Improving Image-Based Localization by Active Correspondence Search

Torsten Sattler[1], Bastian Leibe[2], and Leif Kobbelt[1]

[1] RWTH Aachen University, Aachen, Germany
[2] UMIC Research Centre, RWTH Aachen University, Aachen, Germany

Abstract. We propose a powerful pipeline for determining the pose of a query image relative to a point cloud reconstruction of a large scene consisting of more than one million 3D points. The key component of our approach is an efficient and effective search method to establish matches between image features and scene points needed for pose estimation. Our main contribution is a framework for actively searching for additional matches, based on both 2D-to-3D and 3D-to-2D search. A unified formulation of search in both directions allows us to exploit the distinct advantages of both strategies, while avoiding their weaknesses. Due to active search, the resulting pipeline is able to close the gap in registration performance observed between efficient search methods and approaches that are allowed to run for multiple seconds, without sacrificing runtime efficiency. Our method achieves the best registration performance published so far on three standard benchmark datasets, with run-times comparable or superior to the fastest state-of-the-art methods.

1 Introduction

Image-based localization addresses the problem of finding the position and orientation from which a given picture was taken. It is encountered in many interesting applications, such as touristic landmark recognition [1, 2], robot localization and loop closure [3, 4], Augmented Reality [5, 6], and Structure-from-Motion [7].

Available approaches [1–6, 8–13] aim at providing a camera pose estimate relative to a given scene. The choice of the scene representation has a direct impact on the way how the localization problem is solved. Using an image collection to model a scene enables the use of efficient image retrieval techniques [14, 15] for scalable localization. Recently proposed systems are able to handle hundreds of thousands [12, 13] or even millions of images [2, 9]. They estimate the position of the query image using the GPS tags of retrieved images [12, 13]. If higher localization accuracy is required, a 3D point cloud is a more suitable scene representation. Using correspondences between 2D image features and 3D scene points, the full camera pose, *i.e.*, position and orientation, can be estimated with high precision [16]. Also, 3D models offer a more compact representation than image database, which contain many features not related to 3D scene points.

In this paper, we represent a scene as a 3D point cloud obtained from an offline Structure-from-Motion reconstruction, where every 3D point is associated

A. Fitzgibbon et al. (Eds.): ECCV 2012, Part I, LNCS 7572, pp. 752–765, 2012.

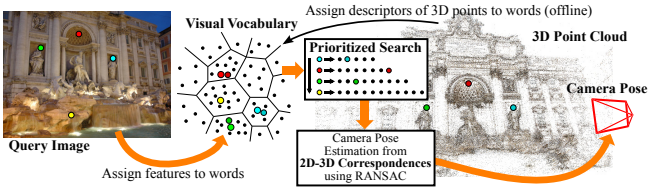

Fig. 1. Illustration of the localization pipeline. The descriptors of 2D features and 3D points are assigned to visual words. The camera pose is estimated from 2D-3D matches between features and points by using an n-point pose algorithm inside a RANSAC-loop.

with its corresponding image descriptors. The main challenge in this setting is to efficiently search for correspondences between thousands of 2D query image features and millions of 3D scene points. Afterwards, the camera pose can be estimated using RANSAC [17]. The number of RANSAC iterations depends on the quality of the found matches. Thus, the correspondence search not only has to be efficient but also highly accurate in order to avoid spending too much time on pose estimation. Sattler *et al*. have shown that a simple, tree-based approximate search through all point descriptors yields good registration performance [11]. However, this approach requires several seconds for the search alone. Recently, efficient algorithms have been proposed that are up to one order of magnitude faster [1, 10, 11]. So far, this speed-up comes at the cost of a lower number of images that can be registered. In this paper, we propose a novel approach that (a) is able to close the performance gap to slower tree-based search *and* (b) is still one order of magnitude faster. At its core is a novel active search step that uses established 2D-to-3D correspondences to trigger 3D-to-2D matching. These additional matches required to close the performance gap can be found efficiently by taking advantage of an intrinsic similarity between searches in both directions. We balance the additional effort of the active search step by several visibility filtering steps. We thoroughly evaluate the effects of all system parameters and demonstrate state-of-the-art performance on three challenging datasets.

Related Work. Approaches such as SLAM [3, 5, 18] and PTAM [6] compute the camera pose relative to an online-built 3D model of the scene. Since larger reconstructions quickly become infeasible, they are restricted to smaller scenes.

Given an offline-built point cloud of a large-scale scene, Irschara *et al*. propose an image retrieval-based system [1]. Their approach obtains correspondences by matching 2D features from the query image to 2D features in the most similar database images. These image features in turn belong to 3D points in the scene. Besides the images used in the reconstruction, the database also contains synthetic views created by back-projecting 3D points into virtual cameras. For real-time performance, GPU implementations for vocabulary tree-based retrieval [14] and feature matching are used. Wendel *et al*. extend this pipeline to localize MAVs by creating virtual views in full 3D instead of on the ground [4].

