Hamming Embedding and Weak Geometry Consistency for Large Scale Image Search - extended version

Hervé Jégou, Matthijs Douze, Cordelia Schmid

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Hamming Embedding and Weak Geometry Consistency for Large Scale Image Search – Extended version –

Hervé Jegou — Matthijs Douze — Cordelia Schmid

N° 6709

October 2008

Thème COG

ISSN 0249-6399 ISRN INRIA/RR--6709--FR+ENG

INSTITUT NATIONAL
DE RECHERCHE
EN INFORMATIQUE
ET EN AUTOMATIQUE

INRIA

centre de recherche
GRENOBLE - RHÔNE-ALPES

Hamming Embedding and Weak Geometry Consistency for Large Scale Image Search – Extended version –

Hervé Jegou*, Matthijs Douze*, Cordelia Schmid*

Thème COG — Systèmes cognitifs
Équipe-Projet Lear

Rapport de recherche n° 6709 — October 2008 — 27 pages

Abstract: This technical report presents and extends a recent paper we have proposed 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 one million of 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.

Key-words: image retrieval, nearest neighbor search, image matching, geometrical transform, large image databases

* firstname.lastname@inria.fr

Centre de recherche INRIA Grenoble – Rhône-Alpes
655, avenue de l'Europe, 38334 Montbonnot Saint Ismier
Téléphone : +33 4 76 61 52 00 — Télécopie +33 4 76 61 52 52

Recherche d'image par Hamming Embedding et Contraintes Géometriques faibles

Résumé : Ce rapport technique reprend et étend un article récent sur la recherche d'images dans des grandes bases. Les méthodes de l'état de l'art reposent sur une représentation des images par sac de mots. Nous exprimons la mise en correspondance de ces descripteurs dans le contexte de la recherche approximative de plus proches voisins. Nous montrons que cette représentation est sous-optimale. Ceci nous amène à définir une représentation plus précise, basée sur 1) l'immersion dans un espace de Hamming (HE) et 2) des contraintes géométriques faibles (WGC). Le HE ajoute aux descripteurs une signature binaire qui permet d'affiner leur mise en correspondance. Le WGC filtre les correspondances de points dont les caractéristiques d'angle et d'échelle ne sont pas cohérentes. HE et WGC sont intégrés dans une structure de fichier inversé et appliqués à toutes les images, même pour de très grandes bases. Des expériences sur un million d'images montrent que la signature binaire et la contrainte géométrique faible améliorent significativement la précision, sans allongement des temps de calcul. Le réordonnancement des meilleures images par l'estimation d'une transformation géométrique complète est complémentaire avec notre WGC, et améliore encore la précision.

Mots-clés : recherche d'image, recherche de plus proches voisins, appariemment d'image, transformation géométriques, grandes bases d'image

1 Introduction

We address the problem of searching for similar images in a large set of images. Similar images are defined as images of the same object or scene viewed under different imaging conditions, cf. Fig. 13 for examples. Many previous approaches have addressed the problem of matching such transformed images [1, 2, 3, 4, 5]. 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 [1] or sparse coding with an overcomplete basis set [6] 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 [4] 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) [7, 8]. 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 [5, 9] or the search for frequency vectors [10, 11]. Others improve the discriminative power of the visual words [12], 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 [10] or using multiple assignment of descriptors to visual words [10, 13] at the cost of reduced efficiency. Finally, post-processing with spatial verification, a reoccurring technique in computer vision [1], improves the retrieval performance. Such a post-processing is evaluated in [9].

In this report 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 [14], where they are used to compact global GIST descriptors [15]. Secondly, we integrate a weak geometric consistency (WGC) check 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 [9] of how to integrate geometrical information in the index for very large datasets.

n	number of images in the dataset
d	dimension of the local descriptors
m_j	number of descriptors describing the image j of the dataset
m'	number of descriptors describing the query
k	number of centroids (=visual words) defining the quantizer
$x_{i,j}$	i^{th} descriptor of image j
$y_{i'}$	i'^{th} descriptor of the query image
$q(.)$	quantizer: $q(x_{i,j})$ is the quantized index associated with $x_{i,j}$
s_j^*	final score associated with dataset image j
$\delta_{x,y}$	Kronecker delta function: $\begin{cases} 1 & \text{if } x = y, \\ 0 & \text{otherwise.} \end{cases}$
$f(.,.)$	descriptor matching function, see (1)
$h(.,.)$	Hamming distance (9)

Figure 1: Notations.

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 described in sections 3 and 4. Complexity issues of our approach in the context of an inverted file system are discussed in Section 6. Finally, Section 7 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 perspective. The main notations used in this paper are summarized in Fig. 1.

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

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

where f is a matching function that reflects the similarity between descriptors $x_{i,j}$ and $y_{i'}$. For a matching system based on ε-search or $k-$NN, $f(.,.)$ is defined as

$$f_\varepsilon(x,y) = \begin{cases} 1 & \text{if } d(x,y) < \varepsilon \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

and

$$f_{k\text{-NN}}(x,y) = \begin{cases} 1 & \text{if } x \text{ is a } k\text{-NN of } y \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

where $d(.,.)$ is a distance (or dissimilarity measure) defined on the descriptor space. SIFT descriptors are typically compared using the Euclidean distance.

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

$$s_j^* = g_j\left(\sum_{i'=1..m'} \sum_{i=1..m_j} f(x_{i,j}, y_{i'})\right). \tag{4}$$

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

$$\begin{array}{rccc} q: & \mathbb{R}^d & \rightarrow & [1,k] \\ & x & \mapsto & q(x) \end{array} \tag{5}$$

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

$$f_q(x,y) = \delta_{q(x),q(y)}, \tag{6}$$

allows the efficient comparison of the descriptors based on their quantized index. Injecting this matching function in (4) and normalizing the score by the number of descriptors of both the query image and the dataset image j, we obtain

$$s_j^* = \frac{1}{m_j\ m'} \sum_{i'=1..m'} \sum_{i=1..m_j} \delta_{q(x_{i,j}),q(y_{i'})} = \sum_{l=1..k} \frac{m'_l}{m'} \frac{m_{l,j}}{m_j}, \tag{7}$$

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 [4] 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

$$f_{\text{tf-idf}}(x, y) = \left(\text{tf-idf}\left(q(y)\right)\right)^2 \delta_{q(x),q(y)}, \tag{8}$$

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 [4]. 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.

