Hyperpoints and Fine Vocabularies for Large-Scale Location Recognition

Torsten Sattler[1], Michal Havlena[2], Filip Radenović[3], Konrad Schindler[2], Marc Pollefeys[1]
[1]Department of Computer Science, ETH Zürich, Switzerland
[2]Institute of Geodesy and Photogrammetry, ETH Zürich, Switzerland
[3]CMP, Faculty of Electrical Engineering, Czech Technical University in Prague
{sattlert,pomarc}@inf.ethz.ch,{havlena,schindler}@geod.baug.ethz.ch,radenfil@cmp.felk.cvut.cz

Abstract

Structure-based localization is the task of finding the absolute pose of a given query image w.r.t. a pre-computed 3D model. While this is almost trivial at small scale, special care must be taken as the size of the 3D model grows, because straight-forward descriptor matching becomes ineffective due to the large memory footprint of the model, as well as the strictness of the ratio test in 3D. Recently, several authors have tried to overcome these problems, either by a smart compression of the 3D model or by clever sampling strategies for geometric verification. Here we explore an orthogonal strategy, which uses all the 3D points and standard sampling, but performs feature matching implicitly, by quantization into a fine vocabulary. We show that although this matching is ambiguous and gives rise to 3D hyperpoints *when matching each 2D query feature in isolation, a simple voting strategy, which enforces the fact that the selected 3D points shall be co-visible, can reliably find a locally unique 2D-3D point assignment. Experiments on two large-scale datasets demonstrate that our method achieves state-of-the-art performance, while the memory footprint is greatly reduced, since only visual word labels but no 3D point descriptors need to be stored.*

1. Introduction

The continuously increasing amount of available imagery enables Structure-from-Motion (SfM) systems to produce larger and larger 3D scene models. In turn, these reconstructions can be used in visual navigation tasks to enable humans or autonomous vehicles to stay localized in their surrounding, by estimating camera poses w.r.t. to the model. Naturally, scalable localization becomes an issue as the size of the reconstructions increases.

Many state-of-the-art approaches for structure-based localization associate image descriptors [5, 21] with the scene points during SfM reconstruction, and use these to establish the 2D-3D matches required for pose estimation [9, 15, 20, 29, 34]. The most obvious issue w.r.t. scalability is memory consumption: large models easily contain hundreds of millions of descriptors, which require tens of gigabytes of storage space (actually often more than the underlying image database). Model compression schemes try to mitigate this problem by reducing the 3D model to a small fraction of all available scene points [6, 19, 26]. These schemes achieve impressive compression rates, but the smaller number of points also translates into fewer matches and thus a larger number of images that cannot be localized.

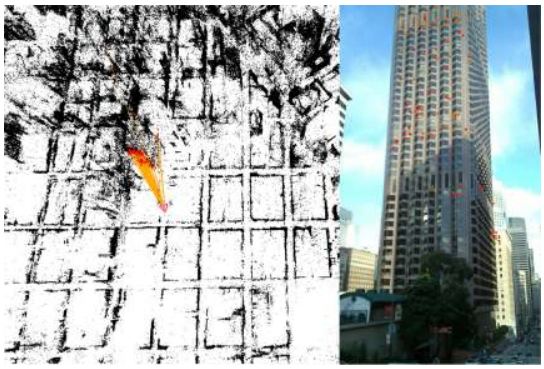


Figure 1: Correct camera pose estimation despite the presence of a repetitive pattern, as achieved by the proposed method for one of the query images of San Francisco.

A second, more subtle problem with growing model size is that the discriminative power of the descriptors decreases. As the descriptor space becomes more densely populated, matching features by nearest-neighbor search in that space becomes more ambiguous; and the widely used *ratio test* [21], which defines a match as a point whose distance to the nearest neighbor in descriptor space is a lot lower than to the 2nd-nearest neighbor, becomes too strict. One can use location recognition [3, 8, 31, 35] and image retrieval techniques [10, 27, 32] to first narrow down the search to smaller sub-regions of the model [7, 15, 30] with fewer scene points to match to. Empirically, though, pure descriptor matching mostly outperforms retrieval-based approaches on large datasets, since the latter are prone to false positives (wrong localizations).

In this paper, we propose to perform *model compression by quantizing the point descriptors*, instead of removing 3D scene points or using "mean descriptors" per 3D point. Using a fine visual vocabulary [13, 22, 33] enables us to represent each descriptor just by its word ID. Thanks to the size of the vocabulary (16M visual words), this does not introduce too much quantization noise. The vocabulary implicitly defines reasonable neighborhood sizes in the descriptor space, without the need to explicitly search for the 2$^{\text{nd}}$-nearest neighbor. Note that each 3D point can be, and typically will be, assigned to more than one word ID, because of the variability of its descriptors from different images. This makes the approach very different from using the mean descriptors per 3D point. At the same time, each ID will be assigned to many 3D points in the model. That means that when the 2D features from the query image are quantized with the same vocabulary, there are multiple potentially matching 3D points for each of them. The trick in our method is to not force each individual 2D feature to find a unique match, but instead to form *hyperpoints*, *i.e.*, sets of potential 3D correspondences for each 2D feature, which are *locally unique*. These hyperpoints can be disambiguated a lot more reliably than individual matches: by aggregating information from all hyperpoints through a simple voting procedure (as in image retrieval) the model is pruned to a set of promising locations. At each location, the hyperpoints turn into ordinary one-to-one matches, because of their local uniqueness. Having established the matches, one can proceed to pose estimation (geometric verification) with standard RANSAC [12]. We will show that this "delayed disambiguation" of matches compares favorably both against direct matching of point descriptors and against location recognition. The hyperpoint method achieves a higher number of correct poses (recall) with the same precision, respectively a lower number of false localizations with the same recall.

