Evaluation of Local Detectors and Descriptors for Fast Feature Matching

Ondrej Miksik
CMP, Czech Republic
ondra.miksik@gmail.com

Krystian Mikolajczyk
CVSSP, UK
k.mikolajczyk@surrey.ac.uk

Abstract

Local feature detectors and descriptors are widely used in many computer vision applications and various methods have been proposed during the past decade. There have been a number of evaluations focused on various aspects of local features, matching accuracy in particular, however there has been no comparisons considering the accuracy and speed trade-offs of recent extractors such as BRIEF, BRISK, ORB, MRRID, MROGH and LIOP. This paper provides a performance evaluation of recent feature detectors and compares their matching precision and speed in randomized kd-trees setup as well as an evaluation of binary descriptors with efficient computation of Hamming distance.

1 Introduction

One of the most influencing papers in computer vision is the seminal work on SIFT [8]. Although the SIFT keypoint detector and descriptor have stood the test of time and have been widely used in various applications including panorama stitching, object recognition, image retrieval or visual navigation, the high dimensional descriptor suffers from a computational complexity, which makes it unsuitable for time-constrained applications e.g. SLAM, object tracking, real-time recognition.

During the past decade, a variety of improvements were proposed [3, 14, 5, 7, 15, 17]. Such descriptors were reported to improve the efficiency and matching accuracy upon SIFT however, in the literature there is no quantitative comparison of the publically available implementations (e.g. OpenCV) tested within the same experimental framework. The aim of this work is to provide an independent evaluation on common data and guidance in chosing amongst the recent implementations when speed and accuracy of matching are the main criteria.

Detectors and descriptors. Efficient computation of features similar to SIFT is proposed by SURF [3], which approximates Hessian matrix and gradients by a set of box-type filters and integral images. A different idea is proposed by MSER [10] where the regions are extracted with efficient watershed segmentation.

A very efficient keypoint detector is FAST [14] and its variants, which compare pixels on a ring centered at a feature point. Recently proposed ORB [15] detector extends FAST by efficiently computed orientations based on the intensity centroid moment. Similarly, BRISK [7] extends FAST by searching for maxima in a 3D scale-space.

Rotation invariance is also investigated in MRRID and MROGH descriptors [6] by pooling local features based on their intensity orders in multiple support regions. This concept is further exploited in LIOP [17] together with segmentation based location grid in contrast to 4×4 regular grid in SIFT or SURF.

A different approach is to use binary descriptors [5, 7, 15]. In contrast to SIFT and SURF and expensive computing of gradient distributions, a set of simple binary tests are used together with the Hamming distance which is computationally cheap on modern architectures with efficient binary *xor* and *population count* instructions. An example of such descriptor is BRIEF [5], which is a binary string constructed from a set of intensity comparisons within an image patch.

Efficient data structures. Much research is focused on improving the efficiency of feature matching via nearest neighbour (NN) search. Widely used algorithms in computer vision applications are kd-trees [1, 9, 13] and hashing methods [16]. Hashing is a fast NN search approach that relies on projection functions that map similar data points into the same buckets that can be efficiently accessed in Hamming space. The kd-tree belongs to a category of a geometric data structures, which is based on iterative partitioning of individual dimensions. Its issues with high dimensional nearest neighbor searching may be overcome by an *ϵ-approximate* nearest neighbor (ϵ-ANN) search [2], where search is

This research was supported by the Czech Science Foundation grant GACR P103/12/G084 (OM) and by EU CHIST-ERA and UK EPSRC funding (KM).

terminated if a certain condition is satisfied e.g. maximum number of leaves visited or a termination parameter which guarantees that the distance to ANN found so far is smaller than distance to the true NN multiplied by a constant i.e. $(\epsilon + 1)d_{NN} \geq d_{ANN}$.

The remainder of this paper is organized as follows: evaluation criteria and overview of datasets are given in section 2, experimental results are discussed in 3 and the paper is concluded in section 4.

2 Evaluation framework

We follow the evaluation protocol from [12], which is based on a number of correct and false matches obtained for a given image pair. However, we evaluate the matching performance in a database of features that also contain distractors, that is features from different images. We use the repeatability score, precision-recall as well as a speedup factor which are discussed below.

Repeatability score. The descriptor correspondences are established by using the groundtruth homography as in [12]. The repeatability score is $S_A = A^+/A^*$, with A^+ the number of correspondences and A^* the number of all detected features in a query image, which projected by a homography are within the reference image. A^+ is the number of keypoints for which the overlap with the projected features A^* is more than 50%. [12].

