Feature Map Hashing: Sub-linear Indexing of Appearance and Global Geometry

Yannis Avrithis
National Technical University
of Athens
Iroon Polytexneiou 9
Zografou, Greece
iavr@image.ntua.gr

Giorgos Tolias
National Technical University
of Athens
Iroon Polytexneiou 9
Zografou, Greece
gtolias@image.ntua.gr

Yannis Kalantidis
National Technical University
of Athens
Iroon Polytexneiou 9
Zografou, Greece
ykalant@image.ntua.gr

ABSTRACT

We present a new approach to image indexing and retrieval, which integrates appearance with global image geometry in the indexing process, while enjoying robustness against viewpoint change, photometric variations, occlusion, and background clutter. We exploit shape parameters of local features to estimate image alignment via a single correspondence. Then, for each feature, we construct a sparse spatial map of all remaining features, encoding their normalized position and appearance, typically vector quantized to visual word. An image is represented by a collection of such *feature maps* and RANSAC-like matching is reduced to a number of set intersections.

Because the induced dissimilarity is still not a metric, we extend min-wise independent permutations [3] to *collections* of sets and derive a similarity measure for feature map collections. We then exploit sparseness to build an inverted file whereby the retrieval process is sub-linear in the total number of images, ideally linear in the number of relevant ones. We achieve excellent performance on 10^4 images, with a query time in the order of milliseconds.

Categories and Subject Descriptors

H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing[Indexing methods]; H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Search process*

General Terms

Algorithms and Experimentation

Keywords

sub-linear indexing, indexing geometry, image retrieval, hashing, feature maps

1. INTRODUCTION

Geometry is essential in many problems of computer vision like feature correspondence, image registration, wide baseline stereo matching, object recognition, and retrieval. And it has been more so in early years when features were non-discriminative, *e.g.* points. With the advent of more discriminative features and descriptors, discarding geometry altogether has been an "easy" way to deal with viewpoint change and occlusion. The success of the *bag-of-words* model, largely due to its very low computational cost, has come as quite a surprise to many, for instance in the seminal work of Sivic and Zisserman [23].

In order to boost performance at large scale however, geometry is still essential. Even if weaker or stronger geometric models are feasible in tasks like registration or recognition, this is clearly not the case for image retrieval. State of the art approaches are still based mostly on appearance in the filtering stage, while geometric or spatial constraints typically come as a second, re-ranking stage. This is the case *e.g.* in Philbin *et al.* [19] where the need is identified for including spatial information in the index itself. Even in more recent work, this has only been achieved in the form of weak geometric constraints as in Jegou *et al.* [13], or local geometry, as in Chum *et al.* [7]. On the other hand, global geometry indexing is at least as old as *geometric hashing* by Lamdan and Wolfson [16]. To our knowledge, no work has been reported that can index appearance and global geometry for large scale image retrieval.

This is exactly our attempt in the present work. One of our starting points is [19] where spatial matching is performed as a special case of RANSAC [11]. Shape parameters of local features are used to generate each hypothesis using a single feature correspondence. We go a step further and for each feature we encode the normalized position and appearance of all remaining features in a sparse histogram that we call a *feature map*. We then extend *min-wise independent permutations* [3] and derive a similarity measure for feature map collections. With the use of an inverted file, the retrieval process becomes sub-linear in the total number of images. Further, the returned images are associated with a rough estimate of the relevant geometric transformation. For the same processing time, we effectively increase the number of images we verify by an order of magnitude.

Because our work draws on several existing approaches, section 2 provides a background on a number of related problems in shape matching and feature correspondence. Section 3 derives our novel feature map representation along with the associated matching process. Hashing and indexing is then presented in section 4, while other related work is not discussed until our approach has been presented (section 5). Implementation, experiments and discussion are provided in sections 6, 7 and 8, respectively.

2. BACKGROUND

We start by examining a number of simple models for matching sets of features that are based on geometry, appearance, or both. We observe how these models can provide solutions for alignment, correspondence, and outliers and derive a single model that we will attempt to further simplify in the following sections.

Shape matching. Let $P, Q \subseteq \mathbb{R}^2$ be two finite sets of points denoting the positions of the features of two images. The features are taken as non-discriminative, that is, only their coordinates are known. Assume for the moment that $|P| = |Q|$ and that there is a known one-to-one mapping $\pi : P \to Q$. In the statistical theory of shape (Dryden and Mardia [10]), one of the most well studied problems is the estimation of the optimal geometric transformation *aligning* the two sets

$$S_T(P, Q; r) = \max_{B,t} \sum_{p \in P} r(Bp + t, \pi(p)), \tag{1}$$

where $r(p, q)$ is an arbitrary spatial similarity (proximity) measure, t is a translation and $B \in \mathbb{R}^{2\times2}$ is typically a similarity transformation, which we will assume here to be affine. This problem never appears as such in our case, due to unknown feature correspondence and outliers.

Feature correspondence. Now, drop the known mapping assumption and rather assume discriminative features specified by finite descriptor sets $X, Y \subseteq \mathbb{R}^d$. Ignoring positions for now, the following *assignment* problem can deal with unknown correspondence and, partially, outliers:

$$\begin{aligned} S_A(X, Y; s) &= \max_{\{a\}} \sum_{x \in X} \sum_{y \in Y} a_{x,y} s(x, y) && (2) \\ s.t. \quad & \sum_{x \in X} a_{x,y} \le 1, \quad \forall y \in Y && (3) \\ & \sum_{y \in Y} a_{x,y} \le 1, \quad \forall x \in X && (4) \\ & a_{x,y} \in \{0, 1\}, \quad \forall x \in X, y \in Y && (5) \end{aligned}$$

where $s(x, y)$ is again an arbitrary similarity measure in the descriptor space $\mathbb{R}^d$, and $a_{x,y} = 1$ represents correspondence between x and y. Despite the loss of geometry, this is a very important problem because it can work well in practice if the features are discriminative enough.

Bag-of-words. Further, define a visual vocabulary or codebook $\mathcal{V} \subseteq \mathbb{R}^d$ with $|\mathcal{V}| = k_v$ elements or visual words, derived *e.g.* by vector quantization on a training set. Let $v(x)$ be the quantized version of descriptor x, $H_v(X) = \{x \in X : v(x) = v\}$ the set of elements of X mapped to word v, and $h_v(X) = |H_v(X)|$ their count. Defining similarity $s_{\mathcal{V}}(x, y) = \mathbb{1}_{v(x)=v(y)}$, it is easy to see that

$$S_A(X, Y; s_{\mathcal{V}}) = \sum_{v \in \mathcal{V}} \min(h_v(X), h_v(Y)), \tag{6}$$

that is, the histogram intersection of the visual word representations of sets X and Y. In an analogous way, we may replace the *one-to-one* matching scheme of (2)-(5) with an *one-to-many* voting scheme

$$S_M(X, Y; s) = \sum_{x \in X} \sum_{y \in Y} s(x, y) \tag{7}$$

and confirm that the similarity of the visual word representations is equivalent to an inner product (Jegou *et al.* [13]),

$$S_M(X, Y; s_{\mathcal{V}}) = \sum_{v \in \mathcal{V}} h_v(X) h_v(Y). \tag{8}$$

When histograms are normalized, this is the well-known *cosine similarity* measure used in information retrieval. Either way, combined *e.g.* with an inverted file structure to exploit sparsity, this is a simple and fast method that is very common in the filtering stage of retrieval.

