

# Learning Affine Robust Binary Codes Based on Locality Preserving Hash

Wei Zhang<sup>1,2</sup>, Ke Gao<sup>1</sup>, Dongming Zhang<sup>1</sup>, and Jintao Li<sup>1</sup>

<sup>1</sup> Advanced Computing Research Laboratory,  
Beijing Key Laboratory of Mobile Computing and Pervasive Device,  
Institute of Computing Technology,  
Chinese Academy of Sciences, 100190 Beijing, China

<sup>2</sup> University of Chinese Academy of Sciences, 100049 Beijing, China  
{zhangwei, kegao, dmzhang, jtli}@ict.ac.cn

**Abstract.** In large scale vision applications, high-dimensional descriptors extracted from image patches are in large quantities. Thus hashing methods that compact descriptors to binary codes have been proposed to achieve practical resource consumption. Among these methods, unsupervised hashing aims to preserve Euclidean distances, which do not correlate well with the similarity of image patches. Supervised hashing methods exploit labeled data to learn binary codes based on visual or semantic similarity, which are usually slow to train and consider global structure of data. When data lie on a sub-manifold, global structure can not reflect the inherent structure well and may lead to incompact codes. We propose locality preserving hash (LPH) to learn affine robust binary codes. LPH preserves local structure by embedding data into a sub-manifold, and performing binarization that minimize false classification ratio while keeping partition balanced. Experiments on two datasets show that LPH is easy to train and performs better than state-of-the-art methods with more compact binary codes.

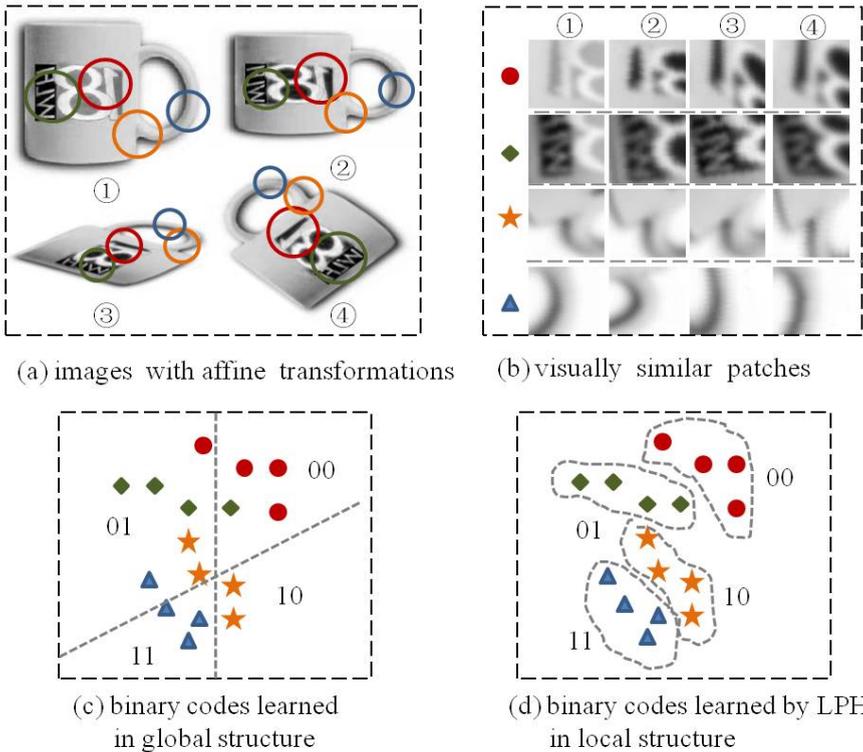
**Keywords:** Similarity Search, Binary Code Learning, Locality Preserving Hash, Locality Preserving Projection.

## 1 Introduction

Descriptor extracting is fundamental to computer vision applications such as image retrieval, copy detection, object recognition, etc. The descriptors are always high-dimensional and in large quantities. To allow for fast matching with practical memory consumption, there have been many recent attempts that compact descriptors to binary codes in Hamming space based on hashing techniques [1-11].

