Improved Binary Feature Matching through Fusion of Hamming Distance and Fragile Bit Weight

Dongye Zhuang
Institute of Computing Technology
Chinese Academy of Sciences
University of Chinese Academy of Sciences
zhuangdongye@ict.ac.cn

Dongming Zhang
Institute of Computing Technology
Chinese Academy of Sciences
dmzhang@ict.ac.cn

Jintao Li
Institute of Computing Technology
Chinese Academy of Sciences
jtli@ict.ac.cn

Qi Tian
University of Texas at San Antonio
qitian@cs.utsa.edu

ABSTRACT

In this paper, we define a new metric, the `Fragile Bit Weight` (`FBW`), which is used in binary feature matching and measures how two features differ. High FBWs are associated with genuine matches between two binary features and low FBWs are associated with impostor ones. One bit in binary feature is deemed `fragile` if its sign of value reverses easily across the local image patch that has changed slightly. Previous research ignoring the fact that the signs of fragile bits are not stable through image transform. Rather than ignore fragile bits completely, we consider what beneficial information can be obtained from the fragile bit. In our approach, we exploit FBW as a measure in binary feature match to remove the false matches. In experiments, using FBW can effectively remove the false matches and highly improve the accuracy of feature match. Then, we find that fusion of FBW and Hamming distance work better in feature matching than Hamming distance alone. Furthermore, FBW can easily integrate in the well-established binary features if those binary bit in features extracting from comparison of pixels.

Categories and Subject Descriptors

H.5.1 [**Multimedia Information Systems**]: methodology

Keywords

Binary Feature, Feature Match, Local Feature, Fragile Bit

IMMPD'13, October 22, 2013, Barcelona, Spain.

http://dx.doi.org/10.1145/2505483.2505492.

1. INTRODUCTION

With the explosive development of mobile internet, the mobile and portable devices have become the important platforms for multimedia applications. Because of the widely use of low-power processor and memory chips, not only the processing capacity of mobile platforms are weaker than the traditional PC platforms, but the storage space are also much less than the traditional platforms. In image search applications, the mobile device usually need processing a large quantity of visual features, but the well-established features (such as SIFT [5] and SURF [6]) are all high-dimensional vectors that require a huge amount memory space to store. Furthermore, the computational complexity is so high for search the near neighbors in dataset that the mobile devices cannot satisfied. Compared with the traditional platform, the mobile one require to take more factors to consideration, include the limited computational resource and storage capacity. For these reasons, binary features (such as BRIEF[3], ORB[1], BRISK[2], FREAK[4], IDQ[8]) have attracted more and more attentions in computer vision community. These binary features use bit string to represent the local image patch around a interest point. In general, they are much more compact represented and can be extracted by very simple algorithm.

Because of the binary representation, the Hamming distance can be the distance metric between the binary features. Hamming distance based on logical exclusive-or (XOR) function is used because it ensures great performance in terms of speed and accuracy, thus this similarity measure can significantly improve the performance in binary feature based vision applications over the Euclidean distance used in the floating-based ones (such as SIFT [5] and SURF [6]).

Fig.1 shows the process of how to use the Hamming distance to match the binary features. The Hamming distance between two binary strings of equal length is the number of positions at which the corresponding symbols are different.

In order to measure the similarity between binary strings precisely, exploiting Hamming distance as a similarity measure should satisfy the assumption that every bit position in feature vector should have the equal weight. In most cases, however, this assumption is not always satisfied. In fact, the

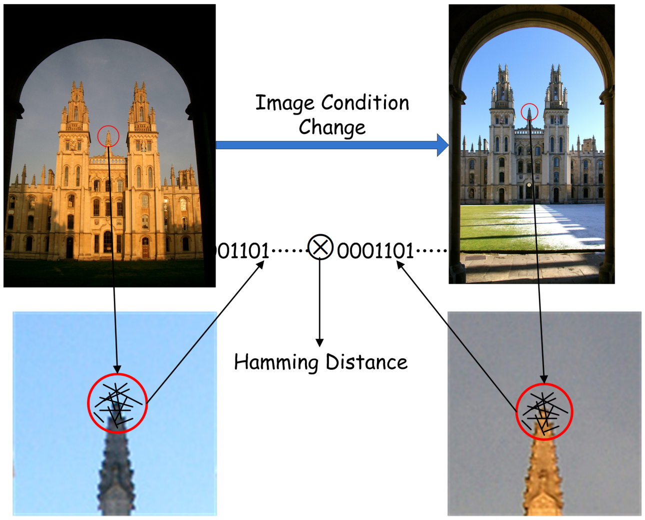

Figure 1: The toy example of process of matching the keypoints using binary feature with Hamming distance. The black line represents a pair of pixels chosen by random pattern[3]. When the lumination on the local patch of image are changed, a number of intensity differences between pixels will be reversed. Actually, the number of these reversed signs is the Hamming distance.

