A Vote-and-Verify Strategy for Fast Spatial Verification in Image Retrieval

Johannes L. Schönberger[1⋆], True Price[2⋆], Torsten Sattler[1⋆],
Jan-Michael Frahm[2], Marc Pollefeys[1,3]

[1] ETH Zürich, [2] UNC Chapel Hill, [3] Microsoft

Abstract. Spatial verification is a crucial part of every image retrieval system, as it accounts for the fact that geometric feature configurations are typically ignored by the Bag-of-Words representation. Since spatial verification quickly becomes the bottleneck of the retrieval process, runtime efficiency is extremely important. At the same time, spatial verification should be able to reliably distinguish between related and unrelated images. While methods based on RANSAC's hypothesize-and-verify framework achieve high accuracy, they are not particularly efficient. Conversely, verification approaches based on Hough voting are extremely efficient but not as accurate. In this paper, we develop a novel spatial verification approach that uses an efficient voting scheme to identify promising transformation hypotheses that are subsequently verified and refined. Through comprehensive experiments, we show that our method is able to achieve a verification accuracy similar to state-of-the-art hypothesize-and-verify approaches while providing faster runtimes than state-of-the-art voting-based methods.

1 Introduction

Image retrieval, *i.e.*, finding relevant database images for a given query picture, is a fundamental problem in computer vision with applications in object retrieval [1, 2], location recognition [3–5], image-based localization [6, 7], automatic photo annotation [8], view clustering [9, 10], loop-closure [11], and Structure-from-Motion [12–14]. Although methods that use compact image representations [15–17] have gained popularity, especially in combination with deep learning [18–20], state-of-the-art systems [2, 21–25] still follow the *Bag-of-Words* (BoW) paradigm [1] proposed over a decade ago. The BoW model represents each image as a set of *visual words* obtained by quantizing the local feature space. Visually similar database images can then be found by searching for photos with similar visual words, usually implemented efficiently using inverted files [1]. For the sake of computational efficiency, BoW models generally only consider the presence and absence of visual words in an image and largely ignore their spatial configuration. Thus, a subsequent *spatial verification* phase [26] is typically used to filter retrieved photos whose visual words are not spatially consistent with the

⋆ These authors contributed equally to the paper.

words in the query image. If not implemented efficiently, this spatial verification step quickly becomes the bottleneck of an image retrieval pipeline.

Spatial verification computes a geometric transformation, *e.g.*, a similarity or affine transformation [26], from feature correspondences between the query and database images. The correspondences are obtained from common visual word assignments of the query and database features. This often leads to many wrong correspondences, especially in the presence of repetitive structures or when using small vocabularies to reduce quantization artifacts. This makes traditional RANSAC-based approaches [27] that estimate transformations from multiple matches infeasible, as their runtime grows exponentially with the percentage of outliers. A key insight for fast spatial verification is to leverage the local feature geometry to hypothesize a geometric transformation from a single correspondence [26, 28]. This significantly reduces the number of hypotheses that need to be verified. Spatial verification can be accelerated by replacing the hypothesize-and-verify framework with a Hough voting scheme based on quantizing the space of transformations [28, 29]. These methods approximate the similarity between two images by the number of matches falling into the same bin in the voting space. Quantization artifacts are typically handled by using hierarchical voting schemes [29] or by allowing each match to vote for multiple bins [30].

Recent work demonstrates that a better verification accuracy can be obtained by explicitly incorporating verification into voting schemes, *e.g.*, as a more detailed verification step [31] or when casting multiple votes [30]. However, in this paper, we show that even such advanced voting schemes still achieve lower accuracy than classic hypothesize-and-verify approaches [26]. To close this gap, we propose a novel spatial verification approach that incorporates voting into a hypothesize-and-verify framework. In detail, we propose a hierarchical voting approach to efficiently identify promising transformation hypotheses that are subsequently verified and refined on all matches. Instead of finding the correct hypothesis through random sampling, we use a progressive sampling strategy on the most probable hypotheses. Furthermore, our approach offers multiple advantages over voting-based methods: First, rather than explicitly handling quantization artifacts in the voting space, our approach only requires that a reasonable estimate for the true transformation can be obtained from some matches falling into the same bin. Quantization artifacts are then automatically handled by the subsequent verification and refinement stages. As a result, our approach is rather insensitive to the quantization of the voting space and the visual vocabulary size. Second, in contrast to voting-based methods, which usually return only a similarity score, our approach explicitly returns a transformation and a set of inliers. Hence, it can be readily combined with query expansion (QE) schemes [2, 21, 32, 33] as well as further reasoning based on the detected inliers [4, 34]. Experimental evaluation on existing and new datasets show that our approach achieves accuracy equivalent to state-of-the-art hypothesize-and-verify methods while providing runtimes faster than state-of-the-art voting-based methods. The new query and distractor image datasets and the source code for our method are released to the public at https://github.com/vote-and-verify.

2 Related Work

In the following, we discuss prior work on spatial verification in the context of image retrieval. We classify these works based on whether they employ *weak geometric models*, use RANSAC's [27] *hypothesize-and-verify framework*, or follow a *Hough voting*-based approach. In this context, our method can be seen as a hybrid between the latter two types of approaches, as it replaces RANSAC's hypothesis generation stage with a voting scheme.

Weak geometric models. Instead of explicitly estimating a geometric transformation between two images, methods based on weak geometric models either use partial transformations [25, 31] or local consistency checks [35–37]. For a feature match between two images, both Sivic & Zisserman [35] and Sattler *et al*. [36] define a consistency score based on the number of matches shared in the spatial neighborhoods of the two features. A threshold on this score is then used to prune matches. The geometry of the local features, *i.e*., their position, orientation, and scale, can be used to hypothesize a similarity transformation from a single correspondence [26, 28]. Jegou *et al*. [25] focus on the change in scale and orientation predicted by a correspondence. They quantize the space of changes into a fixed set of bins and use Hough voting to determine a subset of matches that are all similar in terms of either scale or orientation change. Pairwise Geometric Matching (PGM) of Li *et al*. [31] uses a two-stage procedure to handle noise in the voting process caused by inaccurate feature frames. First, voting provides a rough estimate for the orientation and scale change between the images, as well as putative matches. PGM then checks whether the transformations obtained from correspondence pairs are consistent with the initial estimate. To improve the runtime, Li *et al*. perform a pruning step that enforces 1-to-1 correspondences. PGM's computational complexity is $\mathcal{O}(n + m^2)$, where n is the total number of matches and m is the number of matches falling into the best bin. $O(n)$ is required for voting for the best bin and $O(m^2)$ for pairwise verification. If all matches are correct, $m = n$ holds, and the complexity is $O(n^2)$. This worst case actually happens in practice, *e.g*., when the query is a crop of a database image. In comparison, while we also apply pruning, our vote-and-verify strategy has $\mathcal{O}(n)$ complexity and achieves both faster runtimes and better verification accuracy. A drawback to methods based on weak geometric models is they only determine a similarity score and do not identify individual feature correspondences. Thus, query expansion (QE) [21, 32], which transforms features from the database into the query image, cannot be directly applied.

