

# Large scale image search

H. Jegou, M. Douze and C. Schmid  
INRIA, LEAR, LJK  
Grenoble  
firstname.lastname@inria.fr

## Abstract

*This paper introduces recent methods for large scale image search. State-of-the-art methods build on the bag-of-features image representation. We first analyze bag-of-features in the framework of approximate nearest neighbor search. This shows the sub-optimality of such a representation for matching descriptors and leads us to derive a more precise representation based on 1) Hamming embedding (HE) and 2) weak geometric consistency constraints (WGC). HE provides binary signatures that refine the matching based on visual words. WGC filters matching descriptors that are not consistent in terms of angle and scale. HE and WGC are integrated within an inverted file and are efficiently exploited for all images, even in the case of very large datasets. Experiments performed on a dataset of million images show a significant improvement due to the binary signature and the weak geometric consistency constraints, as well as their efficiency. Estimation of the full geometric transformation, i.e., a re-ranking step on a short list of images, is complementary to our weak geometric consistency constraints and allows to further improve the accuracy.*

## 1 Introduction

We address the problem of searching for similar images in a large set. Similar images are defined as images of the same object or scene viewed under different imaging conditions. Many previous approaches have addressed the problem of matching such transformed images [6, 8, 7, 15, 9]. They are in most cases based on local invariant descriptors, and either match descriptors between individual images or search for similar descriptors in an efficient indexing structure. Various approximate nearest neighbor search algorithms such as kd-tree [6] or sparse coding with an over-complete basis set [10] allow for fast search in small datasets. The problem with these approaches is that all individual descriptors need to be compared to and stored.

In order to deal with large image datasets, Sivic and Zisserman [15] introduced the bag-of-features (BOF) image representation in the context of image search. Descriptors are quantized into visual words with the  $k$ -means algorithm. An image is then represented by the frequency histogram of visual words obtained by assigning each descriptor of the image to the closest

visual word. Fast access to the frequency vectors is obtained by an inverted file system. Note that this approach is an approximation to the direct matching of individual descriptors and somewhat decreases the performance. It compares favorably in terms of memory usage against other approximate nearest neighbor search algorithms, such as the popular Euclidean locality sensitive hashing (LSH) [1, 14]. LSH typically requires 100–500 bytes per descriptor to index, which is not tractable, as a one million image dataset typically produces up to 2 billion local descriptors.

Some recent extensions of the BOF approach speed up the assignment of individual descriptors to visual words [9, 11] or the search for frequency vectors [3, 2]. Others improve the discriminative power of the visual words [13], in which case the entire dataset has to be known in advance. It is also possible to increase the performance by regularizing the neighborhood structure [3] or using multiple assignment of descriptors to visual words [3, 12] at the cost of reduced efficiency. Finally, post-processing with spatial verification, a re-occurring technique in computer vision [6], improves the retrieval performance. Such a post-processing is evaluated in [11].

In this paper we present an approach complementary to those mentioned above. We make the distance between visual word frequency vectors more significant by using a more informative representation. Firstly, we apply a Hamming embedding (HE) to the descriptors by adding binary signatures which refine the visual words. The idea of using short binary codes was recently proposed in [16] to compact global GIST descriptors, and in [4] for SIFT descriptors. Secondly, we integrate weak geometric consistency (WGC) within the inverted file system which penalizes the descriptors that are not consistent in terms of angle and scale. We also use a-priori knowledge on the transformations for further verification. This contribution can be viewed as an answer to the question stated in [11] of how to integrate geometrical information in the index for very large datasets.

This paper is organized as follows. The interpretation of a BOF representation as an image voting system is given in Section 2. Our contributions, HE and WGC, are respectively described in sections 3 and 4. Complexity issues of our approach in the context of an inverted file system are discussed in Section 5. Finally, Section 6 presents the experimental results.

## 2 Voting interpretation of bag-of-features

In this section, we show how image search based on BOF can be interpreted as a voting system which matches individual descriptors with an approximate nearest neighbor (NN) search. We then evaluate BOF from this point of view.

### 2.1 Voting approach

Given a query image represented by its local descriptors  $y_{i'}$  and a set of database images  $j$ ,  $1 \leq i \leq n$ , represented by its local descriptors  $x_{i,j}$ , a voting system can be summarized as follows:

1. Dataset images scores  $s_j$  are initialized to 0.
2. For each query image descriptor  $y_{i'}$  and for each descriptor  $x_{i,j}$  of the dataset, increase the score  $s_j$  of the corresponding image by

$$s_j := s_j + f(x_{i,j}, y_{i'}), \quad (1)$$

where  $f$  is a matching function that reflects the similarity between descriptors  $x_{i,j}$  and  $y_{i'}$ . In many systems  $f(\cdot, \cdot)$  is based on  $\varepsilon$ -search or  $k$ -NN search.

3. The image score  $s_j^* = g_j(s_j)$  used for ranking is obtained from the final  $s_j$  by applying a post-processing function  $g_j$ . It can formally be written as

$$s_j^* = g_j \left( \sum_{i'=1..m'} \sum_{i=1..m_j} f(x_{i,j}, y_{i'}) \right). \quad (2)$$

The simplest choice for  $g_j$  is the identity, which leads to  $s_j^* = s_j$ . In this case the score reflects the number of matches between the query and each database image. Note that this score counts possible multiple matches of a descriptor. Another popular choice is to take into account the number of image descriptors, for example  $s_j^* = s_j/m_j$ . The score then reflects the rate of descriptors that match.