Alcantarilla *et al*. learn a similarity metric between images based on camera poses and appearance information [8]. Given a query camera and a rough pose

estimate, the metric can predict which 3D points should be visible in it. Li *et al.* also use visibility information from the images used for the reconstruction to guide a 3D-to-2D matching process [10]. They propose to accelerate the search process with prioritization and early termination: When a match for a 3D point is found, the priorities of the points visible with it in a database image increase.

In contrast, Sattler *et al.* investigate direct 2D-to-3D matching strategies [11]. Motivated by the registration performance of tree-based search, they propose an efficient vocabulary-based search approach to establish 2D-to-3D matches. Although their quantized representation does not achieve the same performance as tree-based search, their approach still considerably improves on the methods by Irschara *et al.* [1] and Li *et al.* [10], while achieving similar run-times.

In this paper, we combine 2D-to-3D and 3D-to-2D search into a novel active correspondence search step that allows our approach to achieve the same high registration performance as tree-based matching, while resulting in run-times comparable to or faster than the most efficient available approaches. In preparation of presenting the active search mechanism in Sec. 3, we discuss several important aspects of the image-based localization problem in the next section. In a slight abuse of notation, we will refer to 2D features and their descriptors simply as *features* and to 3D points and their descriptors as *points*.

2 The Image-Based Localization Problem

Fig. 1 visualizes the image-based localization problem. Given a 3D point cloud of a scene and a query image, we want to compute the accurate camera pose of the query image. Since the point cloud was obtained by Structure-from-Motion, every 3D point is associated with at least two SIFT descriptors [19] from database images. By extracting local features in the query image and matching them to the database descriptors, we can establish correspondences between 2D features and 3D points, using the SIFT ratio test [19] to reject ambiguous matches. The main challenge in image-based localization is to efficiently find a large enough number of high-quality correspondences to facilitate pose estimation.

2D-to-3D *vs*. 3D-to-2D Search. Since an image contains several orders of magnitude fewer features than there are points in the model, matching a single point against the features (3D-to-2D) is more efficient than matching in the other direction. This comes at the cost of matching quality; Li *et al.* note that about every $500th$ point matches by pure chance [10]. In contrast, the slower 2D-to-3D matching leads to higher-quality matches: If there are several points in the database that closely match a feature, the ratio test will reject the feature as too ambiguous. In return, this denser descriptor space is also more likely to reject correct correspondences, especially for larger datasets. In the next section, we therefore show how to combine the advantages of both search strategies – retrieving more correspondences, while maintaining a high confidence level – through our novel active search mechanism.

Prioritized Search. Correspondence search can be made more efficient by considering only a fraction of all features/points. This requires a prioritization

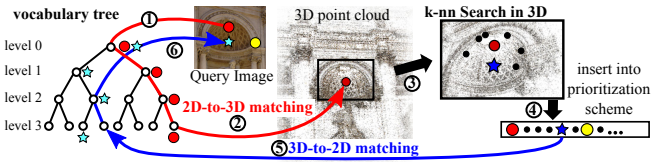

Fig. 2. After finding a 2D-to-3D match, we actively search for 3D-to-2D correspondences for the N_{3D} points closest to the matched point (red). Using coarser levels in the tree, we are able to recover matches that would otherwise be lost due to quantization.

scheme that first evaluates the most promising features/points, *e.g.*, those most likely to yield a match, and that stops when enough correspondences have been found. For 3D-to-2D matching, Li *et al.* propose a prioritization scheme purely based on co-visibility of 3D points [10]. When a match for a point p is established, the priorities of all other points visible together with p in one database image are increased. For 2D-to-3D matching, Sattler *et al.* propose a strategy purely based on appearance [11]. They assign both points and features to a set of visual words. The cost of processing a feature depends on the number of descriptor distance computations needed to find its nearest neighbor. This number of computations is proportional to the number of points assigned to the visual word of the feature. Features are then evaluated in order of increasing cost, preferring words with only few points and thus highly distinctive appearance. In this paper, we exploit the intrinsic similarity between 2D-to-3D and 3D-to-2D matching to optimize the strategy from [11] such that it prioritizes over both appearance and co-visibility.

Bridging the Performance Gap. There is a noticeable gap in the number of images that can be registered with more efficient approaches [1, 10, 11] compared to simple but slow tree-based search [11]. The main reason for this gap is that the tree-based approach places fewer restrictions on the viewing conditions under which a query image can be registered. Because of the prohibitive run-times of tree-based search, there is however a strong need for more efficient, yet equally effective and robust algorithms for correspondence search.

In the next section, we introduce our novel active correspondence search scheme. This core component of our approach incorporates 3D-to-2D search into 2D-to-3D prioritized matching. Sec. 4 discusses an efficient implementation of our method. As active search, although computationally efficient, requires additional effort, Sec. 5 shows how to further improve the run-times of both active search and RANSAC-based pose estimation through the use of visibility information. Compared to tree-based search, our approach is one order of magnitude faster at similar or superior registration rates.