Hypothesize-and-verify methods. Probably the most popular approach to compute a geometric transformation in the presence of outliers is RANSAC [27] or one of its many variants [36, 38–41]. However, matching features through their visual word assignments usually generates many wrong correspondences. This makes it impractical to use any RANSAC variant that samples multiple matches in each iteration, as the runtime grows exponentially with the outlier ratio. Philbin *et al*. [26] therefore propose a more efficient spatial verification

approach, termed Fast Spatial Matching (FSM), exploiting the fact that a single correspondence already defines a transformation hypothesis. Consequently, they generate and evaluate all possible n transformations and apply local optimization [38, 39] whenever a new best model is found. FSM is the *de facto* standard for spatial verification and is used in most state-of-the-art retrieval pipelines [2, 21–25]. Its main drawback is the $\mathcal{O}(n^2)$ computational complexity due to evaluating all n hypotheses on all n correspondences. In practice, FSM can be accelerated by integrating it into a RANSAC framework and using early termination once the probability of finding a better model falls below a given threshold. Still, the worst-case complexity remains $\mathcal{O}(n^2)$. Our approach uses voting to efficiently identify promising transformation hypotheses in time $\mathcal{O}(n)$, and we show that it is sufficient to progressively sample a constant number of these hypotheses, resulting in an overall complexity of $\mathcal{O}(n)$. As such, our approach can be seen as borrowing the hypothesis prioritization of PROSAC [40] and using voting to achieve linear complexity. PROSAC orders matches based on matching quality and initially only generates transformation hypotheses from matches more likely to be correct. However, PROSAC is not directly applicable in our scenario, since matching via visual words does not provide an estimate of the matching quality. The output of FSM is a set of inliers and a geometric transformation, which can be used as input to QE. Our approach provides the same output and can thus directly be combined with existing QE methods.

Hough voting-based approaches. Lowe [28] uses Hough voting to identify a subset of all putative matches, consisting of all matches whose corresponding similarity transformation belongs to the largest bin in the voting space. Next, they apply RANSAC on this consistent subset of matches to estimate an affine transformation. To reduce the complexity to $\mathcal{O}(n)$, Avrithis & Tolias [29] propose to restrict spatial verification to the voting stage. To mitigate quantization artifacts, their Hough Pyramid Matching (HPM) uses a hierarchical voting space, where every match votes for a single transformation on each level. For each match, they then compute a strength score based on the number of other correspondences falling in the the same bins and aggregate these strengths into an overall similarity score for the two images. Wu & Kashino propose to handle quantization artifacts by having each match vote for multiple bins [30]. For each feature match m, their Adaptive Dither Voting (ADV) scheme finds neighboring correspondences, similar to [35, 36]. If m is consistent with the transformation hypothesis of its neighbor, a vote is cast in the neighbor's bin. This additional verification step helps ADV to avoid casting unrelated votes, resulting in better accuracy compared to HPM. Since ADV finds a fixed number of nearest neighbors in the image space, its computational complexity is $\mathcal{O}(n \log n)$. Similar to methods based on weak geometric models, both HPM and ADV only provide an overall similarity score and thus cannot be directly combined with standard QE. In addition, our approach achieves superior verification accuracy at faster runtimes than ADV.

Verification during retrieval. All previously discussed methods operate as a post-processing step after image retrieval. Ideally, spatial verification should be per-

formed during the retrieval process to detect and reject incorrect votes. While there exist a variety of approaches that directly integrate geometric information [42–46], this usually comes at the price of additional memory requirements or longer retrieval times. Thus, most state-of-the-art approaches still apply spatial verification only as a post-processing step [2, 21–23].

3 Vote-and-Verify for Fast Spatial Verification

One inherent problem of hypothesize-and-verify methods is that finding a good hypothesis through either random sampling or exhaustive search often takes quite some time. In this paper, we propose to solve this problem by finding promising transformation hypotheses through voting. Starting with the most promising ones, we verify a fixed number of these hypotheses on all matches, *i.e.*, perform inlier counting. As in FSM [26], local optimization [38,39] is applied every time a new best transformation is found.

At first glance, the voting stage of our approach may seem identical to existing voting-based approaches [29–31], but there are two important differences between previous methods and our approach: i) Existing works [29–31] use voting with the aim of identifying *all* geometrically consistent matches. As such, handling quantization artifacts in the voting space is very important and the methods are rather sensitive to the number of bins in the voting space. In contrast, we only require to find transformations that need to be geometrically consistent with *some* of the matches. Matches missed due to quantization are then automatically detected during inlier counting and are subsequently used during local optimization to refine the model. ii) Instead of only using the number of matches associated with the same bin, we are also interested in the transformation induced by these matches. To this end, we find that simply taking the transformation defined by the center of a bin does not provide an accurate enough hypothesis, even for high levels of quantization. Consequently, we propose a refinement process to obtain more accurate transformation hypotheses. As a result, our approach is rather insensitive to the quantization level.

In the following, we first recapitulate the process of computing a similarity transformation from a single feature match (Sec. 3.1). We next detail the voting procedure (Sec. 3.2) and explain how to derive transformation hypotheses from the voting space (Sec. 3.3). Sec. 3.4 then describes the verification and local optimization stage, while Sec. 3.5 analyses the computational complexity of our method. Finally, Algorithm 1 gives an overview of our proposed method.

3.1 From Local Features to Similarity Transformation

Consider a local image feature f, *e.g.*, a SIFT feature [28], defined by its local feature frame $f = (f_x, f_y, f_\sigma, f_\theta)$. Here, $(f_x, f_y)^T$ and f_σ are the location and scale of the detected feature in the image, while f_θ denotes the feature orientation. Following [29], each feature f is associated with a canonical coordinate

Algorithm 1 Proposed Vote-and-Verify algorithm: VERIFY(VOTE($m_{i=1...n}$)).

procedure VOTE($m_{i=1...n}$)
  for $i = 1:::n$ **do**
    for $l = 1:::L$ **do**
      Vote for $\mathtt{C}(m_i; l)$ with $w(l)$ in Hough space
  Find bins $b_{t=1...T}$ with highest votes $w(b_t)$
  return Hypotheses $\mathtt{T}(b_{t=1...T})$ ordered in decreasing number of votes
procedure VERIFY($\mathtt{T}(b_{t=1...T})$)
  Set initial score $\hat{\mathtt{s}} = 0$ and transformation $\hat{\mathtt{T}}$ as invalid
  for $t = 1:::T$ **do**
    Verify $\mathtt{T}(b_t)$ and count the number of inliers $\mathtt{s}_t$
    if $\mathtt{s}_t > \hat{\mathtt{s}}$ **then**
      Refine $\mathtt{T}(b_t)$ with local optimization and update $\mathtt{s}_t$
      Update best score $\hat{\mathtt{s}} = \mathtt{s}_t$ and transformation $\hat{\mathtt{T}} = \mathtt{T}(b_t)$
      Update probability p of finding better $\hat{\mathtt{T}}$
      if $p < \hat{p}$ **then**
        break
  return Effective inlier count $\hat{\mathtt{s}}_{\text{eff}}$ for $\hat{\mathtt{T}}$

frame in which the feature is located at the origin with unit scale and zero orientation. The transformation $\mathtt{M}_f(\mathbf{x})$ maps a location $\mathbf{x} = (x, y)^T$ in the image to a position in the canonical frame as

$$\mathtt{M}_f(\mathbf{x}) = \frac{1}{f_\sigma}\mathtt{R}(f_\theta)\left(\begin{bmatrix}x\\y\end{bmatrix} - \begin{bmatrix}f_x\\f_y\end{bmatrix}\right) = \frac{1}{f_\sigma}\begin{bmatrix}\cos f_\theta & \sin f_\theta\\ -\sin f_\theta & \cos f_\theta\end{bmatrix}\left(\begin{bmatrix}x\\y\end{bmatrix} - \begin{bmatrix}f_x\\f_y\end{bmatrix}\right) \quad , \quad (1)$$

where the rotation matrix $\mathtt{R}(f_\theta)$ performs a clockwise rotation by f_θ degrees. Consequently, a feature match $m = (f^{\mathcal{Q}}, f^{\mathcal{D}})$ between a query image $\mathcal{Q}$ and a database image $\mathcal{D}$ defines a similarity transformation between the two images:

$$\mathtt{M}_{(f^{\mathcal{Q}}, f^{\mathcal{D}})}(\mathbf{x}) = M_{f^{\mathcal{Q}}}^{-1}\left(M_{f^{\mathcal{D}}}(\mathbf{x})\right) \quad (2)$$

$$= \frac{f_\sigma^{\mathcal{Q}}}{f_\sigma^{\mathcal{D}}}\mathtt{R}(f_\theta^{\mathcal{Q}})^T\mathtt{R}(f_\theta^{\mathcal{D}})\left(\begin{bmatrix}x\\y\end{bmatrix} - \begin{bmatrix}f_x^{\mathcal{D}}\\f_y^{\mathcal{D}}\end{bmatrix}\right) + \begin{bmatrix}f_x^{\mathcal{Q}}\\f_y^{\mathcal{Q}}\end{bmatrix} = \sigma\mathtt{R}(\theta)\mathbf{x} + \mathbf{t} \ . \quad (3)$$