### 2.2 Bag-of-features: voting and approximate NN interpretation

Bag-of-features (BOF) image search uses descriptor quantization. A quantizer  $q$  is formally a function

$$q : \begin{array}{l} \mathbb{R}^d \rightarrow [1, k] \\ x \mapsto q(x) \end{array} \quad (3)$$

that maps a descriptor  $x \in \mathbb{R}^d$  to an integer index. The quantizer  $q$  is often obtained by performing  $k$ -means clustering on a learning set. The resulting centroids are also referred to as *visual words*. The quantizer  $q(x)$  is then the index of the centroid closest to the descriptor  $x$ . Intuitively, two descriptors  $x$  and  $y$  which are

close in descriptor space satisfy  $q(x) = q(y)$  with a high probability. The matching function  $f_q$  defined as

$$f_q(x, y) = \delta_{q(x), q(y)}, \quad (4)$$

allows the efficient comparison of the descriptors based on their quantized index. Injecting this matching function in (2) and normalizing the score by the number of descriptors of both the query image and the dataset image  $j$ , we obtain

$$s_j^* = \frac{1}{m_j m'} \sum_{i'=1..m'} \sum_{i=1..m_j} \delta_{q(x_{i,j}), q(y_{i'})} = \sum_{l=1..k} \frac{m'_l}{m'} \frac{m_{l,j}}{m_j}, \quad (5)$$

where  $m'_l$  and  $m_{l,j}$  denote the numbers of descriptors, for the query and the dataset image  $j$ , respectively, that are assigned to the visual word  $l$ . In this equation, the normalizing value  $m'$  does not affect the ordering of the dataset images. Note that these scores correspond to the inner product between two BOF vectors. They are computed very efficiently using an inverted file, which exploits the sparsity of the BOF, i.e., the fact that  $\delta_{q(x_{i,j}), q(y_{i'})} = 0$  for most of the  $(i, j, i')$  tuples.

At this point, these scores do not take into account the *tf-idf* weighting scheme (see [15] for details), which weights the visual words according to their frequency: rare visual words are assumed to be more discriminative and are assigned higher weights. In this case the matching function  $f$  can be defined as

$$f_{\text{tf-idf}}(x, y) = (\text{tf-idf}(q(y)))^2 \delta_{q(x), q(y)}, \quad (6)$$

such that the *tf-idf* weight associated with the visual word considered is applied to both the query and the dataset image in the BOF inner product. Using this new matching function, the image scores  $s_j$  become identical to the BOF similarity measure used in [15]. This voting scheme normalizes the number of votes by the number of descriptors ( $L_1$  normalization). In what follows, we will use the  $L_2$  normalization instead. For large vocabularies, the  $L_2$  norm of a BOF is very close to the square root of the  $L_1$  norm. In the context of a voting system, the division of the score by the  $L_2$  norm is very similar to  $s_j^* = s_j/\sqrt{m_j}$ , which is a compromise between measuring the number and the rate of descriptor matches.

### 2.3 Weakness of quantization-based approaches

Image search based on BOF combines the advantages of local features and of efficient image comparison using inverted files. However, the quantizer reduces significantly the discriminative power of the local descriptors. Two descriptors are assumed to match if they are assigned the same quantization index, i.e., if they lie in the same Voronoi cell. Choosing the number of centroids  $k$  is a compromise between the quantization noise and the descriptor noise.

Fig. 1(b) shows that a low value of  $k$  leads to large Voronoi cells: the probability that a noisy version of

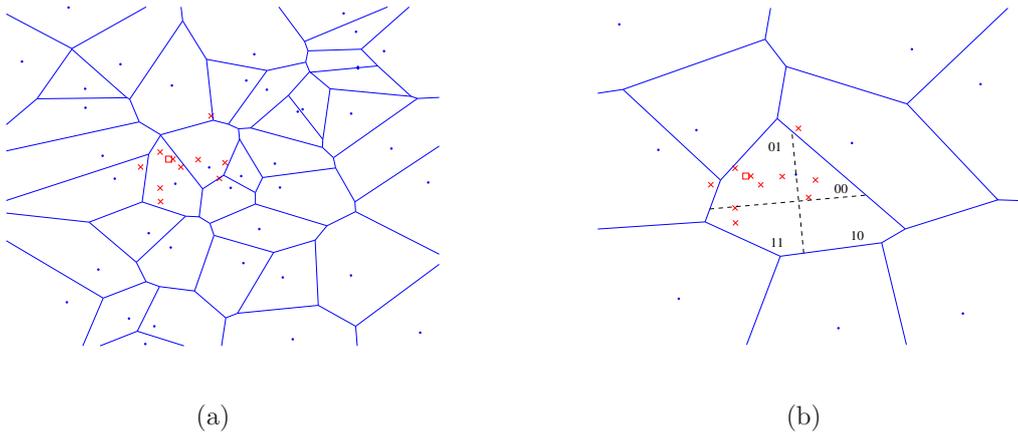


Figure 1: Illustration of  $k$ -means clustering and our binary signature. (a) Fine clustering. (b) Low  $k$  and binary signature: the similarity search within a Voronoi cell is based on the Hamming distance. Legend:  $\cdot$ =centroid,  $\square$ =descriptor,  $\times$ =noisy versions of this descriptor.

a descriptor belongs to the correct cell is high. However, this also reduces the discriminative power of the descriptor: different descriptors lie in the same cell. Conversely, a high value of  $k$  provides good precision for the descriptor, but the probability that a noisy version of the descriptor is assigned to the same cell is lower, as illustrated in Fig. 1(a).

We measure the quality of the approximate nearest neighbor search performed by BOF in terms of the trade-off between

- the average recall for the ground truth nearest neighbor
- and the average rate of vectors that match in the dataset.

Clearly, a good approximate nearest neighbor search algorithm is expected to make the nearest neighbor vote with high probability, and at the same time arbitrary vectors vote with low probability. In BOF, the trade-off between these two quantities is managed by the number  $k$  of clusters.

For the evaluation, we have used an approximate nearest neighbor evaluation set. It has been generated using the affine covariant features [8] and the SIFT descriptor [6]. A one million vector set to be searched and a test query set of 10000 vectors has been constructed. All these vectors have been extracted from the INRIA Holidays image dataset described in Section 6.

