Milan Pultar, Dmytro Mishkin, and Jiri Matas. Leveraging outdoor webcams for local descriptor learning. In *Computer Vision Winter Workshop*, 2019. 5, 8

[31] Jerome Revaud, Philippe Weinzaepfel, César De Souza, Noe Pion, Gabriela Csurka, Yohann Cabon, and Martin Humenberger. R2D2: Repeatable and Reliable Detector and Descriptor. In *arXiv Preprint*, 2019. 1, 2

[32] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. ORB: An Efficient Alternative to SIFT or SURF. In *ICCV*, 2011. 2

[33] Torsten Sattler, Will Maddern, Carl Toft, Akihiko Torii, Lars Hammarstrand, Erik Stenborg, Daniel Safari, Masatoshi Okutomi, Marc Pollefeys, Josef Sivic, Fredrik Kahl, and Tomas Pajdla. Benchmarking 6DOF Outdoor Visual Localization in Changing Conditions. In *CVPR*, 2018. 1

[34] Johannes L. Schönberger and Jan-Michael Frahm. Structure-From-Motion Revisited. In *CVPR*, 2016. 5

[35] Johannes L. Schönberger, Hans Hardmeier, Torsten Sattler, and Marc Pollefeys. Comparative Evaluation of Hand-Crafted and Learned Local Features. In *CVPR*, 2017. 8

[36] Qi Shan, Changchang Wu, Brian Curless, Yasutaka Furukawa, Carlos Hernandez, and Steven M. Seitz. Accurate Geo-registration by Ground-to-Aerial Image Matching. In *3DV*, 2014. 2

[37] Eli Shechtman and Michal Irani. Matching Local Self-Similarities Across Images and Videos. *CVPR*, 2007. 3

[38] Edgar Simo-Serra, Eduard Trulls, Luis Ferraz, Iasonas Kokkinos, Pascal Fua, and Francesc Moreno-noguer. Discriminative Learning of Deep Convolutional Feature Point Descriptors. In *ICCV*, 2015. 1, 2, 3, 4

[39] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Learning Local Feature Descriptors Using Convex Optimisation. *PAMI*, 2014. 1, 2

[40] Christoph Strecha, Alex Bronstein, Michael Bronstein, and Pascal Fua. LDAHash: Improved Matching with Smaller Descriptors. *PAMI*, 34(1), January 2012. 2

[41] Yurun Tian, Bin Fan, and Fuchao Wu. L2-Net: Deep Learning of Discriminative Patch Descriptor in Euclidean Space. In *CVPR*, 2017. 1, 2, 3, 5

[42] Engin Tola, Vincent Lepetit, and Pascal Fua. A Fast Local Descriptor for Dense Matching. In *CVPR*, 2008. 1, 3

[43] Eduard Trulls, Iasonas Kokkinos, Alberto Sanfeliu, and Francesc Moreno-Noguer. Dense Segmentation-Aware Descriptors. *Dense Image Correspondences for Computer Vision*, 2015. 2

[44] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Improved Texture Networks: Maximizing Quality and Diversity in Feed-Forward Stylization and Texture Synthesis. In *CVPR*, 2017. 4

[45] Xing Wei, Yue Zhang, Yihong Gong, and Nanning Zheng. Kernelized Subspace Pooling for Deep Local Descriptors. In *CVPR*, 2018. 1, 2

[46] Simon Winder and Matthew Brown. Learning Local Image Descriptors. In *CVPR*, June 2007. 1, 3

[47] Alessio Xompero, Oswald Lanz, and Andrea Cavallaro. MORB: A Multi-Scale Binary Descriptor. In *ICIP*, 2018. 2

[48] Kwang Moo Yi, Eduard Trulls, Vincent Lepetit, and Pascal Fua. LIFT: Learned Invariant Feature Transform. In *ECCV*, 2016. 2, 5, 8

[49] Kwang Moo Yi, Eduard Trulls, Yuki Ono, Vincent Lepetit, Mathieu Salzmann, and Pascal Fua. Learning to Find Good Correspondences. In *CVPR*, 2018. 1

[50] Sergey Zagoruyko and Nikos Komodakis. Learning to Compare Image Patches via Convolutional Neural Networks. In *CVPR*, 2015. 1, 2, 3

[51] Dang Zheng, Kwang Moo Yi, Yinlin Hu, Fei Wang, Pascal Fua, and Mathieu Salzmann. Eigendecomposition-Free Training of Deep Networks with Zero Eigenvalue-Based Losses. In *ECCV*, 2018. 1

[52] Lei Zhou, Siyu Zhu, Tianwei Shen, Jinglu Wang, Tian Fang, and Long Quan. Progressive Large Scale-Invariant Image Matching In Scale Space. In *ICCV*, 2017. 2