PROBING THE LOCAL-FEATURE SPACE OF INTEREST POINTS

Wei-Ting Lee, Hwann-Tzong Chen

Department of Computer Science
National Tsing Hua University, Taiwan

ABSTRACT

We describe an empirical study on the feature space of interest points for natural images. Although local features have been widely used in image analysis as building blocks of various algorithms, there is still a lack of study on the space of local features, in particular the distributions of local features extracted from the locations of interest points. Based on an approximate neighborhood probing scheme, we devise an efficient representation that is not only helpful in visualizing the distributions of local features, but also effective in modeling the neighborhood structures of local features for fast matching. We present various experimental results that provide useful insights into feature description and fast image matching.

Index Terms— Local Features, Image Matching

1. INTRODUCTION

Local features have been extensively used to represent images for various problems, such as image registration, scene modeling, and image retrieval. A typical process of extracting local features usually begins by detecting interest points in an image, and then the feature descriptors that quantitatively characterize the local regions are computed at the detected locations. Gradient information and histogram-based descriptors are quite often used to represent the detected local regions, and the dissimilarity between two feature descriptors may be easily measured using Euclidean distance. Lots of local feature detectors and local feature descriptors have been proposed during recent years, for example, *maximally stable extremal regions* (MSER) [1], *difference-of-Gaussian filter* (DoG) and *scale-invariant feature transform* (SIFT) [2], *affine invariant detector* [3], [4], and *histogram of oriented gradients* (HOG) [5]. Such local feature detectors/descriptors have been shown to be useful in matching two images and also in building 'visual words' [6] or 'codebooks' [7] for object categorization and scene analysis.

The effectiveness of interest-point detectors and local descriptors has been demonstrated through many applications. Nevertheless, there is still a lack of study on the feature space of local descriptors, except a few results like [8] and [9]. Vector quantization techniques have been popularly adopted to partition the feature space or to build a visual vocabulary of image features [6], [10], [11], [12]. Sivic and Zisserman [6] carry out k-means clustering to learn SIFT-based visual words for object retrieval. Ke *et al*. [11] apply locality-sensitive hashing to feature indexing and image retrieval. Grauman and Darrell [10] present an approximate bipartite-matching method to compute correspondences between two sets of feature vectors. In this paper, we seek to explore the space of the invariant local features that correspond to the interest points in natural images. We present an efficient representation which are useful in visualizing the distributions of invariant local descriptors and modeling their neighborhood structures. We highlight our results as follows:

1. We present an empirical analysis of the feature space of interest points detected in natural images. More specifically, we describe a way to visualize and explore the high-dimensional space constituted by the local-feature vectors that are extracted from the locations of interest points. Our results suggest that, for oriented-gradients-based local features such as SIFT descriptors, better descriptiveness is achieved by applying the descriptors to the image patches containing interest points. Applying the feature descriptors to random image patches does not yield as much distinctive information as to the interest-point patches.

2. Based on the efficient representation of the space of interest-point feature vectors, we perform an approximate method for the fast matching between two sets of interest points detected in two images. Unlike most previous approaches, the proposed method does not require additional online training or tree-construction steps. We show that the complexity of matching M points to N points can be reduced from $O(MN)$ to $O(M + N)$ using our matching method, while the recall and the error rates are adequately maintained.

2. EMPIRICAL ANALYSIS

Our aim is to explore the spatial structure represented by the invariant local features that are extracted from the image patches corresponding to interest points. We use *locality-sensitive hashing with 2-stable distributions* [13] to analyze

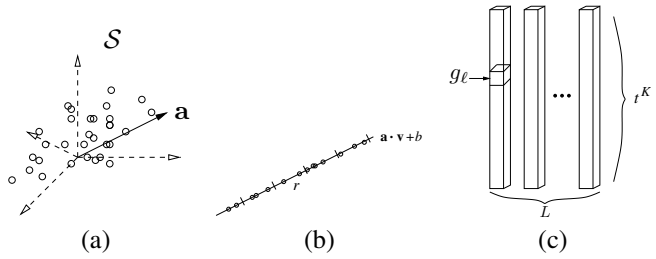


Fig. 1. (a) A hash function is given by a random vector $\mathbf{a}$ sampled from a Gaussian distribution over the high-dimensional feature space S. (b) The dot product $\mathbf{a} \cdot \mathbf{v}$ projects each vector to the real line. The real line is quantized into t equal-width segments of width r. (c) In this example L hash tables are created. Each hash table is equipped with K hash functions. Hence each hash table has t^K buckets, which are indexed by the function g_ℓ for the ℓth table.

