D2D: Learning to find good correspondences for image matching and manipulation

Olivia Wiles  
VGG, University of Oxford  
ow@robots.ox.ac.uk

Sebastien Ehrhardt  
VGG, University of Oxford  
hyenal@robots.ox.ac.uk

Andrew Zisserman  
VGG, University of Oxford  
az@robots.ox.ac.uk

Abstract

We propose a new approach to determining correspondences between image pairs under large changes in illumination, viewpoint, context, and material. While most approaches seek to extract a set of reliably detectable regions in each image which are then compared (sparse-to-sparse) using increasingly complicated or specialised pipelines, we propose a simple approach for matching all points between the images (dense-to-dense) and subsequently selecting the best matches. The two key parts of our approach are: (i) to condition the learned features on both images, and (ii) to learn a distinctiveness score which is used to choose the best matches at test time. We demonstrate that our model can be used to achieve state of the art or competitive results on a wide range of tasks: local matching, camera localization, 3D reconstruction, and image stylization.

Figure 1: Correspondences obtained with the D2D model. These demonstrate the model’s robustness in the face of challenging scene shift: changes in illumination (a,d), viewpoint (a-d), context (a,d), or style (b).

1 Introduction

Determining correspondence between two images of the same scene or object is a fundamental challenge in computer vision, important for many applications ranging from optical flow and image manipulation to 3D reconstruction and camera localisation. This task is challenging due to scene-shift: a scene dramatically changes over time due to variations in illumination (e.g. day to night), viewpoint, texture, and season (e.g. snow in winter versus flowering trees in spring).

Methods that solve the correspondence task typically follow one of two paradigms: sparse-to-sparse or dense-to-dense. Under the sparse-to-sparse paradigm, most methods follow a detect-and-describe approach: first detecting distinctive regions [5, 16, 29, 34, 54] and then extracting descriptors [5, 6, 21, 25, 29, 54] for these regions. These descriptors are then matched between images, often using additional geometric constraints [17]. Recent approaches have sought to learn either or both of these components [3, 8, 9, 41, 58, 63, 64, 75, 77]. These methods typically only find matches at textured locations, and do not find matches over smooth regions of an object. Additionally, finding these repeatable detections with invariance to scene-shift is challenging [2, 55, 60].

Under the dense-to-dense paradigm, methods will typically obtain a dense feature map which is compared between images using, optionally, additional spatial and smoothness constraints [7, 10, 28].
These approaches mostly focus on obtaining optical flow for limited changes in camera within the same video; the scenes have limited scene shift.

We introduce a new approach for obtaining dense correspondences between a pair of images. Previous methods (under both paradigms) learn features for each image without knowledge of the other image. Thus, their intermediary features must be invariant to changes – e.g. to scale changes, shift, etc. However, as features become more and more invariant, they become increasingly ambiguous to match (if a feature was invariant to everything, it would be confused for everything). We forsake this invariance and instead condition the feature maps on both images. This allows the features to be modified based on the differences between the images (e.g. a change in global illumination). Traditionally, this was infeasible, but we can learn such a model efficiently using neural networks.

We additionally regress a distinctiveness score: the likelihood that a point in one image will uniquely and correctly match to a point in the other image. This score is regressed from the conditioned features, so the likelihoods can model the distinctiveness of a point given the other image. At test time the model ingests a pair of images and outputs a set of dense correspondences, together with their distinctiveness scores. The scores can be used to select the best matches.

In summary, we propose a novel approach and network, D2D, a Dense-2-Dense approach for determining correspondences between an image pair. The approach is based on two key insights: (i) conditioning learned features on both images, and (ii) learning a distinctiveness score for each feature. As will be seen, the model is simple and scalable and eschews a number of techniques used by other methods to improve performance: high dimensional descriptors (we use a 64D descriptor, half the size of the current smallest descriptor), and multiple scales (we only operate at a single scale, whereas other methods use multiple scales).

We apply D2D to a variety of tasks: local matching, camera localisation, 3D reconstruction, and style transfer. We achieve state-of-the-art (sota) or comparable on all tasks despite our simple approach.

## 2 Related Work

In this section, we review related work on finding correspondences, broken down into three main paradigms: dense-to-dense, sparse-to-sparse, sparse-and-then-dense.

### Sparse-to-sparse

Traditional 3D reconstruction and camera localisation pipelines require accurate correspondences across multiple images. Traditional methods solved this by using a detect-and-describe approach: first a distinctive region is detected in textured regions \([5, 16, 29, 54]\) and second a hand crafted descriptor is used to describe the region with varying degrees of invariance to scale, illumination, rotation, and general affine transformations \([5, 6, 21, 25, 29, 54]\). In modern approaches, the descriptor step \([3, 13, 63, 64]\), detector step \([58, 77]\), or both \([8, 9, 41, 75]\) are replaced with a neural network. We forego the detection step and condition features on both images.

### Sparse-and-then-dense

Given sparse-to-sparse correspondences, the camera between images can be estimated and a second algorithm, e.g. PatchMatch \([4, 61]\), used to find dense correspondences between the images. If the images have been rectified using multiple-view geometry \([17]\) and have limited scene shift, stereo algorithms such as \([15, 44, 66, 74]\) (and reviewed by \([67]\)) can be used to obtain a dense reconstruction.

### Dense-to-dense

Other methods consider how to find dense correspondences between two images. How to define correspondence differs based on the method.

Initially, finding dense correspondence meant finding the precise pixel correspondence between two images of the same scene. Traditional approaches \([28, 70]\) generalised sparse descriptors. Modern methods learn to match all pixels between frames of a video \([7, 10, 19, 47, 57, 62, 65, 73]\). However, these methods assume photometric consistency between frames: their quality degrades the more the frames differ in time, as the pose and viewpoint progressively change. \([62]\) uses additional semantic information and \([51]\) uses pretrained features to find dense matches between images of the same scene to mitigate this problem.

Later, region flow defined correspondence as a match between a pixel and the corresponding region in the other image (e.g. a point on a dog to the entire dog). This is useful if there is large viewpoint change or non rigid motion \([24, 71]\). Also related are approaches that seek to learn correspondence between similar scenes \([28]\) or instances of the same semantic class \([22, 26, 40, 49, 50, 52]\).
Figure 2: D2D overview for one input image. To obtain descriptor vectors $D^1$ for one input image $I^1$ conditioned on another $I^2$, both images ($I^1, I^2$) are encoded via an attention mechanism. The conditioned features are then decoded to obtain $D^1$. We also regress a distinctiveness mask which is used at test time to ignore unmatchable regions (e.g. the sky or regions visible in only one image). The descriptor vectors $D^2$ for $I^2$ are obtained by swapping the input images.

3 Method

Our task is to find dense correspondences between a pair of images of the same scene. The pipeline has two stages. The first stage obtains dense descriptor vectors for each image and a distinctiveness score. The descriptors are conditioned on both images so they only have to be invariant to the changes particular to that pair of images. The second stage compares these descriptor vectors to obtain a set of high quality matches. We first introduce our pipeline before describing how it is trained.

3.1 D2D Architecture

Our pipeline for obtaining descriptor vectors and a distinctiveness score for one image $I^1$ (Fig. 2), is composed of four components. The first component, the encoder, projects both images $I^1, I^2$ to obtain feature maps at two resolutions: $f^1_L, f^1_S$. The second component, the attention mechanism, is used to determine spatial correspondence between the feature maps of the different images. The third component, the decoder, uses the attention mechanisms to concatenate the feature maps of the two images; these concatenated features are used to obtain a grid of spatial descriptor vectors $D^1$ that are conditioned on both images. The final component, the regressor, learns a distinctiveness score for each grid position which encodes how likely the match is to be accurate. To obtain descriptor vectors $D^2$ for the other image, we operate precisely as described above, except that the order of the input images is flipped. This gives a grid of descriptor vectors $D^1, D^2$ for images $I^1, I^2$ respectively.