The contributions of the present paper are: *(i)* we demonstrate that using an inverted file with 16M visual words (IDs) is sufficient to achieve a localization rate on par with state-of-the-art approaches that must manage hundreds of millions of descriptors [20]; *(ii)* we show that enforcing local uniqueness of the matches returned by the vocabulary significantly boosts the localization performance; *(iii)* we demonstrate that the benefit of having a 3D model (as opposed to pure image retrieval) lies not only in the stricter geometric verification; rather, it also boosts the number of votes in the retrieval stage; *(iv)* we analyze common failure cases that lead to wrong pose estimates (false positives), and show that some of these can be avoided by re-ranking based on a simple measure of the inlier distribution in the image.

2. Related Work

When a SfM model of the scene is available, query image descriptors can be directly matched with 3D scene points, by storing for each 3D point the descriptors of the 2D points it originates from. In order to accelerate the matching, prioritized search strategies try to only consider a small subset of features that is likely to yield matches at low cost [28], or a set of scene points that is likely to be visible in the query image [9, 19]. For larger models, some of the matches lost due to the strictness of the ratio test can be recovered by back-matching selected 3D points against the image, exploiting the co-visibility of scene points [9, 20, 29]. State-of-the-art methods relax the threshold of the ratio test, and prefer to handle a larger proportion of false correspondences through a more involved geometric verification [20, 34]. In contrast, using a large vocabulary to find a set of locally unique correspondences makes our approach largely independent of the density in descriptor space.

In order to accelerate 3D-to-2D matching, [19] select a subset of all 3D points in such a way that every camera used in the reconstruction observes enough selected points, which is formulated as a set cover problem. The same problem is modeled as a mixed-integer quadratic program in [26], allowing them to incorporate additional constraints (at the cost of higher run-time). [6] demonstrate that by incorporating information about the density in descriptor space one can achieve a high rate of correct localizations with fewer points. Still, the method involves a trade-off between memory consumption and localization rate, and can localize fewer images than, *e.g.*, [20]. Instead of using descriptors, [11] learn classifiers to correlate query features and 3D points, simultaneously reducing the memory requirements and providing a run-time independent of the model size. However, their method requires a rough prior on the position, *e.g.*, from GPS, to handle large datasets.

Location recognition methods represent a scene as a set of geo-tagged images, and then use bag-of-words type image retrieval [27, 32] to identify database images similar to the query. The basic method has been refined to include vocabulary trees [25], query expansion [10], and burstiness suppression [17]. [31] select discriminative features that only occur at a few locations to improve the retrieval performance, and conversely [18] remove frequent visual words that are shared between unrelated locations. [7] cluster related database images and learn classifiers on top of the bag-of-words model to better distinguish between different places in the scene, effectively also down-weighting words that occur often. In contrast, [35] actively exploit repetitive structures to improve location recognition in urban scenes where they typically appear.

In order to better handle viewpoint changes, [15] generate synthetic views to improve the retrieval performance. [30] use Hamming embedding [16] to simulate a finer vocabulary and suppress incorrect votes. [3] show that down-weighting uninformative parts of the Hamming space based on density estimation boosts the recall of location retrieval.

[4, 8] use vanishing points to normalize features against changes in viewpoint before attempting retrieval, and also confirm that obvious position priors based on GPS improve location recognition in large scenes.

While the methods mentioned so far rely on visual vocabularies, [36, 37] work with the original feature descriptors. [36] use the nearest neighbors found for each query feature to vote for their associated GPS positions. [37] is related to our approach in that they also determine a consistent set of matches for all query features from a larger set of potential matches with multiple images. Their formulation leads to an NP-hard generalized minimum clique problem. In contrast, we solve the disambiguation problem using image retrieval by exploiting the local uniqueness of hyperpoints.

Recently, [33] have shown that exhaustive geometric verification is feasible even for databases containing billions of images, if one uses a fine vocabulary. A similar idea was used in the context of SfM computation in [13], where descriptor matching is replaced by finding collisions in a very fine vocabulary [22]. These examples show that quantization w.r.t. a fine vocabulary, as opposed to the coarse vocabularies commonly used for retrieval, can to some extent replace a matching routine. In this paper, we explore the potential of a fine vocabulary for the localization problem.

3. Locally Unique Descriptor Matching

Rather than removing points from a reconstruction, and thus losing useful information, this paper presents a compression scheme based on quantizing descriptors. Fig. 2 illustrates our pipeline. We use a fine visual vocabulary, consisting of 16M words, to find potentially matching 3D points for each feature in a query image. The advantage of our approach is that it easily scales to large datasets since adding a point and its descriptors requires only a few additional bytes. Naturally, establishing matches based on quantization leads to ambiguous matches where a feature is linked with two or more points that can be observed together. As the main contribution of the paper, we therefore propose a method that selects a subset of *locally unique matches* and shows that enforcing local consistency improves the localization rate. Having obtained sets of locally unique matches, we can efficiently find the parts of the model most likely to be visible in the query image by solving an image retrieval problem. In turn, each retrieved image defines a set of unique matches, enabling the use of simple RANSAC-based camera pose estimation.

The use of locally unique matches is motivated by the way SfM methods construct 3D models, where feature tracks are obtained by matching local features between database images, followed by rejecting ambiguous matches using the ratio test. Thus, only locally unique matches are triangulated to 3D points and we can expect that enforcing local uniqueness will not reject too many correct matches.

