Research Article

An Image Similarity Acceleration Detection Algorithm Based on Sparse Coding

Luan Xidao,¹ Xie Yuxiang ,² Zhang Lili,² Zhang Xin ,² Li Chen,² and He Jingmeng²

¹Department of Mathematics and Computer Science, Changsha University, Changsha 410003, China

²College of Systems Engineering, National University of Defense Technology, Changsha 410073, China

Correspondence should be addressed to Xie Yuxiang; yxxie@nudt.edu.cn

Received 2 January 2018; Accepted 25 March 2018; Published 20 May 2018

Academic Editor: Xosé M. Pardo

Copyright © 2018 Luan Xidao et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Aiming at the problem that the image similarity detection efficiency is low based on local feature, an algorithm called ScSIFT for image similarity acceleration detection based on sparse coding is proposed. The algorithm improves the image similarity matching speed by sparse coding and indexing the extracted local features. Firstly, the SIFT feature of the image is extracted as a training sample to complete the overcomplete dictionary, and a set of overcomplete bases is obtained. The SIFT feature vector of the image is sparse-coded with the overcomplete dictionary, and the sparse feature vector is used to build an index. The image similarity detection result is obtained by comparing the sparse coefficients. The experimental results show that the proposed algorithm can significantly improve the detection speed compared with the traditional algorithm based on local feature detection under the premise of guaranteeing the accuracy of algorithm detection.

1. Introduction

Image similarity detection is a hot issue in the field of multimedia information processing. Similar image is a set of images obtained from an image of the same scene or the same object taken from different environmental conditions such as different angles or different lighting conditions and edited transformations of the same original image through different ways. Examples of some similar images are shown in Figure 1.

Image similarity detection is to judge the similarity of visual content by matching the image. According to the adopted feature, image similarity detection methods can be divided into two categories, namely, global-feature-based detection methods and local-feature-based detection methods. The global feature of the image refers to the use of one or a few feature vectors to represent whole image content. Common global features include color histograms, texture features, and block features. Because the number of feature points is small, the calculation speed of image content similarity detection based on global feature is usually very fast. However, due to the singularity of its feature selection and the roughness of the description image, the global feature

is very susceptible to edits and local transformations. For example, image similarity detection with color histogram as global feature is very sensitive to the illumination of the image. Usually, similar images are created by editing transformation; similarity detection accuracy is generally relatively low based on the global features of the image content.

In recent years, some scholars have proposed local features for image similarity detection. Compared with global features, local features of the image usually have some local invariance for the illumination, rotation and scaling of the image and have been widely applied in the field of content-based image and video retrieval. Local feature points are usually local extreme points in an area of the image, and have more obvious features than the rest of the pixels in the region. Description of the local feature points is generally the combination of the characteristics of the key points and the information of the surrounding area, thus ensuring the local invariance of the feature.

The SIFT feature proposed in [1] is still the most accurate feature of image matching because of its good invariance of rotation, scaling, and scale of the local neighborhood of

FIGURE 1: Examples of some similar images.

the image, but its 128-dimensional high dimensional feature vectors and the number of feature points detected will usually bring a large computational burden and reduce the efficiency of the algorithm. In the literature [2], dimension of the SIFT feature point is reduced by principal component analysis (PCA), and a PCA-SIFT feature is proposed. This feature reduces the SIFT feature vector from 128 degrees to 32 degrees or less, which improves the efficiency of the algorithm. Hong Kong City University [3] proposes a similar key frame detection algorithm based on the PCA-SIFT, better than the previous algorithm. Based on the timing information of key frames, the algorithm divides the key frames into different time series groups, performs similar key frame detection in PCA-SIFT in each group, and then analyzes the correlation between different story units. Although the algorithm improves the efficiency of the algorithm by PCA dimensionality reduction, it is still difficult to meet the real-time requirement of large data processing. In the literature [4], a SIFT feature filtering algorithm is proposed, which effectively reduces the SIFT feature points extracted from an image by the punishment mechanism, reduces the computational complexity, and improves the matching accuracy of the SIFT algorithm. In [5, 6], the image is classified into different categories by using the clustering algorithm. It calculates the distance between the image to be detected and different clustering center and selects the nearest several types of images and exactly matches their SIFT characters, which reduces the SIFT feature matching data and improves the detection speed of the algorithm.

