SymCity: Feature Selection by Symmetry for Large Scale Image Retrieval

Giorgos Tolias, Yannis Kalantidis and Yannis Avrithis
National Technical University of Athens
Iroon Polytexneiou 9, Zografou, Greece
{gtolias,ykalant,iavr}@image.ntua.gr

ABSTRACT

Many problems, including feature selection, vocabulary learning, location and landmark recognition, structure from motion and 3d reconstruction, rely on a learning process that involves wide-baseline matching on *multiple views* of the same object or scene. In practical large scale image retrieval applications however, most images depict unique views where this idea does not apply. We exploit self-similaries, symmetries and repeating patterns to select features within a *single image*. We achieve the same performance compared to the full feature set with only a small fraction of its index size on a dataset of *unique views* of buildings or urban scenes, in the presence of one million distractors of similar nature. Our best solution is linear in the number of correspondences, with practical running times of just a few 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, selection process*; I.4.10 [**Image Processing and Computer Vision**]: Image Representation

General Terms

Algorithms and Experimentation

Keywords

feature selection, symmetry detection, self-similarity, indexing, image retrieval

1. INTRODUCTION

The *bag-of-words* (BoW) representation on local features and descriptors, along with geometry verification, has been very successful at particular object retrieval [29][25]. At large scale, the current bottleneck appears to be the *memory footprint* of the index rather precision or speed, and it becomes even more significant in recent attempts to *geometry indexing* [12]. One solution is to abandon BoW in favor of more global representations like *Fisher kernels* [24][13], but these are not compatible with *geometry verification* [25][23], so they cannot reach the same levels of precision yet.

So far, *feature selection* is probably the only practical alternative. An increasingly popular idea is to select features from *multiple views*, identified by a baseline retrieval system [32]. Assuming similar images are indeed present in the dataset at hand, this idea has been successful at other tasks as well, including vocabulary learning [10][20], location and landmark recognition [26][9] and 3d reconstruction [1]. In practice however, most images are *unique*, in the sense that they depict a unique view of an object or scene in the dataset and there is nothing to compare to. *Or is there?*

We consider exactly such unique images in this work, using the very same idea: feature selection by *self-similarity*. We develop two *self-matching* alternatives inspired by state-of-the-art spatial matching methods: *fast spatial matching* (FSM) introduced by Philbin *et al*. [25], and *Hough pyramid matching* (HPM), introduced by Tolias and Avrithis [31]. We apply them between each image and either itself or its reflection, where tentative feature correspondences are found in the descriptor space, without quantization. In ef-

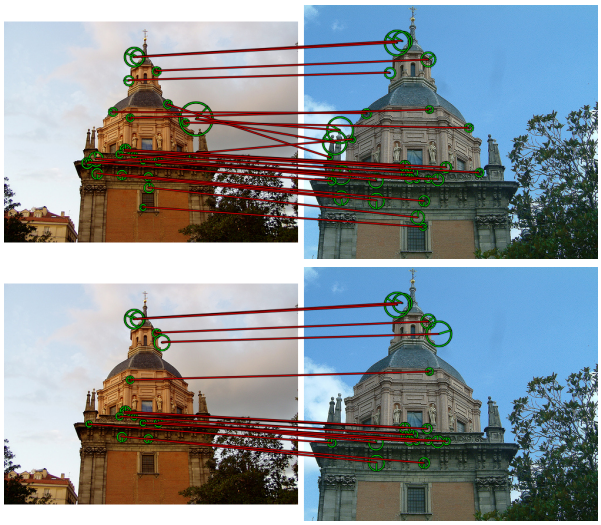

Figure 1: Tentative correspondences using all features (top), and selected features (bottom). Feature selection only applied to the image on the right.

MM'12, October 29–November 2, 2012, Nara, Japan.

fect, we detect *repeating patterns* or *local symmetries* and index only the participating features. The underlying motivation is that *features repeating within a single image are quite likely to repeat across different views as well*, hence are good candidates for a more compact image representation that does not sacrifice performance.

Though a limiting choice at first sight, we do achieve a generic method that is successful at least at urban image collections, where symmetries are inherent. But symmetries and self-similarities are not limited to man-made structures like buildings and natural creations like leaves and butterflies: when looking at different scales, *local* symmetries are everywhere. For instance, Bagon *et al*. [3] develop a generic method of *segmentation by composition*, while Shechtman and Irani [28] build a generic *self-similarity descriptor*; both methods operate at pixel level.

In the example of Figure 1, the image on the left (right) is considered a query (database) image, so feature selection is only applied on the right one. There are plenty of local symmetries, giving enough correspondences for matching even after selection, which also appear 'cleaner' in terms of outliers. Less than 20% of index space is needed after selection for this particular image.

On a dataset of one million images, we achieve the same performance with the full feature set, requiring only a small fraction of its memory. Alternatively, we outperform criteria like *feature scale* or *strength*, with the same amount of memory. Since unique images have only been treated as distractors so far, we also build a new *annotated dataset* with a large number of small groups of images depicting urban scenery; one image of each group is in the dataset and rest are only used as queries.

After a study of related work in section 2, we develop our main contribution on unique-view feature selection in section 3. We then present our experiments in section 4 and discuss our result and future directions in section 5.

2. RELATED WORK

Feature selection has recently become a popular way of reducing index space for image retrieval. This is typically an application-dependent task, and two of the earliest approaches have been applied to *location recognition*. Given a dense street-view geo-tagged database, Schindler *et al*. [26] select *informative* features, *i.e*. features occurring mostly in images of some specific location. Similarly, Li and Kosecka [17] obtain an information content probability for each feature with respect to location identification. Both methods rank image features accordingly and keep only a specific subset or percentage of features per image.

Knopp *et al*. [15] densely compute a *local confusion score* for image regions of a geo-tagged database by a sliding window scheme and remove features inside regions with high score. Gammeter *et al*. [9] start from image clusters extracted by geographical proximity and visual similarity, and then use feature matching statistics to estimate a bounding box around the *foreground object* of each image. Although they only index features inside this box, most are still kept. Avrithis *et al*. [2] further build a *scene map* from selected features of aligned views, but the result is a boost in recall rather than a considerable reduction in index size.

All previous models are supervised, making use of location information. Turcot and Lowe [32] on the other hand have introduced an *unsupervised* approach to select *useful* features. The latter are features appearing as inliers during *spatial verification*, when issuing each database image as a query to a baseline retrieval system. Naikal *et al*. [22] avoid spatial matching or structure from motion and, using a training set of images per category, apply sparse PCA on the covariance matrix of the bag-of-words histograms. However, they work with vocabularies of size only up to 10^3, making the solution unsuitable for large scale image retrieval.