One can see in Fig. 2 that the performance of BOF as an approximate nearest neighbor search algorithm is of reasonable accuracy: for  $k = 1000$ , the NN recall is of 45% and the proportion of the dataset points which are retrieved is of 0.1%. One key advantage of BOF is that its memory usage is much lower than concurrent approximate nearest neighbor search algorithms. For instance, with 20 hash functions the memory usage of LSH [1] is of 160 bytes per descriptors compared to

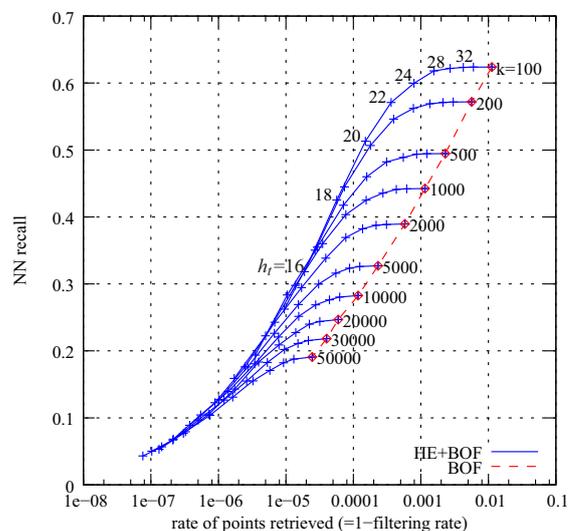


Figure 2: Approximate nearest neighbor search accuracy of BOF (dashed) and Hamming Embedding (plain) for different numbers of clusters  $k$  and Hamming thresholds  $h_t$ .

about 4 bytes for BOF. In next section, we will comment on the other curves of Fig. 2, which provide a much better performance than standard BOF.

### 3 Hamming embedding of local image descriptors

In this section, we present an approach which combines the advantages of a coarse quantizer (low number of centroids  $k$ ) with those of a fine quantizer (high  $k$ ). It consists in refining the quantized index  $q(x_i)$  with a  $d_b$ -dimensional binary signature  $b(x_i) = (b_1(x_i), \dots, b_{d_b}(x_i))$  that encodes the localization of the descriptor within the Voronoi cell, see Fig. 1(b). It is

designed so that the Hamming distance

$$h(b(x), b(y)) = \sum_{1 \leq i \leq d_b} 1 - \delta_{b_i(x), b_i(y)} \quad (7)$$

between two descriptors  $x$  and  $y$  lying in the same cell reflects the Euclidean distance  $d(x, y)$ . The mapping from the Euclidean space into the Hamming space, referred to as Hamming Embedding (HE), should ensure that the Hamming distance  $h$  between a descriptor and its NNs in the Euclidean space is small.

Note that this is significantly different from the Euclidean version of LSH (E2LSH) [1, 14], which produces several hash keys per descriptor. In contrast, HE implicitly defines a single partitioning of the feature space and uses the Hamming metric between signatures in the embedded space.

We propose in the following a binary signature generation procedure. We distinguish between 1) the *offline* learning procedure, which is performed on a learning dataset and generates a set of fixed values, and 2) the binary signature computation itself. The offline procedure is performed as follows:

1. **Random matrix generation:** A  $d_b \times d$  orthogonal projection matrix  $P$  is generated. We randomly draw a matrix of Gaussian values and apply a QR factorization to it. The first  $d_b$  rows of the orthogonal matrix obtained by this decomposition form the matrix  $P$ .
2. **Descriptor projection and assignment:** A large set of descriptors  $x_i$  from an independent dataset is projected using  $P$ . These descriptors ( $z_{i1}, \dots, z_{id_b}$ ) are assigned to their closest centroid  $q(x_i)$ .
3. **Median values of projected descriptors:** For each centroid  $l$  and each projected component  $h = 1, \dots, d_b$ , we compute the median value  $\tau_{l,h}$  of the set  $\{z_{ih} | q(x_i) = l\}$  that corresponds to the descriptors assigned to the cell  $l$ .

The fixed projection matrix  $P$  and  $k \times d_b$  median values  $\tau_{h,l}$  are used to perform the HE of a given descriptor  $x$  by:

1. **Assigning**  $x$  to its closest centroid, resulting in  $q(x)$ .
2. **Projecting**  $x$  using  $P$ , which produces a vector  $z = Px = (z_1, \dots, z_{d_b})$ .
3. **Computing the signature**  
 $b(x) = (b_1(x), \dots, b_{d_b}(x))$  as

$$b_i(x) = \begin{cases} 1 & \text{if } z_i > \tau_{q(x), i}, \\ 0 & \text{otherwise.} \end{cases} \quad (8)$$

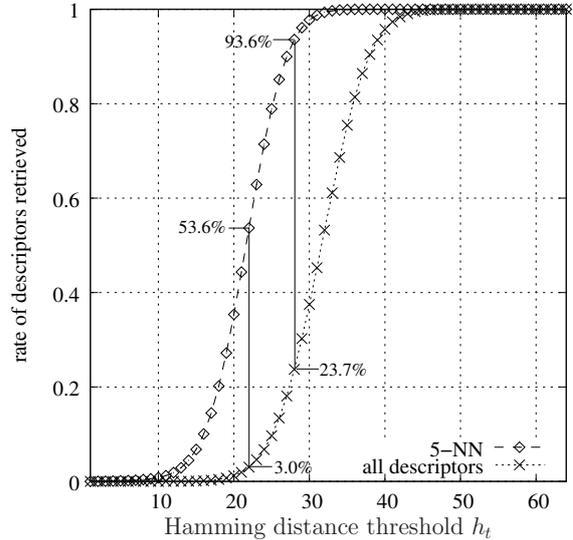


Figure 3: HE filtering effect on the descriptors within a cell and on the 5 NNs: trade-off between the rate of cell descriptors and the rate of NN that are retrieved for  $d_b = 64$ .

