# Image Retrieval for Image-Based Localization Revisited

Torsten Sattler[1]
tsattler@cs.rwth-aachen.de
Tobias Weyand[2]
weyand@vision.rwth-aachen.de
Bastian Leibe[2]
leibe@vision.rwth-aachen.de
Leif Kobbelt[1]
kobbelt@cs.rwth-aachen.de

[1] Computer Graphics Group
RWTH Aachen University, Germany
[2] Computer Vision Group
RWTH Aachen University, Germany

**Abstract**

To reliably determine the camera pose of an image relative to a 3D point cloud of a scene, correspondences between 2D features and 3D points are needed. Recent work has demonstrated that directly matching the features against the points outperforms methods that take an intermediate image retrieval step in terms of the number of images that can be localized successfully. Yet, direct matching is inherently less scalable than retrieval-based approaches. In this paper, we therefore analyze the algorithmic factors that cause the performance gap and identify false positive votes as the main source of the gap. Based on a detailed experimental evaluation, we show that retrieval methods using a *selective voting* scheme are able to outperform state-of-the-art direct matching methods. We explore how both selective voting and correspondence computation can be accelerated by using a Hamming embedding of feature descriptors. Furthermore, we introduce a new dataset with challenging query images for the evaluation of image-based localization.

## 1 Introduction

Image-based localization is the task of determining the exact location from which a query photo was taken. With the widespread adoption of mobile camera phones, image-based localization has recently received increased attention as it enables many interesting applications such as mobile landmark recognition [9], real-time camera pose tracking [13, 21, 37], robot navigation [19], or community-based city reconstruction [36].

If the full camera pose, *i.e.*, both position and orientation, is required, information about the 3D structure of the scene needs to be known. Given a set of images, such a 3D model can be computed efficiently using modern Structure-from-Motion (SfM) approaches [8, 10, 31, 33]. The camera pose can then be computed from 2D-to-3D correspondences between 2D image features and 3D points [12]. Due to the reconstruction process, feature descriptors are available for every 3D point. Thus, correspondences can be found by finding nearest neighbors for image descriptors and the image-based localization problem becomes a descriptor matching problem. Previous approaches to this search problem can be divided into


http://dx.doi.org/10.5244/C.26.76

two main classes. Approaches based on *image retrieval* [25, 30] first find images similar to the query image. Since features in the database images are projections of 3D points, 2D-to-3D correspondences can be established by matching the query image against the database images [13]. *Direct matching* approaches skip this image matching step and instead try to find correspondences by directly matching image features against 3D scene points. As reported in [17, 28], direct matching methods achieve a better localization performance than retrieval-based approaches as they are able to localize more query images. However, this performance gain comes at the cost of memory consumption, since direct methods require to store the descriptors of the 3D points in memory. In contrast, the inverted file indices and feature geometry needed for image retrieval can be stored very efficiently [15, 24].

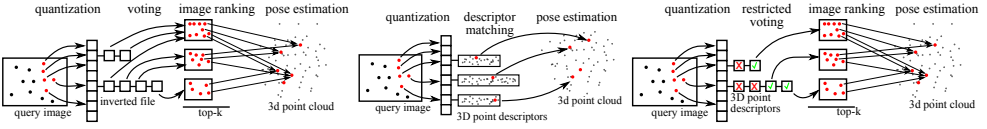
Due to their memory efficiency, image retrieval-based methods offer a better scalability. In this paper, we therefore analyze the performance gap between retrieval-based approaches and direct matching documented in [17, 28]. We discuss the factors that limit the effectiveness of retrieval-based methods and experimentally determine their influence on the localization performance. We show that *selective voting* schemes designed to handle the main error source of retrieval approaches, false positive votes, are not only able to close the performance gap but can even outperform the state-of-the-art in direct matching. Using Hamming embedding [14] of the SIFT descriptor space, such schemes are both time and memory efficient, even compared to classic image retrieval. Furthermore, we show that we can drastically improve the efficiency of the pose estimation step using quantization and Hamming embedding and we point out the resulting tradeoffs between performance and computational costs.

Our experiments are based on two large-scale image based localization datasets: The Vienna dataset from [13], consisting of 1.1M 3D points and 266 query images and our novel Aachen dataset, consisting of 1.5M 3D points and 369 query images. [1]

**Related Work.** Early works on image-based localization exhaustively compared a query image against all database images [27, 41]. Using image-retrieval techniques based on descriptor quantization, inverted file scoring, and fast spatial matching [22, 25, 30], larger datasets can be handled. Schindler *et al*. select only informative features [29] to construct a vocabulary tree [22] while Knopp *et al*. remove confusing features [16]. By aggregating multiple database images into scene maps, Avrithis *et al*. increase recall and decrease the length of the inverted files [1]. Torii *et al*. exploit the spatial relationship between database images by interpolating between views [35]. Zamir & Shah observe that using the original SIFT descriptors to vote for camera locations increases localization performance [40].