Still, all models mentioned so far, including [32], require *multiple views* of the same object or scene. For *unique* images, Turcot and Lowe [32] keep the *largest-scale* features in the index, which is equivalent to reducing image resolution. In section 4 we show that this approach fails in the presence of a large scale distractor dataset. In this work we rather detect *self-similarities*, *repeating patterns* and *symmetries* to select features in a *single image*. To our knowledge, this is the first approach of this kind.

Earlier approaches on *symmetry detection* consider the entire image as a signal and look for one or more axes of symmetry *globally* [30][14]. More recent approaches work on *local features* instead. Tuytelaars *et al*. [33] begin with *invariant neighborhoods* and apply cascaded Hough transform to detect collinear intersections, a solution of high complexity. Cornelius *et al*. [7] also use Hough voting but do not require image descriptors; they detect *local affine frames* (LAFs) from an image and its reflection, match them using their local geometry, and allow each matching LAF pair to vote for a symmetry axis. Similarly, Loy and Eklundh [19] detect symmetric local feature *constellations* via Hough voting, but feature matching is based again on descriptors.

Lazebnik *et al*. [16] *expand* local feature configurations towards geometrically invariant *semi-local* parts of 3d objects. To limit the exponential number of hypotheses, they apply strong geometric and appearance-based consistency constraints on the seed configurations. Still, the complexity remains quadratic in the number of correspondences. We show that *local symmetry detection does not amount to more than any spatial matching scheme*, deriving our two solutions by such methods, one being quadratic [25], and the other *linear* [31] in the number of correspondences.

Structured and *repetitive pattern detection* has recently been applied to image retrieval, but not really for feature selection. Schindler *et al*. [27] detect periodic 2d patterns, matching them to a 3d database of textured *facades*. Doubek *et al*. [8] retrieve buildings by matching only lattice or line repetitive patterns, representing *tiles* by specifically tuned intensity and color descriptors. Both methods lack in generality. Jegou *et al*. [11] rather detect repeated visual elements in a single image to lower the weight of such features in bag-of-words scoring. Although taking into account this *intra-image burstiness*, they still keep all features.

3. FEATURE SELECTION

We introduce here two alternative unsupervised methods for feature selection, both focusing on geometrically consistent feature groups within a single image, roughly representing *repeating patterns* or *local symmetries*. The two methods are developed in sections 3.2 and 3.4. Under both methods, each image is matched to itself as well as its reflection, handling direct and opposite transformations, respectively. The two processes, referred to as *direct* and *flipped* matching, are presented in sections 3.1 and 3.3, respectively. Section 3.1 begins with our general image representation.

The entire method is based on the observation that symmetry or repeating pattern detection in a single image does not differ much from spatial matching between two images. In fact, we only see two issues requiring attention:

1. Spatial matching usually (but not always) assumes *one-to-one correspondence* between features of the two images. This is exactly why constraints like the *ratio test* [18] are usually employed when matching descriptors. This is not the case here because a pattern may be repeating more than twice.

2. Seeking robustness against 'outliers', a *single transformation model* is usually assumed, *e.g.* similarity or homography. This is not the rule, as *e.g.* HPM [31] can find multiple transformations. But for every symmetry or repeating pattern we do need a different transformation: HPM is more appropriate in this sense, but we also extend FSM [25] in this direction.

3.1 Representation

We assume that an image is represented by a set of local features X. Each local feature $x \in X$ is associated with a D-dimensional *descriptor* $d(x) \in \mathbb{R}^D$, encoding *local appearance*, which is then quantized to *visual word* $w(x) \in W$, where W is a given visual *vocabulary*. Assuming features are scale and rotation covariant, it is also associated with position $p(x) \in \mathbb{R}^2$ on the image plane, log-scale $\sigma(x) \in \mathbb{R}$ and orientation $\theta(p) \in (-\pi, \pi]$. All this information is conveniently represented in vector

$$g(x) = [p(x)^{\mathrm{T}} \ \sigma(x) \ \theta(x)]^{\mathrm{T}} \tag{1}$$

in $\mathbb{R}^4$, encoding *local geometry*. Log-scale is employed so that relative scales are expressed as differences rather than ratios. Orientation is alternatively represented by orthogonal matrix $R(x) \in \mathbb{R}^{2\times 2}$ with $\det R(x) = 1$. Finally, position is alternatively represented in *homogeneous coordinates* by vector $\mathbf{p}(x)$ in projective space $\mathbb{P}^2(\mathbb{R})$.

A feature *correspondence* within image X is a pair of features $c \in X^2$. Given correspondence $c = (x, y)$, it is possible to define a similarity transformation that aligns features $x, y \in X$ in the image plane. It turns out [31] that this transformation can also be represented by vector

$$g(x, y) = g(c) = [p(c)^{\mathrm{T}} \ \sigma(c) \ \theta(c)]^{\mathrm{T}} \tag{2}$$

in $\mathbb{R}^4$, where $\sigma(c) = \sigma(y) - \sigma(x)$, $\theta(c) = \theta(y) - \theta(x)$ are the relative scale and orientation respectively, and $p(c) = p(y) - M(c)p(x)$ is the relative position (translation), with $M(c) = \sigma(c)R(c)$ and $R(c) = R(y)R(x)^{-1}$.

It is now is possible to use norm $\|g(x, y)\|$ to measure the *geometric proximity* of features x, y. It is understood here that relative orientation $\theta(y)-\theta(x)$ is first taken to its principal value in $(-\pi, \pi]$. Because feature detectors often respond with multiple overlapping features around highly distinctive regions, we need this proximity measure to exclude nearby features from our search for symmetries. In particular, we say (x, y) is a *valid* pair and write $v(x, y)$ if and only if $\|g(x, y)\| \geq \rho$, with *response threshold* $\rho > 0$. Then, given image X, we define its set of *valid* correspondences

$$C_v(X) = \{(x, y) \in X^2 : v(x, y)\}. \tag{3}$$

This set also excludes *trivial* correspondences (x, x) of a feature x with itself, since $\|g(x, x)\| = 0$.