In Eqn. 3, $\sigma = f_\sigma^{\mathcal{Q}}/f_\sigma^{\mathcal{D}}$, $\theta = f_\theta^{\mathcal{Q}} - f_\theta^{\mathcal{D}}$, and $\mathbf{t} = (t_x, t_y)^T$ define the relative scale, rotation angle, and translation of the similarity transformation. Thus, each match $m = (f^{\mathcal{Q}}, f^{\mathcal{D}})$ can be associated with a 4-dimensional coordinate

$$\mathtt{C}(m) = [\sigma, \theta, t_x, t_y] \ . \quad (4)$$

3.2 From Similarity Transformation to Hough Voting

Each of the n feature matches between a query and database image defines a transformation hypothesis. We are interested in determining a fixed-sized set of transformations that is consistent with as many of these n hypotheses as

possible. This can be done very efficiently using Hough voting, as similar transformations are likely to fall into the same voting bin. This allows us to obtain a set of promising transformation hypotheses from the best scoring bins. However, standard Hough voting suffers from quantization artifacts, caused by inaccuracies in the detected feature frames. This is especially problematic when only few matches are correct, in which case it becomes harder to distinguish between bins corresponding to the underlying geometric transformation and bins receiving votes from wrong matches. Thus, we use a hierarchical voting scheme similar to [29], which we describe in the following.

Following Avrithis & Tolias [29], we quantize each of the four similarity transformation parameters independently. The Hough voting space of transformations is then defined as the product space of the four individual quantizations. We discretize the space at different resolution levels $l = 0...L$ and use n_x, n_y, n_σ, and n_θ bins for translation, scale, and orientation at the finest resolution $l = 0$. Each successive resolution level divides the number of bins in half until only two bins are left for each dimension. Out of the four dimensions, only the rotation space is naturally bounded by $[0, 2\pi)$, while translation and scale in theory can take any value from $\mathbb{R}^2$ and $\mathbb{R}^+$, respectively. In practice, the space of possible scale changes σ is bounded, since feature detectors only consider a few octaves of scale space [29]. A feature match inducing a large scale change between two images can usually be safely discarded as an incorrect correspondence. Consequently, we only consider scale changes in the range $[1/\sigma_{\max}, \sigma_{\max}]$ [29]. In addition, we bound the translation parameters by $\max(|t_x|, |t_y|) \leq \max(W, H)$, where W and H are the width and height of the query image. Matches violating at least one constraint are ignored [29].

Given a feature match $m = (f^{\mathcal{Q}}, f^{\mathcal{D}})$ with transformation parameters σ, θ, and $\mathbf{t}$ as defined above, we obtain the corresponding Hough space coordinate as

$$C_x(t_x, l) = 2^{-l} \lfloor n_x \left(t_x + \max(W, H)\right) / \left(2 \max(W, H)\right) \rfloor \ , \tag{5}$$

$$C_y(t_y, l) = 2^{-l} \lfloor n_y \left(t_y + \max(W, H)\right) / \left(2 \max(W, H)\right) \rfloor \ , \tag{6}$$

$$C_\sigma(\sigma, l) = 2^{-l} \lfloor n_\sigma \left(\log_2(\sigma) + \log_2(\sigma_{\max})\right) / \left(2 \log_2(\sigma_{\max})\right) \rfloor \ , \tag{7}$$

$$C_\theta(\theta, l) = 2^{-l} \lfloor n_\theta \left(\theta + \pi\right) / \left(2\pi\right) \rfloor \ . \tag{8}$$

For uniform sampling in the scale space, we linearize the scale change using the logarithmic function. The factor 2^{-l} normalizes the Hough coordinates to the respective resolution level of the voting space. Here, each match $m = (f^{\mathcal{Q}}, f^{\mathcal{D}})$ determines a coordinate $\mathtt{C}(m, l)$ at level l in the voting space, with

$$\mathtt{C}(m, l) = [C_x(t_x, l), C_y(t_y, l), C_\sigma(\sigma, l), C_\theta(\theta, l)] \ . \tag{9}$$

The match m then contributes a level-dependent weight $w(l) = 2^{-l}$ to the score of its corresponding voting bin at level l. Next, we describe how to use these scores to detect the T most promising transformation hypotheses.

3.3 Hypothesis Generation

The goal of our voting scheme is to provide a set of transformation hypotheses for subsequent verification. As described in the last section, the center of any bin

in the hierarchical representation defines a transformation, and we could simply pick the transformations corresponding to the bins with the highest scores. For coarser levels in the hierarchy, however, it is unlikely that the center of the bin is close to the actual transformation between the images. As such, we only hypothesize transformations corresponding to bins at level $l = 0$. To mitigate quantization artifacts, we propagate the scores from coarser levels to the bins at level 0. Each bin b at level 0 uniquely defines a path through the hierarchy to the coarsest level L. The total score for this bin is computed by summing the scores of all bins along this path as $w(b) = \sum_{l=0...L} w(b, l)$. Finally, we simply select the T bins that received the highest scores $w(b)$.

As we will show in Sec. 4, the naive approach of associating each bin at level 0 with the transformation defined by the bin's center coordinate does not perform well, even when using a reasonably deep hierarchy. In other words, the center coordinate of a bin can be rather far away from the true image transformation. In order to obtain a better estimate, we use the mean transformation of all matches falling into the bin instead. Let $\mathcal{M}(b)$ be the matches falling into bin b. Following Eqn. 4, the mean transformation $\mathtt{T}(b)$ is defined as $\mathtt{T}(b) = \frac{1}{|\mathcal{M}(b)|} \sum_{m \in \mathcal{M}(b)} \mathtt{C}(m)$ and can be computed efficiently during voting without a significant memory overhead by maintaining a running average. Intuitively, one can think of this as local optimization (*cf.* [38]) on the level of hypothesis generation.

It is well-known that outliers, *i.e.*, wrong matches falling into a bin, significantly impact the computation of the mean. As a more robust alternative, one could use the median transformation instead. However, the mean can be computed much more efficiently, and experiments in Sec. 4.3 show that its performance is very robust to the choice of the quantization resolution.