Further information, such as building outlines or 3D geometry, can be combined with the image database to improve localization approaches [2, 4]. When the coverage of the scene by the images is dense enough, SfM techniques can be used to reconstruct a 3D point cloud of the scene [8, 10, 31, 33]. Based upon such a 3D model, Irschara *et al*. propose an efficient, GPU-based image retrieval approach for accurate localization [13]. To handle a larger variety of viewpoints for query images, they generate additional database images by placing virtual cameras on a ground plane. Wendel *et al*. consequently generalize this concept to virtual views in full 3D to handle image-based localization for flying vehicles [37]. Li *et al*. show that a prioritized, direct matching of 3D points to 2D features using feature descriptors vastly improves the localization performance compared to image retrieval-based methods [17]. Another direct matching approach by Sattler *et al*. improves the performance even further [28]. Interestingly, the method from [28] uses data structures similar to retrieval systems, such as a visual vocabulary and inverted files. We use this similarity to identify the

[1] The dataset is available at http://www.graphics.rwth-aachen.de/localization .




(a) (b) (c)

Figure 1: Image-based localization methods. (a) Image retrieval, (b) Direct matching [28], (c) *Selective voting* as proposed in this paper.

reasons for the performance gap between image retrieval methods and direct matching.

In the next section, we compare the method by Irschara *et al*. [13] as an example for a retrieval-based system to the approach by Sattler *et al*. [28]. Based on the discussion of both works, we analyze the possible reasons for the shortcomings of image retrieval in Section 3 and point out possible solutions. We thoroughly evaluate these possible solutions in Section 4. Section 5 concludes the paper by offering a discussion of the results.

## 2 Retrieval Vs. Direct Matching

In a 3D point cloud obtained from SfM, every 3D point is created from 2D features found in the database images. Thus, we associate a 3D point with the descriptors of its corresponding 2D features. Following [17, 28], we will refer to 2D features and their descriptors as *features* and 3D points and their associated descriptors as *points*. Since the same type of descriptors is used, finding 2D-to-3D correspondences is a descriptor matching problem. After establishing matches, the camera pose is estimated using RANSAC [9] in combination with an $n$-point-pose algorithm [9, 11, 17]. A camera is considered as localized or registered against the model if the best pose found by RANSAC has a certain number of inliers.

Both Irschara *et al*. and Sattler *et al*. use a *visual vocabulary*, obtained by clustering SIFT descriptors [18] using k-means [22, 25]. Each cluster defines a *visual word* and descriptors are assigned to the closest cluster center through approximate nearest neighbor search [20]. Since a 3D point consists of multiple descriptors, it is usually assigned to multiple words.

**Localization by Retrieval.** There are fundamental differences between classical image retrieval systems [7, 25, 30] and image retrieval approaches for image-based localization. The former aim at finding as many relevant database images as possible for a given query image. For the latter, it is sufficient to retrieve just one relevant image if the correspondences found by matching query and database image suffice to estimate the camera pose of the query image. Recall-improving techniques such as query expansion [7] thus do not improve the performance of image-based localization. Furthermore, fast, approximate spatial verification approaches [25, 34] cannot be used since we are interested in accurately estimating the full camera pose. For mobile applications of image-based localization, the localization method needs to be as fast as possible. We therefore prefer efficiency to accuracy, *i.e*., we will accept a slight drop in the number of images that can be localized if it improves the run-time.

To establish the correspondences for camera pose estimation, Irschara *et al*. perform image retrieval to find similar views in a set of database images [13]. Given a new query image, SIFT features are extracted and each feature is assigned to visual words using a vocabulary tree [22]. Each entry in the inverted file of an activated visual word then votes for a database image. Based on this voting, only the top-k ranked images are kept for further processing. Applying *regular SIFT matching* for every top-ranked view, the SIFT features belonging to 3D points are matched against the query image and correspondences are established using

the SIFT ratio test, followed by pose estimation. Fig. 1(a) illustrates this approach.

There are three major differences between [13] and classical image retrieval systems. First, instead of using *tf∗idf* weighting [30] to vote for images, Irschara *et al*. use a *probabilistic ranking*: Given a database image $\mathcal{D}$ that received $k$ votes from a query image $\mathcal{Q}$ with $|\mathcal{Q}|$ features, its score is the ratio $P(\#\text{votes} = k \mid \mathcal{Q} \equiv \mathcal{D}) / P(\#\text{votes} = k \mid \mathcal{Q} \not\equiv \mathcal{D})$, where $\mathcal{Q} \equiv \mathcal{D}$ denotes the event that $\mathcal{Q}$ and $\mathcal{D}$ depict the same scene [13]. The second difference is in the use of synthetic images. By placing virtual cameras onto the ground plane of the model, new views can be generated that help improve registration performance. Third, only features in the database images belonging to 3D points are used.