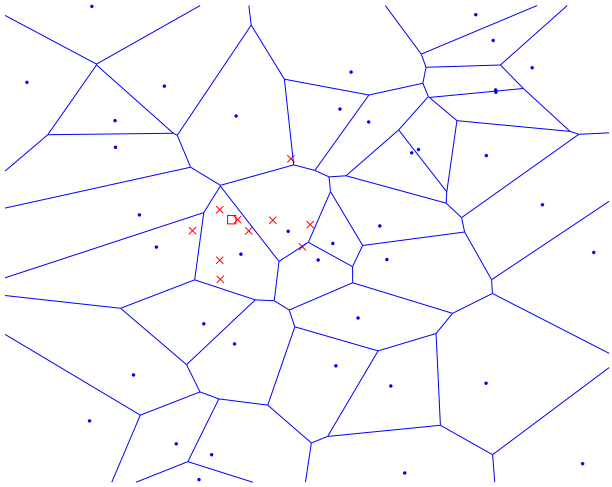
Fig. 2(b) shows that a low value of k leads to large Voronoi cells: the probability that a noisy version of a descriptor belongs to the correct cell is high. However, this also reduces the discriminative power of the descriptor: different descriptors lie in the same cell. Conversely, a high value of 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. 2(a).

Fig. 3 shows the impact of this trade-off when matching real images. The matches obtained by a BOF between two similar images are analyzed. A coarse clustering clearly leads to many bad matches, as shown in Fig. 3(a). We can observe that many of the corresponding regions are quite different. Using a larger codebook, many bad matches are removed (see Fig. 3(b)), but at the same time many correct matches are also removed.

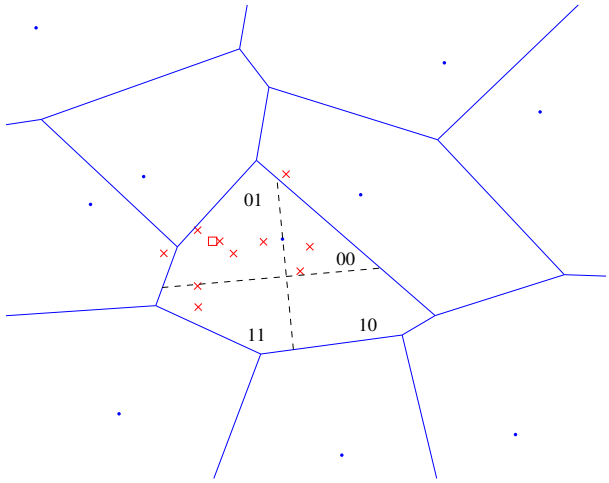
From a more quantitative point of view, have measured the quality of the approximate nearest neighbor search performed by BOF in terms of the trade-off between

- 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



(a)



(b)

Figure 2: 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. Key: $\cdot$=centroid, $\square$=descriptor, $\times$=noisy versions of this descriptor.

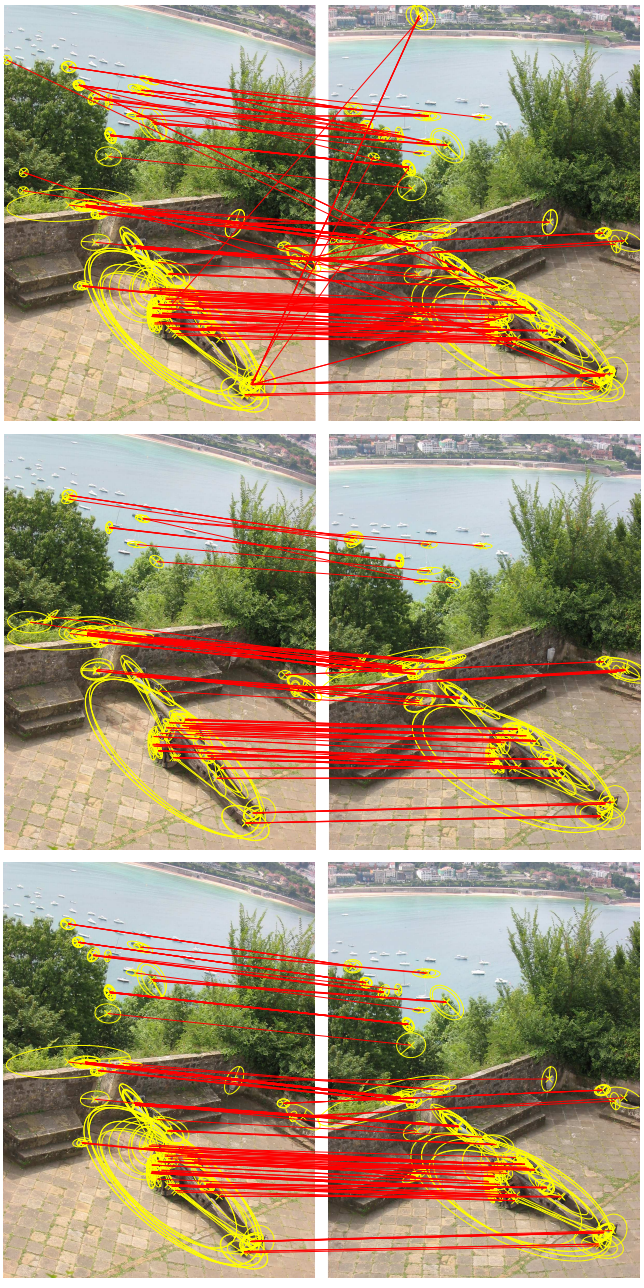


Figure 3: (a) Coarse clustering ($k = 20000$), (b) Fine clustering ($k = 200000$), (c) Coarse clustering with Hamming Embedding ($k = 20000$, $h_t = 24$)

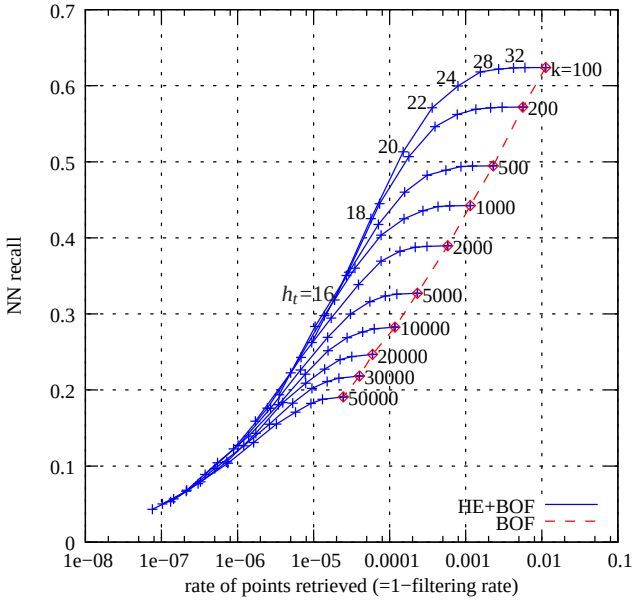


Figure 4: Approximate nearest neighbor search accuracy of BOF (dashed) and Hamming Embedding (plain) for different numbers of clusters k and Hamming thresholds h_t.

