Predicting Matchability

Wilfried Hartmann Michal Havlena Konrad Schindler
Institute of Geodesy and Photogrammetry, ETH Zürich, Switzerland
firstname.lastname@geod.baug.ethz.ch

Abstract

The initial steps of many computer vision algorithms are interest point extraction and matching. In larger image sets the pairwise matching of interest point descriptors between images is an important bottleneck. For each descriptor in one image the (approximate) nearest neighbor in the other one has to be found and checked against the second-nearest neighbor to ensure the correspondence is unambiguous. Here, we asked the question how to best decimate the list of interest points without losing matches, i.e. *we aim to speed up matching by filtering out, in advance, those points which would not survive the matching stage. It turns out that the best filtering criterion is* not *the response of the interest point detector, which in fact is not surprising: the goal of detection are repeatable and well-localized points, whereas the objective of the selection are points whose descriptors can be matched successfully. We show that one can in fact learn to predict which descriptors are matchable, and thus reduce the number of interest points significantly without losing too many matches. We show that this strategy, as simple as it is, greatly improves the matching success with the same number of points per image. Moreover, we embed the prediction in a state-of-the-art Structure-from-Motion pipeline and demonstrate that it also outperforms other selection methods at system level.*

1. Introduction

Matching interest points between different images is a fundamental operation of computer vision. Matches—*i.e.* (putative) projections of the same 3D point—form the basis of state-of-the-art methods for image registration [12], Structure-from-Motion (SfM) computation [15, 9], and feature-based tracking [24], and support geometric verification during object detection and image retrieval [26, 21].

The elementary computational building blocks to find matching points in two images are simple: after independently finding distinctive points in both images with an interest point detector (*e.g.* Harris, Difference-of-Gaussians), the detected points are encoded with descriptors based on their neighborhood (*e.g.* raw intensities, SIFT, SURF, *etc.*). Then, for each point from the *source* image one searches the best match among the points from the *target* image, *i.e.* the nearest neighbor in the (usually high-dimensional) descriptor space. Finally, the set of putative correspondences thus found is verified by robustly fitting a suitable geometric transformation, usually with RANSAC-type sampling methods, and discarding matches which do not support the consensus transformation ("outliers").

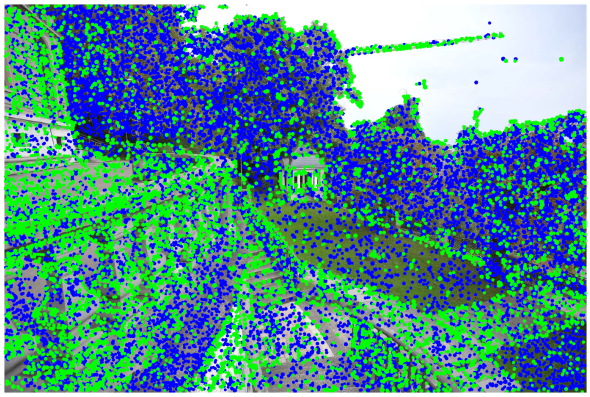


Figure 1: By looking at their descriptors, a classifier can predict which points are suitable for subsequent matching (green) and which are not (blue). Non-matchable points, *e.g.* those detected on vegetation, can then be discarded without compromising later processing steps.

The strategy suffers from two main problems. On the one hand, in practice, a large portion of the interest points detected in the images cannot be matched reliably because multiple candidates with comparable distances exist. One thus only has the choice between rejecting all ambiguous matches, or keeping all of them. The former option, championed by Lowe [21], is more widely used: one finds not only the best matching candidate, but also the second-best one, and whenever the two descriptors match similarly well (*i.e.* the ratio of the two distances is near 1) the candidate is rejected. An alternative approach tries to find mutually nearest neighbors, which doubles the computational effort,

but empirically does not greatly improve the matching.[1]

On the other hand, matching is computationally expensive: for n interest points in each image an exact implementation needs $\mathcal{O}(n^2)$ comparisons; and even inexact solutions based on either approximate nearest neighbor (ANN) search in space-partitioning trees [3] or on locality-sensitive hashing techniques [11] always have superlinear complexity. In large-scale applications such as unordered SfM or retrieval, one is thus left with two complementary options: either reduce the number m of images that are pairwise matched [1, 16, 10], at the cost of completely losing some matchable image pairs; or reduce the number n of interest points to be matched per image, at the cost of losing some correct matches.

In the present work we address the second option. The question we ask is *can we predict which descriptors are matchable already before the matching stage*, thereby reducing the number of interest points without hurting the later processing stages?