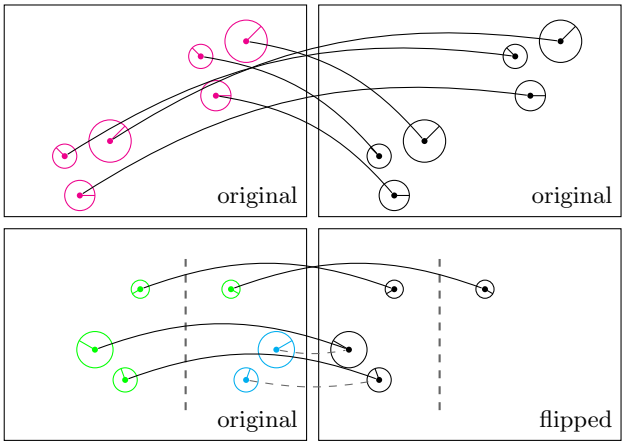

Figure 2: (Top) repeating pattern found by direct matching; magenta: *direct* selection. (Bottom) symmetry found by flipped matching; green: *flipped* selection, cyan: *back-projected* selection. (Left) original image. (Right) image to match; all features in black. Continuous line: tentative correspondences; dashed: back-projection + symmetry axes.

Now, feature correspondences are actually based on appearance. The simplest way is that of the *bag-of-words* model: two features are in correspondence simply when assigned the same *visual word*, which is a vector-quantized version of their descriptor. Depending on the method used for vocabulary construction, quantization may incur significant information loss [5], and since feature selection is an off-line process, we choose to work directly in the descriptor space. In fact, since features are not assigned a visual word before selection, it is also more efficient to seek neighbors among the features of X rather than within an entire vocabulary of typical size 10^6.

In particular, we define the *appearance dissimilarity* $d(x, y)$ of features $x, y \in X$ as their distance $\|d(x) - d(y)\|$ in the descriptor space $\mathbb{R}^D$. We say that features x, y are *similar* if $d(x, y) \leq \delta$, with *similarity threshold* $\delta > 0$. Now, given feature $x \in X$, let its *neighborhood* $N(x)$ be the set of similar features that is restricted to the set $\mathcal{N}_X^k(x)$ of its k-nearest neighbors in X according to dissimilarity d,

$$N(x) = \{y \in X : y \in \mathcal{N}_X^k(x) \wedge d(x, y) \leq \delta\}. \tag{4}$$

We use nearest neighbors to limit the number of correspondences, making the subsequent matching process more efficient; in practice, an *approximate* nearest neighbor scheme is used. Then, the set of *appearance-* or *descriptor-based* correspondences of image X contains all pairs of $x \in X$ with their neighbors,

$$C_d(X) = \{(x, y) \in X^2 : y \in N(x)\}, \tag{5}$$

Observe that $C_d(X)$ is a relation that is not symmetric in general. More important, contrary to typical image matching applications, we do not impose any *mapping constraint* towards *one-to-one* correspondence, like the *ratio test* [18]: we are seeking repeating patterns and a pattern may be repeating more than twice. Definitions (4), (5) allow up to k repetitions.

Finally, the set of *tentative* correspondences of X contains pairs of similar features that are also valid,

$$C_t(X) = C_d(X) \cap C_v(X). \tag{6}$$

All correspondence definitions so far refer to *direct* matching, and will be modified accordingly for flipped matching in section 3.3. Direct matching is illustrated in Figure 2(top), showing a pattern of three features repeating twice while undergoing a translation. Observe that tentative correspondences, being valid, exclude ones between a feature and itself in the two images. The two groups of three correspondences are essentially the same, being of the form (x, y) and (y, x). However, correspondences are not always symmetric. The pattern could repeat three times or more, while all similarity transformations are allowed.

3.2 Spatial self-matching

In order to detect local symmetries or repeating patterns, we need stronger evidence than just a set of independent correspondences, each based on the similarity of a single pair of features. We follow two different approaches, both inspired by existing methods for *spatial matching* between two different images, which however we apply to *self-matching* within a single image. Our first approach seeks *groups* of geometrically verified correspondences, called *inliers*; it is inspired by *fast spatial matching* (FSM) [25].

Given image X, each correspondence $c = (x, y) \in C_t(X)$ gives rise to a similarity transformation represented by vector $g(c)$ defined in (2). This transformation has four degrees of freedom and may as well be represented by a matrix $t(c) \in \mathbb{R}^{3\times3}$, with

$$t(x, y) = t(c) = \begin{bmatrix} M(c) & p(c) \\ \mathbf{0}^{\mathrm{T}} & 1 \end{bmatrix}. \tag{7}$$

This alternative formulation is useful when representing position in homogeneous coordinates. Then, given transformation $h = t(c) = t(x, y)$, the position $\mathbf{p}(z) \in \mathbb{R}^3$ of $z \in X$ is transformed to $h\mathbf{p}(z)$, the latter standing for a matrix-vector product.

FSM is a RANSAC-like process. Given image X and its set of tentative correspondences $C = C_t(X)$, each $c \in C$ defines a transformation *hypothesis* $h = t(c)$ that is verified by looking for *inliers* $I_C(h)$ among all correspondences $(x, y) \in C$, with

$$I_C(h) = \{(x, y) \in C : \|\mathbf{p}(y) - h\mathbf{p}(x)\| < \epsilon\} \tag{8}$$

for $h \in \mathbb{R}^{3\times3}$. The inlier set relies on *inlier threshold* $\epsilon > 0$, given in pixels. Among all hypotheses $t(C) = \{t(c) : c \in C\}$, FSM then seeks the hypothesis $h \in t(C)$ with the highest *inlier support* $|I_C(h)|$. Here is where we differentiate: we seek the best hypothesis *per individual inlier*.

In particular, for each correspondence $c = (x, y) \in C$, we define its set of *associated hypotheses* $H_C(c) = H_C(x, y) \subseteq t(C)$ that align c as an inlier,

$$H_C(x, y) = \{h \in t(C) : \|\mathbf{p}(y) - h\mathbf{p}(x)\| < \epsilon\}. \tag{9}$$

We can now define the *inlier strength* $\alpha(c)$ of correspondence c as the largest inlier support $|I_C(h)|$ over all its associated hypotheses $h \in H_C(c)$,

$$\alpha_C(c) = \max\{|I_C(h)| : h \in H_C(c)\}. \tag{10}$$

The entire self-matching process is summarized in Algorithm 1, which we will refer to as *spatial self-matching* (SSM). The original algorithm [25] is quadratic in the number of correspondences, since all correspondences are considered as inliers to all hypotheses. To speed up the process, we skip hypotheses arising from correspondences that have already been counted as inliers for previous hypotheses (line 6). We have observed that this does not affect feature selection in practice. The process is now quadratic only in the worst case, *i.e.* when no inliers are found at all, but in practice we get significant computational savings.

