Image Matching across Wide Baselines: From Paper to Practice

Yuhe Jin\textsuperscript{1}  Dmytro Mishkin\textsuperscript{2}  Anastasiia Mishchuk\textsuperscript{3}  Jiří Matas\textsuperscript{2}  Pascal Fua\textsuperscript{3}  Kwang Moo Yi\textsuperscript{1}  Eduard Trulls\textsuperscript{4}
\textsuperscript{1}University of Victoria  \textsuperscript{2}Czech Technical University in Prague  \textsuperscript{3}École Polytechnique Fédérale de Lausanne  \textsuperscript{4}Google Research

Abstract

We introduce a comprehensive benchmark for local features and robust estimation algorithms, focusing on the downstream task – the accuracy of the reconstructed camera pose – as our primary metric. Our pipeline’s modular structure allows us to easily integrate, configure, and combine methods and heuristics. We demonstrate this by embedding dozens of popular algorithms and evaluating them, from seminal works to the cutting edge of machine learning research. We show that with proper settings, classical solutions may still outperform the perceived state of the art.

Besides establishing the actual state of the art, the experiments conducted in this paper reveal unexpected properties of SfM pipelines that can be exploited to help improve their performance, for both algorithmic and learned methods. Data and code are online\textsuperscript{1}, providing an easy-to-use and flexible framework for the benchmarking of local feature and robust estimation methods, both alongside and against top-performing methods. This work provides the basis for an open challenge on wide-baseline image matching\textsuperscript{2}.

1. Introduction

Matching two or more views of a given scene is at the core of fundamental computer vision problems, including image retrieval [48, 7, 69, 91, 63], 3D reconstruction [3, 43, 79, 106], re-localization [74, 75, 51], and SLAM [61, 30, 31]. Despite decades of research, it remains unsolved in the general, wide-baseline scenario, as the number of factors to deal with can be exceedingly large: viewpoint, scale, rotation, illumination, occlusions, and camera properties render the problem very challenging in combination. Because of this, it has traditionally been approached with sparse methods – that is, with local features.

Recent efforts have moved towards end-to-end solutions [45, 10, 22], but they do not yet outperform classical methods [77, 105] that break the problem into separate steps. For example, in stereo one may extract local features, such as SIFT [48], build a list of putative matches by a nearest-neighbour search in descriptor space, and retrieve the pose with a minimal solver inside a robust estimator, such as the 7-point algorithm [41] in a RANSAC loop [36]. To build a 3D reconstruction out of a set of images, we would feed the same matches to a bundle adjustment pipeline [40, 93] to jointly optimize the camera intrinsics, extrinsics, and 3D point locations. This modular structure simplifies the problem and allows for incremental improvements, of which there have been hundreds, if not thousands.

New methods for each of these sub-problems, such as feature extraction or pose estimation, are typically studied in isolation, using intermediate metrics, which simplifies their evaluation. However, there is no guarantee that gains in one part of the pipeline will translate to the final application, as these components interact in complex ways. For example, patch descriptors, including very recent works [42, 97, 90, 60], are often evaluated on Brown’s seminal patch retrieval database [20], showing dramatic im-

\textsuperscript{1}https://github.com/vcg-uvic/image-matching-benchmark
\textsuperscript{2}https://vision.uvic.ca/image-matching-challenge
provements over handcrafted methods such as SIFT, but it is unclear whether this is also true on real-world applications – in fact, we later demonstrate that the gap may narrow dramatically when baselines are properly tuned.

We posit that it is time to look beyond intermediate metrics and focus on downstream performance. This is particularly crucial now, with deep networks seemingly outperforming algorithmic solutions on classical problems such as outlier filtering [100, 70, 102, 85, 18], bundle adjustment [86, 81], SfM [96, 5] and SLAM [87, 46]. To this end, we introduce a benchmark for wide-baseline image matching, including: (a) A dataset with 30k images with depth maps and ground truth poses (called posed images later). (b) A modular pipeline incorporating dozens of methods for feature extraction and matching, and pose estimation, both classical and state-of-the-art, as well as multiple heuristics, which can be swapped out and tuned separately. (c) Two downstream tasks – stereo and multi-view reconstruction – evaluated with downstream and intermediate metrics. (d) A thorough study of dozens of methods and techniques, hand-crafted and learned, and their combination, along with a procedure for hyper-parameter selection.

This framework will enable researchers to evaluate how a new approach performs in a standardized pipeline, both against its competitors, and alongside state-of-the-art solutions for other components, from which it cannot be truly detached. This is crucial, as true performance can be easily hidden by sub-optimal hyperparameters. Data and code are publicly available.

2. Related Work

The literature on image matching is too vast for a thorough overview. We cover relevant methods for feature extraction and matching, pose estimation, 3D reconstruction, datasets, and evaluation frameworks.

Local features. Local features became a staple in computer vision with the introduction of SIFT [48]. They typically involve three distinct steps: keypoint detection, orientation estimation, and descriptor extraction. Other popular, hand-crafted solutions are SURF [15], ORB [73], and AKAZE [4]. Modern descriptors train deep networks on pre-cropped patches, typically from SIFT keypoints (i.e. Difference of Gaussians or DoG). They include DeepDesc [82], TFeat [11], L2-Net [89], Hardnet [57], SOSSNet [90], and LogPolarDesc [34] – most of them are trained on the same dataset [20]. Recent work leverage additional cues, such as geometry or global context, including GeoDesc [50] and ContextDesc [49]. There have been multiple attempts to learn keypoint detectors separately from the descriptor, including TILDE [95], TCDDet [103], QuadNet [78], and KeyNet [13]. An alternative is to treat this as an end-to-end learning problem, a trend that started with the introduction of LIFT [99] and also includes DELF [63], SuperPoint [31], LF-Net [64], D2-Net [33] and R2D2 [72].

Robust matching. Inlier ratios in wide-baseline stereo can be below 10% – and sometimes much lower. This is typically approached with iterative sampling schemes based on RANSAC [36], relying on closed-form solutions for pose solving such as the 5- [62], 7- [41] or 8-point algorithm [39]. Improvements to this classical framework include local optimization [24], MLESAC [92], PROSAC [23], DEGENSAC [26], GC-RANSAC [12], and MAGSAC [29]. Recent efforts, starting with CNe (Context Networks) in [100], train deep networks for outlier rejection taking correspondences as input, often followed by a RANSAC loop. Follow-up works include [70, 104, 85, 102]. Despite their promise, it remains unclear how well they perform in real settings.