The starting point is the observation that at the stage of interest point detection one aims for points that are well localized and repeatable, but *interest point detection does not explicitly search for points that can later be matched successfully*. In fact, it is known from experience that certain strong interest points (*e.g*. those on vegetation, or on the road surface) are rarely matched successfully, see Fig. 1.

We thus propose to learn a classifier that predicts which descriptors will have a high chance of finding a match, and reduce the number of points in such a way that mainly useless points are discarded, and the success rate of the matching is improved. Note that the computational cost of the proposed prediction is linear in the number of interest points, and negligible compared to the actual construction of the descriptor vectors, and hence can significantly cut down computation time.

We will show that this simple strategy works surprisingly well and substantially speeds up the matching step, which is the main bottleneck in applications where many image pairs need to be matched, without sacrificing accuracy.

2. Related Work

Interest point detection has been a classic problem of computer vision for many years, see [30] for an overview. The most successful and most widespread methods are based on first [14] or second [5] derivatives of the image brightness, respectively fast approximations of the second derivative with the Difference-of-Gaussians method [21] or even with box filters [4]. Recently there have also been attempts to speed up interest point detection with machine learning methods, by learning computationally very cheap detectors based on direct grayvalue comparisons [23]. Our work is fundamentally different in that we aim to speed up *matching* rather than *detection* with the help of statistical learning. Importantly, the design of interest point detectors aims for points which are repeatable (*i.e*. the same point is found independently in different views of the same scene) and which can be accurately localized. These criteria by themselves do however not guarantee that a point's neighborhood is also suitable for matching, *i.e*. from a learning perspective they do not maximize the real objective.

To find matches between interest points from different images, the points are encoded with invariant descriptors. The state-of-the-art are descriptors based on the local gradient distribution, most notably SIFT [21] and SURF [4]. It has been proposed to use more compact descriptors [18], and to make descriptors binary, such that their comparison is computationally efficient [8]. Although such methods can significantly speed up individual comparisons, matching large sets of points is still computationally demanding. Naive comparison between two sets of n interest points would have complexity $\mathcal{O}(n^2)$. A standard solution is to use approximate nearest neighbor (ANN) search [3]. To that end the descriptors from one image are organized in a KD-tree (or a similar space-partitioning tree), and the tree is queried with the descriptors of the other image. Since exact NN through space partitioning loses its efficiency in high dimensions, only an approximate search is used in practice: rather than finding a strict nearest (and if needed second-nearest) neighbor, the search is stopped after visiting a fixed number t of points. ANN can be further improved by randomizing the splits to some degree and constructing multiple KD-trees in the spirit of random forests, which can then be queried in parallel [25]. By stopping after a fixed number of tree look-ups, ANN reduces the complexity to $\mathcal{O}(n \log n)$ per image pair (plus $\mathcal{O}(n \log n)$ per image to build the trees).

Still, even with ANN it is infeasible to match all $\frac{1}{2}m(m-1)$ possible image pairs in an unordered image set, thus researchers have proposed different ways to bring down the number of image pairs. Instead of exhaustive pairwise matching [1] focuses on a small number ($\approx$10) of most promising candidates, found via shared occurrence of visual words [26]. The method [16] builds on the same principles to achieve efficient reconstruction from unordered image sets by favoring promising candidates in the SfM process. Another possibility is to exploit the redundancy in the image set and reduce the number of images to a representative subset. In [20, 10] input images are clustered using the GIST [22] descriptor, and only one representative per cluster is passed on to the matching procedure. Alternatively, the reduction to a representative subset can also be formulated as a search for a connected dominating set [13]

[1] Alternatively, one could accept *all* point pairs with sufficiently high similarity as candidate matches for further processing. Since at most one of them can be correct this greatly increases the proportion of outliers, hence it is rarely done.

of an image graph with edges weighted according to the co-occurring visual words [17].

To summarize, fast nearest neighbor search reduces the complexity of matching a fixed number n of points, whereas pre-selection of promising image pairs reduces the number m of images that need to be matched. Complementary to both strategies, our work aims to reduce the number of features n per image that need to be considered, without degrading the reliability of the matching.

A problem related to the one in this paper is treated in [19] where also image points are removed. In principle these should be points whose correspondence is ambiguous, and thus the ones we also aim to remove from the matching process. In [19] every image is matched to a certain number of other images from the same dataset which are sufficiently far away. Note that it is assumed that images are already geolocated, so that the distances are known. A confusion score is computed for the matches in a sliding window fashion, and image regions containing confusing features are masked out. There are several conceptual differences to our approach. First, our ground truth labeling is based on matching performance only and no geolocation or other metadata are needed; second, we do not look for confusing regions, but assess interest points individually; and third, our training is a one-off, off-line procedure on a large enough set of diverse image pairs. Note also that in [19] the confusion score is computed after matching and thus does not improve the computational efficiency, whereas we compute it for single images and remove ambiguous points before matching.