Towards RANSAC. One-to-many matching may give unexpected results according to our perception of dissimilarity [22]. It is however easier to estimate, especially when using a codebook. Following (7), let us start with a set of *tentative correspondences*, either defined in a nearest neighbor sense, or

$$\mathcal{X}(X, Y; s) = \{(x, y) \in X \times Y : s(x, y) > \delta_s\}, \tag{9}$$

When we do use a codebook that is large enough, tentative correspondences (9) do not differ much from the one-to-one scheme.

Coming back to geometry, we now assume features are equipped with both position p and descriptor x. To simplify notation, we represent a feature by either p or x alone, depending on the context. We write the association as $p = p(x)$. Similarly, $q = q(y)$ for a second image. Given a specific set of correspondences $\mathcal{X}$ and the pairs of relevant positions $\mathcal{P} = \mathcal{P}(\mathcal{X}) = \{(p, q) \in P \times Q : p = p(x) \wedge q = q(y) \wedge (x, y) \in \mathcal{X}\}$, return to (1) and maximize w.r.t. transformation (B, t) over a finite set of *hypotheses* $\mathcal{H}$:

$$S_R(P, Q; \mathcal{P}, r) = \max_{(B,t) \in \mathcal{H}} \sum_{(p,q) \in \mathcal{P}} r(Bp + t, q). \tag{10}$$

When hypotheses are selected at random following a specific strategy and spatial similarity is defined by a uniform kernel $r_\epsilon(p, q) = \mathbb{1}_{\|p-q\|_2 < \epsilon}$ that just counts inliers, the above result is not too different from RANSAC. Given appropriate correspondences, it can jointly solve for alignment and outliers.

3. FEATURE MAPS

Local patches. We assume here that each local feature is additionally associated with an image patch L. Again, for convenience, we represent a feature by p, x or L alone, depending on the context. We write the association between L and p as $L = L(p)$. Similarly, $R = R(q)$ for a second image. Following Rothganger *et al.* [21], the patch is a parallelogram represented by matrix

$$L = L(p) = \begin{bmatrix} a & b & p \\ 0 & 0 & 1 \end{bmatrix}, \tag{11}$$

where p is now the center and $a, b \in \mathbb{R}^2$ are the vectors from p to the midpoints of the two sides, as shown in Figure 1. The *rectified* patch $\mathcal{R}_0$ is represented by the identity matrix I_3 and is transformed to the patch via L, while the patch is rectified back to $\mathcal{R}_0$ via L^{-1}. So L stands either for a patch

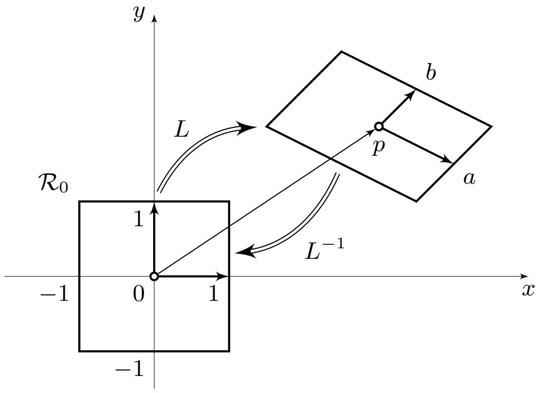

Figure 1: Original and rectified image patch. Axes are spatial, do not confuse with descriptor vectors $x \in X$, $y \in Y$.

or an affine transform. The above formulation is equivalent to *local affine frames* [5].

Single correspondence hypotheses. Given a patch correspondence $L \leftrightarrow R$, the transformation from one patch to the other is RL^{-1}. If the two images are related via a homography, and since by extraction patches are affine-covariant, Köser *et al.* [15] *extrapolate* the transformation to the entire image frame and study image alignment via this single affine correspondence. Philbin *et al.* [19] further observe that each correspondence provides a transformation hypothesis. Hypotheses are now $O(n)$ and we can enumerate them all:

$$S_H(P, Q; \mathcal{P}, r) = \max_{A \in \mathcal{H}(\mathcal{P})} \sum_{(p,q) \in \mathcal{P}} r(A\mathbf{p}, \mathbf{q}). \qquad (12)$$

Here, $\mathbf{p}$, $\mathbf{q}$ denote the *homogeneous coordinates* of vectors p, q that are 3-vectors in projective space $\mathbb{P}^2$, while $A \in \mathbb{R}^{3\times3}$ is an affine transform that includes translation, unlike B in (10). The set of hypotheses is now specified by the set of correspondences, $\mathcal{H}(\mathcal{P}) = \{A = RL^{-1} : L = L(p) \wedge R = R(q) \wedge (p,q) \in \mathcal{P}\}$.

Feature set rectification. Instead of constructing transformations RL^{-1} for all correspondences and performing spatial matching at query time like Philbin *et al.* [19], we *extrapolate* each local transform to the entire image frame and *rectify the entire set of features in advance*. Let $p^{(\hat{p})} \in \mathbb{R}^2$ be the Euclidean counterpart of $\mathbf{p}^{(\hat{p})} = \hat{L}^{-1}\mathbf{p}$ for $\hat{L} = \hat{L}(\hat{p})$, that is feature p rectified with respect to feature $\hat{p}$ of the same image. Also let $P^{(\hat{p})} = \{p^{(\hat{p})} : p \in P\}$ be the entire *rectified feature set* with *origin* $\hat{p}$. Similarly define $q^{(\hat{q})}$ and $Q^{(\hat{q})}$ for the second image, using $\hat{R} = \hat{R}(\hat{q})$. Figure 2 shows a random feature set, a transformed and distorted version, and the rectified counterparts with their origins in correspondence — notice how the latter are roughly aligned. Under this formulation, the same set of correspondences $\mathcal{P}$, obtained solely via descriptor matching, is used both for inlier counting and aligning:

$$\begin{aligned} \hat{S}_H(P, Q; \mathcal{P}, r) &= \max_{(\hat{p},\hat{q}) \in \mathcal{P}} \sum_{(p,q) \in \mathcal{P}} r(p^{(\hat{p})}, q^{(\hat{q})}) && (13) \\ &= \max_{(\hat{p},\hat{q}) \in \mathcal{P}} I(\mathcal{P}; \hat{p}, \hat{q}, r), && (14) \end{aligned}$$

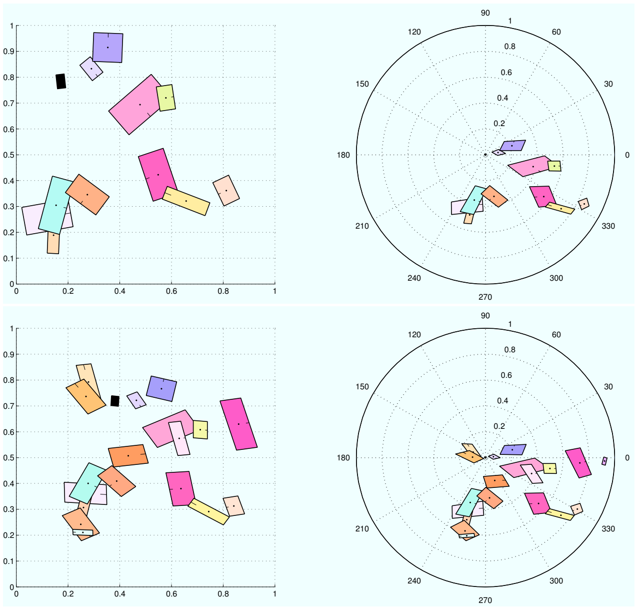

Figure 2: Top left: A random set of patches. Bottom left: The same set under affine transform, where patch position and local shape are distorted, and more patches are inserted. Right: The rectified counterparts; origins are the two black patches on the left. The polar grid specifies the spatial bin limits for the feature maps, with $\tau = 0.95$, $k_\rho = 5$ and $k_\theta = 12$ (see section 6).