Compacting methods such as locality sensitive hashing (LSH) [1] and spectral hashing (SH) [2] are unsupervised and aim to preserve the Euclidean distances of original vectors, but do not take the visual similarity of patches into account. As can be seen from Fig.1, descriptors of visually similar patches are not always adjacent to each other in vector space. The Euclidean distance cannot reflect similarity relationships

properly in such situations. Christoph Sander et al. proposed supervised LDAHash [9], which performs linear discriminant analysis (LDA) to preserve similarity relationships of a training set. However, LDA considers the global structure of data and has little to do with the manifold or local structure [10]. When data lie on a sub-manifold, it is hard to handle the inherent structure well and may lead to incompact binary codes[10].



**Fig. 1.** An illustration of affine robust binary codes learned by locality preserving hash (LPH) in local structure (d), compared with binary codes learned in global structure (c). Visually similar patches (b) are detected and normalized from images with affine transformations (a) and consistent in visual contents and geometric information.

To overcome the drawbacks in existing methods mentioned above, we propose locality preserving hash (LPH), as illustrated in Fig.1. LPH generates affine robust and compact binary codes by preserving the local structure of data. Specifically, LPH has the following new characteristics:

(1)LPH embeds data into a sub-manifold by locality preserving projection (LPP) [12] with a patch similarity matrix. Locality preserving projection (LPP) is a linear algorithm and has shown its effectiveness in exploring the intrinsic manifold structure in face recognition. However, the original LPP is unsupervised. To exploit the training samples, we define a new similarity matrix following the supervised learning scheme [10].

The matrix is based on similarity of image patches, which are automatically labeled by mining stable patches in affine transformed images [13]. In this way, binary codes learned are affine robust. Supervised learning methods are usually prone to over fitting when labeled data are of small amount or noisy [5]. This can be avoided in our case, for training samples are automatically generated and labeled.

(2)After the embedding, a binarization process is performed which partitions the data space and represents data as binary codes. The traditional methods mainly focus on the partition balance of data space [2, 14-15] for efficiency. We propose a novel binarization method that minimizes false classification ratio while keeping partition balanced. Therefore the local structure is preserved as much as possible in the partitioned space while an efficient retrieval is achieved.

The rest of the paper is organized as follows: section 2 presents the related work. In section 3, we present our LPH by formulating the hashing problem and giving our solutions. In section 4, we introduce how to conduct off-line training and then evaluate our method on two datasets. Finally, we conclude the paper in section 5 and give research perspectives for the future of this work.

## 2 Related Work

In large-scale computer vision applications, high-dimensional descriptors are extracted in huge quantities to represent image contents, which challenge the limited computational and storage resources. There have been many attempts at compacting descriptors to allow for fast matching with practical memory consumption. One class is based on quantization such as bag of visual words [16] and product quantization [17]. However, these approaches are not sufficient to produce short codes without loss of performance. Another class is to compact descriptors into binary codes. These approaches are promising, for binary codes can retain the desired similarity information with much reduced memory. And various methods have been proposed [1-11] in recent years.

A simple idea is to embed data to a new data space using a matrix, whose elements are random variables sampled from Gaussian distributions as in LSH [1]. Since the distribution of high dimensional data in vision applications is far from uniform [18], binary codes generated by machine learning approaches [2-5, 9-11] are always more compact than random methods. Weiss et al. propose Spectral Hashing [2] that attempts to preserve the given similarities defined by Euclidean metric and form codes with each bit carrying more information, which achieves significant improvement over LSH. Transform coding [19] and integrated binary codes [8] utilize principle component analysis (PCA) to embed data which aim to maximally preserve the variance.

The methods mentioned above aim to preserve the neighborhood relationships in Euclidean space of descriptors. Unfortunately, the descriptors of visually similar image patches may be far away in the vector space and thus those methods are not sufficient to reflect the underlying relationships of such patches.