Structure from Motion. In Structure-from-Motion (SfM) one jointly optimizes the location of the 3D points and the camera intrinsics and extrinsics. Many improvements have been proposed over the years [3, 43, 27, 37, 106]. The most popular frameworks are VisualSFM [98] and COLMAP [79] – we rely on the latter, to generate the ground truth and as the backbone of our multi-view task.

Datasets and benchmarks. Early work relied on the Oxford dataset [54], with 48 images and ground truth homographies. It helped establish two common metrics: repeatability and matching score. Repeatability evaluates the keypoint detector – given keypoint sets over two images, projected into each other, it is defined as the ratio of keypoints whose support region overlap is above a threshold. The matching score (MS) is similarly defined, but also requires their descriptors to be nearest neighbours. Both require pixel-to-pixel correspondences – features outside valid areas are ignored. A modern alternative to Oxford is HPatches [9], which contains 696 images with differences in illumination or viewpoint – the scenes are planar, without occlusions. Other datasets include DTU [1], Edge Foci [107], Webcam [95], AMOS [67], and Strecha’s [83]. They all have limitations – narrow baselines, noisy ground truth, or contain few images. Learned descriptors are often trained and evaluated on [21], where they outperform SIFT by orders of magnitude – probably due to overfitting. Datasets used for navigation, re-localization or SLAM in outdoor environments are also relevant, including Kitti [38], Aachen [76], Robotcar [52], and CMU seasons [75, 8], but may not feature the wide range of transformations present in Phototourism data. This includes Megadepth [47], a dataset which builds on COLMAP – and could be folded into ours.

Modern benchmarks, by contrast, are few and far between – they include VLBenchmark [44], HPatches [9], and SILDa [35] – all limited in scope. A large-scale benchmark for SfM was proposed in [80] – without ground truth. A recent work by Bian et al. [17] evaluates different methods for
pose estimation on several datasets – however, few methods are considered and they are not carefully tuned. We are, to the best of our knowledge, the first to introduce a public, comprehensive, and modular benchmark for sparse methods with downstream metrics.

3. The Phototourism Dataset

A wide range of imaging conditions and devices is necessary in order to compile a strong dataset – Phototourism images fit this description and are readily available. We thus build on 26 collections of popular landmarks originally selected in [43, 88], each with hundreds to thousands of images. We downsample them to a maximum size of 1024 pixels and pose them with COLMAP [79]. In addition to point clouds, COLMAP provides noisy but dense depth estimates. We remove occlusions from them using the reconstructed model: see Fig. 2 for examples. We rely on these depth maps to compute pixel-wise metrics – repeatability and matching score. We find that some images are flipped 90°; and use the poses to rotate them so they are roughly ‘upright’. We select 2 scenes for validation, 11 for testing, 90 and matching score. We find that some images are flipped 90°; and use the poses to rotate them so they are roughly ‘upright’. We select 2 scenes for validation, 11 for testing, and the rest for training – see the appendix for details.

Our core assumption is that we can obtain ‘accurate’ poses from large sets of images, which we then use as ‘ground truth’ to evaluate image matching on pairs or small subsets of images – a harder, proxy task. The improvement in accuracy with more images can be measured by subsets which have at least 100 3D points in common, as in [90, 102]. We find both criteria work well in practice.

4. Pipeline

Our pipeline is outlined in Fig. 3. It takes as input \(N=100\) images per scene. The feature extraction module computes up to \(K\) features from each image. The feature matching module generates a list of putative matches for each image pair, i.e. \(\frac{1}{2}N(N-1)=4950\) combinations. These matches can be optionally processed by an outlier pre-filtering module. They are then fed to two tasks: stereo, and multi-view reconstruction with SfM. We now describe each of these components in detail.

Feature extraction. We consider three broad families of local features. The first includes full, hand-crafted pipelines: SIFT [48] (and RootSIFT [6]), SURF [15], ORB [73], and AKAZE [4]. We take these from OpenCV. For all of them, except ORB, we lower the detection threshold to extract more features, which increases performance. We also consider DoG alternatives from VLFeat [94]: (VL-)DoG, Hessian [16], Hessian-Laplace [55], Harris-Laplace [55], MSER [53]; and their affine-covariant versions: DoG-Affine, Hessian-Affine [55, 14], DoG-AffNet [59], and Hessian-AffNet [59]. The second group includes descriptors learned on DoG keypoints: L2-Net [89], HardNet [57], Geodesc [50], SOSNet [90], ContextDesc [49], and LogPolarDesc [34]. The last group consists of pipelines learned end-to-end (e2e): Superpoint [31], LF-Net [64], and D2Net [33] – the latter with single- (SS) and multi-scale (MS) variants. Finally, we consider Key.Net [13], a learned detector paired with HardNet descriptors – their implementation performs poorly for us so we pair it with our own HardNet.

Feature matching. We break this step into four stages. Given images \(I_i\) and \(I_j\), \(i \neq j\), we create an initial set of matches by nearest neighbor (NN) matching from \(I_i\) to \(I_j\).
obtaining a set of matches $m_{i+j}$. We optionally do the same in the opposite direction, $m_{j+i}$. We then apply Lowe’s ratio test [48] to each list to filter out non-discriminative matches, with a threshold $r \in [0, 1]$, creating ‘curated’ lists $\tilde{m}_{i+j}$ and $\tilde{m}_{j+i}$. We obtain the final set of putative matches by taking their intersection, $\tilde{m}_{i+j} \cap \tilde{m}_{j+i} = \tilde{m}_{i+j} \cap \tilde{m}_{j+i}$ (known in the literature as one-to-one, mutual NN, bipartite, or cycle-consistent), or their union $\tilde{m}_{i+j} \cup \tilde{m}_{j+i} = \tilde{m}_{i+j}$ (symmetric). We refer to them as ‘both’ and ‘either’ respectively. We also implement simple unidirectional matching, i.e., $\tilde{m}_{i+j}$. Finally, we optionally apply a distance filter, removing matches whose distance is above a threshold.