In summary, the shortcomings of the current image similarity detection are as follows: although the global feature is faster, it does not have local information, which is usually sensitive to the partial change of the image and is not robust enough. Although the local feature can guarantee the local deformation of the feature points, the computational efficiency is very low due to the high dimension of the feature points. At present, the research based on local features mainly focuses on the acceleration of local invariant features, and some studies focus on the comprehensive use of global features and local features [7]. How to detect the content of similar images quickly and effectively is still a problem.

2. Principle of ScSIFT Algorithm

The algorithm of Scale Invariant Feature Transform based on Sparse Coding (ScSIFT) focuses on the sparse SIFT feature vectors. After extracts of the SIFT feature, the algorithm conducts sparse coding on the 128-dimension SIFT feature vector and establishes the feature sparse coding index to improve the matching speed. The merits of the sparse representation of the original signal depend largely on the selection and design of the overcomplete dictionary. The errors generated by the partial eigenvectors are less than the errors produced by the default fixed dictionaries when sparsity represents the rest of the eigenvectors, and different dictionary learning algorithm to learn the error between the dictionaries is basically the same. This paper uses the dictionary learning algorithm proposed in [8] to train the complete dictionary.

The main idea of the ScSIFT algorithm is to use the SIFT feature extracted from the key frame image of the query library as a training sample to train the overcomplete dictionary to obtain a set of overcomplete bases. The SIFT feature vector of the query image is sparsely encoded with the overcomplete dictionary, and the sparse feature vector is indexed. Furthermore, the SIFT feature of the image to be detected is matched with the query image feature index by using the overcomplete dictionary to obtain a set of similar candidate sets, compare the sparse coefficient of the image to be detected and the sparse coefficient of the candidate image and get similar image detection results. SIFT feature sparse coding of the sparse coefficient vector is called the image ScSIFT features.

While learning the dictionary as a training sample with the SIFT feature, the SIFT feature should be normalized. Standardization process can be seen in

$$\begin{aligned} X' &= X - X_{j\text{means}}, \\ X_s &= \frac{X'}{|X'|_{j\text{mod}}}, \end{aligned} \quad (1)$$

where X is a $128 \times n$ SIFT feature matrix. $X_{j\text{means}}$ is a $128 \times n$ matrix with the mean of the columns of eigenvector group X . $|X'|_{j\text{mod}}$ is a $128 \times n$ matrix with the mod of the columns of eigenvector X' . X_s is a normalized SIFT feature vector group.

In general, the distance between the image's SIFT feature points uses Euclidean distance or absolute distance measurement. Euclidean distance is also known as L_2 norm distance, and the distance calculation can be seen in

$$\begin{aligned} L_2(X, Y) &= \|X - Y\|_2 = \sqrt{(X - Y)(X - Y)^T} \\ &= \sqrt{\sum_{i=1}^{128} (x_i - y_i)^2}. \end{aligned} \quad (2)$$

The sparse representation of the eigenvector is performed by the overcomplete dictionary D obtained by training, as shown in

$$X = D \cdot \alpha, \text{ sparse}. \quad (3)$$

The sparse feature vector is as shown in

$$\begin{aligned} \|X - Y\|_2^2 &= (X - Y)(X - Y)^T \\ &= [D(\alpha_X - \alpha_Y)] [D(\alpha_X - \alpha_Y)]^T \\ &= D(\alpha_X - \alpha_Y)(\alpha_X - \alpha_Y)^T D^T \\ &= D \|\alpha_X - \alpha_Y\|_2 D^T, \end{aligned} \quad (4)$$

where $\|X - Y\|_2$ is the Euclidean distance of feature vectors X and Y . $\|\alpha_X - \alpha_Y\|_2$ is the Euclidean distance of sparse vectors α_X and α_Y .