Precision-recall. Precision $S_B = B^*/B$, is a ratio between the number of correct matches B^* and NN features B found in the database. B is obtained by thresholding the NN distance between the query feature F_q and the feature found in a database F_d by a factor t_b. The number of correct matches B^*, is obtained by verifying whether the matched features are also the correspondences provided by the groundtruth homographies. Recall $S_{BA} = B^*/A^+$ denotes a ratio between the correct matches B^* and correspondences A^+.

Speed-up. Speed-up factor $S_T = T_S/T_E$ is used to measure the speed efficiency rather than the absolute time, where T_S is the time for sequential search, T_E is the time for efficient matching.

Dataset. The experiments are carried out with Graffiti sequences [12] which consist of eight image sequences with different geometric and photometric transformations including rotation, scale change, viewpoint change, image blur, jpeg compression and change in illumination. Each image sequence is composed by six images and the ground truth homographies between the first image and the rest of images in the sequence. The features from the reference image of every sequence are used to create the database. In addition, we use features from 1000 images from Pascal VOC as matching distractors. The features detected in all other images from the Graffiti dataset are used as respective queries for matching experiment. There are 132 687 query features and 786 968 database features. Althought it is not a large scale database it is sufficient to reliably compare the speed and matching accuracy of state-of-the-art descriptors.

Implementation details. OpenCV [4] is used for extraction of all keypoints except BRISK, LIOP, MROGH and MRRID, which are recent descriptors, hence the binaries from authors are used. The two most extensively used implementations of kd-trees are ANN [2] and FLANN [13]. The ANN library is used for all experiments.

All the measurements have been taken on a 24 ×3.47GHz/12MB cache Intel Xeon X5690, 99GB RAM, x64 Ubuntu 10.04, the code is compiled by the GCC 4.4.3 with SSE 4.2 optimization for fast `POPCNT`.

3 Experimental results

The performance evaluation of recent detectors and descriptor is presented in this section. First, the speed and quantity of features extracted by various methods are discussed. Next, we investigate the accuracy and speed of fast matching by multiple randomized kd-trees. Finally, the descriptors in kd-tree setup are compared against the efficient Hamming distance based matching of the binary descriptors. All the features are obtained with default parameters given by their implementation except FAST for which the threshold was increased to reduce the number of features. Scales of multiscale FAST keypoints are increased by a factor of 2 otherwise the regions from the finest scale level are too small i.e. 10x10 pixels.

3.1 Feature extraction comparison

The extraction times and quantities of keypoints and descriptors are compared in this section. All results are computed on a set of 100 images from Graffiti and Pascal VOC data sets. Table 1 shows the averaged results.

The largest number of keypoints are extracted by multiscale FAST detector and the least number of regions is provided by MSER. The variation in the number of features is expected, since the various detectors respond to different types of image structures. This can be controlled to a small extent by parameter settings but the order of numbers remains the same. The most efficient extractor is FAST which is 88× faster than SURF and 169× faster than DoG. On average it takes 2ms per image.

Similar experiment is done for the calculation of descriptors. To make the comparison fair we use the same number of keypoints extracted by SURF. The results are summarized in table 2. The fastest descriptor is BRIEF

Table 1. Averaged computation times for the different detectors

Detector	Run time [ms.]	Speed-up [-]	# keypoints
SURF	176	1.9	2 911
DoG	338	1.0	1 552
FAST	**2**	169.0	**5 158**
STAR	17	19.9	849
MSER	60	5.6	483
BRISK	**10**	33.8	1 874
ORB	**7**	48.3	594

(32 bytes), followed by ORB (32 bytes) and BRISK (64 bytes). Simple binary tests can be performed up to $31\times$ or $118\times$ faster than SURF or SIFT, respectively. In general, the extraction time scales approximately linearly with increasing number of features.

The accuracy of detectors is measured by the repeatability scores as a function of overlap error and presented in Fig 1. (a). The best results are obtained by FAST which produces the largest number of keypoints and SURF. Note that for DoG we also report the result for the original implementation (ODoG) which gives better repeatability than the OpenCV one (DoG).