the space of invariant local features. The 2-stable locality-sensitive hashing scheme provides an approximate nearest neighbor structure that is efficient for the searches under l_2 norm. Using 2-stable distribution for sketching high-dimensional vectors can be viewed as computing multi-dimensional histograms, which yield a good visualization of the distribution of feature vectors.

2.1. Locality-Sensitive Hashing based on 2-Stable Distributions

Locality-sensitive hashing (LSH) is a hashing scheme for fast computing approximate nearest neighbors in large databases. Here we consider the case of searching the nearest neighbor in the feature space S with a metric given by the l_2 norm. Given the parameters (r_1, r_2, p_1, p_2), we design a family of functions $\{h\}$, such that, for any $\mathbf{u}, \mathbf{v} \in S$, we have

- if $\|\mathbf{v} - \mathbf{u}\| \leq r_1$, then $\Pr[h(\mathbf{u}) = h(\mathbf{v})] \geq p_1$;
- if $\|\mathbf{v} - \mathbf{u}\| > r_2$, then $\Pr[h(\mathbf{u}) = h(\mathbf{v})] \leq p_2$.

To get a meaningful hash family, we require $p_1 > p_2$ and $r_1 < r_2$. Since a Gaussian distribution is 2-stable and we want to apply LSH to nearest neighbor search under l_2 norm, we use a locality-sensitive hashing scheme based on a Gaussian distribution. A hash function of the intended hash family is then given by $h_{\mathbf{a},b} = \lfloor \frac{\mathbf{a}\cdot\mathbf{v}+b}{r} \rfloor$, where $\mathbf{a}$ is a random vector sampled from a Gaussian distribution over the feature space S, and b is a real value chosen uniformly from the range $[0, r]$. The dot-product $\mathbf{a} \cdot \mathbf{v}$ projects each vector to the real line. We divide the real line into equal width segments of size r, and determine the hash values for vectors according to which segment they project onto, as illustrated in Figs. 1a and 1b. A hash function obtained via this process will preserve locality due to the stability property of Gaussian distributions [13].

We may choose the width r based on the minimum and maximum values of the dot-products, such that we get a fix number of segments on each projected real line, say, t segments. For convenience's sake, we label the segments by assigning a positive integer $i \in \{1, 2, \ldots, t\}$ to each segment according to their order on the dot-product real line. We then define a modified hash function $\hat{h}$ that maps a vector into a segment label $i \in \{1, 2, \ldots, t\}$. The output value of a modified hash function $\hat{h}$ can be easily derived from the value of the original hash function $h_{\mathbf{a},b}$.

We use the above process to build L hash tables, *i.e.*, L sets of hash functions, where each set contains K hash functions. These hash functions are denoted by $\{h_k^\ell | k = 1, \ldots, K\}$, for $\ell = 1, \ldots, L$. We define an index function $g_\ell(\hat{h}_1^\ell, \hat{h}_2^\ell, \ldots, \hat{h}_K^\ell; t)$ using the modified hash function $\hat{h}$ to get the indexing for the ℓth table

$$g_\ell = (\hat{h}_K^\ell - 1)t^{K-1} + (\hat{h}_{K-1}^\ell - 1)t^{K-2} + \cdots + (\hat{h}_1^\ell - 1)t^0 + 1\,. \quad (1)$$

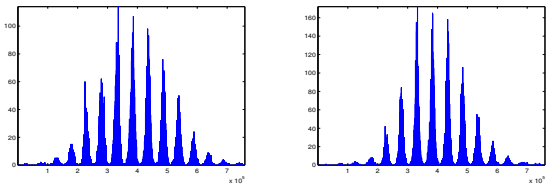
In sum, given a feature vector $\mathbf{v}$, we compute the hash functions $\{h_k^\ell(\mathbf{v})\}$ and thus get the modified hash values $\{\hat{h}_k^\ell(\mathbf{v})\}$. We then compute $g_\ell(\mathbf{v})$ for indexing to the bucket in the ℓth table.