**Localization by Direct Matching.** Irschara *et al*. limit the search space for 2D-to-3D correspondences by quantizing view directions [13]. The direct matching approach by Sattler *et al*. directly establishes matches between features and points and thus does not require the query images to be similar enough to database images [28]. To speed up the correspondence search, a *quantized SIFT matching* is applied: For a given query feature $f$ with SIFT descriptor $s_f$, only points whose descriptors are assigned to the same visual word are considered. Inside a word, Sattler *et al*. find the two nearest neighboring points $p$ and $q$, $p \neq q$, with descriptors $s_p$ and $s_q$, and $\|s_f - s_p\|_2 \leq \|s_f - s_q\|_2$ using linear search. A correspondence between $f$ and $p$ is established if the SIFT ratio test $\|s_f - s_p\|_2 < 0.7 \cdot \|s_f - s_q\|_2$ is passed. If multiple correspondences containing the same point are found, only the one with the smallest descriptor distance between point and feature is kept. The matches obtained this way are again used to estimate the camera pose with RANSAC. To quickly determine the relevant points and descriptors, Sattler *et al*. use an inverted file that maps each visual word to its corresponding set of point ids and SIFT descriptors [28]. Fig. 1(b) illustrates this approach.

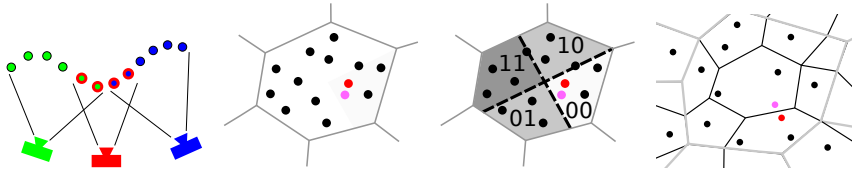
**Comparison.** Sattler *et al*. need to keep the actual feature descriptors in memory at all time. To reduce these memory requirements, they cluster descriptors. If more than one descriptor of the same 3D point is assigned to a word, only the mean descriptor, with entries rounded to the nearest integer value, is stored, reducing memory requirements by about 1/3 [28]. Still, every inverted file entry contains a SIFT descriptor, which adds 128 bytes per entry compared to classical inverted files. Computing the Euclidean distance for every entry in an inverted file instead of casting a vote introduces additional computational cost. In theory, direct matching will be two orders of magnitude slower than retrieval, but in practice the gap is smaller due to the vectorization performed by modern CPUs. Still, the memory requirements and additional computations hinder the scalability of [28] compared to [13].

However, Sattler *et al*. report that the localization performance of their method is at least 15% better than image retrieval-based methods [28]. This gap is especially interesting considering the similarity of both pipelines (*cf*. Fig. 1(a)-(b)). Finding the reason behind this gap, and ways to close it, are the aims of this paper.

# 3 Image Retrieval Revisited

There are two key problems for image retrieval-based methods that can cause the gap in registration performance: Incorrect votes might corrupt the re-ranking process, while using pairwise image matching to establish 2D-to-3D correspondences might restrict the set of views that can be registered. In the following, we discuss both aspects and point out solutions to the problems. Furthermore, we explore different ways to speed up the matching process.

**Selective Voting.** In image retrieval-based methods, every feature votes for all database images containing its visual word. Since a feature can correspond to at most one 3D point,




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

Figure 2: (a) *View quantization* reduces the number of matches that can be found. (b) Unrelated entries in a word cause *incorrect votes*. (c) Hamming embedding can be used to discard incorrect votes. (d) Using a coarser quantization improves *correspondence selection*.

this one-to-many mapping results in many unnecessary and incorrect votes. Fig. 2(b) illustrates this problem. The query feature (pink) is assigned to the same visual word as its corresponding 3D point (red). But instead of only voting for the images observing the point, the feature also votes for all other images in the inverted file of the word, even if the corresponding descriptors are very far away (black points). Since the majority of the votes will be for unrelated images, dealing with these incorrect votes is challenging even for advanced re-ranking schemes such as *tf∗idf* weighting or *probabilistic ranking*. Since pose estimation is only attempted for the top-$k$ images, failure to rank any of the relevant images among the top-$k$ will have a negative impact on localization performance.

Employing a finer vocabulary helps reduce the number of wrong votes, but also reduces the probability that a feature and its 3D point are assigned to the same word due to quantization errors. Instead, we propose to use a *selective voting* mechanism, independently of the chosen quantization. As illustrated in Fig. 1(c), votes are cast only for a selected subset of entries in the inverted file of a visual word. Images are then ranked in decreasing order of the number of votes and only the top-k ranked images are considered for pose estimation.