In the following, we first define the property of local uniqueness for a set of 2D-3D matches. As a result of this new matching strategy, a query feature might be linked with multiple 3D points. Inspired by the definition of *hypergraphs*, where an edge can contain multiple vertices, we refer to such a collection of points as a *hyperpoint*. After defining local uniqueness, we describe an approach for computing hyperpoints based on a fine visual vocabulary [22] (*c.f.* Sec. 3.2). Section 4 then details how image retrieval is used together with hyperpoints for structure-based localization, where Sections 4.3 and 4.4 explain how to exploit co-visibility information from the SfM process to improve the retrieval performance. Finally, Section 4.5 discusses problems in camera pose estimation specific to matching with vocabularies.

3.1. Hyperpoints

The 3D points and cameras[1] in a SfM reconstruction define the bipartite *visibility graph* $\mathcal{G} = (\mathcal{P} \cup \mathcal{C}, E)$ [19] (*c.f.* Fig. 2). Each node $p \in \mathcal{P}$ corresponds to a 3D point in the model while each node $c \in \mathcal{C}$ corresponds to a camera. The graph contains an edge $\{p, c\}$ if the camera corresponding to c observes the 3D point p. Typically, the visibility graph is used to define the co-visibility relation between pairs of points [19, 20, 29]. Let

$$\mathcal{C}(p) = \{c \in \mathcal{C} \mid \{p, c\} \in E\} \tag{1}$$

be the set of cameras observing a point p. Two points p_1, p_2 are considered co-visible if $\mathcal{C}(p_1) \cap \mathcal{C}(p_2) \neq \emptyset$.

Let $r_s(f, p)$ be any function rating the similarity between the descriptor of a query feature f and the descriptors of a 3D point p. Given some threshold τ on the similarity function, we can obtain a set of potentially matching points. For example, the red feature in the query image in Fig. 2 potentially matches to two 3D points that are not co-visible. In that case, consistency with other feature matches can be exploited to disambiguate the correspondence. In contrast, two of the points matching the blue feature are co-visible and thus cannot be disambiguated by other features. We seek to find a set of *locally unique matches* $H(f) = \{p \mid r_s(f, p) > \tau\}$, in the following also referred to as a *hyperpoint*, containing no co-visible points by detecting and removing local ambiguities.

Given that most local descriptors are not viewpoint invariant, we experimented with using only a subset of all cameras from $\mathcal{C}(p)$ to define co-visibility. Yet, initial experiments showed that using all cameras gives the best results.

3.2. Hyperpoints and Fine Vocabularies

Based on the definition of local uniqueness given above, we aim to compute a hyperpoint $H(f)$ for each feature f in

[1] To avoid ambiguities, we refer to the images in the reconstruction as *cameras* throughout the paper.

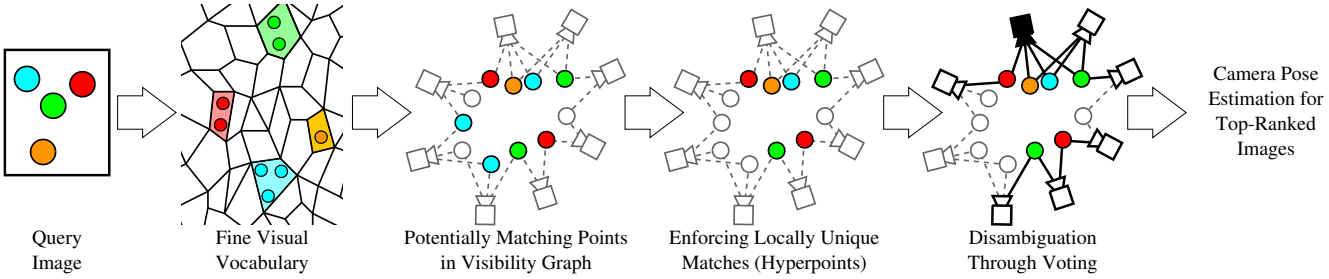


Figure 2: The proposed localization pipeline. Using a fine visual vocabulary, we obtain a set of potential matching points for each feature in the query image. The visibility graph, encapsulating the visibility relation between 3D points and cameras in a reconstruction, is then used to obtain a *hyperpoint*, *i.e.*, a set of locally unique matching points, for each feature (*c.f.* Sec. 3). In order to find sets of probable matches, each point contained in a hyperpoint votes for the cameras in the reconstruction from which it was triangulated (*c.f.* Sec. 4). For each such set, the camera pose for the query image is estimated using RANSAC, starting with the camera that receives most votes (black).

the query image. We solve this problem by first computing a set of potentially matching points $\mathcal{P}(f) = \{p \mid r_s(f,p) > \tau\}$ and then enforcing local uniqueness.

A vocabulary-based similarity function. Ideally, we would like to include a point into a hyperpoint only if its descriptor is "very similar" to the feature's descriptor while rejecting as many points as possible. Yet, finding a good similarity function r_s is hard. Many descriptor spaces, *e.g.*, SIFT [21] and SURF [5], consist of regions of varying density, which needs to be considered when designing r_s. K-means quantization, *i.e.*, training a visual vocabulary, automatically adapts to this fact by creating smaller Voronoi cells in regions with higher density. While it has been shown that vocabularies containing hundreds of thousands words are too coarse to create discriminative matches [30], fine vocabularies containing tens of millions of descriptors have recently enjoyed some success in image retrieval [22, 23, 33] and feature matching [13]. In this paper, we thus use a fine vocabulary $\mathcal{V}$ to compute the set $\mathcal{P}(f)$ of potentially matching points.