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 the approximate nearest neighbor evaluation set available at [16]. It has been generated using the affine covariant features program of [17]. A one million vector set to be searched and a test query set of 10000 vectors are provided. All these vectors have been extracted from the INRIA Holidays image dataset described in Section 7.

One can see in Fig. 4 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 [7] is of 160 bytes per descriptors compared to about 4 bytes for BOF. In the next section, we will comment on the other curves of Fig. 4, which show a much better performance than the standard BOF.

3 Hamming embedding of local 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 $b(x_i) = (b_1(x_i), \ldots, b_{d_b}(x_i))$ that encodes the localization of

the descriptor within the Voronoi cell, see Fig. 2(b). It is designed so that the Hamming distance

$$h(b(x), b(y)) = \sum_{1 \leq i \leq d_b} |b_i(x) - b_i(y)| \tag{9}$$

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

Note that this method is significantly different from the Euclidean version of LSH (E2LSH) [7, 8], 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.

3.1 Binary signature generation

We propose in the following a binary signature generation procedure. We distinguish between 1) the *off-line* 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 $(z_{i1}, ..., z_{id_b})$ are assigned to their closest centroid $q(x_i)$.

3. **Median values of projected descriptors:** For each centroid l and each projected component $h = 1, \ldots, d_b$, 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 $z = Px = (z_1, \ldots, z_{d_b})$.

3. **Computing the signature** $b(x) = (b_1(x), \ldots, b_{d_b}(x))$ as

$$b_i(x) = \begin{cases} 1 & \text{if } z_i > \tau_{q(x),i}, \\ 0 & \text{otherwise.} \end{cases} \tag{10}$$

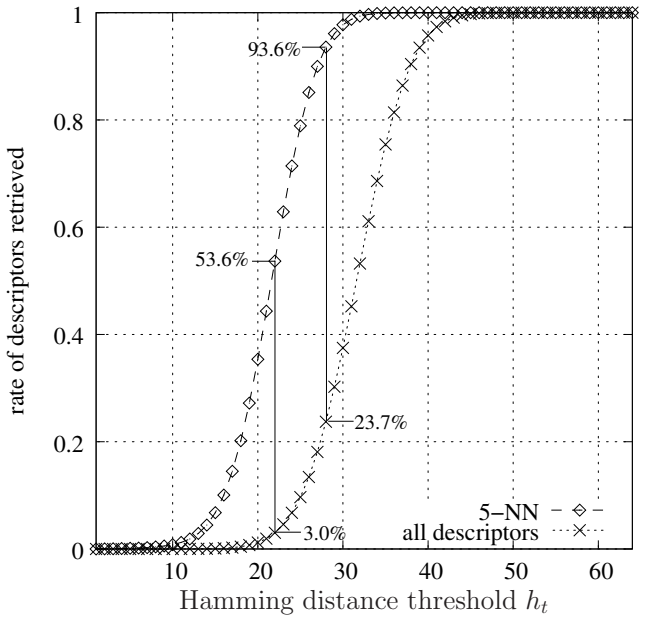


Figure 5: HE: filtering effect on the descriptors within a cell and on the 5 NNs: trade-off between the rate of cell descriptors and the rate of NN that are retrieved for $d_b = 64$.

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

$$f_{\mathrm{HE}}(x, y) = \begin{cases} \text{tf-idf}(q(x)) & \text{if } q(x) = q(y) \text{ and } h\left(b(x), b(y)\right) \leq h_t \\ 0 & \text{otherwise} \end{cases} \tag{11}$$

where h is the Hamming distance defined in (9) 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. 5 and 6 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.

3.2 Evaluation of nearest neighbor search using HE

Fig. 5 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. 6 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).

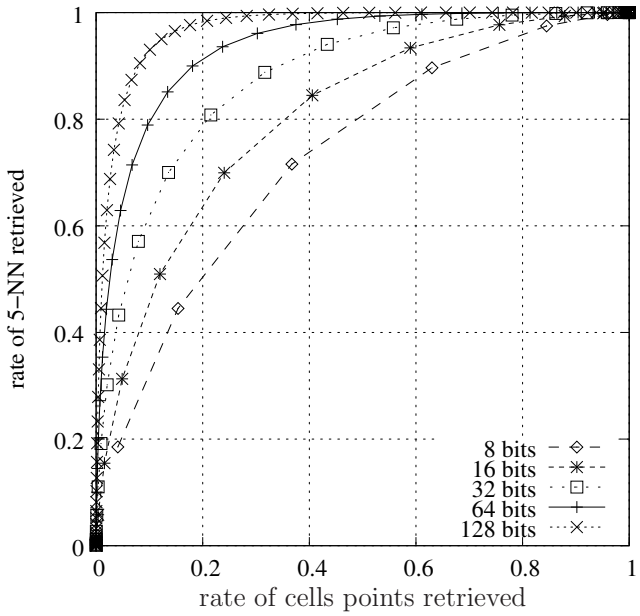


Figure 6: HE: filtering effect on the descriptors within a cell and on the 5 NNs: impact of the number of bits d_b of the binary signature length.

A comparison with standard BOF shows that the approximate nearest neighbor search performed by BOF+HE is much better. This is qualitatively shown in Fig. 3-(c), where one can observe that the bad matches have been removed without removing the correct ones. This is confirmed by the quantitative evaluation of Fig. 4. Using HE for the same number of vectors that are retrieved increases the probability that the NN is among these voting vectors.

3.3 Weighting the Hamming distance

In this section, we propose a weighting based on the Hamming distance, i.e., smaller distances result in higher matching scores. In the spirit of the *tf-idf* weighting scheme, the weight $w_d(a)$ associated with an observed distance $a = h(b(x), b(y))$ is obtained as the *Shannon information content* [18] of the outcome "distance between binary signatures is lower than or equal to a". The Shannon information content of an event with *a priori* probability p is $-\log_2 p$. This quantity gives the surprise of observing a particular event. Note that the entropy is the expectation of the Shannon information content over all realizations.