From formula (4), we can see that the square of the distance between the SIFT features is related to the square of the distance between the sparse representation of the eigenvector. Thus, the distance between the original SIFT

features can be represented by the distance of the sparse feature vector.

$$\|\alpha_X - \alpha_Y\|_2 = \sqrt{\sum_i (\alpha_{Xi} - \alpha_{Yi})^2}. \quad (5)$$

Since the dimension of ScSIFT feature vector is much higher than SIFT feature vector, it would be time-consuming to compute the distance by (5). While the vast majority of sparse vectors are zero, the ScSIFT feature distance calculation can be simplified as

$$\|\alpha_X - \alpha_Y\|_2 = \sqrt{\sum_i \alpha_{Xi}^2 + \sum_j \alpha_{Yj}^2 + \sum_k (\alpha_{Xk} - \alpha_{Yk})^2}, \quad (6)$$

where i is the element ordinal for those nonzero values in vector α_X and zero value in vector α_Y . Similarly, j is the element ordinal for those zero values in vector α_X and nonzero value in vector α_Y . k is the element ordinal for those nonzero values in both vectors α_X and α_Y .

When comparing the ScSIFT features of the image and the candidate set image, the ScSIFT feature of the candidate set image can be indexed to improve the retrieval efficiency. The indexing process is as follows.

By sparse coding, the 128-dimensional SIFT eigenvector $F = (f_1, f_2, \dots, f_{128})$ has been transformed into an n -dimensional ScSIFT eigenvector $\alpha = (0, \dots, 0, \alpha_{i1}, 0, \dots, 0, \alpha_{ik}, 0, \dots, 0, \dots)_n$, $n > 128$. Only k -dimensional data is nonzero in α , and then the secondary index table can be established.

The ScSIFT feature vector is transformed into a binary vector; that is,

$$\begin{aligned} \alpha_{\text{bool}} &= \text{bool}(\alpha) \\ &= \text{bool}((0, \dots, 0, \alpha_{i1}, 0, \dots, 0, \alpha_{ik}, 0, \dots, 0, \dots)_n) \\ &= (0, \dots, 0, 1_{i1}, 0, \dots, 0, 1_{ik}, 0, \dots, 0, \dots)_n, \end{aligned} \quad (7)$$

where

$$\text{bool}(x) = \begin{cases} 1 & x \neq 0, \\ 0 & x = 0. \end{cases} \quad (8)$$

The binary vector α_{bool} is used as a secondary index of the ScSIFT feature. α_{bool} is an n -bit binary string. In the worst case, α_{bool} is adopted as an index for 2^n comparisons. Since $n > 128$, only using α_{bool} as an index may still result in significant computational consumption. So, on the basis of the secondary index α_{bool} , the first-level index can be created.

Calculate the number num of nonzero elements in the secondary index α_{bool} . Because the vast majority of elements in α_{bool} are 0, the range of num is not large and varies in the $[0, 20]$ range. Use num as the first-level index of the ScSIFT feature.

When the query ScSIFT eigenvector is to be compared with the ScSIFT vector in dataset, the ScSIFT eigenvector β is transformed into a binary vector β_{bool} according to formula (7), and the number num_β of nonzero elements in β_{bool} is calculated. Then compare num_β with the first-level

index to the corresponding column num_r , and compare β_{bool} and column num_r to retrieve the corresponding ScSIFT eigenvector $(\alpha_{i1}, \alpha_{i2}, \dots, \alpha_{ik})$ in the dataset. Finally, match the ScSIFT feature vector β and $(\alpha_{i1}, \alpha_{i2}, \dots, \alpha_{ik})$ to find the nearest neighbor.

