BSIFT: Toward Data-Independent Codebook for Large Scale Image Search

Wengang Zhou, Houqiang Li, Senior Member, IEEE, Richang Hong, Member, IEEE, Yijuan Lu, Member, IEEE, and Qi Tian, Senior Member, IEEE

Abstract—Bag-of-Words (BoWs) model based on Scale Invariant Feature Transform (SIFT) has been widely used in large-scale image retrieval applications. Feature quantization by vector quantization plays a crucial role in BoW model, which generates visual words from the high-dimensional SIFT features, so as to adapt to the inverted file structure for the scalable retrieval. Traditional feature quantization approaches suffer several issues, such as necessity of visual codebook training, limited reliability, and update inefficiency. To avoid the above problems, in this paper, a novel feature quantization scheme is proposed to efficiently quantize each SIFT descriptor to a discriminative bit-vector, which is called binary SIFT (BSIFT). Our quantizer is independent of image collections. In addition, by taking the first 32 bits out from BSIFT as code word, the generated BSIFT naturally lends itself to adapt to the classic inverted file structure for image indexing. Moreover, the quantization error is reduced by feature filtering, code word expansion, and query sensitive mask shielding. Without any explicit codebook for quantization, our approach can be readily applied in image search in some resource-limited scenarios. We evaluate the proposed algorithm for large scale image search on two public image data sets. Experimental results demonstrate the index efficiency and retrieval accuracy of our approach.

Manuscript received October 31, 2012; revised April 1, 2013, April 29, 2014, and December 22, 2014; accepted December 24, 2014. Date of publication January 9, 2015; date of current version January 30, 2015. The work of H. Li was supported in part by the 973 Program under Contract 2015CB351803 and in part by NSFC under Contract 61325009 and Contract 61390514. The work of W. Zhou was supported in part by NSFC under Contract 61472378, in part by the Fundamental Research Funds for the Central Universities under Contract WK2100060014 and Contract WK2100060011, in part by the Start-Up Fund through the University of Science and Technology of China under Contract KY2100000048, and in part by the Intell Collaborative Research Institute on Mobile Networking and Computing Project. The work of R. Hong was supported in part by the Program for New Century Excellent Talents in University under Contract NCET-13-0764 and in part by NSFC under Contract 61472116. The work of Y. Lu was supported in part by the Texas State University Research Enhancement Program, Army Research Office (ARO), under Grant W911NF-12-1-0057, and in part by the National Science Foundation under Grant CRI 1305302. The work of Q. Tian was supported in part by ARO under Grant W911NF-12-1-0057 and in part by the Faculty Research Awards through NEC Laboratories of America. This work was supported by the National Natural Science Foundation of China (NSFC) under Contract 61429201. The associate editor coordinating the review of this manuscript and approving its publication was Prof. Dacheng Tao.

W. Zhou and H. Li are with the CAS Key Laboratory of Technology in Geo-Spatial Information Processing and Application System, Department of Electronic Engineering and Information Technology, University of Science and Technology of China, Hefei 230027, China (e-mail: zwg@ustc.edu.cn; libh@ustc.edu.cn).

R. Hong is with the Department of Computer Science, Hefei University of Technology, Hefei 230009, China (e-mail: hongrc.hfut@gmail.com).

Y. Lu is with the Department of Computer Science, Texas State University, San Marcos, TX 78666 USA (e-mail: yl12@txstate.edu).

Q. Tian is with the Department of Computer Science, University of Texas at San Antonio, San Antonio, TX 78249 USA (e-mail: qitian@cs.utsa.edu).

Index Terms—Large scale image retrieval, scalar quantization, binary SIFT, visual matching, feature filtering.

I. INTRODUCTION

T

HE last decade has witnessed the great advance in content based image retrieval on large scale image databases. Many state-of-the-art approaches utilize SIFT feature [1] to represent images and leverage the BoW model [2] to index large scale image dataset for scalable retrieval [3]. Some post-processing techniques, such as spatial verification [4]–[6] and query expansion [7], [8], are also explored to further boost the retrieval accuracy. In those approaches, one of the key steps is feature quantization, which generates visual words from the high dimensional SIFT features beforehand, and quantize features to the corresponding visual words for indexing and scalable search.