where $I(\cdot; \hat{p}, \hat{q}, \cdot)$ is the total count of inliers for hypothesis $(\hat{p}, \hat{q})$. Due to multiplication by $\hat{R}^{-1}$ the similarity measure is not the same as in (12), but in fact distance is now measured in a *rectified coordinate frame* and this seems more appropriate. It makes sense to define the *alignment score*

$$\mathcal{A}(P, Q; \mathcal{P}, \widetilde{\mathcal{P}}, r) = \frac{1}{|\widetilde{\mathcal{P}}|} \sum_{(\hat{p},\hat{q}) \in \widetilde{\mathcal{P}}} \mathbb{1}_{I(\mathcal{P};\hat{p},\hat{q},r) > \gamma} \qquad (15)$$

as the ratio of *seed* correspondences $\widetilde{\mathcal{P}}$ that give more than γ inliers, hence contribute to alignment. Seed correspondences may be set *e.g.* to the correspondences obtained via repeatability or matching score [17]. While these scores focus more on position and appearance respectively, alignment score is significantly more sensitive to local shape. We have measured the alignment score of Hessian-affine [17] and SURF [1] features on 50 pairs of images depicting the same scene from different viewpoints. Varying γ from 5 to 10 inliers, Hessian-affine drop from 18.4% to 14.2% and SURF from 16.3% to 13.4%. Observe that such performance may be of statistical significance, since a single correspondence is enough for our purpose. Despite not supporting affine transforms, SURF appear to be applicable as well.

Quantization. Observe that unlike (12), the summand of (13) assumes aligned features and resembles an overlap measure. It looks like we could use some form of spatial quantization in the rectified frames. Adopt the visual codebook scheme of section 2 and further define *spatial codebook* $\mathcal{U} \subseteq \mathbb{R}^2$ with $|\mathcal{U}| = k_u$ bins. Quantization can be uniform in this case. However, encoding all positions in a finite set is

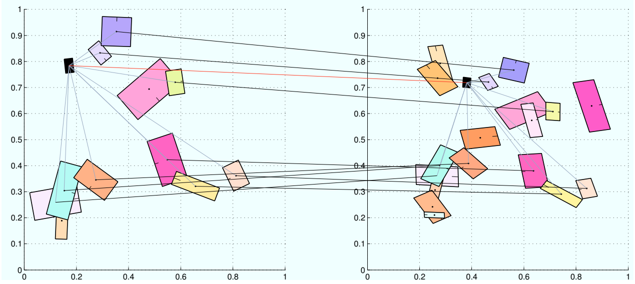

Figure 3: Inliers between two sets of features. Each inlier corresponds to a non zero term of the inner product between corresponding feature maps. Black lines connect inliers. Red line connects the origins. Grey lines connect origins with inlier features.

not trivial; see section 6. Let $u(p)$ be the quantized version of position p, $w = (v, u)$ a joint visual-spatial bin and, for convenience, $w(p) = (v(x), u(p))$ the joint bin of feature p, such that $p = p(x)$. Then, for any rectified feature set $\hat{P}$ (with any origin), let $H_w(\hat{P}) = \{p \in \hat{P} : w(p) = w\}$ be the set of elements of $\hat{P}$ mapped to bin w, and $h_w(\hat{P}) = |H_w(\hat{P})|$ their count. Further, define the *joint codebook* $\mathcal{W} = \mathcal{V} \times \mathcal{U}$ with $|\mathcal{W}| = k_v k_u = k$ bins. The joint histogram of any rectified feature set $\hat{P}$, denoted as $h_{\mathcal{W}}(\hat{P}) \in \mathbb{R}^k$, may be represented as

$$h_{\mathcal{W}}(\hat{P}) = \sum_{w\in\mathcal{W}} h_w(\hat{P})\mathbf{e}_w = \sum_{p\in\hat{P}} \mathbf{e}_{w(p)}, \tag{16}$$

where $\{\mathbf{e}_w \in \mathbb{R}^k : w \in \mathcal{W}\}$ is the standard basis of $\mathbb{R}^k$, such that $\forall w, w' \in \mathcal{W}$, $\mathbf{e}_w^{\mathrm{T}}\mathbf{e}_{w'} = \mathbb{1}_{w=w'}$. Similarly to (8), defining spatial similarity $r_{\mathcal{U}}(p,q) = \mathbb{1}_{u(p)=u(q)}$, (13) becomes

$$\hat{S}_H(P,Q;\mathcal{P},r_{\mathcal{U}}) = \max_{(\hat{p},\hat{q})\in\mathcal{P}} \sum_{w\in\mathcal{W}} h_w(P^{(\hat{p})})h_w(Q^{(\hat{q})}). \tag{17}$$

Using a visual codebook means that correspondences in $\mathcal{X}$ are features x, y belonging to the same visual word. Define $V(X) = \{v \in \mathcal{V} : H_v(X) \neq \emptyset\}$ as the set of visual words present in a feature set and $V(X,Y) = V(X) \cap V(Y)$ the common visual words of feature sets X, Y. Then, if $f_P(\hat{x}) = h_{\mathcal{W}}(P^{(\hat{p}(\hat{x}))})$ is the histogram of P's counterpart that is rectified with respect to $\hat{p}(\hat{x})$, our overall image similarity measure becomes

$$S_F(P,Q;X,Y) = \max_{v\in V(X,Y)} \max_{\substack{\hat{x}\in H_v(X)\\ \hat{y}\in H_v(Y)}} f_P^{\mathrm{T}}(\hat{x}) f_Q(\hat{y}). \tag{18}$$

Feature maps. We call $f_P(\hat{x})$ the *feature map* of P with *origin* $\hat{x}$. The set $F_P = \{f_P(\hat{x}) : \hat{p}(\hat{x}) \in P\}$ is the *feature map collection* of P. Along with the visual word representation $\{H_v(X) : v \in \mathcal{V}\}$, this is all the information we store for image (P, X). Visually, a feature map may be understood as the assignment of rectified features in spatial bins, as on the right of Figure 2. The exact bin layout is discussed in section 6. There is a different map for each origin: we may then think of each origin's map as a *local* descriptor, that encodes the *global* feature set rectified in a *local* coordinate frame. Well aligned feature sets are likely to have maps with a high degree of overlap.

Returning to the example of Figure 2, inliers of the two rectified feature sets are the features lying in the same bins of the joint histogram. These inliers are explicitly shown as correspondences by black lines in Figure 3. Each inlier corresponds to a non-zero term in the inner product of (18). In fact, Figure 3 illustrates all correspondences of the two feature maps having the two black patches as origins. Maximizing over all origins $\hat{x}$, $\hat{y}$ yields our image similarity of (18), where potential origins are constrained to the same visual word. This similarity is the value of the inner product for the best aligned pair of origins from the two images.

We see in (18) a clear separation between (a) *alignment* via inlier count, based on spatial information (P, Q) and (b) *correspondence* based on visual information (X, Y). Each inlier count is written as an inner product operation of two feature maps. This operation is reminiscent of our one-to-many choice in (9); we could equally use a histogram intersection, that we have seen to be more appropriate. On the other hand, the one-to-many scheme increases the number of hypotheses and the chances of alignment. An apparent option for speed and robustness is to use as origins $\hat{x}, \hat{y}$ only features that *map uniquely to visual words*, as in [7]. Precisely, add constraint $h_v(X) = h_v(Y) = 1$ in the outer maximum of (18).