One possible selection criterion can be derived from the *quantized SIFT matching* [28] (*cf*. Sec. 2). A feature $f$ assigned to visual word $w$ casts a vote for a database image $I$ if $I$ contains a point $p$ whose descriptor $s_{p,I}$ is also mapped to $w$, and $f$ and $p$ form a correspondence found by *quantized SIFT matching*. In the following, we refer to this particular approach of *selective voting* as *correspondence voting*.

Unfortunately, *correspondence voting* again requires us to store and compare SIFT descriptors. Jégou *et al*. propose to use Hamming embedding to cast votes only for a subset of all entries in a visual word [14]. Hamming embedding first projects every SIFT descriptor $s_j$ into a $d$-dimensional space using a random, orthogonal projection matrix $P \in \mathbb{R}^{d\times 128}$. Let $p_j = P \cdot s_j$ be the projection of a descriptor $s_j$ assigned to visual word $w$ and let $t_w \in \mathbb{R}^d$ be a vector of thresholds defined for $w$. We denote the $n^{\text{th}}$ component of a vector $v$ as $v(n)$. A binary representation $b_j \in \{0,1\}^d$ for $s_j$ is obtained by thresholding $p_j$, setting $b_j(n) = 1$ if $p_j(n) > t_w(n)$ and $b_j(n) = 0$ otherwise [14]. $t_w(n)$ is given by $t_w(n) = \text{median}(\{p_j(n) \mid s_j \in S(w)\})$, where $S(w)$ denotes the descriptors of the 3D points assigned to word $w$.

Using Hamming embedding instead of SIFT descriptors yields *Hamming voting*, another selective voting approach. Let $f$ be a feature assigned to visual word $w$, with binary representation $b_f$. Consider a point $p$, observed in image $I$ with descriptor $s_{p,I}$, also assigned to $w$. Then $f$ votes for $I$ if the Hamming distance between $b_f$ and $b_{s_{p,I}}$ is below a certain distance threshold. Fig. 2(c) illustrates a Hamming embedding with $d = 2$ together with the resulting binary strings. Using *Hamming voting*, the votes cast by the pink query feature can be restricted to include only points with similar Hamming descriptors, avoiding many incorrect votes. While additional computational costs for projection and thresholding occur

compared to image retrieval, using Hamming descriptors instead of SIFT descriptors significantly reduces memory requirements, *e.g.*, by a factor of 16 when using $d = 64$ dimensions. Furthermore, modern CPUs often offer specialized instructions to compute Hamming distances [4]. Using SSE4.2 together with the POPCNT instruction, one million Hamming distance computations for $d = 64$ dimensions can be done in about 2ms. In our experiments, projecting and thresholding the features took less than 23ms on average for a query image.

**View Quantization.** Since correspondences are obtained by matching features in the query image against points in a database image, both images need to share enough matches to facilitate pose estimation, *i.e.*, the overlap between both images has to be large enough. This quantizes the viewpoints from which query images can be registered as shown in Fig. 2(a). In this example, the red camera does not share enough common points with either database camera to allow estimating its pose. Both synthetic views [13] and the view interpolation from [35] can provide possible solutions to this problem. However, in this paper, we will focus on *selective voting* and ignore the effects of view quantization.

**Correspondence Selection.** Irschara *et al*. perform a *regular SIFT matching* between the query and the top-ranked database image to obtain correspondences between features and points [13]. For every point in the database image they find its two (approximate) nearest neighboring features in the query image and establish matches using the SIFT ratio test.

*Regular SIFT matching* introduces additional computational costs, especially if search structures have to be build. To save computation time, information from the voting stage could be re-used for correspondence selection. The simplest strategy, *visual word matching*, establishes a match between a point and a feature if they are mapped to the same visual word (without considering the actual descriptors). Although requiring no additional computations, this approach is likely to produce many wrong correspondences. Similar to *correspondence voting*, *quantized SIFT matching* [28] can be used to remove many of these wrong matches. *Quantized SIFT matching* is likely to miss matches due to quantization errors introduced by the fine vocabularies required for voting. This is shown Fig. 2(d), where the query feature (pink) and its corresponding point (red) are mapped to different words in the fine vocabulary (black lines). Using a coarser vocabulary (gray lines) instead decreases the likelihood of missed matches, but introduces additional computations since the words in the coarser vocabulary contain more points (black points). Building the coarser vocabulary on top of the fine one, the assignments to the coarser vocabulary are given by the mapping to the finer one and require no additional computations.

*Quantized Hamming matching* uses the idea of *quantized SIFT matching* and replaces SIFT descriptors with binary strings. A correspondence between two features $f$, $g$ mapped to the same word $w$ is establish if $b_f$ is the nearest neighbor of $b_g$ and the Hamming distance between $b_f$ and $b_g$ is below a certain threshold. If Hamming voting is used, this method can re-use both the vocabulary tree quantization and the random projections of the descriptors. The only computation required is thus the thresholding for the coarser vocabulary.