The most popular method for feature quantization is vector quantization. Originally used in lossy data compression, vector quantization divides a large set of training SIFT features into non-overlapped groups by clustering. Each feature group corresponds to a sub-space in the feature space, and is represented by its clustering center, which is called visual word [2]. All visual words constitute a visual vocabulary/codebook. Then, given a novel feature, vector quantization assigns it the visual word ID of the sub-space where the feature falls in. The most intuitive visual codebook generation approach is k-means clustering [2]. When the visual codebook size becomes very large (e.g. one million), it is infeasible to train the codebook with k-means, and hierarchical k-means [9] is more preferable to improve codebook generation speed and enhance feature quantization efficiency.

Traditional vector quantization involves several non-trivial issues. 1) Necessity of visual codebook training: visual codebook has to be trained in advance and the training process is computationally expensive especially with a large amount of sample features. For example, in order to train a large visual codebook containing one million visual words, usually ten times more SIFT feature samples may be needed, considering both feature space coverage and affordable memory size. However, for the SIFT descriptor space with as high as 128 dimensions, it is still unknown how many SIFT features are enough to capture the SIFT feature distribution of, say, one billion images. Even if the memory of computing servers would afford several orders of magnitude more training features, it would take intolerable time cost to finish the clustering for codebook generation. Although the visual codebook training is usually performed one-time and off-line, it is still more preferable to bypass such training
process. 2) **Limited reliability**: the effectiveness of codebook construction in vector quantization relies on the collection of image features and codebook generation methods. Different collections of image features may produce totally different codebooks. Even with the same collection of images and the same clustering methods, the generated codebook (clustering results) may be still different due to the variability of $k$-means. As a result, quantization error may be not easy to be efficiently controlled. 3) **Update inefficiency**: with many new features collected, the codebook/quantizer should be updated accordingly. However, in vector quantization, updating a large visual codebook needs a lot of effort. The huge amount of features has to be re-clustered, which is computationally inefficient.

To avoid the above problems, in this paper, a novel quantization strategy, scalar quantization, is proposed. Distinguished from the traditional vector quantization methods, the proposed scalar quantization approach does not involve any form of visual codebook training. Instead, it first transforms each feature to a bit-vector called Binary SIFT (BSIFT) with a quantizer, which is independent of collections of image features. The BSIFT generation is very efficient and requires low computational cost. The generated BSIFT achieves compact representation of the original SIFT descriptor, and is demonstrated to keep the discriminative power of the SIFT feature. Since our quantization method is independent of collections of images, there is no need to update our quantizer with new collected features.

Moreover, with the generated binary signature, we can index image features with the classic inverted file structure easily by extracting some bits from the quantized bit-vector to generate code word. And the remaining bits of the quantized bit-vector are stored in the inverted file list for matching verification.

Furthermore, to reduce the quantization loss, three schemes are presented. We propose query sensitive mask shielding scheme to mask unreliable bins and feature filtering scheme to remove unreliable features. Besides, we propose a soft-decision scheme for code word expansion by enumerating the nearest neighbors of code word. Consequently, more reliable BSIFT candidate matches are included for matching verification, which greatly boost the retrieval accuracy on large scale image database.

In this paper, we focus on the feature quantization step. To further improve the image retrieval performance, our approach can also be flexibly integrated with many other algorithms, such as descriptive feature selection [10], [11], weak geometric consistency [12], fast spatial matching [4], geometric verification [5], [13], query expansion [14], [15], contextual weighting [16] and learning [17], [18], etc.

Our approach does not involve any explicit codebook training. Also, there is no need for our approach to store a large codebook in memory for quantization. Such advantage makes our approach to be readily applied in image search based on mobile phones, which casts strict limit on memory usage.

The rest of the paper is organized as follows. Section II reviews related work in large scale image search. Section III discusses the proposed algorithm in details. Experimental results are given in Section IV. Finally, the conclusion is made in Section V.

## II. RELATED WORK

In large scale content based image search applications, Bag-of-Words (BoW) model based on local features has been widely adopted. Generally, in those BoW-based approaches, there are five key components: local feature representation, feature quantization, index strategy, retrieval scoring, and post-processing. In this section, we make a review of related work in each component.

### A. Local Feature Representation

Generally, local feature extraction involves two key steps, i.e. interest point detection and feature description. The detected interest points are expected to be highly repeatable over various changes [19]. Popular detectors include Difference of Gaussian (DoG) [1], MSER [20], Hessian affine [21], and FAST [22]. After interest point detection, a descriptor is extracted to capture the visual appearance of the local region centered at the interest point. Usually, the descriptor should be invariant to rotation and scale change, and also robust to affine distortion, addition of noise, and illumination changes, etc [23], [24]. The most popular choice with the above merits is SIFT feature [1]. As a variation, SURF [25] is demonstrated with comparable performance but better efficiency. Recently, binary feature BRIEF [26] and its variants, such as ORB [27], FREAK [28], and BRISK [29], have been proposed and have attracted lots of attentions in visual matching applications. With the advantage in efficiency, those binary features based on FAST [22] may also have great potential in large scale image search.

