Scalable Mining of Small Visual Objects (with new experiments)

Pierre Letessier, Olivier Buisson, Alexis Joly

Scalable Mining of Small Visual Objects (with new experiments)

Pierre Letessier
INA & INRIA Sophia-Antipolis
4 av de l'Europe
94360 Bry sur Marne, France
pletessier@ina.fr

Olivier Buisson
INA
4 av de l'Europe
94360 Bry sur Marne, France
obuisson@ina.fr

Alexis Joly
INRIA Sophia-Antipolis
161 rue Ada
34095 Montpellier, France
alexis.joly@inria.fr

ABSTRACT

This paper presents a scalable method for automatically discovering frequent visual objects in large image collections even if their size is very small. It extends the work initially published in [12] with additional experiments comparing the proposed method to the popular Geometric Min-hashing method. The basic idea of our approach is that the collision frequencies obtained with hashing-based methods can actually be converted into a prior probability density function given as input to a weighted adaptive sampling algorithm. This allows for an evaluation of any hashing scheme effectiveness in a more generalized way, and a comparison with other priors. In this work, we introduce a new hashing strategy, working first at the visual level, and then at the geometric level. It allows integrating weak geometric constraints into the hashing phase and not only neighborhood constraints as in previous works. Experiments show that this strategy boosts the performances considerably and clearly outperforms the state-of-the-art Geometric Min-Hashing method.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: Search process

Keywords

Computer Vision, Mining, Discovery, Scalable, Small Objects, Weak geometry, Hashing, LSH, RMMH

Résumé

Cet article présente une méthode scalable de découverte d'objets visuels fréquents dans de grandes collections d'image. Il s'agit d'une extension d'un travail publié initialement dans [12] et qui montre que la complexité de ce probème est principalement lié à la taille des objets relativement à la taille des images (logos, petits objets manufacturés, etc.). Les nouvelles expérimentations présentées ici en plus du contenu d'origine concerne la comparaison avec une méthode populaire de l'état de l'art, appelée Geometric Min-hashing. Le principe de notre approche étant d'utiliser les fréquences de collision dans des tables de hachage pour initier un échantillonnage adaptatif, il en effet possible de la généraliser à tout type de hachage et ainsi de les comparer entre eux. Les nouveaux résultats montrent d'une part les limitations de Geometric Min-hashing pour découvrir des objets de petite taille et d'autre part l'apport important du hachage géométrique faible que nous avons introduits.

1. INTRODUCTION

Automatically linking multimedia documents that contain one or several instances of the same visual object is gaining more and more interest. Visual features and rigid objects matching techniques are actually mature enough to produce accurate visual links, trustable from a user's point of view. Automatically discovering such objects and links in a given multimedia collection or across different media has many applications including: salient media events detection, content-based filtering recommendation, web browsing, etc.

Whereas efficient methods now exist that aim to searching for rigid objects in large collections, discovering them from scratch is still challenging in terms of scalability, particularly when the targeted objects are rather small compared to the whole visual content. It is for instance noticeable that most previous works on object discovery were evaluated on the popular Oxford buildings dataset, where the targeted objects (buildings) occupy a very large fraction of the images themselves.

One of the claims of this paper is that the complexity of mining repeated visual objects is closely related to the relative size of the targeted objects and to their frequency of occurrence. Therefore, the complexity of classical mining algorithms for discovering repeated item sets is known to be highly related to their frequency. To illustrate the variety of problems, let us compare the objects considered in the Oxford buildings dataset to those considered in the BelgaLogos dataset, another popular evaluation dataset focused on trademark logos. Figure 1 shows the repartition of the instance sizes for these two datasets. The measure used here to analyze the sizes of instances is the number of SIFT features falling into the bounding boxes of the objects in the ground truth. We found 29,968,910 features in Oxford buildings, with 6,056,353 belonging to objects in the ground truth, and 38,093,296 descriptors in BelgaLogos, with 184,698 belonging to the annotated logos. We observe that the coverage of objects in Oxford buildings is around

20% while it is only 0.5% in BelgaLogos.

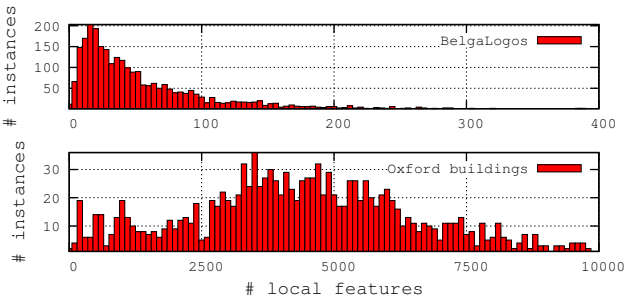

Figure 1: Comparison of instances size in BelgaLogos and Oxford buildings

2. RELATED WORKS

The scalable mining of visual content database is recently emerging [2, 16, 11, 20, 19, 21]. Most of these methods are dedicated and tested to the discovery of large objects (more than 20% of the images) in image collections [2, 16, 19, 21]. There are very few papers about small visual objects discovery and mining. In [2] is presented a qualitative evaluation of small visual objects mining in large scale dataset. Consistent Visual Words Mining with Adaptive Sampling [11] is a rare work where the technology is dedicated to discover and mine small visual objects and where a quantitative evaluation is presented.

This method [11] can be summarized by four main successive steps:

Image description: The images are described by Sift features [14], one of the most widely used.

Features Indexing A Posteriori Multi-Probe Locality Sensitive Hashing (APMP-LSH) method proposed by Joly et al. [7] that allows sublinear search time with reduced memory space costs. This method approximates the nearest neighbor search of similar Sift features.

Seeds Generation: Seeds are Region Of Interest where there are potential instances of repeated objects. To generate the seeds, they propose a generic framework based on an adaptive weighted sampling scheme. This sampling process starts with some prior distribution and iteratively converges to unvisited ROI. Different natures of prior distribution allow them to define different types of ROI. For examples, they propose a prior distribution based on a face detector in order to discover ROI containing faces. They also proposed another prior to avoid textured areas (to suppress false alarms). In [11], the priors only exploit intra-image saliency information.

Seeds Verification and Retrieval: In [11], to suppress the ROI containing false alarms where there are no object instances, they directly propose to launch a process of local region search. In fact, the selected seeds or ROI are queries of this process. The local region search is based on the framework proposed in [8] to retrieve small objects: APMP-LSH method [7] for Sift similarity search, match verification (RANSAC [5]) to suppress false matches and an A Contrario Query Expansion (QE) [8] to improve the recall of small object instances. An equivalent strategy of match verification and QE is also used to discover small objects in [2].

The framework, proposed in [11], is then flexible and generic, because it could be possible to define new types of priors to discover different objects or to change the discovering strategy. The main computational cost of this framework is the last step: Seeds Verification and Retrieval. These costs could be reduced if the adaptive sampling would more quickly focus on the ROI where the instances of repeated objects are. It could be faster if the priors used by the adaptive sampling were more efficient to discover small visual objects.

In this paper, we propose a formulation to accurately analyze these costs and a new type of priors to drastically decrease the computational costs. We then integrate these priors into the framework proposed by [11]. As discussed below, our priors are based on Weak Geometry (WG) and hashing methods.

The evaluation dataset BelgaLogos used in [11], and proposed in [8], is designed to evaluate methods to retrieve small visual objects' instances. This dataset's ground truth is not fully adapted to evaluate a mining process. Apart from the queries' ground truth, in this image dataset, there are a lot of visual objects' instances which are not annotated (buildings, faces, ...). These instances could be considered as false alarms by the evaluation process. To our knowledge, there doesn't exist specific dataset which is designed to evaluate small visual object discovery and mining methods! We therefore propose an adapted dataset and ground truth to evaluate theses two tasks.