[29] propose to pre-select "useful" features, *i.e*. those which are likely to be found in more than one image of the same object or location. Contrary to our approach, the selection is dependent on the given images and is not transferable to unrelated image sets. We rely on statistics obtainable from arbitrary, unrelated images. In [7] an adaptive non-maxima suppression is presented, which allows one to select a fixed number of interest points with a good spatial distribution in the image. This goal is orthogonal to our goal of ensuring that the points can afterwards be matched. The two methods could even be combined, since also well-distributed features only help if correspondence can be established between them.

3. Matchable Points in Structure-from-Motion

Our application scenario is Structure-from-Motion computation from unordered image sets. While the proposed framework is generic and could be adapted to other computer vision tasks, SfM is maybe the most obvious example. In classical unordered SfM, as represented *e.g*. by Bundler [28], pairwise image matching dominates the computation time and is the major bottleneck, since the number of image pairs to be matched grows quadratically with the input image set.[2]

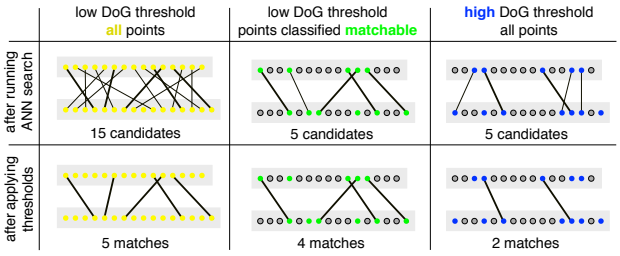


Figure 2: Matching two point sets. In the standard setting a large number of matching candidates are found, only to be decimated by (distance, ratio) thresholds *(left)*. Predicting who will pass the thresholds shortens the candidate list with little impact on the number of matches *(middle)*. A stricter point detector also shortens the list, but a high detector response does not guarantee good matchability *(right)*.

In the following we describe the standard matching method on which we build, explain the details of our matchability prediction, and discuss other potential ways to reduce the number of interest points to be matched. As will be shown in the experimental evaluation, Sec. 4, the selection based on classification yields a drastic reduction in the number of features, with minimal impact on the following SfM computation. On the contrary, pruning to equally small numbers of interest points randomly or based on the point detector destabilizes SfM computation.

3.1. Structure-from-Motion Method

We build on a standard SfM pipeline. Feature points are detected with the Difference-of-Gaussians (DoG) method and represented with SIFT descriptors. In our implementation we use the open-source VLFeat library [31] for these steps. Interest point matching uses approximate nearest neighbor search with seven randomized KD-trees per image and best-bin-first search limited to a total of 128 comparisons per query. Matches are accepted if the distance between the descriptors is $d_{NN} < \sqrt{30,000}$ (for descriptors stored in `byte` values), and if the distance ratio between the best and second-best descriptor is $\frac{d_{NN}}{d_{NN2}} < 0.8$. Matching is also implemented with VLFeat. We point out that in particular the ratio test eliminates a major part of all nearest neighbors, hence the time to find those neighbors in the first place is wasted. Our aim is to filter out interest points which are likely to be weeded out during matching.

For SfM computation the accepted matches are fed into Bundler [28]. Following the default settings only image pairs with $\geq$16 matches are considered for 3D modeling.

[2] Strategies which reduce the number of image pairs to be matched by pair selection [1] or image clustering [10] are complementary. Moreover, even with such methods matching is the most expensive part of SfM.

3.2. Predicting Matchable Features

As explained in the previous paragraphs, although many detected interest points do not appear in the list of matches and consequently do not contribute to the further processing, still a lot of time is spent on trying to match them. To mitigate this wastage and improve efficiency, we propose to predict the matchability of the interest points before passing them to the matching stage, see Fig. 2. The prediction only has to be done once for each point and is thus linear $\mathcal{O}(mn)$ in the number of interest points, respectively the number of images. Moreover, the computation of the descriptors is linear, too, hence the prediction comes at almost no extra cost, only increasing the computation time by a factor $(1 + q)$ with $q \ll 1$. See Sec. 4 for timings.

For simplicity we will in the remainder of the paper refer to a hard classification into "matchable" and "non-matchable" points. In principle one could go further and design a probabilistic matching procedure that exploits the soft classifier score. Still, discarding non-matchable features with a hard threshold is in our view the most effective strategy, due to the reduced memory requirements. Note, if the classification is correct—*i.e*. the discarded point would not have survived the matching thresholds anyway—then no loss of accuracy or reliability is associated with the filter.