As a rough approximation, we assume that the probability mass function of binary signatures is uniform on the Hamming hypercube $\{0,1\}^{d_b}$. The weights are then given by

$$w_d(a) = -\log_2 \left(\frac{1}{2^{d_b}} \sum_{i=0}^{a} \binom{i}{d_b} \right). \qquad (12)$$

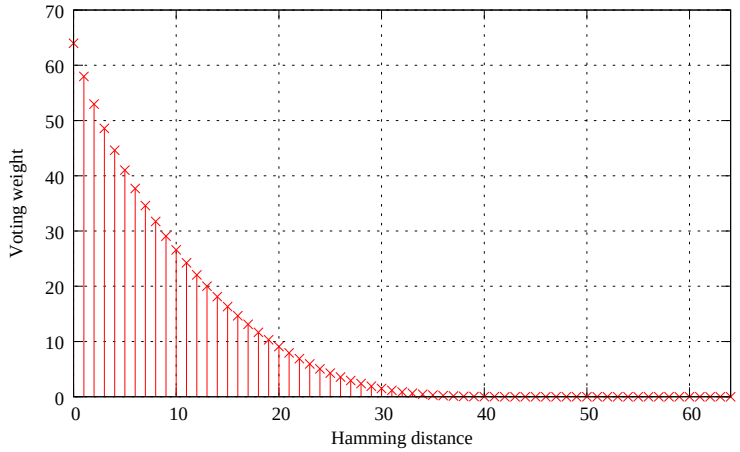


Figure 7: Weights associated with the Hamming distance for $d_b = 64$.

These weights are stored in a look-up table of $d_b + 1$ elements, corresponding to all possible Hamming distances. They are used in combination with the *tf-idf* in (8).

Fig. 7 depicts the weights as a function of the Hamming distance for $d_b = 64$. The weight obtained when the Hamming distance is equal to zero is d_b. One can observe that Hamming distances above $d_b/2$ have a very low impact on the score: their weight is lower than 1. That is why in practice we can just set them to zero, as done in (11). This improves the efficiency of querying the inverted file.

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 [9] 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 [1, 9]. In [1] 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 [9] 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 [9] can, in addition, be applied on a shortlist to estimate

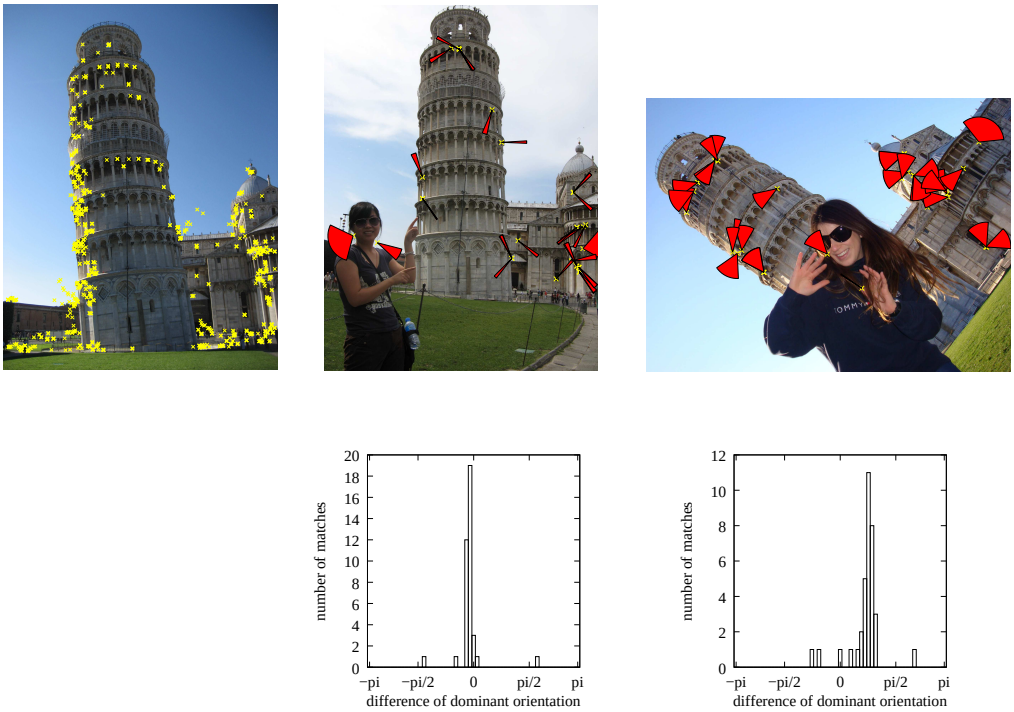


Figure 8: Orientation consistency. *Top-left:* Query image and its interest points. *Top-right:* two images of the same location viewed under different image rotations. The slices in the right top images show for each matched interest point the difference between the *estimated dominant orientations* of the query image and the image itself. Matches are obtained with our approach HE. *Bottom-right:* Histogram of the differences between the *dominant orientations* of matching points. The peak clearly corresponds to the global angle variation.

the full geometric transformation. It is complementary to the weak consistency constraints (see Section 7).

4.1 Analysis of weak geometric information

In order to obtain orientation and scale invariance, region of interest detectors extract the dominant orientation of the region [1] and its characteristic scale [19]. 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 8 for an illustration in the case of image rotations. It shows the difference in dominant orientations for pairs of 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 angle differences: for two images with different geometrical layout, the histogram of orientation differences is uniformly distributed.

Similarly, the characteristic scales of interest points are consistently scaled between two images of the same scene or object, as shown on Fig. 9.