weights of bit position are not equal and there are a number of pairs of pixel (black line in Fig.1) whose signs are very unstable when image condition has changed slightly, because their intensity differences are small and easy to be disturbed by slight change in image patch. Thus the feature bits from those pairs will be the noise in feature matching and thus reduce the matching accuracy. Therefore, those bit position which easy to be disturbed should have lower weight than the normal ones.

Moreover, we find in experiments that binary feature matching performance is very unstable in different datasets when using the same threshold of Hamming distance. Those unstable results means that the matching performance is highly dependent upon appropriate value of threshold. Especially, the values of threshold in Hamming space are all discrete integers, thus it is hardly to get a good and stable threshold to satisfy all cases.

Therefore, Hamming distance may not be the best choice to measure the similarity between binary features. In this paper, we propose a new metric as auxiliary measure to increase the matching accuracy of binary features.

2. RELATED WORKS

In this section, we review the binary feature extraction algorithms that are related closely to our work.

For more compact representation and fast nearest-neighbor search, many binarization approaches have been proposed including Locality Sensitive Hashing (LSH) [14] [15], Spectral Hash (SH) [16], Kernel LSH (KLSH) [17], Locality Sensitive Binary Codes (LSBC) [18], Semi-Supervised Hashing (SSH) [19] [20], LDAHash [21], Hashing with Graphs [12], Supervised hashing with kernels [13] and so on. They generally consist of transforming the initial floating-vector descriptor in Hamming space where the Hamming distance H can be employed. These binary embedding methods address both the computational and memory issue: the Hamming distance between two binary features can be computed extremely efficiently and the memory footprint is drastically reduced.

Unlike those binarization schemes through supervised learning method, Zhou *et al.* [23] [22] proposed a unsupervised approach (Scalar Quantization) to quantize the floating-vector based descriptor to binary bits through scalar quantization, meaning that their method extracts binary features from the original SIFT [5] descriptor.

In addition to the Hamming embedding based methods, some works, which use the binary strings to represent a local image patch, have appeared in the international conferences and journals. This idea is firstly proposed by Gupta *et al.* [7] and attracted more and more attentions and the binary features have become a hot topic in recent years. Calonder *et al.* [3] proposed the algorithm to combine the FAST [24] interest point detector and the BRIEF (Binary Independent Elementary Feature) descriptor to detect and describe the local features with very high speed. Nevertheless, the BRIEF cannot handle the viewpoint change and multiscale problem. Rublee *et al.*[1] improves the rotation invariance of BRIEF and proposes the ORB (Oriented BRIEF) method. Almost simultaneously, BRISK [2] was also proposed to improve the robustness of BRIEF to orientation and scale. FREAK [4] is another effective binary feature, unlike ORB and BRIEF, in which the pattern of point selection is pre-trained by human retina model to improve the performance. Moreover, Ozuysal *et al.* [25] formulate the wide-baseline matching problem as a classification problem, but which need a learning process before feature extraction and consumes more memory than the conventional methods. Tang *et al.* [26] modeled the complex brightness changing and proposed OSID (Ordinal Spatial Intensity Distribution) which can handle the complex nonlinear brightness changes.

The bits in binary feature shown in Fig.1 are generated from the intensity comparative function, and the idea of feature extracting method firstly appeared in [7]. In this section, we mainly take the works in [3] [1] into account. Considering a smoothed image local patch p, a binary test τ is defined by:

$$\tau\left(p;x,y\right) := \begin{cases} 0, & p(x) < p(y) \\ 1, & p(x) \geq p(y) \end{cases} \tag{1}$$

Where $p(x)$ is the intensity of the point x. The feature is defined as a vector of n binary tests:

$$f_n\left(p\right) := \sum_{1 \leq i \leq n} 2^{i-1} \tau\left(p;x,y\right) \tag{2}$$

The experiments in [3] shows that selecting the points according to Gaussian distribution around the center of patch can achieve the best performance. In this paper, we continue to use this method to extract feature bits.