For our purposes, a classifier is needed that is on one hand discriminative (since the distributions of matchable / non-matchable descriptors are expected to be rather complex and thus not amenable to generative modeling), and on the other hand fast at test time (since the time spent on classifying points reduces the gain of the method). We use a random forest [2, 6]. The forest is trained off-line (in 450 s for our training set), with no influence on the matching time.

As training data we collect 64 short image sequences with a total of 455 images, acquired at different locations under varying lighting conditions. The images were recorded with a fish-eye lens. A large aperture angle reduces the number of points that would be suitable for matching in terms of their appearance, but are lost because they lie outside the field-of-view in one of the two images.

To generate training matches, each image is matched to 13 preceding and 13 subsequent images (in the order they were recorded) through ANN search and application of the distance and ratio thresholds. Pairs with $<$50 matches are discarded to prevent situations with overly low field-of-view overlap. All points that appear in at least one match form the positive class, while all others (which were always discarded) form the negative class. From each of the two classes 485,000 random descriptors are selected and used to train a random forest with 25 decision trees, using the Gini impurity as the splitting criterion. Tree depth is limited to 25 to prevent possible over-training.

At test time the descriptors of all interest points are fed into the random forest to classify them as matchable (*i.e*. keep the point and pass it on to matching) or non-matchable (*i.e*. discard the point). The classification needs to be done only once per image, as it is independent of other images. [3]

One could think of going even further and training with only inliers after geometric verification (RANSAC fitting of an essential matrix) in the positive class, *i.e*. to learn which interest points are *correctly* matchable. In preliminary experiments we did not observe an improvement with that strategy. A further investigation is left for future work.

3.3. Baseline Methods

Varying the DoG Threshold. A straightforward way to reduce the number of features per image is to raise the threshold for the DoG response, such that fewer interest points are created. In our baseline method we use the standard threshold of 0.001, which generates on the order of a few thousand DoG points per image (excluding edge responses which are removed in the process). Obviously, one can simply use a stricter threshold to get fewer points and speed up matching. However, high scores do not necessarily correlate well with matchability—*e.g*. vegetation generates a lot of strong interest points, especially under strong sunlight, most of which cannot be matched. To evaluate this method we chose the DoG threshold such that the number of interest points is approximately the same as the number of points that survive matchability prediction. In the experiments we will show that this leads to less useful interest points, fewer matches, and less reliable SfM results.

Random Selection of Detected Features. One may go even further and ask whether the set of interest points with the standard settings is just unnecessarily big. As a naive baseline to test that case we bypass our matchability classifier and simply select the same number of features randomly. This baseline serves as a sanity check which any reasonable prediction should beat.

4. Experiments and Results

4.1. Datasets

The first dataset URBAN is a sequence of 1,000 frames recorded at 4 fps, corresponding to a 4-minute walk through a mixed urban/natural environment. The same fish-eye camera was used with which we also captured the training data (in a different recording session). The circular field-of-view covers 185° and has a diameter of 1,320 pixels. For the sequence, ground truth 6D trajectory data is available. The ground truth was generated by fusing navigation-grade IMU and GPS measurements, delivering post-processed position and orientation data with an accuracy of $\pm$5 cm, respectively $\pm$0.25°. The camera was rigidly mounted on the nav-

[3] Project page with classifier code is available at: `http://www.igp.ethz.ch/photogrammetry/research/pm`.