Let $\mathcal{V}(p) \subset \mathcal{V}$ be the set of visual words to which the descriptors associated with p are quantized, and let $\omega_1, \ldots, \omega_k$ be the k words in $\mathcal{V}$ closest to the descriptor of a query feature f, where ω_i is the i^{th}-nearest neighboring word of f. The set of potentially matching points for f then consists of all points mapped to one of the k closest words of f, *i.e.*,

$$\mathcal{P}(f) \quad = \quad \{p \mid \mathcal{V}(p) \cap \{\omega_1, ..., \omega_k\} \neq \emptyset\} \ . \qquad (2)$$

After determining $\mathcal{P}(f)$ through look-ups in an inverted file structure, we next select a subset of all points from $\mathcal{P}(f)$ to form the hyperpoint $H(f)$. Intuitively, we want to add a point $p \in \mathcal{P}(f)$ to $H(f)$ if there is no other point $p' \in \mathcal{P}(f)$ co-visible with p that has a descriptor more similar to f. Since we use visual words instead of the full descriptors, we measure descriptor similarity by the *word rank* $\text{rank}(p, f)$. The point p has word rank i, *i.e.*, $\text{rank}(p, f) = i$, if one of the descriptors of p is assigned to ω_i, while at the same time none of p's descriptors is assigned to a closer word ω_j with $j < i$. A point $p \in \mathcal{P}(f)$ is then added to $H(f)$ if it has a smaller word rank than any of its co-visible points, *i.e.*,

$$H(f) = \{p \in \mathcal{P}(f) \mid \forall p' \in \mathcal{P}(f) \setminus \{p\} : \mathcal{C}(p) \cap \mathcal{C}(p') = \emptyset \vee \text{rank}(p, f) < \text{rank}(p', f)\} \ . \qquad (3)$$

In case two co-visible points have the same word rank, none of them is included in $H(f)$. As a result, all locally ambiguous matches, *e.g.*, arising from locally repeating structures, are removed. Thus, we obtain a unique hyperpoint per feature that does not contain any pair of co-visible points.

In practice, at most one feature in the query image can correspond to the projection of a given 3D point. We thus actively enforce that each point is contained in at most one hyperpoint. Before computing the set $\mathcal{P}(f)$ for each feature f, we check for every point whether its visual words are contained in the closest words of multiple features. A point p is only added to $\mathcal{P}(f)$ if f is its nearest neighbor in the image, *i.e.*, $\text{rank}(p, f) < \text{rank}(p, f')$ holds for every other feature $f' \neq f$. Consequently, all hyperpoints are disjoint, *i.e.*, no 3D point is contained in more than one hyperpoint.

4. Localization Using Hyperpoints

Having obtained a hyperpoint $H(f)$ for each query feature f, we next seek to find a set of consistent matches to be used for camera pose estimation inside a RANSAC loop [12]. A set of consistent matches contains at most one point from each hyperpoint such that all selected points are co-visible, *i.e.*, come from the same part of the scene. At the same time, no point should be contained in more than one correspondence. Given a set of 1-to-many matches for each feature in a query image, [37] consider the similar problem of selecting one match per feature. They model this problem as a variant of the NP-hard generalized minimum clique problem, which they solved approximately. In the following, we show that the local uniqueness property of the hyperpoints and the visibility graph allow us to express the match selection problem as a (computationally) much simpler image retrieval problem.

4.1. Hyperpoints And The Visibility Graph

Consider a camera $c \in \mathcal{C}$ from the visibility graph and the set of points $\mathcal{P}(c) = \{p \in \mathcal{P} \mid \exists\{p, c\} \in E\}$ visible in that camera. By the definition of hyperpoints, each feature is contained only once in the set of matches

$$M(c) = \{(f, p) \mid p \in H(f) \cap \mathcal{P}(c)\} \tag{4}$$

obtained by intersecting $\mathcal{P}(c)$ with the points contained in the hyperpoints. This is due to local uniqueness enforced when constructing a hyperpoint. If there were correspondences (f, p), $(f, p') \in M(c)$, then $H(f)$ would contain two co-visible points. Thus, selecting a camera in the viewing graph automatically defines a set of consistent matches since we enforce that each point is contained in at most one hyperpoint. As noted above, we observe that wrong matches are often spread over the full model while correct matches form a dense cluster in the visibility graph. Under the assumption of perfect co-visibility information, the consistent set of matches containing the correct correspondences can be found by determining the camera $c \in \mathcal{C}$ associated with the largest set $M(c)$. This camera can be efficiently computed by having each point p contained in hyperpoint $H(f)$ vote for all images from $\mathcal{C}(p)$, *i.e.*, by solving an image retrieval problem.

4.2. Solving The Retrieval Problem

In practice, cameras observing many points are more likely to obtain more votes. Thus, it is necessary to weight the raw number of consistent matches for c based on $\mathcal{P}(c)$. Instead of simply counting the number of matches per camera, the goal of using a weighting scheme is thus to identify a region which contains significantly more matches than can be explained by the background noise caused by incorrect correspondences. One possible weighting scheme is the well-known *inverse document frequency* weight [32], which gives visual words contained in many cameras less influence. Another possibility is to employ the *inter-image burstiness*[2] scheme from [17] that weights each vote by $1/\sqrt{|\mathcal{P}(c)|}$. Notice that intra-image burstiness, *i.e.*, the same word co-occurring multiple times in a camera, is automatically handled by our definition of hyperpoints. In addition to a weighting scheme, we also employ one of the weak geometric consistency filters from [16]. For a camera c, the feature orientations of the query feature f and its matching point $p \in \mathcal{P}$ define a constraint on the rotation along the principal axis between the query camera and c. We quantize the space of relative orientations voting into 12 bins, each covering 30°. Each match (f, p) adds its weight to all bins whose center differs by at most 30° from the relative orientation between f and p. When performing spatial verification (*c.f.* Sec. 4.5), only the features associated with the bin with the largest weight are used. Naturally, we use an inverted file system for efficient voting.

[2]Burstiness is the phenomenon that some combination of words occur much more often together than predicted by assuming an independent uniform distribution.