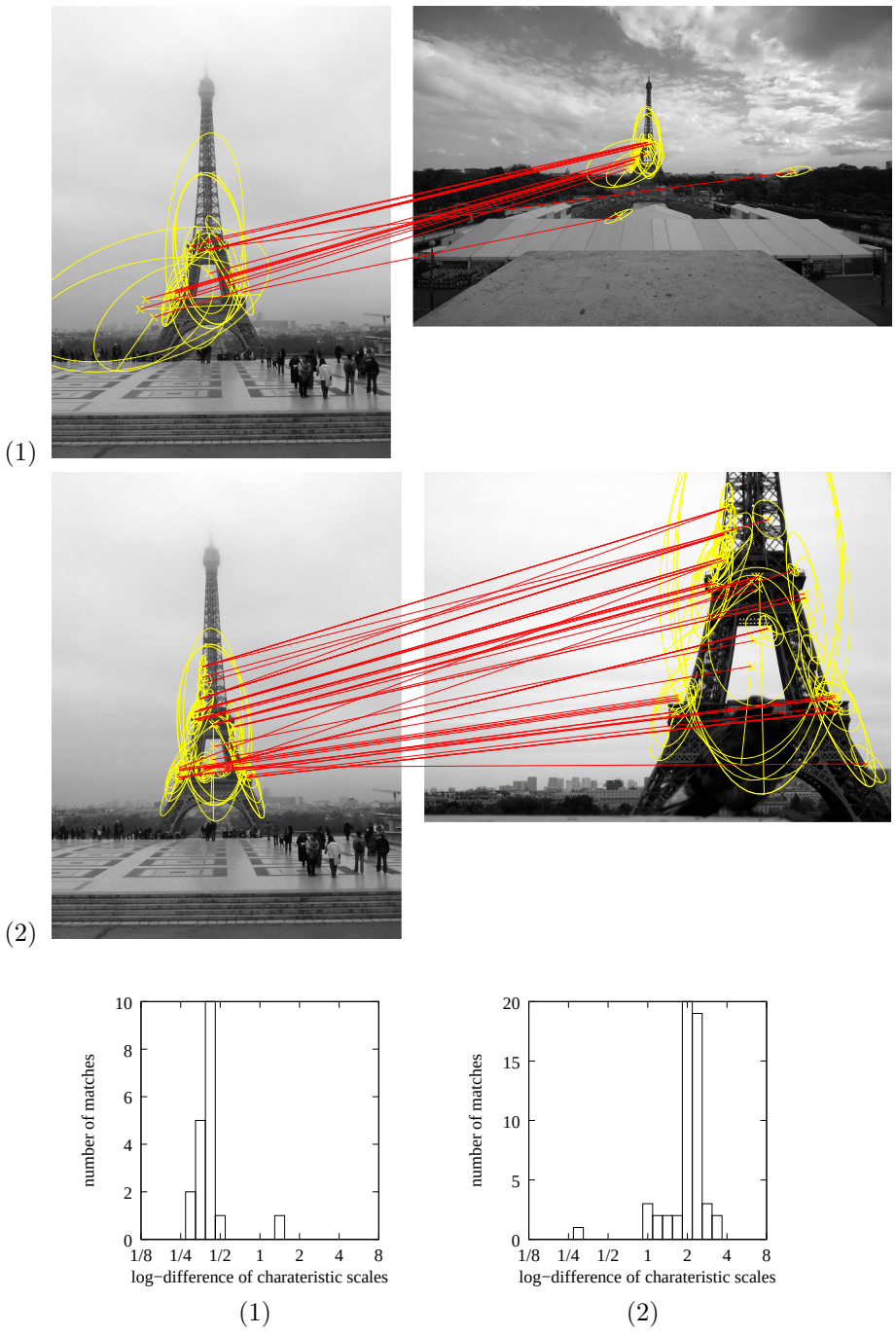


Figure 9: Scale consistency for two pairs of matching images. *Top two rows:* The matched interest point regions. *Bottom:* The corresponding histograms of log-scale differences between the characteristic scales of matched points. The peak clearly corresponds to the scale change between images.

4.2 Weak geometrical consistency

The key idea of our method is to verify the consistency of the angle and scale measures 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, computed from the characteristic angle and scale 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 query. The update step of (1) is modified:

$$s_j(\delta_a, \delta_s) := s_j(\delta_a, \delta_s) + f(x_{i,j}, y_{i'}), \tag{13}$$

where δ_a and δ_s are the quantized angle and log-scale differences between the interest regions. The image score becomes

$$s_j^* = g\left(\max_{(\delta_a,\delta_s)} s_j(\delta_a, \delta_s)\right). \tag{14}$$

The motivation behind the scores of (14) is to use angle and scale information to reduce the scores of images whose 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 δ_a and δ_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 (13). In this case two histograms s_j^a and s_j^s are separately updated by

$$\begin{array}{l} s_j^a(\delta_a) := s_j^a(\delta_a) + f(x_{i,j}, y_{i'}), \\ s_j^s(\delta_s) := s_j^s(\delta_s) + f(x_{i,j}, y_{i'}). \end{array} \tag{15}$$

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

$$s_j^* = g\left(\min\left(\max_{\delta_a} s_j^a(\delta_a),\ \max_{\delta_s} s_j^s(\delta_s)\right)\right) \tag{16}$$

is a reasonable estimate of the maximum of (14). This approximation will be used in the following. It significantly reduces the memory requirements. In practice, the histograms are smoothed by a moving average to reduce the angle and log-scale quantization artifacts. Note that a translation model could theoretically be included in WGC. However, for a large number of images, the number of parameters should be kept below 2, otherwise the memory and CPU costs of obtaining the scores would not be tractable.

4.3 Injecting a priori knowledge

Fig. 10(a) shows that the repartition of the angle difference δ_a between matched descriptors is different for matching and non-matching image pairs. As a matching image pair also includes incorrectly matched points, this suggests that the

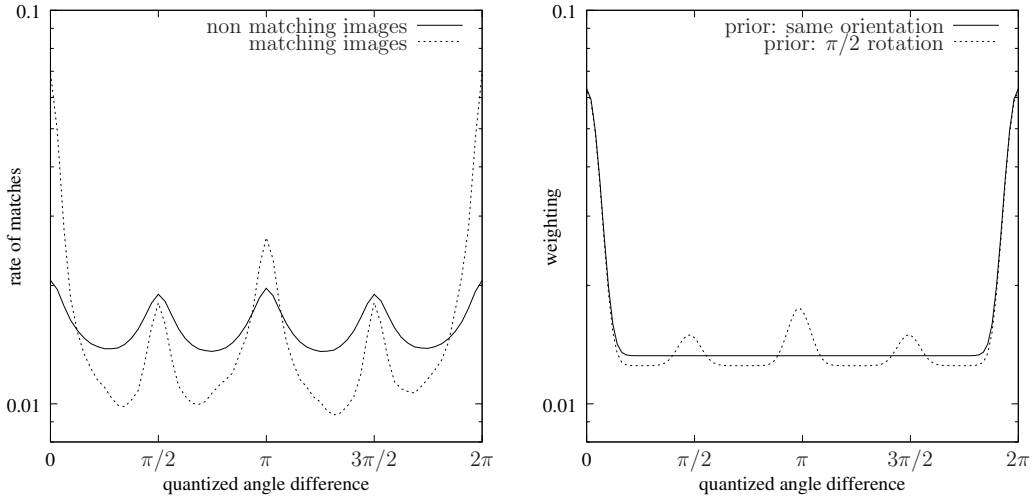


Figure 10: (a): histogram of δ_a values accumulated over all query images of the Holidays dataset; (b): weighting function applied in the g_j computation.

probability mass function of angle differences for truely matching points follows a highly non-uniform repartition. This is due to the higher frequency of horizontal and vertical gradients in photos and to the human tendency to shoot either in "portrait" or "landscape" mode. A similar bias is observed for δ_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 "$\pi/2$ rotation" for more general bases, see Fig. 10(b).

5 Descriptor quantization

5.1 Codebook construction and complexity

In contrast with the hierarchical method of [5] and to the approximate clustering of [9], we use an exact brute-force k-means algorithm to generate the visual vocabulary. This is computationally expensive, but as this step is performed *offline*, it has no impact at search time. Compared to [5], an exact k-means algorithm generates more balanced clusters, i.e. the lengths of the lists in the inverted file are of the same order. This results in a better efficiency when querying the inverted file, as the expected computing cost C associated with a single query descriptor is

$$C = n_d \sum_{i=1}^{k} p_i^2, \tag{17}$$

where n_d is the number of descriptors stored in the inverted file, and p_i denotes the probability that a given SIFT descriptor is assigned to the i^{th} visual word.