**Outlier pre-filtering.** Context Networks [100], or CNe, proposed a method to find sparse correspondences with a permutation-equivariant deep network based on PointNet [68], and sparked a number of follow-up works [70, 28, 104, 102, 85]. We embed CNe into our framework. It works best paired with RANSAC [100, 85], so we consider it as an optional pre-filtering step – for both stereo and multi-view. As their published model was trained on one of our validation scenes, we re-train it on ‘Notre Dame’ and ‘Buckingham Palace’, following their training protocol, with 2000 SIFT features, unidirectional matching, and no ratio test.

**Stereo task.** The list of putative matches is given to a robust estimator, which we use to estimate $F_{i,j}$, the Fundamental matrix between $I_i$ and $I_j$. In addition to (locally-optimized) RANSAC [36, 25], as implemented in OpenCV [19] and sklear [65], we consider recent algorithms: DEGEN-RANSAC [26], GC-RANSAC [12] and MAGSAC [29]. For DEGEN-RANSAC we additionally consider disabling the degeneracy check, which theoretically would be equivalent to the OpenCV and sklear implementations – we call this variant ‘PyRANSAC’. Given $F_{i,j}$, we use the known intrinsics $K_{i,j}$ to compute the Essential matrix $E_{i,j}$, as $E_{i,j} = K_{i}^T F_{i,j} K_{j}$. Finally, we recover the relative rotation and translation vectors with a chirality check with OpenCV’s recoverPose.

**Multi-view task.** Large-scale SfM is notoriously hard to evaluate, as it requires accurate ground truth. Since our goal is to benchmark local features, not SfM itself, we opt for a different strategy. In this task we reconstruct a scene from small image subsets, which we call ‘bags’. We consider bags of 3, 5, 10, and 25 images, which are randomly sampled from the original set of 100 images per scene, with a co-visibility check. We create 100 bags for bag sizes 3 and 5, 50 for bag size 10, and 25 for bag size 25 – 275 SfM runs in total. We use COLMAP [79], feeding it the matches computed by the previous module – note that this comes before the robust estimation step, as COLMAP implements its own RANSAC. If multiple reconstructions are obtained, we consider the largest one. We also collect statistics such as the number of landmarks or the average track length. Both statistics and error metrics are averaged over the four bag sizes, each of which is in turn averaged over its bags.

**Error metrics.** We compute the angular error between the estimated and ground-truth translation and rotation vectors between two cameras, threshold over a given value, and compute the mean Average Precision (mAP) by integrating the curve up to that threshold – we find $10^\circ$ adequate. We also use this metric for multi-view, as the scenes are not up to scale and it is not possible to measure translation error in metric terms (we will explore this in the future). To do so, we average the error every pair of cameras (setting it to $\infty$ for unregistered views). For stereo, we can report this value for different co-visibility thresholds: we use $v = 0.1$, which preserves ‘hard’ pairs. Finally, we consider repeatability and matching score – as end-to-end methods do not report scale, we threshold by pixel distance. We also compute the Absolute trajectory error (ATE) [84], a metric widely used in SLAM, for our multiview task. As, again, the reconstructed model is without scale, we first scale the reconstructed model to the scale of the ground truth and then compute ATE. It needs a minimum of three points to align the two models – we only compute it for reconstructions with at least three registered images.

**Implementation.** Code is open-sourced, along with every method used in the paper\(^3\). Our implementation relies on Slurm for job scheduling – we will provide on-the-cloud, ready-to-go images. It can also run on a standard computer, sequentially. It is computationally expensive, as it requires matching over 50k image pairs. The most costly step is feature matching: 2-6 s. per image pair, depending on descriptor size. Outlier pre-filtering takes about 0.5-0.8 s. per pair, overhead aside. RANSAC methods vary between 0.5-1 s. – as explained in Section 5 we limit their number of iterations based on a compute budget, but the actual cost depends on the number of matches. We find COLMAP to vary drastically between set-ups. New methods will be continuously added, with contributions welcome.

5. Details are Important

Our experiments indicate that each method needs to be carefully tuned. In this section we outline the methodology we used to find the right hyperparameters on the validation set, and demonstrate why it is crucial to do so.

**RANSAC: Leveling the field.** Robust estimators are the most sensitive part of the stereo pipeline. All methods have three parameters in common: the confidence level in their estimates, $\tau$; the outlier epipolar threshold, $\eta$; and the maximum number of iterations, $\Gamma$. The confidence value is the least sensitive, so we set it to $\tau = 0.999999$. We evaluated

\(^3\)https://github.com/vcg-uvic/image-matching-benchmark-baselines
each method for different $\Gamma$ and $\eta$ pictured in Fig. 5, along with their computational cost in Fig. 4. We place all methods on an ‘even ground’ by setting a common budget of 0.5 seconds, where all methods have mostly converged – we do so by choosing $\Gamma$ as per Fig. 4, instead of actually enforcing a time limit. Optimal values for $\Gamma$ can vary drastically, from 10k for MAGSAC to 250k for PyRANSAC. The best results are obtained by MAGSAC, followed by DEGENSAC.

We patch OpenCV to increase the limit of iterations, which is hardcoded to $\Gamma = 1000$. This increases performance by 10-15% relative, within our budget. However, PyRANSAC is better, so we use it as our ‘vanilla’ RANSAC instead. The sklearn implementation is too slow for practical use.

We find that, in general, default settings can be woefully inadequate. For example, OpenCV sets $\tau = 0.99$ and $\eta = 3$ pixels, which results in a mAP at $10^\circ$ of 0.5292 on the validation set – a performance drop of 23.9% relative.

**RANSAC: One method at a time.** The last free parameter is the inlier threshold $\eta$. We expect the optimal value for this parameter to be different for each local feature, with looser thresholds required for methods operating on higher recall/precision. We report a wide array of experiments in Fig. 5, which confirm our intuition: descriptors learned on DoG keypoints are clustered, while others vary significantly. Optimal values are also different for each RANSAC variant. We use the ratio test with recommended values for each feature (or a reasonable value if no recommendation exists), as there are too many outliers otherwise.

**Ratio test: One feature at a time.** Having set RANSAC, we turn to the feature matcher. Bidirectional matching with the ‘both’ strategy – the one we used so far – performs best overall. Unidirectional matching is slightly worse, and depends on the order of the images. Bidirectional matching with ‘either’ produces too many (false) matches, increasing the computational cost in the estimator, and requires very small ratio test thresholds – as low as $r=0.7$. We report these results in the appendix, and focus on the best strategy, i.e., ‘both’. We select PyRANSAC as a ‘baseline’ RANSAC and evaluate different ratio test thresholds in Fig. 6.