4.3. Recovering Votes Lost To Quantization

In a traditional image retrieval system, each entry in an inverted file corresponds to a feature in a database image. Using a visual vocabulary, *i.e.*, a quantization of the descriptor space, introduces artifacts, where a query feature and its corresponding image feature are mapped to different words. Consequently, correct votes are lost due to quantization, even if $k > 1$ words are used for each query feature. This is especially true for the large vocabularies that we are using in this paper [22]. As a result, the point actually corresponding to a feature f might be (partially) missing from $H(f)$. [22] therefore propose to learn a probabilistic similarity measure that related visual words. Unfortunately, learning this measure requires a massive amount of data.

Fortunately, some of the lost votes can easily be recovered by exploiting the visibility graph. If a point p is contained in a hyperpoint $H(f)$, the visibility graph defines all cameras observing p, even if p's descriptor extracted from the image belonging to a camera c is mapped to a different word. For this reason, our approach casts votes for all cameras from $\mathcal{C}(p)$. Our experimental results will show that exploiting these associations increases the retrieval performance compared to the classic approach of treating each inverted file entry as an independent observation.

4.4. Match Expansion

The cameras contained in the SfM reconstruction represent a subset of all possible views on the scene. Thus, it is likely that there exists no camera that observes all correct matches between query features and model points. Only considering points visible in a camera c for pose estimation will thus ignore correct correspondences, which in turn can severely degrade the quality of the pose estimate or prevent successful pose estimation altogether if too many matches are lost. Again, the visibility graph enables us to recover additional matches for a camera c before pose estimation by including correspondences found for similar cameras.

For a given camera c observing points $\mathcal{P}(c)$, let

$$\mathcal{C}(c) = \bigcup_{p \in \mathcal{P}(c)} \mathcal{C}(p) \tag{5}$$

be the set of other cameras that observe at least one point from $\mathcal{P}(c)$. In order to obtain more matches for camera pose estimation, we consider the set of 3D points

$$\mathcal{P}_{\text{ext}}(c) = \{p \mid \exists c' \in \mathcal{C}(c) : p \in \mathcal{P}(c')\} \tag{6}$$

visible in any of these cameras. We consequently enlarge the set of initial matches $M(c)$ found inside the hyperpoints (*c.f.* Eq. 4) to

$$M_{\text{ext}}(c) = M(c) \cup \{(f, p) \mid p \in H(f) \cap \mathcal{P}_{\text{ext}}(c)\} \ , \tag{7}$$

Figure 3: A sample query image with a pose that has more than 30 inliers even though it is obviously wrong. Inliers are only found in a small region of the image, forcing pose estimation to place the camera far away from the scene. Note that structure-based localization approaches typically use a threshold of 12 inliers to detect bad poses.

i.e., we use all matches found for the cameras from $\mathcal{C}(c)$. Obviously, we could extend the set of matches even further by expanding $\mathcal{C}(c)$. However, an experimental analysis showed that from a set of correctly matching points, only few are not contained in the extended set.

The match expansion step might violate the local uniqueness property. However, we found that only few features actually receive multi-matches. In addition, the *effective inlier rate* [15] that we use for re-ranking after performing pose estimation for the top-N ranked cameras automatically handles these multi-matches (*c.f.* Sec. 4.5). Thus, we do not adapt the definition of hyperpoints.

Notice that we do not use the additional matches during voting. The rationale behind this decision is to avoid casting too many votes as these lead to a higher background noise during voting, making it harder to distinguish between unrelated cameras with many associated matches and relevant cameras with only few votes.

Obviously, our match expansion is strongly related to query expansion [10]. Query expansion augments the query with geometrically verified features from already retrieved images. In contrast, match expansion is performed *before* spatial verification.

4.5. Ranking Pose Estimates

Structure-based localization methods that employ image retrieval typically rely on the original feature descriptors to establish the 2D-3D matches required for camera pose estimation [7, 15, 30]. For each camera c, they only match against the points visible in it. Thus, they can safely apply the ratio test, enabling them to reject most wrong correspondences before RANSAC-based camera pose estimation. As a result, poses estimated from incorrect matches usually have few inliers. Re-ranking the poses computed for the top-N ranked cameras based on their number of inliers therefore usually lists the best pose first. Unfortunately, we found out that poses computed with our method do not necessarily follow this behavior.

Even when using a fine vocabulary, we noticed that we usually find quite many matches for unrelated cameras if the query image is highly textured, *e.g.*, when showing skyscrapers. Consequently, we were able to observe camera poses that look obviously wrong to a human observer but nonetheless received many inliers and were considered best after re-ranking.

In particular, we noticed two popular failure cases (*c.f.* Fig. 3). Often, the computed camera pose differs significantly from the viewing direction of the camera c which triggered pose estimation (to a degree that the query camera was placed behind the walls of a building) and/or most of the inliers were found in a small region of the image. In the following, we discuss two simple approaches to handle these error cases.

The first failure case, an estimated pose differing significantly from the pose of the camera c, can be handled by enforcing a geometric constraint during RANSAC. Let $\mathbf{p}$, $\mathbf{c} \in \mathbb{R}^3$ be the positions of a point p and the camera center of a camera c, respectively. Consider the current pose estimate of RANSAC with camera center $\mathbf{c}_{\text{est}}$ and a point p that satisfies the threshold on the reprojection error. p should only be counted as an inlier if the estimated pose observes p under a similar viewing direction as c, *i.e.*, if

$$\frac{(\mathbf{c}_{\text{est}} - \mathbf{p})^T (\mathbf{c} - \mathbf{p})^T}{\|\mathbf{c}_{\text{est}} - \mathbf{p}\|_2 \|\mathbf{c} - \mathbf{p}\|_2} \leq \cos\alpha \ , \tag{8}$$

where α is some threshold on the angle.