Algorithm 1: Spatial self-matching (SSM)

```
1  procedure α ← SSM(C, t; τ_α)
   input     : correspondences C, transformations t
   parameter: inlier threshold τ_α
   output    : inlier strengths α
2  for c ∈ C do                                  ▷ initialize
3      inlier(c) ← FALSE                         ▷ mark as outlier
4      α(c) ← 0                                  ▷ zero strength
5  for c ∈ C do                                  ▷ for all hypotheses
6      if inlier(c) then continue                ▷ skip hypothesis?
7      h ← t(c)                                  ▷ current hypothesis
8      I ← I_C(h)                                ▷ current inliers (8)
9      if |I| < τ_α then continue                ▷ verified hypothesis?
10     for c' ∈ I do                             ▷ for all inliers
11         inlier(c') ← TRUE                     ▷ mark as inlier
12         α(c') ← max(α(c'), |I|)               ▷ update strength
13 return α                                      ▷ inlier strengths
```

Once all inlier strengths have been computed, the set of spatially *verified* correspondences $\alpha(C) \subseteq C$ is

$$\alpha(C) = \{c \in C : \alpha_C(c) \geq \tau_\alpha\}, \tag{11}$$

with *selection threshold* $\tau_\alpha > 0$. Finally, given image X with tentative correspondences $C = C_t(X)$, we *select* those features $x \in X$ that are participating in some verified correspondence in $\alpha(C)$,

$$\alpha_d(X) = \pi^1(\alpha(C)) \cup \pi^2(\alpha(C)), \tag{12}$$

where, for $i = 1, 2$, $\pi^i(S)$ is the i-th *projection* of binary relation $S \subseteq X_1 \times X_2$, collecting the i-th element of all its pairs,

$$\pi^i(S) = \{x^i \in X_i : (x^1, x^2) \in S\}. \tag{13}$$

We call $\alpha_d(X)$ the *direct* selection of features in X.

3.3 Flipped matching

So far, we have only considered *direct* similarity transformations, that is, hypotheses h with $\det h > 0$. How about *opposite* transformations with $\det h < 0$, like *reflections*? In fact, once the image is reflected, the patch of each local feature is reflected as well, and its descriptor is no longer the same, unless the patch is symmetric itself. So reflecting the local geometry (1) is not enough: we actually need to reflect the entire image and extract a new set of features and descriptors.

Any opposite transformation will do, and we choose *horizontal flipping*. Let Y be the local feature set extracted from the flipped image. We assume each feature $y \in Y$ has a flipped, *back-projected* counterpart y'. This is the projection of y on the original image with

$$g(y') = [w - p^1(y) \;\; p^2(y) \;\; \sigma(y) \;\; \pi - \theta(y)]^{\mathrm{T}}, \tag{14}$$

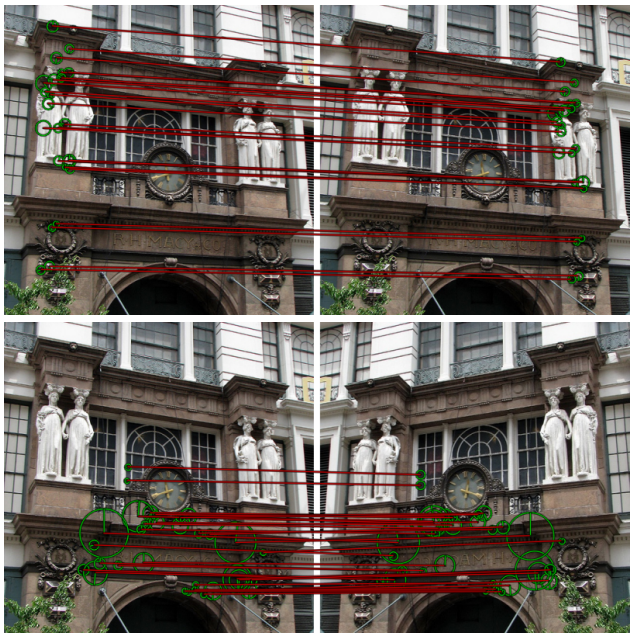

Figure 3: Top: sample group of inliers found by SSM on original image, capturing a repeating pattern. Bottom: sample group found by SSM between original and flipped images, capturing a symmetry.

where w is the image width and $p^i(y), i = 1, 2$ are the coordinates of y on the image plane. No descriptor is available for those features in the original image before selection.

Correspondences are formed exactly as in self-matching, but are now defined between features $x \in X, y \in Y$ of the original and flipped image respectively. The sets of *valid*, *appearance-based* and *tentative* correspondences (3), (5), (6) are modified respectively as

$$C_v(X,Y) = \{(x,y) \in X \times Y : v(x,y')\}, \quad (15)$$
$$C_d(X,Y) = \{(x,y) \in X \times Y : y \in N(x)\}, \quad (16)$$
$$C_t(X,Y) = C_d(X,Y) \cap C_v(X,Y). \quad (17)$$

Observe that valid pairs refer to the same image, so we use the back-projected feature y' instead of y in (15).

Given the set $C_f = C_t(X,Y)$ of tentative correspondences between X and Y, the SSM process of section 3.2 remains identical. Let $\alpha(C_f)$ be the resulting set of spatially verified correspondences. The second set of features we select, the *flipped* selection, contains those features of X that participate in a verified correspondence in $\alpha(C_f)$ and are nowhere near any feature in $\alpha_d(X)$ that is already selected,

$$\alpha_f(X) = \pi^1(\alpha(C_f)) \setminus_v \alpha_d(X). \quad (18)$$

By $A \setminus_v B$ we denote those features of A that are *valid with respect to* B,

$$A \setminus_v B = \{a \in A : v(a,b) \text{ for all } b \in B\}. \quad (19)$$

The third set of features, the *back-projected* selection, contains those features of Y that participate in a verified correspondence. We actually use their back-projected counterpart in this case, again ignoring ones that are near selected features in $\alpha_d(X) \cup \alpha_f(X)$,

$$\alpha'_f(X) = [\pi^2(\alpha(C_f))]' \setminus_v (\alpha_d(X) \cup \alpha_f(X)), \quad (20)$$

where $A' = \{y' : y \in A\}$ denotes the back-projection of feature set $A \subseteq Y$. It is now time to extract descriptors from the original image for those selected, back-projected features. Finally, the *complete* set of selected features contains the direct, flipped and back-projected selections,

$$\alpha(X) = \alpha_d(X) \cup \alpha_f(X) \cup \alpha'_f(X). \quad (21)$$

We will just say that $\alpha(X)$ is the set of *selected* features for X. They are the only features to be assigned a visual word and indexed for retrieval.