Encoder. Given two images of the same scene, $I^1 \in \mathbb{R}^{H \times W \times 3}$ and $I^2 \in \mathbb{R}^{H \times W \times 3}$, we obtain spatial feature maps: $f^1_L$ and $f^1_S$ at a larger and smaller resolution. These will be concatenated within a UNet framework [53] to form the decoder. A CNN with shared parameters is used to encode the images and obtain these spatial feature maps. In practice, we use the feature maps after the last two blocks in a ResNet50 [18] architecture.

Attention Mechanism. We wish to concatenate features from both images in order to condition the model on both input images. However, for a given spatial location, the relevant (corresponding) feature in the other image may not be at the same spatial location. As a result, we use an attention mechanism to model long range dependencies when concatenating features.

An attention mechanism is used to determine where a location $i$ in one set of features $g$ from one image should attend to in another set of features $h$ from another image [71]. For each location $i$ in $g$, it obtains a feature $\hat{g}_i$ that is a weighted sum over all spatial features in $h$ where $A$ is the similarity matrix comparing $g$ and $h$ using the inner product followed by the softmax normalisation step.

$$\hat{g}_i = \sum_j A_{ij} h_j$$

$$A_{ij} = \frac{\exp(g_i^T h_j)}{\sum_k \exp(g_i^T h_k)}$$

(1)
To train these descriptors, we use a standard contrastive hinge loss to separate true and false correspondences (we consider other contrastive losses in the appendix). For the set $P$ of positive correspondences, the correspondence loss is computed as

$$L_c = \frac{1}{L} \sum_{(x,y) \in P} d(D^1_x, D^2_y).$$

To apply this attention mechanism, we operate as follows for $f^1_L$ (and similarly for $f^2_L$). As standard, to perform dimensionality reduction the features are projected (with two MLPs $g^1(\cdot), g^2(\cdot)$) and then normalised: $g = \frac{g^1(f^1_L)}{\|g^1(f^1_L)\|_2}, h = \frac{g^2(f^2_L)}{\|g^2(f^2_L)\|_2}$. The attended features $f^1_L$ are then computed as in (1).

**Decoder: Conditioned Features.** The attended features are incorporated into a UNet architecture in order to obtain a grid of spatial descriptors $D^1 \in \mathbb{R}^{H \times W \times C}$ (see Fig. 2). The attended features $f^1_L$ and $f^2_L$ are concatenated with the original features and passed through the decoder portion of the UNet. The resulting feature map is $L2$ normalised over the channel dimension to obtain the final descriptors. This ensures that the final descriptors are conditioned on both images.

**Regressor: Distinctiveness Score.** We regress a distinctiveness score $r(\cdot)_{ij} \in [0, 1]$, for each pixel $(i, j)$, which approximates its matchability and is used at test time to select the best matches. $r(\cdot)_{ij}$ approximates how often the descriptor at $(i, j)$ is confused with negatives in the other image. If it is near 1, the descriptor is unique; if it is near 0, the descriptor is often confused. To regress these values, we use an MLP, $r(\cdot)$, on top of the unnormalised descriptor feature maps.

**Determining Matches at Test Time.** We want matches at locations $k$ and $l$ in images $I^1$ and $I^2$ respectively that are accurate and distinctive (e.g. no matches in the sky). We use the scalar product to compare the normalised descriptor vectors to find the best matches and the distinctiveness score to determine the most distinctive matches. The following similarity score $c_{kl}$ combines these properties such that a value near 1 indicates a distinct accurate match:

$$c_{kl} = r(D^1_k) r(D^2_l) \left[(D^1_k)^T D^2_l\right].$$

Finally, we select the best $K$ matches. We consider all matches which are mutual nearest neighbours: e.g. if the best match for location $m$ in one image is location $n$ in another, and the best match for location $n$ is $m$, then match $(n, m)$. So if the following holds:

$$m = \arg\max_n c_{nj} \quad \text{and} \quad n = \arg\max_m c_{im}$$

These matches are ranked according to their similarity score and the top $K$ used. To select candidate matches, we compare all normalised descriptor vectors in one image to all in the other.

### 3.2 Training and Loss Functions

**Selecting Correspondences at Train Time.** Given a ground-truth correspondence map, we randomly select $L$ positive correspondences. For each positive correspondence, we randomly select a large number ($K = 512$) of negative correspondences. These randomly chosen positive and negative correspondences are used to compute both the distinctiveness and correspondence losses.

**Correspondence Loss.** The correspondence loss is used to enforce that the normalised descriptor maps $D^1$ and $D^2$ can be compared using the scalar product to obtain the best matches. At a location $i$ in $D^1$ and $j$ in $D^2$ then the standard Euclidean distance metric $d(D^1_i, D^2_j)$ should be near 0 if the corresponding normalised descriptor vectors are a match.

To train these descriptors, we use a standard contrastive hinge loss to separate true and false correspondences (we consider other contrastive losses in the appendix). For the set $P$ of $L$ true pairs, the loss $\mathcal{L}_p$ enforces that the distance between descriptors is near 0. For the set $N$ of $LK$ negative pairs, the loss $\mathcal{L}_n$ enforces that the distance between descriptors should be above a margin $M$.

$$\mathcal{L}_p = \frac{1}{L} \sum_{(x,y) \in P} d(D^1_x, D^2_y), \quad \mathcal{L}_n = \frac{1}{LK} \sum_{(x,y) \in N} \max(0, M + c_x - d(D^1_x, D^2_y)).$$

$c_x = d(D^1_x, D^2_y)$, $(x,y) \in P$ re-weights the distance of the false correspondence according to that of the positive one: the less confident the true match, the further the negative one must be from $M$ [9].

**Distinctiveness Loss.** To learn the $r(\cdot)$ MLP, we need an estimate of how often a descriptor in one image is confused with the wrong descriptors in the other image. Given a set $K$ of $K$ negative matches in the other image and the margin $M$, the number of times a descriptor at location $x$ is confused is $m_x = \sum_{y \in K} \mathbbm{1}[d(D^1_x, D^2_y) < M]$. This value is used to regress $r(\cdot)$ which is near 1 if the feature has a unique match (the true match), near 0 otherwise ($\tau$ is a hyper-parameter set to $\frac{1}{2}$):

$$\mathcal{L}_r = \frac{1}{L} \sum_{(x,:) \in P} \left| r(D^1_x), \frac{1}{1 + m_x} \right|^\tau.$$
Training Setup. Our model is trained on MegaDepth [27], which consists of a variety of landmarks, registered using SfM [59]. As each landmark consists of many images taken under differing conditions, we can obtain matches between images that are unmatchable when considered independently. We train the features end-to-end, but train the distinctiveness score separately by not allowing gradients to flow. In practice we backpropagate on all randomly chosen positive pairs $L_p$, negative pairs $L_n$, and additionally the hardest $H = 3$ negative pairs for each positive pair.

The model is trained with a learning rate of 0.0001, the ADAM optimizer [23], a batch size of 16, $M=1$, and $L=512$. At train time we use an image size of 256, at test time an image size of 512. We use $K=2000$ for HPatches and Aachen, and $K=8192$ when performing SfM. For SfM, we find it is important to use more, rougher correspondences to obtain more coverage in the 3D reconstruction.