At this point, a descriptor is represented by  $q(x)$  and  $b(x)$ . We can now define the HE matching function as

$$f_{\text{HE}}(x, y) = \begin{cases} \text{tf-idf}(q(x)) & \text{if } q(x) = q(y) \text{ and} \\ & h(b(x), b(y)) \leq h_t \\ 0 & \text{otherwise} \end{cases} \quad (9)$$

where  $h$  is the Hamming distance defined in Eqn. 7 and  $h_t$  is a fixed Hamming threshold such that  $0 \leq h_t \leq d_b$ . It has to be sufficiently high to ensure that the Euclidean NNs of  $x$  match, and sufficiently low to filter many points that lie in a distant region of the Voronoi cell. Fig. 3 and 4 depict this compromise. These plots have been generated by analyzing a set of 1000 descriptors assigned to the same centroid. Given a descriptor  $x$  we compare the rate of descriptors that are retrieved by the matching function to the rate of 5-NN that are retrieved.

Fig. 3 shows that the choice of an appropriate threshold  $h_t$  (here between 20 and 28) ensures that most of the cell's descriptors are filtered and that the descriptor's NNs are preserved with a high probability. For instance, setting  $h_t = 22$  filters about 97% of the descriptors while preserving 53% of the 5-NN. A higher value  $h_t = 28$  keeps 94% of the 5-NN and filters 77% of the cell descriptors. Fig. 4 represents this trade-off for different binary signature lengths. Clearly, the longer the binary signature  $d_b$ , the better the HE filtering quality. In the following, we have fixed  $d_b = 64$ , which is a good compromise between HE accuracy and memory usage (8 bytes per signature).

The comparison with standard BOF shows that the approximate nearest neighbor search performed by BOF+HE is much better. This is confirmed by the quantitative evaluation of Fig. 2. Using HE for the same number of vectors that are retrieved increases the probability that the NN is among these voting vectors.

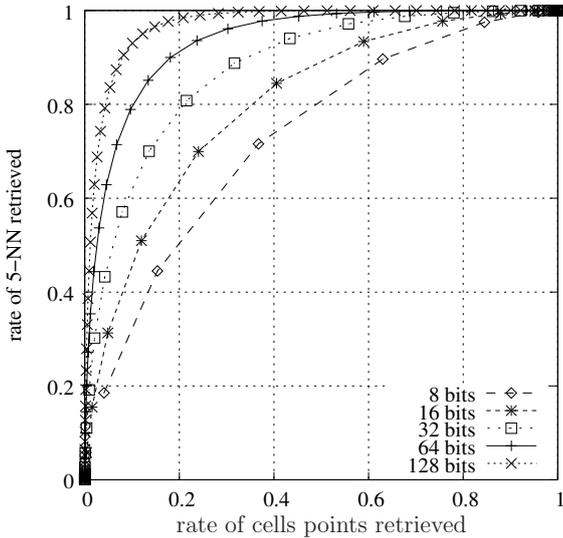


Figure 4: HE: filtering effect on the descriptors within a cell and on the 5 NNs: impact of the number of bits  $d_b$  of the binary signature length.

#### 4 Large-scale geometric consistency

BOF based image search ranks the database images without exploiting geometric information. Accuracy may be improved by adding a *re-ranking* stage [11] that computes a geometric transformation between the query and a shortlist of dataset images returned by the BOF search. To obtain an efficient and robust estimation of this transformation, the model is often kept as simple as possible [6, 11]. In [6] an affine 2D transformation is estimated in two stages. First, a Hough scheme estimates a transformation with 4 degrees of freedom. Each pair of matching regions generates a set of parameters that “vote” in a 4D histogram. In a second stage, the sets of matches from the largest bins are used to estimate a finer 2D affine transform. In [11] further efficiency is obtained by a simplified parameter estimation and an approximate local descriptor matching scheme.

Despite these optimizations, existing geometric matching algorithms are costly and cannot reasonably be applied to more than a few hundred images. In this section, we propose to exploit weak, i.e., partial, geometrical information without explicitly estimating a transformation mapping the points from an image to another. The method is integrated into the inverted file and can efficiently be applied to all images. Our weak geometric consistency constraints refine the voting score and make the description more discriminant. Note that a *re-ranking* stage [11] can, in addition, be applied on a shortlist to estimate the full geometric transformation. It is complementary to the weak consistency constraints (see Section 6).

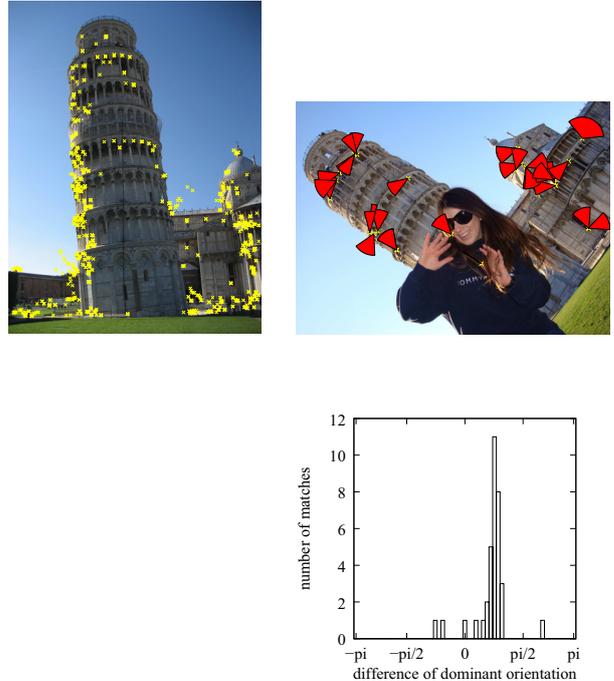


Figure 5: Orientation consistency. *Top-left*: Query image and its interest points. *Top-right*: an image of the same location viewed under an image rotation. The slices in the right top image show for each matched interest point the difference between the *estimated dominant orientations* of the query image and the image itself. *Bottom*: Histogram of the differences between the *dominant orientations* of matching points. The peak clearly corresponds to the global angle variation.