The remainder of this paper is organized as follows: In Sec.3, we give the details of `Fragile Bit Weight`. In Sec.4, we provide experimental results and discuss it more specifically. Finally we draw conclusions in Sec.5.

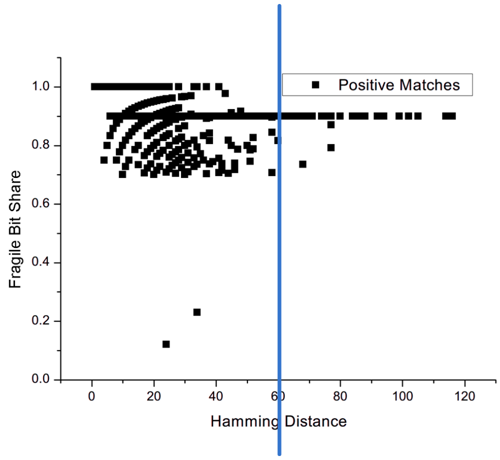

Figure 2: The relationship between the share of fragile bit and the Hamming distance in correct matches set. The vertical line (blue line) represents the threshold (Hamming Distance).

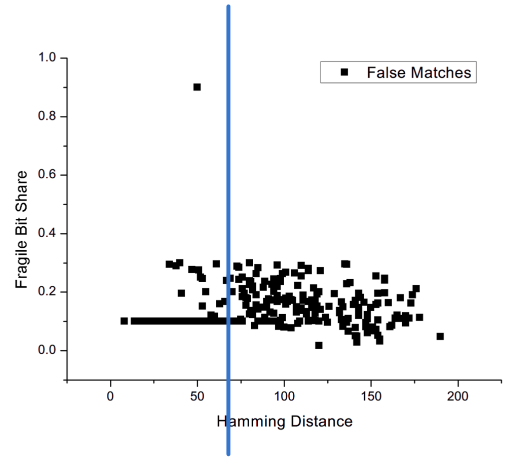

Figure 3: The relationship between the share of fragile bit and the Hamming distance in false matches set. The vertical line (blue line) represents the threshold (Hamming Distance).

3. OUR METHOD

3.1 Fragile Bit

Fig.1 provide some indication of how sensitive the sign of feature bit reverse when an image is transformed. Specifically, if the absolute value of difference between test pair is low, the probability of sign reverse is high when the local patch around the keypoint has changed. On the contrary, if the absolute value is high, the probability of reverse is low. More formally, we define τ is the fragile bit if related pixel intensity $p(x)$, $p(y)$ satisfy the inequality as follows:

$$|p(x) - p(y)| < \delta, \delta \in (0, 255) \tag{3}$$

Intuitively, we define `fragile bit` as those bit position at which the sign of intensity difference between pixels in pair has higher probability to reverse when the image is transformed slightly. By looking at the output of feature extraction, we determine the value of δ is 15.

3.2 Fragile Bit Weight

In order to compute the number of `fragile bits`, we need to indicate the fragile bit during the feature extraction step. Therefore, each feature will consist of two same size bit strings: a feature bit string f and a mark bit string m. In mark string, fragile bits are represented with ones and the others are represented with zeros. Take two binary features, feature A and feature B, the the number of fragile bits is compute as follows:

$$FB = \|m_A \bigwedge m_B\| \tag{4}$$

Where FB is the number of fragile bits, $\bigwedge$ represents the AND operator, the norm ($\|\|$) of a bit string represents the number of ones in the string. Then we can define the share of fragile bit in Hamming distance as follows:

$$s = FB/HD \tag{5}$$

where HD is Hamming distance.

To measure how well two binary features align, We introduce a metric called the `fragile bit weight` (`FBW`). And the `FBW` α is defined as a ratio:

$$\alpha = FB/HD^2 \tag{6}$$

We will discuss more details about `FBW` in sec.4.