3. PROBLEM FORMULATION

3.1 Notations

Let us first introduce some notations. We consider a dataset $\mathbf{I}$ of N_I images described by a set $\mathbf{X}$ of N local feature vectors $\mathbf{x}_i$, each being extracted at position $\mathbf{p}_i = (I_i, \chi_i, \psi_i)$ where I_i is the identifier of the image and (χ_i, ψ_i) the coordinates of the local feature in the image.
Now, we consider a set $\mathbf{O}$ of objects O^m, each being represented by S_m instances O_s^m. An instance O_s^m is associated with a unique area A_s^m (in a single image) and contains a set of local features:

$$\mathbf{X}_s^m = \{\mathbf{x}_i \mid \mathbf{p}_i \in A_s^m\}_{1 \leq i \leq N}$$

We will now introduce some basic definitions that will help us to formalize our object's discovery and mining problems, and model our approach.

Definition - Global cover
The global cover $c_{\mathbf{X}}(O^m)$ of an object O^m is defined by:

$$c_{\mathbf{X}}(O^m) = \frac{1}{N} \sum_{s=1}^{S_m} |\mathbf{X}_s^m| \quad (1)$$

It measures the fraction of features in the dataset covered by all instances of a given object.

Definition - Average cover
The average cover $c(O^m)$ of an object O^m is defined by:

$$c(O^m) = \frac{1}{S_m} \sum_{s=1}^{S_m} \frac{1}{N_{I_s}} |\mathbf{X}_s^m| \quad (2)$$

with N_{I_s} the number of features in the image including O_s^m.

It measures the average fraction that an instance of the object occupies in an image.

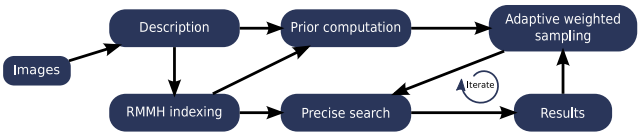

Figure 2: Object mining framework

Definition - Frequency
The *frequency* $f(O^m)$ of an object O^m is defined by:

$$f(O^m) = \frac{S_m}{N_I} \tag{3}$$

Definition - $\{c, f\}$-frequent object
An object O^m is said $\{c, f\}$-*frequent* if:

$$\begin{cases} c(O^m) = c \\ f(O^m) = f \end{cases}$$

We also notice here that if the number of features per image is rather stable then we can consider that $c_{\mathbf{X}} \approx c.f$

3.2 Problems

We define the two following problems as the main objectives to be solved by objects mining and discovery methods.

Objects Discovery: find at least one instance O_q^m of all $\{c, f\}$-*frequent* objects O^m such that:

$$\begin{cases} c \geq c_0 \\ f \geq f_0 \end{cases}$$

Objects Mining: find all instances O_s^m of all $\{c, f\}$-*frequent* objects O^m such that:

$$\begin{cases} c \geq c_0 \\ f \geq f_0 \end{cases}$$

The underlying idea is that the complexity of these two problems depends mainly on the parameters c and f, so we suggest including them in the query formulation of the problem as done in classical data mining approaches (e.g. for mining frequent item sets).

4. OBJECT MINING FRAMEWORK AND MODELING

We use the same object mining framework as the one proposed in [11]. The main principle of this approach is to use a *weighted* and *adaptive* sampling strategy to select the most relevant query regions to be issued afterwards to a precise object search algorithm (e.g. based on large scale matching and RANSAC registration). To avoid querying all possible regions of interest while keeping a good coverage of the contents, *Sampling* is indeed a simple yet efficient statistical paradigm allowing to yield some knowledge about a population without surveying it entirely. Adaptive weighted sampling is a more advanced paradigm allowing to iteratively update the sampling distribution according to the results obtained during previous iterations. This allows the mining process to progressively focus on unvisited image regions and consequently reduce the number of required probes for achieving a good completeness of the mining.

Our global object mining framework is presented in Figure 2. It is composed of three main steps:

SIFT features extraction: All images in $\mathbf{I}$ are described with SIFT features [14], resulting in a set $\mathbf{X}$ of N 128-dimensional feature vectors $\mathbf{x}_i$. Each extracted feature $\mathbf{x}_i$ is associated to a position $\mathbf{p}_i = (I_i, \chi_i, \psi_i)$, a principal orientation θ_i and a local scale σ_i.

Prior distribution computation: The aim of this step is to compute a prior probability mass function p_0 on $\mathbf{X}$, where $p_0(\mathbf{x}_i)$ is the probability of a given $\mathbf{x}_i$ to be sampled at the first iteration of the adaptive weighted sampling step. Since building p_0 is the core contribution of our work, we will come back to this issue in section 5.

Adaptive Weighted Sampling and Precise Search: This step works in the same way as in [11], so we will let the reader refers to this work for more details. The algorithm iteratively samples a candidate feature $\mathbf{x}_t$ $(0 \leq t < T)$ according to the probability mass function p_t on $\mathbf{X}$. A query window centered around $\mathbf{x}_t$ is then issued to a *precise search* algorithm allowing it to find other instances of the object captured by the query window (if any exist in the dataset). The probability mass of the features belonging to the retrieved instances are then decreased, resulting in a new probability mass function p_{t+1} to be used in the next sampling iteration.

As the prior distribution p_0 is the initial condition of the Adaptive Weighted Sampling algorithm, the way it is built has a strong impact on the whole performances of the mining process, i.e. on the number T of iterations required to discover instances of (c, f)-frequent objects. As we will see in the rest of the paper, building an accurate prior distribution can divide by up to 32 the number of required probes compared to a uniform sampling.
Admittedly, the speed and the effectiveness of the *precise search* algorithm also have an influence on the performance of the method, but in a more limited way. The overall complexity of the approach actually remains $O(T)$ whatever the speed of the search algorithm used, e.g. [16, 6, 1, 8, 9] (because one single search remains an expensive process). Regarding the effectiveness of the precise search algorithm, things are more complex. The quality of the mining after T iterations actually depends on the recall and precision of the search strategy, in a way that is difficult to model accurately. Starting with a perfect search algorithm however gives us some insights about what happens overall.
Let us actually consider a dataset with one single $\{c, f\}$-*frequent* object O^m. Now let $\mathbf{x}_t$ be the candidate query feature selected at random from $\mathbf{X}$ according to the probability mass function p_t. While $\mathbf{x}_t$ does not belong to an instance O_s^m of O^m, p_t remains equal to p_0 if we consider a precise search algorithm with a perfect precision. On the other side, if $\mathbf{x}_t$ belongs to an instance O_s^m of O^m, the full mining problem is completed (when considering a precise search algorithm with perfect recall). So that the number of required iterations T follows a negative binomial distribution $BR(1, 1-c_0(O^m))$ where the failure probability $c_0(O^m)$ is the probability that a sampled feature belongs to an instance O_s^m of O^m, i.e.

$$c_0(O^m) = \sum_{\mathbf{x}_i \in \{O_s^m\}} p_0(\mathbf{x}_i)$$

The expected number of probes $\hat{T}$ is therefore equal to

$$\hat{T} = \frac{1 - c_0(O^m)}{c_0(O^m)} = \frac{1}{c_0(O^m)} - 1 \quad (4)$$

and the complexity of the algorithm is $O(\frac{1}{c_0(O^m)})$. We generally refer to $c_0(O^m)$ as the prior statistical cover of the object O^m since it measures the fraction of the prior distribution captured by the object. This shows that the complexity of the algorithm is strongly related to the prior distribution p_0. If we consider for instance a perfect prior distribution, so that $p_0(\mathbf{x}_i) = 0$, $\forall \mathbf{x}_i \notin \{X_s^m\}$ and $p_0(\mathbf{x}_i) = 1/N$, $\forall \mathbf{x}_i \in \{X_s^m\}$, then we have $c_0(O^m) = 1$ and the complexity of the algorithm is $O(1)$.
On the other side, if we consider an uniform prior distribution so that $p_0(\mathbf{x}_i) = 1/N$, $\forall \mathbf{x}_i \in \mathbf{X}$ then the complexity is $O(c_{\mathbf{X}}(O^m))$ where $c_{\mathbf{X}}(O^m)$ is the global cover of the object as defined in section 3.1. And since $c_{\mathbf{X}}(O^m) \approx c(O^m).f(O^m)$ the complexity of mining a (c, f)-frequent object with an uniform sampling is roughly $O(\frac{1}{cf})$. This illustrates well why mining small objects (with small values of c) might be a much more difficult task than objects occupying a large fraction of the image ($c \approx 1$).