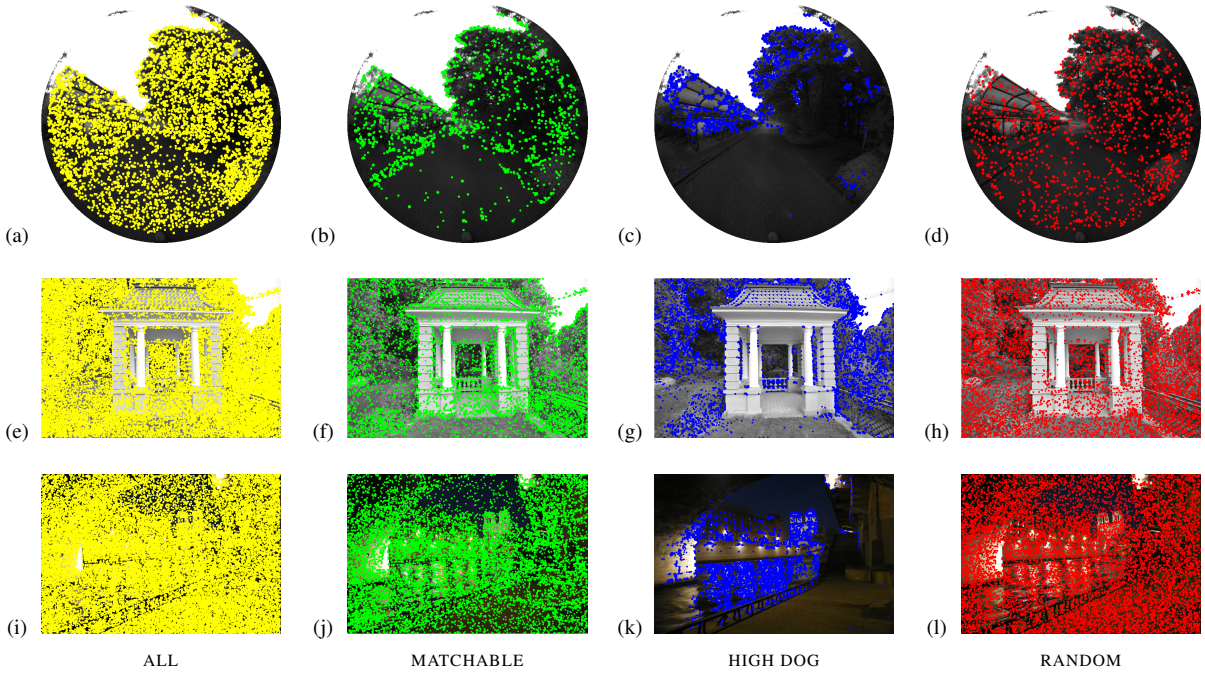


Figure 3: Sample images from sequences URBAN *(top)*, PARK *(middle)*, and NOTREDAME *(bottom)*. Interest points of the reduced point sets are denoted by color dots.

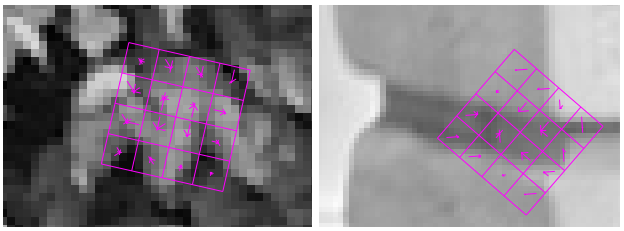
igation system. The offset between IMU center and camera center had to be estimated to convert the navigation data to ground truth camera poses.

The second dataset PARK consists of 121 images with 6 Mpix resolution, acquired in an urban park with a standard perspective DSLR camera. No ground truth camera poses are available for this dataset. The publicly available NOTREDAME dataset [27], which consists of 715 images downloaded from Flickr [32], was used for evaluation too. The images have different resolutions and were taken in various lighting conditions and different seasons. Note that for these additional datasets the same classifier as for URBAN was used, *i.e*. we did not retrain for the different camera and/or dataset type.

4.2. Qualitative Results

In Fig. 3 we visually compare the different methods for decimating the feature point sets before matching and SfM computation. Noticeable differences between the methods can be observed on URBAN. In (Fig. 3b) features are deemed non-matchable on the street surface, but not along the stone curb. Many features on the roof structure are classified as matchable, whereas few on the vegetation are. With a higher detector threshold (Fig. 3c), almost no points in the foreground are retained. Instead, the points are concentrated in high-contrast regions along the border of the sky region. While they do cover the roof on the left reasonably well, very few points remain on the road, whereas a part of the tree on the right is covered where the gradients are high (but potentially too unstructured for matching). Note also that with the HIGH DOG method the point distribution in the image is largely governed by large-scale contrast differences and hence less favorable for SfM computation. The RANDOM features are the other extreme (Fig. 3d). Their distribution across the image is even, irrespective of whether the local appearance gives rise to unique descriptors. Consequently, a lower fraction of the points will survive matching, such that one ends up with fewer correspondences, distributed similarly to the MATCHABLE ones.

Similar results can be observed for PARK. Non-matchable points are concentrated on regular textures such as vegetation (which is known to create few useful matches), whereas regions with more structured, and thus less confusing, gradients, *e.g*. man-made objects, are largely preserved (Fig. 3f). We also point out that interest points like those on the pavilion are particularly important for SfM computation, because they are stable when seen from different viewpoints. Again, the HIGH DOG method exhibits the most uneven concentration of features in few high-contrast areas, and quite some points on bushes, which are unlikely to generate a valid match (Fig. 3g). The RANDOM points are of course again distributed evenly, but such that few points remain on favourable structures like the pavilion, and many points will be lost altogether during matching (Fig. 3h).