3.4 Accurate and Efficient Hypothesis Verification

The scores associated with each of the T similarity transformation hypotheses only provides an estimate on how well the transformation explains the matches. As a next step, we thus perform detailed hypothesis verification using inlier counting. For this stage, we follow FSM [26] and consider a match an inlier to a transformation if its two-way reprojection error is below a threshold and if the scale change between two corresponding features induced by the transformation is consistent with the scales of the two features. The $t = 1...T$ transformation hypotheses are verified in decreasing order of their scores, and we apply local optimization [38, 39] every time a new best model is found. The latter step refines the transformation by drawing a constant number of non-minimal sets of inliers to obtain a least squares estimate for the transformation. If there are at least $s = 3$ inliers, we estimate an affine transformation in local optimization. Affine transformations have been shown to perform better than similarity transformations [26], as they can handle more general geometric configurations.

We progressively verify the hypotheses ordered in decreasing number of votes until the probability $p = (1 - e)^t$ of finding a better model at the current inlier ratio e falls below a threshold. Our method is a variant of PROSAC [40] using Hough voting for ordering and 1-point sampling with a fixed-size hypothesis set.

One advantage over voting-based methods is that we explicitly determine the set of inliers. This allows us to use image similarity functions that are more discriminative than simply counting the number of inliers [4]. One such function is the *effective inlier count* [34], which has been shown to outperform raw inlier counting for image-based localization [7] and image retrieval [4] tasks. The effective inlier count is defined as

$$s_{\text{eff}} = \frac{|\cup_{i=1}^{\hat{n}} A_i|}{\sum_{i=1}^{\hat{n}} |A_i|} \hat{n} \ . \tag{10}$$

Here, $\hat{n}$ is the number of inlier matches, A_i is a region centered around the i-th inlier feature in the query image, $|\cup_{i=1}^{\hat{n}} A_i|$ is the area of the union of all regions, and $\sum_{i=1}^{\hat{n}} |A_i|$ is the maximum area that could be covered if none of the regions would overlap. In our experiments, we use square regions of size 24×24px.

3.5 Computational Complexity

For a fixed number of levels in the hierarchy, each match votes for a fixed number of bins, where each transformation bin can be computed in time $\mathcal{O}(1)$. Performing a single vote is also a constant time operation, because it only requires incrementing a counter and updating the running mean for the transformation. Consequently, voting can be done in time $\mathcal{O}(n)$ for n matches. Since T is a constant, finding and evaluating the T best hypotheses also requires time $\mathcal{O}(n)$. Each hypothesis is evaluated on all matches, which can again be done in time $\mathcal{O}(n)$. Local optimization is performed at most once for each hypothesis using a fixed-sized non-minimal set, *i.e.*, each least squares transformation can be computed in constant time as well. Thus, the overall computational complexity is linear in the number n of matches.

4 Experimental Evaluation

In this section, we first introduce the datasets and describe our experimental setup. Next, we perform an ablation study and analyze the impact of the different parameters on the performance of our approach. Finally, we provide an extensive comparison with state-of-the-art spatial verification methods.

4.1 Query and Distractor Datasets

Following standard procedure [30,31], we primarily evaluate on the Oxford5k [26] and Paris6k [47] datasets. These image sets, collected from Flickr, contain ∼5k and ∼6k images, respectively, each consisting of 11 distinct landmarks, with 5 query images per landmark. In addition, we created the new *World5k* dataset consisting of 5320 images from 61 landmarks around the world, where the images were obtained from the Yahoo 100M images dataset [48]. The landmark images were selected based on geo-tags and using overlap information from Heinly *et*

al. [49] for ground-truth. Each landmark has between 30 and 100 database images and 1 to 3 associated query photos, resulting in a total of 163 query images. Different from Oxford5k and Paris6k, which represent an object retrieval task where query photos are obtained by cropping regions from database images, our query images are full-resolution and are not contained in the database.

To simulate larger datasets and thus create harder retrieval scenarios, it is common to combine the individual datasets with an additional "distractor" dataset of ∼100k unrelated images collected from Flickr [26]. It has been shown that adding this Flickr100k distractor set significantly impacts the retrieval performance. However, the Flickr100k (F100k) dataset mostly contains very unrelated photos, obtained by searching for generic terms, such as "graffiti", "uk", or "vacation". We thus collected four additional distractor sets from the Yahoo 100M images dataset. The first distractor set consists of 140k images taken between 2km and 50km from the center of the University of Oxford. The other three distractor sets consist of images taken from 30 cities around the UK (171k images), 30 cities throughout Europe (179k), and 30 cities across the US (233k). The collections were formed using the set of geo-tagged images within a 1km radius (2km for the US images) of the geographic center of the city. We expect to find more geometrically consistent matches for distractor images, *e.g.*, due to buildings with similar architectural styles, on the new distractor datasets compared to the Flickr100k set. As such, we expect that our new distractor sets represent more challenging scenarios for spatial verification. In the following, we refer to the distractor sets as *Ox* (Oxford), *UK*, *EU* (Europe), and *US*. Further information about our new datasets is available in the supplementary material.

4.2 Experimental Setup

Retrieval system. We employ a state-of-the-art retrieval system using Hamming embedding [25] and visual burstiness weighting [24]. We use a vocabulary containing 200k words[1] to quantize RootSIFT descriptors [2, 28] extracted from keypoints provided by an upright Hessian affine feature detector [50]. Following standard procedure [23], we ensure that the vocabulary used for each dataset has been trained on another image collection. Correspondingly, we use a vocabulary trained on Oxford5k for the experiments on Paris6k and our new dataset. A vocabulary trained on Paris6k is then used for all experiments performed on the Oxford5k dataset. After retrieval, the top-1000 ranked database images are considered for spatial verification and re-ranked based on the similarity scores computed during verification. We enforce 1-to-1 matches prior to verification [31], since initial experiments showed that this significantly improved verification efficiency and quality for all spatial verification methods.

Evaluation protocol. We follow the standard evaluation procedure and assess the verification performance using mean average precision (mAP), which essentially averages the area under the precision-recall curves. For each verification

[1] Results obtained with 20k and 1M words can be found in the supplementary material.

approach, we report the total time in seconds required to verify the 1000 top-ranked retrievals for all query images. We ignore retrieval and setup time, *e.g.*, the time required to enforce a 1-to-1 matching, since this is separate from verification. To facilitate comparability, we run single-threaded implementations on an Intel E5-2697 2.7GHz CPU with 256GB RAM.

Table 1. The impact of the number of voting bins (n_σ, n_θ, n_x, n_y) and the number T of verified transformation hypotheses on the performance and efficiency of our method. Results, obtained on Oxford5k, with the highest mAP (80.1%) are highlighted in green.

T		10						20						30					
$n_{x/y}$		16		32		64		16		32		64		16		32		64	
n_σ	n_θ	mAP	time	mAP	time	mAP	time	mAP	time	mAP	time	mAP	time	mAP	time	mAP	time	mAP	time
8	8	**80.1**	0.7	79.7	0.7	79.8	0.8	79.9	0.8	79.9	0.9	79.9	0.9	80.0	0.9	79.9	0.9	80.0	1.0
8	16	79.9	0.7	79.7	0.8	79.8	0.8	80.0	0.8	79.8	0.9	79.9	0.9	80.0	0.8	79.9	0.9	80.0	1.0
8	32	79.7	0.7	79.8	0.8	79.9	0.8	80.0	0.9	79.9	0.9	79.9	0.9	79.9	0.9	79.9	0.9	80.0	1.0
16	8	79.9	0.7	79.7	0.7	79.8	0.8	80.0	0.8	79.9	0.9	79.9	0.9	**80.1**	0.8	80.0	0.9	**80.1**	0.9
16	16	79.9	0.8	79.8	0.7	79.8	0.8	80.0	0.9	79.9	0.9	79.9	0.9	80.0	1.0	80.0	0.9	80.0	0.9
16	32	79.7	0.7	79.9	0.8	79.8	0.8	80.0	0.8	79.9	0.9	80.0	0.9	80.0	0.9	80.0	0.9	80.0	0.9
32	8	79.9	0.7	80.0	0.8	79.8	0.8	80.0	0.8	80.0	0.9	79.9	0.9	80.0	0.9	80.0	0.9	**80.1**	0.9
32	16	79.7	1.0	79.9	0.7	79.7	0.8	80.0	0.9	80.0	0.9	79.9	0.9	80.0	0.8	80.0	0.9	**80.1**	0.9
32	32	79.8	0.7	79.7	0.9	80.0	0.8	79.9	0.8	80.0	1.0	80.0	0.9	80.0	0.9	80.0	1.1	**80.1**	0.9

4.3 Ablation Study

We evaluate the impact of the different parameters of our approach on its verification performance and efficiency. All experiments presented in this section have been performed on the Oxford5k dataset without any distractor images.