We will refer to (18) as the *feature map similarity* (FMS) between two images. If n is the average number of features per image, the time required for the intersection operation is proportional to the size of feature maps, that is $O(n)$. When the visual codebook is large enough and we use unique origins as specified above, the maximum is taken over $O(j)$ combinations of features where j is the average number of common visual words. The total operation is typically $O(nj)$, and $O(n^2)$ in the worst case. Space requirements are $O(n)$ for a feature map and $O(n^2)$ for a collection in worst case. Savings can be made by constraints on *spatial proximity* via range parameter τ, or *selection of origins*. Both are discussed in section 6. A visualization of fast spatial matching (FastSM) [19] and FMS feature correspondence using SURF is shown in Figure 4.

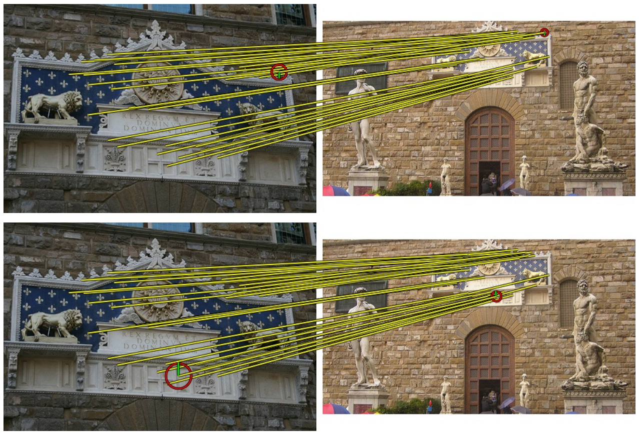

Figure 4: Above: Inliers using FastSM with single feature correspondence. Below: The same for FMS. There are 35 and 32 inliers, respectively. Origins shown in red circles with scale and rotation of the feature. Inliers are shown in yellow lines.

4. FEATURE MAP HASHING

Equation (18) provides a very fast way for matching two images. This is still not enough for indexing, though. The inner product or intersection operation is $O(n)$ with a sparse representation and we will need a sketching function to provide an approximate similarity measure. Before we proceed, let us first relax maximization over all combinations of features. Given two feature map collections F, G representing two images, and assuming there is a one-to-one correspondence between features and feature maps, (18) gives

$$\hat{S}_F(F,G) = \max_{f \in F} \max_{g \in G} s_F(f,g), \tag{19}$$

where $s_F(f,g)$ is either inner product or intersection. Observe the similarity to one-to-many scheme of (7), where summation is replaced by maximum. This implies that a process for matching sets of features could be possibly adjusted to account for sets of feature maps. Locality sensitive hashing typically provides a fast, unsupervised solution. We give a short overview below, highlight the problems, and then present our solution.

Locality sensitive hashing. As defined in [4], given a feature space $\mathbb{F}$ (in our case $\mathbb{F} = \mathbb{R}^k$), by *hash function* we refer to a random mapping $h : \mathbb{F} \to \mathbb{H}$ such that the probability that two objects in $\mathbb{F}$ are mapped to the same hash value in space $\mathbb{H}$ reflects their similarity. A *locality sensitive hashing* (LSH) scheme is a distribution on a family $\mathcal{F}$ of such mappings such that for all $f, g \in \mathbb{F}$,

$$\Pr_{h \in \mathcal{F}}[h(f) = h(g)] = s_{\mathcal{F}}(f,g), \tag{20}$$

where $s_{\mathcal{F}}(\cdot,\cdot) \in [0,1]$ is a similarity measure. In our case it should be in the form of either inner product or intersection, appropriately normalized either way. $\mathbb{H}$ depends on this choice and remains yet to be specified. For an arbitrary hash function $h \in \mathcal{F}$, we define similarity measure $s_h(f,g) = \mathbb{1}_{h(f)=h(g)}$ such that $\mathbb{E}_{h \in \mathcal{F}}[s_h(f,g)] = s_{\mathcal{F}}(f,g)$.

A very interesting result of [4] is that for any similarity function $s_{\mathcal{F}}(\cdot,\cdot)$ that admits an LSH family, distance $1 - s_{\mathcal{F}}(\cdot,\cdot)$ satisfies triangle inequality. We can use this result to show that our similarity measure of entire feature map collections as defined in (19) *cannot* admit an LSH family. Assume inner product similarity $s_F(f,g) = f^{\mathrm{T}} g$ and take for instance collections $F = \{f\}$, $G = \{g\}$ and $H = \{f, g\}$ for feature maps $f \neq g$ normalized such that $\|f\| = \|g\| = 1$ and thus $\hat{S}_F(\cdot,\cdot) \in [0,1]$. Then, $\hat{S}_F(F,H) = \hat{S}_F(H,G) = 1$ and $\hat{S}_F(F,G) = f^{\mathrm{T}} g < 1$, so that the triangle inequality is *not* satisfied by $1 - \hat{S}_F(\cdot,\cdot)$. This implies that we should seek for sketches of *individual* feature maps rather than collections, as shown next.

Random permutations. A feature map is an extremely sparse histogram. The number of features in a bin of a feature map is a random variable that, under uniform distribution in bins (a reasonable assumption at least for the spatial part), is given by a Binomial distribution $\mathrm{Bi}(\cdot; n, k^{-1})$. For n, k_v and k_u in the order of 10^3, 10^5 and 10^2, respectively, the expected value is in the order of 10^{-4}. The bin count thus typically takes values in $\{0,1\}$. At this point, given a feature map f, we define set $\bar{f} \subset \mathcal{W}$ containing only those elements of $\mathcal{W}$ for which the respective bin in f is nonempty: $\bar{f} = \{w \in \mathcal{W} : f^{\mathrm{T}} \mathbf{e}_w \neq 0\}$. The feature space now is $\mathbb{F} = 2^{\mathcal{W}}$, the set of all subsets of $\mathcal{W}$. In this case, the inner product and histogram intersection of two feature maps f, g are both very well approximated by $|\bar{f} \cap \bar{g}|$.

This gives rise to *min-wise independent permutations* [3]. The hashing function here maps objects back to $\mathcal{W}$, that is $\mathbb{H} = \mathcal{W}$ and $h : 2^{\mathcal{W}} \to \mathcal{W}$. Given a feature map $\bar{f} \subset \mathcal{W}$, the function is defined as $h(\bar{f}) = \min\{\pi(\bar{f})\}$, where $\pi : 2^{\mathcal{W}} \to 2^{\mathcal{W}}$ is a permutation chosen uniformly at random in a min-wise independent family $\mathcal{F}$. Then, for all $\bar{f}, \bar{g} \subset \mathcal{W}$,

$$s_{\mathcal{F}}(\bar{f},\bar{g}) = \frac{|\bar{f} \cap \bar{g}|}{|\bar{f} \cup \bar{g}|} = J(\bar{f},\bar{g}), \tag{21}$$

that is, $\bar{f}, \bar{g}$ are mapped to the same value with probability that reflects their Jaccard similarity coefficient.