4 Experiments

We evaluate the D2D model on three different tasks. The first task directly assesses how well D2D can estimate correspondences between images pairs. The second task uses the D2D correspondences to perform camera localization and obtain high quality 3D reconstructions in challenging situations, with a large amount of scene shift. The final task is stylization, and assesses D2D’s dense matches.

Ablations. The full model uses the ResNet50 backbone, the attention mechanism and the distinctivness score to reweight matches. We ablate multiple variants. The first (ours) is our full model. The second (ours w/o conf) is our model without the distinctiveness score. The third (ours w/o cond) is our model without conditioning (i.e. the attention mechanism). The final (ours-E-B1) is our full model but using an EfficientNet-B1 backbone [68]. This ablation uses a smaller (7M params vs 23M params) and faster (0.7GFlops vs 4.1GFlops) backbone architecture; it is more suitable for practical applications. The appendix includes further ablations to validate our choices (e.g. the loss function and grid size) and datasets (e.g. YFCC100M) which show our superior results.

4.1 Correspondence Evaluation

We test our model on local matching by evaluating on the HPatches [2] benchmark. We compare to a number of baselines and achieve state-of-the-art results.

Hpatches Benchmark. The HPatches benchmark evaluates the ability of a model to find accurate correspondences between pairs of images, related by a homography, that vary in terms of illumination or viewpoint. We follow the standard setup used by D2Net [9] by selecting 108 of the 116 sequences which show 6 images of larger and larger illumination and viewpoint changes. The first image is matched against the other 5, giving 540 pairs.
Evaluation Setup. We follow the evaluation setup of D2Net [9]. For each image pair, we compute the number of correct matches (using the known homography) and report the average number of correct matches as a function of the pixel threshold error in Fig. 3. We then compare to a number of detect-then-describe baselines used in D2Net using their software: RootSIFT [29] with the Affine keypoint detector [33], HesAffNet [36] with HardNet++ [35], LF-Net [41], SuperPoint [8], DELF [39]; as well as to D2Net [9] and R2D2 [48]. These methods vary in terms of whether the detector and descriptors are hand crafted or learned.

Results. As shown in Fig. 3, all variants of our model outperform previous methods for larger pixel thresholds, demonstrating the practicality and robustness of our approach. In comparison to other methods, D2D performs extremely well when the images vary in illumination: it outperforms all other methods. D2D is superior under viewpoint changes for larger pixel thresholds (> 6px). Using the smaller, more efficient (ours-E-B1) actually improves performance over the larger ResNet model (ours). Using a simple refinement strategy (described in the appendix) boosts our model’s performance under viewpoint changes, giving results superior or comparable to sota methods for all thresholds. Compared to the other datasets we evaluate on, e.g. [55] below, the components of our model have a limited impact at improving performance on the HPatches benchmark, presumably because this dataset has less scene shift than the others.

4.2 Using Correspondences for 3D Reconstruction

In these experiments, we evaluate the robustness of our approach on images that vary significantly in terms of illumination and viewpoint, and our model’s ability to scale to larger datasets. D2D achieves sota or comparable results on all datasets.

4.2.1 Camera Localization

Aachen Benchmark. In order to evaluate our approach under large illumination changes, we use the Aachen Day-Night dataset [55][56]. For each of the 98 query night-time images, the goal is to localize the image against a set of day-time images using predicted correspondences.

Evaluation Setup. The evaluation measure is the percentage of night time cameras that are localized within a given error threshold [55]. We use the pipeline and evaluation server of [55] with the matches automatically obtained with our method (Section 3.1). We compare against detect-and-describe methods (RootSIFT descriptors from DoG keypoints [29], HardNet++ with HesAffNet features [35][36], DELF [39], SuperPoint [8], and D2Net [9]) as well as a dense approach (DenseSFM [55]).

Results. Table 1 shows that our method does better than other dense-to-dense approaches and comparably or better than sota detect-and-describe approaches. These results imply that the traditional approach of first finding reliably detectable regions may be unnecessary; using a grid to exhaustively find matches is, perhaps surprisingly, superior in this case. These results also show that our architectural improvements (i.e. using an attention mechanism and distinctiveness score) boost performance and that a more efficient architecture (Ours-E-B1) has only a limited impact on performance.
Table 2: SfM. We compare our approach to using SIFT features on 3D reconstruction for two datasets: Large SfM and Sculpture. Both are collections of images from the web containing large variations in illumination and viewpoint. These metrics are a proxy for 3D reconstruction quality, so we encourage the reader to view the reconstructions in the appendix. X: failure. ↑: higher is better. ↓: lower is better. blue: best result.

| Landmark: | Large SfM | Sculpture Dataset |
|-----------|-----------|-------------------|
| # Images: | 1344 1463 1576 | 12 124 250 266 78 31 358 238 74 |
| # Reg. Ims ↑ | 702 1072 967 | 12 108 194 215 67 25 284 201 57 |
| SIFT [29]: | 500 1035 804 X 103 198 503 267 61 22 266 201 53 |
| Ours: | 116K 338K 239K X 48K 70K 102K 28K 9K 132K 99K 23K |
| # Sparse Pts ↑ | 116K 338K 239K | 6.32 5.52 7.62 |
| SIFT [29]: | 500 1035 804 X 103 198 503 267 61 22 266 201 53 |
| Ours: | 116K 338K 239K | 6.32 5.52 7.62 |
| Track Len ↑ | 4.00 0.60 0.64 | 0.60 0.60 0.64 |
| SIFT [29]: | 4.00 0.60 0.64 | 0.60 0.60 0.64 |
| Ours: | 1.30 1.34 1.32 |
| Reproj Err (px) ↓ | 1.30 1.34 1.32 |
| SIFT [29]: | 1.30 1.34 1.32 |
| Ours: | 1.30 1.34 1.32 |
| # Dense Pts ↑ | 1.8M 4.2M 3.1M | 1.8M 4.2M 3.1M |
| SIFT [29]: | 1.8M 4.2M 3.1M | 0.2K 1.88K 156K |
| Ours: | 1.8M 4.2M 3.1M |

4.2.2 Structure from Motion (SfM)

The objective here is to evaluate the correspondences obtained with our model for the task of 3D reconstruction using a smaller or larger number of views. The assessment is on two datasets: first, the standard SfM benchmark that contains many (∼1500) images of the same scene; second, a challenging scenario where the dataset contains fewer (∼10 – 100) images of the same scene and where the object (a sculpture) may differ in material, location, and context.

Baselines. We compare to SIFT [29]. This method first finds repeatably detectable regions for which features are extracted and compared between images. This method works well when there are distinctive textured regions that can be found and matched. Our method, however, aims to find dense matches, so our approach should be more robust when there are fewer textured regions.

SfM Dataset. In this setup, we evaluate on the Local Feature Evaluation Benchmark [60]. This dataset consists of multiple landmarks for which many internet images have been collected. We use Madrid Metropolis (MM), Gendarmenmarkt (Gen), and Tower of London (Tow). Table 2 demonstrates that, using D2D, we consistently register more images and obtain significantly more sparsely reconstructed 3D points (visualisations are given in the appendix). However, the pixel error is higher and there are fewer dense points. These results imply that our model is more robust for matching and registering challenging images but, presumably due to the grid discretization when creating candidate correspondences, the correspondences are not as pixel accurate, affecting the reprojection error.

Sculpture Dataset. We use images from the Sculpture dataset [12], which consists of images of the same sculpture downloaded from the web. As an artist may create the same sculpture multiple times, a sculpture’s material (e.g. bronze or marble), location, or context (e.g. the season) may change in the images (refer to the appendix for examples). In particular, we evaluate on nine sculptures by the artist Henry Moore. These sets of images contain large variations and the sculpture itself is often smooth, leading to less texture for finding repeatedly detectable regions.