The second failure case is caused by finding many inliers in a small region of the image. In this case, we observed that RANSAC preferred to find poses far away from the scene, even if the internal calibration was known. Instead of rating a pose based on its number of inliers, we use a slightly adapted version of the *effective inlier count* from [14, 15], which takes the distribution of inliers in the query image into account. Each matching feature covers an square of side length $2r$, where r is the threshold on the reprojection error that is used to distinguish inliers and outliers. Let A_{inliers} be the total area covered by all inlier features. The effective inlier count is then defined as

$$I_{\text{effective}} = I \cdot A_{\text{inliers}} / (I \cdot 4r^2) = A_{\text{inliers}} / (4r^2) \ , \tag{9}$$

where I is the number of inliers found by RANSAC. The measure thus expresses how well the distribution of inliers in the query image resembles a "perfect" distribution where all inliers are at least $2r$ pixels apart. Notice that a feature matching to multiple points will contribute at most once to the effective inlier count.

Instead of using a constant radius, [14, 15] use a radius that is proportional to the number of feature matches and normalize by the area covered by all matching features. We use our formulation because it is simpler to compute and we did not notice much difference between the two formulations. Similarly, we tried normalizing by the area covered

by all features found in the query image to obtain a score that is comparable between images. We found little difference in re-ranking compared to the effective inlier count.

While pretty simple, we show in Section 5 that the proposed re-ranking scheme significantly improves the ranking quality compared to using the number of inliers. Notice that this measure only depends on the feature geometry in the image and is thus also applicable for standard retrieval pipelines where spatial verification is performed using 2D-2D matches. As a nice side effect, multi-matches found for a feature will contribute at most once to the measure.

Finally, we want to point out one advantage of our approach compared to structure-based methods using image retrieval. Since the latter methods perform feature matching using the original descriptors, they can only consider a small number of top-ranked cameras (usually set to 10) in a reasonable time. In contrast, our method uses the same matches for voting and pose estimation, making RANSAC-based pose estimation the bottleneck in our pipeline. Still, we found that we can easily consider the top-100 ranked images in a few seconds per image.

5. Experiments

The proposed method is validated on the San Francisco dataset [8], namely model SF-0 [20] using 610k out of the 1.06M available images and having 30M 3D points, and the Landmarks dataset [20] depicting 1,000 famous landmarks in 205k images and 38M 3D points.

5.1. San Francisco

First, a fine vocabulary specific for the dataset is created using SF-0's SIFT descriptors corresponding to the projections of the triangulated scene 3D points. We use an approximate hierarchical k-means approach with an average branching factor of 4,096 to efficiently create a vocabulary of 16M visual words [22]. This two-level setup decreases the average assignment time by better balancing the tree. For assigning efficiency, an approximate nearest neighbor search algorithm, FLANN [24], is used. It has been shown in [22] that, given a sufficiently large training set, using a higher number of visual words will boost retrieval performance while reducing query time. We think that a vocabulary containing 16M visual words is a good compromise between the time needed to build a vocabulary on one side and the performance with average query time on the other side. The described vocabulary was recently proven to work in efficient image detail retrieval as well [23].

Having the vocabulary, the descriptors of both the database and query images are quantized w.r.t. it. The need to quantize query image descriptors brings just a small overhead in means of runtime, which is not drastically different from the time needed to extract the features themselves. To deal with quantization noise, we assign each query descriptor to up to nine nearest neighboring words that can be located in the five best branches of the first vocabulary level. In contrast, only the nearest word of each 3D point descriptor is used as to reduce the size of the inverted files and minimize the query time.

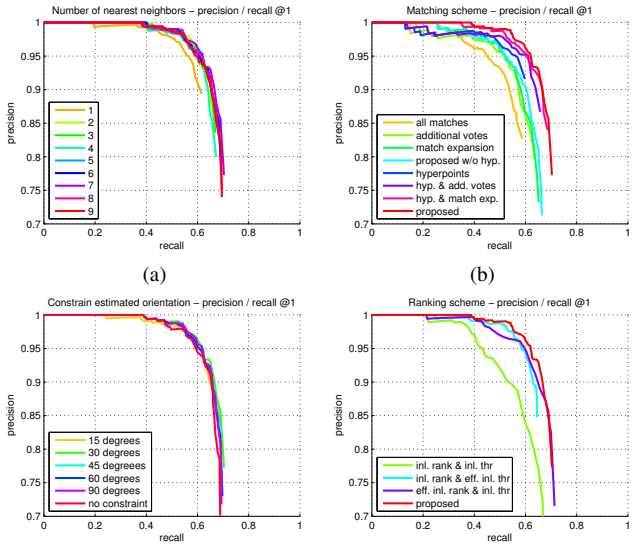


Figure 4: Ablation study for our method on SF-0 dataset. The proposed approach uses 7 nearest neighbors, employs all the proposed enhancements, retrieves top-100 images, constrains the camera orientation by 30°, and uses effective inlier rate to re-rank and select the best pose (if any). Note that the achieved recall for 95% precision is 61.9%.

The results are reported for the improved GT from 2014 [3]. Following the protocol from [20], a query image is considered being correctly localized if at least one of the inliers lies on the correct building. Namely, we are interested in `precision/recall @1` measure, which is the most relevant for the localization task: the user wants to know her location in the model with high reliability – the performance is usually measured by the recall at 95% precision value [8]. The measure typically used by image retrieval methods, `recall@N`, does not apply well in here because user interaction would be needed to select from multiple retrieved poses.