As expected, each feature requires different settings, as the distribution of their descriptors is different. We also observe that optimal values vary significantly between stereo and multi-view, even though one would expect that bundle

---

**Figure 4. RANSAC – Performance vs. cost.** We evaluate six RANSAC variants, using 8k SIFT features with ‘both’ matching and ratio test $r=0.8$. The inlier threshold $\eta$ and iterations limit $\Gamma$ are variables – we plot only the best $\eta$ for each method, for clarity, and set a budget of 0.5s. (dotted red line). Computed on ‘n1-standard-2’ VMs on Google Compute (2 vCPUs, 7.5 GB).

**Figure 5. RANSAC – Inlier threshold $\eta$.** We determine $\eta$ for each combination, using 8k features with ‘both’ matching (2k for LF-Net and SuperPoint) with their recommended ratio test threshold. Optimal parameters (diamonds) are listed in the appendix.

**Figure 6. Feature matching.** We evaluate bidirectional matching with the ‘both’ strategy (the optimal one), and different ratio test thresholds $r$, for each feature type. We use 8k features (2k for SuperPoint and LF-Net). For stereo, we use PyRANSAC.
adjustment would be able to better deal with outliers. We suspect this might be due to potentially sub-optimal parameters on COLMAP’s RANSAC—we will evaluate this in the future. Interestingly, D2-Net is the only method which performs best without the ratio test. Note how the ratio test is critical for performance, and one could arbitrarily select a threshold that favours one method over another, which shows the importance of proper benchmarking.

Additionally, we implement the first-geometric-inconsistent ratio threshold, or FGNN [58], but find that although it improves over unidirectional matching, its gains disappear against matching with ‘both’—see appendix.

**Choosing the number of features.** The ablation tests in this section use (up to) \( K = 8000 \) features\(^4\), a number commensurate with that used by SfM frameworks [98, 79]. Fig. 7 evaluates different values for \( K \). We use PyRANSAC with optimal settings, three matching strategies, and the recommended ratio test \( r = 0.8 \) [48]. As expected, performance is strongly correlated with the number of features. We find 8k to be a good compromise between performance and cost, and also consider 2k (actually 2048) as a ‘cheaper’ alternative—this also provides a fair comparison with some learned methods which only operate on that regime.

Stereo mAP is higher than multi-view in Figs. 6-7. This might be surprising, as stereo is typically harder—this is due to how we compute it. First, we feed the multi-view task image subsets with looser co-visibility requirements. Second, every pair of images which contains one unregistered image is classified as a failure—and we consider only the largest reconstructed model, for simplicity, marking the images on smaller models as unregistered (this is rare).

**Binary features.** Binary descriptor papers favour a distance threshold in place of the ratio test to reject non-discriminative matches [73]. We evaluate both in Fig. 8. The ratio test works better for both ORB and AKAZE.

**On the influence of the detector.** We embed several popular blob and corner detectors into our pipeline, with OpenCV’s DoG [48] as a baseline. We combine multiple methods, taking advantage of the VLFeat library: DoG, Hessian, HesLap) and corner detectors (Harris, HarLap), and MSER. **Right:** affine shape estimation for DoG and Hessian keypoints, versus the plain version. We consider a classical approach, Baumberg (Affine) [14], and the recent, learned AffNet [59]—they provide a small but inconsistent boost.

**Benchmarking detectors.** Performance on stereo, paired with SIFT descriptors. The dashed, black line indicates OpenCV SIFT. **Left:** OpenCV DoG vs. VLFeat implementation of blob detectors (DoG, Hessian, HesLap) and corner detectors (Harris, HarLap), and MSER. **Right:** affine shape estimation for DoG and Hessian keypoints, versus the plain version. We consider a classical approach, Baumberg (Affine) [14], and the recent, learned AffNet [59]—they provide a small but inconsistent boost.

---

\(^4\)2k for SuperPoint and LF-Net, as they are built for fewer keypoints.
Number of Inliers produced by RANSAC; and (NI)

Table 2. Stereo – Test set. We report: (NF) Number of Features; (NI) Number of Inliers produced by RANSAC; and (mAP(10\(^o\)))

| Method                  | NF  | RF | TL | mAP(2k) | mAP(8k) | mAP(10\(^o\)) | TL | mAP(2k) | mAP(8k) | mAP(10\(^o\)) | Rank |
|-------------------------|-----|----|----|---------|---------|---------------|----|---------|---------|---------------|------|
| CV-SIFT                 | 7879.0 | 153.9 | 416.0 | 222.7 | 468.0 | 270.6 | .4866 | 13      |
| VL-SIFT                 | 7970.0 | 166.2 | 437.0 | 241.2 | 464.3 | 301.0 | .4638 | 12      |
| VL-Hessian-SIFT         | 8000.0 | 186.5 | 491.5 | 264.0 | 448.9 | 318.0 | .4394 | 15      |
| VL-DogMerts-SIFT        | 7910.0 | 219.5 | 509.0 | 229.7 | 463.5 | 291.9 | .4624 | 8       |
| VL-HessianNet-SIFT      | 8000.0 | 190.0 | 481.0 | 208.1 | 465.9 | 328.5 | .4581 | 10      |
| CV-Sift                 | 7884.0 | 176.3 | 434.0 | 257.4 | 492.1 | 317.4 | .4981 | 6       |
| Surf                    | 7749.0 | 113.0 | 232.6 | 117.8 | 245.2 | 136.1 | .2841 | 19      |
| Akaze                   | 7878.9 | 184.3 | 513.9 | 232.7 | 314.2 | 284.4 | .3054 | 14      |
| VL-DoG/Merts-SIFT      | 7884.0 | 121.3 | 513.9 | 316.9 | 512.5 | 375.5 | .3011 | 3       |
| Key-Net/HardNet         | 7998.0 | 153.3 | 390.0 | 375.0 | 470.0 | 363.6 | .4529 | 9       |
| Geodesc                 | 7884.3 | 197.9 | 438.0 | 264.0 | 474.7 | 340.3 | .4755 | 5       |
| ContextDesc             | 4811.1 | 248.8 | 428.3 | 261.4 | 485.6 | 356.1 | .4662 | 7       |
| SOSNet                  | 7884.3 | 215.1 | 459.5 | 319.8 | 5233 | 418.5 | .5177 | 2       |
| LogPolarDesc            | 7884.3 | 243.5 | 449.5 | 366.0 | 5080 | 461.0 | .5001 | 4       |
| SuperPoint(2k)          | 1178.9 | 88.1 | 2359 | 84.7 | 2669 | 113.2 | .4638 | 18      |
| LF-Net(2k)              | 2024.8 | 95.1 | 1945 | 100.8 | 2253 | 134.2 | .2164 | 20      |
| D2-Net(S)               | 5540.7 | 273.5 | 1432 | 241.4 | 1639 | 428.0 | .1560 | 23      |
| D2-Net(MS)              | 6806.3 | 193.0 | 1690 | 322.8 | 1836 | 505.1 | .1731 | 21      |