3.3 Using FBW to Remove False Matches

In our scheme, we break the feature match process into two stages. In the first stage, we use a relatively big threshold to match binary features using Hamming distance, and through this stage we can get a set of coarse matches which contains many false matches. In the second stage, we exploit the discriminative power of `FBW` to remove those false matches and effectively increase the precision.

4. EXPERIMENTS

4.1 Consistency of Fragile Bit Weight

First of all, we label the positive and negative matches manually and these matches come from the feature matching experiments on Oxford 5K Dataset [10]. The positive set contains 10500 correct matches and the negative set contains 7500 false matches. Next, we investigate what portion of Hamming distance from the sign reversal at the fragile bit position.

Firstly, we consider the positive matches set. As shown in Fig.2, we can observe that almost all matches have the big share of fragile bit (greater than 0.6). As the previous approaches, if we apply the Hamming distance as the threshold (the blue line, $threshold = 60$ in Fig.2), the points on the left side of blue line are identified as the correct matches but the ones on the right are identified as the false matches. Although the points in Fig.2 are all correct matches, the points on the right side of blue line cannot be matched by Hamming distance alone.

Secondly, we consider the negative matches set, as shown in Fig.3, we notice that the share of fragile bit are below 0.4 for the most of matches. As previous method, we exploit the Hamming distance as the metric to classify these point, and the blue line ($threshold = 60$) is the threshold. Apparently, the points on the left of blue line are all false matches but

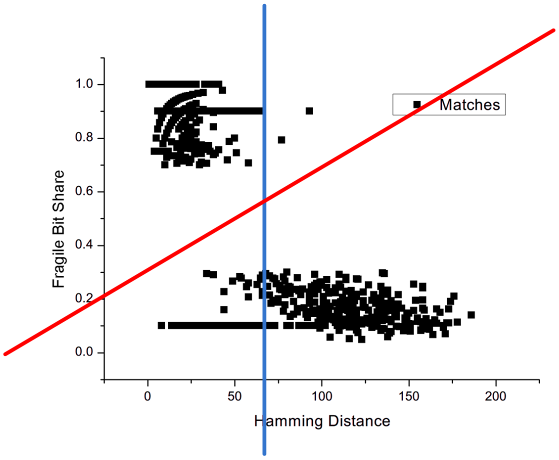

Figure 4: The red line represents the threshold (`FBW`) and it can classify the positive and negative matches more precisely than the blue line (Hamming distance). The vertical line (blue line) represents the threshold (Hamming Distance).

the Hamming distance threshold cannot distinguish them well.

Through the above analysis, we can conclude that applying the Hamming distance threshold alone to match the binary features will increase a number of false matches while lose a few of genuine matches.

Finally, We merge Fig.2 and Fig.3 into Fig.4. Obviously, the distribution of positive matches and negative matches can be approximately characterized by the diagonal symmetry. If we consider the task of feature matching as a classification problem, the classification result by the blue line can make many false matches with loss of recall of genuine matches. And the red line is ideal classification line to distinguish the correct and false matches precisely. Mathematically, the red line is the function graph of equation 6.

4.2 Performance Evaluation

In order to verify the effectiveness of our method, the dataset [9] is used in our experiments, and it contains 48 images with five different changes in imaging conditions (viewpoint changes, scale changes, image blur, JPEG compression, and illumination). The performance of our method is compared with several original descriptors (BRIEF [3], ORB [1], BRISK[2], FREAK[4]). And the result are presented with *recall* versus *precision*.

As shown in Fig.6, because `FWB` can effectively remove the false matches, the precision of feature match increase significantly without loss of recall. Fig.6 demonstrate that fusion of Hamming distance and `FBW` can achieve the better matching performance than the original binary features. In Fig.5, some typical images in dataset Oxford5K are selected as queries to demonstrate the image search performance. For these queries, compared with the baseline approach, our method can improve the mAP from 0.25 to 0.52 with almost 200% improvement. Especially when the viewpoint and light condition of image change, our method still can match the corresponding feature vectors.