Sketch matching. In practice, we can estimate $s_{\mathcal{F}}(\bar{f},\bar{g})$ by just approximating $\mathbb{E}_{h \in \mathcal{F}}[s_h(\bar{f},\bar{g})]$ in a statistical sense. All we need to do is construct a set $\Pi = \{\ \pi_i : i = 1, \ldots, m\}$ of m independent random permutations and represent each feature map $\bar{f}$ by *map sketch* $\mathbf{f} \in \mathcal{W}^m$,

$$\mathbf{f} = \mathbf{f}(\bar{f}) = [\min\{\pi_1(\bar{f})\}, \ldots, \min\{\pi_m(\bar{f})\}]^{\mathrm{T}}. \tag{22}$$

Define sketch similarity as simply as $s_K(\mathbf{f},\mathbf{g}) = m - \|\mathbf{f} - \mathbf{g}\|_0$, that is, the number of elements that sketches $\mathbf{f}$, $\mathbf{g}$ have in common. When there is at least one such element, we say that the sketches *collide*. If $\mathbf{F} = \mathbf{F}(F) = \{\mathbf{f}(\bar{f}) : f \in F\}$ is the *map sketch collection* of F, then image similarity reduces to *sketch similarity*

$$S_M(\mathbf{F},\mathbf{G}) = \max_{\mathbf{f} \in \mathbf{F}} \max_{\mathbf{g} \in \mathbf{G}} s_K(\mathbf{f},\mathbf{g}). \tag{23}$$

It is now straightforward to re-establish the constraints of (18) into (23) and maximize over a limited subset of $\mathbf{F} \times \mathbf{G}$ corresponding to features of the two images mapping to the same, unique visual words. All we need to do is for each origin of unique visual word $\hat{v}$ to append $\hat{v}$ to each element $w \in \mathcal{W}$ of the relevant sketch. For two sketches to collide, their origins should then be in correspondence as well. Since each element w is also associated with one permutation π, it is now represented by triplet $(\hat{v}, w, \pi)$.

Since $m \ll n$, (23) gives an extremely fast way of approximating image similarity, with running time $O(mj)$. Space requirements in this case are $O(mn)$ and savings can be made by *origin selection*. What is more important, when m is small enough, for all $\mathbf{f}, \mathbf{g}$ in a pair of unrelated images, $s_K(\mathbf{f},\mathbf{g})$ — therefore $S_M(\mathbf{f},\mathbf{g})$ as well — is zero with high probability. This is because the probability of all m hash values of two feature maps being different is $(1-p_j)^m$, where p_j is the Jaccard coefficient of the maps. A linear scan over all images in the database would then give a very sparse response. This gives rise to an inverted file structure for *sub-linear indexing*, as detailed in section 6. On the other hand, one may show that *collision probability is boosted* for relevant images.

Sketching typically comes at the expense of *low recall* for relevant images [7], which is exactly the reason it was first used to detect *near duplicate* documents [3]. In our case though, at least one feature map pair needs to collide in (23), with probability approximately equal to $np_j p_a[1-(1-p_j)^m]$, where p_a is the probability of precise alignment. With np_j being roughly the average number of inliers, it is quite likely that the number of *aligned* inliers is on average $np_j p_a > 1$, such that *collision probability is boosted* for relevant images.

As the alignment scores presented in section 3 imply, collisions may appear for several pairs of feature maps between two similar images. This fact is not captured by the max operator in (23) which keeps the number of collisions for just the best aligned pair. We therefore expect the *sum* over collisions of all pairs of feature maps to better distinguish relevant from non-relevant images and provide a better final ranking.

$$S_K(\mathbf{F},\mathbf{G}) = \sum_{\mathbf{f}\in\mathbf{F}} \sum_{\mathbf{g}\in\mathbf{G}} s_K(\mathbf{f},\mathbf{g}). \qquad (24)$$

We will refer to this similarity measure (24) as *feature map hashing* (FMH). Like (10) and matching with RANSAC, so does (23) keep only the best transformation hypothesis in order to count inliers. With the use of sum, (24) becomes similar to (7) and the one-to-many voting scheme. However by using as origins only features that map uniquely to visual words it becomes similar to the one-to-one voting scheme. In effect, we let each feature map of one set match at most one feature map of the other set as in (23), however in (24) we use the sum over all sketch similarities.

5. RELATED WORK

Normalizing a set of planar points in a reference coordinate frame defined by a number of reference points is quite common. Examples are *Bookstein* and *Kendall coordinates* [10], where the first two points are arbitrarily chosen as reference, effectively removing transformations up to similarity. To deal with point correspondence and outliers, *geometric hashing* [16] does the same for every possible combination of reference points in the original set. Larger sets of reference points are also considered to remove more complex transformations, *e.g.* 3-point combinations for affine. Positions are quantized as in our work. The complexity is such that it is typically applied to a small number of prototypes for recognition.

A single feature is enough to define each reference coordinate frame in our work, so we can effectively decompose all images in the database and the query image at query time as well. Chum and Matas [5] also implement geometric hashing with a single feature defining each reference frame, but for each feature they encode local shape rather than appearance. We claim it is enough to take local shape into account only when rectifying — on the other hand, we integrate appearance in our joint codebook, rendering a feature map very discriminative. A feature map, seen as a local descriptor, is a concept very close to *shape context* [2], in that the position of all neighboring points is quantized in a log-polar map. However, geometric invariance is only based on global measurements.

Philbin *et al.* [19] approximate RANSAC based on the single correspondence assumption. This spatial verification procedure is used to re-rank up to the top 1000 images. We rather precompute rectified feature maps and relevant sketches and integrate them in the index, so images returned as similar to a query are already geometrically verified. Focusing on memory efficiency, Perdoch *et al.* [18] vector-quantize local shapes without significant loss in precision. Jegou *et al.* [14] also focus on memory usage providing very high index compression but precision is sacrificed.

Jegou *et al.* [13] make another attempt to integrate geometry in the index via *weak geometric consistency* (WGC). They extend bag-of-words (BoW) voting by separately recoding log-scale and orientation differences between features. Local shape is thus taken into account, though not extendable to handle affine transformations; feature position is lost altogether. Baseline BoW and WGC are the two methods we consider in our experiments for comparisons with different re-ranking options, see section 7.

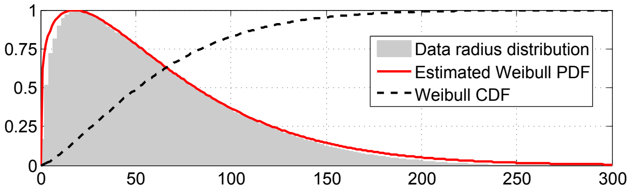

Figure 5: Distribution of radius ρ over 40K rectified feature sets from 200 images of the *European Cities* dataset (see section 7), containing 8M features. ML fitting of Weibull distribution gives $\lambda = 1.23$ and $\kappa = 68.7$.

There are numerous approaches that use hashing in image retrieval, for instance *pyramid match hashing* [12] and *random histograms* [9], without representing geometry. An exception is Chum *et al.* [7], who represent *local* geometry via *geometric min-hashing* (GmH). GmH uses mutual position of features only to verify/reject collisions which is clearly not sub-linear. Geometry is imposed on local neighborhoods only, while in our work it is global and encoded directly into the map sketch element. Origins are chosen randomly in [7] and GmH is used on clustering and small object discovery. We rather focus on general image retrieval and choose origins amongst features that tend to be better aligned, as described in section 6.