We rank them by mAP at 10\(^o\) and AMOS [67], performs worse than the original model trained only on the ‘Liberty’ scene from Brown. Properly tuned, SIFT – RootSIFT specifically – performs within 7.4\% relative of the state of the art. Other ‘classical’ local features do not fare so well. Key.Net produces more inliers than DoG, but its poses are slightly worse – note that these are the inliers estimated by each method, which may still contain outliers. D2-net performs poorly on our benchmark, despite state-of-the-art results on others – this might be related to its incompatibility with the ratio test, resulting in too many matches, and may require a different approach.

Table 2 reports results for the multi-view task. We see similar results, with deep descriptors operating on DoG points at the top, with HardNet first. Key.Net and D2-Net are able to reconstruct more views and produce more landmarks, with lower mAP. Interestingly, the rank may change between the tasks, with ORB performing better than single-scale D2-Net on stereo but significantly worse on multi-view, despite extracting 29% more features.

**Performing on a budget.** We also consider reducing the number of features from 8k to 2k, for the stereo task. This provides a fairer comparison with LF-Net and Superpoint, which we use with their recommended settings, as despite our best efforts we could not obtain meaningful results with 8k point, as they are not trained for it. We show the results in Fig. 10. The top performing methods do not change, but the gap with traditional features is wider. Interestingly, Key.Net outperforms DoG (paired with HardNet) and tops the chart – which makes sense, as it was trained for this regime.

**Outlier pre-filtering with deep networks.** Next, we study the performance of CNe [100] for outlier rejection. Its training data does not use the ratio test, so we omit it here too. Our initial experiments, with SIFT, are encouraging: CNe aggressively filters out about 80% of the matches in a single forward pass, boosting mAP at 10\(^o\) by 3-5% relative on the stereo task across different RANSAC variants, despite problems.
In contrast with classical methods, which estimate the orientation of each keypoint, modern, end-to-end pipelines [31, 33, 71] often skip this step, assuming that the images are roughly aligned (upright), with the descriptorshouldering the increased invariance requirements. As our images meet this condition, we experiment with setting the orientation of keypoints to a fixed value (0). We list the results in Table 4. This allows us to remove keypoints with duplicate orientations, effectively increasing our feature budget and improving performance across the board – albeit by a small margin.

**Pose mAP vs. traditional metrics.** Fig. 11 shows how repeatability and matching score correlate with mAP. While the matching score results seem sensible, repeatability is harder to interpret (as explained in Section 4, our implementation differs from the standard). ContextDesc performs well despite poor repeatability, and AKAZE obtains the best repeatability but performs poorly in terms of mAP and matching score, which indicates the descriptor may hurt its performance. We compute these metrics at a 3-pixel threshold, and provide more granular results in the appendix.

**Breakdown by scene.** Results vary drastically between scenes. Please refer to the appendix for a legend.

---

### Table 3. Outlier pre-filtering with CNe – Test set.

| Method          | PyRANSAC mAP(10°) △(%) | mAP(10°) △(%) | mAP(10°) △(%) | Multi-view Task Degensac mAP(10°) △(%) | Degensac △(%) | Degensac mAP(10°) △(%) |
|-----------------|------------------------|---------------|---------------|----------------------------------------|--------------|------------------------|
| CV-SIFT         | 4200 ±2.39             | 4727 ±2.58    | 4737 ±3.26    | .4169 ±0.52                            |              |                        |
| CV-ORB          | 4280 ±1.55             | .4867 ±1.11   | .4762 ±2.63   | .4317 ±4.08                            |              |                        |
| AKAZE           | 3010 ±9.92             | .5359 ±12.66  | 3427 ±12.22   | .3738 ±10.35                           |              |                        |
| SURF            | 1620 ±17.94            | 2067 ±19.92   | 1961 ±20.16   | .2330 ±22.30                           |              |                        |
| DoG-HardNet     | 4063 ±12.96            | .4685 ±11.38  | .4579 ±11.03  | .4259 ±8.31                            |              |                        |
| KeyNet-HardNet  | 2994 ±24.97            | .3798 ±19.18  | .3733 ±17.59  | .4005 ±8.61                            |              |                        |
| GeoDesc         | 3752 ±10.32            | .4390 ±8.28   | .4323 ±9.04   | .3983 ±7.93                            |              |                        |
| ContentDesc     | 3450 ±19.44            | .4143 ±14.69  | .3952 ±15.23  | .3903 ±11.15                           |              |                        |
| SOSNet          | 3911 ±14.88            | .4589 ±12.30  | .4559 ±11.93  | .4126 ±11.28                           |              |                        |
| LogPolarDesc    | 3774 ±16.04            | .4485 ±13.68  | .3434 ±13.34  | .4032 ±12.76                           |              |                        |
| SuperPoint (2k) | 1898 ±19.55            | .2375 ±11.02  | .2256 ±13.90  | .2957 ±11.03                           |              |                        |
| LF-Net (2k)     | 1734 ±10.85            | .2197 ±2.51   | .2091 ±3.40   | .2929 ±2.5                             |              |                        |
| D2-Net (SS)     | 1058 ±26.10            | .1410 ±13.96  | .1262 ±19.10  | .1991 ±3.18                            |              |                        |
| D2-Net (MS)     | .0855 ±49.41           | .1102 ±39.99  | .1007 ±41.80  | .1673 ±21.93                           |              |                        |

Table 4. **Upright features.** We report (NI) the number of inliers and mAP at 10° for stereo, with Degensac. As DoG may return multiple orientations for the same point [48] (about 15%), we report: (top) with orientation estimation; (middle) setting the orientation to zero while removing duplicates; and (bottom) adding new points until hitting the 8000-feature budget.