##### 4.1 Variations of geometrical characteristics: analysis

In order to obtain orientation and scale invariance, region of interest detectors extract the dominant orientation of the region [6] and its characteristic scale [5]. This extraction is performed independently for each interest point. When an image undergoes a rotation or scale change, these quantities are consistently modified for all points, see Fig 5 for an illustration in case of image rotation. It shows the difference of the dominant orientations for individual matching regions. We can observe that only the incorrect matches are not consistent with the global image rotation. This is confirmed by the histograms over the angle differences which illustrate the additional filtering effect of the weak geometric consistency constraints explained in next subsection. Similarly, the characteristic scales of interest points are consistently scaled between two images of the same scene or object.

##### 4.2 Weak geometrical consistency

The key idea of our method is to verify the consistency of the angle and scale parameters for the set of matching descriptors of a given image. We build upon and extend the BOF formalism of (1) by using *several* scores  $s_j$  per image. For a given image  $j$ , the entity

$s_j$  then represents the histogram of the angle and scale differences, obtained from angle and scale parameters of the interest regions of corresponding descriptors. Although these two parameters are not sufficient to map the points from one image to another, they can be used to improve the image ranking produced by the inverted file. This is obtained by modifying the update step of (1) as follows:

$$s_j(\delta_a, \delta_s) := s_j(\delta_a, \delta_s) + f(x_{i,j}, y_{i'}), \quad (10)$$

where  $\delta_a$  and  $\delta_s$  are the quantized angle and log-scale differences between the interest regions. The image score becomes

$$s_j^* = g \left( \max_{(\delta_a, \delta_s)} s_j(\delta_a, \delta_s) \right). \quad (11)$$

The motivation behind the scores of (11) is to use angle and scale information to reduce the scores of the images for which the points are not transformed by consistent angles and scales. Conversely, a set of points consistently transformed will accumulate its votes in the same histogram bin, resulting in a high score.

Experimentally, the quantities  $\delta_a$  and  $\delta_s$  have the desirable property of being largely independent: computing separate histograms for angle and scale is as precise as computing the full 2D histogram of (10). In this case two histograms  $s_j^a$  and  $s_j^s$  are separately updated by

$$\begin{aligned} s_j^a(\delta_a) &:= s_j^a(\delta_a) + f(x_{i,j}, y_{i'}), \\ s_j^s(\delta_s) &:= s_j^s(\delta_s) + f(x_{i,j}, y_{i'}). \end{aligned} \quad (12)$$

The two histograms can be seen as marginal probabilities of the 2D histogram. Therefore, the final score

$$s_j^* = g \left( \min \left( \max_{\delta_a} s_j^a(\delta_a), \max_{\delta_s} s_j^s(\delta_s) \right) \right) \quad (13)$$

is a reasonable estimate of the maximum of (11). This approximation will be used in the following. It significantly reduces the memory and CPU requirements. In practice, the histograms are smoothed by a moving average to reduce the angle and log-scale quantization artifacts. Note that the translation could be theoretically included in WGC. However, for a large number of images, the number of parameters should be in fewer than 2 dimensions, otherwise the memory and CPU costs of obtaining the scores would not be tractable.

### 4.3 Injecting a priori knowledge

Fig. 6(a) shows that the repartition of angle differences  $\delta_a$  between matched descriptors is different for corresponding and non-corresponding point pairs. The shallow peaks on multiples of  $\pi/2$  for non-corresponding points are due to the higher frequency of horizontal and vertical gradients in photos. The probability mass function of angle differences for corresponding points follows a highly non-uniform repartition. This is due to the human tendency to shoot

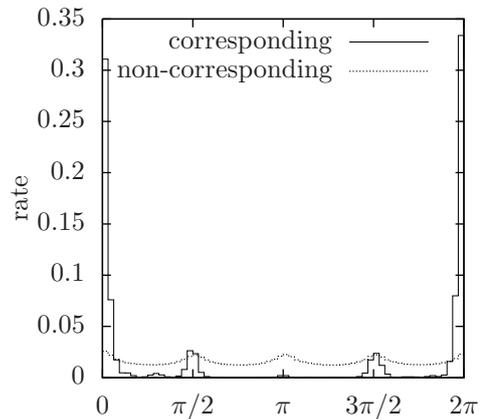


Figure 6: Histogram of  $\delta_a$  values accumulated over all query images of the Holidays dataset. Corresponding pairs are geometrically verified matching points between corresponding images. Non-corresponding pairs are HE-filtered point matches with non-corresponding images.

either in “portrait” or “landscape” mode. A similar bias is observed for  $\delta_s$ : image pairs with the same scale ( $\delta_s = 0$ ) are more frequent.

The orientation and scale priors are used to weight the entries of our histograms before extracting their maxima. We have designed two different orientation priors: “same orientation” for image datasets known to be shot with the same orientation and “ $\pm\pi/2$  rotation” for sets including non-straightened shots.

## 5 Complexity

Both HE and WGC are integrated in the inverted file. This structure is usually implemented as an array that associates a list of entries with each visual word. Each entry contains a database image identifier and the number of descriptors of this image assigned to this visual word. The tf-idf weights and the BOF vector norms can be stored separately. The search consists in iterating over the entries corresponding to the visual words in the query image and in updating the scores accordingly.

An alternative implementation consists in storing one entry per descriptor in the inverted list corresponding to a visual word instead of one entry per image. This is almost equivalent for very large vocabularies, because in this case multiple occurrences of a visual word on an image are rare, i.e., it is not necessary to store the number of occurrences. In our experiments, the overall memory usage was not noticeably changed by this implementation. This implementation is required by HE and WGC, because additional information is stored per local descriptor.

**HE impact on the complexity:** For each inverted file entry, we compute the Hamming distance between

Table 1: Inverted file memory usage.

	bits	WGC	HE	WGC+HE
image id	21	x	x	x
orientation	6	x		x
log-scale	5	x		x
binary signature	64		x	x
total bytes per entry:		4	11	12

the signature of the query and that of the database entry. This is done efficiently with a binary `xor` operation. Entries with a distance above  $h_t$  are rejected, which avoids the update of image scores for these entries. Note that this occurs for a fair rate of entries, as shown in Fig. 3.