6. IMPLEMENTATION

Spatial quantization. In a rectified coordinate frame, we encode positions in polar coordinates (ρ, θ). To ensure that sensitivity to origin scale and orientation errors is independent of distance from the origin, log-polar coordinates are typical, as in [2]. In our case, due to sparsity induced by hashing, it is more important to ensure uniform distribution w.r.t. ρ. As shown in Figure 5, ρ's distribution appears experimentally close to a Weibull distribution $\mathrm{Wb}(\cdot;\lambda,\kappa)$ with λ and κ being the scale and shape parameters, respectively, estimated via maximum likelihood [8]. Then, non-linear transformation with the Weibull CDF $\hat{\rho} = 1-e^{-(\rho/\lambda)^\kappa}$ makes $\hat{\rho}$'s distribution roughly uniform in $[0, 1]$.

Since all large ρ values are mapped close to $\hat{\rho} = 1$ and would otherwise be non-informative, we choose to ignore them by constructing $\bar{\rho} = (\hat{\rho}/\tau)\mathbb{1}_{\hat{\rho}\in[0,\tau]}$ and discarding features with $\bar{\rho} = 0$. *Range parameter* $\tau \in [0, 1]$ controls the balance between local and global geometry. Finally, we uniformly quantize $\bar{\rho}$ and θ in k_ρ and k_θ bins over $[0, 1]$ and $[0, 2\pi]$ respectively, such that $k_\rho k_\theta = k_u$. The spatial mapping $(\bar{\rho}, \theta)$ is illustrated in the right part of Figure 2, where the non-linear distortion near $\bar{\rho} = 1$ is visible.

Origin selection and memory requirements. Using as origins only features that map uniquely to visual words not only makes savings in memory requirements but also speeds up the matching process. Further compression can be achieved by choosing a limited number of features to be used as origins. The role of origins is to provide geometric alignment between the reference frames of two images. We thus retain features assigned to visual words that are top ranked w.r.t. alignment scores, as described in (15). Align-

Image representation	
spatial bin id	5 bits
visual word id	18 bits
joint bin id	4 bytes
map sketch	200 bytes
map collection	40Kbytes

Inverted file	
image id	16 bits
origin id	10 bits
both ids	4 bytes
total	40Kbytes

Table 1: Memory usage for image representation using map sketch collections and for inverted file per image indexed. Memory is calculated for $k_\rho = 4$, $k_\theta = 6$, $k_v = 200$K, $\nu = 200$, $m = 50$ and a maximum database size of 55K images.

ment is measured per visual word over a dataset, as an offline process. As outlined in section 7 we vary the percentage of visual words to retain such that performance is not severely affected. Let ν be the average number of origins (or feature maps) per image according to this selection strategy.

Memory requirements are summarized in Table 1. A map sketch collection, which is the total representation for an image, has ν map sketches on average, with m elements each. A map sketch collection, thus, needs 40 Kbytes to be stored. Each image would require νm entries in an inverted file and 40 Kbytes of memory.

Indexing and filtering. To provide sub-linear access to images, we pre-compute all map sketch collections and store them in an inverted file structure. For each combination of origin visual word $\hat{v} \in \mathcal{V}$, feature bin $w \in \mathcal{W}$, and sketch permutation $\pi \in \Pi$, we store a mapping from triplet $(\hat{v}, w, \pi)$ to a *posting list* of all relevant feature maps and associated images found in the database.

At query time, we compute the map sketch collection of the query image, extract all triplets $(\hat{v}, w, \pi)$, access the relevant posting lists and construct a sparse vector of all feature maps and images found therein, along with relevant counts. In effect, for query sketch $\mathbf{f}$ and database sketch $\mathbf{g}$, we estimate similarity $S_K(\mathbf{F}, \mathbf{G})$ without explicitly computing any zero element of terms $s_K(\mathbf{f}, \mathbf{g})$ in (24). In the process, we also keep the map pair $(\mathbf{f}, \mathbf{g})$ of maximum similarity for each database image. For a database of 50K images and $m = 50$ permutations, only about 2K images are retrieved with a query time of about 50ms, on average.

Local optimization and re-ranking. The best matching pair $(\mathbf{f}, \mathbf{g})$ between query and database image gives us a patch correspondence $L \leftrightarrow R$, thus an initial estimate of the transformation RL^{-1} from one image to the other. Even if the estimate is rough, *e.g.* a similarity transformation using SURF features, we can still recover the correct affine counterpart given at least three inliers. We use the initial estimate as a seed for a single step of method 3 (iterative) of LO-RANSAC [6]. We re-estimate model parameters using a linear algorithm on the complete set of inliers found at each iteration. We have found a maximum of 3 iterations to be enough in our experiments. We re-rank a shortlist of filtered images according to the final number of inliers found. This process is one order of magnitude faster than fast spatial matching (FastSM) [19]. The latter is what we use to re-rank shortlists of BoW and WGC methods in our comparisons in section 7, since no initial estimate is available in this case. Execution time for local optimization is 0.5ms per image on average, with FastSM at around 8ms. Times are measured on our own C++ implementation on a 2GHz Quad Core processor.

7. EXPERIMENTS

Datasets. We have conducted experiments on two publicly available datasets, namely *Oxford Buildings*[1] and *INRIA Holidays*[2], as well as on our own *European Cities*[3] dataset. The first two are small in size (5K and 1.4K images respectively) and typically combined with large sets of unrelated distractor images crawled from Flickr via common user tag queries. *European Cities* consists of 50778 geo-tagged images from 14 European cities, crawled from Flickr using geographic queries covering a window of each city center. A subset of 778 images from 9 cities are annotated into 20 groups of images depicting the same scene, building or landmark. Since not all are landmarks, annotation cannot rely on tags; it is rather a combination of visual query expansion and manual clean-up. Five images are selected as queries from each group, for a total of 100 queries. The remaining 50K images from the other 5 cities are the distractors. Most of them depict urban scenery like the ground-truth, making a challenging distractor dataset. Sets of query images selected for evaluation are depicted in Figure 6, while a representative image from each group of the annotated set is presented in Figure 7. Sample images from the distractor set of $50K$ images are presented in Figure 8.

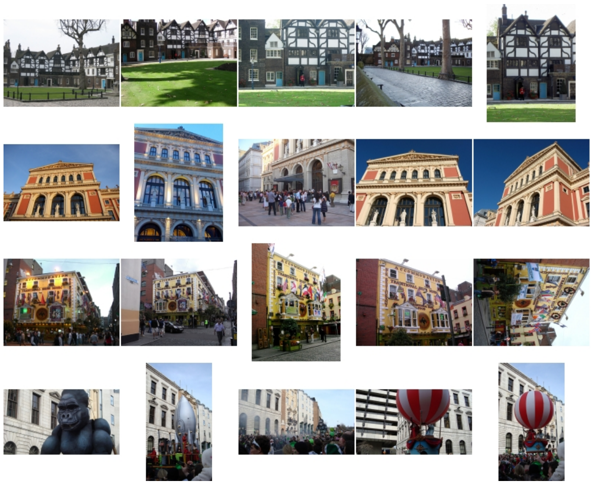

Figure 6: Selected query images of four groups from *European Cities* dataset used in the evaluation.

Evaluation protocol. Our focus has been on demonstrating the benefit from global geometry indexing. Our experiments do include comparisons to baseline bag-of-words and other methods to index and rank according to geometry. In all experiments, we have resized images to a maximum resolution of 500×500 pixels. We have extracted SURF features [1] and kept a maximum number of $n = 1000$ features per image. We use a $k_v = 200$K visual codebook trained from a set of images of urban scenes that are not part of our evaluation datasets. Approximate k-means [19] was used for codebook creation. Our BoW implementation uses dot

[1] http://www.robots.ox.ac.uk/ vgg/data/oxbuildings/

[2] http://lear.inrialpes.fr/ jegou/data.php

[3] http://image.ntua.gr/iva/datasets/

Figure 7: Representative images from all groups of the *European Cities* dataset used in the evaluation.