3 Active Correspondence Search

We start from the framework proposed by [11] (*c.f.* Fig. 1). The points in the model are assigned to visual words in an offline stage. Given a query image, its SIFT features are again mapped to the vocabulary. This results in a list of

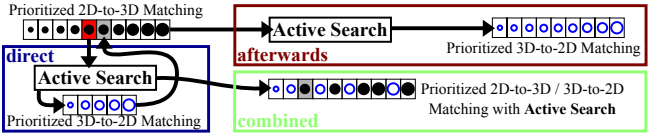

Fig. 3. The three prioritization strategies: The *direct* strategy performs 3D-to-2D matching as soon as candidate points have been found. The *afterwards* strategy finishes 2D-to-3D matching first before performing active search. The *combined* strategy chooses the search direction based on search cost (indicated by dot sizes).

candidate points for every feature and the features are considered in ascending order of the length of their lists. For every feature f, we search through all points stored in its word and find its two nearest neighbors, *i.e.*, the points p_1, p_2 with the most similar descriptors. A correspondence $(f, p_1, \|f - p_1\|_2)$ is established if the SIFT ratio test $\|f - p_1\|_2 / \|f - p_2\|_2 < 0.7$ is passed. Here, $\|f - p_1\|_2$ is the descriptor distance between the descriptors of f and p_1.

Active Correspondence Search. The use of a visual vocabulary induces quantization effects that limit the number of correspondences that can be found. A simple approach to resolve these effects would be to use soft assignments, mapping one feature to multiple visual words [20]. We instead propose to use *active correspondence search*, a more efficient hierarchical approach. Its basic idea is shown in Fig. 2: Once a 2D-to-3D correspondence between a feature f and a point p has been found, there is a high probability that points in the 3D region around p are also visible in the query image. While the method from [11] ignores this information and continues matching features to points, we actively search for correspondences among the N_{3D} points closest in 3D space to p. Every such point p' is inserted into the prioritization scheme and once it is activated, we search for a 3D-to-2D match among image features with descriptors similar to p'. This set of similar features can again be identified using a visual vocabulary. While we require a rather fine vocabulary for 2D-to-3D matching to limit the search space, we need to use a coarser vocabulary for the 3D-to-2D search step to guarantee that enough features are considered. A key observation is that information about a coarser vocabulary can be obtained without additional cost by using a vocabulary tree [14, 21] to perform the initial 2D-to-3D matching. Using a coarser level in the tree has the additional benefit of recovering matches lost due to quantization effects (*c.f.* Fig. 2). We provide details on an efficient implementation of active correspondence search in Sec. 4. Note that active search is only triggered by 2D-to-3D correspondences and not by 3D-to-2D matches.

Prioritization. Prioritization is key for efficient correspondence search. In our formulation, correspondence search in any direction is modeled as finding the two nearest neighbors stored in a visual word for a given descriptor. The prioritization scheme from [11], in which the search cost of a descriptor depends on the number of comparisons needed to find its nearest neighbors, is thus used

to prioritize search *independently* of the direction. *We stop the search when N matches have been found.* The remaining questions are when to perform the active search and which kind of information, appearance (2D-to-3D) or co-visibility (3D-to-2D), should be preferred. When co-visibility is more important, a *direct* prioritization strategy can perform active search as soon as a 2D-to-3D match is found. 2D-to-3D search is resumed after matching the N_{3D} candidate points against the image features. This can lead to finding many matches in a small region in the image, yielding unstable configurations for pose estimation and worse localization accuracy. If appearance information is preferred, 2D-to-3D search is performed first and active search and 3D-to-2D matching follow *afterwards*. This strategy might benefit only little from active search. We thus propose a *combined* strategy to balance both directions. It performs active search as soon as a new match is found and then uses the predicted search costs of the candidates to sort them into a common prioritization scheme for both directions. This strategy treats both kinds of information equally, always preferring the kind that is cheaper to evaluate. Fig. 3 illustrates the three prioritization strategies.

Computational Complexity. Given a point cloud with P points and a vocabulary of size W, the mean number of points stored in a word is $\frac{P}{W}$. Considering c+1 words instead of one, soft assignment on average adds a search cost of $c \cdot \frac{P}{W}$ for *each* considered feature. This results in an additional search time of $\mathcal{O}(c \cdot F \frac{P}{W})$ for an image containing F features. In contrast, active search is triggered at most N times until enough matches are found. Using a kd-tree, the N_{3D} nearest points can be found in time $\mathcal{O}(N_{3D} \log_2(P))$. Using a coarser vocabulary of size W', one point is on average matched against $\frac{F}{W'}$ features. Since N and N_{3D} are constants, active search introduces an additional cost of $\mathcal{O}(\log_2(P)\frac{F}{W'})$, which is more efficient than soft assignments for large datasets. In practice, $\frac{F}{W'}$ can be considered constant for a fixed W', since an image contains 1k-20k features.