---

Figure 11. **Downstream vs. traditional metrics.** We cross-reference stereo mAP at 10° with CNe, on stereo and multi-view, and its increase in performance w.r.t. Table 2 – positive △ meaning CNe helps. When using CNe, we disable the ratio test.

Figure 12. **Breakdown by scene.** Results vary drastically between scenes. Please refer to the appendix for a legend.
7. Conclusions

We introduce a comprehensive benchmark for local features and robust estimation algorithms. The modular structure of its pipeline allows to easily integrate, configure, and combine methods and heuristics. We demonstrate this by evaluating dozens of popular algorithms, from seminal works to the cutting edge of machine learning research, and show that classical solutions may still outperform the perceived state of the art with proper settings.

The experiments carried out through the benchmark and reported in this paper have already revealed unexpected, non-intuitive properties of various components of the SfM pipeline, which will benefit SfM development, e.g., the need to tune RANSAC to the particular feature detector and descriptor and to select specific settings for a particular RANSAC variant. Other surprising facts have been uncovered by our tests, such as that the optimal set-ups across different tasks (stereo and multiview) may differ, or that end-to-end methods are very sensitive to the scene.

Our work is open-sourced and makes the basis of an open challenge for image matching with sparse methods.

Acknowledgements

This work was partially supported by the Natural Sciences and Engineering Research Council of Canada (NSERC) Discovery Grant “Deep Visual Geometry Machines” (RGPIN-2018-03788), by systems supplied by Compute Canada, and by Google’s Visual Positioning Service. DM and JM were supported by OP VVV funded project CZ.02.1.01/0.0/0.0/16 019/0000765 “Research Center for Informatics”. DM was also supported by CTU student grant SGS17/185/OHK3/3T/13 and by the Austrian Ministry for Informatics”. DM and JM were supported by OP VVV funded project (RGPIN-2018-03788), by systems supplied by Compute Canada, and by Google’s Visual Positioning Service. DM and JM were supported by OP VVV funded project CZ.02.1.01/0.0/0.0/16 019/0000765 “Research Center for Informatics”. DM was also supported by CTU student grant SGS17/185/OHK3/3T/13 and by the Austrian Ministry for Transport, Innovation and Technology, the Federal Ministry for Digital and Economic Affairs, and the Province of Upper Austria in the frame of the COMET center SCCH. AM was supported by Swiss National Science Foundation.

References

[1] H. Aanaes, A. L. Dahl, and K. Steenstrup Pedersen. Interesting Interest Points. *IJCV*, 97:18–35, 2012.
[2] H. Aanaes and F. Kahl. Estimation of Deformable Structure and Motion. In *Vision and Modelling of Dynamic Scenes Workshop*, 2002.
[3] S. Agarwal, N. Snavely, I. Simon, S. M. Seitz, and R. Szeliski. Building Rome in One Day. In *ICCV*, 2009.
[4] P. F. Alcantarilla, J. Nuevo, and A. Bartoli. Fast Explicit Diffusion for Accelerated Features in Nonlinear Scale Spaces. In *BMVC*, 2013.
[5] Anonymous. DeepSFM: Structure From Motion Via Deep Bundle Adjustment. In *Submission to ICLR*, 2020.
[6] Relja Arandjelovic. Three things everyone should know to improve object retrieval. In *CVPR*, 2012.
[7] Relja Arandjelovic, Petr Gronat, Akihiko Torii, Tomas Pajdla, and Josef Sivic. NetVLAD: CNN Architecture for Weakly Supervised Place Recognition. In *CVPR*, 2016.
[8] Hernan Badino, Daniel Huber, and Takeo Kanade. The CMU Visual Localization Data Set. http://3dvis.ri.cmu.edu/data-sets/localization, 2011.
[9] V. Balntas, K. Lenc, A. Vedaldi, and K. Mikolajczyk. HPatches: A Benchmark and Evaluation of Handcrafted and Learned Local Descriptors. In *CVPR*, 2017.
[10] Vassileios Balntas, Shuda Li, and Victor Prisacariu. RelocNet: Continuous Metric Learning Relocalisation using Neural Nets. In *The European Conference on Computer Vision (ECCV)*, September 2018.
[11] V. Balntas, E. Riba, D. Ponsa, and K. Mikolajczyk. Learning Local Feature Descriptors with Triplets and Shallow Convolutional Neural Networks. In *BMVC*, 2016.
[12] Daniel Barath and Ji Matas. Graph-cut ransac. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.
[13] Axel Barroso-Laguna, Edgar Riba, Daniel Ponsa, and Kristian Mikolajczyk. Key.Net: Keypoint Detection by Hand-crafted and Learned CNN Filters. In *Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision*, 2019.
[14] A. Baumberg. Reliable Feature Matching Across Widely Separated Views. In *CVPR*, pages 774–781, 2000.
[15] H. Bay, T. Tuytelaars, and L. Van Gool. SURF: Speeded Up Robust Features. In *ECCV*, 2006.
[16] P. R. Beaudet. Rotationally invariant image operators. In *Proceedings of the 4th International Joint Conference on Pattern Recognition*, pages 579–583, Kyoto, Japan, Nov. 1978.
[17] Jia-Wang Bian, Yu-Huan Wu, Ji Zhao, Yun Liu, Le Zhang, Ming-Ming Cheng, and Ian Reid. An Evaluation of Feature Matchers for Fundamental Matrix Estimation. In *BMVC*, 2019.
[18] Eric Brachmann and Carsten Rother. Neural-Guided RANSAC: Learning Where to Sample Model Hypotheses. In *ICCV*, 2019.
[19] G. Bradski. The OpenCV Library. Dr. Dobb’s Journal of Software Tools, 2000.
[20] M. Brown, G. Hua, and S. Winder. Discriminative Learning of Local Image Descriptors. *PAMI*, 2011.
[21] M. Brown and D. Lowe. Automatic Panoramic Image Stitching Using Invariant Features. *IJCV*, 74:59–73, 2007.
[22] Mai Bui, Christoph Baur, Nassir Navab, Slobodan Ilic, and Adrien Bartoli. Fast Explicit Diffusion for Accelerated Features in Non-linear Scale Spaces. In *BMVC*, 2013.
[23] Ondrej Chum, Jiří Matas, and Josef Kittler. Locally Optimized RANSAC. In *CVPR*, 2003.
[24] Ondřej Chum, Jiří Matas, and Josef Kittler. Locally Optimized ransac. In *Pattern Recognition*, 2003.
[97] X. Wei, Y. Zhang, Y. Gong, and N. Zheng. Kernelized Subspace Pooling for Deep Local Descriptors. In CVPR, 2018.