We report the results in Table 2 and visualise samples in the appendix. As can be seen by these results, our approach is able to consistently obtain more 3D points for each image set. Also, in one challenging case D2D succeeds and SIFT fails. These results validate that our dense approach does indeed make our model robust in this context.

4.3 Using Correspondences for Stylization

Previously, we focused on using our matching pipeline for extracting a set of sparse correspondences to be used for localization and 3D reconstruction. Here we evaluate how well our dense features can be used for a task that requires dense matching: stylization. The goal is, given two images $I_s$, $I_v$ of the same scene, to generate an image with the style of $I_s$ but the pose and viewpoint of $I_v$.

Setup. To achieve this, we first use the D2D model to transform $I_s$ into the position of $I_v$. The approach is simple: instead of only choosing the mutual nearest neighbours as in [4], we consider
Figure 5: Stylization. Given $I_s$ and $I_v$, the task is to generate an image with the pose and viewpoint of $I_v$ and style of $I_s$. We show results for D2D and a baseline that uses semantics [30]. We also show the resampled image (GT$_S$) which is computed using the true correspondences from the MegaDepth dataset [27]. While [30] works well for easy cases, it sometimes copies style from $I_v$ (as shown by the red background in (a), unlit building in (b), and red hued building in (g)). [30] also fails if the semantic prediction is incorrect (e.g. (c) and (e)).

The best match for every pixel location. We then use the color of the best match in $I_s$ to color the corresponding location in $I_v$. This gives us the sampled image. The next step is to remove artefacts. We do this by training a refinement model on top of the sampled image in order to obtain an image $I_g$ in the pose of $I_v$ and style of $I_s$. Full details of the architecture and training are given in the appendix material.

Comparison to Pix2Pix. In a standard image to image translation task (e.g. Pix2Pix [20]), the two images (e.g. image and semantic map) are aligned. In our case, the images are not aligned. We effectively use our correspondences to align the images and then run a variant of Pix2Pix.

Experimental Setup. To evaluate our results, we use the test set of the MegaDepth dataset (these are landmarks unseen at training time). We randomly select 400 pairs of images and designate one the viewpoint $I_v$ image and the other the style image $I_s$. We task the models to generate a new image $I_g$ with the style of $I_s$ in the viewpoint of $I_v$. From the MegaDepth dataset, we can obtain ground truth correspondence for regions in both images and so the true values of $I_g$ for this region. The reported error metric is the mean $L_1$ distance between the generated image and true value within this region.

Results. We compare against a stylization approach that uses semantics to perform style transfer [30] in Fig. 5. We also determine the $L_1$ error for both setups and obtain 0.22 for [30] and 0.14 for our method, demonstrating that our method is more accurate for regions that can be put in correspondence. The qualitative results demonstrate that our method is more robust, as [30] produces poor results if the semantic prediction is wrong and sometimes copies style from $I_v$, as opposed to $I_s$ (e.g. it creates a colored $I_g$ image when $I_s$ is grey-scale). As we sample from $I_s$ in the first step and then refine the sampled image, our model rarely copies style from $I_v$. Finally, our full method runs in seconds at test time whereas [30] takes minutes due to a computationally intensive iterative refinement strategy.

5 Conclusion

We introduced a new approach for obtaining correspondences for image pairs. We obtain sota or comparable results on a range of tasks, yet using a smaller network. We achieve these results using a simple strategy of exhaustively comparing features and ranking matches by a distinctiveness score.

Our work has demonstrated that with modern networks, the traditional detect-and-describe approach to remove distracting regions is perhaps no longer necessary. Comparable or better performance can be obtained by a simple exhaustive match over image features. This shows there is the potential for a rethink of SfM pipelines. Instead of using trajectories of points over many images, one could use denser, rougher correspondences between pairs of images. However, we leave this for future work.
A Overview

We include additional implementation details and results in Section B. These experiments provide results on an additional dataset, demonstrate how our approach can use a refinement step to achieve state-of-the-art results for all thresholds >1px on HPatches, and provide additional ablations including the use of a Noise Contrastive (NCE) loss as opposed to the hinge loss presented in the paper. We define the metrics used in obtaining the SfM results in Section B.3 and visualise our reconstructed 3D models on the large dataset in the video.

Additionally, we provide further details of the architectures used in Section C for the main model and the stylisation model.

We also provide qualitative samples of the confidence maps learned by our model in Section D.1. We provide additional qualitative results for the stylisation experiment, HPatches dataset, SfM experiment, and Aachen dataset in Section D.2.

B Additional Experiments

In this section we report the results of additional experiments which consider additional ablations of the grid size chosen at test time (Section B.1), a refinement of our model to improve local matching (Section B.2) to achieve state-of-the-art results on HPatches, further comparisons of our model on the 3D reconstruction task (Section B.4), results on the YFCC100M dataset (Section B.5), and results using another popular contrastive loss (NCE) (Section B.6).

B.1 Further Ablations

In this section we discuss and ablate how we select candidate matches at test time.

In order to compare all descriptor vectors at test time, we operate as follows. We create a $G \times G$ pixel grid and bilinearly interpolate on this grid from both the descriptor maps and distinctiveness scores. (Note that we have to normalise the interpolated descriptors.) We consider all pairs of descriptors as candidate matches and compare all descriptors in one image to those in the other. In practice we use $G = 128$.

At test time, we could use a larger grid size for better granularity. However, this comes with the additional computational cost of performing $G^4$ comparisons. We tried using a larger grid ($G = 256$) on the Aachen-Day Night dataset in Table 5 but obtained comparable results to using $G = 128$. As a result, we continued to use $G = 128$ for all our experiments. Using a larger grid for the larger datasets in SfM (where the number of images are approximately 1500) would have made these experiments intractable using current pipelines.

However, we note that because we only consider points on a grid, we are losing some granularity which presumably impacts the performance on the 3D reconstruction tasks. We discuss a method to refine the correspondences to obtain a finer granularity in the next section and the resulting complications.

B.2 Refining Local Matching

In this section, we demonstrate that our local matching results can be improved by using a simple refinement strategy.

Description of Refinement Strategy. In order to refine the matches obtained using D2D, we use a local neighbourhood to refine the match in the second image. Given a correspondence with location $(x_1, y_1)$ in the first image and $(x_2, x_2)$ in the second image, we look at the similarity score between $(x_1, y_1)$ and the locations in a 3x3 grid centered on $(x_2, y_2)$.

These scores are used to reweight the location in the second image using a normalised weighted sum with one difference. Because the similarity scores are not evenly distribute and cluster around 0.5, they give too much weight to less likely matches. As a result, we subtract the minimum similarity from all scores in a local neighbourhood before computing the normalised weighted sum.
In this experiment, we look at the results of using a refinement strategy based on a local neighbourhood to improve the accuracy of the detected correspondences. We see that using the refinement scheme, we obtain a big boost in performance. In particular, we improve our results for fine-grained pixel thresholds on the viewpoint task. We achieve comparable results with state-of-the-art methods for small pixel thresholds and better performance for larger thresholds (> 4px). On illumination we maintain performance except for very small thresholds (≤ 1px). This small degradation is probably due to limited noise that is introduced with the refinement strategy. In general, using this strategy we improve our results to achieve state of the art performance for all pixel thresholds.

Results. The results are given in Fig. 6. As can be seen, this simple refinement scheme gives a large boost in performance. In particular, using this simple modification gives superior results to state of the art approaches for pixel thresholds > 4px for viewpoint and all thresholds > 2px for illumination. Overall, our model with the refinement scheme achieves state-of-the-art results for all thresholds > 1px.