**WGC impact on the complexity:** WGC modifies the score update by applying (12) instead of (1). Hence, two bins are updated, instead of one for a standard inverted file. The score aggregation as well as histogram smoothing have negligible computing costs. With the tested parameters, see Table 1, the memory usage of the histogram scores is 128 floating point values per image, which is small compared with the inverted lists.

**Runtime:** All experiments were carried out on 2.6 GHz quad-core computers. As the new inverted file contains more information, we carefully designed the size of the entries to fit a maximum 12 bytes per point, as shown in Table 1.

Table 2 summarizes the average query time for a one million image dataset. We observe that the binary signature of HE has a negligible computational cost. Due to the high rate of zero components of the BOF for a visual vocabulary of  $k = 200000$ , the search is faster. Surprisingly, HE reduces the inverted file query time. This is because the Hamming distance computation and thresholding is cheaper than updating the scores. WGC reduces the speed, mostly because the histograms do not fit in cache memory and their memory access pattern is almost random. Most interestingly the search time of HE + WGC is comparable to the inverted file baseline. Note that for  $k = 200000$  visual words, the assignment uses a fast approximate nearest neighbor search, i.e., the computation is not ten times slower than for  $k = 20000$ , which here uses exhaustive search.

## 6 Experiments

### 6.1 Datasets and image description

**Datasets and evaluation criterion.** We perform our experiments on two annotated datasets: the *Holidays* dataset [4], see Fig. 9, and the *Oxford5k* dataset [11]. To evaluate large scale image search we also introduce a distractor dataset downloaded from Flickr. For evaluation we use mean average precision

Table 2: Query time per image for a quad-core (Flickr1M dataset)

	$k = 20000$	$k = 200000$
compute descriptors	0.88 s	
quant. + bin. sig.	0.36 s	0.60 s
search, baseline	2.74 s	0.62 s
search, WGC	10.19 s	2.11 s
search, HE	1.16 s	0.20 s
search, HE+WGC	1.82 s	0.65 s

(mAP) [11], i.e., for each query image we obtain a precision/recall curve, compute its average precision and then take the mean value over the set of queries. In detail:

The *Holidays* dataset mainly contains personal holiday photos. The remaining ones were taken on purpose to test the robustness to various transformations: rotations, viewpoint and illumination changes, blurring, etc. The dataset includes a very large variety of scene types (natural, man-made, water and fire effects, etc) and images are of high resolution. The dataset contains 500 image groups, each of which represents a distinct scene or object. The first image of each group is the query image and the correct retrieval results are the other images of the group.

The *Oxford5k* dataset contains images of Oxford buildings. All the dataset images are in “upright” orientation because they are displayed on the web.

The *Flickr60k* and *Flickr1M* datasets contain arbitrary images from Flickr. Flickr60k is used to learn the quantization centroids and the HE parameters (median values). For these tasks we have used respectively 5M and 140M descriptors. Flickr1M are distractor images for large scale image search.

**Image description.** Descriptors are obtained by the Hessian-Affine detector [8] and the SIFT descriptor [6]. Clustering is performed with  $k$ -means on the independent Flickr60k dataset. The number of clusters is specified for each experiment.

**Impact of the clustering learning set.** Learning the visual vocabulary on a distinct dataset shows more accurately the behavior of the search in very large image datasets, for which 1) query descriptors represent a negligible part of the total number of descriptors, and 2) the number of visual words represents a negligible fraction of the total number of descriptors. This is confirmed by comparing our results on Oxford to the ones of [11], where clustering is performed on the evaluation set. In our case, i.e., for a distinct visual vocabulary, the improvement between a small and large  $k$  is significantly reduced when compared to [11], see first row of Table 3.

Table 3: Results for *Holidays* and *Oxford* datasets. mAP scores for the baseline, HE, WGC and HE+WGC. Angle prior: same orientation for *Oxford*,  $0, \pi/2, \pi$  and  $3\pi/2$  rotations for *Holidays*. Vocabularies are generated on the independent Flickr60K dataset.

	Parameters		Holidays		Oxford	
	HE: $h_t$	WGC	$k = 20000$	$k = 200000$	$k = 20000$	$k = 200000$
baseline			0.4463	0.5488	0.3854	0.3950
HE	20		0.7268	0.7093	0.4798	0.4503
HE	22		0.7181	0.7074	0.4892	0.4571
HE	24		0.6947	0.7115	0.4906	0.4585
HE	26		0.6649	0.6879	0.4794	0.4624
WGC		no prior	0.5996	0.6116	0.3749	0.3833
WGC		with prior	0.6446	0.6859	0.4375	0.4602
HE+WGC	20	with prior	0.7391	0.7328	0.5442	0.5096
HE+WGC	22	with prior	0.7463	0.7382	0.5472	0.5217
HE+WGC	24	with prior	0.7507	0.7439	0.5397	0.5252
HE+WGC	26	with prior	0.7383	0.7404	0.5253	0.5275