Comparison with Existing Methods. Compared to [11], our approach is able to recover correspondences lost due to the fine vocabulary used. We show in Sec. 6 that these matches are crucial to close the performance gap. In contrast to [10], our active search is based on 2D-to-3D matches, which are inherently more reliable than the 3D-to-2D matches used by Li *et al*. This allows our algorithm to achieve both better performance and efficiency than [10]. Both results indicate that merging the two search directions is critical to achieve good performance. Active search even offers an advantage over tree-based search. Since the density of the descriptor space is related to the number of 3D points, the ratio test will remove more correct matches for larger datasets. By relaxing the matching to the sparser space in the image, our method can recover these lost matches.

4 Efficient Implementation

In this section, we provide details on the implementation of our method.[1] A fine visual vocabulary of 100k words is used for 2D-to-3D matching, generated

[1] Source code is available at `http://www.graphics.rwth-aachen.de/localization`

Algorithm 1. 3D-to-2D correspondence search

Given: point p' with set of descriptors $D(p')$, set of features F found in image, level l in the tree, set C of correspondences
Compute set AW of words on level l assigned to descriptors from $D(p')$
Initialize the two nearest neighbors: $f_1 = f_2 = \text{noNNfound}$, $\text{dist}_1 = \text{dist}_2 = \infty$
/* Find two nearest neighbors in image */
for $w \in AW$ **do**
  $F(w) =$ features from F assigned to word w on level l
  for all pairs $(d \in D(p')$ assigned to w, $f \in F(w))$ **do**
    Compute descriptor distance: $\text{dist} = \|f - d\|_2$
    use *dist* to update nearest neighbors and distances dist_1, dist_2, s.t. $f_1 \neq f_2$
/* Establish correspondence using SIFT ratio test */
if $\text{dist}_1, \text{dist}_2 < \infty$ and $\text{dist}_1/\text{dist}_2 < 0.6$ **then**
  if C contains correspondence $c = (f_1, q, \text{dist})$ **then**
    if c not found by 2D-to-3D matching, **replace** (f_1, q, dist) with (f_1, p', dist_1) if $\text{dist}_1 < \text{dist}$
  else
    add $C = C \cup \{(f_1, p', \text{dist}_1)\}$
Return: Set of correspondences C

with approximate k-means clustering [15] of SIFT descriptors. A vocabulary tree with branching factor 10 is constructed on top of this vocabulary [21]. Similar to [11], we represent a point stored in a word by the mean of all of its descriptors assigned to this word. For memory efficiency, the entries of the mean are rounded to the nearest integer values. Active search is incorporated into the 2D-to-3D pipeline from [11] (*c.f.* Sec. 3). After finding $N = 100$ matches, the camera pose is estimated with a RANSAC-variant [22] using the 6-point DLT algorithm [16].

Alg. 1 shows our implementation of 3D-to-2D matching. As every point in the model is visible in multiple images, a candidate point p' for 3D-to-2D matching usually has multiple descriptors $D(p')$ stored in different words of the fine vocabulary. They define a set AW of activated visual words on level l in the tree. We find the two features f_1, f_2 assigned to words from AW that have the smallest descriptor distance to the descriptors from $D(p')$. The two for-loops in Alg. 1 thus define the search cost of p'. We enforce that f_1 and f_2 are distinct features so that the SIFT ratio test can reject ambiguous matches. To reflect the larger uncertainty inherent in 3D-to-2D correspondences, 3D-to-2D matches cannot replace 2D-to-3D correspondences. Features for which a 3D-to-2D correspondence has been found are not considered for 2D-to-3D matching anymore.

The choice of the level l used in the tree directly controls the number of assigned image features and thus the search cost for the candidate point p'. Since a query image typically contains 1k-20k features depending on its resolution, we use level two (100 words) when at most 5k features are found in the image and level three (1000 words) otherwise. Thus, on average $5-50$ features are assigned to a word, *i.e.*, the search cost for p' can be assumed to be constant.

5 Incorporating Additional Visibility Information

Although more efficient than soft matching, the active search steps need additional computational effort. In this section we therefore show how to speed up the localization pipeline in order to compensate for this run-time increase. The

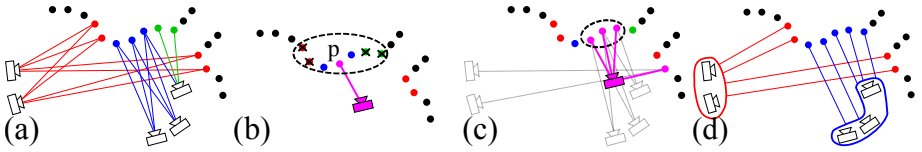

Fig. 4. (a) Points and cameras define a bipartite graph G. (b) Neighboring points not directly visible with the matched point p (pink) are removed. (c) We consider only the largest connected component (dashed) in the subgraph G_C (pink) defined by the matches. (d) Clustering cameras into sets defines a new bipartite graph.

improvements are based on the observation that the images used for the reconstruction approximate the set of viewpoints from which a point is visible. This information allows us to speed up both 3D-to-2D matching and RANSAC-based pose estimation by filtering out points unlikely to be visible. Due to their approximative nature, such filters can also remove correct points. We propose a simple strategy based on camera sets to recover the lost performance.

The filtering steps can be expressed as operations on a bipartite graph G defined by the cameras and 3D points in the model. Here, one set of nodes in G represents the images and the other one the points (*c.f.* Fig. 4(a)).