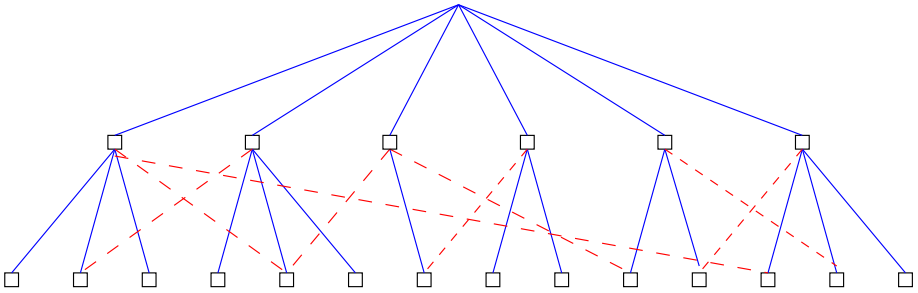


Figure 11: The two-layer graph structure used for approximate assignment of visual words. *Plain:* The original connections between the two layers, as generated by the clustering. *Dashed:* Additional connections learned on an independent dataset.

The minimum of C is obtained when $p_i = 1/k$ for all visual words, i.e., when the inverted lists are of equal length. In that case, for a single input descriptor, the expected number of entries analyzed is equal to n/k.

We can define *the unbalance factor* of a quantizer and a vocabulary as

$$u_f = k \sum_{i=1}^{k} p_i^2, \tag{18}$$

which is ratio of the number of entries analyzed over the one analyzed for a minimal C. In other terms, for a fixed vocabulary size k, the unbalance factor is a measure of the query cost associated with a given visual word distribution. This quantity was empirically measured in [11]. Using k-means, the unbalance factors obtained for vocabulary sizes of $k = 20000$ and $k = 200000$ are equal to 1.21 and 1.34, respectively. Hierarchical clustering such as proposed in [5, 11] leads to much higher values: [11] reports factors between 4 and 5 for hierarchical clustering.

5.2 Approximate visual word assignment

In order to efficiently assign descriptors to visual words when using large vocabularies (e.g., $k = 200000$), we use an approximate assignment scheme. It relies on a layered graph structure (Fig. 11) constructed as follows:

Clustering: We compute a full k-means clustering on the training set to obtain a vocabulary of size k. The resulting centroids are the second-layer nodes of our structure.

Tree construction: k-means clustering is performed on the visual words, producing k' centroids. These centroids form the first layer of our hierarchical structure. Each visual word of the original codebook is a leaf in the second layer, and is connected with its closest centroid in the first level.

Compared to the *top-down* approach of [5], the *bottom-up* construction of the tree is clearly more costly, as it requires to perform k-means clustering for a large vocabulary. However, this construction is performed *off-line*. Its efficiency is therefore not a critical point. The motivation for using standard

centroids as tree leaves is to preserve the k-means Voronoï cells, which minimize the reconstruction error between a descriptor and its visual word.

Graph construction: At this point, the tree structure can already be used to assign descriptors to visual words as in [5]. However, the nearest centroid of a descriptor in the second layer may not be connected to the one in the first layer. In that case, assigning a descriptor using the tree structure leads to find a visual word which is not the closest from the descriptor considered. The greedy N-best paths search strategy proposed in [20] addresses this issue by keeping several nodes in each level and by exploring their children, which may be of interest when using a large number of layers and a small branching factor.

Here, we propose to complete the graph structure by connecting the first-layer nodes and the leaves that may be the nearest neighbours of the same descriptor. The connections are learned on a large dataset, by quantizing each descriptor with both quantizers, and by connecting the nodes of the resulting visual words. Using $k' = 20000$ and $k = 200000$, on average an internal node is connected with 552 leaves, against $k/k' = 10$ leaves for the original tree structure.

To assign a descriptor to a visual word, we first search the nearest neighbor in the first layer. We then search the nearest neighbor in the leaves connected to it. The advantage of this method is that the structure is computed *off-line* and takes into account the statistic of the data. If the training set used for learning the connections is large enough, only connections that occur with low probability are missed.

5.3 Multiple assignment

At query time, instead of choosing only the nearest neighbor, each descriptor is assigned to *several* nearest visual words. This strategy is similar to the multiple descriptor assignment proposed in [10] or the soft quantization method proposed in [13]. The method we propose hereafter is slightly different from [10] in that

- we perform multiple assignment for the query only, not for the images of the datasets: memory usage is unchanged.
- the distance d_0 to the nearest centroid is used to filter out the selected centroids whose distances with the descriptor are higher than αd_0 (typically, $\alpha = 1.2$). This criterion reduces the number of cells to explore as well as the noise.

Under this criterion, the average number of assignments we keep, out of 10 nearest neighbours, is 4. Hence, the query time is approximately multiplied by this value. This loss of efficiency is compensated by a significantly better accuracy, a shown in the experimental section 7: the best mean average precision measured for our image search system rises from 0.763 to 0.810 on the Holidays dataset.

6 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.

Table 1: Inverted file memory usage

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

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 accumulating 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. In our experiments, the overall memory usage was not noticeably changed by this implementation, which 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 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 most of the entries, as shown in Fig. 5.

WGC impact on the complexity: WGC modifies the score update by applying (15) 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, we use the approximate nearest neighbor search (section 5), i.e., the assignment is not ten times slower than for $k = 20000$, based on exhaustive search.

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

	$k = 20000$	$k = 200000$
compute descriptors	0.88 s	
quantization + binary signature	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

Table 3: Datasets used in our experiments.

Dataset	#images	#queries	#descriptors
Holidays	1,491	500	4,455,091
Oxford5k	5,062	55	4,977,153
Flickr60k	67,714	N/A	140,211,550
Flickr1M	1,000,000	N/A	2,072,739,475

7 Experiments

We perform our experiments on two annotated datasets: our own *Holidays* dataset, see Fig. 13, and the Oxford5k dataset. To evaluate large scale image search we also introduce a distractor dataset downloaded from Flickr. For evaluation we use mean average precision (mAP) [9], 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. Descriptors are obtained by the Hessian-Affine detector and the SIFT descriptor, using the software of [17] with the default parameters. Clustering is performed with k-means on the independent Flickr60k dataset. The number of clusters is specified for each experiment.

7.1 Datasets

In the following we present the different datasets used in our experiments, see Table 3 for an overview.

Holidays. We have collected a new dataset which 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. The first image of each group is the query image and the correct retrieval results are the other images of the group. The dataset is available at [16].

Oxford5k. We also used the Oxford dataset first used in [9]. The images represent Oxford buildings. All the dataset images are in "upright" orientation because they are displayed on the web.