Discussion. While we could achieve high quality performance on HPatches using this simple modification, we note that it is not straightforward to apply this to the camera localisation or 3D reconstruction tasks. This is because both of these tasks require a single point to be tracked over multiple images in order to perform 3D reconstruction (the camera localisation performs 3D reconstruction as part of the evaluation pipeline). 3D reconstruction pipelines assume a detect and describe pipeline for extracting matches, which implicitly have this assumption baked in to their setup, as they match the same detected points across different images.

However, this assumption is not implicit to our approach, as we only find correspondences between pairs of images at a time. Further refining the points means that the location of a refined point from one pair of images will not necessarily match that of another pair, invalidating the track and negatively affecting the 3D reconstruction. Incorporating these refined points would require rethinking how we incorporate refined correspondences in a 3D reconstruction pipeline. For example, we could use a reference image against which further images are compared and incorporated. Once a reference image has been exhausted, a new reference image would be chosen and so on. We leave such an investigation to future work, but the boost in performance demonstrates the further potential of our dense-to-dense setup.

B.3 SfM Terminology

Here we define the metrics used in reporting the SfM results. Note that these metrics are only indicative of the quality of the 3D model; please look at the reconstructed models in the zipped video for a qualitative estimate as to their respective quality.

1. ↑ # Reg. Ims: The number of registered images. This is the number of images that are able to be put into correspondence and for which cameras were obtained when doing the 3D reconstruction. A higher value means more images were registered, implying a better model.

2. ↑ # Sparse Pts: The number of sparse points. This is the number of sparse points obtained after performing the 3D geometry estimation. The higher the number indicates a better model, as more correspondences were able to be triangulated.

3. ↑ Track Len: The track length. How many images a given 3D point is seen in on average. If this is higher, it indicates that the model is more robust, as more images see that 3D point.
Table 3: SfM. We compare our approach to three baselines on 3D reconstruction for three scenes with a large number of images. Our method obtains superior performance across the metrics except for reprojection error and dense correspondences, despite using coarse correspondences and a single scale. In particular, our method registers more images and obtains more sparse 3D points. ↑ denotes higher is better. ↓ denotes lower is better.

| LMark       | Method          | ↑ # Reg. Imgs | ↑ # Sparse Pts | ↑ Track Len | ↓ Reproj. Err | ↑ # Dense Pts |
|-------------|-----------------|---------------|----------------|-------------|---------------|--------------|
|             | RootSIFT [29]   | 500           | 116K           | 6.32        | 0.60px        | 1.82M        |
| Madrid      | GeoDesc [32]    | 495           | 144K           | 5.97        | 0.65px        | 1.56M        |
|            | Ours            | 702           | 256K           | 6.09        | 1.30px        | 1.10M        |
| Metropolis  | RootSIFT [29]   | 1035          | 338K           | 5.52        | 0.69px        | 4.23M        |
| 1344 images | GeoDesc [32]    | 1004          | 441K           | 5.14        | 0.73px        | 3.88M        |
|             | Ours            | 1072          | 570K           | 6.60        | 1.34px        | 2.11M        |
| Gendarmen-  | RootSIFT [29]   | 804           | 239K           | 7.76        | 0.61px        | 3.05M        |
| markt       | GeoDesc [32]    | 776           | 341K           | 6.71        | 0.63px        | 2.73M        |
| 1463 images | Ours            | 967           | 452K           | 5.82        | 1.32px        | 1.81M        |
| Tower of     | RootSIFT [29]   | 708           | 287K           | 5.20        | 1.34px        | 2.86M        |
| London       | GeoDesc [32]    | 965           | 301K           | 5.55        | 1.28px        | 3.15M        |
|             | Ours            | 1072          | 570K           | 6.60        | 1.34px        | 2.11M        |

Table 4: Results on the YFCC dataset [69]. Higher is better. Our approach outperforms all other approaches (e.g. all but RANSAC-Flow) that operate directly on features for smaller angle errors and performs competitively for larger angle errors. Additionally, this dataset clearly demonstrates the utility of our ablations. Note that RANSAC-Flow [62] is a two stage approach that first registers images and then learns optical flow. Such an approach could be added on top of ours.

| Method        | mAP@5° | mAP@10° |
|---------------|--------|---------|
| SIFT [29]     | 46.83  | 68.03   |
| ContextDesc   | 47.68  | 69.55   |
| Superpoint    | 30.50  | 50.83   |
| PointCN [38]  | 47.98  | -       |
| PointNet++ [45] [77] | 46.23 | -       |
| N²Net [33]    | 49.13  | -       |
| DFE [46]      | 49.45  | -       |
| OANet [77]    | 52.18  | -       |
| RANSAC-Flow   | 64.68  | 82.40   |
| Ours (w/o conf) | 31.60  | 40.80   |
| Ours (w/o cond) | 53.43  | 65.13   |
| Ours          | 55.58  | 66.79   |
| Ours (E-B1)   | 57.23  | 68.39   |

4. ↓ Reproj err: The reprojection error. This is the average pixel error between a 3D point and its projection in the images. If this is lower, it indicates the 3D points are more accurate.

5. ↑ # Dense Points: The number of dense points. This is the number of dense points in the final 3D model following the dense matching step. The higher this is, the more of the 3D structure was reconstructed.

B.4 Further Results for SfM

In this section we compare our 3D reconstruction on the Local Feature Evaluation Benchmark [60] to two additional baselines [9, 32] in Table 3. These results were not included in the paper as these additional baselines perform similarly to SIFT [29] and there was limited space. However, we note that both of these baselines use learned descriptors, yet they do not perform any better than SIFT in terms of the number of registered images and sparse 3D points. Our method performs significantly better across all three large scale datasets for obtaining sparse 3D points and registering images, demonstrating the robustness of our approach.

B.5 The YFCC100M Dataset

Here we report results for our model and ablations on the YFCC100M [69] dataset. This dataset further demonstrates the superiority of our approach to other detect and describe setups and the
In this experiment, we look at the results of using a NCE loss as opposed to a hinge loss. We see that using a NCE loss, we still achieve high quality, state of the art results, demonstrating the robustness of our approach.

Table 5: Results on Aachen Day-Night using a NCE loss. We can see that training with NCE is slightly worse than our hinge loss, but it is competitive with other state-of-the-art methods. Higher is better. * indicates the method was trained on the Aachen dataset.

| Method            | Type | Threshold | 0.25m (2°) | 0.5m (5°) | 5m (10°) |
|-------------------|------|-----------|------------|-----------|----------|
| Upright RootSIFT  | Spa  | 36.7      | 54.1       | 72.5      |
| DenseSFM          | Den  | 39.8      | 60.2       | 84.7      |
| Han+, HN++        | Spa  | 39.8      | 61.2       | 77.6      |
| Superpoint        | Spa  | 42.8      | 57.1       | 75.5      |
| DELF              | Spa  | 39.8      | 61.2       | 85.7      |
| D2-Net            | Spa  | **44.9**  | 66.3       | **88.8**  |
| R2D2*             | Spa  | 45.9      | 66.3       | **88.8**  |
| Ours (nce)        | Den  | 42.9      | 62.2       | 87.8      |
| Ours (G = 256)    | Den  | **44.9**  | 68.4       | 87.8      |
| Ours              | Den  | **44.9**  | **70.4**   | **88.8**  |

Utility of each component of our model (i.e. the confidence and attention mechanism to condition the features on both images).