Filtering 3D Points. Close proximity in 3D space does not imply co-visibility of two points; they could have been seen from very different directions (*c.f.* Fig. 4(a)). Our *point filter* removes all points from the N_{3D} nearest neighbors not directly visible with the point p that triggered active search. Only points two edges away from p in G are used for 3D-to-2D matching (*c.f.* Fig. 4(a),(b)).

A RANSAC Pre-filter. As shown in Fig. 4(c), the established matches define a subgraph G_C of G. Points in different connected components in G_C should not be visible together. Our *RANSAC pre-filter* finds the connected component in G_C containing the most 3D points and keeps only matches pointing to this component. This accelerates the RANSAC-based pose estimation since outliers are removed. Notice that this filter has little impact on 3D-to-2D matches since it is unlikely to find further correspondences around a wrong 2D-to-3D match.

Using Camera Sets. The filtering steps might be too aggressive (*c.f.* green points in Fig. 4(b)). By merging cameras we try to find a better, more continuous approximation of point visibility. For every image I_j, we define the set of similar images $\mathrm{sim}(I_j)$ as all images among the k closest cameras whose viewing direction differs from I_j by at most 60°. A minimal subset $S' \subset S = \{\mathrm{sim}(I_j)\}$ of the camera sets is selected such that every image I_j is contained in at least one set $s \in S'$. This set cover problem is solved using a Greedy algorithm which iteratively selects the set containing the most cameras that have not yet been covered [10]. As shown in Fig. 4(d), the 3D points and S' again form a bipartite graph G', replacing an edge $\{p, I_j\}$ with $\{p, s\}$ if $I_j \in s$ for $s \in S'$. When using camera sets, G' is used instead of G to define the two filtering steps described above.

Table 1. Datasets used for experimental evaluation

	# Cameras	# 3D Points	# Descriptors	# Query Images
Dubrovnik	6044	1,886,884	9,606,317	800
Rome	15,179	4,067,119	21,515,110	1000
Vienna	1324	1,123,028	4,854,056	266

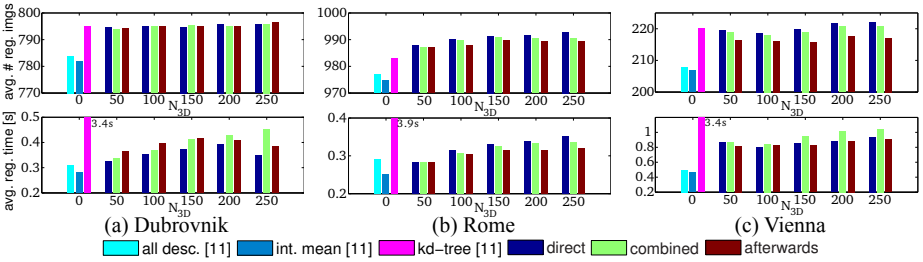

Fig. 5. Effect of N_{3D} on the mean number of registered images (top row) and the mean registration time (bottom row). The matches found through active search enable us to achieve the performance of tree search while being substantially faster. Compared to [11] the registration time increases with N_{3D} due to the added computational effort.

6 Experimental Evaluation

In this section we evaluate our localization approach based on active search, as well as the optimizations discussed in the last section. To allow an easy and fair comparison, we perform our experiments on the datasets already used in [1, 10, 11], kindly provided by Irschara *et al.* and Li *et al.* The Dubrovnik and Rome datasets were obtained by generating 3D point clouds from Flickr images and removing some images together with their descriptors and all points visible in only one remaining camera [10]. The removed images are used as query images. The Dubrovnik model consists of a single connected component, while the Rome model consists of multiple reconstructions of individual landmarks. The Vienna model was generated from images of distinct landmarks taken with a single camera at regular intervals. Images from the Panoramio website are used for testing [1]. All query images have a maximal side length of at most 1600 pixels. Details on the datasets can be found in Table 1. Following [10, 11], we consider an image as registered if the best pose found by RANSAC has at least 12 inliers.

All experiments for our methods were performed using a single CPU thread on a PC with an Intel i7-920 CPU with 2.79GHz. We report the mean registration performance and registration times of our pipeline, excluding feature extraction, averaged over 10 repetitions. Localization accuracy is measured on a metric version of the original Dubrovnik model. Following [11], we report the deviation of the averaged registered camera positions from the ground truth positions.

Evaluation of Active Search. We first evaluate only the proposed active correspondence search together with the three prioritization strategies described in Sec. 3 for different values of N_{3D}. (None of the optimizations from Sec. 5 is

Table 2. Localization errors for the Dubrovnik dataset using the three prioritization strategies and different values for N_{3D}. The *direct* strategy performs worse than the other strategies due to its tendency to find many matches in the same image region.