By contrast, many supervised methods take advantage of similarity information contained in training data, and thus can map similar descriptors closer and dissimilar

ones far apart. There are various forms of supervision to define similarity, such as learned Mahalanobis metric [20], Restricted Boltzmann Machines (RBMs) [4], and Binary Reconstruction Embeddings [3]. However, as discussed in [21], these methods are suffered from the difficulty of optimization and slow training mechanisms. LDAHash performs supervised optimization easily and tries to capture the global structure of data [9].

We propose LPH to learn affine robust binary codes by exploring the underlying manifold structure of training samples, with effective and efficient optimization. We define image patches as visually similar if they are detected from transformed images but consistent in visual contents and geometric information, as is shown in Fig.1. We aim to preserve the local structure of visual similar patches when compacting descriptors to binary codes. This is a relaxation of the similarity preserving problem, compared with methods [4, 11] that define patches with quite different visual contents as similar. Besides, our training method is simple and easy to implement.

### 3 Locality Preserving Hash

Compacting descriptors to binary codes is usually implemented by hashing, which include two key steps: (1) embed data to a new space represented by a projection matrix and (2) binarize the embedded or projected values with thresholds. The two steps boil down to the generation of projection matrix and selection of optimal thresholds. In this section, we formulate the problem and present our solution of the generation of projection matrix and the selection of binarization thresholds in LPH.

#### 3.1 Problem Formulation

Given a training set of descriptors containing groups of visually similar patches, the aim of LPH is to map descriptors into affine robust and compact binary codes, by preserving the local structure of similar patches. Formally, let  $X=\{x_1, \dots, x_n\}$  be a set of  $d$ -dimensional descriptors and  $h(x_i)$  be a hash function, we seek a binary feature vector  $y_i=[h_1(x_i), \dots, h_k(x_i)]$  that preserves the visual similarity relationships using a Hamming distance.

Suppose we want to get a  $k$ -bit code  $y_i$  of  $x_i$ , then  $k$  hash functions leading to  $k$  Hamming embeddings are needed. We use linear projection and threshold as the hash function, and then binary code is computed using the following equation:

$$y_i = \text{sgn}(Px_i^T + t) . \quad (1)$$

where  $P$  is a  $k*d$  matrix and  $t$  is a  $k*1$  vector.

The projection matrix  $P$  is the key of embedding and threshold  $t$  is the key of binarization. The existing methods of generating  $P$  and  $t$ , randomly or supervised, have been discussed in the previous section. Ours are discussed in the next section.

### 3.2 Projection Matrix Generation

Given a matrix  $W$  with weights characterizing the similarity of two image patches, we want to learn a  $P$  to preserve the local structure, which satisfies the following:

$$\text{minimize: } \sum_{ij} \|y'_i - y'_j\|^2 W_{ij} . \tag{2}$$

where  $y'$  is the data projected by  $P$ , i.e.  $y'=Px^T$  and  $W_{ij}$  is defined as follows:

$$W_{ij} = \begin{cases} 1, & \text{if image patches of } x_i \text{ and } x_j \text{ are visually similar} \\ 0, & \text{otherwise} \end{cases} . \tag{3}$$

To simplify the similarity computation between different points, we just set the weight equal to 1. The similarity of image patches is automatically obtained when collecting descriptors. And descriptors are collected by simulating affine transformations of images, extracting descriptors from each transformed image and mining stable ones that are consistent in visual contents and geometric information. The details can be found in [13]. Then we define stable descriptors as visually similar to construct  $W$ . The similarity learning method is the spirit of LPP and  $P$  can be obtained by solving a generalized Eigen value problem as in [12].