(a) high response, non-matchable (b) low response, matchable

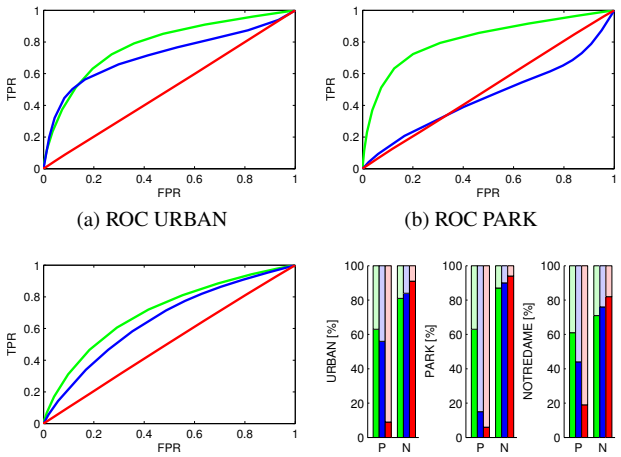
Figure 4: Detector response *vs*. matchability. Not all high-contrast points have unambiguous descriptors *(a)*, whereas some low-contrast points do *(b)*.

In the night scene of NOTREDAME the methods perform similarly again. The features classified matchable are distributed on all the structure in the image (Fig. 3j) whereas for the higher DoG threshold (Fig. 3k) there are many more features on the water surface and only in a small area of the interesting building and bridge structure. Random selection gives again an even distribution of features and has more features on the sky than the other two strategies (Fig. 3l).

The qualitative experiment shows that using the right objective function for interest point selection does make a difference—MATCHABLE points are significantly different from points with HIGH DOG score. Reasonably high DoG responses guarantee sufficient contrast, and thus good localization and repeatability, but very high DoG responses merely restrict the points to high-contrast regions. But since invariant descriptors normalize for contrast, the structure of the local gradients is more important than their magnitude. To further illustrate this point we look at exemplary SIFT descriptors. Fig. 4a shows a feature on vegetation which has a very high DoG response, but cannot be matched. Conversely, in Fig. 4b is a descriptor on a pillar which is matchable, but will not be found with a stricter DoG threshold.

4.3. Quantitative Results

We go on to quantitatively compare the matching results of different methods. Starting from all points, we find the ones that classify as matchable, then also pick a similarly sized set by increasing the higher DoG threshold, and pick a set of the same size randomly. Next, we exhaustively match all image pairs in each dataset, using all points found with the standard DoG setting of 0.001 (independent of the classifier prediction) so as to obtain a baseline of what is possible if computation time is not a concern. The set of nearest-neighbor assignments is then separated into four parts: *(i)* TP – those which could be successfully matched and where both the point from the source and the target image are in the reduced point sets from the MATCHABLE, HIGH DOG, or RANDOM methods, respectively, meaning that keeping the points was correct; *(ii)* TN – those rejected by the matching criteria where at least one point was not in the reduced set, meaning the rejection was correct; *(iii)* FP – those rejected although both points were in the reduced lists, meaning that keeping them was incorrect; and *(iv)* FN – those which could be matched although at least one of the points was not in the reduced set, meaning that the rejection was false. We point out that this procedure is biased *against* the selection methods: typically the number of matches would be larger, because a smaller number of image points brings a higher chance to survive the ratio test. However, it would also mean that the results after pruning the point sets would no longer be subsets of the ones without pruning. In the system-level SfM experiments (Sec. 4.4) the correct procedure was used, where only the reduced point sets are fed into the matcher.



(a) ROC URBAN (b) ROC PARK

(c) ROC NOTREDAME (d) classifier threshold 0.5

Figure 5: *(a-c)* ROC curves of the three evaluated methods. *(d)* Confusion bars for the three datasets. MATCHABLE *(green)*, HIGH DOG *(blue)*, and RANDOM *(red)*.

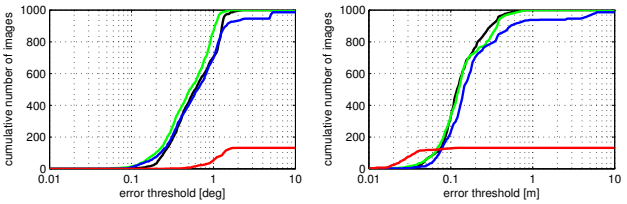
Fig. 5 shows the statistics for 499,500 image pairs from URBAN, 7,260 image pairs from PARK, and 255,255 image pairs from NOTREDAME. Especially for very wide-baseline pairs the vast majority of nearest neighbors do not pass the ratio and distance tests. For that portion the performance of all three methods is similar. More than 80% of the matches which would be rejected by the ratio and distance tests anyway are pruned due to the point set reduction. For those points which did survive the classification, higher threshold, or random selection, respectively, the results do show clear differences: About 60% of the matches accepted by the ratio and distance tests have both endpoints classified as matchable and therefore survive the reduction. The other 40% of the accepted matches are (unnecessarily) pruned—and thus lost to the subsequent SfM computation—but we show that this loss is still tolerable and does not impair SfM computation (Sec. 4.4).