Table 4: Results for *Holidays* and *Oxford* datasets as in [21] (no Hamming distance weighting, no multiple assignment). 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		prior	0.6446	0.6859	0.4375	0.4602
HE+WGC	20	prior	0.7391	0.7328	0.5442	0.5096
HE+WGC	22	prior	0.7463	0.7382	0.5472	0.5217
HE+WGC	24	prior	0.7507	0.7439	0.5397	0.5252
HE+WGC	26	prior	0.7383	0.7404	0.5253	0.5275

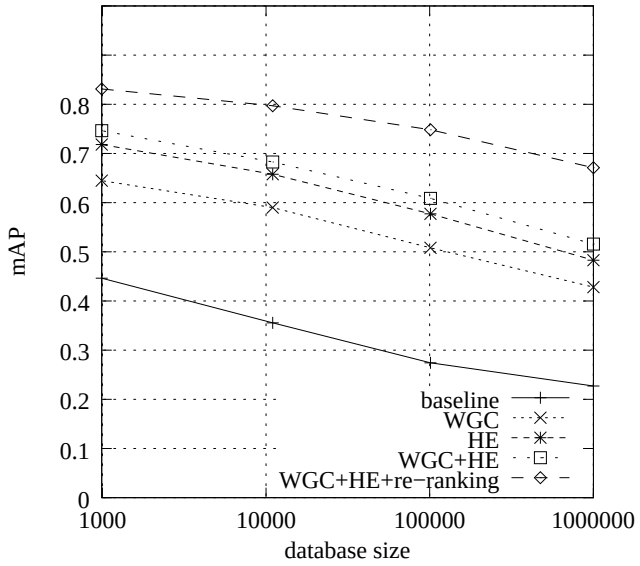
Flickr60k and **Flickr1M**. We have retrieved arbitrary images from Flickr and built two distinct sets: Flickr60k is used to learn the quantization centroids and the HE parameters (median values). For these tasks we have used respectively 5M and 140M descriptors. Flickr1M are distractor images for large scale image search. Compared to *Holidays*, the Flickr datasets are slightly biased, because they include low-resolution images and more photos of humans.

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 [9], 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 [9], see first row of Table 4.

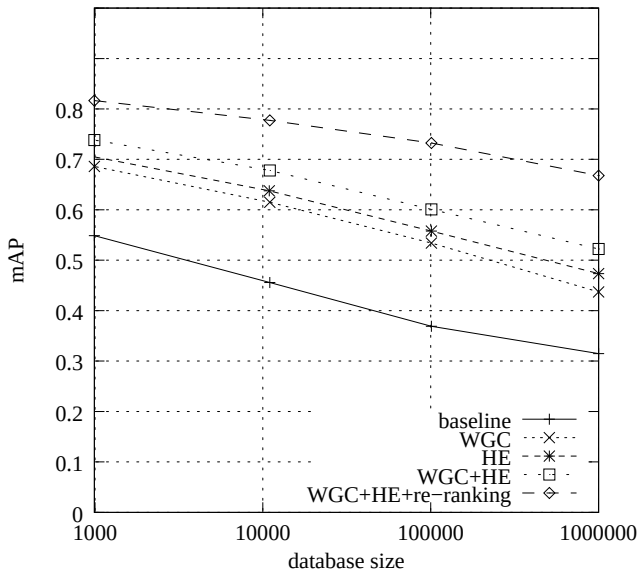
7.2 Evaluation of HE and WGC

INRIA Holidays and Oxford building datasets: Table 4 compares the proposed methods with the standard BOF baseline. We can observe that both HE and WGC result in significant improvements. Most importantly, the combination of the two further increases the performance. Note that these results have been are obtained without spatial verification nor query expansion.

Table 4 show the improvement in terms of mAP when using our Hamming distance weights of subsection 3.3 and the multiple assignment of subsection 5.3. One can see that both methods significantly increase the accuracy. Note that our best result on the Oxford Building dataset is 0.610, which is much better than the state-of-the-art [13] of 0.493 in a comparable setup, i.e., without spatial



$k = 20000$



$k = 200000$

Figure 12: 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*.

Table 5: Results for *Holidays* and *Oxford* datasets: impact of Hamming distance weighting and multiple assignment (MA). The other parameters are the same as in Table 4.

	Parameters		Holidays		Oxford	
	HE: h_t	WGC	$k = 20000$	$k = 200000$	$k = 20000$	$k = 200000$
HE	24		0.7485	0.7399	0.5201	0.5022
WGC		prior	0.6645	0.6769	0.4620	0.4687
HE+WGC	24	prior	0.7630	0.7584	0.5602	0.5389
MA+HE	24		0.7545	0.7779	0.5600	0.5635
MA+HE+WGC	24	prior	0.8105	0.7997	0.6007	0.6101

Table 6: *Holidays dataset + Flickr1M:* Rate of true positives as a function of the dataset size for a shortlist of 100 images, $k = 200000$.

dataset size	1490	11490	101490	1001490
BOF	0.673	0.557	0.431	0.306
WGC+HE	0.855	0.789	0.708	0.618

re-ranking and query expansion (which require to produce a correct short-list in the first place).

Large scale experiments: Fig. 12 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. 13. The third and fourth rows show that some images from the *Flickr1M* dataset artificially decrease the results in terms of mAP given in Fig. 12, as *false* false positive, marked by FP(?), are some images which are actually relevant to the query image. We can observe in the two first rows that HE and WGC improve the quality of the ranking significantly for these queries. Here again, some false false positives (not displayed here) are interleaved with the correct returned images.

Table 6 shows 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 estimation of an affine transformation with our implementation of [1]. Fig. 12 also shows the results obtained with a shortlist of 100 images. We can observe further improvement, which confirms that it is complementary with WGC.