In Fig. 4, we present the results of the ablation study. The curves are obtained by varying the (effective) inlier threshold that decides when an image is considered localized. Thus, the curves can have a positive slope as increasing recall by lowering the threshold does not necessarily increase the number of false positive localizations. Fig. 4b shows results for the different matching schemes when using the 7 nearest words for each query feature, the proposed re-ranking scheme using the effective inlier counts, and an angle threshold of $\alpha = 30°$ (*c.f.* Eqn. 8). Compared to casting additional votes and performing match expansion, hyperpoints provide the largest improvement over the base-

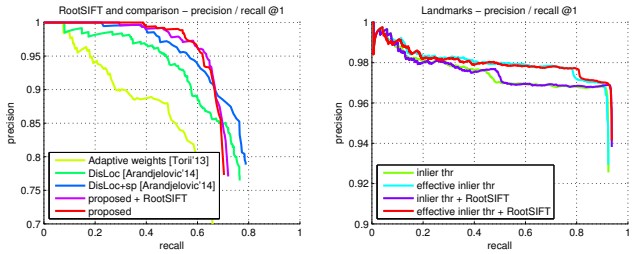
line of using all matches. Thus, using hyperpoints is the most important part of our approach. Fig. 4a shows that changing the number of nearest neighboring words only has a minor impact on recall for SF-0 when using at least two words. The constraint on camera orientation has a relatively small impact on the performance of the method, but yet can yield additional 2% of recall (*c.f*. Fig. 4c). Fig. 4d shows that re-ranking based on raw inlier counts performs inferior to using effective inlier counts.

The noticeable drop in Fig. 4a when using just the nearest word suggests that a significant amount of matching points are quantized to a different word than their corresponding image features. Due to a lack of ground truth, exactly estimating how many matches are lost due to quantization is impossible. For a same-sized vocabulary, [13] report that about 50% of all potential matches are lost when using only the nearest word. We found that for about 1/3 of all points, more than 50% of their descriptors quantize to the same word. The largest inverted file record contains 75 descriptors, while the mean size of non-empty records is 9.

Next, we show how our performance relates to competing approaches. As most of them rely on RootSIFT [2], we trained a RootSIFT-based vocabulary for SF-0, too, which in turn brought us on average an additional 1.5% of recall, see Fig. 5a. For the absolute recall value, DisLoc [3] works the best, especially in the variant with spatial verification (+sp), but the proposed method (namely its RootSIFT variant) is better for precision higher than 90%. At a precision of 95%, [3] achieve a 56.6% recall. Our approach with RootSIFT obtains 63.5%, *i.e*., in the high-precision regime we outperform [3] by 12% relatively. To compare with [20], the original GT from 2011 was used. For 95% precision, [20] achieve a recall of 54.2% using all point descriptors (memory footprint 19.0 GB, see the supplementary material [1] for details), respectively 50.2% when using a single mean descriptor per point (4.9 GB). In contrast we achieve a 59.1% recall (also using 4.9 GB), corresponding to 9%, respectively 17.7% relative improvement. This result is interesting, as it shows that matching based on a visual vocabulary can actually outperform matching with full descriptors.

5.2. Landmarks

In order to test our approach on the Landmarks dataset, we trained two new specific fine vocabularies, one for SIFT and another for RootSIFT. Note that this is due to the fact that SF-0 and Landmarks use different SIFT extractors whose outputs are not fully compatible. In the evaluation, a landmark is considered being correctly recognized if at least 12 inliers are found and the majority of them belongs to the GT landmark. For a fair comparison with [7], we retrieve only the top-10 images. We also only use the nearest word for each feature. Using more neighbors reduces the overall recall as the large number of features found in the



(a) San Francisco - RootSIFT (b) Landmarks

Figure 5: (a) Comparison of the proposed method with competing approaches on San Francisco. Note that, opposed to these methods, we do not use all the available database images but only a subset (SF-0) which makes localization of some of the query images impossible. (b) Results for Landmarks dataset using only the nearest word and top-10 retrieved images.

query images leads to many votes for unrelated cameras. Using a single-threaded C++ implementation on a standard Intel i7-based PC, we processed the 10,000 query images in about 3.5 hours (*c.f*. Fig. 5b). For the 12 inlier threshold, we correctly localized 89.1% of images with precision 96.8% for SIFT and 91.1% of images with precision 96.9% for RootSIFT. This clearly outperforms even the `recall@10` value 81.2% from [7] showing the benefit of having a fine vocabulary, opposed to a coarse one. Also the registration performance reported for the compressed models [6] is outperformed. Yet, our method does not reach the almost perfect performance of the method using all the descriptors [20].

6. Conclusions

We have presented a localization pipeline able to deliver reliable camera poses w.r.t. large-scale 3D scene models in favorable run-time thanks to the novel concept of hyperpoints. Hyperpoints allow to separate the difficult problem of finding a unique 2D-3D matching into two simpler ones: *(i)* establishing locally unique 2D-3D matches using a fine visual vocabulary and the visibility graph of the model and *(ii)* disambiguating these matches globally by using image retrieval techniques and the visibility information from the model again. The proposed method proved to have state-of-the-art localization performance for the SF-0 dataset and a competitive performance on Landmarks considering the methods performing model compression.

Future work comprises of testing the concept of hyperpoints in combination with more advanced image retrieval approaches which are able to achieve a higher overall recall than our simple voting scheme. We would also like to test the performance of the proposed method in conjuncture with the projection to lower dimensional spaces.

Acknowledgements. This work was supported by a MSMT LL1303 ERC-CZ grant.