Impact of the level of quantization. As a first experiment, we evaluate the impact of the number of bins for rotation (n_θ), scale (n_σ) and translation (n_x, n_y) as well as the number T of transformation hypotheses that are verified. Tab. 1 shows the results obtained for different parameter configurations. For this experiment, we used the mean transformation per bin to generate the hypotheses (*cf.* Sec. 3.3). As can be seen from the table, the verification performance of our approach is rather insensitive against the number of bins and verified transformations, although increasing T has a slightly positive impact on the measured mAP. Naturally, increasing the number of bins and T also increases the overall runtime, but the increase is rather small. We also experimented with fewer (2 and 4) and more (128) bin sizes but found that the former resulted in a significant drop in mAP while the latter did not noticeably improve mAP. For all following experiments, we use $T = 30$ transformation proposals, $n_x = 64$ and $n_y = 64$ translation bins, $n_\sigma = 32$ scale bins, and $n_\theta = 8$ rotation bins.

Impact of refining the transformation hypotheses. In Sec. 3.3, we proposed to use the mean transformation for the bins in the voting space, arguing that the transformation defined by the center coordinate is not accurate enough. We measure an mAP of 76.2 when using the center coordinate and an mAP of 80.1 when using the mean transformation. At the same time, we do not observe an increase in runtime when computing the running mean. This clearly confirms

our approach for refining the transformation hypotheses. In addition, we also experimented with using the median transformation per bin. As expected, the measured mAP increases to 80.3 since the median is less affected by outliers than the mean. However, this increase comes at significantly slower runtimes of 2.5 seconds, compared to 0.9 seconds when using the mean. This increase is caused by the fact that computing the median requires the individual transformations to be stored and then partially sorted.

4.4 Comparison with State-of-the-Art Spatial Verification Methods

In the next experiments, we verify the claim that our approach achieves a verification accuracy similar to hypothesize-and-verify methods while obtaining faster runtimes than voting-based methods. Towards this goal, we compare our approach against state-of-the-art methods for spatial verification. Hypothesize-and-verify approaches are represented by different variants of FSM [26]: The original *FSM* method exhaustively evaluates each transformation hypothesis obtained from a single feature match. The 1-point-RANSAC version of FSM (*FSM-R*) randomly samples from the hypotheses and terminates spatial verification once the probability of finding a better hypothesis falls below a threshold of $\hat{p} = 0.99$. For both FSM and FSM-R, we use two variants that either estimate an affine (*Aff.*) or a similarity (*Sim.*) transformation from each correspondence. All variants use local optimization to estimate an affine transformation from the inlier matches, independent of the type of transformation estimated from the individual correspondences. Besides ranking transformation hypotheses based on their numbers of inliers, we also evaluate FSM and FSM-R in combination with the effective inlier count (*cf.* Eqn. 10). We again evaluate two variants. The first variant uses the effective inlier count instead of the standard inlier count during verification (*Eff. Inl. Eval*). The second variant simply applies the effective inlier count as a post-processing step (*Eff. Inl. Post*) on the best transformation found by FSM and FSM-R. The effective inlier count of this hypothesis is then used for re-ranking after spatial verification.

We also compare our approach against the current state-of-the-art approaches for voting-based verification: HPM [29], ADV [30], and PGM [31]. Since the three methods do not return inlier matches, they cannot be combined with the effective inlier count. Notice that the results reported in [30, 31] are not directly comparable due to using different types of features and vocabularies of different sizes trained on different datasets. Thus, our results were obtained with our own implementations of HPM (without idf-weighting), ADV, and PGM.

Tables 2, 3, and 4 present the accuracy and runtimes on the Oxford5k, Paris5k, and new datasets, respectively. There are multiple interesting insights to be gained from our results: Both FSM and FSM-R outperform HPM, ADV, and PGM in terms of mAP. The result is especially pronounced on the Paris6k dataset (*cf.* Tab. 3). Using early stopping (FSM-R) rather than evaluating all possible transformations (FSM) significantly accelerates the verification without a significant impact on mAP. In fact, FSM-R is not more than a factor-of-4 slower than ADV, which is surprising given that one of the main arguments [29]

for voting-based methods is that they are about an order of magnitude faster than FSM. Moreover, there is little difference between using an affine or similarity transformation for both FSM and FSM-R, likely due to the local optimization step. Compared to both ADV and PGM, our method achieves faster runtimes, which is most pronounced on our new dataset, where more features are found in each image. At the same time, our method also achieves a better accuracy. Especially on the Oxford5k and Paris6k datasets, our approach performs nearly as well as FSM and FSM-R, which are significantly slower than our method.

Table 2. Verification accuracy and efficiency measured on the Oxford5k dataset. The best, second-best, and third-best results are highlighted for each column.