### 3.3 Optimal Thresholds Selection

The threshold selection is a separate problem from projection matrix generation. In this section,  $y'_i=P_i x_i^T$  denotes the  $i$ -th element of the projected  $x$ , and  $y=\text{sgn}(P_i x_i^T + t_i)$  will generate the  $i$ -th bit of the final code. Usually, thresholds are selected to partition data space in balance for efficiency considerations. Here, we consider both effectiveness and efficiency aspects of a partition. On one hand, we want to minimize the ratio of false partitions that divide in-class data into two parts. In this way, the local structure will be preserved as much as possible in the partitioned space. The false ratio ( $FR$ ) is also the probability that the value of a threshold  $t$  between the projected in-class data, which is given by:

$$FR(t) = \Pr(\min\{Y'\} < t < \max\{Y'\}) . \tag{4}$$

where  $Y'$  is projected data from a set of similar patches.

On the other hand, we want to generate a balanced partition for further efficient retrieval of binary codes. A balanced partition means that data are distributed uniformly in each sub-parts and indicates that less noises are accessed during searching. Since entropy can be used to measure the distribution of data, a balanced partition is achieved by maximizing the following partition entropy ( $PE$ ):

$$\begin{aligned} PE(t) &= - p_1 \log p_1 - p_2 \log p_2 , \quad p_1 = \Pr(y' \leq t) \\ & \quad p_2 = \Pr(y' > t) . \end{aligned} \tag{5}$$

where  $y'$  denotes any vectors  $x$  projected by  $P$ .

Suppose the value of  $\alpha$  and  $\beta$  represent the importance of the two sub functions  $FR$  and  $PE$ , then the final optimization problem is as follows:

$$\text{minimize: } \alpha FR(t) - \beta PE(t). \quad (6)$$

If  $\alpha=0$ , each sub-partition is of the same size and the most balanced partition is obtained. If  $\beta=0$ , data remains unpartitioned when  $FR(t)$  is maximized, which is meaningless. In the following experiments, we set  $\alpha$  and  $\beta$  equal to 1. We solve the problem approximately by computing  $FR(t)$  and  $PE(t)$  with different thresholds and select the threshold when the optimal trade-off is gained.

## 4 Experimental Results

We perform our experiments on two datasets: (1) image patches extracted from INRIA Holiday dataset [15] and (2) the Ukbench dataset [22] with a distractor dataset downloaded from Internet. For evaluation we compute the average recall of each query and then take the mean value over the set of queries. All experiments are run on a workstation with 16GB memory and 4 CPUs of 2.13GHz.

**Image Patches Extracted from Holiday Dataset** (*91,400 image patches, 500 queries*). We collect images of different scene types from Holiday dataset and simulate affine transformations like ASIFT [23]. Then descriptors are obtained by DoG detector and SIFT descriptor [24]. Visually similar patches are obtained by mining descriptors that are stable [13] in various transformations. We define a descriptor as stable if it appears in more than 5 transformed images with slightly changed geometric information. And severely changed descriptors in vector space are filtered to reduce the gap between feature space and vector space during learning. Then the patches of stable descriptors are collected as the database. The queries are randomly chosen from groups of visually similar patches and the correct retrieval results are the other ones of the corresponding groups.

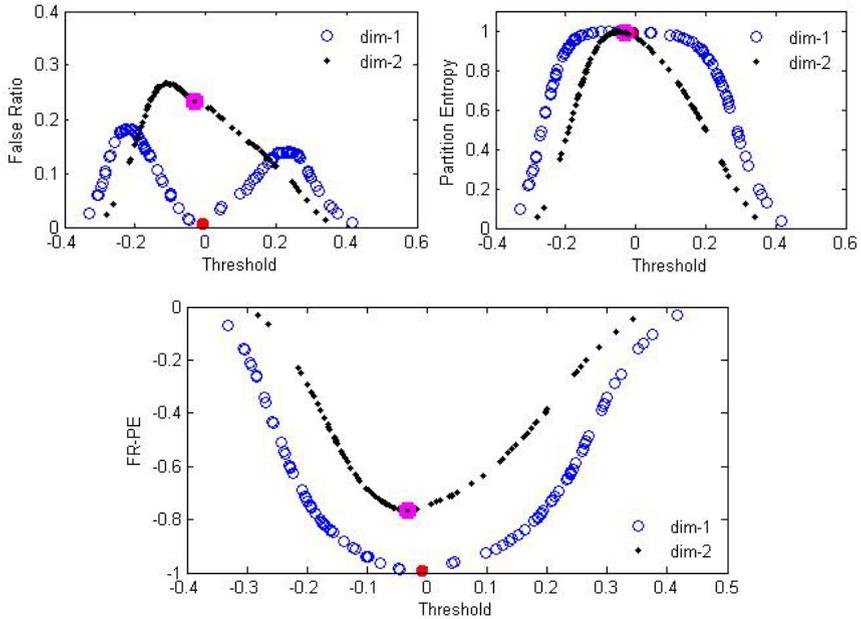
**Ukbench Dataset with Distractor Images** (*10,200 images, 10,000 distractor images, 7.8million descriptors*). We apply our method to object retrieval on Ukbench dataset. The dataset contains groups of 4 objects shot in different viewpoints and illuminations. To evaluate in a larger database, we also introduce distractor images downloaded from Internet with various objects. The descriptors are obtained by DoG detector and SIFT descriptor. There are 7.8 million descriptors in total. The queries are randomly chosen from 500 groups and the correct retrieval results are the 4 images of the corresponding groups.