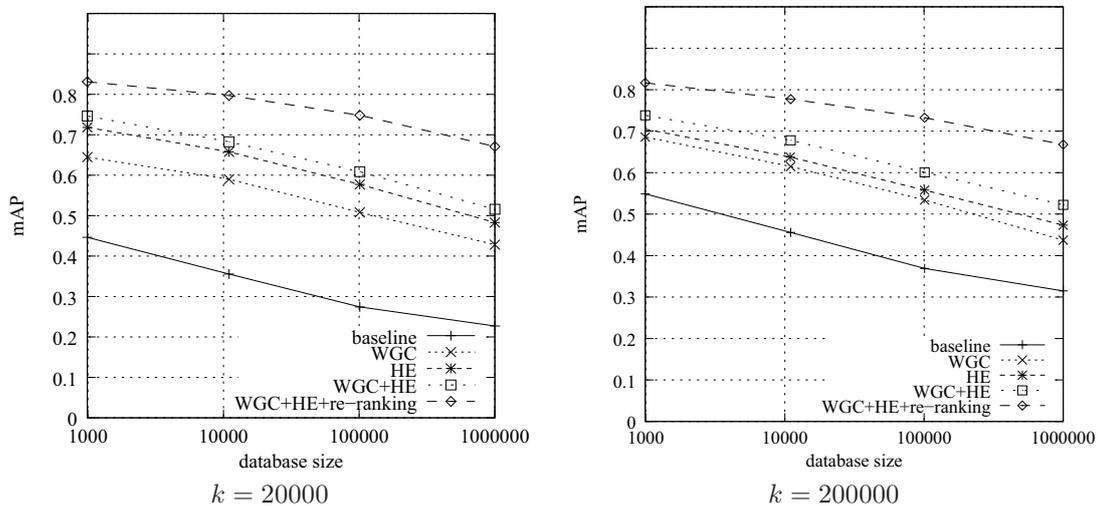


Figure 7: Performance of the image search as a function of the dataset size for BOF, WGC, HE ( $h_t = 22$ ), WGC+HE, and WGC+HE+re-ranking with a full geometrical verification (shortlist of 100 images). The dataset is *Holidays* with a varying number of distractors from *Flickr1M*.

## 6.2 Evaluation of HE and WGC

### INRIA Holidays and Oxford building datasets:

Table 3 compares the proposed methods with the standard BOF baseline. We can observe that both HE and WGC result in significant improvements. Most importantly, these approaches are complementary, as it is shown that the combination of the two further increases the performance.

**Large scale experiments:** Fig. 7 shows an evaluation of the different approaches for large datasets, i.e., we combined the Holidays dataset with a varying number of images from the 1M Flickr dataset. We clearly see that the gain of the variant WGC + HE is very significant. In the case of WGC + HE the corresponding curves degrade less rapidly when the number of images in the database increases.

Results for various queries are presented in Fig. 9. The first and second rows show that some images from the *Flickr1M* dataset artificially decrease the results in terms of mAP given in Fig. 7, as *false* false positive, marked by FFP, are some images which are actually relevant to the query image. We can observe in figure 8 that HE and WGC improve the quality of the ranking significantly for the given queries.

Table 4 measures the improvement of the ranking. It gives the rate of true positives that are in a shortlist of 100 images. For a dataset of one million images, the baseline only returns 31% of the true positives, against 62% for HE+WGC. This reflects the quality of the shortlist that will be considered in a re-ranking stage.

**Re-ranking:** The re-ranking is based on the estima-

Table 4: *Holidays dataset + Flickr1M*: Rate of true positives as a function of the dataset size for a shortlist of 100 images,  $k = 200000$ .

dataset size	991	10991	100991	1000991
BOF	0.673	0.557	0.431	0.306
WGC+HE	0.855	0.789	0.708	0.618

tion of an affine transformation with our implementation of [6]. Fig. 7 also shows the results obtained with a shortlist of 100 images. We can observe further improvement, which confirms the complementary of this step with WGC.

## 7 Conclusion

This paper has introduced two ways of improving a standard bag-of-features representation. The first one is based on a Hamming embedding which provides binary signatures that refine visual words. It results in a similarity measure for descriptors assigned to the same visual word. The second is a method that enforces weak geometric consistency constraints and uses a priori knowledge on the geometrical transformation. These constraints are integrated within the inverted file and are used for all the dataset images. Both these methods improve the performance significantly, especially for large datasets. Interestingly, our modifications do not result in an increase of the runtime.

**Acknowledgments.** We would like to acknowledge the ANR projects GAIA and RAFFUT as well as QUAERO for their financial support.

## References

[1] M. Datar, N. Immorlica, P. Indyk, and V.S. Mirrokni. Locality-sensitive hashing scheme based on p-stable distributions. In *Proceedings of the Symposium on Computational Geometry*, pages 253–262, 2004.

[2] F. Fraundorfer, H. Stewenius, and D. Nister. A binning scheme for fast hard drive based image search. In *CVPR*, 2007.

[3] H. Jegou, H. Harzallah, and C. Schmid. A contextual dissimilarity measure for accurate and efficient image search. In *CVPR*, 2007.

[4] Herve Jegou, Matthijs Douze, and Cordelia Schmid. Hamming embedding and weak geometric consistency for large scale image search. In *ECCV*, 2008.

[5] T. Lindeberg. Feature detection with automatic scale selection. *IJCV*, 30(2):77–116, 1998.

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

[7] J. Matas, O. Chum, U. Martin, and T. Pajdla. Robust wide baseline stereo from maximally stable extremal regions. In *BMVC*, pages 384–393, 2002.

[8] K. Mikolajczyk and C. Schmid. Scale and affine invariant interest point detectors. *IJCV*, 60(1):63–86, 2004.

[9] D. Nistér and H. Stewenius. Scalable recognition with a vocabulary tree. In *CVPR*, pages 2161–2168, 2006.

[10] D. Omercevic, O. Drbohlav, and A. Leonardis. High-dimensional feature matching: employing the concept of meaningful nearest neighbors. In *ICCV*, 2007.

[11] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *CVPR*, 2007.

[12] 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.

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

[14] G. Shakhnarovich, T. Darrell, and P. Indyk. *Nearest-Neighbor Methods in Learning and Vision: Theory and Practice*, chapter 3. MIT Press, Mar 2006.

[15] J. Sivic and A. Zisserman. Video Google: A text retrieval approach to object matching in videos. In *ICCV*, pages 1470–1477, 2003.

[16] A. Torralba, R. Fergus, and Y. Weiss. Small codes and large databases for recognition. In *CVPR*, 2008.