[98] Changchang Wu. Towards Linear-Time Incremental Structure from Motion. In 3DV, 2013.

[99] Kwang Moo Yi, Eduard Trulls, Vincent Lepetit, and Pascal Fua. LIFT: Learned Invariant Feature Transform. In ECCV, 2016.

[100] K. M. Yi, E. Trulls, Y. Ono, V. Lepetit, M. Salzmann, and P. Fua. Learning to Find Good Correspondences. In CVPR, 2018.

[101] S. Zagoruyko and N. Komodakis. Learning to Compare Image Patches via Convolutional Neural Networks. In CVPR, 2015.

[102] Jiahui Zhang, Dawei Sun, Zixin Luo, Anbang Yao, Lei Zhou, Tianwei Shen, Yurong Chen, Long Quan, and Hongen Liao. Learning Two-View Correspondences and Geometry Using Order-Aware Network. ICCV, 2019.

[103] Xu Zhang, Felix X. Yu, Svebor Karaman, and Shih-Fu Chang. Learning Discriminative and Transformation Covariant Local Feature Detectors. In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), July 2017.

[104] Chen Zhao, Zhiguo Cao, Chi Li, Xin Li, and Jiaqi Yang. NM-Net: Mining Reliable Neighbors for Robust Feature Correspondences. In CVPR, 2019.

[105] Qunjie Zhou, Torsten Sattler, Marc Pollefeys, and Laura Leal-Taixe. To learn or not to learn: Visual localization from essential matrices. arXiv Preprint, 2019.

[106] Siyu Zhu, Runze Zhang, Lei Zhou, Tianwei Shen, Tian Fang, Ping Tan, and Long Quan. Very Large-Scale Global SfM by Distributed Motion Averaging. In CVPR, June 2018.

[107] C.L. Zitnick and K. Ramnath. Edge Foci Interest Points. In ICCV, 2011.
8. Appendix

In this section we provide details about our data, experiments, and qualitative results that were omitted from the main paper due to space constraints. All results are on the validation set, unless stated otherwise.

8.1. Dataset details

We provide over 25k images for training, including 2D/3D points, camera poses, and dense depth estimates. We use ‘sacre_coeur’ and ‘st_peters_square’ for the validation and ablation test experiments of Section 5. We generate the validation data following the same procedure we used for the test data, outlined in Section 3. These two subsets will also be released, so that the validation results are reproducible and comparable. Additionally, We use ‘Notre Dame Front Facade’ and ‘Buckingham Palace’ to retrain CNe [100], which was originally trained on one of our validation sequences, as outlined in Section 4. The rest of the ‘training’ sequences are not used in this paper. We test the new models on the test set and observe that performance improves.

The test scenes as listed in Table 5, along with the acronym used in Fig. 12. Note that we only release 100 images for each, to keep the load manageable, as every possible combination of two images needs to be considered – the ground truth, which will remain behind a public evaluation server, is computed with the full image set. Links for both will be provided after the review phase.

8.2. Co-visibility metric example

In Section 3 we introduce how we compute a simple co-visibility measure to determine how much content is shared between two images, by extracting the 3D keypoints in common across both images, projecting them into the camera plane, and computing their bounding box – Fig. 13 shows an example. We also plot the co-visibility histograms for every scene in Fig. 14. This value gives an indication of how ‘hard’ each scene is, accounting by changes in scale, but not necessarily large-scale occlusions – notice, for instance, how USC (‘United States Capitol’) is one of the hardest scenes in the breakdown in Fig. 12. For stereo, we set the minimum co-visibility threshold to 0.1.

8.3. Breakdown by co-visibility threshold

Here we show results for stereo at different co-visibility thresholds, for different features in Fig. 15, and for different RANSAC methods in Fig. 16. Performance for all local feature and RANSAC variants increases about 20% (absolute) as the co-visibility threshold increases from o.1 to 0.6, where SuperPoint benefits the most (a 60% relative performance boost). By contrast, we do not observe significant variations across different RANSAC methods, with DEGENSAC and MAGSAC showing very similar performance. Note that at co-visibility 0 some image pairs may not be matchable, and that results at high co-visibility thresholds are noisy as we have few samples for them – we

Table 5. Dataset – Training and validation.

| Name                        | Images | 3D points |
|-----------------------------|--------|-----------|
| brandenburg_gate            | 1363   | 100040    |
| buckingham_palace           | 1676   | 234052    |
| colosseum_exterior          | 2063   | 259807    |
| grand_place_brussels        | 1083   | 229788    |
| hagia_sophia_interior       | 888    | 235541    |
| notre_dame_front_facade     | 3765   | 488895    |
| palace_of_westminster       | 983    | 115868    |
| pantheon_exterior           | 1401   | 166923    |
| prague_old_town_square      | 2316   | 558600    |
| sacre_coeur (SC)            | 1179   | 140659    |
| st_peters_square (SPS)      | 2504   | 232329    |
| taj_mahal                   | 1312   | 94121     |
| temple_nara_japan           | 904    | 92131     |
| trevi_fountain              | 3191   | 580673    |
| westminster_abbey            | 1061   | 198222    |
| **Total**                   | **25.6k** | **3.7M**  |

Table 6. Dataset – Test.

| Name                        | Images | 3D points |
|-----------------------------|--------|-----------|
| british_museum (BM)         | 660    | 73569     |
| florence_cathedral_side (FCS)| 108   | 44143     |
| lincoln_memorial_statue (LMS)| 850  | 58661     |
| london_bridge (LB)          | 629    | 72235     |
| milan_cathedral (MC)        | 124    | 33905     |
| mount_rushmore (MR)         | 138    | 45350     |
| piazza_san_marco (PSM)      | 249    | 95895     |
| reichstag (RS)              | 75     | 17823     |
| sagrada_familia (SF)        | 401    | 120723    |
| st_pauls_cathedral (SPC)    | 615    | 98872     |
| united_states_capitol (USC)| 258    | 35095     |
| **Total**                   | **4107** | **696k**  |

Figure 13. Co-visibility computation example. We project the 3D points obtained with COLMAP into two images, and color them green if they are seen from both views, and red otherwise. The ‘visibility’ value is the ratio between the area of the bounding box of the shared points, and that of the image. The ‘co-visibility’ value for these two images is the smallest of the two.
do not plot co-visibility values above 0.6 as these bins are very sparsely populated.