Figure 4: Initial (left) and selected (right) features by SSM. Original, flipped and back-projected selections shown in red, green and blue respectively.

Flipped matching is illustrated in Figure 2(bottom), showing a pattern of six features that is symmetric with a vertical axis of symmetry. Four of the features (in green) are detected on the original image, giving the *flipped* selection, and four (in black) on the flipped image; the two groups have two features in common, detected in both images. The two features that are only detected on the flipped image give the *back-projected* selection (in cyan) on the original image. Apart from this exception, we are showing for each image only the features that are detected and participating in some correspondence of the chosen pattern or group.

Figure 3 illustrates SSM matching between an image X and itself as well as its flipped counterpart Y, detecting a feature group corresponding to a repeating pattern and a symmetry, respectively, on a real image. There are a lot more detected feature groups not shown here. For the same image, Figure 4 depicts all available original features along with the three different sets of selections made by SSM, in particular the *direct* (α_d), *flipped* (α_f) and *back-projected* (α'_f) selections.

3.4 Relaxed spatial self-matching

Given a set of tentative correspondences C, the spatial self-matching process of section 3.2 is based on FSM [25] and is quadratic in the number of correspondences, $|C|$, in the worst case. On the other hand, *Hough pyramid matching* (HPM) [31] is a recent, *relaxed* spatial matching method, which is linear in $|C|$ and is shown to outperform FSM in spatial re-ranking for image retrieval. HPM is not only faster by not requiring inlier counting, but also free of any threshold defining what an inlier is, like ϵ in (8) or (9). Further, it assigns a *strength* value to each individual correspondence,

Algorithm 2: Hough pyramid self-matching (HPSM)

1 **procedure** $\beta \leftarrow \text{HPSM}(C, L)$
  input : correspondences C, levels L
  output: strengths β
2 **begin**
3 $B \leftarrow$PARTITION(L) ▷ partition space in L levels
4 **for** $c \in C$ **do** $\beta(c) \leftarrow 0$ ▷ initialize strengths
5 HPSM-REC$(\beta, C, L-1, B)$ ▷ recurse at top
6 **return** $\beta / \max(\beta)$ ▷ normalize

7 **procedure** HPSM-REC(β, C, ℓ, B)
  in/out : strengths β
  input : correspondences C, level ℓ, partition map B
8 **begin**
9 **if** $\ell < 0$ **then return**
10 **for** $b \in B_\ell$ **do** $F(b) \leftarrow \emptyset$ ▷ initialize histogram
11 **for** $c \in C$ **do** ▷ populate histogram
12 $F(q_\ell(c)) \leftarrow F(q_\ell(c)) \cup c$ ▷ ... by quantizing
13 **for** $b \in B_\ell$ **do**
14 $F \leftarrow F(b)$ ▷ correspondences in b
15 **if** $|F| < 2$ **then continue** ▷ exclude singles
16 HPSM-REC$(\beta, F, \ell-1, B)$ ▷ recurse down
17 **if** $\ell = L-1$ **then** $m \leftarrow 2$ **else** $m \leftarrow 1$
18 **for** $c \in F$ **do** ▷ update strengths in b
19 $\beta(c) \leftarrow \beta(c) + 2^{-\ell} m |F|$ ▷ ... as in (22)

reflecting whether it is geometrically consistent with others. This is accomplished without ever enumerating all pairs of correspondences.

We consider HPM as the basis for a faster self-matching alternative for feature selection. Instead of an overall matching score between the image and itself (which would be pointless), we rather use the *individual strength* of each correspondence. In this sense, HPM is easier than FSM to apply to self-matching, since individual strengths are inherent in its matching process. However, there are still a couple of deviations from [31].

Given a set of tentative correspondences C, either a subset of X^2 (self-matching) or $X \times Y$ (flipped matching), each correspondence $c = (x, y) \in C$ gives rise to a 4-dof transformation, which we now represent by *transformation parameter* vector $g(c) \in \mathbb{R}^4$ given by (2). Seeing these vectors as points in a 4-dimensional *transformation space*, the problem is now to detect regions of high density in this space, otherwise known as *mode seeking*. Because the dimensionality is low, a *pyramid* matching scheme is an ideal way to linearize the process by avoiding pairwise interactions. We briefly discuss this scheme here.

An L-level *hierarchical partition* of the transformation space is constructed as a sequence of partitions $B_0, \ldots, B_{L-1}$. B_0 is at the finest (bottom) level, while B_{L-1} is at the coarsest (top) level and has a single bin. Each bin in B_ℓ is split into 2^4 bins in $B_{\ell+1}$. A *histogram pyramid* is then constructed by distributing correspondences to bins at all levels of the hierarchy. Given any bin b, let $F(b) = \{c \in C : g(c) \in b\}$ be the set of correspondences with parameter vector falling in b, and $f(b) = |F(b)|$ its count, representing the bin *frequency* in the histogram.

Correspondences falling into the same bin are *grouped*, and we expect to find the most geometrically consistent

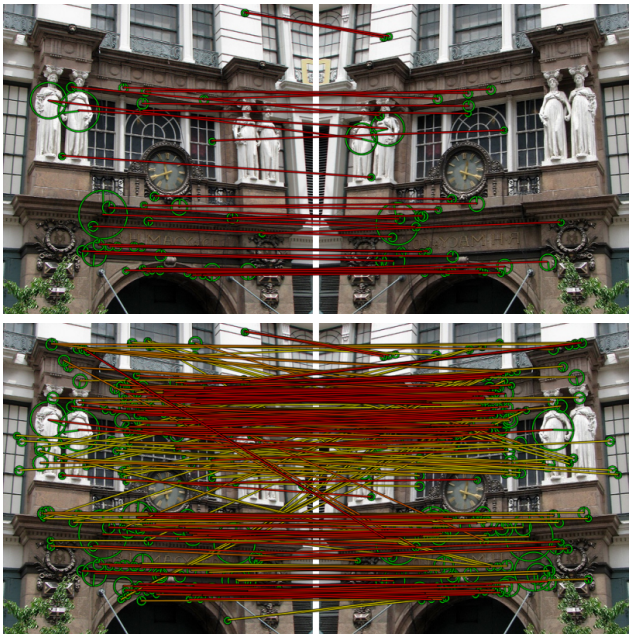

Figure 5: Flipped matching with HPSM at $L = 5$ levels. (Top) correspondences in a single bin at level 0, revealing a symmetric feature group. (Bottom) all tentative correspondences, with red (yellow) being the strongest (weakest).