URBAN	ALL	MATCHABLE	HIGH DOG
# image pairs		499,500	
DoG+SIFT	1,530 s	1,530 s	990 s
build KD-trees	140 s	35 s	35 s
query KD-trees	65,240 s	18,230 s	18,230 s
classification	—	80 s	—
total	66,910 s	19,875 s	19,255 s

PARK	ALL	MATCHABLE	HIGH DOG
# image pairs		7,260	
DoG+SIFT	780 s	780 s	345 s
build KD-trees	215 s	35 s	35 s
query KD-trees	11,895 s	1,960 s	1,960 s
classification	—	70 s	—
total	12,890 s	2,845 s	2,340 s

NOTREDAME	ALL	MATCHABLE	HIGH DOG
# image pairs		255,255	
DoG+SIFT	2,820 s	2,820 s	1,740 s
build KD-trees	600 s	205 s	205 s
query KD-trees	180,755 s	74,680 s	74,680 s
classification	—	205 s	—
total	184,175 s	77,910 s	76,625 s

Table 1: Matching times for the three datasets.

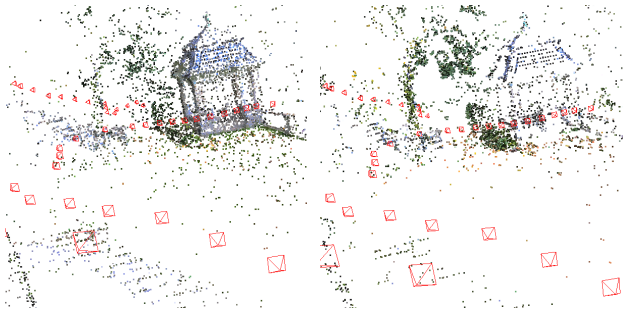


(a) absolute rotation vs. GT (b) absolute position vs. GT

Figure 6: Accuracies of camera poses for URBAN, see Fig. 5 for the color legend. Note that high position accuracy of RANDOM is due to the small number of camera centers.

After random selection $<10\%$ of the possible matches remain, which is natural because the reduction is independently done for each image, such that most points end up without a correct matching partner. The higher DoG threshold performs only slightly worse than matchability prediction for the URBAN dataset, which is dominated by building facades. On the other hand the higher threshold has a serious impact on the PARK sequence, where there are larger vegetation regions. The matchability classifier rejected on average a larger portion of features for PARK than for the other datasets, so to keep the sizes of reduced feature sets comparable the higher DoG threshold setting was 0.01 for URBAN, 0.03 for PARK, and 0.014 for NOTREDAME. Feature repeatability greatly suffers on PARK from such overly strict DoG threshold.

Next, we quantify the speed-up achieved through the matchability prediction. Timings (C++ code on a multi-core desktop) are given in Tab. 1. Obviously the portion of matchable features, and hence the gain, depends on the scene. For URBAN matching is faster by a factor of 3, while for PARK the factor is >4. Due to the smaller number of descriptor computations and the missing classification step the HIGH DOG baseline is slightly faster than MATCHABLE.



(a) MATCHABLE (b) HIGH DOG

Figure 7: 3D point clouds of dataset PARK triangulated from two of the examined selection methods. Notice different point densities on the vegetation and the pavilion.

4.4. Accuracy of Estimated Camera Poses

Six sets of SIFT feature points—all detected feature points, feature points classified matchable, feature points for the higher DoG threshold, and three random selections from all feature points—were used in the SfM experiment for dataset URBAN.

In each of the six independent experiments, the corresponding feature point sets of all image pairs were matched using the standard thresholds of $\sqrt{30,000}$ for distance and 0.8 for the ratio test. The point coordinates were transformed to virtual perspective views (horizontal field-of-view 90°, aspect ratio 4:3) using the known calibration, because Bundler cannot handle fish-eye lenses. SfM computation was then run with radial distortion estimation disabled and fixed internal camera parameters. The resulting six 3D models and corresponding camera poses were examined and only the best-performing of the three random selections was kept for further evaluation.

The camera trajectories of the four remaining reconstructions were aligned with the ground truth trajectory from GPS/IMU by fitting a 3D similarity transform to the camera centers. For the HIGH DOG method, pose estimation failed towards the end of the trajectory, so only the first 935 camera centers were used for alignment. Also, only 132 out of 1,000 images were connected into one model for the best random selection. The trajectory computed from the MATCHABLE points fits nicely to the one estimated from all detected feature points (which actually has a small error in one of the camera turns, too).

The accuracy of the different trajectories can be seen in Fig. 6. Note that the error when using only the matchable points is as low as when using all points.