Method / Strategy	N_{3D}	Median [m]	Quartiles [m] 1st	Quartiles [m] 3rd	#images with error < 18.3m	#images with error > 400 m
direct	50	*1.9*	*0.5*	*7.6*	*674*	12
combined		1.3	0.4	5.3	707	10
afterwards		1.3	0.4	6.1	693	15
direct	100	*2.0*	*0.6*	*10.4*	*664*	12
combined		1.3	0.4	6.1	694	8
afterwards		1.3	0.4	5.2	688	16
direct	150	*2.2*	*0.6*	*9.8*	*647*	*16*
combined		1.3	0.4	5.8	696	13
afterwards		1.4	0.4	5.2	697	14
direct	200	*2.4*	*0.7*	*11.0*	*655*	*13*
combined		1.3	0.4	5.6	700	9
afterwards		1.4	0.4	5.6	693	10
direct	250	*2.6*	*0.7*	*11.5*	*642*	*17*
combined		1.3	0.4	5.6	700	12
afterwards		1.3	0.4	5.5	694	13
P2F [10]		9.3	7.5	13.4	655	-
all desc. [11]		1.4	0.4	5.9	685	16
int. mean [11]		1.3	0.5	5.1	675	13

used yet.) Fig. 5 shows the mean number of registered images and the mean registration times for the three datasets. The results clearly validate our approach, showing that the additional 3D-to-2D matches found through active search are instrumental for achieving a similar registration performance as tree-based search. Fig. 5 also shows that our method is slower than pure 2D-to-3D matching [11], observing an increase in registration times by 0.4s on the Vienna dataset. Note that the higher increase in registration times for the Vienna dataset is caused by RANSAC. Although active search finds the matches required to register additional images, the inlier ratio for these newly located images is rather low. Still, our approach is substantially faster than the 3 seconds required by kd-tree search. Of particular interest is the Rome dataset, with a noticeable increase in registration performance from 980 to 990 images compared to tree search. The reason is that the Rome dataset contains the most descriptors. The resulting descriptor space is so dense that the SIFT ratio test also tends to remove many correct matches. Active correspondence search is able to recover these lost matches and offers better registration performance than tree search.

Among the three prioritization strategies, the *direct* strategy that prefers visibility information offers the best registration performance. As discussed in Sec. 3, it is however susceptible to finding many correspondences in a small part of the image, resulting in a worse localization accuracy, as shown in Tab. 2 for Dubrovnik. Comparing the other two strategies with the location accuracies from [10, 11] shows that we do not trade registration performance for localization accuracy, as the additionally registered images do not result in larger localization errors. Tab. 2 shows that our combination of 2D-to-3D and 3D-to-2D matching clearly outperforms pure 3D-to-2D search [10] in terms of accuracy, as evident by a decrease of 8 meters in the median localization error.

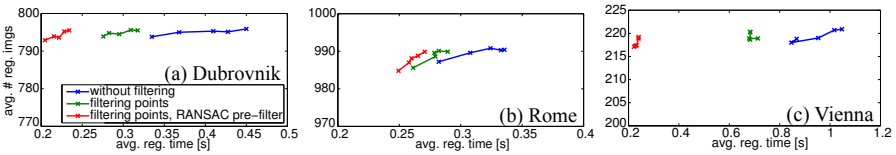

Fig. 6. Influence of the proposed filters on registration performance and efficiency. The *combined* strategy and different values for N_{3D} from $\{50, 100, 150, 200, 250\}$ were used to create the curves, with the rightmost point belonging to $N_{3D} = 250$. Filtering significantly reduces the registration times while slightly decreasing registration performance.

The experiments show that the *combined* strategy offers a good compromise between the registration performance of the *direct* and the localization accuracy of the *afterwards* prioritization. This demonstrates that both appearance (*direct*) and visibility (*afterwards*) information are equally important for localization. We therefore use the *combined* strategy for the following experiments.

Faster Registration Using Filtering. Although the focus in this paper is on registration performance, we are nonetheless interested in an as-fast-as-possible localization method. The point filter and RANSAC pre-filter described in Sec. 5 were designed for this purpose. In this experiment, we evaluate the impact of the filters depending on the value of N_{3D} for the *combined* strategy. Fig. 6 shows the effect of applying the filters on registration performance and run-times. As expected, using the filters reduces the mean registration performance slightly. This decrease by at most 2 images for a given value of N_{3D} is insignificant compared to the gain in efficiency. We note that the behavior of the filters differs for the datasets. The RANSAC pre-filter has the largest impact on the Vienna dataset (*c.f.* Fig 6(c)), which contains the fewest descriptors among all datasets. The 2D-to-3D matching part of our approach is thus more likely to find wrong correspondences by chance. Since the wrong matches are distributed all over the model, the RANSAC pre-filter can remove most of them. In contrast, the effect of the filters is less pronounced on the Rome dataset (*c.f.* Fig 6(b)). The Rome dataset contains multiple reconstructions of different landmarks, each consisting of images taken from very similar viewpoints. Thus many points are visible together in at least one image. This limits the effectiveness of the RANSAC pre-filter. Furthermore, the denser descriptor space of the dataset makes it less likely to find wrong matches which could be removed by the pre-filter.