groups in the lower levels of the hierarchy, with larger groups considered to be more consistent. A single correspondence gives no matching evidence and cannot form a group, so the *group size* of a bin is defined as $s(b) = [f(b) - 1]_+ = \max(0, f(b) - 1)$. Let $b_0 \subseteq \ldots \subseteq b_\ell$ be the sequence of bins containing a correspondence c up to level ℓ. The *strength up to level* ℓ of each correspondence $c \in C$, reflecting the above considerations, is given by

$$\beta_\ell(c) = s(b_0) + \sum_{i=1}^{\ell} 2^{-i} \{s(b_i) - s(b_{i-1})\}. \tag{22}$$

The total strength of c collected up to the top level $L-1$ is then simply $\beta_{L-1}(c)$, which we convert to a *relative strength*, normalizing by the maximum strength over all correspondences in C:

$$\beta(c) = \beta_{L-1}(c) / \max_{a \in C} \beta_{L-1}(a). \tag{23}$$

The latter normalization scheme is a deviation from [31], giving the strongest correspondences a chance to participate in feature selection, even if they are not strong enough in absolute value. Another deviation is that we *do not impose an one-to-one mapping*, since this would contradict detection of repeating patterns within a single image, as discussed in section 3.1. The matching process is thus considerably simplified compared to [31], as summarized in Algorithm 2. We will refer to the latter as *Hough pyramid self-matching* (HPSM). Mapping $q_\ell : C \to B_\ell$ appearing in line 12 is used to *quantize* a correspondence to a bin of level ℓ. Histogram representation is sparse and the algorithm remains linear in $|C|$. As in [31], strength update in line 19 is equivalent to the definition given by (22).

Figure 6: Sample images, along with selected features found by HPSM at $L = 5$ levels, colored as in Figure 4.

Now, replacing α by β, we can use HPSM to find the *verified* correspondences $\beta(C) \subseteq C$, modifying (10) as

$$\beta(C) = \{c \in C : \beta(c) \geq \tau_\beta\}, \tag{24}$$

with *selection threshold* $\tau_\beta \in [0, 1]$. Similarly, let $\beta(C_f)$ be the verified correspondences after flipped matching with HPSM. Then the *direct* (β_d), *flipped* (β_f), *back-projected* (β'_f) and *complete* (β) feature selections follow exactly as in sections 3.2, 3.3, simply by substituting β for α in (12), (18), (20), (21) respectively. We will refer to $\beta(X)$ just as the *selected* set of features for image X in this case.

Figure 5 illustrates flipped matching with HPSM, for the same image as in Figures 3,4. In this case, there are no groups of correspondences that are inliers to a single transformation. One may consider the set of correspondences in a single bin instead; the lower the level, the tighter the fit to one transformation. Each tentative correspondence is scored independently according to whether it is geometrically consistent with others. The strongest correspondences are the ones to be selected.

Figure 6 provides further examples of several images, depicting the complete feature selection made by HPSM, including the *direct* (β_d), *flipped* (β_f) and *back-projected* (β'_f) selections. There are plenty of symmetries found despite perspective projection, with most selected features on the main foreground object or surface.

4. EXPERIMENTS

In this section we explore the behavior of SSM and HPSM under different parameter settings and compare their performance and memory usage against the *full feature set* representation on large scale experiments. We also compare the proposed methods against other feature selection approaches for single images based on options available from the feature detector, in particular the feature scale as in [32], and the feature strength.

4.1 Datasets

We use two datasets, namely the new *SymCity* dataset and *World Cities*[1] [31]. *SymCity* is a dataset of 953 annotated images from *Flickr*, split in 299 small groups depicting urban scenes. From each group, a single image is inserted in the database, while all other 654 images are only used as queries. For each query image we then wish to retrieve only a unique database image, and the precision measurement actually depends only on its ranked position. The distractor set of *World Cities* consists of 2M images from different cities; we use the first one million as distractors in our experiments, which we shall refer to as the *distractor* set.

SymCity has been built by a semi-automatic process. We have followed the method of Avrithis *et al.* [2] to first create geo-clusters and then visual clusters of images from 10 cities. Keeping only clusters of up to 4 images, we have selected through visual inspection a set of clusters depicting indeed the same object, building or scene. The 10 cities considered are different from the ones of the distractor set, ensuring that depicted scenes do not appear in both sets. We shall refer to *SymCity* as the *annotated* set; sample images are shown in Figure 7.

4.2 Protocol

We extract SURF [4] features and descriptors for each image and assign them visual words from a generic 100K vocabulary using FLANN [21] and keeping only one nearest neighbor under Euclidean distance. The vocabulary is constructed using approximate k-means (AKM) [25] on an independent set of 15K images depicting urban scenes.

[1] `http://image.ntua.gr/iva/datasets/wc/`

Figure 7: Sample images from the *SymCity* dataset.

In the feature selection process, descriptor matching for tentative correspondences is performed with approximate nearest neighbor search, again using FLANN [21]. Upon failure of symmetry detection, we add more features based on detector information. In particular, SSM fails when it yields no verified hypotheses, while HPSM when it selects less than τ_α features, which is again the minimum number of inliers that would be required to verify a hypothesis. In either case, we select a percentage $\lambda = 15\%$ of the features with the highest detector strength.

For retrieval, we build an independent *inverted file* for each selected feature set. For each query image we then retrieve its visual words from the inverted file and rank them by dot product on ℓ^2-normalized BoW histograms and *tf-idf* weighting [29].

We measure retrieval performance by *mean Average Precision* (mAP) over all queries. In the particular case where there is only a single image to be retrieved for each query, the *average precision* is just the inverse of its rank in the retrieved list of images. We measure *index size* by the number of entries in the inverted index per image, which is the number of unique visual words per image. For each selection method, we define *memory ratio* as the ratio of its index size to that required by the *full feature set*, where all features are kept. The index size of the latter is 635 on average; in particular, an image has 710 features and 635 unique visual words on average before selection.

Running time refers to the entire process of feature selection for one image including both direct and flipped matching for each method, but excluding the time needed to generate feature correspondences and extract features and descriptors in the flipped image, as well as descriptors for back-projected features. We use our own C++ implementations for all methods evaluated and measure times on a 2GHz quad core processor. In all experiments we perform selection only on the 299 images of the *SymCity* dataset ground truth and index *all* features of the distractor images. This is preferred in order to allow a fair comparison to baseline methods, since all methods are now tested using the same amount of distractor features.

4.3 Tuning

We first fix *similarity threshold* (4) $\delta = 0.1$ and SSM *inlier threshold* (8) $\epsilon = 7$ pixels, which we choose after thorough subjective evaluation by visual inspection of correspondences and selected features. The former is specifically chosen for SURF descriptors; the latter is not needed by HPSM. We also fix $L = 5$ levels for HPSM, as in [31].