If we now consider a real precise search algorithm rather than a perfect one, so that it returns only a fraction r of the instances of an object O^m when a query feature $\mathbf{x}_t$ belonging to one of the instance is sampled. Then, good recall will be achieved only if a sufficient number μ of instances of the same object are selected by the sampling. Under the hypothesis that the retrieved instances are independent from one query to another, the probability that an instance is missed after μ queries is equal to $(1 - r)^\mu$. If we want this probability to be lower than a probability ϵ, then we get $\mu > \log(\epsilon)/\log(1 - r)$. In that case, the number of required iterations T would follow a negative binomial distribution $BR(\mu, 1 - c_0(O^m))$ instead of $BR(1, 1 - c_0(O^m))$. And the expected number of probes would be

$$\hat{T} = \frac{\log(\epsilon)}{\log(1 - r)} \left(\frac{1}{c_0(O^m)} - 1 \right) \quad (5)$$

The complexity of the whole mining process is therefore still $O(\frac{1}{c_0(O^m)})$ for fixed values of r and ϵ. For relatively good recall values r and moderate variations between existing search methods, we can argue that the prior distribution p_0 has a much larger impact on the complexity than the effectiveness of the precise search algorithm used on top of the adaptive sampling algorithm.

5. HASHING-BASED PRIOR DISTRIBUTION COMPUTATION

As stated before, our objective is to compute a prior probability mass function p_0 on $\mathbf{X}$, so that $p_0(\mathbf{x}_i)$ reflects as much as possible the likelihood that a given local feature $\mathbf{x}_i$ belongs to a (c, f)-frequent object. In other words, we attempt to maximize $c_0(O^m)$ for all (c, f)-frequent objects in the dataset. Rather than using visual saliency measures computed at the image level as in [11], we propose building much more effective prior distributions based on the construction of visual and WG hash tables on the whole dataset. The cost to compute such prior is actually higher than simple image-based priors but the complexity reduction of the sampling-and-search phase still makes it widely profitable.

As done in conventional weighted random sampling methods [15], our hashing-based probability mass function p_0 is computed in practice by normalizing a scoring function z_0 on $\mathbf{X}$. The computation of all individual scores $z_0(\mathbf{x}_i)$ mainly relies on collisions frequency in hash tables making our algorithm scalable and easily distributable if needed. It includes the three following steps:

1. Construction of a hashing-based visual index (cf. 5.1)
2. Visual filtering and WG hashing (cf. 5.2)
3. Prior distribution computation (cf. 5.3)

5.1 Visual hashing

There are two main strategies for efficient visual similarity search of local features: visual vocabulary associated with inverted-list [16, 1] and the hashing [8, 9]. These two types of methods have a common aim: to reduce the computational cost of the similarity search by compressing the local features.

Most visual similarity search methods are based on visual vocabulary. Visual vocabulary allows us to describe images from a set of quantified local features. The vocabulary is usually generated by a K-means algorithm [18, 3, 16, 1] applied on the local features set of the considered corpus. In [16], the retrieval efficiency is accurately studied in functions of the vocabulary size K and the database size N (local feature number). It demonstrates that $K \approx \beta.N$ with $\beta \in [0.05, 0.10]$ to have efficient visual retrieval. Moreover, the best visual retrieval results are obtained if the visual vocabulary is learned with the current database to index. In the case of very large databases ($N > 10^8$), [16] propose distributed Approximate K-Means (AKM) which has a complexity of $O(I.N.log(K))$, with I the iteration number. For example in [16], it took 5 hours for $N = 17.0 \times 10^6, K = 10^6, I = 30$ with 32 CPU cores. Based on AKM complexity and these CPU performances, we can coarsely estimate the computational time for $N = 500 \times 10^6$, $K = 35 \times 10^6$, $\beta = 0.07$ and $I = 30$: 185 hours. This computational time is too prohibitive for a large scale process, because at this stage, the discovery and mining process has not yet started.

Due to these computational costs, we have chosen to use hashing based methods to index the visual feature. The hashing based methods have demonstrated equivalent performances to retrieval visual objects in [8, 11]. Moreover, we demonstrate in the evaluation chapter that the computational time to hash a dataset of $N = 500 \times 10^6$ Sift is lower than 4 hours with 12 CPU Core. This visual indexing is very efficient and adapted for a large scale process.

Rather than using classical LSH functions, our method is based on a more efficient method: Random Maximum Margin Hashing (RMMH, [9]), an original hash function family introduced recently. The main originality of RMMH is to **train** purely **random splits** of the data, regardless of the closeness of the training samples (i.e. without any supervision). Concretely, the method works by learning a set of *randomly trained classifiers* from a small fraction of the dataset. For each hash function, M training points are selected at random from $\mathbf{X}$ and are then **randomly labeled** (half of the points with -1 and the other half with 1). If we denote as x_j^+ the resulting $\frac{M}{2}$ positive training samples and

as $\mathbf{x}_j^-$ the $\frac{M}{2}$ negative training samples, each hash function is then computed by training a binary classifier $h(\mathbf{x})$ such that:

$$h(\mathbf{x}) = argmax_{h_\theta} \sum_{j=1}^{\frac{M}{2}} h_\theta(\mathbf{x}_j^+) - h_\theta(\mathbf{x}_j^-) \quad (6)$$

Using a linear Support Vector Machine (SVM) as a binary classifier, we get:

$$h_(\mathbf{x}) = sgn\left(\sum_{i=1}^{m} \alpha_i^* \mathbf{x}_i^*.\mathbf{x} + b_m\right) \quad (7)$$

where $\mathbf{x}_i^*$ are the m support vectors selected by the SVM ($\mathbf{x}_i^* \in \{\mathbf{x}_j^+, \mathbf{x}_j^-\}$) and it estimates α_i^* and b_m. In [9], they demonstrate that $M = 40$ is adapted for different types of features and different size of feature databases. In the following chapter, we finally denote as $\mathbf{h}_l(\mathbf{x})$ the k-length hash code produced by the concatenation of k individual binary hash functions for the l-th hash function.

5.2 Visual filtering and WG hashing

Once the visual index has been created, our algorithm computes individual scores $z_0(\mathbf{x}_i)$ by independently processing all images $I_j \in \mathbf{I}$ one by one (either iteratively or across multiple threads or machines). Let us denote I_Q the image processed at the Q-th iteration and $\mathbf{X}_Q$ the set of local features belonging to it.