### 4.1 Off-line Training

The training dataset contains 10,000 groups of patches randomly chosen from our first dataset. The training process includes two steps, i.e. projection matrix generation and threshold selection. When visual similarity relationships are available, the similarity

matrix  $W$  in (3) is constructed. The supervised LPP that generate the projection matrix is performed using Deng's source codes [25].

Then we select thresholds on each separate dimension after projecting data to the matrix. On each dimension of the projected data, we compute  $FR$  in (4) and  $PE$  in (5) with different thresholds. And then following (6), a threshold is selected when an optimal trade-off between false partitioned ratio and partition balance is achieved.



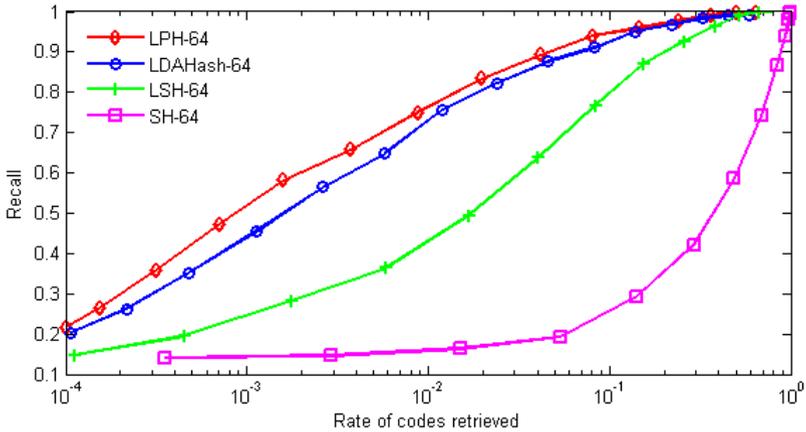
**Fig. 2.** Histograms of false ratio ( $FR$ ), partition entropy ( $PE$ ) and  $FR-PE$  in (6) computed with different threshold settings on two sample dimensions of projected data. The threshold is marked when optimal value is gained.

Fig.2 shows the selection of thresholds on separate dimensions. As can be seen, false ratio is minimized while partition entropy is maximized on the first dimension, which is the perfect case. This means that data are partitioned uniformly and only 0.65% data are false partitioned. The false ratio is higher on the second dimension for the partition balance is emphasized. If the false ratio is emphasized,  $\alpha$  in (6) should be enlarged. Besides, the false ratio can be compensated by exploring a Hamming ball during query process.