Based on the results from Fig. 6, we fix $N_{3D} = 200$ (second to rightmost point on the curves) for the following experiments as higher values decrease the efficiency, while lower values yield worse registration performance.

Using Camera Sets. The reason that the filters decrease registration performance is that the images in the reconstruction offer only a discrete approximation to the regions in which a point is visible. In Sec. 5, we therefore proposed to use a more continuous representation by clustering sets of cameras. Fig. 7 shows the results for the *combined* strategy with $N_{3D} = 200$, both filters, and values from $\{0, 5, 10, 15, 20\}$ for the number k of nearest cameras considered for generating

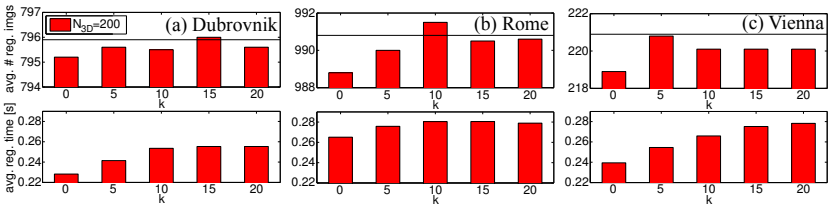

Fig. 7. Using camera sets instead of the original images improves the registration performance. Results (*combined* strategy, $N_{3D} = 200$, both filters) are shown for (a) Dubrovnik, (b) Rome, and (c) Vienna, where $k = 0$ uses the original images. The black line denotes the registration performance without filtering. The choice of k is data dependent, but $k = 10$ yields a good balance between performance and efficiency.

the sets. As predicted, using camera sets improves registration performance compared to using the original images ($k = 0$). At the same time, it increases the registration times slightly, since less points and matches can be filtered. As more and more unrelated cameras are clustered together with increasing k, the performance gain decreases. Since the clustering strongly depends on the distribution of camera positions, the actual choice of k is dataset dependent. However, values from the set $\{5, 10, 15\}$ all provide better registration performance than $k = 0$, while still yielding faster registration times on all datasets but Rome compared to not using filtering. Therefore, we propose to use $k = 10$ as a good compromise.

Comparison with State-of-the-Art. Table 3 compares our proposed approach (using the *combined* strategy with $N_{3D} = 200$, $k = 10$ and both filters) with published results of current state-of-the-art methods. P2F denotes the 3D-to-2D matching by Li *et al.*, where P2F+F2P also performs 2D-to-3D matching if P2F fails [10]. Since results for the method from [1] are only available for Vienna, Li *et al.* tested a vocabulary tree-based approach on Dubrovnik and Rome, using either all features in the images or only those belonging to 3D points [10].

Our approach achieves the same or better registration performance than tree-based search. At the same time, it is on average one order of magnitude faster due to the filtering steps. Compared to [10], our approach achieves significantly better registration performance, being able to register at least 5% more images on each dataset. Compared to [11], a similar performance increase is achieved for Vienna while the gain is smaller on the other datasets as the room for improvements is smaller. We notice that only 0.6% of all images for Dubrovnik and Rome could not be registered at least once during the 10 repetitions of the experiments. Thus our approach effectively solves both datasets. The better performance does not come at the expense of efficiency, *e.g.*, the registration times for Vienna are decreased by 190ms compared to [11]. On the other datasets, we achieve similar registration times as [11]. Furthermore, the proposed RANSAC pre-filter can help us to significantly reduce rejection times as well, saving more than 1.1s on both Dubrovnik and Vienna. The results show that our combination of both matching directions outperforms approaches matching only into one direction.

Table 3. Comparison of the method proposed in this paper ($N_{3D} = 200$, *combined* prioritization, both filters, and $k = 10$) with current state-of-the-art approaches. We achieve superior registration performance at registration times comparable to or faster than the other methods. Except for [1], mean registration and rejection times are given.

Method	Dubrovnik			Rome			Vienna		
	# reg. images	registr. time [s]	reject. time[s]	# reg. images	registr. time [s]	reject. time [s]	# reg. images	registr. time [s]	reject. time [s]
active search	**795.5**	**0.25**	**0.56**	**991.5**	0.28	2.14	**220.1**	**0.27**	**0.52**
all desc. [11]	783.9	0.31	2.22	976.9	0.29	1.90	207.7	0.50	2.40
int. mean [11]	782.0	0.28	1.70	974.6	**0.25**	**1.66**	206.9	0.46	2.43
P2F [10]	753	0.73	2.70	921	0.91	2.93	204	0.55	1.96
P2F+F2P [10]	753	0.70	3.96	924	0.87	4.67	205	0.54	3.62
Voc. tree (all)[10]	668	1.4	4.0	828	1.2	4.0	-	-	-
Voc. tree (pts.)[10]	677	1.3	4.0	815	1.2	4.0	-	-	-
Voc. tree GPU[1]	-	-	-	-	-	-	165	$\leq$ **0.27** (worst case)	
kd-tree [11]	**795**	3.4	14.45	983	3.97	6.27	**220**	3.44	2.72

Table 4. Localization errors for our proposed method on Dubrovnik (*combined* strategy, $N_{3D} = 200$, $k = 10$, both filters). Although our approach is able to register more images than the competing methods, it maintains a similar localization accuracy.