For all correspondence selection schemes, we match the 3D points contained in the database images against the 2D features in the query image.

# 4 Evaluation

In this section, we evaluate the impact of different selective voting strategies on the problem of incorrect votes on two large scale datasets. We further evaluate different correspondence selection schemes. We obtained two visual vocabularies, one containing 100k and the other

| | # Cameras | # Points | # Descriptors | # Query Images | Mean # Features per Query Image |
|---|---|---|---|---|---|
| Aachen | 3047 | 1,540,786 | 7,281,501 | 369 | 8648.66 |
| Vienna | 1324 | 1,123,028 | 4,854,056 | 266 | 9707.29 |

Table 1: Statistics of the datasets used for the experimental evaluation.

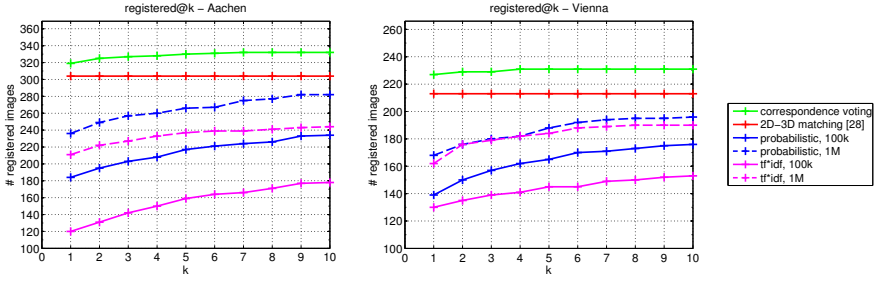



Figure 3: Registration performance for different ranking schemes compared with direct 2D-3D matching. The *correspondence voting* scheme is able to achieve significantly better results than standard ranking schemes due to its ability to discard incorrect votes.

containing 1M visual words, from approximate k-means clustering [25]. To compute the visual word assignments, we use a single kd-tree built using the FLANN library [20] and stop the search for the closest word after visiting at most 10 leaves. For retrieval-based methods, only features in the database images corresponding to 3D points are used. To implement *regular SIFT matching*, we use the FLANN library to construct a single kd-tree for every query image. Correspondences between the points visible in a database image and the features in the query image are established using approximate search in the tree, visiting at most 300 leaves, followed by the SIFT ratio test with threshold 0.7. If multiple matches for the same feature are found, we keep only the one with the smallest distance between its descriptors. The camera pose of the query images is estimated from these matches using the 6-point DLT algorithm [12] inside a RANSAC loop [5, 6]. As in [17, 28], a query image is considered registered if the best pose found by RANSAC has at least 12 inliers. Similar to [13], we only consider the top-10 ranked images.

**Datasets.** We use two datasets for experimental evaluation. Table 1 gives a brief overview over their important statistics. The Vienna dataset has already been used in [13, 17, 28] and thus allows a fair comparison with current state-of-the-art localization methods. We furthermore introduce a new dataset, covering the historic center of the city of Aachen, Germany. 3047 high-resolution pictures were taken with different cameras. An SfM reconstruction of the scene was obtained using the freely available Bundler software [31, 39] and SiftGPU [38]. To obtain a metric reconstruction in which distances can be measured in meters, the 3D model was aligned to the building outlines of the city obtained from [23] using the approach from [33]. A set of 369 query images for the dataset was taken using the camera of a Motorola Milestone mobile phone, exhibiting the typical image quality shortcomings of mobile phone cameras such as motion blur, lack of focus and rolling shutter artifacts. The images were taken at different times of day and year over a period of two years. As a result, they cover a wide range of lighting and weather conditions as well as temporary occlusions by construction sites and changes in architecture not present in the 3D model. All query images used in this paper have a maximal dimension of $1600 \times 1600$ pixels.

**Impact of Incorrect Votes.** In the first experiment, we evaluate the impact of incorrect votes on image retrieval-based localization. As a baseline, we use the 2D-3D matching approach

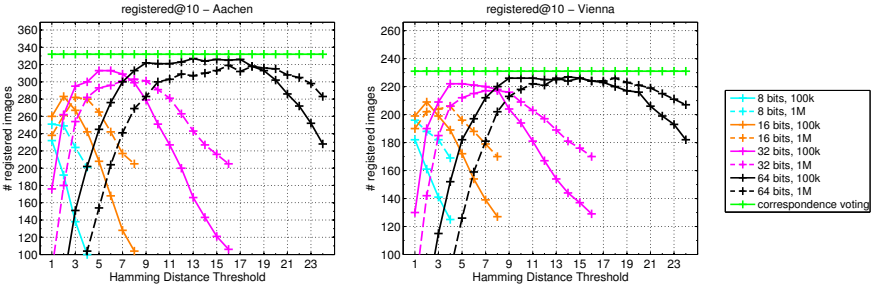



Figure 4: *Hamming voting* using 32- and 64-bit Hamming vectors achieves nearly the same performance as *correspondence voting*, which utilizes SIFT descriptors.