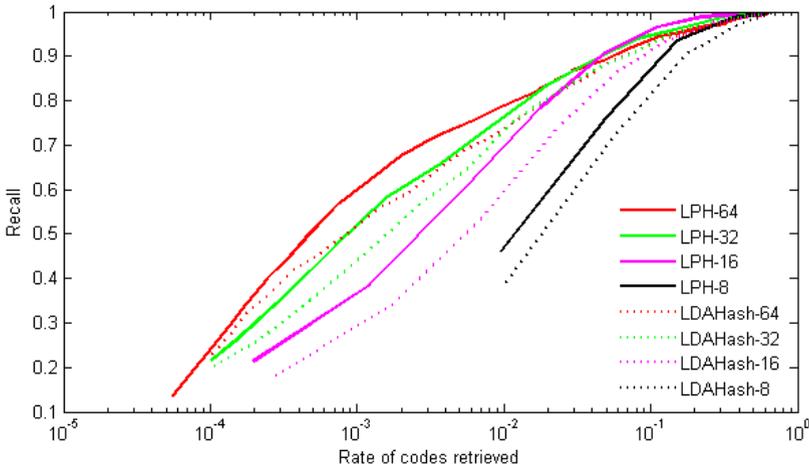
## 4.2 Image Patch Retrieval

In this section, our proposed LPH is evaluated on the image patch database, for which the ground truth of visually similar patches are available. The average recall as a function of the rate of codes retrieved is used as the evaluation metric for this retrieval task. A higher recall with a lower rate of codes retrieved indicates that more correct

codes are found with fewer noises returned. After the off-line training process, we compact database descriptors into binary codes. Then we adopt linear scan for codes retrieval. In addition, the efficiency can be improved further by using multi-index [26]. By setting Hamming radius of the search from 0 to 32, comparisons of LPH with the state-of-the-art methods are demonstrated in Fig.3 and Fig.4.



**Fig. 3.** Comparison of locality preserving hash (LPH), linear discriminant analysis hash (LDAHash) [9], spectral hashing (SH) [2], and locality sensitive hash (LSH) [1] with 64-bits binary codes on our image patch dataset



**Fig. 4.** Comparison of locality preserving hash (LPH) and linear discriminant analysis hash (LDAHash) [9] with different number of bits on our image patch dataset

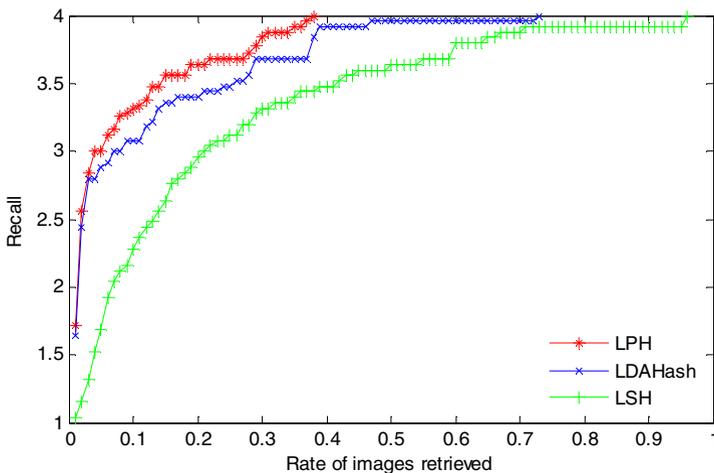
As can be seen from Fig.3, LPH and LDAHash using supervised information show significant improvements over SH and LSH. And LPH performs the best. This is because LPH generates more robust binary codes according to the local structure

preserving hash functions. An untypical observation is that SH performs worse than LSH on this dataset. Since SH aims to preserve Euclidean distances between data points, it performs well when the ground truth data are nearest neighbors in Euclidean space. However, in this experiment the ground truth data are based on visually similar patches. Hence SH degrades. This also proves that Euclidean distance can not reflect visual patch similarity properly. Besides, since we filter severely changed descriptors in the ground truth data of visually similar patch groups, LSH can compact similar points to similar codes in some extent and thus shows a higher recall than SH.

We further assess the impacts of the number of bits, as is shown in Fig.4. As the number of bits increases, recall is improved for both LPH and LDAHash but the improvement becomes smaller. With the same number of bits, LPH always perform better than LDAHash. LPH with 32 bits shows comparable performance to LDAHash with 64 bits and even better when the recall is larger than 0.5. This indicates that our binary codes are more compact, i.e. preserve more similarity information with the same number of bits.