Setup. The task of this dataset is to perform two view geometry estimation on four scenes with 1000 pairs each. Given a pair of images, the task is to use estimated correspondences to predict the essential matrix using the known intrinsic matrices. The essential matrix is decomposed into the rotation and translation component [17]. The reported error metric is the percentage of images that have the rotation and translation error (in degrees) less than a given threshold.

To run D2D on this dataset, we first use D2D to extract high quality matches for each pair of images. We use the known intrinsics to convert these to points in camera space. We then use RANSAC [11] and the 5-point algorithm in order to obtain the essential matrix [17].

Results. The results are given in Table 4. They demonstrate that our model achieves superior performance for low error thresholds to other methods that directly operate on extracted features. The results further demonstrate that the uniqueness score and conditioning using attention mechanisms are crucial for good performance, validating our model design.

Finally, our model does a bit worse than RANSAC-Flow [62]. However, we note [62] uses a segmentation model to restrict correspondences to only regions in the foreground of the image (e.g. the segmentation model is used to remove correspondences in the sky). Additionally, this method first registers images under a homography using predetected correspondences and then trains a network on top of the transformed images to perform fine-grained optical flow. As a result, this method performs as well as the underlying correspondences. Considering that our method has consistently been demonstrated to perform comparably or better than previous approaches for obtaining correspondences, we note that this method could be used on top of ours for presumably further improved performance. However, as this unpublished work was only recently released on arXiv, we leave this for future work.
B.6 An InfoNCE Loss

In the paper we demonstrate the robustness of our approach to the precise choice of architecture (e.g. we can achieve impressive results using a ResNet [18] or EfficientNet [68] backbone).

Here, we consider using the CPC objective [42], which is inspired by Noise-Contrastive Estimation (NCE) approaches [14, 37], and so is called an InfoNCE loss. While this is also a contrastive loss, similarly to the hinge loss in the main paper, the implementation is different. We find that we can still achieve impressive results with this loss, demonstrating the robustness of the approach to the precise choice of loss.

B.6.1 Implementation

We follow the implementation of [42], except that because we use normalised features, we add a temperature $\tau$. This is essential for good performance.

The setup is the same as that described in the paper, except for the implementation of the loss. Assume we have two descriptor maps $D_1$ and $D_2$ corresponding to the two input images $I_1$ and $I_2$. At a location $i$ in $D_1$, we obtain the descriptor vector $d_1^i \in \mathbb{R}^c$. To compare descriptor vectors, we first normalise and then use the scalar product to obtain a scalar matching score:

$$s(d_1^i, d_2^j) = \frac{(d_1^i \cdot d_2^j)}{||d_1^i||_2 \cdot ||d_2^j||_2}. \quad (6)$$

If the score is near 1, this is most likely a match. If it is near $-1$, it is most likely not a match.

Again, as in the paper, given two images of a scene $I_1$ and $I_2$ with a known set of correspondences from MegaDepth [27], we randomly select a set $P$ of $L$ true correspondences. For each positive correspondence $p$, we additionally select a set $N_p$ of $K$ negative correspondences. The loss is then

$$L_{nce} = -\log \frac{1}{L} \sum_{p=(x,y) \in P} \frac{\exp(\tau \cdot s(d_1^x, d_2^y))}{\sum_{(x,y) \in \mathbb{N}_p} \exp(\tau \cdot s(d_1^x, d_2^y))} \quad (7)$$

where $\tau = 20$ is a temperature.

B.6.2 Experiments

We train the InfoNCE model using the $L_{nce}$ in the same manner as in the paper and evaluate it on two of the datasets discussed in the paper: HPatches [2] and Aachen [55, 56].

**HPatches.** The results are given in Fig. 7. From here we see that our model with an NCE loss performs competitively on this dataset, obtaining superior results to that of the model in the paper.

**Aachen Day-Night.** The results are given in Table 5. These results demonstrate that using an NCE loss with our backbone achieves results competitive with other state-of-the-art approaches but it performs a bit worse than the hinge loss used in the paper.

**Discussion.** These experiments have shown that we can achieve high quality results when using a different but effective contrastive loss. As a result, our approach is robust to not only the backbone architecture (as shown in the paper) but also the precise choice of the contrastive loss.

C Architectures

**Architecture for Main Model.** The components of the main model are described in the main text. Here we give further details of the different components. The encoder is a ResNet50 model, except that we extract the features from the last two blocks to obtain feature maps $f^S_1$ and $f^L_1$. The details are given in Table 6.

The features $f^S_1$ and $f^L_1$ are projected in order to reduce the number of channels using linear layers. There are four linear layers (one for each of $f^L_1$, $f^S_1$, $f^S_2$, $f^L_2$). The linear layers operating at the larger resolution ($f^L_1$) project the features from 2048 size vectors to 256. The linear layers operating at the smaller resolution ($f^S_2$) project the features from 1024 size vectors to 128.
Table 6: Encoder of Main Model. The encoder is a ResNet50 [18] encoder. The convolutions column denotes the convolutional and max-pooling operations. Implicit are the BatchNorm and ReLU operations that follow each convolution.

| layer name | output size | convolutions |
|------------|-------------|--------------|
| conv 1     | 128 × 128   | 7 × 7, 64, stride 2 |
| conv2_x    | 64 × 64     | 3 × 3 max pool, stride 2 |
|            |             | (1 × 1, 64) |
|            |             | (3 × 3, 64) |
|            |             | (1 × 1, 1, 256) |
| conv3_x    | 32 × 32     | (1 × 1, 128) |
|            |             | (3 × 3, 128) |
|            |             | (1 × 1, 512) |
| conv4_x (\(f^L_i\)) | 16 × 16 | (1 × 1, 256) |
|            |             | (3 × 3, 256) |
|            |             | (1 × 1, 1, 1024) |
| conv5_x (\(f^S_i\)) | 8 × 8 | (1 × 1, 512) |
|            |             | (3 × 3, 512) |
|            |             | (1 × 1, 1, 2048) |

Table 7: Decoder of Main Model. The decoder is a UNet [53] variant. The convolutions column denotes the convolutional operations. Implicit are the BatchNorm and ReLU operations that follow each convolution as well as the bi-linear upsampling operation that resizes features from the previous layer before the convolutional blocks.

| layer name | inputs | output size | convolutions |
|------------|--------|-------------|--------------|
| deconv_5 | conf5_x (×2), \(\hat{f}^S_i\) | 16 × 16 | \(3 × 3, 256\) |
|           |        |             | \(3 × 3, 256\) |
| deconv_4 | deconv_5, conv4_x, \(f^L_i\) | 32 × 32 | \(3 × 3, 256\) |
|           |        |             | \(3 × 3, 256\) |
| deconv_3 | deconv_4, conv3_x | 64 × 64 | \(3 × 3, 128\) |
|           |        |             | \(3 × 3, 128\) |
| deconv_2 | deconv_3, conv2_x | 128 × 128 | \(3 × 3, 128\) |
|           |        |             | \(3 × 3, 128\) |
| deconv_1 | deconv_2, conv1_x | 256 × 256 | \(3 × 3, 64\) |
|           |        |             | \(3 × 3, 64\) |

The decoder consists of a sequence of decoder layers. A layer takes the bi-linearly upsampled features from the previous layer, the corresponding encoded features, and optionally the attended features. The details are given in Table 7. Finally, the unnormalised features are passed to a MLP which regresses the distinctiveness score. The MLP consists of three blocks of linear layer (with no bias) and batch normalisation followed by a sigmoid layer. The channel dimensions are 64 → 1 → 1.