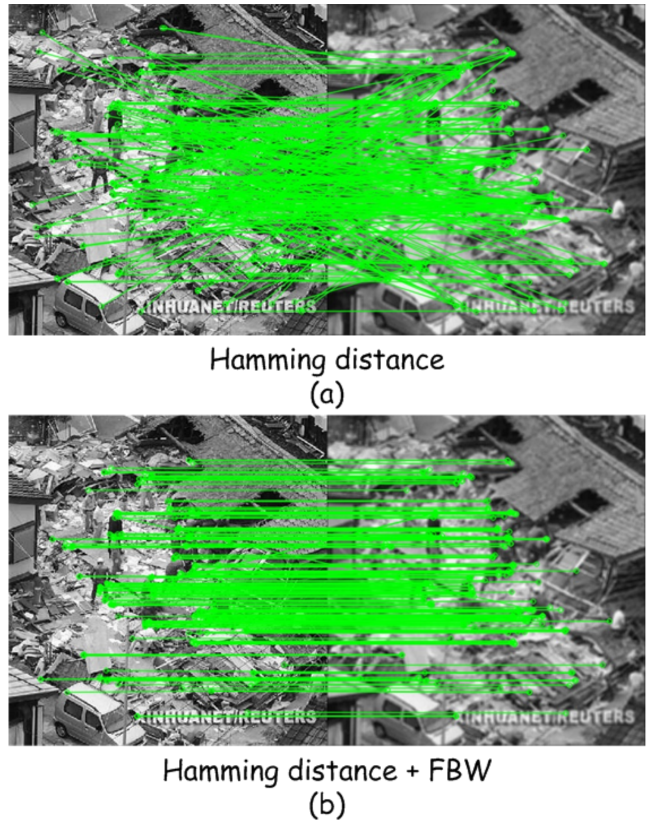

Figure 7: Figure (a) shows the keypoints matching using BRIEF (dimensionality of descriptor is 128) when the image is transformed by the Gaussian Blur, and the Hamming distance threshold is 80. Figure (b) shows the keypoints matching in the same case, the only difference is that we apply the `FBW` threshold (0.6) to refine the matches from Figure (a). Apparently, using the `FBW` (threshold is 0.6) could effectively remove those false matches.

Fig.7 shows the demonstration in BRIEF feature matching experiment. As shown in the Fig.7(a), when the image condition has changed slightly (apply a Gaussian Blur to the image), only using the Hamming distance threshold would induce many false matches. Then, using the `FBW` (threshold is 0.6) can effectively remove those false matches (Fig.7(b)).

We run the experiments on PC platform with Intel Core 2 Quad CPU 2.83GHz and 4GB memory. Compare with the original binary features, because `FBW` need to store the mark string and compute the intensity difference, the memory consumption and computation time are higher than the original binary features. Table 1 summarizes the memory usage as well as the running time.

5. CONCLUSIONS

To fulfill the requirement of compact visual descriptor on mobile platform, a number of binary features are proposed recently. Because some bits in feature string are fragile, however, the descriptor cannot remain stable when the image condition change. To eliminate the affect induced by unsta-

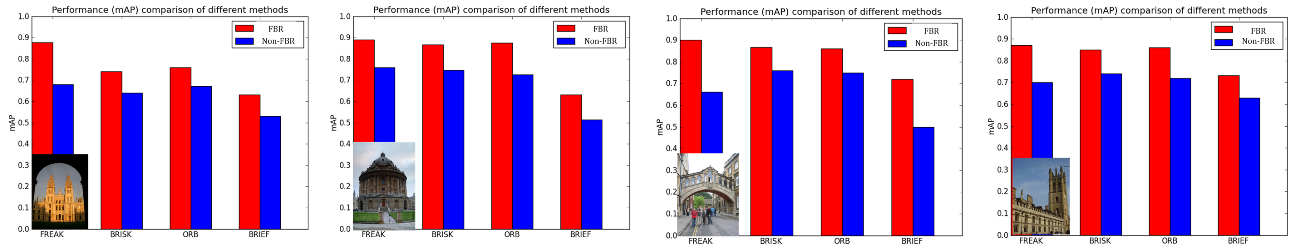

Figure 5: Experimental results comparing the baseline and our approach for example queries in the Oxford5K dataset.