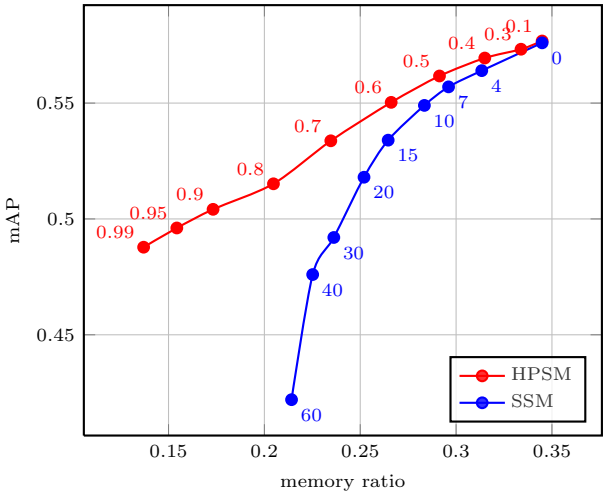

Figure 8: mAP performance of SSM and HPSM for varying selection thresholds τ_α and τ_β respectively versus memory ratio in the presence of 100K distractors, with $k = 3$ nearest neighbors. Selection threshold is shown with text labels near markers.

With these settings, we perform further parameter tuning on the *SymCity* dataset, using a subset of 100K distractors from *World Cities*. In Figure 8 we show mAP and memory ratio measurements for SSM and HPSM with varying *selection thresholds* τ_α (11) and τ_β (24) respectively. Both methods perform equally for high memory ratio, selecting in fact all features participating in tentative correspondences. However, as we become more selective, SSM collapses while HPSM continues to degrade smoothly.

Figure 9 measures average running time for both methods, for varying number k of *nearest neighbors* used for tentative correspondences in (4). The running time for SSM also depends on selection threshold τ_α. A lower threshold will allow more hypotheses to be skipped, degrading performance. The higher the threshold is, the closer the running time is to that of FSM [25]. HPSM is 5 to 15 times faster than SSM on average, so given its higher performance as well, it has a clear advantage. We choose to continue further tuning and large scale experiments with HPSM only.

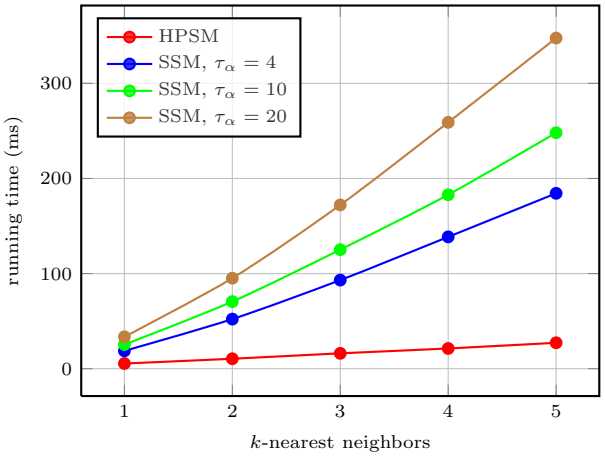

Figure 9: Average running time (ms) for HPSM and SSM versus k-nearest neighbors for tentative correspondences, where τ_α is varying for SSM.

k	1	2	**3**	4	5
$\tau_\beta = \mathbf{0.4}$	0.545	0.566	**0.569**	0.566	0.568
$\tau_\beta = 0.6$	0.522	0.538	0.550	0.551	0.547
$\tau_\beta = 0.8$	0.484	0.511	0.515	0.524	0.529

Table 1: HPSM mAP performance versus k-nearest neighbors for varying τ_β in the presence of 100K distractors. Chosen parameters and mAP shown in boldface.

Table 1 compares HPSM performance for varying number k of nearest neighbors. Performance almost stabilizes or even drops with more than 3 neighbors. We therefore choose $k = 3$ nearest neighbors for our remaining experiments. We also choose $\tau_\beta = 0.4$ as default, although there is a further experiment under varying τ_β. The average running time of HPSM for this setting is 16.2ms. Although feature selection is an off-line process, its speed is critical when indexing millions of images. HPSM is particularly efficient, with a running time that is negligible compared *e.g.* to feature detection and visual word assignment.

4.4 Comparisons

Working at larger scale, with up to the full 1M distractor set of *World Cities*, we compare our selection method to the *full feature set*. We also compare to three alternative, simple selection criteria for single images, where in particular a fixed number of features n is selected, having the highest *detector strength*, largest *feature scale* σ as in [32], or *uniformly at random* among the full feature set.

Figure 10 shows mAP performance under varying number of distractor images. While HPSM selection is, not quite unexpectedly, outperformed at small scale, it reaches the full feature set performance at large scale. This may be interpreted as *keeping a limited amount of information, which degrades quality when matching* e.g. *a single image pair, but which withstands severe distractor noise*. An observation of the two curve slopes at 10^6 distractors suggests that HPSM

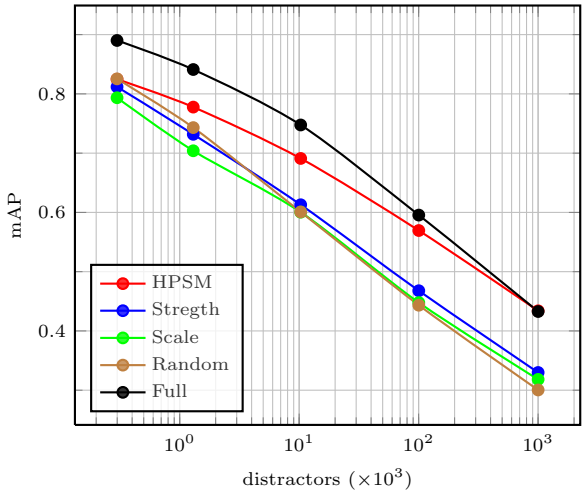

Figure 10: Mean average precision comparison versus number of distractors, with $\tau_\beta = 0.4$ for HPSM and a fixed number of features $n = 300$ for the remaining selection criteria.

will actually *outperform* the full feature set at even larger scale.

This experiment is carried out with the default selection threshold $\tau_\beta = 0.4$, which yields an average memory ratio of 0.31 on the annotated set, corresponding to 284 selected features per image on average. To allow a fair comparison to the alternative selection criteria, we select a fixed number of $n = 300$ features per image, so that the average memory ratio is roughly the same. HPSM outperforms all three criteria: interestingly, its performance is near that of the three criteria at small scale, but gradually shifts towards that of the full feature set at large scale.