from [28], where we disabled the early termination of correspondence search and instead opted to find as many correspondences as possible. For image retrieval, we compare *tf∗idf*-weighted ranking [30] with probabilistic ranking [13] using both vocabularies. For both retrieval methods, *regular SIFT matching* is used to establish the correspondences needed for pose estimation. Fig. 3 compares the two ranking methods with the baseline. There is a significant gap in registration performance between direct matching (304 images (82%) for Aachen and 213 images (80%) for Vienna) and the two ranking schemes (at most 280 images (76%) for Aachen and less than 200 images (75%) for Vienna). To verify that this gap is caused by incorrect votes, we evaluate *correspondence voting* using the 100k vocabulary and direct SIFT matching. Note that *correspondence voting* uses the same matches found by [28] to vote for the database images. As can be seen in Fig. 3, this *selective voting* scheme achieves much better results than the other two ranking methods due to its ability to discard incorrect votes. It even outperforms 2D-3D matching by registering 332 images (90%) for Aachen and 231 images (87%) for Vienna. This performance increase is caused by the more detailed matching performed to establish correspondences, which is able to recover matches lost in [28] due to quantization. In contrast to *correspondence voting*, 2D-3D matching is not bound to specific views. Thus, the results also show that the gap can be closed without handling *view quantization*, justifying our decision to focus on *selective voting*.

**Correspondence Voting *vs*. Hamming Voting.** To reduce memory requirements and distance computation times of *correspondence voting*, we proposed *Hamming voting*. In this experiment, we compare both *selective voting* approaches, again using the top-10 ranked images and *regular SIFT matching* to establish matches. Fig. 4 shows the results for using different numbers of bits for the Hamming embedding, both vocabularies, and different thresholds to restrict the voting for images. As can be seen, using 32-bit or 64-bit Hamming vectors instead of SIFT descriptors has only a slight impact on registration performance. On the Aachen dataset, 327 images (89%) can be registered using 64-bit compared to the 332 images (90%) obtained by *correspondence voting*. Using 32-bit vectors we can still register 313 images (85%), outperforming the method from [28] while using 32 times less memory for the same number of inverted file entries. On the Vienna dataset, *Hamming voting* can register 227 (85%) respectively 222 images (83%) using 64- and 32-bit compared to the 231 images (87%) *correspondence voting* can register, again outperforming the method from [28]. Using a coarser vocabulary yields better results due to less quantization errors.

**Correspondence Selection.** We now evaluate the alternative correspondence selection approaches presented in Sec. 3, using *Hamming voting* with 64-bits and the distance threshold set to 15 for retrieval. Our goal is to keep a low false positive rate, while reducing the com-

| Matching Method | Voc. Size | # Images Registered | Correspondence Search [ms] | RANSAC ok [ms] | RANSAC err [ms] | Preprocessing Time [ms] |
|---|---|---|---|---|---|---|
| Regular SIFT | - | 320.0 (87%) | 300.3 | 0.9 | 0.0 | 30.1 |
| Quantized SIFT | 100 | 319.0 (86%) | 14.5 | 3.1 | 155.3 | - |
| | 1k | 304.0 (82%) | 2.5 | 17.4 | 2705.8 | - |
| | 10k | 246.0 (67%) | 1.0 | 10.2 | 22.5 | - |
| Quantized Hamming (64-bit) | 100 | 307.0 (83%) | 3.6 | 141.6 | 2825.0 | 3.6 |
| | 1k | 300.0 (81%) | 0.8 | 3.5 | 35.9 | 3.6 |
| | 10k | 272.0 (74%) | 0.5 | 0.9 | 0.0 | 3.9 |

Table 2: Pose estimation performance and timings per image pair for different correspondence selection methods on the Aachen dataset, averaged over 10 runs on an Intel Core i7 3.4 GHz CPU. RANSAC timings are median values over all image pairs for successful and failed pose estimation. The preprocessing time is required once per query image, building the search index for *regular SIFT* and thresholding for *quantized Hamming*.

putation time needed for pose estimation. To obtain the coarser vocabularies needed for the quantized matching approaches, we built a vocabulary tree with branching factor 10 on top of the 100k vocabulary. Coarser levels in the tree thus define coarser vocabularies. Using the vocabulary to compute the word assignments for voting instead of the kd-tree used in the previous experiments did not result in a significant difference in localization performance (87% *vs*. 89% for Aachen and 86% *vs*. 85% for Vienna when using *regular SIFT matching*).