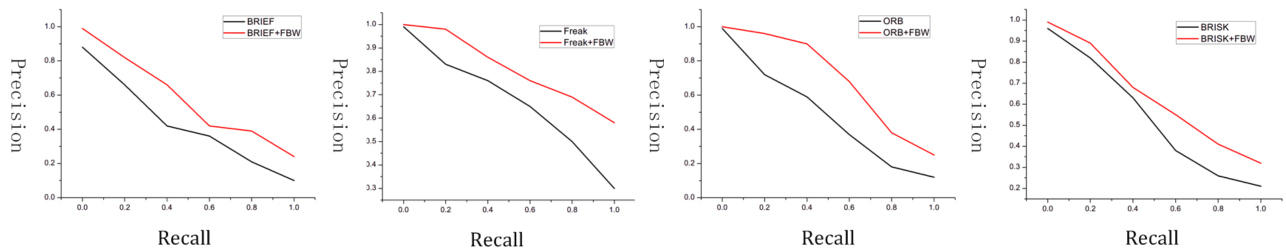

Figure 6: Performance evaluation on the dataset introduced by Mikolajczyk and Schmid [9]. This experimental result shows that our methods can effectively remove the false matches and increase the match precision significantly without loss of recall.

Table 1: The memory usage and average running time per query on the dataset.

Method	memory[MB]	extraction[ms]	matcch[ns]
BRIEF	40	0.010	24
BRIEF+FBW	80	0.30	29
FREAK	40	0.020	25
FREAK+FBW	80	0.38	30
ORB	40	0.012	23
ORB+FBW	80	0.30	32
BRISK	40	0.012	23
BRISK+FBW	80	0.45	29

ble bits, in this paper, we introduce the notion of `Fragile bit`, and then we investigate the relationship between fragile bits share and Hamming distance in the dataset we labeled manually. In the experiments, we observe that the fragile bits have much more bigger share for positive matches than negative matches. Therefore we propose a novel metric, called `Fragile Bit Weight`, as a measure to remove the false matches. The experimental demo is shown in Fig.7, and our method can significantly improve the performance of feature matching. Moreover, except for BRIEF, the `FBW` can also easily apply to other binary features.

6. ACKNOWLEDGMENTS

This work is supported by National Nature Science Foundation of China (61273247, 61271428), National Key Technology Research and Development Program of China (2012BAH-39B02). This work is supported in part to Dr. Qi Tian by ARO grant W911BF-12-1-0057, NSF IIS 1052851, Faculty Research Awards by Google, FXPAL, and NEC Laboratories of America, and 2012 UTSA START-R Research Award respectively. This work is supported in part by NSFC 61128007.

7. REFERENCES

[1] Rublee, E.; Rabaud, V.; Konolige, K. and Bradski, G. "ORB: An efficient alternative to SIFT or SURF." Computer Vision (ICCV), 2011 IEEE International Conference on, 2011, 2564 -2571

[2] Leutenegger, S.; Chli, M. and Siegwart, R. "BRISK: binary robust invariant scalable keypoints." Computer Vision (ICCV), 2011 IEEE International Conference on, 2011, 2548 -2555

[3] Calonder, M.; Lepetit, V.; Ozuysal, M.; Trzcinski, T.; Strecha, C. and Fua, P. "BRIEF: Computing a local binary descriptor very fast." Pattern Analysis and Machine Intelligence, IEEE Transactions on, 2012, 34, 1281 -1298

[4] Alahi, A.; Ortiz, R. and Vandergheynst, P. "FREAK: Fast Retina Keypoint." Computer Vision and Pattern Recognition, 2012. CVPR '12. IEEE Conference on, June 16-21, 2012

[5] Lowe, D. G. "Distinctive Image Features from Scale-Invariant Keypoints." International Journal of Computer Vision, 2004, 60, 91-110

[6] Bay, H.; Tuytelaars, T. and Van G., L. "SURF: Speeded Up Robust Features." ECCV 2006, 2006, 3951, 404-417

[7] Gupta, R.; Mittal, A. Forsyth, D.; Torr, P. and Zisserman, A. "SMD: a Locally atable monotonic change invariant feature descriptor." ECCV 2008, Springer Berlin / Heidelberg, 2008, 5303, 265-277

[8] Zhuang, D.; Zhang, D.; Li, J. and Tian, Q. ”a novel binary feature from intensity difference quantization between random sample of points.” Advances in Multimedia Modeling, Springer, 2013, 7733, 228-238

[9] Mikolajczyk, K. and Schmid, C. ”A performance evaluation of local descriptors.” Pattern Analysis and Machine Intelligence, IEEE Transactions on, 2005, 27, 1615 -1630

[10] Philbin, J.; Chum, O.; Isard, M.; Sivic, J. and Zisserman, A. ”Object retrieval with large vocabularies and fast spatial matching.” Computer Vision and Pattern Recognition 2007. CVPR ’07. IEEE Conference on, 2007, 1 -8

[11] Tan, Xiaoyang, and Bill Triggs. ”Enhanced local texture feature sets for face recognition under difficult lighting conditions.” Analysis and Modeling of Faces and Gestures. Springer Berlin Heidelberg, 2007. 168-182.