	–	F100k	Ox	UK	EU	US	Ox+UK	Ox+EU	Ox+US	UK+EU	UK+US	EU+US	Ox+UK+EU	Ox+UK+US	Ox+EU+US	UK+EU+US	Ox+UK+EU+US	All
mAP [%]																		
Pure Retrieval	76.2	66.4	64.1	60.3	60.3	59.6	57.6	57.4	57.2	56.3	55.9	55.8	54.4	54.3	54.1	53.7	52.3	51.7
FSM Aff	**79.9**	75.1	72.9	**70.3**	70.9	71.4	68.6	69.2	69.7	68.3	68.6	69.1	67.2	67.3	67.9	67.5	66.4	66.3
+ Eff. Inl. Eval	**79.8**	**75.9**	**73.7**	**71.0**	**71.8**	**72.5**	**69.7**	**70.3**	**71.0**	**69.6**	**69.9**	**70.5**	**68.6**	**68.8**	**69.5**	**69.1**	**68.1**	**67.9**
+ Eff. Inl. Post	79.6	75.2	73.3	70.1	**71.1**	71.7	68.8	69.6	70.2	**68.5**	68.8	**69.5**	67.5	67.7	68.4	67.8	66.8	66.6
FSM-R Aff	**79.9**	75.3	72.9	**70.3**	70.9	71.4	68.6	69.2	69.6	68.3	68.6	69.1	67.2	67.3	67.9	67.4	66.4	66.3
+ Eff. Inl. Eval	**79.8**	**75.9**	**73.7**	**71.0**	**71.8**	**72.5**	**69.7**	**70.3**	**70.9**	**69.6**	**69.9**	**70.5**	**68.6**	**68.8**	**69.4**	**69.0**	**68.1**	**67.7**
+ Eff. Inl. Post	79.6	75.2	73.1	70.0	71.0	71.7	68.6	69.5	70.0	68.3	68.7	69.4	67.3	67.5	68.3	67.6	66.6	66.3
FSM Sim	**79.8**	74.8	72.3	69.4	70.2	70.9	67.6	68.3	68.9	67.3	67.7	68.3	66.1	66.4	67.0	66.5	65.5	65.3
+ Eff. Inl. Eval	**79.9**	**75.4**	**73.5**	**70.3**	**71.2**	**72.1**	**69.0**	**69.8**	70.5	**68.6**	**69.0**	**69.7**	**67.7**	**68.0**	**68.7**	68.1	**67.2**	66.7
+ Eff. Inl. Post	79.1	74.4	72.5	69.1	70.0	71.0	67.7	68.5	69.2	67.2	67.8	68.5	66.2	66.6	67.4	66.6	65.7	65.3
FSM-R Sim	**79.8**	74.9	72.3	69.3	70.2	70.9	67.6	68.3	68.9	67.3	67.7	68.3	66.1	66.4	67.0	66.5	65.5	65.3
+ Eff. Inl. Eval	**79.8**	75.3	**73.4**	70.1	71.0	71.9	68.8	69.5	70.2	68.3	68.8	**69.5**	67.4	67.8	68.5	67.8	67.0	66.4
+ Eff. Inl. Post	79.0	74.3	72.4	69.0	69.9	70.8	67.6	68.3	69.0	67.1	67.6	68.3	66.1	66.5	67.2	66.5	65.5	65.2
HPM	73.4	65.4	63.3	59.7	59.6	60.0	58.1	58.1	58.2	57.1	57.1	57.4	56.0	56.0	56.3	55.8	54.9	54.4
ADV	78.5	73.7	71.9	68.2	68.8	70.0	66.8	67.4	68.3	66.1	66.5	67.0	65.1	65.4	65.9	65.2	64.3	64.0
PGM	76.0	64.6	62.5	58.7	59.1	57.9	55.5	55.3	54.9	54.6	53.9	53.8	52.6	52.3	52.0	51.4	50.2	49.4
Ours	**80.1**	74.5	71.9	68.7	69.5	69.7	67.0	67.6	67.8	66.6	66.7	67.2	65.3	65.4	65.9	65.4	64.4	64.1
+ Eff. Inl. Post	**79.8**	**75.7**	**73.5**	**70.5**	**71.1**	**72.3**	**69.2**	**69.7**	**70.6**	**68.6**	**69.2**	**69.7**	**67.6**	**68.1**	68.6	**67.9**	67.0	**66.8**
Runtime [s]																		
FSM Aff	31.1	43.4	44.1	46.3	45.9	49.7	68.0	68.1	70.1	69.9	70.5	68.7	73.5	71.2	69.9	69.8	73.3	73.9
+ Eff. Inl. Eval	309.6	356.4	381.0	456.5	446.8	408.6	476.8	467.1	438.0	500.7	488.6	475.3	523.4	495.0	484.7	512.1	522.7	506.2
+ Eff. Inl. Post	31.0	43.2	44.5	46.8	46.1	49.3	68.9	68.4	71.0	70.4	71.3	71.1	73.9	71.2	71.5	71.0	75.5	77.2
FSM-R Aff	4.3	5.7	6.6	8.4	8.0	7.5	10.4	9.9	9.7	11.2	11.1	10.6	12.1	11.2	10.7	11.3	12.0	12.0
+ Eff. Inl. Eval	193.1	237.1	258.2	333.9	324.8	282.5	347.9	339.2	303.9	370.1	357.5	344.1	387.0	363.2	354.4	381.0	390.0	378.7
+ Eff. Inl. Post	4.4	5.9	6.9	8.9	8.4	7.9	11.3	10.7	10.4	12.2	11.8	11.4	13.2	12.2	11.7	12.4	14.2	15.0
FSM Sim	33.0	45.0	47.5	48.8	47.8	51.0	70.9	70.9	73.5	72.2	73.3	72.1	75.9	73.4	74.2	73.4	76.3	76.1
+ Eff. Inl. Eval	317.3	367.3	389.6	476.3	454.6	416.4	485.7	476.2	446.9	508.1	497.5	482.6	531.2	502.1	491.8	520.8	530.7	516.1
+ Eff. Inl. Post	33.1	45.6	47.4	53.7	48.2	50.2	72.2	71.6	75.0	73.4	74.9	72.9	77.4	74.6	73.6	75.3	78.4	79.2
FSM-R Sim	4.2	5.5	6.3	8.1	7.6	7.2	10.1	9.5	9.5	10.8	10.6	10.1	11.6	10.8	10.2	10.9	11.6	11.5
+ Eff. Inl. Eval	192.5	236.6	257.0	332.4	323.2	281.2	346.0	336.4	301.7	368.7	356.2	341.3	385.9	361.4	352.9	378.1	386.7	376.2
+ Eff. Inl. Post	4.3	5.8	6.7	9.0	8.1	7.7	10.8	10.3	10.4	11.8	11.5	11.0	12.6	11.7	11.2	11.9	13.6	14.3
HPM	**1.1**	**1.4**	**1.7**	**1.9**	**1.8**	**1.6**	**2.2**	**2.2**	**1.9**	**2.3**	**2.2**	**2.2**	**2.3**	**2.2**	**2.1**	**2.4**	**2.5**	**2.6**
ADV	2.3	3.0	3.4	4.6	4.3	3.9	5.1	4.9	4.6	5.5	5.5	5.3	6.0	5.4	5.0	5.1	**5.4**	**5.3**
PGM	15.0	15.2	15.4	15.6	15.6	15.9	15.8	15.7	16.2	16.0	16.2	15.9	16.7	16.1	16.1	16.0	16.1	15.5
Ours	**0.9**	**1.6**	**1.7**	**2.2**	**2.0**	**1.8**	**2.4**	**2.3**	**2.2**	**2.5**	**2.4**	**2.4**	**2.8**	**2.7**	**2.9**	**2.9**	**3.1**	**2.8**
+ Eff. Inl. Post	**1.2**	**1.8**	**1.9**	**2.5**	**2.3**	**2.0**	**2.7**	**2.7**	**2.5**	**3.0**	**2.8**	**2.9**	**3.4**	**3.4**	**3.6**	**3.5**	5.6	5.9

The influence of the distractor sets. Tables 2, 3, and 4 report the impact of combining each dataset with various combinations of the distractor sets. While using the effective inlier count provides little benefits without distractors, we observe a noticeable gain when adding distractors and thus making the problem harder. Naturally, the best results are obtained by directly incorporating the count into the verification stage of FSM and FSM-R. However, this comes at significant runtime costs since it needs to be evaluated often. Yet, using the count as a post-processing step incurs only negligible cost. Combining the effective inlier count with our method further increases the accuracy of our method.

The distractor set showing the largest decrease in mAP in combination with the Oxford5k dataset was the image set from 30 cities around the UK and not *Ox*. This is somewhat counter-intuitive, since it might be expected that the distractor

Table 3. Verification accuracy and efficiency measured on the Paris6k dataset.