2.2. Sketching the Feature Space

We use the Berkeley segmentation database [14] as the source of natural image patches. To collect image patches of interest points, we have tried the difference of Gaussians (DoG) detector [2] and the Hessian-affine detector [3]. From the Berkeley database we use each of the two detectors to find about $200,000$ interest points. The corresponding image patch for each interest point is then extracted, according to the position, scale, and orientation (or affine shape) indicated by the detectors. We represent the extracted image patches using the SIFT descriptor [2]. The SIFT descriptor transforms an image patch from the image space to the feature space. This descriptor computes oriented-gradient histograms over subregions inside an image patch, and then stacks and normalizes the histograms to produce a unit vector with 128 dimensions.

We create a hash table ($L = 1$) with five projections ($K = 5$) and 15 segments on each dot-product real line ($t = 15$). The total number of buckets is $15^5 = 759,375$. For the $200,000$ SIFT features of the interest points found by the DoG detector, we compute their indexes to the buckets, and view the indexing results as a histogram of the distribution of feature vectors. We repeat the process for interest points detected by the Hessian-affine detector.

The feature vector distributions of interest points detected by the DoG detector and the Hessian-affine detector are shown in Fig. 2. Consequently, the hashing results and the entropy values indicate that the descriptiveness of SIFT features for invariant interest points is fairly good, even though interest points occur rarely in natural images—usually a few thousands of interest points are detected among millions of possible locations, scales, and orientations.



Entropy = 4.2251 (a) DoG

Entropy = 4.0622 (b) Hessian-affine

Fig. 2. The feature vector distribution of interest points detected by (a) the DoG detector and (b) the Hessian-affine detector.

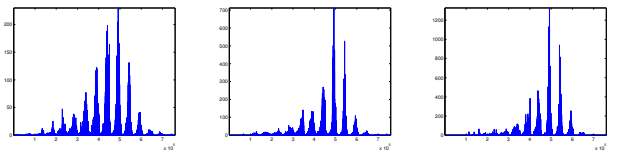
(a) Natural image patches (b) Noise patches

Fig. 3. (a) Some examples of randomly-sampled natural-image patches from the Berkeley segmentation database. (b) Randomly-generated noise patches.

For comparison, we apply the same hashing scheme to common image patches. We randomly sample a large number of image patches of different sizes from the Berkeley natural images. We collect three sets of image patches of size 16×16, 32×32, and 64×64 pixels, and each set consists of $200,000$ patches. Some examples are shown in Fig. 3a. The distributions of the SIFT feature vectors for these patches are depicted in Figs. 4a, 4b, and 4c. It can be observed that as the size of image patch increases, the distribution of SIFT features on common image patches becomes more peaked, and the entropy decreases, which implies less descriptiveness.

To further examine the property of the feature space represented by the SIFT descriptor, we generate three sets of uniformly-distributed noise patches of size 16×16, 32×32, and 64×64 pixels. Some sample patches are shown in Fig. 3b. Each set also contains $200,000$ patches. Figs. 5a, 5b, and 5c illustrate the distributions of the SIFT feature vectors for noise patches. The feature distributions are all highly peaked and have very low entropy, although the original noise patches are by nature of high entropy.

Observation: The combination of interest point detection and SIFT-like local feature description indeed yields a feature space that is most 'informative'. Applying the SIFT descriptor to randomly sampled patches might not be appropriate, particularly when the correct scale is unknown.

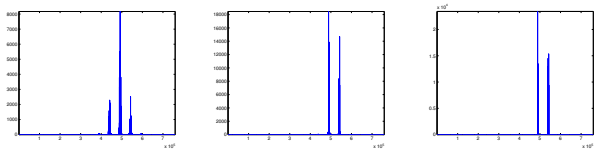


Entropy = 4.1074 (a) 16×16

Entropy = 3.8442 (b) 32×32

Entropy = 3.5174 (c) 64×64

Fig. 4. The feature vector distribution of randomly-sampled natural-image patches (Fig. 3a).