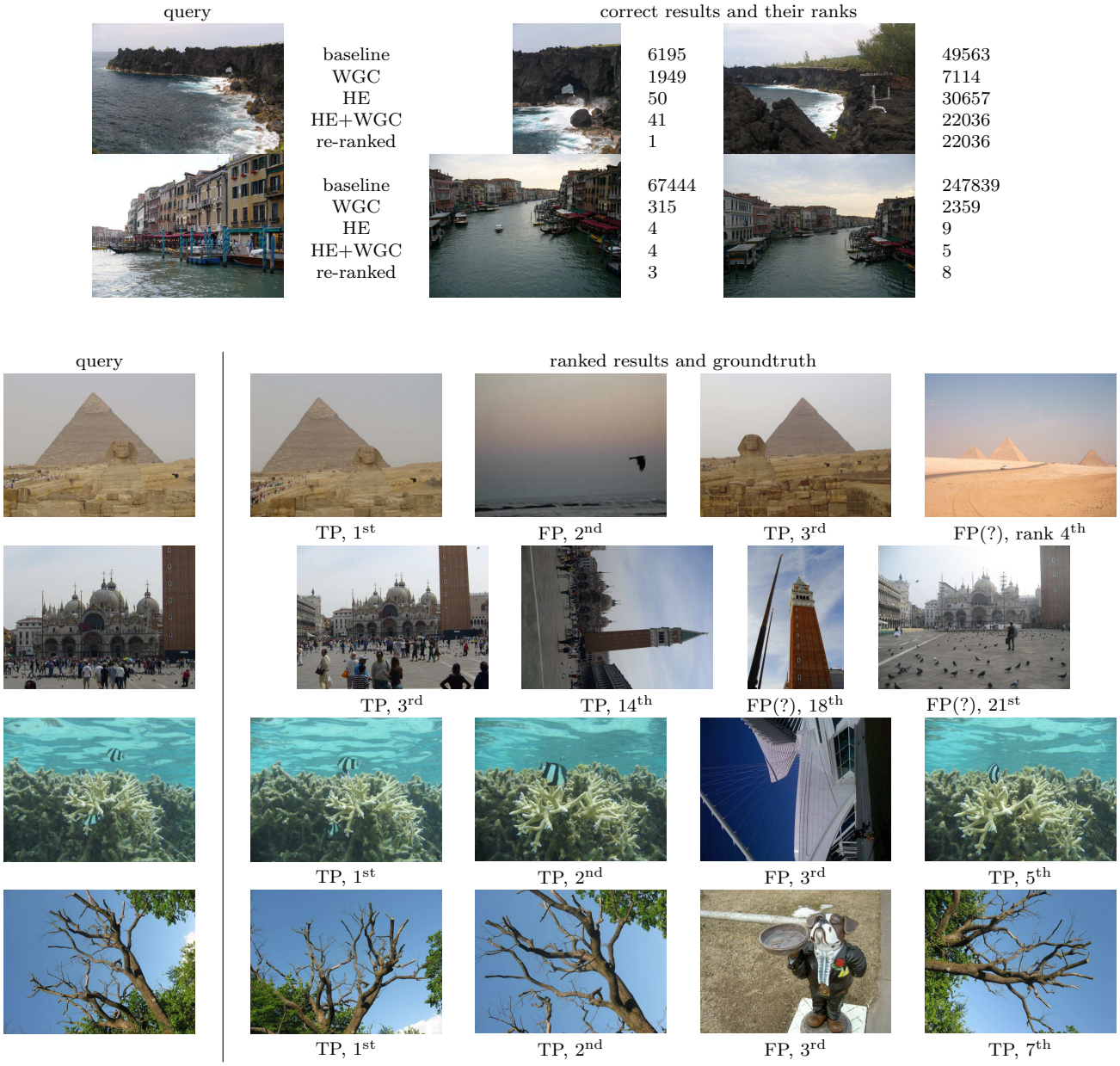


Figure 13: Queries from the *Holidays* dataset and some corresponding results for *Holidays*+1M distractors from Flickr1M. Rows 1 and 2: how the different methods rank the true matches. Below: example results labeled as true positives (TP) or false positives (FP). Note that the displayed images are interleaved with TPs and FPs. As the *Holidays* dataset includes pictures of popular tourist attractions, casual matches were found in the distractor dataset. They count as false positives and are marked with FP(?).

8 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, RAFFUT and QUAERO as well as GRAVIT for their financial support.

References

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

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

[3] Matas, J., Chum, O., Martin, U., Pajdla, T.: Robust wide baseline stereo from maximally stable extremal regions. In: BMVC. (2002) 384–393

[4] Sivic, J., Zisserman, A.: Video Google: A text retrieval approach to object matching in videos. In: ICCV. (2003) 1470–1477

[5] Nistér, D., Stewénius, H.: Scalable recognition with a vocabulary tree. In: CVPR. (2006) 2161–2168

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

[7] Datar, M., Immorlica, N., Indyk, P., Mirrokni, V.: Locality-sensitive hashing scheme based on p-stable distributions. In: Proceedings of the Symposium on Computational Geometry. (2004) 253–262

[8] Shakhnarovich, G., Darrell, T., Indyk, P.: 3. In: Nearest-Neighbor Methods in Learning and Vision: Theory and Practice. MIT Press (2006)

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

[10] Jegou, H., Harzallah, H., Schmid, C.: A contextual dissimilarity measure for accurate and efficient image search. In: CVPR. (2007)

[11] Fraundorfer, F., Stewénius, H., Nistér, D.: A binning scheme for fast hard drive based image search. In: CVPR. (2007)

[12] Schölkopf, B., Smola, A.: Learning with Kernels: Support Vector Machines, Regularization, Optimization and Beyond. MIT Press, Cambridge, MA (2002)

[13] 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)

[14] Torralba, A., Fergus, R., Weiss, Y.: Small codes and large databases for recognition. In: CVPR. (2008)

[15] Oliva, A., Torralba, A.: Modeling the shape of the scene: a holistic representation of the spatial envelope. IJCV **42** (2001) 145–175

[16] Jegou, H., Douze, M.: INRIA Holidays dataset. `http://lear.inrialpes.fr/people/jegou/data.php` (2008)

[17] Mikolajczyk, K.: Binaries for affine covariant region descriptors. `http://www.robots.ox.ac.uk/~vgg/research/affine/` (2007)

[18] MacKay, D.: 2. In: Information Theory Inference and Learning Algorithms. Cambridge University Press (2006)

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

[20] Schindler, G., Brown, M., Szeliski, R.: City-scale location recognition. In: CVPR. (2007)

[21] Jegou, H., Douze, M., Schmid, C.: Hamming embedding and weak geometric consistency for large scale image search. In: ECCV. (2008)

Centre de recherche INRIA Grenoble – Rhône-Alpes
655, avenue de l'Europe - 38334 Montbonnot Saint-Ismier (France)

Centre de recherche INRIA Bordeaux – Sud Ouest : Domaine Universitaire - 351, cours de la Libération - 33405 Talence Cedex
Centre de recherche INRIA Lille – Nord Europe : Parc Scientifique de la Haute Borne - 40, avenue Halley - 59650 Villeneuve d'Ascq
Centre de recherche INRIA Nancy – Grand Est : LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex
Centre de recherche INRIA Paris – Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex
Centre de recherche INRIA Rennes – Bretagne Atlantique : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex
Centre de recherche INRIA Saclay – Île-de-France : Parc Orsay Université - ZAC des Vignes : 4, rue Jacques Monod - 91893 Orsay Cedex
Centre de recherche INRIA Sophia Antipolis – Méditerranée : 2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399