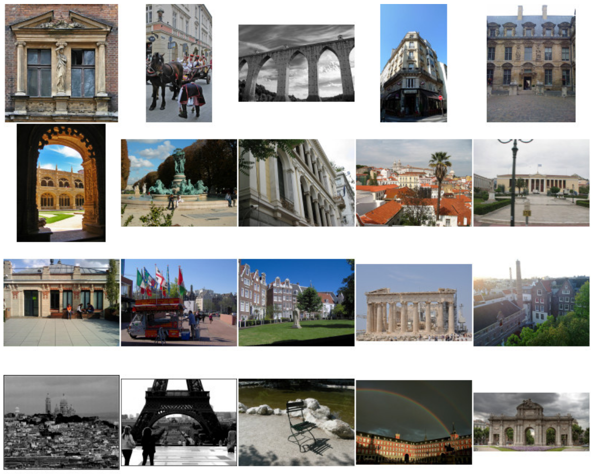

Figure 8: Sample distractor images from *European Cities* dataset.

product similarity on L_1-normalized vectors including tf-idf weighting. Our WGC implementation uses no prior knowledge for scale and orientation. We evaluate overall performance via mean average precision (mAP).

Tuning. Experimenting on the ground truth images of *European Cities*, we have found sketch length $m = 50$ to be a good compromise between high recall and sparse enough responses. mAP measurements on the 100 queries of the same dataset for $\tau = 0.95$ in Table 2 show best performance for intermediate levels of spatial quantization. Coarse bins loosen spatial matching and retrieve more distractors, while fine ones increase sensitivity to feature alignment. We have chosen $k_\rho = 4$ and $k_\theta = 6$. Similar range parameter τ experiments indicate stable performance in [.5, .95]. We have selected $\tau = 0.7$ as a compromise between global geometry representation and space requirements.

We use the 778 annotated images from *European Cities* dataset to measure alignment per visual word as described in section 6 and vary the percentage of visual words to retain. We measure mAP over the 100 query images. Table 3 shows mAP for varying average number of origins ν. When using as origins only features that map uniquely to visual words, without any other selection strategy, ν is 720 on average. We finally force ν to be equal to 200 by choosing the appropriate percentage of visual words, as a good compromise between performance and memory requirements.

$k_\rho \times k_\theta$	2×3	4×6	8×12	16×24
mAP	0.663	0.689	0.648	0.618

Table 2: Mean average precision for different spatial quantization levels using FMH on the *European Cities* dataset with $m = 50$ and $\tau = 0.95$.

ν	720	500	300	200	100
mAP	0.687	0.682	0.678	0.676	0.642

Table 3: Mean average precision for varying average number of origins ν on the *European Cities* dataset.

Results. Figure 9 presents the comparison of the proposed approach with bag-of-words (BoW) and weak geometric consistency (WGC)[13] on the *European Cities* dataset with and without re-ranking, for a varying number of distractor images. BoW with FastSM is exactly the method proposed in [19]. FMH clearly outperforms other methods showing a benefit from global geometry indexing, especially at larger scale. It is rather surprising that precision stabilizes at the high end of database sizes. For re-ranking we have allowed a shortlist of 1000 images for local optimization with FMH, which takes less time than 100 images for FastSM with BoW and WGC, that is, on average, 500ms and 800ms respectively. Figure 10 shows example queries and ranked retrieved images from the *European Cities* dataset with 50K distractors using FMH without re-ranking. When using BoW without any geometric information more false images are retrieved and in a higher rank as shown in Figure 11. Despite the use of geometry in FMS false correspondences may appear due to "noisy" visual words as in the example of Figure 12. However this is not always a problem in our retrieval process. A false image would also need to get a vote after hashing with random permutations (FMH). This is not usually the case for maps with 3 or less inliers with FMS. Finally the re-ranking procedure can filter out such an unsuccessful example.

Table 4 summarizes similar results on the *Holidays* and *Oxford*. Without distractors, FMH ranks slightly higher on *Holidays* but is outperformed by WGC on *Oxford*. On the contrary, FMH clearly outperforms all other methods on both datasets in the presence of the 50K distractors of *European Cities*, with or without re-ranking. The effect appears more evident on *Oxford*, possibly because of the same type of urban scenes in the distractor dataset.

Our score for BoW on *Oxford* dataset (0.372) is not directly comparable to the best score (0.618) achieved in [19], which is using a specific codebook generated from the query dataset. It is rather directly comparable to the score in [20] (0.403) where the codebook is generated from another dataset, as in our case. However more losses are induced by

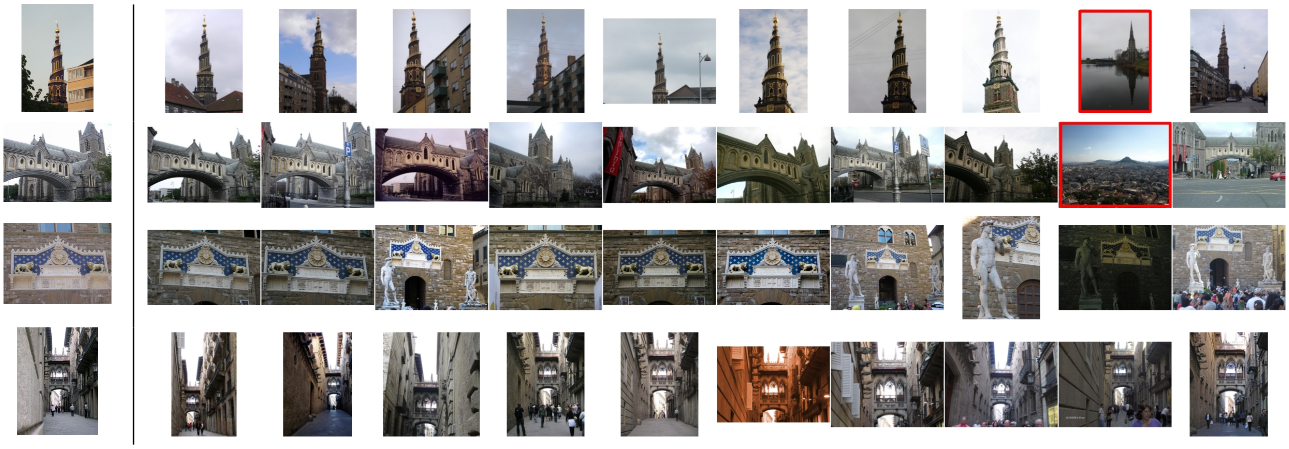

Figure 10: Sample queries and ranked retrieved images from *European Cities* dataset with 50K distractors using FMH without re-ranking. False images are depicted in a red bounding box.

Figure 11: Sample queries and ranked retrieved images from *European Cities* dataset with 50K distractors using BOW without re-ranking. False images are depicted in a red bounding box.