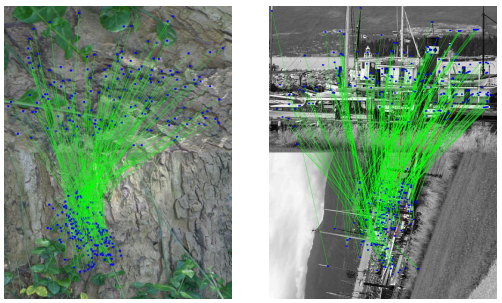


Entropy = 2.3806 (a) 16×16

Entropy = 1.4775 (b) 32×32

Entropy = 1.2404 (c) 64×64

Fig. 5. The feature vector distributions of noise patches, *e.g.*, those shown in Fig. 3b.



(a) (b)

Fig. 6. Examples of matching results using (a) '2-stable LSH + DoG detector + SIFT descriptor' and (b) '2-stable LSH + Hessian-affine detector + SIFT descriptor'. The test images are from [4].

3. FAST MATCHING

We have shown that the LSH scheme is useful to visualize the distributions in the high-dimensional feature space. We further use the efficient representation derived from LSH to achieve fast matchings between two sets of interest points. For evaluation, the data set of Mikolajczyk and Schimid [4] is employed. The data set contains 48 images of eight real scenes: Six images are taken for each scene to include different types of transformations such as rotation, scaling, viewpoint changes, illumination changes, blur, and compression. We create $L = 16$ hash tables to probe the 128-dimensional SIFT-feature space. Each table is equipped with five 2-stable projections, and the projected values are quantized into 15 segments, *i.e.*, $K = 5$ and $t = 15$. To match two sets of interest points, we compute the bucket index of each interest point based on its SIFT feature. With LSH, we do not need to compare all bipartite matchings between the two sets. We only need to compare the pairs of feature vectors that are mapped into the same bucket in each hash table. Since the collision rate is very low (usually less than 3 collisions per bucket), we achieve a very fast approximate-matching speed.

In the first experiment, we use the DoG detector to identify interest points. The image patches around the interest points are represented by SIFT features. The first of the six

Table 1. 2-stable LSH matching vs. exhaustive matching on the feature vectors obtained by 'DoG detector + SIFT descriptor'. The parameters 0.95 and 0.97 are thresholds of the dot-product values. Two feature vectors are considered to be a match if their dot-product is larger than the threshold.

Scene	$\frac{\text{LSH}_{0.95}}{\text{exhaustive}_{0.95}}$ recall	$\text{LSH}_{0.95}$ error rate	$\frac{\text{LSH}_{0.97}}{\text{exhaustive}_{0.95}}$ recall	$\text{LSH}_{0.97}$ error rate	$\text{exhaustive}_{0.95}$ error rate
bark	0.637	0	0.523	0	0
bikes	0.755	0.064	0.672	0.021	0.121
graf	0.661	0.109	0.529	0.069	0.152
leuven	0.752	0.139	0.659	0.096	0.229
trees	0.523	0.067	0.331	0.039	0.076
ubc	0.780	0.065	0.702	0.027	0.122
boat	0.652	0.250	0.523	0.114	0.358
wall	0.552	0.003	0.374	0.001	0.007

Table 2. 2-stable LSH matching vs. exhaustive matching on the feature vectors obtained by 'Hessian-affine detector + SIFT descriptor'. The parameter settings are the same as in Table 1.

Scene	$\frac{\text{LSH}_{0.95}}{\text{exhaustive}_{0.95}}$ recall	$\text{LSH}_{0.95}$ error rate	$\frac{\text{LSH}_{0.97}}{\text{exhaustive}_{0.95}}$ recall	$\text{LSH}_{0.97}$ error rate	$\text{exhaustive}_{0.95}$ error rate
bark	0.537	0.007	0.335	0	0.021
bikes	0.594	0.301	0.440	0.197	0.390
graf	0.538	0.469	0.366	0.239	0.567
leuven	0.612	0.124	0.465	0.061	0.186
trees	0.491	0.278	0.294	0.059	0.368
ubc	0.703	0.160	0.583	0.069	0.252
boat	0.562	0.106	0.397	0.036	0.159
wall	0.534	0.032	0.342	0.005	0.049