[12] Wei Liu, Jun Wang, Sanjiv Kumar, Shih-Fu Chang. ”Hashing with Graphs.” ICML 2011: 1-8.

[13] Wei Liu, Jun Wang, Rongrong Ji, Yu-Gang Jiang, Shih-Fu Chang. ”Supervised hashing with kernels.” CVPR 2012: 2074-2081.

[14] Charikar, M. S. ”Similarity estimation techniques from rounding algorithms.” Proceedings of the thiry-fourth annual ACM symposium on Theory of computing, ACM, 2002, 380-388

[15] Indyk, P. and Motwani, R. ”Approximate nearest neighbors: towards removing the curse of dimensionality.” Proceedings of the thirtieth annual ACM symposium on Theory of computing, ACM, 1998, 604-613

[16] Weiss, Y.; Torralba, A. B. and Fergus, R. ”Spectral Hashing.” Neural Information Processing Systems, 2008, 1753-1760

[17] Kulis, B. and Grauman, K. ”Kernelized locality-sensitive hashing for scalable image search.” International Conference on Computer Vision, 2009, 2130-2137

[18] Raginsky, M. and Lazebnik, S. ”Locality-sensitive binary codes from shift-invariant kernels.” The Neural Information Processing Systems, 2009, 22

[19] Wang, J.; Kumar, S. and Chang, S. Semi-supervised hashing for scalable image retrieval Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on, 2010, 3424-3431

[20] Wang, J.; Kumar, S. and Chang, S.-F. ”Semi-Supervised Hashing for Large-Scale Search.” Pattern Analysis and Machine Intelligence, IEEE Transactions on, 2012, 34, 2393 -2406

[21] Strecha, C.; Bronstein, A.; Bronstein, M. and Fua, P. ”LDAHash: Improved Matching with Smaller Descriptors.” Pattern Analysis and Machine Intelligence, IEEE Transactions on, 2012, 34, 66-78

[22] Zhou, W.; Li, H.; Lu, Y. and Tian, Q. ”Scalar Quantization for Large Scale Image Search.” ACM Multimedia, Full Paper,, 2012

[23] Zhou, W.; Li, H.; Wang, M.; Lu, Y. and Tian, Q. ”Binary SIFT: Towards Efficient Feature Matching Verification for Image Search.” ACM ICIMCS, 2012

[24] Rosten, E. and Drummond, T. ”Fusing points and lines for high performance tracking Computer Vision.”, 2005. ICCV 2005. Tenth IEEE International Conference on, 2005, 2, 1508 -1515 Vol. 2

[25] Ozuysal, M.; Calonder, M.; Lepetit, V. and Fua, P. ”Fast Keypoint Recognition Using Random Ferns.” Pattern Analysis and Machine Intelligence, IEEE Transactions on, 2010, 32, 448 -461

[26] Tang, F.; Lim, S.; Chang, N. and Tao, H. ”A novel feature descriptor invariant to complex brightness changes.” Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on, 2009, 2631 -2638

[27] Zhang, L.; Zhang, Y.; Tang, J.; Lu, K. and Tian, Q. ”Binary Code Ranking with Weighted Hamming Distance.” 2013 IEEE International Conference on Computer Vision and Pattern Recognition (CVPR), June 23-28, 2013.

[28] Zhou, W.; Lu, Y,; Li, H.; Song, Y. and Tian, Q. ”Spatial Coding for Large Scale Partial-Duplicate Web Image Search.” ACM Multimedia, October 25-29, 2010.

[29] Zhang, S.; Tian, Q.; Hua, G.; Huang, Q. and Li, S. ”Descriptive Visual Words and Visual Phrases for Image Applications.” ACM Multimedia 2009, October 19-23, 2009.