### B. Feature Quantization

Usually, hundreds or thousands of local features can be extracted from a single image. To achieve a compact representation, high dimensional local features are quantized to visual words, and an image can be represented as a “bag” of visual words [30]. Therefore, a visual codebook needs to be generated beforehand. The most intuitive visual codebook generation method is $k$-means [2], or hierarchical $k$-means [9] for large size visual codebook generation.

With visual codebook defined, feature quantization is to assign a visual word ID to each feature. The most naive choice is to find the closest (the most similar) visual word of a given feature by linear scan, which, however, suffers expensive computational cost. Usually, approximate nearest neighbor (ANN) search methods are adopted to speed up the searching process, with sacrifice of accuracy to some extent. In [1], a $k$-d tree structure [31] is utilized with a best-bin-first modification to find approximate nearest neighbors to the descriptor vector of the query. In [9], based on the hierarchical vocabulary tree, an efficient approximate nearest neighbor search is achieved by propagating the query feature vector from the root node down the tree by comparing the corresponding child nodes and choosing the closest one.

In [32], a $k$-d forest approximation algorithm is proposed with reduced time complexity. To reduce the quantization loss, a
descriptor-dependent soft assignment scheme [33] is proposed to map a feature vector to a weighted combination of multiple visual words. In [34], the high dimensional SIFT descriptor space is partitioned into regular lattices for the task of image classification with promising performance.

C. Index Strategy

Inspired by the success of text search engines, inverted file structure has been successfully used for large scale image search [2], [4], [5], [9], [14], [35], [36]. In essence, inverted file structure is a compact representation of a sparse matrix, where the row and the column denote visual word and image, respectively. In on-line retrieval, only those images sharing common visual words with the query image need to be checked [37]. Therefore, the number of candidate images to be compared is greatly reduced, achieving an efficient response.

In the inverted file structure, each visual word is followed by an inverted file list of entries. Each entry stores the ID of the image where the visual word appears, and some other clues for verification or similarity measurement. For instance, Bundled Feature [38] stores the x-order and y-order of each SIFT feature located in the bundled area. The geometric clues, such as feature position, scale, and orientation, are also stored in the inverted file list for geometric consistency verification [4], [5], [13], [38].

To further reduce the memory cost of inverted file structure, a visual word vector is mapped to a low-dimensional representation by a group of min-hash functions [18], [19]. Consequently, only a small constant amount of data per image need to be stored.

D. Retrieval Scoring

In the online retrieval stage, after identifying those relevant images of a query by looking up the index table, it is necessary to determine the relevant score of those target images to the query image. Usually, the relevance score is defined by the normalized distance between the BoW vectors of the query and the database images. When the codebook size is much larger than the local feature amount in images, the image vector by BoW is very sparse and we only need to check those visual words appearing in both images [9], which results in very efficient in implementation. To distinguish the significance of visual words in different images, term frequency (TF) and inverted document/image frequency (IDF) are widely applied in many existing state-of-the-art algorithms [2], [9], [33], [38]. In [38], the classic TF-IDF is further enhanced with a weighting term by matching bundled feature sets. In [39] and [40], Zheng et al propose a novel $L_p$-norm IDF to extend the classic IDF weighting scheme. In [5], the relevance score is simply defined by counting how many pairs of local feature are matches across two images. In [16], contextual weighting is introduced to improve the classic vocabulary tree approach. Statistics of neighboring descriptors both on the vocabulary tree and in image spatial domain are efficiently incorporated.

E. Post Processing

The initially returned result list can be further refined by exploring the spatial context or enhancing the original query. Spatial verification [4], [5], [13], [41], [42] and query expansion [8], [14] are two of the most successful post-processing techniques to boost the accuracy of large scale image search.

Spatial context is an important clue to remove false positive visual matches. Lots of work has been done on spatial verification. In [2], local spatial consistency is imposed to filter visual-word matches with low support. In [4], global spatial verification is performed based on a variation of RANSAC [43]. An affine model is estimated to filter local matches that fail to fit the model. In [5] and [13], the geometric context among local features is quantitatively encoded into binary maps. Then it recursively removes geometrically inconsistent matches by analyzing those maps.