3.2 Efficient matching

In this section the matching quality and speed-up is investigated as a function of various parameter settings. A widely used approach for fast matching of real valued descriptors is by multiple randomized kd-trees [9, 11]. The descriptors are computed for keypoints extracted with SURF. We build N sets of descriptors by random sampling and build a kd-tree for each set. The matching is done by running the query features through all trees simultaneously. The main trade-off in the kd-tree based matching is between the speed and the NN approximation which can be controlled by parameter ϵ. The matching performance as a function of the number of trees, NN approximation, and similarity threshold is presented in Fig. 1 (b)(c)(d).

It should be noted that the curves do not intersect each other. For various numbers of trees N, the precision is comparable while the accuracy loss against exhaustive sequential search is approximately 2.5% (Fig. 1 b). Larger differences are in recall (Fig. 1 c); the smallest loss of 4% is achieved by $N = 160$ while the matching is almost 14 890× faster (Fig. 1 d) due to parallelization. For $N = 20$, the loss in recall is 5% and speed-up is 1 719.

3.3 Comparison of descriptors

This section gives a comparison of well established descriptors such as SIFT and SURF against recently proposed LIOP, MRRID and MROGH. Moreover, all the real valued descriptors are compared to recent binary features such as BRIEF, BRISK and ORB. All descriptors are computed for the SURF keypoints.

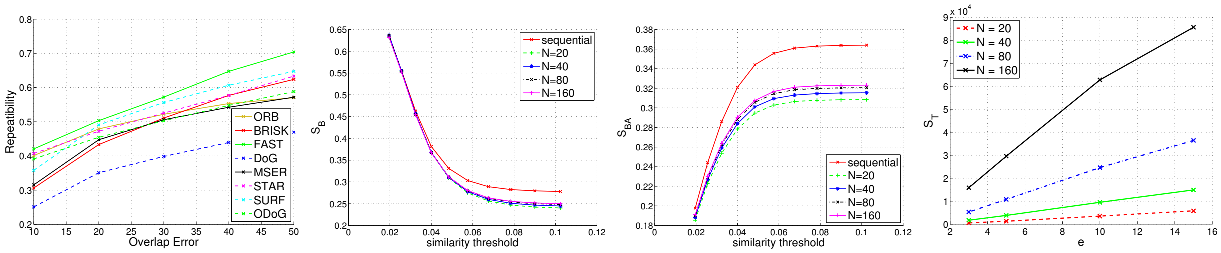
Table 2. Computation times for the different descriptors for 1000 SURF keypoints

Descriptor	Run time [ms.]	Speed-up [-]
SURF	117.1	3.83
SIFT	448.6	1.00
BRIEF	**3.8**	118.05
BRISK	**10.6**	42.32
ORB	**4.2**	106.80
LIOP	1 801.1	0.25
MROGH	2 976.8	0.15
MRRID	5 625.1	0.08

Real valued descriptors are evaluated in the multiple randomized kd-tree setup. Binary descriptors are matched in a brute force manner with efficient evaluation of the Hamming distance. To give a fair comparison of binary descriptors speed-up, which can also be improved by parallelization, matching time is divided by the number of multiple trees.

Table 3 summarizes the speed-up against the sequential search of the SURF descriptor. All results are obtained with 40 trees and $\epsilon = 3$, which represent a reasonable trade-off between the precision and time of matching with respect to the number of features in the database and descriptor size. The fastest matching is achieved by BRIEF and ORB descriptors, which are binary features with 32 bits only. The matching of binary descriptors is approximately as fast as kd-trees with SURF and $5.6\times$ faster than SIFT. One can argue, that the results would be more favorable for SIFT and SURF for larger values of ϵ, however the precision/recall would be much lower. The efficient matching of the binary descriptors has linear computational complexity, i.e. the results would be more favourable for kd-trees in the case of large databases. On the other hand, it is very cheap to extend the feature space of binary descriptors (e.g. 512 bits BRISK and 256 bits BRIEF/ORB), while the matching and memory requirements of high-dimensional descriptors in the kd-setup are much more costly.

Table 4 summarizes precision(Sb)/recall(Sba) for all descriptors. Since we compare descriptors with various numbers of dimensions, distance, etc., to avoid clutter we select saturation point s on the performance curves (c.f. Fig 1. (b)(c)) to report the matching performance. Both, SURF and SIFT are outperformed by most descriptors. Better results are reached by LIOP, MROGH and MRRID, while LIOP matching is approximately $4.4\times$ faster than MROGH and $2.1\times$ faster than MRRID. ORB is outperformed by BRIEF, however this might be explained by the fact, that only one test sequence contains in-plane rotation. Although the repeatability scores for FAST are high we believe it is due to the large number of regions which accidentally overlap, since its matching performance when combined with SIFT is very low.