Tab. 2 shows results for the different methods on the Aachen dataset. Fig. 5 compares the distributions of inlier ratios for the registered query images. *Regular SIFT matching* (*RSM*) [13] yields the best performance, being able to register 320 (87%) and 228 images (86%) on Aachen and Vienna. Due to the low false positive rate, RANSAC-based pose estimation is very fast, but matching takes on average 300ms per image pair, and an additional 30ms per query image are necessary to build the kd-tree for efficient feature matching. As expected, *visual word matching* generates the most false positives (Fig. 5, left), which reduces the number of registered images to 220 (60%) and 197 (74%) on Aachen and Vienna, respectively. *Quantized SIFT matching* (*QSM*) [28] drastically reduces the matching time by performing SIFT matching only within the visual word of the query point. The use of larger vocabularies offers a higher speedup, but also increases the false positive rate (Fig. 5, middle), requiring more RANSAC iterations. *Quantized Hamming matching* (*QHM*), with the distance threshold set to 15, yields a consistent speedup for correspondence search. For 100 visual words, the high number of false positives (Fig. 5, right) drastically slows down RANSAC. At 1k visual words, registration performances of *QHM* and *QSM* are comparable, but RANSAC is much faster for *QHM*, because *QSM* is more prone to generate false positives at larger vocabulary sizes. Since larger vocabularies contain fewer features per word, the SIFT ratio test employed by *QSM* is more likely to accept wrong matches, even for far away descriptors (*cf*. Fig. 2(d)). Due to its uses of an absolute distance threshold, *QHM* is able to avoid such matches. The vocabulary with 10k words is too fine, resulting in too many missed matches.

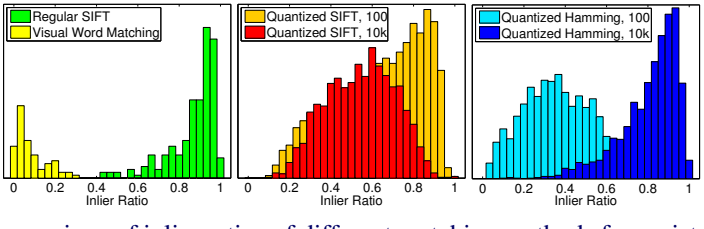



Figure 5: Comparison of inlier ratios of different matching methods for registered images.

# 5 Conclusion

In this paper, we have analyzed the performance gap observed between direct matching methods and image retrieval-based approaches for image-based localization. As shown experimentally, this gap is caused by the large number of incorrect votes cast by standard re-ranking schemes. Applying *selective voting* using SIFT descriptors combines the correspondence search of direct matching with a simple voting scheme that is able to discard those incorrect votes. Since it uses a much more detailed matching step between query images and the top-ranked database images, *correspondence voting* is even able to outperform direct matching. Its main disadvantages, memory consumption and computational overhead, can be resolved without a significant impact on registration performance by using Hamming embedding. Furthermore, we have shown that the remaining bottleneck of image retrieval-based localization, *regular SIFT matching*, can be accelerated using simpler matching schemes. We expect that both voting and correspondence selection will improve with a better Hamming embedding that can be obtained from learnt projection matrices [26, 32].

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

# References

[1] Y. Avrithis, Y. Kalantidis, G. Tolias, and E. Spyrou. Retrieving Landmark and Non-Landmark Images from Community Photo Collections. In *ACM Multimedia*, 2010.

[2] G. Baatz, K. Köser, D. Chen, R. Grzeszczuk, and M. Pollefeys. Handling Urban Location Recognition as a 2D Homothetic Problem. In *ECCV*, 2010.

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

[4] D.M. Chen, G. Baatz, K. Köser, S.S. Tsai, R. Vedantham, T. Pylvänäinen, K. Roimela, Xin Chen, J. Bach, M. Pollefeys, B. Girod, and R. Grzeszczuk. City-scale landmark identification on mobile devices. In *CVPR*, 2011.

[5] O. Chum and J. Matas. Optimal Randomized RANSAC. *PAMI*, 30(8):1472 –1482, 2008.

[6] O. Chum, J. Matas, and S. Obdržálek. Enhancing RANSAC by Generalized Model Optimization. In *ACCV*, 2004.

[7] O. Chum, J. Philbin, J. Sivic, and A. Zisserman. Total Recall: Automatic Query Expansion with a Generative Feature Model for Object Retrieval. In *ICCV*, 2007.

[8] D. Crandall, A. Owens, N. Snavely, and D. P. Huttenlocher. Discrete-continuous optimization for large-scale structure from motion. In *CVPR*, 2011.