Method	# reg. images	Median [m]	Quartiles [m]		#images with error	
			1st	3rd	< 18.3m	> 400 m
active search	**795.5**	1.4	**0.4**	5.3	**704**	**9**
all desc. [11]	783.9	1.4	**0.4**	5.9	685	16
int. mean [11]	782.0	**1.3**	0.5	**5.1**	675	13
P2F [10]	753	9.3	7.5	13.4	655	-

Finally, Tab. 4 compares the localization accuracy of our approach with the methods from Li *et al.* and Sattler *et al.* Our approach achieves a similar accuracy as [11] for the mean and 3rd quartile errors, while it can register 19 more images with an error < 18.3m. This shows that the better registration performance of our algorithm does not come from badly localized images.

7 Conclusion and Future Work

In this paper we have presented an active search approach that efficiently finds those additional correspondences needed to close the gap in registration performance between fast methods and slower tree-based search that has been observed in [11]. The resulting combination of 2D-to-3D and 3D-to-2D matching achieves superior registration performance on three standard benchmark datasets. It is one order of magnitude faster than tree-based search, while yielding comparable or better run-times than published methods. We show the advantage of combining both search directions compared to matching in only a single direction.

The results on the Rome dataset indicate that our approach offers a better scalability than, *e.g.*, tree-based search. We plan to investigate this behavior in more detail. Our method will also benefit from better visibility prediction.

Acknowledgments. This project has been funded by UMIC (DFG EXC 89).

References

1. Irschara, A., Zach, C., Frahm, J.-M., Bischof, H.: From structure-from-motion point clouds to fast location recognition. In: CVPR (2009)
2. Chen, D.M., Baatz, G., Köser, K., Tsai, S.S., Vedantham, R., Pylvänäinen, T., Roimela, K., Chen, X., Bach, J., Pollefeys, M., Girod, B., Grzeszczuk, R.: City-scale landmark identification on mobile devices. In: CVPR (2011)
3. Cummins, M., Newman, P.: FAB-MAP: Probabilistic Localization and Mapping in the Space of Appearance. IJRR 27, 647–665 (2008)
4. Wendel, A., Irschara, A., Bischof, H.: Natural landmark-based monocular localization for mavs. In: ICRA (2011)
5. Castle, R.O., Klein, G., Murray, D.W.: Video-rate localization in multiple maps for wearable augmented reality. In: ISWC (2008)
6. Klein, G., Murray, D.: Parallel tracking and mapping for small AR workspaces. In: ISMAR (2007)
7. Crandall, D., Owens, A., Snavely, N., Huttenlocher, D.P.: Discrete-continuous optimization for large-scale structure from motion. In: CVPR (2011)
8. Alcantarilla, P.F., Ni, K., Bergasa, L.M., Dellaert, F.: Visibility learning in large-scale urban environment. In: ICRA (2011)
9. Avrithis, Y., Kalantidis, Y., Tolias, G., Spyrou, E.: Retrieving landmark and non-landmark images from community photo collections. In: ACM Multimedia (2010)
10. Li, Y., Snavely, N., Huttenlocher, D.P.: Location Recognition Using Prioritized Feature Matching. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010, Part II. LNCS, vol. 6312, pp. 791–804. Springer, Heidelberg (2010)
11. Sattler, T., Leibe, B., Kobbelt, L.: Fast image-based localization using direct 2d-to-3d matching. In: ICCV (2011)
12. Schindler, G., Brown, M., Szeliski, R.: City-scale location recognition. In: CVPR (2007)
13. Zamir, A.R., Shah, M.: Accurate Image Localization Based on Google Maps Street View. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010, Part IV. LNCS, vol. 6314, pp. 255–268. Springer, Heidelberg (2010)
14. Nister, D., Stewenius, H.: Scalable recognition with a vocabulary tree. In: CVPR (2006)
15. Philbin, J., Chum, O., Isard, M., Sivic, J., Zisserman, A.: Object retrieval with large vocabularies and fast spatial matching. In: CVPR (2007)
16. Hartley, R.I., Zisserman, A.: Multiple View Geometry in Computer Vision, 2nd edn. Cambridge Univ. Press (2004)
17. Fischler, M., Bolles, R.: Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Comm. ACM 24, 381–395 (1981)
18. Eade, E., Drummond, T.: Scalable monocular slam. In: CVPR (2006)
19. Lowe, D.: Distinctive image features from scale-invariant keypoints. IJCV 60, 91–110 (2004)
20. Philbin, J., Chum, O., Isard, M., Sivic, J., Zisserman, A.: Lost in quantization: Improving particular object retrieval in large scale image databases. In: CVPR (2008)
21. Muja, M., Lowe, D.G.: Fast approximate nearest neighbors with automatic algorithm configuration. In: VISAPP (2009)
22. Chum, O., Matas, J.: Optimal Randomized RANSAC. PAMI 30, 1472–1482 (2008)