Architecture for Stylisation. In Fig. 8 we illustrate further our method for stylising images using our initial set of dense correspondences. Given two images \(I_v\) and \(I_S\), the task is to generate an image with the viewpoint of \(I_v\) and the style of \(I_S\). In brief, we first use the dense correspondences to sample from \(I_S\) to obtain the initial image. We then use a refinement network to fix errors and fill in missing regions. We do this in two stages. The first stage fixes errors and can be trained with an L1 loss. However, we wish to use a discriminator loss, but using this directly on the intermediary image causes information to leak and the generated image to match \(I_v\), which is input to the network. To use a discriminator loss without leaking information, we use a second network (a sequence of ResNet blocks) which is trained with both an L1 and discriminator loss. Crucially, we do not allow gradients to flow from the second network (the set of ResNet blocks) to the first.

D Additional Results

D.1 Visualising Confidence Maps

In Fig. 9 we illustrate the predicted confidence maps. Here we can see that the most confident parts of both images are regions that exist in both images. We further test this by looking at what the confident
Figure 8: Illustration of the architecture used for the downstream stylization task. We first use our model to predict correspondences to transform the input image $I_S$ into the position of $I_v$. The next step is to learn how to fill in and fix errors with a discriminator. However, we additionally can train the portion of the generated image visible in both input images to match the true transformed image. We also can use high frequency information in $I_v$ when performing this transformation. However, if we train end-to-end then information will leak from $I_v$ to the generated image. As a result, we train in two stages. We first generate an intermediary image using a UNet [53] which is trained using an L1 loss on the generated intermediary image for regions that are in common between the two images (and for which we can determine what the true pixel colour should be). We input this intermediary image to a set of ResNet blocks to refine the original prediction: this is trained with both a discriminator (pix2pixHD [72]) and L1 loss.

Figure 9: Random pairs associated with their predicted confidence on the MegaDepth test set. Top row shows the input pairs, bottom row the associated confidence maps.

maps look like when we keep one input view the same but change the other in Fig. 10. We can see that the confidence maps change depending on the input images: the output is indeed dependent on both input images.

D.2 Qualitative Results

In Fig. 11 we show random matches obtained by our method on random samples from the Aachen Day-Night test set and similarly in Fig. 12 for HPatches, Fig. 13 for 3D reconstruction using SfM and Fig. 14 for the stylisation task.
Figure 10: Random pairs associated with their predicted confidence on the MegaDepth test set. Top: Pairs of image of the same scene. Bottom: Pairs with one image from the top associated with another from a different scene. Notice that the confidence changes between the two cases and becomes irrelevant.

Figure 11: Random sample matches found on the Aachen Day-Night test set. We show the original image pairs top and the pairs with a random subset of located correspondences overlaid below.
Figure 12: Random sample matches found on the HPatches test set. Rows show progressively harder illumination or viewpoint changes. We first show the original image pairs followed by the pairs with a random subset of located correspondences overlaid.
Figure 13: Additional SfM results. Randomly selected input images and 3D models reconstructed using our matches and those obtained using the SIFT [29] baseline. This figure demonstrates the variety of the input images and their scene shift as well as that both our and [29] are of similar quality for these image sets.
Figure 14: Additional stylisation results. These results are randomly sampled from the test set of MegaDepth.
References

[1] R. Arandjelović and A. Zisserman. Three things everyone should know to improve object retrieval. In Proc. CVPR, 2012.

[2] V. Balntas, K. Lenc, A. Vedaldi, and K. Mikolajczyk. Hpatch: A benchmark and evaluation of handcrafted and learned local descriptors. In Proc. CVPR, 2017. V. Balntas and K. Lenc contributed equally to this work.

[3] V. Balntas, E. Riba, D. Ponsa, and K. Mikolajczyk. Learning local feature descriptors with triplets and shallow convolutional neural networks. In Proc. BMVC, 2016.

[4] C. Barnes, E. Shechtman, A. Finkelstein, and D. B. Goldman. PatchMatch: A randomized correspondence algorithm for structural image editing. In ACM Transactions on Graphics (ToG), volume 28, page 24. ACM, 2009.

[5] H. Bay, T. Tuytelaars, and L. Van Gool. SURF: Speeded up robust features. In Proc. ECCV, May 2006.

[6] M. Calonder, V. Lepetit, C. Strecha, and P. Fua. BRIEF: Binary robust independent elementary features. In Proc. ECCV, 2010.

[7] C. B. Choy, J. Gwak, S. Savarese, and M. Chandraker. Universal correspondence network. In NeurIPS, 2016.

[8] D. DeTone, T. Malisiewicz, and A. Rabinovich. SuperPoint: Self-supervised interest point detection and description. In CVPR workshops, 2018.

[9] 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 Proc. CVPR, 2019.

[10] M. E. Fathy, Q.-H. Tran, M. Zeeshan Zia, P. Vernaza, and M. Chandraker. Hierarchical metric learning and matching for 2D and 3D geometric correspondences. In Proc. ECCV, 2018.

[11] M. A. Fischler and R. C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. Comm. ACM, 24(6):381–395, 1981.

[12] D. F. Fouhey, A. Gupta, and A. Zisserman. 3d shape attributes. In Proc. CVPR, 2016.

[13] H. Germain, G. Bourmaud, and V. Lepetit. S2DNet: Learning accurate correspondences for sparse-to-dense feature matching. arXiv preprint arXiv:2004.01673, 2020.

[14] M. Gutmann and A. Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In Proc. AISTATS, 2010.

[15] M. J. Hannah. Computer matching of areas in stereo images. PhD thesis, 1974.

[16] C. G. Harris and M. Stephens. A combined corner and edge detector. In Proc. Alvey Vision Conf., pages 147–151, 1988.

[17] R. I. Hartley and A. Zisserman. Multiple View Geometry in Computer Vision. Cambridge University Press, ISBN: 0521623049, 2000.

[18] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. arXiv preprint arXiv:1512.03385, 2015.

[19] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox. FlowNet 2.0: Evolution of optical flow estimation with deep networks. In Proc. CVPR, 2017.

[20] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. In Proc. CVPR, 2017.

[21] Y. Ke and R. Sukthankar. PCA-SIFT: A more distinctive representation for local image descriptors. In Proc. CVPR, Jun 2004.

[22] S. Kim, S. Lin, S. R. Jeon, D. Min, and K. Sohn. Recurrent transformer networks for semantic correspondence. In NeurIPS, 2018.

[23] D. P. Kingma and M. Welling. Auto-encoding variational bayes. In Proc. ICLR, 2014.