images of each scene is used as the reference image, and the remaining five images are used to find the correspondence points with respect to the first image. If the dot-product between the pair of feature vectors is larger than a threshold $\theta \in [0, 1]$, then these two feature vectors are considered to be from two correspondence points. For LSH, we use two threshold values of dot-product, $\theta = 0.95$ and $\theta = 0.97$, to determine whether a pair of feature vectors in the same bucket yields a match. The matching results are listed in Table 1. The recall rate is computed by the ratio of 'the number of correct correspondences found by LSH' to 'the number of correct correspondences found by exhaustive search with $\theta = 0.95$'. The error rate is defined as the ratio of 'the number of incorrect correspondences' to 'the total number of found correspondences'. The error rates of LSH are much lower than the error rates of exhaustive search, while the recall rates are generally higher than 0.5. In the second experiment we perform the same test on Hessian-affine interest points. Similar results are reported in Table 2. Some matching examples are shown in Fig. 6. Theoretically, LSH matching has a linear time complexity with respect to the number of points to be matched. In practice we find that matching by LSH is 2 to 15 times faster than matching by exhaustive search.

4. CONCLUSION

We have presented in this paper an empirical analysis, as well as a visualization technique, for the distributions of SIFT-feature vectors, using the approximate nearest-neighbor probing scheme derived from 2-stable locality-sensitive hashing. We have found that SIFT local features are particularly descriptive for image patches of interest points, while the SIFT features of random or noisy patches are less descriptive, particularly when the correct scale is unknown. We make use of the efficient representation of the SIFT-feature space, and present a fast feature-matching method for finding correspondences between two sets of interest points.

Acknowledgment. This research was supported in part by grants 98-EC-17-A-19-S2-0052 and NTHU 99N2511E1.

5. REFERENCES

[1] Jiri Matas, Ondrej Chum, Martin Urban, and Tomás Pajdla, "Robust wide baseline stereo from maximally stable extremal regions," in *BMVC*, 2002.

[2] David G. Lowe, "Distinctive image features from scale-invariant keypoints," *IJCV*, vol. 60, no. 2, pp. 91–110, 2004.

[3] Krystian Mikolajczyk and Cordelia Schmid, "Scale & affine invariant interest point detectors," *IJCV*, vol. 60, no. 1, pp. 63–86, 2004.

[4] Krystian Mikolajczyk and Cordelia Schmid, "A performance evaluation of local descriptors," *TPAMI*, vol. 27, no. 10, pp. 1615–1630, 2005.

[5] Navneet Dalal and Bill Triggs, "Histograms of oriented gradients for human detection," in *CVPR*, 2005, vol. 1, pp. 886–893.

[6] Josef Sivic and Andrew Zisserman, "Video google: A text retrieval approach to object matching in videos," in *ICCV*, 2003, pp. 1470–1477.

[7] B. Leibe and B. Schiele, "Interleaved object categorization and segmentation," in *BMVC*, 2003, pp. 759–768.

[8] Kent Shi and Song-Chun Zhu, "Mapping natural image patches by explicit and implicit manifolds," in *CVPR*, 2007.

[9] Tinne Tuytelaars and Cordelia Schmid, "Vector quantizing feature space with a regular lattice," in *ICCV*, 2007.

[10] Kristen Grauman and Trevor Darrell, "Approximate correspondences in high dimensions," in *NIPS*, 2006, pp. 505–512.

[11] Yan Ke, Rahul Sukthankar, and Larry Huston, "An efficient parts-based near-duplicate and sub-image retrieval system," in *ACM Multimedia*, 2004, pp. 869–876.

[12] David Nistér and Henrik Stewénius, "Scalable recognition with a vocabulary tree," in *CVPR*, 2006, vol. 2, pp. 2161–2168.

[13] Mayur Datar, Nicole Immorlica, Piotr Indyk, and Vahab S. Mirrokni, "Locality-sensitive hashing scheme based on p-stable distributions," in *SCG*, 2004, pp. 253–262.

[14] D. Martin, C. Fowlkes, D. Tal, and J. Malik, "A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics," in *ICCV*, 2001, vol. 2, pp. 416–423.