8.4. Alternative matching strategies

As discussed in the main paper, the best matching strategy is bidirectional matching with ‘both’, and the only one we consider for the vast majority of the experiments. Fig. 7 (in the main paper) shows that performance is slightly worse with unidirectional matching. In this section we evaluate the ‘either’ strategy as an alternative – results are shown in Fig. 17, which shows that performance is typically worse. It also requires more strict (lower) ratio test thresholds in order to restrict the number of matches, which is noticeably higher, and slows RANSAC down very significantly, even for the same limit of iterations.

8.5. Number of inliers per step

We report the number of input matches and their resulting inliers for the stereo task in Table 7. We list: the number of input matches; the number of inliers produced by each method (they may still contain outliers); their ratio; and the mAP at 10°. We use PyRANSAC with optimal settings for
each method, including the ratio test, using bidirectional matching with the ‘both’ strategy. We see that inlier-to-outlier ratios hover around 40% for all feature types except D2-Net, which seems adequate for RANSAC, and might explain why D2-Net, which is not amenable to the ratio test, performs poorly on our benchmark.

### 8.6. Image pre-processing

Contrast normalization is key to invariance against illumination changes – local feature methods typically apply some normalization strategy over small patches. Therefore, we experiment with contrast-limited adaptive histogram quantization (CLAHE) [66], as implemented in OpenCV. We apply it to SIFT and to several learned descriptors, and display the results in Fig. 18. Performance decreases for all learned methods, presumably because they are not trained for it. Contrary to our initial expectations, SIFT does not benefit much from it: the only significant increase in performance comes from applying it for descriptor extraction, at 2.4% relative for stereo and 1.9% relative for multi-view. This might be due to the small number of night-time images in our data. It also falls in line with the observations of J. Dong et al. in ‘Multi-View Feature Engineering and Learning’, CVPR 2015, which show that SIFT descriptors are actually optimal under certain assumptions.

### 8.7. Optimal settings breakdown

We summarize the optimal hyperparameter combinations from Figs. 4, 5 and 6, for clarity, in Table 8. We set the confidence value to $\tau=0.999999$ for all RANSAC variants.

### 8.8. Feature matching with advanced ratio test

We also compared the benefits of applying first-geometrically-inconsistent-neighbor-ratio (FGINN) [58] to DoG/SIFT, DoG/HardNet and Key.Net/HardNet, against Lowe’s standard ratio test rule [48]. FGINN performs the ratio-test with second-nearest neighbors that are “far enough” from the putative match (10 pixels in [58]). In other words, it loosens the test to allow for nearby-thus-similar points. We test it for 3 matching strategies: unidirectional (‘uni’), ‘both’ and ‘either’. We report the results in Fig. 19. As shown, FGINN provides minor improvements over the standard ratio test in case of unidirectional matching, and not as much when ‘both’ is used. It also behaves differently compared to the standard strategy, in that performance at stricter thresholds degrades less.

### 8.9. Finding the optimal support region

As outlined in Section 6, we find that the recommended support region used to extract deep descriptors is already optimal, or near so. We plot these results in Fig. 20. Interestingly, SIFT descriptors do benefit from increasing the scaling factor from 12 to 16, but the increase in performance is small. This suggests that deep descriptors such as HardNet might also be able to increase performance slightly training on larger patches.
8.10. Breakdown by (pixel) error threshold

In Fig. 11 in the main paper we plot repeatability and matching score at a fixed error threshold of 3 pixels. In Fig. 21 we show them at different pixel thresholds. Note that repeatability is typically lower than matching score, whereas the matching score is computed with optimal matching settings (bidirectional matching with the ‘both’ strategy and ratio test), which results in a much smaller pool, from about 8000 keypoints to 150-600 matches (see Table 7). This better isolates the performance of the detector and the descriptor, where it matters.
8.11. Breakdown by (angular) error threshold

We summarize pose accuracy by mAP at 10° (optionally, at 5°), in order to have a single number. In this section we show how performance varies across different error thresholds. Fig. 22 shows how it affects different local feature types, and Fig. 23 shows how it affects different RANSAC variants, with SIFT. Better features perform better at any threshold, at least up to 10°. The same applies to differences between RANSAC methods. Differences between methods are more pronounced for multi-view.

8.12. Outlier pre-filtering on the validation set

Due to space constraints, we do not include results on the validation set with CNe from the paper. We list them in Table 9, for SIFT with 8000 features, bidirectional matching with the ‘both’ strategy, and optimal ratio test thresholds. Without CNe we use optimal ratio test thresholds; with CNe we do not use the ratio test, as it was not trained for it and performs much worse in that case. Note that there are no parameters to set, which is why we report results directly on the test set in the main paper.

Additionally, we noticed one omission from the paper after the deadline: CNe was originally trained to estimate the Essential matrix instead of the Fundamental matrix [100], i.e., assuming known intrinsics. In order to use it within our setup, we simply normalize by the size of the image instead of using ground truth calibration matrices.

8.13. Qualitative results

Fig. 24 shows qualitative results for stereo task – we draw the inliers produced by each method as in Fig. 1 in the main paper. The pair of images in the middle row have a very large baseline, and only HardNet is able to establish correct matches. Fig. 25 shows multiview results, displaying the importance of the detector, where D2-Net and AKAZE pick up more noise on the sky.
Figure 24. **Qualitative results for the stereo task** – Among the predicted matches using each method, we display the inliers in green, and the outliers in red.
Figure 25. **Qualitative results for the multi-view task** – We show images and detected keypoints for each method, with the 3D points identified by COLMAP in blue, and the ones that are not used in the 3D model as red. More blue points mean that more points are useful to reconstruct a 3D model. These results correspond to the ones where we use 25 images to reconstruct a 3D model.