[24] Z. Lai, E. Lu, and W. Xie. MAST: A memory-augmented self-supervised tracker. In Proc. CVPR, 2020.
[25] S. Leutenegger, M. Chli, and R. Siegwart. BRISK: Binary robust invariant scalable keypoints. In Proc. ICCV, pages 2548–2555, 2011.
[26] S. Li, K. Han, T. W. Costain, H. Howard-Jenkins, and V. Prisacariu. Correspondence networks with adaptive neighbourhood consensus. In Proc. CVPR, 2020.
[27] Z. Li and N. Snavely. Megadepth: Learning single-view depth prediction from internet photos. In Proc. CVPR, 2018.
[28] C. Liu, J. Yuen, and A. Torralba. Sift Flow: Dense correspondence across scenes and its applications. IEEE PAMI, 33(5):978–994, 2010.
[29] D. Lowe. Distinctive image features from scale-invariant keypoints. IJCV, 60(2):91–110, 2004.
[30] F. Luan, S. Paris, E. Shechtman, and K. Bala. Deep photo style transfer. Proc. CVPR, 2017.
[31] 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. In Proc. CVPR, 2019.
[32] 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 Proc. ECCV, 2018.
[33] K. Mikolajczyk and C. Schmid. Scale & affine invariant interest point detectors. IJCV, 1(60):63–86, 2004.
[34] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. Van Gool. A comparison of affine region detectors. IJCV, 65(1/2):43–72, 2005.
[35] A. Mishchuk, D. Mishkin, F. Radenovic, and J. Matas. Working hard to know your neighbor’s margins: Local descriptor learning loss. In NeurIPS, 2017.
[36] D. Mishkin, F. Radenovic, and J. Matas. Repeatability is not enough: Learning affine regions via discriminability. In Proc. ECCV, 2018.
[37] A. Mnih and K. Kavukcuoglu. Learning word embeddings efficiently with noise-contrastive estimation. In NeurIPS, 2013.
[38] K. Moo Yi, E. Trulls, Y. Ono, V. Lepeitit, M. Salzmann, and P. Fua. Learning to find good correspondences. In Proc. CVPR, 2018.
[39] H. Noh, A. Araujo, J. Sim, T. Weyand, and B. Han. Large-scale image retrieval with attentive deep local features. In Proc. ICCV, 2017.
[40] D. Novotny, D. Larlus, and A. Vedaldi. Anchornet: A weakly supervised network to learn geometry-sensitive features for semantic matching. In Proc. CVPR, 2017.
[41] Y. Ono, E. Trulls, P. Fua, and K. M. Yi. LF-Net: learning local features from images. In NeurIPS, 2018.
[42] A. v. d. Oord, Y. Li, and O. Vinyals. Representation learning with contrastive predictive coding. arXiv preprint arXiv:1807.03748, 2018.
[43] T. Plötz and S. Roth. Neural nearest neighbors networks. In NeurIPS, 2018.
[44] M. Pollefeys. Self-calibration and metric 3D reconstruction from uncalibrated image sequences. PhD thesis, ESAT-PSI, K.U.Leuven, 1999.
[45] C. R. Qi, L. Yi, H. Su, and L. J. Guibas. PointNet++: Deep hierarchical feature learning on point sets in a metric space. In NeurIPS, 2017.
[46] R. Ranftl and V. Koltun. Deep fundamental matrix estimation. In Proc. ECCV, 2018.
[47] A. Ranjan and M. J. Black. Optical flow estimation using a spatial pyramid network. In Proc. CVPR, 2017.
[48] J. Revaud, P. Weinzaepfel, C. De Souza, N. Pion, G. Csurka, Y. Cabon, and M. Humenberger. R2D2: Repeatable and reliable detector and descriptor. In NeurIPS, 2019.
[49] I. Rocco, R. Arandjelovic, and J. Sivic. Convolutional neural network architecture for geometric matching. In Proc. CVPR, 2017.
[50] I. Rocco, R. Arandjelovic, and J. Sivic. End-to-end weakly-supervised semantic alignment. In Proc. CVPR, 2017.
[51] I. Rocco, R. Arandjelović, and J. Sivic. Efficient neighbourhood consensus networks via submanifold sparse convolutions. CoRR, abs/2004.10566, 2020.

[52] I. Rocco, M. Cimpoi, R. Arandjelović, A. Torii, T. Pajdla, and J. Sivic. Neighbourhood consensus networks. In NeurIPS, 2017.

[53] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In Proc. MICCAI, pages 234–241. Springer, 2015.

[54] E. Rublee, V. Rabaud, K. Konolige, and G. R. Bradski. Orb: An efficient alternative to SIFT or SURF. In Proc. ICCV, pages 2564–2571, 2011.

[55] T. Sattler, W. Maddern, C. Toft, A. Torii, L. Hammarstrand, E. Stenborg, D. Safari, M. Okutomi, M. Pollefeys, J. Sivic, F. K. Kahl, and T. Pajdla. Benchmarking 6dof outdoor visual localization in changing conditions. In Proc. CVPR, 2018.

[56] T. Sattler, T. Weyand, B. Leibe, and L. Kobbelt. Image retrieval for image-based localization revisited. In Proc. BMVC, 2012.

[57] N. Savinov, L. Ladicky, and M. Pollefeys. Matching neural paths: transfer from recognition to correspondence search. In NeurIPS, 2017.

[58] N. Savinov, A. Seki, L. Ladicky, T. Sattler, and M. Pollefeys. Quad-Networks: unsupervised learning to rank for interest point detection. In Proc. CVPR, 2017.

[59] J. L. Schönberger and J.-M. Frahm. Structure-from-motion revisited. In Proc. CVPR, 2016.

[60] J. L. Schönberger, H. Hardmeier, T. Sattler, and M. Pollefeys. Comparative evaluation of hand-crafted and learned local features. In Proc. CVPR, 2017.

[61] J. L. Schönberger, E. Zheng, M. Pollefeys, and J.-M. Frahm. Pixelwise view selection for unstructured multi-view stereo. In Proc. ECCV, 2016.

[62] X. Shen, F. Darmon, A. A. Efros, and M. Aubry. RANSAC-Flow: generic two-stage image alignment. arXiv preprint arXiv:2004.01526, 2020.

[63] E. Simo-Serra, E. Trulls, L. Ferraz, I. Kokkinos, P. Fua, and F. Moreno-Noguer. Discriminative learning of deep convolutional feature point descriptors. In Proc. ICCV, 2015.

[64] K. Simonyan, A. Vedaldi, and A. Zisserman. Learning local feature descriptors using convex optimisation. IEEE PAMI, 2014.

[65] D. Sun, X. Yang, M.-Y. Liu, and J. Kautz. PWC-Net: CNNs for optical flow using pyramid, warping, and cost volume. In Proc. CVPR, 2018.

[66] J. Sun, Y. Li, S. B. Kang, and H.-Y. Shum. Symmetric stereo matching for occlusion handling. In Proc. CVPR, 2005.

[67] R. Szeliski. Computer vision: algorithms and applications. Springer Science & Business Media, 2010.

[68] M. Tan and Q. V. Le. EfficientNet: Rethinking model scaling for convolutional neural networks. Proc. ICML, 2019.

[69] B. Thomée, D. A. Shamma, G. Friedland, B. Elizalde, K. Ni, D. Poland, D. Borth, and L.-J. Li. Yfcc100m: the new data in multimedia research. Communications of the ACM, 59(2):64–73, 2016.

[70] E. Tola, V. Lepeit, and P. Fua. A fast local descriptor for dense matching. In Proc. CVPR, 2008.

[71] C. Vondrick, A. Shrivastava, A. Fathi, S. Guadarrama, and K. Murphy. Tracking emerges by colorizing videos. In Proc. ECCV, 2018.

[72] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, and B. Catanzaro. High-resolution image synthesis and semantic manipulation with conditional GANs. In Proc. CVPR, 2018.

[73] X. Wang, A. Jabri, and A. A. Efros. Learning correspondence from the cycle-consistency of time. In Proc. CVPR, 2019.

[74] O. J. Woodford, P. H. S. Torr, I. D. Reid, and A. W. Fitzgibbon. Global stereo reconstruction under second order smoothness priors. IEEE PAMI, 31(12):2115–2128, 2009.
[75] K. M. Yi, E. Trulls, V. Lepetit, and P. Fua. LIFT: Learned invariant feature transform. In Proc. ECCV, 2016.

[76] J. Zhang, D. Sun, Z. Luo, A. Yao, L. Zhou, T. Shen, Y. Chen, L. Quan, and H. Liao. Learning two-view correspondences and geometry using order-aware network. In Proc. ICCV, 2019.

[77] L. Zhang and S. Rusinkiewicz. Learning to detect features in texture images. In Proc. CVPR, 2018.