3. Process of ScSIFT Algorithm

The implementation of image similarity acceleration detection algorithm based on sparse coding is mainly divided into three subprocesses: sparse dictionary learning algorithm, query image offline sparse coding algorithm, and real-time matching algorithm of images.

3.1. Sparse Dictionary Learning. The basic process of sparse dictionary learning algorithm is as follows:

- (1) Select the n images $I_1, I_2, \dots, I_n$ from the query library, extract the SIFT features of these images, and constitute the training feature set F .
- (2) Normalize the training feature set F according to formula (1).
- (3) Train the overcomplete dictionary D using the trained feature set F .
- (4) Output dictionary D , end.

3.2. Offline Sparse Coding. Offline sparse coding is carried out for the similar images in the query library; the basic process is as follows:

- (1) Read the first image I of the query library.
- (2) Extract the SIFT feature of the image I to constitute the image feature set F .
- (3) The training feature set F is normalized according to formula (1), and the normalized feature set F_s is obtained.
- (4) The sparse algorithm is performed on the normalized feature set F_s ; preserve the sparse coefficient α , that is, the ScSIFT feature.
- (5) If there is an uncoded image in the query library, continue to read the next image I , and repeat steps (2), (3), and (4). Otherwise, go to step (6).
- (6) Create index and save all images' ScSIFT features in the query library.
- (7) End.

3.3. Real-Time Matching of Images. The basic process of the algorithm of real-time matching images is as follows:

- (1) Read the image I , extract its SIFT feature to form the feature set F , and normalize it to F_s .
- (2) Get the sparse representation of the normalized feature set F_s by the sparse algorithm, and obtain the ScSIFT feature β .
- (3) Set the ScSIFT feature similarity distance threshold σ and image similarity threshold θ , and read the first column of ScSIFT feature $x = \beta_0$.

- (4) Search for the nearest k column coefficients α'_k of the ScSIFT feature x according to the index index .
- (5) Calculate the distance d_k of α'_k for each column and x according to formula (6). If $d_k \leq \sigma$, then the feature matching amount $\sum_i$ of the image i which the column belongs to is increased by 1.
- (6) Repeat steps (4) and (5) until all columns of the ScSIFT feature are cycled once.
- (7) For each feature matching amount of image i which meets the condition $\sum_i \neq 0$, calculate the total number of sparse feature points Total_i and the similarity degree $\sum_i / \text{Total}_i$. If $\sum_i / \text{Total}_i \geq \theta$, then the image i is the similar image to the query image.
- (8) End.

4. Experimental Results and Analysis

In the experiment, we select a total of 10816 frames extracted from 1000 videos as the query library images. Five different images are selected from the library, and, for each image, Gaussian blur, the mark transform, the gray scale transformation, and the scale cropping transform are carried out. The original image and the transformed images, altogether 50 images, are compared with the query library by similarity detection, and comparisons of experimental results are carried out.

4.1. Sparse Coding Dictionary Learning Experiment. Error of sparse coding depends mainly on the choice of overcomplete dictionary. In this paper, we choose the overcomplete dictionary based on feature learning and the overcomplete dictionary of DCT as a comparison and carry out sparse coding experiments to test the influence of different dictionaries on sparse coding. Experimental factors include two aspects: time of training dictionary and feature coding and error of the dictionary sparse coding.

The sparse coding error R is calculated by

$$R = \frac{1}{2} (x - D\alpha)^2 + \lambda |\alpha|, \quad (9)$$

where x is the original data signal, D is the overcomplete dictionary, α is the sparse coding of the signal x , and λ is the sparse coding regular parameter.

In this paper, number of blocks, number of cycles, and dictionary dimension are used as the control variable training dictionaries to compare the coding errors and the differences in dictionary learning time between different control variables. At the same time, use the DCT dictionary as the benchmark, and compare the coding error and the feature coding time difference between the feature learning dictionary and the DCT dictionary of different dimension. The experimental results are shown in Figure 2.