5.2.1 Visual collisions filtering

To reduce the amount of collisions in the visual index for all $\mathbf{x}_q \in \mathbf{X}_Q$, we apply three complementary filtering strategies:

Intra-image collision frequency: as claimed in [2], visual features that are unique in the image are more likely to produce relevant candidate matches in the whole dataset. This paradigm actually allows the reduction of a large fraction of false positives produced by textures and other repeated visual structures. Rather than keeping only unique features, we propose a smoother strategy that bounds the collision frequency in the image to a maximum value ζ. So that a reduced set $\mathbf{X'}_Q$ of candidate features is obtained from $\mathbf{X}_Q$ through:

$$\mathbf{X'}_Q = \{\mathbf{x_q} \in \mathbf{X}_Q \mid f_Q(\mathbf{x_q}) < \zeta\} \quad (8)$$

where the intra-image collision frequency f_Q is estimated by its mean across the L visual hash tables:

$$f_Q(\mathbf{x_q}) = \frac{1}{L} \sum_{l=1}^{L} \#\{\mathbf{x} \in \mathbf{X}_Q \mid \mathbf{h}_l(\mathbf{x}) = \mathbf{h}_l(\mathbf{x}_q)\} \quad (9)$$

Empirically, ζ was set to a default value $\zeta = 3$.

Inter-tables collision frequency: As suggested by the Frequency-based LSH method [13], we then filter the set of features colliding with $\mathbf{X'}_Q$ in the whole index, according to their collision frequency across the L hash tables of the visual index. The resulting set of candidate matches can be expressed as:

$$\mathbf{Y}_Q = \left\{(\mathbf{x}_q, \mathbf{x}_i) \in (\mathbf{X}'_Q, \mathbf{X}) \mid f_L(\mathbf{x}_q, \mathbf{x}_i) > \tau\right\} \quad (10)$$

where $f_L(\mathbf{x}_q, \mathbf{x}_i)$ is the number of times $\mathbf{x}_q$ and $\mathbf{x}_i$ are hashed in the same bucket across the L hash tables. As suggested in [13], the threshold τ can be computed numerically as a function of L and α, where α is a lower bound on the collision frequency in one single hash function. This frequency-based thresholding is approximating a range query in the original feature space when LSH is used. With RMMH family, it is more likely to approximate a density-based threshold.

KNN filtering: To reduce the impact of ambiguous visual features that are present many times in the dataset (typically texts), we further reduce the number of candidate matches by a KNN filtering computed with a Hamming distance on the concatenated hash codes. So that the final candidate set of matches for image I_q is given by:

$$\mathbf{Z}_Q = \{(\mathbf{x}_q, \mathbf{x}_m) \in \mathbf{Y}_Q \mid d_H(\mathbf{h}(\mathbf{x}_q), \mathbf{h}(\mathbf{x}_m)) \leq r_H(\mathbf{x}_q, K)\} \quad (11)$$

where $r_H(\mathbf{x}_q, K)$ is the Hamming distance of the K-th nearest neighbor hash code of $\mathbf{h}(\mathbf{x}_q)$. K was empirically fixed to 200 in the experiments.

5.2.2 WG hashing

Now that we have a set $\mathbf{Z}_Q$ of visual matches for the Q-th image I_Q, we propose computing our weighting scores $z_o(\mathbf{x}_q)$ through a WG hashing approach. It is in essence similar to the WG consistency method proposed by Jegou et al. [6] that estimates the geometric consistency of a candidate's visual match based on the spatial metadata associated to each SIFT feature (scale and orientation). Our method differs in three main points. First we suggest using a sparse multi-dimensional voting space rather than voting on each mono-dimensional attribute separately. This will consistently increase the discrimination of the consistency score. Secondly, we propose to use the positions of the features additionally to the scale and orientation parameters. This allows adding a neighboring constraint to the consistency score and consequently favors small objects in detriment to large frequent objects such as backgrounds or near duplicates. Thirdly, our vote process works at the dataset scale; whereas the WG original version [6] is based on image pairs.

More concretely, for each candidate visual match $(\mathbf{x}_q, \mathbf{x}_m) \in \mathbf{Z}_Q$ we compute the following WG attributes vector:

$$\mathbf{\Delta}_{q,m} = (\Delta\theta_{q,m}, \Delta\sigma_{q,m}, \chi_q, \psi_q) \quad (12)$$

where (χ_q, ψ_q) is the position of $\mathbf{x}_q$ in image I_Q, $\Delta\sigma_{q,m} = \sigma_m - \sigma_q$ is the weak scaling factor estimation, and $\Delta\theta_{q,m}$ is the weak rotation angle estimation computed as:

$$\Delta\theta_{q,m} = \frac{\arctan(\sin(\theta_m - \theta_q), \cos(\theta_m - \theta_q))}{\pi} \quad (13)$$

To create a sparse multi-dimensional voting space from the initial WG space in which lies the vectors $\mathbf{\Delta}_{q,m}$, we propose building a LSH family adapted from the classical Euclidean LSH family [4]. Notice that the WG space is first standardized so that all attributes range in $[-1, 1]$. Standardized vectors $\mathbf{\Delta}_{q,m}$ are then hashed with L' distinct LSH functions, each composed of k' random projections of the form:

$$h'(\mathbf{\Delta}) = \left\lfloor \frac{\mathbf{a}.\mathbf{\Delta} + b}{w} \right\rfloor \quad (14)$$

The slight modification that we introduce over this classical LSH family is to have an adaptive value of w for each random projection. An L_2-sensitive hashing is actually not especially adapted for creating our voting space. We rather would like to guaranty a good and normalized dynamic on

each of the projection axis. So that we introduce the following constraints:

$$\min_{\Delta\in[-1;1]^d} \frac{\mathbf{a}.\mathbf{\Delta}+b}{w} = -2^\gamma, \qquad \max_{\Delta\in[-1;1]^d} \frac{\mathbf{a}.\mathbf{\Delta}+b}{w} = +2^\gamma \quad (15)$$

Leading to $w = \frac{\|a\|_1}{2^\gamma}$ where γ is the number of bits used to quantify each projection axis. Finally we get

$$h'(\mathbf{\Delta}) = \left\lfloor 2^\gamma \frac{\mathbf{a}.\mathbf{\Delta}+b}{\|a\|_1} \right\rfloor \quad (16)$$

and we denote as $\mathbf{h}'_l(\mathbf{\Delta})$ the k'-length hash code produced by the concatenation of k' individual hash functions for the l-th hash function.

The WG voting process now simply works by maintaining one associative container H_l^Q for each of the L' hash functions (we use a C++ *map* container in practice). Having such sparse storage for the accumulators is essential regarding the memory space required by the full mining algorithm. The *key value* used to insert the visual matches in the containers is defined as the pair

$$\mathbf{h}''_l(\mathbf{\Delta}_{q,m}) = \left(\mathbf{h}'_l(\mathbf{\Delta}_{q,m}), I_{q,m}\right) \quad (17)$$

where $\mathbf{h}'_l(\mathbf{\Delta}_{q,m})$ is the WG hash code (i.e. the estimation of the geometric transform) and $I_{q,m}$ is the identifier of the image containing the matched feature $\mathbf{x}_m$. On the other side, the *mapped value* of the associative containers is used to count the number of occurrences in the bucket identified by $\mathbf{h}''_l(\mathbf{\Delta}_{q,m})$. Note that the L' accumulators H_l^Q are empty at the beginning of the Q-th iteration of the full algorithm. All accumulators are then incremented in parallel when looping on the visual matches $(\mathbf{x}_q, \mathbf{x}_m) \in \mathbf{Z}_Q$ one by one. Each WG key value $\mathbf{h}''_l(\mathbf{\Delta}_{q,m})$ is mapped onto the accumulators H_l^Q and the associated *mapped values* are incremented according to

$$H_l^Q(\mathbf{h}''_l(\mathbf{\Delta}_{q,m})) \mathrel{+}= f_L(\mathbf{x}_q, \mathbf{x}_m) \quad (18)$$

where we recall that $f_L(\mathbf{x}_q, \mathbf{x}_m)$ is the number of collisions in the visual index (cf. Equation 10).