### 4.3 Object Retrieval

To evaluate our approach on a more challenged dataset, we perform object retrieval on the Ukbench dataset which consists of groups of objects under different viewpoint, rotation, scale and lighting conditions. We set Hamming radius to 2 and the length of binary codes to 32 bits for all approaches. Fig 5 gives the average recall with a varying number of retrieved images.



**Fig. 5.** Performance of object retrieval with different number of retrieved images for locality preserving hash (LPH), linear discriminant analysis hash (LDAHash) [9], and locality sensitive hash (LSH) [1]. The dataset is Ukbench with distractors.

For a dataset of 20,200 images and 7.8 million descriptors, the recall of LPH is 6.03% higher than LDAHash on average, and 45.61% higher than LSH. LPH achieves much better performance than unsupervised LSH and supervised LDAHash

on the dataset. This proves that binary codes generated by LPH are more robust to complex transformations than the baseline methods.

The original SIFT descriptors needs 7.89G for storage while LPH only needs 78.2M. The memory cost of binary codes is one hundredth of original descriptors, which is critical for large scale applications. In this experiment we want to compare descriptor compacting approaches independently and thus adopt a simple vote scheme to rank the final results. Actually, the results can be improved further by geometry consistency verification [27-28].

## 5 Conclusion

This paper describes a new method named locality preserving hash (LPH) to compact descriptors into binary codes. The binary codes are affine robust and compact, for LPH has the following characteristics: (1) the embedding preserves manifold structure of descriptors extracted from visually similar patches; (2) automatic threshold selection achieves a good trade-off between positive classification ratio and partition balance. Besides, the off-line training of LPH is easy to implement. The experimental results on two datasets show that LPH often performs better than the baseline methods. Our future work would focus on the indexing methods in Hamming space to improve efficiency. Furthermore, there have been various improved versions of locality preserving projection and it will be interesting to explore them to improve LPH further.

**Acknowledgments.** This work was supported by the National Nature Science Foundation of China (61271428, 61273247), National Key Technology Research and Development Program of China (2012BAH39B02) and Co-building Program of Beijing Municipal Education Commission.

## References

1. Datar, M., Immorlica, N., Indyk, P., Mirrokni, V.: Locality-sensitive hashing scheme based on p-stable distributions. In: SCG 2004: Proceedings of the Twentieth Annual Symposium on Computational Geometry, pp. 253–262. ACM (2004)
2. Weiss, Y., Torralba, A., Fergus, R.: Spectral hashing. In: Proceedings of the 22nd Annual Conference on Neural Information Processing Systems, NIPS (2008)
3. Kulis, B., Darrell, T.: Learning to Hash with Binary Reconstructive Embeddings. In: Bengio, Y., Schuurmans, D., Lafferty, J., Williams, C.K.I., Culotta, A. (eds.) *Advances in Neural Information Processing Systems*, vol. 22, pp. 1042–1050 (2009)
4. Salakhutdinov, R., Hinton, G.: Semantic hashing. *Int. J. Approx. Reasoning* 50, 969–978 (2009)
5. Wang, J., Kumar, S., Chang, S.-F.: Semi-supervised hashing for scalable image retrieval. In: *Computer Vision and Pattern Recognition (CVPR)*, pp. 3424–3431 (2010)
6. Min, K., Yang, L., Wright, J., Wu, L., Hua, X.-S., Ma, Y.: Compact Projection: Simple and Efficient Near Neighbor Search with Practical Memory Requirements. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, San Francisco, USA (2010)
7. He, J., Radhakrishnan, R., Chang, S.-F., Bauer, C.: Compact hashing with joint optimization of search accuracy and time. In: *2011 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 753–760 (2011)