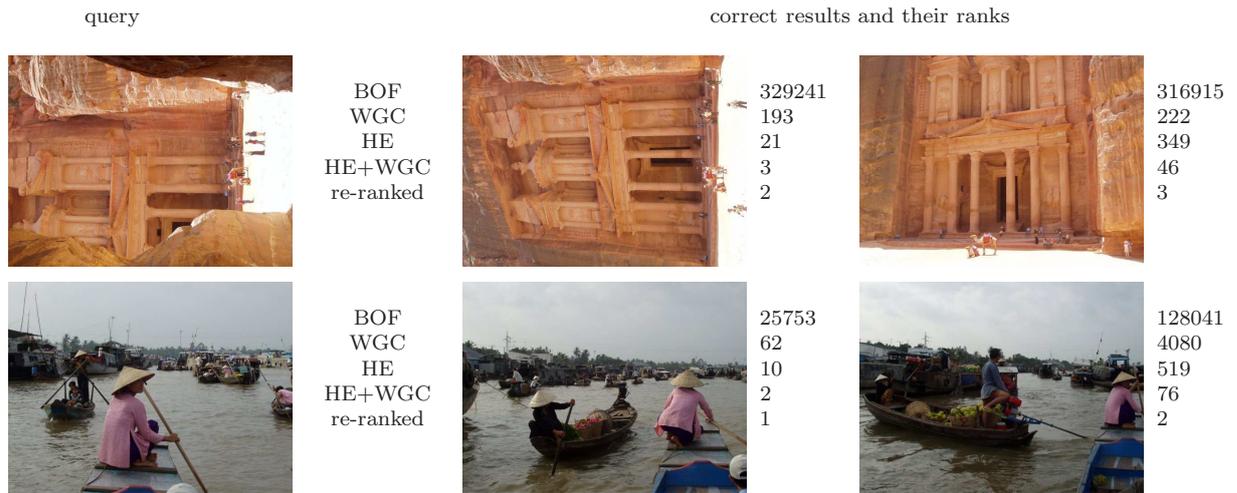


Figure 8: Two queries from the *Holidays* dataset and the ranks obtained with different methods (BOF, WGC, HE, HE+WGC, re-ranked) for two true positives. The database is *Holidays* + 1M. Note that in the first row the “easiest” true positive is not shown.

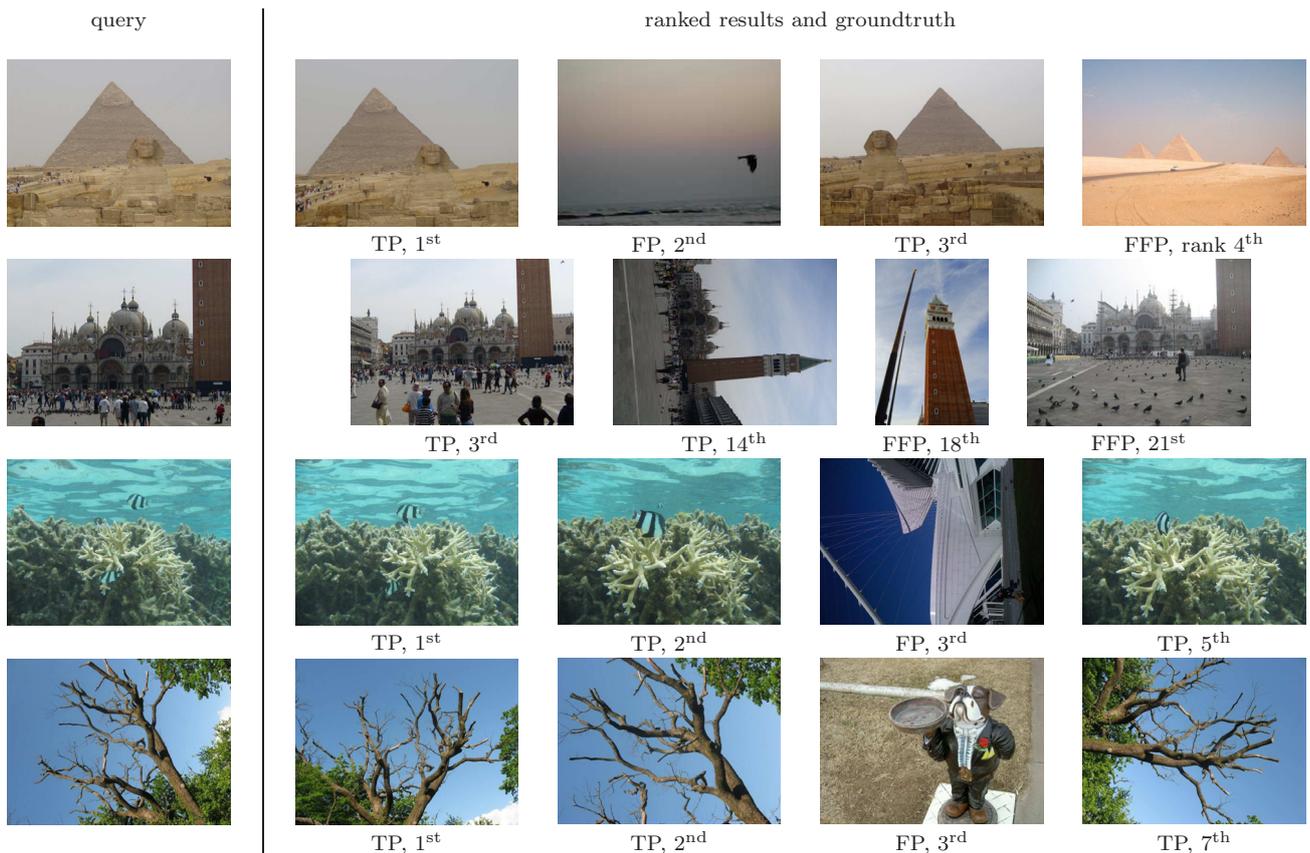


Figure 9: Queries from the *Holidays* dataset and some corresponding results for *Holidays*+1M distractors from Flickr1M. True positives are marked by TP and false positives by FP. As the *Holidays* dataset includes pictures of popular tourist attractions, matches were also found in the distractor dataset. They count as false positives and are marked by FFP (false false positive).