[9] M. Fischler and R. Bolles. Random Sampling Consensus: A Paradigm for Model Fitting with Application to Image Analysis and Automated Cartography. *Comm. ACM*, 24:381–395, 1981.

[10] J-M. Frahm, P. Georgel, D. Gallup, T. Johnson, R. Raguram, C. Wu, Y-H. Jen, E. Dunn, B. Clipp, S. Lazebnik, and M. Pollefeys. Building Rome on a Cloudless Day. In *ECCV*, 2010.

[11] R.M. Haralick, C.-N. Lee, K. Ottenberg, and M. Nölle. Review and analysis of solutions of the three point perspective pose estimation problem. *IJCV*, 13(3):331–356, 1994.

[12] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge Univ. Press, 2nd edition, 2004.

[13] A. Irschara, C. Zach, J.-M. Frahm, and H. Bischof. From Structure-from-Motion Point Clouds to Fast Location Recognition. In *CVPR*, 2009.

[14] H. Jégou, M. Douze, and C. Schmid. Hamming embedding and weak geometric consistency for large scale image search. In *ECCV*, 2008.

[15] H. Jégou, M. Douze, and C. Schmid. Packing bag-of-features. In *ICCV*, 2009.

[16] J. Knopp, J. Sivic, and T. Pajdla. Avoding Confusing Features in Place Recognition. In *ECCV*, 2010.

[17] Y. Li, N. Snavely, and D. P. Huttenlocher. Location Recognition using Prioritized Feature Matching. In *ECCV*, 2010.

[18] D. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *IJCV*, 60(2), 2004.

[19] L. Meier, P. Tanskanen, F. Fraundorfer, and M. Pollefeys. Pixhawk: A system for autonomous flight using onboard computer vision. In *ICRA*, 2011.

[20] M. Muja and D. G. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. In *VISAPP*, 2009.

[21] R.A. Newcombe, S.J. Lovegrove, and A.J. Davison. Dtam: Dense tracking and mapping in real-time. In *ICCV*, 2011.

[22] D. Nister and H. Stewenius. Scalable recognition with a vocabulary tree. In *CVPR*, 2006.

[23] OpenStreetMap. http://www.openstreetmap.org/.

[24] M. Perdoch, O. Chum, and J. Matas. Efficient representation of local geometry for large scale object retrieval. In *CVPR'09*, 2009.

[25] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object Retrieval with Large Vocabularies and Fast Spatial Matching. In *CVPR*, 2007.

[26] J. Philbin, M. Isard, J. Sivic, and A. Zisserman. Descriptor learning for efficient retrieval. In *ECCV*, 2010.

[27] D. Robertson and R. Cipolla. An image-based system for urban navigation. In *BMVC*, 2004.

[28] T. Sattler, B. Leibe, and L. Kobbelt. Fast Image-Based Localization using Direct 2D-to-3D Matching. In *ICCV*, 2011.

[29] G. Schindler, M. Brown, and R. Szeliski. City-Scale Location Recognition. In *CVPR*, 2007.

[30] J. Sivic and A. Zisserman. Video Google: A Text Retrieval Approach to Object Matching in Videos. In *ICCV*, 2003.

[31] N. Snavely, S.M. Seitz, and R. Szeliski. Photo Tourism: Exploring Photo Collections in 3D. In *SIGGRAPH*, 2006.

[32] C. Strecha, A. M. Bronstein, M. M. Bronstein, and P. Fua. LDAHash: Improved matching with smaller descriptors. In *Technical Report*, 2010.

[33] C. Strecha, T. Pylvanainen, and P. Fua. Dynamic and Scalable Large Scale Image Reconstruction. In *CVPR*, 2010.

[34] G. Tolias and Y. Avrithis. Speeded-up relaxed spatial matching. In *ICCV*, 2011.

[35] A. Torii, J. Sivic, and T. Pajdla. Visual Localization by Linear Combination of Image Descriptors. In *ICCV'11 Mobile Vision Workshop*, 2011.

[36] K. Tuite, N. Tabing, D.-Y. Hsiao, N. Snavely, and Z. Popovic. PhotoCity: training experts at large-scale image acquisition through a competitive game. In *CHI*, 2011.

[37] A. Wendel, A. Irschara, and H. Bischof. Natural landmark-based monocular localization for mavs. In *ICRA*, 2011.

[38] C. Wu. SiftGPU: A GPU implementation of scale invariant feature transform (SIFT). http://cs.unc.edu/~ccwu/siftgpu, 2007.

[39] C. Wu, S. Agarwal, B. Curless, and S.M. Seitz. Multicore bundle adjustment. In *CVPR*, 2011.

[40] A. R. Zamir and M. Shah. Accurate Image Localization Based on Google Maps Street View. In *ECCV*, 2010.

[41] W. Zhang and J. Kosecka. Image based localization in urban environments. In *3DPVT*, 2006.