After all visual matches $(\mathbf{x}_q, \mathbf{x}_m) \in \mathbf{Z}_Q$ have been inserted in the accumulators, WG consistency scores $z_0(\mathbf{x}_q)$ are computed by counting the number of collisions across the L' accumulators

$$z_0(\mathbf{x}_q) = \sum_{l=1}^{L'} \sum_{(\mathbf{x}_q,\mathbf{x}_m)\in\mathbf{Z}_Q} H_l^Q(\mathbf{h}''_l(\mathbf{\Delta}_{q,m})) \quad (19)$$

So that $z_0(\mathbf{x}_q)$ measures the number of visual matches in $\mathbf{Z}_Q$ that are geometrically consistent with the direct visual matches $(\mathbf{x}_q, \mathbf{x}_m)$. The higher $z_0(\mathbf{x}_q)$, the more likely $\mathbf{x}_q$ belongs to a frequent object.

5.3 Prior distribution computation

After all images $I_Q \in \mathbf{I}$ have been processed, the conversion of WG scores z_0 to our final probability mass function p_0 is done according to

$$p_0 = \frac{z_0}{z_{\max}} . \frac{1}{p_z(z_0)} \quad (20)$$

where

$$z_{max} = \max_{\mathbf{x}_i\in\mathbf{X}} z_0(\mathbf{x}_i)$$

and $p_z(z)$ represents the probability that the WG score z_0 reaches the value z in the whole dataset:

$$p_z(z) = \frac{\#\{\mathbf{x}_i \in \mathbf{X} \mid z_0(\mathbf{x}_i) = z\}}{N} \quad (21)$$

This equally distributes the probability masses on the range of values z_0 reaches. Simply normalizing z_0 across all features would indeed have put too much probability mass on very low scores that correspond to the majority of features (that do not belong to any frequent object). This is illustrated by Figure 3 that shows the histogram of z_0 score values (i.e. $N.p_z(z_0)$) on the FlickrBelgaLogos dataset used in our experiment. Note that the number of occurrences is in log-scale.

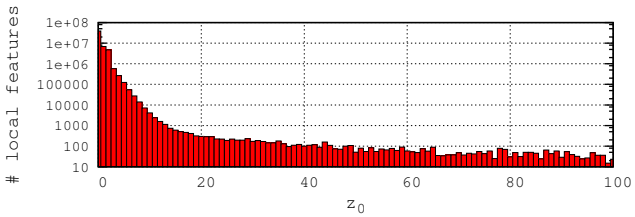

Figure 3: Histogram of WG scores z_0

6. EVALUATION DATASETS

6.1 FlickrBelgaLogos

Evaluating the accuracy of object discovery and mining algorithms is more challenging than evaluating object retrieval with a pre-fixed set of queries. We actually need a complete ground truth with all (c, f)-frequent objects of the dataset and with the precise location of all their instances. No previous evaluation dataset meeting these objectives exists, so that a contribution of this paper was to build one. We first extended the image-level ground truth of BelgaLogos dataset [8] by annotating manually the bounding boxes of all instances of the 37 targeted logos (correcting few errors along the way). The 9842 annotated instances were then visually classified as *kept* or *rejected* by 3 users, depending on whether or not they were all able to recognize the instance with confidence after it had been cropped from its image. After this step, only 2695 instances were classified as *kept*. The resulting dataset, called *BelgaLogos II* is publicly available on the Web[1].

This extended annotation is however not sufficient to evaluate the precision of object mining algorithms. Besides the 37 logos, other objects are actually instantiated several times in the dataset as well (including other logos, buildings, faces, near duplicates, etc.), so that they would be considered as false positives when detected. We therefore decided to create a new synthetic dataset by cutting and pasting the cropped logos of *BelgaLogos II* into a dataset of 10K distractor images crawled from Flickr. To reduce the probability of finding (c, f)-frequent objects in the distractors, all images come from distinct users and distinct geographic areas (1 degree of longitude and latitude). The BelgaLogos instances were then pasted without any modifications (rotation or scaling, ...) at random positions in the distractors. The resulting dataset, called *FlickrBelgaLogos* is also publicly available on

[1]http://www-sop.inria.fr/members/Alexis.Joly/BelgaLogos/BelgaLogos.html

the Web[2].

6.2 FrenchTVFrames

To evaluate the scalability of our method, we also performed some experiments on a larger dataset composed of 1,809,080 video frames extracted from 2,051 hours of 7 French television channels (TF1, France 2, Euronews, BMFTV, France 24, i>TELE, LCI), ranging from June 10 2011 to March 28 2012. Only one hour per day was kept for each channel, corresponding to the hour with the most TV news contents. We described this dataset with 549,484,617 SIFT features.

7. EXPERIMENTS

7.1 Experimental setup

Computers used in the following experiments are equipped with 2 hexa-core processors (Intel X5660). All processing times are measured by absolute time. In order to evaluate the prior performance, we define two measures: the *prior recall* and *prior precision*. The prior recall, for a given threshold η_0, is the percentage of instances in the ground truth having at least one local feature $\mathbf{x}$ with a probability mass greater than η_0, i.e $p_0(\mathbf{x}) > \eta_0$. The prior precision is the percentage of all local features having a prior value greater than η_0 that belong to an instance in the ground truth. Note that these two evaluation criteria are fully independent from the precise search algorithm used on top of our method and are very useful to compare different prior distributions. However, they do not measure the performances of the whole mining and discovery processes. To this end, we propose to analyze the object recall (for the discovery problem) and the instance recall (for the mining problem) with regard to the number T of precise search probes. We remind that the complexity of our algorithm is mainly $O(T)$. Note that these two recall measures depend on the performances of the precise search algorithm but that the relative performances are mainly determined by the prior sampling distributions to be compared. The precise search parameters used in the following experiments are proposed in [7, 8, 9] : RMMH: dot-product kernel, $M = 40$ and 128 bits for Hamming Embedding; APMP-LSH: KNN size is 300 and retrieval recall is 0.80; Match verification: A Contrario threshold is 0.95.

7.2 Prior distributions evaluation and parameters study

7.2.1 WG contribution

All the following experiments have been made on the FlickrBelgaLogos dataset.

WG attributes.

This first evaluation aims at measuring the contribution of the different attributes of the WG vector $\mathbf{\Delta}$. We can see in Figure 4 that using WG features is clearly improving the recall/precision. The curve $0, 0, 0, 0$ measures the performances without using any WG (visual hashing and vote only). The others curves have been computed with a subset of the proposed WG features. They show that each attribute (position, scale and orientation) provides a consistent contribution and that the best recall/precision trade-off is finally obtained with the complete set of attributes.

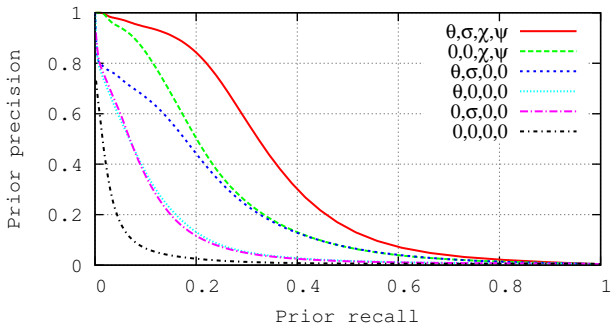

Figure 4: Contribution of WG attributes $\theta, \sigma, \chi, \psi$ on prior recall and precision

WG hash codes length.

In this experiment, we are studying the effect of the number of hash functions k' used to compute the WG hash codes and the number of bits γ used for the scalar quantization of each projection. To improve the readability of the graph displayed in Figure 5, we manually grouped the curves into four clusters (identified by four colors), according to their performance range. The best qualities are obtained with $\gamma \in [3, 5]$ and $k' \in [3, 10]$. We remark that using a high value for γ is a rather a bad idea, since it decreases the probability of having a significant number of collisions. On the contrary, a low value for γ can be counterbalanced by a higher number k' of hash functions, to increase the histogram discrimination. In the following experiments, we use the parameters of the black curve ($k' = 6$ and $\gamma = 4$).