The same experiment was run with dataset PARK, using the original perspective images and the default setting of Bundler, with radial distortion estimation enabled. As no ground truth is available, we compare the reconstructed point clouds. The obtained 3D models are quite different, see Fig. 7. The majority of triangulated 3D points for the higher DoG threshold correspond to unstable vegetation whereas there are many more 3D points on the pavilion and staircase in the model from the feature points classified matchable.

5. Conclusions

We have explored whether one can predict from an interest point's descriptor alone whether that point will be matchable, before feeding it to the actual matching procedure. By learning a binary classifier for that task we were able to reduce the set of image points to $\approx 30\%$ of its original size, but retain $\approx 60\%$ of the matches accepted by the ratio and distance tests. We have shown that by only keeping points with a high probability of generating a valid match it is possible to greatly speed up SfM computation with a minimal loss of robustness and accuracy. In our experiments the descriptor-based prediction clearly outperforms a baseline that uses a stricter detection threshold in terms of correct camera pose estimation.

The idea is orthogonal to methods which speed up matching by selecting promising image pairs. It will be interesting to see whether synergies exist between the two strategies, *e.g*. it may be possible to exploit the matchability predictions for different images to select good pairs.

References

[1] S. Agarwal, N. Snavely, I. Simon, S. Seitz, and R. Szeliski. Building Rome in a day. In *ICCV*, 2009.

[2] Y. Amit and D. Geman. Shape quantization and recognition with randomized trees. *Neural Comput.*, 9:1545–1588, 1997.

[3] 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.

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

[5] P. R. Beaudet. Rotationally invariant image operators. In *4th Int'l Joint Conf. on Pattern Recognition*, 1978.

[6] L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[7] M. Brown, R. Szeliski, and S. Winder. Multi-image matching using multi-scale oriented patches. In *CVPR*, 2005.

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

[9] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse. MonoSLAM: Real-time single camera SLAM. *PAMI*, 29(6):1052–1067, 2007.

[10] J.-M. Frahm et al. Building Rome on a cloudless day. In *ECCV*, 2010.

[11] A. Gionis, P. Indyk, and R. Motwani. Similarity search in high dimensions via hashing. In *VLDB*, 1999.

[12] A. A. Goshtasby. *Image Registration – Principles, Tools and Methods*. Springer, 2012.

[13] S. Guha and S. Khuller. Approximation algorithms for connected dominating sets. *Algorithmica*, 20(4):374–387, 1998.

[14] C. Harris and M. Stephens. A combined corner and edge detector. In *4th Alvey Vision Conference*, 1988.

[15] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, second edition, 2004.

[16] M. Havlena, A. Torii, J. Knopp, and T. Pajdla. Randomized structure from motion based on atomic 3D models from camera triplets. In *CVPR*, 2009.

[17] M. Havlena, A. Torii, and T. Pajdla. Efficient structure from motion by graph optimization. In *ECCV*, 2010.

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

[19] J. Knopp, J. Sivic, and T. Pajdla. Avoiding confusing features in place recognition. In *ECCV*, 2010.

[20] X. Li, C. Wu, C. Zach, S. Lazebnik, and J.-M. Frahm. Modeling and recognition of landmark image collections using iconic scene graphs. In *ECCV*, 2008.

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

[22] A. Oliva and A. Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *IJCV*, 42(3):145–175, 2001.

[23] E. Rosten, R. Porter, and T. Drummond. Faster and better: A machine learning approach to corner detection. *PAMI*, 32(1):105–119, 2010.

[24] J. Shi and C. Tomasi. Good features to track. In *CVPR*, 1994.

[25] C. Silpa-Anan and R. I. Hartley. Optimised KD-trees for fast image descriptor matching. In *CVPR*, 2008.

[26] J. Sivic and A. Zisserman. Video google: A text retrieval approach to object matching in videos. In *ICCV*, 2003.

[27] N. Snavely. BigSFM: Reconstructing the world from internet photos – `http://www.cs.cornell.edu/projects/bigsfm`, 2012.

[28] N. Snavely, S. Seitz, and R. Szeliski. Modeling the world from internet photo collections. *IJCV*, 80(2):189–210, 2008.

[29] P. Turcot and D. G. Lowe. Better matching with fewer features: The selection of useful features in large database recognition problems. In *WS-LAVD*, 2009.

[30] T. Tuytelaars and K. Mikolajczyk. Local invariant feature detectors: A survey. *Foundations and Trends in Computer Graphics and Vision*, 3(3):177–280, 2008.

[31] A. Vedaldi and B. Fulkerson. VLFeat: An open and portable library of computer vision algorithms. `http://www.vlfeat.org`, 2008.

[32] Yahoo! Flickr: Online photo management and photo sharing application – `http://www.flickr.com`, 2005.