	–	F100k	EU	US	EU+US	UK+EU+US	Ox+UK+EU+US	All
mAP [%]								
Pure Retrieval	71.2	60.2	56.2	54.6	51.2	49.5	48.7	47.7
FSM Aff	74.2	**66.0**	62.1	62.0	**59.1**	**57.8**	57.1	**56.2**
+ Eff. Inl. Eval	**74.5**	**66.4**	**62.5**	**62.5**	**59.5**	**58.3**	**57.6**	**56.6**
+ Eff. Inl. Post	73.9	65.6	61.5	61.4	58.3	57.1	56.4	55.6
FSM-R Aff	74.2	**66.0**	62.1	**62.1**	**59.1**	**57.8**	**57.2**	**56.2**
+ Eff. Inl. Eval	**74.3**	**66.2**	**62.3**	**62.3**	**59.3**	**58.1**	**57.4**	**56.4**
+ Eff. Inl. Post	73.7	65.3	61.3	61.2	58.2	56.9	56.3	55.4
FSM Sim	74.1	65.7	61.8	61.8	58.8	57.5	56.9	55.9
+ Eff. Inl. Eval	**74.4**	**66.0**	**62.2**	**62.1**	**59.1**	**57.8**	**57.2**	**56.2**
+ Eff. Inl. Post	73.8	65.2	61.3	61.2	58.2	56.8	56.2	55.3
FSM-R Sim	73.9	65.4	61.5	61.5	58.6	57.3	56.7	55.7
+ Eff. Inl. Eval	74.1	65.6	61.8	61.7	58.8	57.5	56.9	55.9
+ Eff. Inl. Post	73.6	64.9	60.9	60.9	57.9	56.6	55.9	55.0
HPM	70.8	60.8	56.1	55.5	52.2	50.8	50.2	49.3
ADV	71.5	62.9	60.1	60.1	57.3	56.0	55.3	54.2
PGM	70.7	59.3	54.8	53.6	50.0	48.1	47.1	46.2
Ours	73.4	64.9	60.9	60.7	57.8	56.6	56.0	54.9
+ Eff. Inl. Post	73.9	65.6	61.9	61.8	58.9	57.5	57.0	55.9
Runtime [s]								
FSM Aff	55.9	64.7	82.2	87.2	125.1	131.6	137.6	118.6
+ Eff. Inl. Eval	545.1	668.6	782.8	800.9	901.4	964.0	987.3	982.0
+ Eff. Inl. Post	55.6	67.4	83.1	89.5	127.5	133.5	144.9	124.2
FSM-R Aff	10.8	17.5	19.5	21.0	26.7	30.5	31.6	33.8
+ Eff. Inl. Eval	314.0	463.0	529.1	541.2	614.8	671.1	688.1	723.2
+ Eff. Inl. Post	11.0	17.8	20.1	21.7	27.8	32.0	35.2	39.2
FSM Sim	61.7	68.5	91.4	97.2	137.4	142.8	151.6	120.9
+ Eff. Inl. Eval	570.0	674.0	816.6	834.8	925.8	1005.1	1013.7	983.4
+ Eff. Inl. Post	62.0	68.4	93.0	98.3	140.0	155.9	152.8	126.7
FSM-R Sim	10.3	16.6	18.7	20.1	25.4	29.4	30.3	32.3
+ Eff. Inl. Eval	310.2	459.5	523.6	536.2	608.7	663.2	679.3	715.2
+ Eff. Inl. Post	10.6	17.2	19.3	20.9	26.7	31.0	33.9	37.9
HPM	**2.2**	**2.3**	**2.8**	**2.8**	**3.8**	**4.2**	**4.4**	**4.5**
ADV	3.7	5.4	6.0	6.0	7.7	8.5	8.6	**9.1**
PGM	19.1	19.8	19.7	19.9	20.3	20.7	22.1	23.5
Ours	**2.5**	**2.8**	**3.3**	**3.3**	**4.4**	**4.6**	**4.9**	**5.4**
+ Eff. Inl. Post	**2.9**	**3.2**	**3.7**	**3.8**	**5.0**	**5.6**	**8.4**	10.0

Table 4. Verification accuracy and efficiency measured on our new dataset.

	–	EU	US	EU+US	UK+EU+US	Ox+UK+EU+US
mAP [%]						
Pure Retrieval	97.1	92.3	92.5	90.1	89.4	88.9
Aff	**98.2**	**95.8**	**96.6**	**95.2**	**94.9**	**94.7**
+ Eff. Inl. Eval	**98.2**	**95.9**	**96.8**	**95.3**	**95.0**	**94.9**
+ Eff. Inl. Post	97.9	95.5	96.4	94.9	94.6	94.4
Aff RANSAC	**98.2**	**95.8**	**96.6**	**95.2**	**94.8**	**94.7**
+ Eff. Inl. Eval	**98.1**	**95.8**	**96.6**	**95.2**	**94.9**	**94.7**
+ Eff. Inl. Post	97.8	95.4	96.3	94.7	94.4	94.3
Sim	**98.1**	**95.8**	**96.6**	**95.1**	**94.8**	94.6
+ Eff. Inl. Eval	**98.1**	**95.9**	**96.7**	**95.3**	**95.0**	**94.8**
+ Eff. Inl. Post	97.8	95.4	96.2	94.8	94.5	94.3
Sim RANSAC	**98.0**	**95.7**	96.4	95.0	94.7	94.5
+ Eff. Inl. Eval	**98.0**	**95.8**	**96.6**	**95.2**	**94.9**	**94.7**
+ Eff. Inl. Post	97.6	95.2	96.0	94.6	94.3	94.1
HPM	96.0	91.6	92.0	90.3	89.9	89.6
ADV	97.5	94.2	94.9	93.2	92.9	92.7
PGM	96.4	90.8	90.4	88.0	87.3	86.7
Ours	97.8	95.2	95.8	94.4	94.1	93.9
+ Eff. Inl. Post	97.8	95.4	96.1	94.7	94.4	94.3
Runtime [s]						
Aff	255.6	341.4	323.1	437.6	450.1	450.7
+ Eff. Inl. Eval	4234.8	6009.0	5599.8	6372.5	6629.2	6744.8
+ Eff. Inl. Post	256.0	345.2	325.9	441.8	455.7	457.0
Aff RANSAC	55.7	99.5	92.1	123.6	132.2	136.5
+ Eff. Inl. Eval	2485.8	4103.6	3757.2	4361.7	4574.0	4671.2
+ Eff. Inl. Post	57.8	104.0	96.3	128.7	138.6	143.8
Sim	270.5	360.7	339.7	466.5	478.3	475.2
+ Eff. Inl. Eval	4378.2	6146.8	5732.3	6501.5	6758.4	6864.2
+ Eff. Inl. Post	273.5	366.1	344.3	471.9	485.8	483.3
Sim RANSAC	54.8	96.2	88.9	119.3	128.1	132.2
+ Eff. Inl. Eval	2479.8	4087.6	3741.5	4338.6	4544.2	4639.7
+ Eff. Inl. Post	57.2	101.3	93.5	124.9	134.9	139.5
HPM	**12.4**	**17.1**	**16.4**	**21.6**	**21.9**	**22.3**
ADV	24.8	35.7	32.4	39.9	42.1	46.4
PGM	54.5	57.5	54.9	57.9	58.5	59.9
Ours	**13.7**	**19.8**	**19.7**	**26.7**	**27.5**	**23.4**
+ Eff. Inl. Post	**16.5**	**24.3**	**23.8**	**29.7**	**30.8**	**28.0**

set consisting of images taken close to Oxford would be likely to contain images similar to the query. However, since there are fewer cities in this area, it turns out that those (typically non-urban) images are more easily discarded during spatial verification, compared to image sets targeted toward city centers. At the same time, the UK distractor set proved harder than the Europe and US sets, despite its smaller size. The Paris6k dataset had a similar drop in accuracy using our targeted distractor sets, compared to the Flickr100k distractor set.

5 Conclusion

In this work, we presented a novel method for fast spatial verification in image retrieval. Our method is a hybrid of voting-based approaches for efficient hypotheses generation and inlier counting-based methods for accurate hypothesis verification. Comprehensive experiments demonstrate high robustness to the choice of parameters. Our method achieves superior performance in balancing precision and efficiency versus the state of the art. Furthermore, we studied the impact of distractor image distribution and introduced a new query image set, which is released to the public alongside the source code of our method.