A higher selection threshold τ_β would make the process more selective and decrease memory ratio, at the expense of lower mAP as well. This is a way to *trade off* index size for retrieval quality. Figure 11 shows this trade-off on the full 1M distractor set, revealing that a less selective HPSM can even *outperform* the full feature set on mAP for a memory ratio around 35%. On the other hand, its mAP is in general well above that of the three alternative criteria for the same memory ratio, with an overhead of 13% on average.

5. DISCUSSION

It is quite unexpected that symmetry or self-similarity is a generic feature selection criterion that is able to reach or possibly *outperform the full feature set* at large scale. On the other hand, it is also unexpected that more local criteria like feature scale or strength do not in fact give much benefit over random selection. Besides this being the first work to apply symmetry for feature selection, we have shown that *symmetry detection is not much different in nature from spatial matching*, so given any advance in one problem it is straightforward to apply it to the other. In particular, HPSM is the first method that is *linear in the number of correspondences*, being extremely fast in practice.

Among future directions, *soft assignment* [6] and *geometry verification* [31] are probably the first one may combine

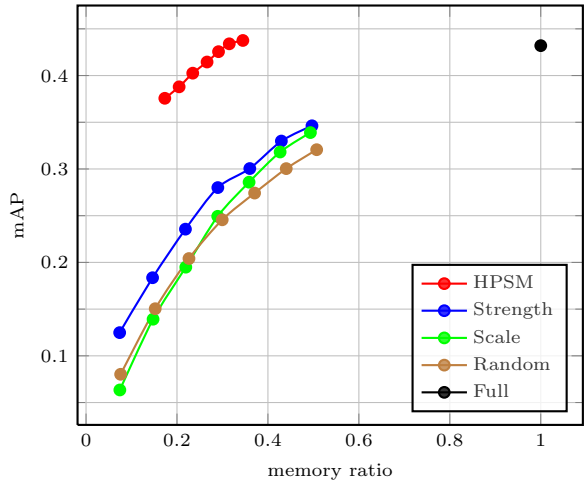

Figure 11: Mean average precision comparison against memory ratio for varying τ_β, n in the presence of 1M distractors.

in the retrieval process. It would be more interesting to extract *feature tracks* from our single image correspondences and employ them to find *visual synonyms* for vocabulary learning, as [20] does from multiple views. All approaches are expected to increase performance without any cost on the index size.

6. REFERENCES

[1] S. Agarwal, N. Snavely, I. Simon, S. M. Seitz, and R. Szeliski. Building Rome in a day. In *ICCV*, 2009.

[2] Y. Avrithis, Y. Kalantidis, G. Tolias, and E. Spyrou. Retrieving landmark and non-landmark images from community photo collections. In *ACM Multimedia*, 2010.

[3] S. Bagon, O. Boiman, and M. Irani. What is a good image segment? A unified approach to segment extraction. In *ECCV*, 2008.

[4] H. Bay, T. Tuytelaars, and L. Van Gool. SURF: Speeded up robust features. In *ECCV*, 2006.

[5] O. Boiman, E. Shechtman, and M. Irani. In defense of nearest-neighbor based image classification. In *CVPR*, 2008.

[6] 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*, 2007.

[7] H. Cornelius, M. Perdoch, J. Matas, and G. Loy. Efficient symmetry detection using local affine frames. In *ECIA*, 2007.

[8] P. Doubek, J. Matas, M. Perdoch, and O. Chum. Image matching and retrieval by repetitive patterns. In *ICPR*, 2010.

[9] S. Gammeter, L. Bossard, T. Quack, and L. V. Gool. I know what you did last summer: Object-level auto-annotation of holiday snaps. In *ICCV*, 2009.

[10] E. Gavves, C. G. M. Snoek, and A. W. M. Smeulders. Visual synonyms for landmark image retrieval. *CVIU*, 2012.

[11] H. Jegou, M. Douze, and C. Schmid. On the burstiness of visual elements. In *CVPR*, 2009.

[12] H. Jegou, M. Douze, and C. Schmid. Improving bag-of-features for large scale image search. *IJCV*, 87(3):316–336, 2010.

[13] H. Jegou, M. Douze, C. Schmid, and P. Perez. Aggregating local descriptors into a compact image representation. In *CVPR*, 2010.

[14] Y. Keller and Y. Shkolnisky. An algebraic approach to symmetry detection. *ICPR*, pages 186–189, 2004.

[15] J. Knopp, J. Sivic, and T. Pajdla. Avoiding confusing features in place recognition. In *ECCV*, 2010.

[16] S. Lazebnik, C. Schmid, and J. Ponce. Semi-local affine parts for object recognition. In *BMVC*, 2004.

[17] F. Li and J. Kosecka. Probabilistic location recognition using reduced feature set. In *ICRA*, 2006.

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

[19] G. Loy and J.-O. Eklundh. Detecting symmetry and symmetric constellations of features. In *ECCV*, 2006.

[20] A. Mikulik, M. Perdoch, O. Chum, and J. Matas. Learning a fine vocabulary. In *ECCV*, 2010.

[21] M. Muja and D. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. In *ICCV*, 2009.

[22] N. Naikal, A. Yang, and S. Shankar Sastry. Informative feature selection for object recognition via sparse pca. In *ICCV*, 2011.

[23] M. Perdoch, O. Chum, and J. Matas. Efficient representation of local geometry for large scale object retrieval. In *CVPR*, 2009.

[24] F. Perronnin, Y. Liu, J. Sanchez, and H. Poirier. Large-scale image retrieval with compressed Fisher vectors. In *CVPR*, 2010.

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

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

[27] G. Schindler, P. Krishnamurthy, R. Lublinerman, Y. Liu, and F. Dellaert. Detecting and matching repeated patterns for automatic geo-tagging in urban environments. In *CVPR*, 2008.

[28] E. Shechtman and M. Irani. Matching local self-similarities across images and videos. In *CVPR*, 2007.

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

[30] C. Sun and D. Si. Fast reflectional symmetry detection using orientation histograms. *Real Time Imaging*, 5(1):63–74, 1999.

[31] G. Tolias and Y. Avrithis. Speeded-up, relaxed spatial matching. In *ICCV*, 2011.

[32] P. Turcot and D. Lowe. Better matching with fewer features: the selection of useful features in large database recognition problems. In *ICCV*, 2009.

[33] T. Tuytelaars, A. Turina, and L. Van Gool. Noncombinatorial detection of regular repetitions under perspective skew. *PAMI*, 2003.