Figures 2(a) and 2(b) show the errors and the dictionary learning time between different number of blocks in the dictionary learning and different number of cycles. Here, the number of blocks in Figure 2(a) and the number of training cycles in Figure 2(b) are set to (100, 200, ..., 1000), respectively. It can be seen from Figures 2(a) and 2(b) that

FIGURE 2: Dictionary learning experiment results.

change of the number of blocks and the number of cycles have no significant effect on the dictionary coding error, and the variation of the dictionary coding error is less than 0.01, while it has a greater impact on dictionary learning time. Therefore, the variable block number and the cycle number can be set to a smaller value to save training time and improve the efficiency of dictionary learning.

Figures 2(c) and 2(e) are the dictionary coding errors corresponding to the dictionary dimension change set (200, 400, ..., 5000) and the dictionary learning time. Figures 2(d) and 2(f) are the coding time required for sparse coding of features in a dictionary of corresponding dimensions. Compared with Figures 2(c) and 2(e), it can be found that the coding error of the feature learning dictionary and the DCT dictionary decreases significantly with the increase of the dictionary dimension, but the DCT dictionary coding error decreases to about 0.43 without obvious change, and its dictionary construction time is basically the same. The error of the feature learning dictionary decreases from 0.28 to 0.15 as the dimension increases, but its dictionary learning time is also significantly increased. At the same time, compared with Figures 2(d) and 2(f), the two dictionary features of the coding time are significantly increased when the dictionary dimension increases, the feature learning dictionary coding time accompanied by the dimension's increasing eventually remains at 15 seconds, and the encoding time of DCT dictionary is increased from 0 to 40 seconds significantly.

By contrast, we can find that the feature learning dictionary has obvious advantages compared with the DCT dictionary. One is that its coding error is lower than that of the DCT dictionary, and it is less than 28%, and the coding time is better than that of the DCT dictionary. When the dictionary is trained by the feature, change of the feature number and the cycle number has little effect on the coding error of the dictionary, but the dictionary learning time is significantly increased. Therefore, these two parameters can be set low to improve the dictionary learning efficiency. In this paper, set the block number batch = 400, the cycle number iter = 100, and the dictionary dimension $w_D = 512$.

4.2. ScSIFT Algorithm Validity Test. The validity of ScSIFT algorithm is verified by comparison with SIFT algorithm and SURF algorithm. In the experiment, select five key frame images and 10 sets of key frame images as query image to test different algorithm's running time. The number of key frame images is (5, 15, ..., 95) frames, respectively.

Feature extraction time, distance calculation time between feature points, average matching time of each frame, and total run time of three algorithms are shown in Table 1. The total runtime is shown in Figure 3.

In Table 1, single-frame detection time is the average similarity detection time for a frame.

From Table 1 and Figure 3, we can find that SIFT feature extraction speed is the fastest with an average of 0.7 seconds.

TABLE 1: Feature extraction and distance detection time of the three algorithms.