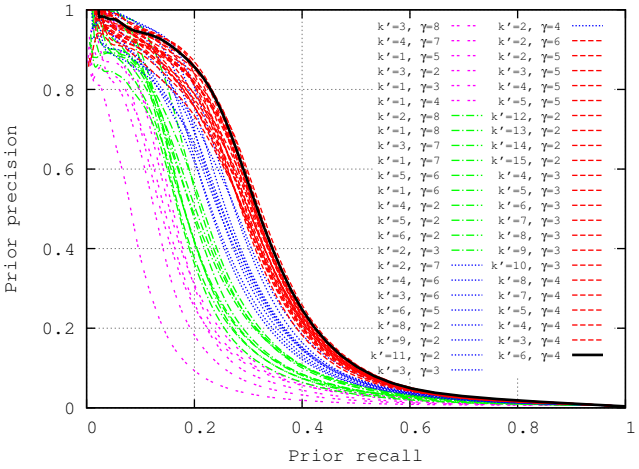

Figure 5: WG hash codes length

Number of WG histograms.

The evaluation presented in Figure 6 is comparing the quality achieved with different value of L', i.e. the number of sparse LSH-based histograms. We also give the performances of a standard multi-dimensional grid, using the

[2]http://www-sop.inria.fr/members/Alexis.Joly/BelgaLogos/FlickrBelgaLogos.html

same number of bits overall. We can see that LSH-based histograms are clearly better than the fix grid. Regarding the value of L', it is not profitable to use more than one single LSH-based histogram, contrary to high-dimensional feature space (the low dimension of our WG feature space is equal to 4 attributes).

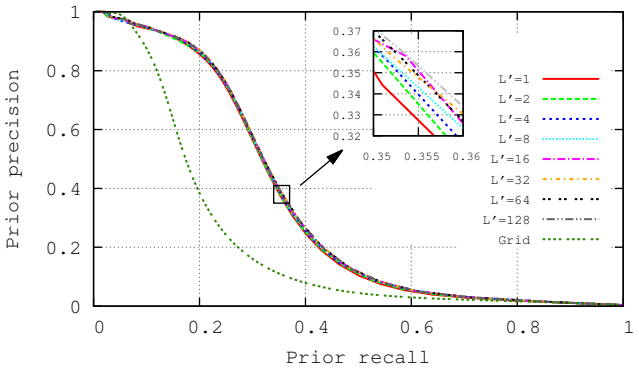

Figure 6: Number of WG histograms

7.2.2 *Visual hashing tuning*

The two main parameters of the visual hashing are L, the number of RMMH tables and α the threshold on the collision probability. For the larger value of $L = 64$ we first estimated an empiric optimum for α that was found to be equal to $\hat{\alpha} = 0.904$. This actually corresponds to keep only very similar visual features. As a comparison, the average collision probability for an arbitrary pair in the exact 1-nn graph is similarly equal to 0.902. That means that a high precision on the kept visual matches is more profitable than high recall. In the following experiments, we used $\hat{\alpha} = 0.9$, that we think is a quite generic value, even for other visual features if they are hashed by RMMH.

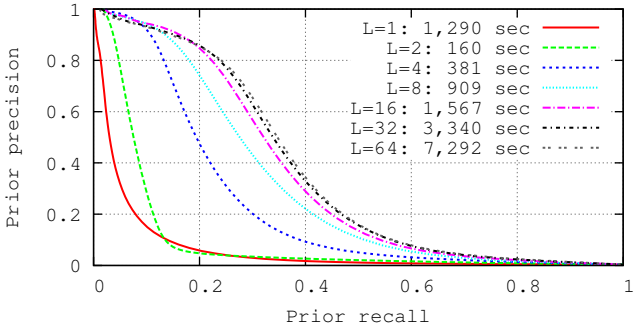

Figure 7: Number of visual hash tables

Regarding Figure 7, using more than 16 visual hash tables is not profitable.

Note that the time needed to compute the prior distribution with a single visual hash table is much more expensive than with two tables, because of the impossibility to filter on the inter-tables collision frequency f_L when $L = 1$.

7.3 Full mining and discovery process evaluation

We now evaluate our complete objects mining algorithm based on the weighted adaptive sampling strategy initialized with our WG prior distribution p_0. To this end, we measure on FlickrBelgaLogos the instance and object recall with regard to the number T of precise search probes.

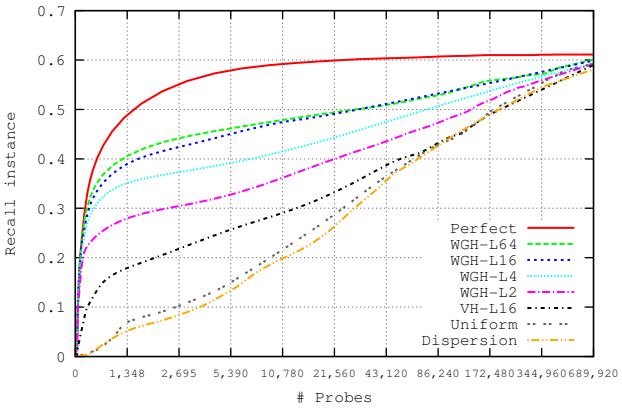

Figure 8: Instance recall vs. number of probes

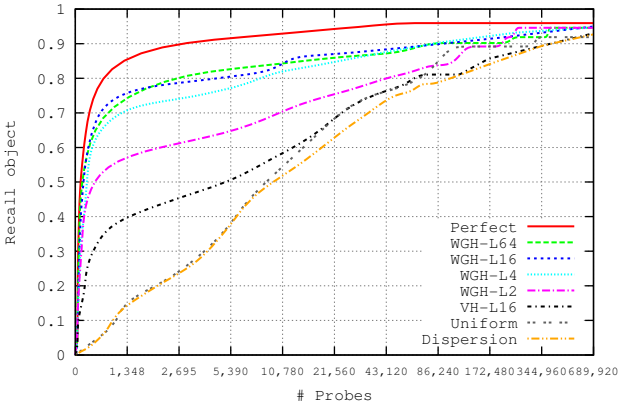

Figure 9: Object recall vs. number of probes

We compare our proposed prior distribution to a uniform distribution ($p_0(\mathbf{x}_i) = 1/N$, $\forall \mathbf{x}_i \in \mathbf{X}$), a perfect prior distribution ($p_0(\mathbf{x}_i) = 0$, $\forall \mathbf{x}_i \notin \{X_s^m\}$ and $p_0(\mathbf{x}_i) = 1/N$, $\forall \mathbf{x}_i \in \{X_s^m\}$), and to the feature-space dispersion prior proposed in [11]:

$$p_0(\mathbf{x}_i) = \frac{1}{d} \sum_{j=1}^{d} \mathrm{Var}_{R_i} \mathbf{x}[\mathbf{j}]$$

with $\mathbf{x} \in R_i$ and R_i is the set of spatial neighbors of $\mathbf{x}_i$, and d the dimension of $\mathbf{x}_i$.

Figures 8 and 9 shows that our new prior distribution (with typically $L = 16$ RMMH tables) is much closer to the perfect distribution than to the uniform one. This proves that the WG scores $z_0(\mathbf{x}_i)$ quantify well the likelihood that a feature $\mathbf{x}_i$ belongs to a small frequent object of the ground truth. Interestingly all evaluated distributions converge to 91% of discovered objects and 61% of discovered instances,

which is actually the maximum reachable with the precise search engine used in this experiment (with a very high number of probes > 600K).
With low numbers of probes, our method discovers much more objects than using other distributions. We see that after 1,348 precise search probes (i.e. half of the number of instances), our best runs have retrieved half of all retrievable instances and objects.
The table 1 gives an overview of the time/quality trade-off of our method compared to the uniform sampling distribution (U) of [11]. We observe that with all costs included, our method allows speed-up factors going from 2 to 18. A visual example of a tentative probe and its precise search results is presented in the Figure 10.