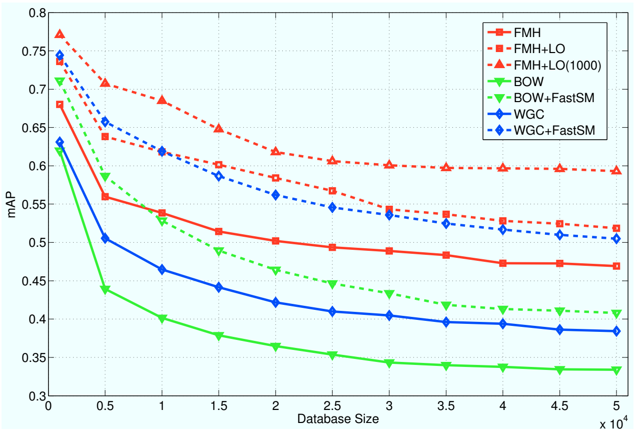

Figure 9: Mean average precision for varying database sizes on the *European Cities* dataset for BoW, WGC and FMH, with and without re-ranking.

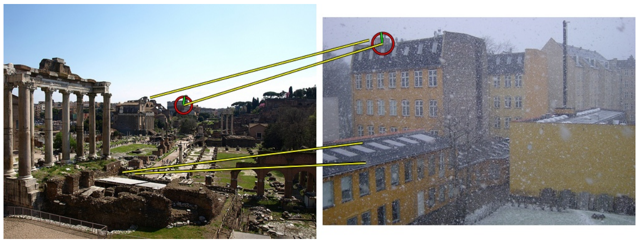

Figure 12: Example of unsuccessful matching with FMS. Origins shown in red circles with scale and rotation of the feature. Inliers are shown in yellow lines.

Dataset	Holidays		Oxford	
Method	1.4K	51.4K	5K	55K
BOW	0.583	0.492	0.372	0.329
WGC	0.591	0.510	**0.375**	0.333
FMH	**0.610**	**0.542**	0.362	**0.362**
BOW+FastSM	0.622	0.537	0.421	0.356
WGC+FastSM	0.626	0.542	**0.436**	0.388
FMH+LO(100)	**0.639**	0.556	0.422	0.391
FMH+LO(1000)	-	**0.571**	0.431	**0.410**

Table 4: Mean average precision for *INRIA Holidays* and *Oxford Buildings* datasets, with and without the distractors. FastSM is performed on the 100 top-ranked results. LO on both 100 and 1000 top-ranked results. Outperforming method shown in boldface.

the fact that we keep 1000 features at maximum from each image. Our scores for BoW and WGC on *Holidays* dataset are comparable to the ones in [13], (0.572) and (0.611) respectively, where a generic codebook is used as well.

Retrieval from our inverted index may incur losses for three reasons: feature misalignment, spatial quantization and hashing. Performing all queries on *European Cities* including distractors, we have quantified each as a percentage of the ground truth images. Feature misalignment accounts for the 8% not retrieved at all from the index, on average. Another 10% is then lost due to spatial quantization. Finally, hashing is responsible for another 3%.

8. DISCUSSION

To our knowledge, the present work is the first to integrate appearance and global geometry in sub-linear image indexing, while being invariant to affine transformations and robust to occlusion. We consider our experiments successful because we make spatial matching work at large scale, and demonstrate how this keeps precision almost unaffected under a significant amount of distractors.

We have found precision to be mostly limited by the very assumption that makes geometry indexing feasible: that a single feature correspondence is enough for image alignment. We see it as a challenge for future feature detectors to achieve better alignment score. We have developed our methodology for affine transformations, and this is because state of the art feature detectors are affine covariant. Extending *e.g.* to homography would be straightforward, should such features mature.

We find the feature map representation the most important contribution of this work. We foresee a new research direction in applying this concept to problems like large scale object recognition and detection, where geometric consistency and invariance are as crucial as in retrieval. More can be found at our project homepage[4].

9. ACKNOWLEDGMENTS

This work was supported by the European Commission under contract FP7-215453 WeKnowIt.

10. REFERENCES

[1] H. Bay, T. Tuytelaars, and L. Van Gool. SURF: Speeded up robust features. In *European Conference Computer Vision*. Springer, 2006.

[2] S. Belongie, J. Malik, and J. Puzicha. Shape context: A new descriptor for shape matching and object recognition. In *Neural Information Processing Systems*, volume 12, pages 831–827, 2000.

[3] A. Broder. Identifying and filtering near-duplicate documents. In *Symposium on Combinatorial Pattern Matching*, page 1. Springer Verlag, 2000.

[4] M. Charikar. Similarity estimation techniques from rounding algorithms. In *ACM Symposium on Theory of Computing*, pages 380–388. ACM New York, NY, USA, 2002.

[5] O. Chum and J. Matas. Geometric hashing with local affine frames. In *International Conference Computer Vision and Pattern Recognition*, volume 1, pages 879–884, 2006.

[6] O. Chum, J. Matas, and J. Kittler. Locally optimized RANSAC. In *DAGM Symposium on Pattern Recognition*, page 236. Springer Verlag, 2003.

[7] O. Chum, M. Perdoch, and J. Matas. Geometric min-hashing: Finding a (thick) needle in a haystack. In *International Conference on Computer Vision and Pattern Recognition*, 2009.

[8] A. Cohen. Maximum likelihood estimation in the weibull distribution based on complete and on censored samples. *Technometrics*, 7(4):579–588, 1965.

[9] W. Dong, Z. Wang, M. Charikar, and K. Li. Efficiently matching sets of features with random histograms. In *ACM International Conference on Multimedia*, pages 179–188, New York, NY, USA, 2008. ACM.

[10] I. L. Dryden and K. V. Mardia. *Statistical Shape Analysis*. WileyBlackwell, July 1998.

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

[12] K. Grauman and T. Darrell. Pyramid match hashing: Sub-linear time indexing over partial correspondences. In *Computer Vision and Pattern Recognition*, pages 1–8, 2007.

[13] H. Jegou, M. Douze, and C. Schmid. Improving bag-of-features for large scale image search. *International Journal of Computer Vision*, pages 1–21, 2010.

[14] H. Jegou, M. Douze, C. Schmid, and P. Perez. Aggregating local descriptors into a compact image representation. In *Computer Vision and Pattern Recognition*, 2010.

[15] K. Köser, C. Beder, and R. Koch. Conjugate rotation: Parameterization and estimation from an affine feature correspondence. In *Computer Vision and Pattern Recognition*, 2008.

[16] Y. Lamdan and H. Wolfson. Geometric hashing: A general and efficient model-based recognition scheme. In *International Conference on Computer Vision*, pages 238–249, 1988.

[17] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. Gool. A comparison of affine region detectors. *International Journal of Computer Vision*, 65(1):43–72, 2005.

[18] M. Perdoch, O. Chum, and J. Matas. Efficient representation of local geometry for large scale object retrieval. In *Computer Vision and Pattern Recognition*, 2009.

[19] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *Computer Vision Pattern Recognition*, 2007.

[20] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Lost in quantization: Improving particular object retrieval in large scale image databases. In *Computer Vision and Pattern Recognition*, 2008.

[21] F. Rothganger, S. Lazebnik, C. Schmid, and J. Ponce. 3d object modeling and recognition using local affine-invariant image descriptors and multi-view spatial constraints. *International Journal of Computer Vision*, 66(3):231–259, 2006.

[22] Y. Rubner, C. Tomasi, and L. Guibas. The earth mover's distance as a metric for image retrieval. *International Journal of Computer Vision*, 40(2):99–121, 2000.

[23] J. Sivic and A. Zisserman. Video google: A text retrieval approach to object matching in videos. In *International Conference Computer Vision*, volume 2, pages 1470–1477, 2003.

[4] http://image.ntua.gr/iva/research/feature_map_hashing