Number of key frames		5	15	25	35	45	55	65	75	85	95
ScSIFT	Feature extraction time	0.70	0.72	0.82	0.68	0.70	0.69	0.69	0.68	0.70	0.70
	Detection time	0.21	0.54	0.71	0.90	1.14	1.32	1.58	1.75	1.87	2.18
	Single-frame detection time	0.04	0.04	0.03	0.03	0.03	0.02	0.02	0.02	0.02	0.02
	Total runtime	0.91	1.26	1.53	1.58	1.84	2.01	2.26	2.43	2.57	2.88
SIFT	Feature extraction time	0.46	0.46	0.45	0.46	0.45	0.45	0.45	0.45	0.46	0.45
	Detection time	0.80	2.30	2.87	3.05	3.51	4.16	4.22	4.89	4.93	5.56
	Single-frame detection time	0.16	0.15	0.11	0.09	0.08	0.08	0.06	0.07	0.06	0.06
	Total runtime	1.27	2.75	3.32	3.51	3.96	4.61	4.67	5.35	5.40	6.02
SURF	Feature extraction time	2.76	2.76	2.76	2.76	2.75	2.77	2.76	2.75	2.76	2.76
	Detection time	0.19	0.44	0.52	0.62	0.70	0.76	0.85	0.93	0.96	1.08
	Single-frame detection time	0.04	0.03	0.02	0.02	0.02	0.01	0.01	0.01	0.01	0.01
	Total runtime	2.95	3.20	3.28	3.38	3.45	3.54	3.61	3.69	3.72	3.84

FIGURE 3: Total runtime comparison of the three algorithms.

SURF feature extraction speed is the slowest with an average of 2.76 seconds. For the same number of key frames similarity detection, SIFT algorithm detection time is 4 times of ScSIFT algorithm. The detection time of ScSIFT algorithm and SURF is very close when the number of key frames is small, and as the number of key frames increases, the speed is slower than SURF. This is mainly because the number of feature points extracted by the SIFT algorithm is larger than the number extracted by the SURF feature. When the number of key frames is large, the disadvantage of increased computational burden is gradually evident. However, due to the slow feature point detection speed of the SURF algorithm, total running time of the algorithm is slower than that of the ScSIFT algorithm and the SIFT algorithm when the number of key frames is small. Experiments show that the average run time of ScSIFT algorithm is 52% higher than that of SIFT algorithm, which is 45% higher than SURF algorithm.

Six frames which are transformed by gray scale transformation, scale cropping transformation, tag adding, and

Gaussian blur are selected. The similarity determination is performed by ScSIFT algorithm and SIFT algorithm, respectively, with other selected fifteen frames to test the robustness of ScSIFT algorithm in different edit styles. ScSIFT algorithm and SIFT algorithm for different transformations of key frame similarity detection results are shown in Tables 2 and 3.

The comparison results of Tables 2 and 3 are shown in Figure 4.

In Figure 4, “○” is the ScSIFT algorithm detection results, “◆” is the SIFT algorithm test results. The abscissa in the figure is the image frame number, and the ordinate is similarity. Combining the results in Tables 2 and 3 and Figure 4, it is shown that the similarity search results of ScSIFT algorithm are basically the same as SIFT algorithm, and it is slightly higher than SIFT for some videos. In summary, the experimental results show that ScSIFT algorithm is similar to SIFT algorithm, but its operation speed is faster, which is 52% higher than that of the latter.

TABLE 2: ScSIFT algorithm similarity detection results.

Number	Original image	Reduce grayscale	Increase grayscale	Add mark	Gaussian blur	Image Crop
1	0.82	0.82	0.82	0.80	0.82	0.54
2	0.41	0.43	0.41	0.40	0.47	0.39
3	0.44	0.42	0.40	0.41	0.40	0.41
4	0.28	0.22	0.19	0.31	0.19	0.19
5	0.60	0.72	0.44	0.68	0.32	0.36
6	0.45	0.39	0.39	0.45	0.27	0.30
7	0.20	0.18	0.20	0.18	0.17	0.16
8	0.63	0.62	0.59	0.57	0.59	0.64
9	0.00	0.00	0.02	0.00	0.00	0.00
10	0.85	0.87	0.82	0.85	0.79	0.54
11	0.32	0.36	0.36	0.34	0.21	0.23
12	0.13	0.14	0.08	0.13	0.16	0.05
13	0.12	0.10	0.10	0.12	0.08	0.08
14	0.02	0.00	0.02	0.02	0.02	0.00
15	0.26	0.26	0.33	0.26	0.18	0.28

TABLE 3: SIFT algorithm similarity detection results.