Acknowledgement

True Price and Jan-Michael Frahm were supported in part by the NSF No. IIS-1349074, No. CNS-1405847.

References

1. Sivic, J., Zisserman, A.: Video Google: A text retrieval approach to object matching in videos. In: ICCV. (2003)
2. Arandjelović, R., Zisserman, A.: Three things everyone should know to improve object retrieval. In: CVPR. (2012)
3. Arandjelović, R., Zisserman, A.: DisLocation: Scalable descriptor distinctiveness for location recognition . In: ACCV. (2014)
4. Sattler, T., Havlena, M., Schindler, K., Pollefeys, M.: Large-Scale Location Recognition and the Geometric Burstiness Problem. In: CVPR. (2016)
5. Torii, A., Arandjelovic, R., Sivic, J., Okutomi, M., Pajdla, T.: 24/7 place recognition by view synthesis. In: CVPR. (2015)
6. Sattler, T., Weyand, T., Leibe, B., Kobbelt, L.: Image Retrieval for Image-Based Localization Revisited. In: BMVC. (2012)
7. Sattler, T., Havlena, M., Radenovic, F., Schindler, K., Pollefeys, M.: Hyperpoints and Fine Vocabularies for Large-Scale Location Recognition. In: ICCV. (2015)
8. Gammeter, S., Quack, T., Van Gool, L.: I Know What You Did Last Summer: Object-Level Auto-Annotation of Holiday Snaps. In: ICCV. (2009)
9. Weyand, T., Leibe, B.: Discovering Favorite Views of Popular Places with Iconoid Shift. In: ICCV. (2011)
10. Weyand, T., Leibe, B.: Discovering Details and Scene Structure with Hierarchical Iconoid Shift. In: ICCV. (2013)
11. Lee, G.H., Fraundorfer, F., Pollefeys, M.: Structureless Pose-Graph Loop-Closure with a Multi-Camera System on a Self-Driving Car. In: IROS. (2013)
12. Schönberger, J.L., Radenović, F., Chum, O., Frahm, J.M.: From single image query to detailed 3d reconstruction. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (2015)
13. Radenović, F., Schönberger, J.L., Ji, D., Frahm, J.M., Chum, O., Matas, J.: From dusk till dawn: Modeling in the dark. In: CVPR. (2016)
14. Schönberger, J.L., Frahm, J.M.: Structure-from-motion revisited. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (2016)
15. Jégou, H., Douze, M., Schmid, C., Pérez, P.: Aggregating local descriptors into a compact image representation. In: CVPR. (2010)
16. Perronnin, F., Dance, C.: Fisher kernels on visual vocabularies for image categorization. In: CVPR. (2007) 1–8
17. Jégou, H., Zisserman, A.: Triangulation embedding and democratic aggregation for image search. In: CVPR. (2014)
18. Arandjelović, R., Gronat, P., Torii, A., Pajdla, T., Sivic, J.: NetVLAD: CNN architecture for weakly supervised place recognition. In: IEEE Conference on Computer Vision and Pattern Recognition. (2016)
19. Radenović, F., Tolias, G., Chum, O.: CNN image retrieval learns from BoW: Unsupervised fine-tuning with hard examples. In: ECCV. (2016)
20. Gordo, A., Almazan, J., Revaud, J., Larlus, D.: Deep Image Retrieval: Learning global representations for image search. In: arXiv:1604.01325. (2016)
21. Chum, O., Mikulik, A., Perdoch, M., Matas, J.: Total Recall II: Query Expansion Revisited. In: CVPR. (2011)
22. Mikulík, A., Perdoch, M., Chum, O., Matas, J.: Learning vocabularies over a fine quantization. IJCV (2013)
23. Tolias, G., Avrithis, Y., Jégou, H.: To aggregate or not to aggregate: Selective match kernels for image search. In: ICCV. (2013)

24. Jégou, H., Douze, M., Schmid, C.: On the burstiness of visual elements. In: CVPR. (2009)
25. Jégou, H., Douze, M., Schmid, C.: Hamming Embedding and Weak Geometric Consistency for Large Scale Image Search. In: ECCV. (2008)
26. Philbin, J., Chum, O., Isard, M., Sivic, J., Zisserman, A.: Object retrieval with large vocabularies and fast spatial matching. In: CVPR. (2007)
27. Fischler, M., Bolles, R.: Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Comm. ACM (1981)
28. Lowe, D.: Distinctive image features from scale-invariant keypoints. IJCV (2004)
29. Avrithis, Y., Tolias, G.: Hough Pyramid Matching: Speeded-up geometry re-ranking for large scale image retrieval. IJCV (2014)
30. Wu, X., Kashino, K.: Adaptive Dither Voting for Robust Spatial Verification. In: ICCV. (2015)
31. Li, X., Larson, M., Hanjalic, A.: Pairwise Geometric Matching for Large-Scale Object Retrieval. In: CVPR. (2015)
32. Chum, O., Philbin, J., Sivic, J., Isard, M., Zisserman, A.: Total recall: Automatic query expansion with a generative feature model for object retrieval. In: ICCV. (2007)
33. Mikulík, A., Radenović, F., Chum, O., Matas, J.: Efficient image detail mining. In: ACCV. (2014)
34. Irschara, A., Zach, C., Frahm, J.M., Bischof, H.: From Structure-from-Motion Point Clouds to Fast Location Recognition. In: CVPR. (2009)
35. Sivic, J., Zisserman, A.: Efficient Visual Search Cast as Text Retrieval. PAMI (2009)
36. Sattler, T., Leibe, B., Kobbelt, L.: SCRAMSAC: Improving RANSAC's Efficiency with a Spatial Consistency Filter. In: ICCV. (2009)
37. Wu, X., Kashino, K.: Robust spatial matching as ensemble of weak geometric relations. In: BMVC. (2015)
38. Chum, O., Matas, J., Kittler, J.: Locally Optimized RANSAC. In: DAGM. (2003)
39. Lebeda, K., Matas, J., Chum, O.: Fixing the locally optimized ransac. In: BMVC. (2012)
40. Chum, O., Matas, J.: Matching with prosac-progressive sample consensus. In: CVPR. (2005)
41. Raguram, R., Chum, O., Pollefeys, M., Matas, J., Frahm, J.: Usac: A universal framework for random sample consensus. PAMI (2013)
42. Chum, O., Perdoch, M., Matas, J.: Geometric min-Hashing: Finding a (Thick) Needle in a Haystack. In: CVPR. (2009)
43. Zhang, Y., Jia, Z., Chen, T.: Image retrieval with geometry-preserving visual phrases. In: CVPR. (2011)
44. Johns, E.D., Yang, G.Z.: Pairwise Probabilistic Voting: Fast Place Recognition without RANSAC. In: ECCV. (2014)
45. Tolias, G., Kalantidis, Y., Avrithis, Y., Kollias, S.: Towards large-scale geometry indexing by feature selection. CVIU (2014)
46. Shen, X., Lin, Z., Brandt, J., Wu, Y.: Spatially-constrained similarity measure for large-scale object retrieval. PAMI (2014)
47. 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)
48. Thomee, B., Shamma, D.A., Friedland, G., Elizalde, B., Ni, K., Poland, D., Borth, D., Li, L.J.: Yfcc100m: The new data in multimedia research. Comm. ACM (2016)

49. Heinly, J., Schönberger, J.L., Dunn, E., Frahm, J.M.: Reconstructing the world* in six days *(as captured by the yahoo 100 million image dataset). In: CVPR. (2015)
50. Perdoch, M., Chum, O., Matas, J.: Efficient representation of local geometry for large scale object retrieval. In: CVPR. (2009)