7.4 Real-world experiment: application to transmedia events discovery

We finally ran our method on the very large FrenchTV dataset (with settings $L = 8$, $\alpha = 0.9$). The visual hashing step was achieved in 4 hours. The computation of the prior distribution (with WG) ran in 17 hours and used a maximum of 70GB of RAM. The precise search took 18 hours to run 50,000 probes making in total 39 hours for discovering objects in more than 1.8M images. Mining such big data gives rise to a well known issue: most of TV frames are composed of opening or closing credits, commercials, etc. To reduce this noise and focus on more informative media event, we reranked all the detections by favoring: (i) visual objects appearing across a large number of TV channels (ii) visual objects appearing only during a short period of time. More precisely, each discovered visual object was affected the following score C:

$$C = \frac{\#\{\mathcal{C}\} . \hat{f}}{f_d} \tag{22}$$

where $\#\{\mathcal{C}\}$ is the number of channels, $\hat{f}$ is the estimation of the object frequency f (number of detections) and f_d is the temporal frequency (number of days). According to this criterion, Figure 2 displays the top-1 discovered object for 3 different categories of object size. They all illustrate one of the most salient media events in the concerned period.

8. NEW EXPERIMENTS

This section provides additional experiments over our initial version of this work published in [12]. This mainly concerns a comparative study with the popular Geometric Min-hashing scheme [2]. Geometric min-Hashing (GmH) is an efficient hashing scheme that combines visual appearance and neighborhood interaction of image features. It has been shown that its high indexing ability - high recall and low false positive rate - makes it an powerful tool for a number of applications, such as image retrieval, large scale image clustering, and automatic object discovery in large collections of images. However, we will show in this section that GmH fails in discovering really small objects and that our weak-geometry hashing scheme can drastically improves performances.

8.1 Using GmH as a prior distribution

Geometric min-Hashing [2] is based on four main steps:

1. Construction of a visual vocabulary using a K-Means clustering on the local features
2. Local features filtering based on their unicity in each image and neighborhood constraints
3. Sketches generation: create random s-tuples of neighboring local features following precomputed random permutations of the visual vocabulary
4. Sketches collisions: each pair of colliding sketches is considered as an object seed and processed further for a more accurate objects detection based on co-segmentation

To compare the ability of GmH to generate relevant object seeds compared to our method, we propose converting the collision scores of GmH to a prior sampling distribution on the features, as done for our weak geometry hashing scheme (see section 5.3). Figures 11 and 12 present the precision/recall curves for various values of the two main parameters of GmH, i.e the number k of sketches per image and the size of the vocabulary. Please notice that we evaluated GmH with much stronger parameters than the one typically used in the original paper. The recommended number of sketches is actually of about 60 sketches per image whereas our curves show that 100K sketches are required to get about 20% recall/precision on FlickrBelgaLogo dataset. Using 8 millions of clusters in the vocabulary was also never experimented earlier to our knowledge (KMeans was computed thanks to its kd-tree based approximation described in [17] and publicly available[3]). Finally, it is important to remark that the curves should not be interpolated up to their intersection with the x-axis because the last point correspond to the case where all local features present in at least one seed are kept.

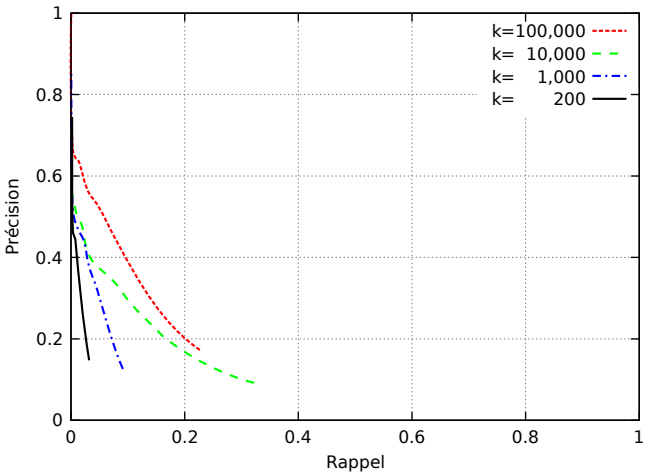

Figure 11: GmH [2]: Number of sketches per image

8.2 Boosting GmH with Weak Geometry Constraints

To clearly understand the contribution of our weak geometry hashing step in the whole process, we also developed a stage strategy on top of the classical GmH approach evaluated above. We therefore remind that an important difference between our first visual hashing step and GmH is that

[3]http://www.robots.ox.ac.uk/ vgg/software/fastann/

Table 1: **Time and speed-up of our method over uniform sampling. VH refers to Visual hashing (without using WG), and WGH to WG hashing. L16 means $L = 16$. Hashing time is the time needed to hash the local features. Note that we need at least 128 bits per feature for the precise search. Prior time is the computation time of the prior distribution. Then for a given recall, we show the total time (hashing + prior + precise search). Speed-up factor (in red) is given, compared to the uniform distribution [11] total time.**

	U [11]	VH-L16	WGH-L2	WGH-L4	WGH-L16	WGH-L64
Hashing time	627s	1,567s	634s	641s	1,567s	7,292s
Prior time	0s	1,128s	163s	320s	1,280s	5,120s
Full time	**U [11]**	**VH-L16**	**WGH-L2**	**WGH-L4**	**WGH-L16**	**WGH-L64**
Rec. ins. = 0.20	8,858s	4,482s / x2.0	919s / x9.6	1,045s / x8.5	2,925s / x3.0	12,491s / x0.7
Rec. ins. = 0.30	23,588s	14,753s / x1.6	2,874s / x8.2	1,282s / x18.4	3,073s / x7.7	12,597s / x1.9
Rec. ins. = 0.40	55,825s	47,731s / x1.2	19,676s / x2.8	7,471s / x7.5	4,113s / x13.6	13,225s / x4.2
Rec. ins. = 0.50	159,973s	158,105s / x1.0	120,289s / x1.3	60,146s / x2.7	30,751s / x5.2	37,066s / x4.3
Full time	**U [11]**	**VH-L16**	**WGH-L2**	**WGH-L4**	**WGH-L16**	**WGH-L64**
Rec. obj. = 0.30	4,078s	3,054s / x1.3	901s / x4.5	1,039s / x3.9	2,916s / x1.4	12,467s / x0.3
Rec. obj. = 0.50	7,854s	7,111s / x1.1	1,136s / x6.9	1,163s / x6.8	2,982s / x2.6	12,520s / x0.6
Rec. obj. = 0.70	20,661s	23,081s / x0.9	9,898s / x2.1	1,830s / x11.3	3,248s / x6.4	12,916s / x1.6
Rec. obj. = 0.90	281,205s	199,128s / x1.4	184,419s / x1.5	55,234s / x5.1	96,441s / x2.9	78,261s / x3.6

Figure 10: The first probe issued from the WGH-L64 prior distribution (left image), and its precise search results, in the FlickrBelgaLogos dataset.

our method generates candidate matches of local features (i.e. pairs of features matched across two images) whereas GmH generates candidate pairs of matched sketches where each sketch is typically itself a pair of two neighboring local features in a given image. Instead of hashing a single weak geometry attributes vector as in our method, we therefore simply generalize it to GmH by hashing separately the two weak geometry attributes vector obtained from the two individual matches of a single sketch collision. The obtained method will be evaluated in the next subsection together with our own approach and the raw GmH method.