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

Figure 1. Evaluation results for multiple randomized kd-trees and SURF descriptors, with N trees and NN approximations $\epsilon = 5$: a) Correspondence score S_A w.r.t overlap error, b) Matching precision S_B, c) Matching recall S_{BA}, d) Matching speed-up S_T w.r.t. sequential matching.

Table 3. Speed-up over the sequential matching of SURF-SURF for the different descriptors and $N = 40$, $\epsilon = 3$

Descriptor	Size [bytes]	Time [ms.]	Speed-up (S_T)
SURF	64	390	859.4
SIFT	128	2 095	160.1
BRIEF	32	**370**	**905.9**
BRISK	64	524	640.1
ORB	32	**370**	**905.9**
LIOP	144	3 466	96.8
MROGH	192	15 317	21.9
MRRID	256	7 363	45.6

4 Conclusion

This paper reports the matching precision and speed of recent feature detectors and descriptors in multiple randomized kd-tree setup and Hamming distance brute-force search. The recently proposed real valued descriptors such as LIOP, MRRID, MROGH outperform state-of-the-art descriptors SURF and SIFT in both, precision and recall, although their efficiency is very low. Similarly, binary descriptors provide a very efficient technique for time-constrained applications with good matching accuracy. Another advantages of binary descriptors are very fast extraction times and very low memory requirements, since each descriptor has only 32 (BRIEF, ORB) or 64 (BRISK) bytes, respectively. These descriptors provide comparable precision/recall results with SURF and SIFT.

References

[1] S. Arya and D. M. Mount. Approximate nearest neighbor queries in fixed dimensions. In *SODA '93: ACM-SIAM Symposium on Discrete algorithms*, 1993.

[2] S. Arya, D. M. Mount, N. S. Netanyahu, R. Silverman, and A. Y. Wu. An optimal algorithm for approximate nearest neighbor searching fixed dimensions. *J. ACM*, 45(6):891–923, 1998.

[3] H. Bay, A. Ess, T. Tuytelaars, and L. J. V. Gool. Speeded-up robust features (surf). *CVIU*, 110(3), 2008.

[4] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

Table 4. Precision/Recall for the different descriptors and $N = 40$, $\epsilon = 3$

Detector	Descriptor	Precision	Recall	MAP
SURF	SURF	0.485	0.513	0.334
SURF	SIFT	0.525	0.533	0.491
SURF	BRIEF	0.517	0.546	0.514
SURF	ORB	0.448	0.470	0.437
SURF	LIOP	0.581	0.597	0.568
SURF	MROGH	0.540	0.567	0.527
SURF	MRRID	0.550	0.569	0.510
SURF	BRISK	0.536	0.553	0.530
BRISK	BRISK	0.504	0.527	0.492
ORB	ORB	0.493	0.495	0.463
FAST	SIFT	0.366	0.376	0.336

[5] M. Calonder, V. Lepetit, C. Strecha, and P. Fua. BRIEF: Binary Robust Independent Elementary Features. In *ECCV*, 2010.

[6] B. Fan, F. Wu, and Z. Hu. Rotationally invariant descriptors using intensity order pooling. *T-PAMI*, 2011.

[7] S. Leutenegger, M. Chli, and R. Siegwart. Brisk: Binary robust invariant scalable keypoints. In *ICCV*, 2011.

[8] D. G. Lowe. Object recognition from local scale-invariant features. In *ICCV*, pages 1150–1157, 1999.

[9] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, Nov. 2004.

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

[11] K. Mikolajczyk and J. Matas. Improving sift for fast tree matching by optimal linear projection. In *ICCV*, 2007.

[12] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. *IEEE T-PAMI*, 27(10), 2005.

[13] M. Muja and D. G. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. In *In VISAPP ICCVTA*, pages 331–340, 2009.

[14] E. Rosten and T. Drummond. Machine learning for high-speed corner detection. In *In ECCV*, 2006.

[15] E. Rublee, V. Rabaud, K. Konolige, and G. R. Bradski. Orb: An efficient alternative to sift or surf. In *ICCV*, 2011.

[16] J. Wang, S. Kumar, and S.-F. Chang. Semi-supervised hashing for scalable image retrieval. In *CVPR*, 2010.

[17] Z. Wang, B. Fan, and F. Wu. Local intensity order pattern for feature description. In *ICCV*, 2011.