8. Zhang, W., Gao, K., Zhang, Y., Li, J.: Efficient approximate nearest neighbor search with integrated binary codes. In: Proceedings of the 19th ACM International Conference on Multimedia, pp. 1189–1192. ACM, Scottsdale (2011)
9. Strecha, C., Bronstein, A., Bronstein, M., Fua, P.: LDAHash: Improved Matching with Smaller Descriptors. *IEEE T. PAMI* 34, 1 (2012)
10. Wong, W.K., Zhao, H.T.: Supervised optimal locality preserving projection. *Pattern Recogn.* 45, 186–197 (2012)
11. Mu, Y., Shen, J., Yan, S.: Weakly-supervised hashing in kernel space. In: 2010 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 3344–3351 (2010)
12. He, X., Niyogi, P.: Locality Preserving Projections. In: *Neural Information Processing Systems*. MIT Press (2003)
13. Gao, K., Zhang, Y., Luo, P., Zhang, W., Xia, J., Lin, S.: Visual stem mapping and geometric tense coding for augmented visual vocabulary. In: *IEEE Conference on Computer Vision and Pattern Recognition, CVPR* (2012)
14. Arya, S., Mount, D.M., Netanyahu, N.S., Silverman, R., Wu, A.Y.: An optimal algorithm for approximate nearest neighbor searching fixed dimensions. *J. ACM* 45, 891–923 (1998)
15. Jegou, H., Douze, M., Schmid, C.: Hamming Embedding and Weak Geometric Consistency for Large Scale Image Search. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) *ECCV 2008, Part I. LNCS*, vol. 5302, pp. 304–317. Springer, Heidelberg (2008)
16. Sivic, J., Zisserman, A.: Video Google: A Text Retrieval Approach to Object Matching in Videos. In: *Proceedings of the Ninth IEEE International Conference on Computer Vision (ICCV)*, vol. 2, p. 1470 (2003)
17. Jegou, H.: Product Quantization for Nearest Neighbor Search. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 33, 117–128 (2011)
18. Poullot, S., Buisson, O., Crucianu, M.: Z-grid-based probabilistic retrieval for scaling up content-based copy detection. In: *Proceedings of the 6th ACM International Conference on Image and Video Retrieval, CIVR* (2007)
19. Brandt, J.: Transform coding for fast approximate nearest neighbor search in high dimensions. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1815–1822 (2010)
20. Jain, P., Kulis, B., Grauman, K.: Fast image search for learned metrics. In: *Computer Vision and Pattern Recognition (CVPR)*, pp. 1–8 (2008)
21. Liu, W., Wang, J., Ji, R., Jiang, Y.-G., Chang, S.-F.: Supervised Hashing with Kernels. In: *IEEE Conference on Computer Vision and Pattern Recognition, CVPR* (2012)
22. Nister, D., Stewenius, H.: Scalable Recognition with a Vocabulary Tree. In: *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2, pp. 2161–2168 (2006)
23. Morel, J.M., Yu, G.: ASIFT: A New Framework for Fully Affine Invariant Image Comparison. *SIAM Journal on Imaging Sciences* 2, 438–469 (2009)
24. Lowe, D.G.: Object recognition from local scale-invariant features. In: *The Proceedings of the IEEE International Conference on Computer Vision*, vol. 1152, pp. 1150–1157 (1999)
25. <http://www.cad.zju.edu.cn/home/dengcai/Data/ReproduceExp.html>
26. Norouzi, M., Punjani, A., Fleet, D.J.: Fast Search in Hamming Space with Multi-Index Hashing. In: *IEEE Conference on Computer Vision and Pattern Recognition, CVPR* (2012)
27. Wang, W., Zhang, D., Zhang, Y., et al.: Robust Spatial Matching for Object Retrieval and Its Parallel Implementation on GPU. *IEEE Trans. on Multimedia* 13(6), 1308–1318 (2011)
28. Xie, H., Gao, K., Zhang, Y., Tang, S., et al.: Efficient Feature Detection and Effective Post-Verification for Large Scale Near-Duplicate Image Search. *IEEE Trans. on Multimedia* 13(6), 1319–1332 (2011)