8.3 Overall comparison

Figure 13 presents the overall comparison results. The red curves correspond to our method with an increasing number of visual hash tables, up to a typical and reasonable value of 64 tables. The blue curve corresponds to GmH with a typical value of 1M visual words and a huge number of 100K sketches per image (much higher than the 60 sketches per image recommended in the original paper). The purple curve corresponds to the augmented GmH algorithm using our weak geometry hashing algorithm as a second step. It shows that it also improves the raw performances of GmH but in a more limited way due to the fact that GmH already uses neighborhood constraints. Overall, the retrieval performances of both GmH and augmented GmH are still far from what we are able to achieve with our full scheme. Using only 4 visual hash tables in our method is sufficient to clearly outperform GmH (based on 1M k-means clusters and 100K sketches per image).

9. CONCLUSION

Several contributions were introduced in this paper: we first revisit the problem of discovering and mining small visual objects with a new formalism and a modeling of the complexity of existing approaches based on sampling. We then proposed a new hashing-based method for computing the likelihood that a given local feature belongs to a small frequent object. Its main originality relies on using Sparse multi-dimensional Histograms of WG attributes that efficiently group geometrically consistent visual matches in the whole dataset. We finally introduced a new evaluation dataset and testbed that allow us to complete the first accurate evaluation of small visual objects discovery algorithms. Results did show that our WG hashing scheme provides a

Table 2: Top 1 detected objects by size category (small, medium, large) in 2051 hours of French TV

G20 Summit *Nov 2011* Small object $< 33\%$ of image area $\#\{\mathcal{C}\} = 6$ $\hat{f} = 15$ $f_d = 2$ $C = 45$			
Kim Jong-il death *Dec 2011* Medium object $< 66\%$ of image area $\#\{\mathcal{C}\} = 6$ $\hat{f} = 8$ $f_d = 1$ $C = 48$			
The Artist *Feb 2012* Large object $> 66\%$ of image area $\#\{\mathcal{C}\} = 7$ $\hat{f} = 39$ $f_d = 8$ $C = 34$			

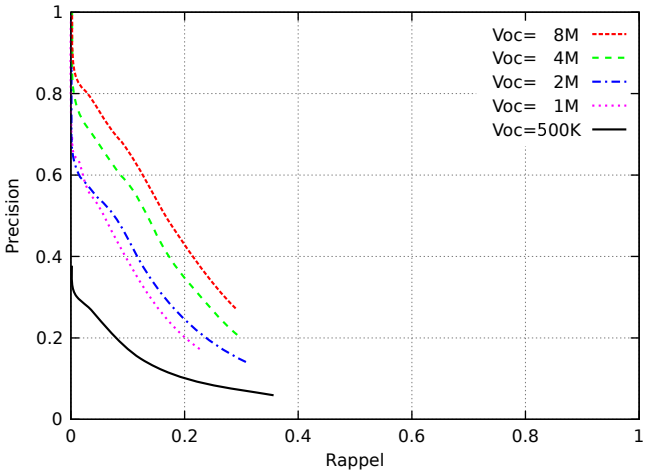

Figure 12: GmH [2]: Vocabulary size

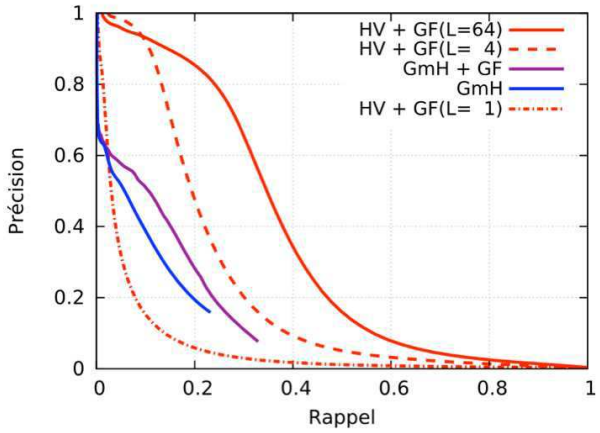

Figure 13: Overall comparison between our method and GmH [2]

good approximation of a perfect distribution, i.e. where only features belonging to an object of the ground truth would be considered as candidates for a precise search. In future work, we foresee evaluating our method by using pairs of local features in the visual hashing step, like done in [2] or [10]. We also plan to extend our WG hashing based on sparse histograms to more attributes including new geometric parameters (such as translation [22]) or contextual information.

10. ACKNOWLEDGMENTS

This work was partially funded by the French ANR project OTMedia[4] (No. 2010CORD015) and the EU project GLOCAL[5] (Grant Agreement No. 248984).

11. REFERENCES

[1] O. Chum, A. Mikulik, M. Perdoch, and J. Matas. Total recall ii: Query expansion revisited. In *CVPR*, pages 889–896, Colorado Springs, USA, 2011. IEEE.

[2] O. Chum, M. Perdoch, and J. Matas. Geometric min-hashing: Finding a (thick) needle in a haystack. In *CVPR*, pages 17–24. IEEE, June 2009.

[4] http://www.otmedia.fr
[5] http://www.glocal-project.eu

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

[4] M. Datar and P. Indyk. Locality-sensitive hashing scheme based on p-stable distributions. In *SCG*, pages 253–262. ACM Press, 2004.

[5] M. A. Fischler and R. C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, 1981.

[6] H. Jégou, M. Douze, and C. Schmid. Hamming Embedding and Weak Geometry Consistency for Large Scale Image Search. Research Report RR-6709, INRIA, 2008.

[7] A. Joly and O. Buisson. A Posteriori Multi-Probe Locality Sensitive Hashing. In *ACM Multimedia*, pages 209–218, Vancouver, Canada, oct 2008.

[8] A. Joly and O. Buisson. Logo retrieval with a contrario visual query expansion. In *ACM Multimedia*, pages 581–584, Beijing, China, october 2009.

[9] A. Joly and O. Buisson. Random Maximum Margin Hashing. In *CVPR*, Colorado Springs, 2011. IEEE.

[10] Y. Kalantidis, L. Pueyo, M. Trevisiol, R. van Zwol, and Y. Avrithis. Scalable triangulation-based logo recognition. In *ICMR*, pages 20:1–20:7, Trento, Italy, 2011. ACM.

[11] P. Letessier, O. Buisson, and A. Joly. Consistent visual words mining with adaptive sampling. In *ICMR*, pages 49:1–49:8. ACM, april 2011.

[12] P. Letessier, O. Buisson, and A. Joly. Scalable mining of small visual objects. In *Proceedings of the 20th ACM international conference on Multimedia*, pages 599–608, 2012.

[13] K. Ling and G. Wu. Frequency based locality sensitive hashing. In *ICMT*, pages 4929 –4932, july 2011.

[14] D. Lowe. Object recognition from local scale-invariant features. In *ICCV*, page 1150, Kerkyra, 1999. IEEE.

[15] F. Olken. *Random Sampling from Databases*. PhD thesis, University of California, 1993.

[16] J. Philbin. *Scalable Object Retrieval in Very Large Image Collections*. PhD thesis, Univ. of Oxford, 2010.

[17] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

[18] 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*, pages 1–8, Anchorage, USA, june 2008. IEEE.

[19] G. F. Pineda, H. Koga, and T. Watanabe. Object discovery by clustering correlated visual word sets. In *ICPR*, pages 750–753, Washington, USA, 2010. IEEE.

[20] T. Tuytelaars, C. H. Lampert, M. B. Blaschko, and W. Buntine. Unsupervised object discovery: A comparison. *IJCV*, 88:284–302, June 2010.

[21] T. Weyand and B. Leibe. Discovering favorite views of popular places with iconoid shift. In *ICCV*, pages 1132–1139. IEEE, Nov. 2011.

[22] W. Zhao, X. Wu, and C.-W. Ngo. On the annotation of web videos by efficient near-duplicate search. *IEEE Transactions on Multimedia*, 12(5):448–461, 2010.