References

[1] Supplementary material. http://www.cvg.ethz.ch/research/large-scale-localization/. 8

[2] R. Arandjelović and A. Zisserman. Three things everyone should know to improve object retrieval. In *CVPR*, 2012. 8

[3] R. Arandjelović and A. Zisserman. DisLocation: Scalable descriptor distinctiveness for location recognition . In *ACCV*, 2014. 1, 2, 7, 8

[4] G. Baatz, K. Köser, D. Chen, R. Grzeszczuk, and M. Pollefeys. Leveraging 3D City Models for Rotation Invariant Place-of-Interest Recognition. *IJCV*, 96(3):315–334, 2012. 3

[5] H. Bay, T. Tuytelaars, and L. V. Gool. SURF: Speeded-Up Robust Features. In *ECCV*, 2006. 1, 4

[6] S. Cao and N. Snavely. Minimal Scene Descriptions from Structure from Motion Models. In *CVPR*, 2014. 1, 2, 8

[7] S. Cao and N. Snavely. Graph-Based Discriminative Learning for Location Recognition. *IJCV*, 112(2):239–254, 2015. 1, 2, 6, 8

[8] D. Chen, G. Baatz, K. Köser, S. Tsai, R. Vedantham, T. Pylvänäinen, K. Roimela, X. Chen, J. Bach, M. Pollefeys, B. Girod, and R. Grzeszczuk. City-scale landmark identification on mobile devices. In *CVPR*, 2011. 1, 3, 7

[9] S. Choudhary and P. J. Narayanan. Visibility Probability Structure from SfM Datasets and Applications. In *ECCV*, 2012. 1, 2

[10] O. Chum, J. Philbin, J. Sivic, M. Isard, and A. Zisserman. Total recall: Automatic query expansion with a generative feature model for object retrieval. In *ICCV*, 2007. 1, 2, 6

[11] M. Donoser and D. Schmalstieg. Discriminative Feature-to-Point Matching in Image-Based Locallization. In *CVPR*, 2014. 2

[12] M. Fischler and R. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Comm. ACM*, 24(6):381–395, 1981. 2, 4

[13] M. Havlena and K. Schindler. VocMatch: Efficient Multi-view Correspondence for Structure from Motion. In *ECCV*, 2014. 2, 3, 4, 8

[14] A. Irschara. *Scalable Scene Reconstruction and Image based Localization*. PhD thesis, Graz University of Technology, 2012. 6

[15] A. Irschara, C. Zach, J.-M. Frahm, and H. Bischof. From Structure-from-Motion Point Clouds to Fast Location Recognition. In *CVPR*, 2009. 1, 2, 6

[16] H. Jegou, M. Douze, and C. Schmid. Hamming Embedding and Weak Geometric Consistency for Large Scale Image Search. In *ECCV*, 2008. 2, 5

[17] H. Jegou, M. Douze, and C. Schmid. On the burstiness of visual elements. In *CVPR*, 2009. 2, 5

[18] J. Knopp, J. Sivic, and T. Pajdla. Avoiding confusing features in place recognition. In *ECCV*, 2010. 2

[19] Y. Li, N. Snavely, and D. P. Huttenlocher. Location Recognition using Prioritized Feature Matching. In *ECCV*, 2010. 1, 2, 3

[20] Y. Li, N. Snavely, D. P. Huttenlocher, and P. Fua. Worldwide Pose Estimation Using 3D Point Clouds. In *ECCV*, 2012. 1, 2, 3, 7, 8

[21] D. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004. 1, 4

[22] A. Mikulík, M. Perdoch, O. Chum, and J. Matas. Learning vocabularies over a fine quantization. *IJCV*, 103(1):163–175, 2013. 2, 3, 4, 5, 7

[23] A. Mikulík, F. Radenović, O. Chum, and J. Matas. Efficient image detail mining. In *ACCV*, 2014. 4, 7

[24] M. Muja and D. G. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. In *VISAPP*, 2009. 7

[25] D. Nister and H. Stewenius. Scalable recognition with a vocabulary tree. In *CVPR*, 2006. 2

[26] H. S. Park, Y. Wang, E. Nurvitadhi, J. C. Hoe, Y. Sheikh, and M. Chen. 3D Point Cloud Reduction using Mixed-integer Quadratic Programming. In *CVPR Workshops*, 2013. 1, 2

[27] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Lost in quantization: Improving particular object retrieval in large scale image databases. In *CVPR*, 2008. 1, 2

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

[29] T. Sattler, B. Leibe, and L. Kobbelt. Improving Image-Based Localization by Active Correspondence Search. In *ECCV*, 2012. 1, 2, 3

[30] T. Sattler, T. Weyand, B. Leibe, and L. Kobbelt. Image Retrieval for Image-Based Localization Revisited. In *BMVC*, 2012. 1, 2, 4, 6

[31] G. Schindler, M. Brown, and R. Szeliski. City-scale location recognition. In *CVPR*, 2007. 1, 2

[32] J. Sivic and A. Zisserman. Video Google: A text retrieval approach to object matching in videos. In *ICCV*, 2003. 1, 2, 5

[33] H. Stewénius, S. H. Gunderson, and J. Pilet. Size Matters: Exhaustive Geometric Verification for Image Retrieval. In *ECCV*, 2012. 2, 3, 4

[34] L. Svarm, O. Enqvist, M. Oskarsson, and F. Kahl. Accurate Localization and Pose Estimation for Large 3D Models. In *CVPR*, 2014. 1, 2

[35] A. Torii, J. Sivic, T. Pajdla, and M. Okutomi. Visual Place Recognition with Repetitive Structures. In *CVPR*, 2013. 1, 2

[36] A. R. Zamir and M. Shah. Accurate Image Localization Based on Google Maps Street View. In *ECCV*, 2010. 3

[37] A. R. Zamir and M. Shah. Image Geo-localization Based on Multiple Nearest Neighbor Feature Matching using Generalized Graphs. *PAMI*, 36(8):1546–1558, 2014. 3, 4