Number	Original image	Reduce grayscale	Increase grayscale	Add mark	Gaussian blur	Image crop
1	0.77	0.75	0.74	0.75	0.77	0.59
2	0.43	0.41	0.41	0.42	0.43	0.38
3	0.36	0.34	0.32	0.36	0.32	0.32
4	0.09	0.03	0.09	0.13	0.03	0.06
5	0.48	0.52	0.44	0.48	0.20	0.24
6	0.18	0.16	0.18	0.20	0.09	0.09
7	0.21	0.18	0.20	0.20	0.18	0.18
8	0.64	0.62	0.60	0.60	0.63	0.63
9	0.00	0.00	0.00	0.00	0.00	0.00
10	0.77	0.74	0.72	0.79	0.62	0.38
11	0.28	0.28	0.19	0.28	0.13	0.17
12	0.06	0.06	0.06	0.06	0.08	0.03
13	0.04	0.04	0.04	0.04	0.06	0.04
14	0.00	0.00	0.00	0.00	0.00	0.00
15	0.79	0.77	0.69	0.79	0.69	0.44

5. Conclusions

Image similarity detection is a hot issue in the field of multimedia information processing. Current image similarity detection methods are mainly based on global features or local features. Image similarity matching method based on global feature is fast but has the disadvantage of poor robustness. Image similarity matching method based on local feature is just the opposite. Aiming at the problem of high computational complexity and low efficiency based on the local feature detection method, an algorithm for image similarity detection based on sparse coding is proposed. The algorithm improves the detection speed by sparsely coding

the local features and indexing them. The experimental results show that, compared with the traditional image similarity detection algorithm based on local feature, the algorithm can improve the detection speed while ensuring the detection efficiency. In the future work, we will further combine global and local features to ensure the best balance between detection accuracy and speed.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

FIGURE 4: The detection results of two algorithms on different keyframe transforms.

Acknowledgments

This work has been supported by the National Natural Science Foundation of China under Contract no. 61571453, Science and Technology of Hunan Education Department (no. 15A020), and Science and Technology of Changsha (no. ZD1601014).

References

- [1] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.
- [2] Y. Ke and R. Sukthankar, "PCA-SIFT: a more distinctive representation for local image descriptors," in *Proceedings of*

the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '04), vol. 2, pp. 506–513, July 2004.

- [3] W.-L. Zhao and C.-W. Ngo, “Scale-rotation invariant pattern entropy for keypoint-based near-duplicate detection,” *IEEE Transactions on Image Processing*, vol. 18, no. 2, pp. 412–423, 2009.
- [4] W. Jinde, L. Xiaoyan, and C. G. ShouLidan, “A SIFT pruning algorithm for efficient near-duplicate image matching,” *Journal of Computer-Aided Design & Computer Graphics*, vol. 22, pp. 1042–1049, 2010.
- [5] H. Xie, K. Gao, Y. Zhang, S. Tang, J. Li, and Y. Liu, “Efficient feature detection and effective post-verification for large scale near-duplicate image search,” *IEEE Transactions on Multimedia*, vol. 13, no. 6, pp. 1319–1332, 2011.
- [6] T. Pönitz and J. Stöttinger, “Efficient and robust near-duplicate detection in large and growing image data-sets,” in *Proceedings of the 18th ACM International Conference on Multimedia ACM Multimedia 2010, MM'10*, pp. 1517–1518, October 2010.
- [7] Y. Wang, Z. Hou, K. Leman, N. T. Pham, T. Chua, and R. Chang, “Combination of local and global features for near-duplicate detection,” *Lecture Notes in Computer Science (including sub-series Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics): Preface*, vol. 6523, no. 1, pp. 328–338, 2011.
- [8] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, “Online learning for matrix factorization and sparse coding,” *Journal of Machine Learning Research*, vol. 11, pp. 19–60, 2010.

Submit your manuscripts at
www.hindawi.com