Query expansion, leveraged from text retrieval, reissues the initially highly-ranked results to generate new queries. Some relevant features, which are not present in the original query, can be used to enrich the original query to further improve the recall performance. Several expansion strategies, such as average query expansion, transitive closure expansion, recursive average query expansion, intra-expansion, and inter-expansion, etc., have been discussed in [8] and [14].

In this paper, our focus is the feature quantization stage. Like many state-of-the-art algorithms, our approach represents images with the classic SIFT feature [1]. However, distinguished from those above methods based on vector quantization, we propose an efficient scalar quantization on SIFT feature and generate a new binary feature, i.e. BSIFT. As demonstrated, the generated BSIFT well keeps the feature distance of the original SIFT descriptors. Based on BSIFT, we generate a code word by simply selecting 32 bits and index database images by the classic inverted index structure. Our approach can also be enhanced by those retrieval scoring and post-processing techniques to further improve the image retrieval performance.

III. OUR APPROACH

We discuss our approach as follows. In Section III-A, we introduce our motivation. In Section III-B, we discuss our BSIFT generation scheme and demonstrate by an experimental study that BSIFT keeps the feature distance of the original SIFT feature. In Section III-C, we explain how to index large scale image database based on BSIFT. In Section III-D, we discuss how to reduce the quantization loss from three aspects. Finally, a summary of our quantization algorithm for image search is given in Section III-E.

A. Motivation

In essence, the key problem of image search is visual matching between images. When images are represented by local features, visual matching is achieved via feature matching between images. Intuitively, considering whether two features from different images are a pair of valid match, the most straightforward criterion is to check whether the distance between them is smaller than a predefined threshold [44]. In traditional Bag-of-Visual-Words based approach, feature matching is implicitly realized by checking whether two features are quantized to the same visual word [9]. However, we
frequently observe that, still many features with large distance from each other are quantized to the same visual word, while many other features with small distance from each other are quantized to different visual words. Such phenomenon easily causes the false positive and true negative [45]. To avoid such drawback, it is more preferable to verify feature matching by feature distance. Besides, since real-time response is a critical requirement in large scale image search, the matching verification should be performed very efficiently.

In visual matching based on SIFT descriptors, one effective criterion is to check the distance of the closest neighbor to that of the second-closest neighbor [1]. If the distance ratio is greater than a pre-defined threshold, the SIFT match is rejected as a false match. However, in large scale image search, given a local feature from a query image, it is infeasible to obtain the nearest and second nearest neighbors of local features in a target image. One feasible and reliable evidence that can be used is the $L_2$-distance from the nearest neighbors. If we can obtain an optimal threshold to distinguish true and false matches, we can exploit it to verify the $L_2$-distance between SIFT features.

In fact, in large scale image search, it is infeasible to directly compare the distance between the original SIFT features, since it is too memory-consuming to store the original SIFT features in the index structure, let alone the high computational cost of $L_2$-distance computing. One solution to address this problem is to approximate the original SIFT feature with a new compact feature and verify the feature distance on the new features. To achieve efficiency on feature distance verification, the new feature shall be binary signature, so the feature distance can be efficiently computed and measured by Hamming distance [37], [45]–[47]. With such motivation, we propose the Binary SIFT, as discussed in details in the next section.

### B. BSIFT Generation

High dimensional SIFT descriptors ($L_2$-normalized 128-D vectors [1]) are extracted from images for discrimination. Each dimension of the descriptor vector corresponds to a bin of the concatenated orientation histograms. Generally, similar SIFT features have relatively smaller distances than irrelevant features. Features from the same source, e.g., image patch, may not be exactly the same due to image noise. But their values on the 128 bins usually share some common patterns, e.g., the pair-wise differences between most of bins are similar and stable. Therefore, it can be easily extended that the differences between bins and a predefined threshold are stable for most bins. Based on such observation, we propose a scalar quantization strategy.

Given a SIFT feature vector $\mathbf{F} = (f_1, f_2, \ldots, f_{128})^T$, where $f_i \in \mathcal{R}$, we define a quantization function to transform $\mathbf{F}$ to a bit vector $\mathbf{b} = (b_1, b_2, \ldots, b_{128})^T$, as follows:

$$b_i = \begin{cases} 1, & \text{if } f_i > \hat{f} \\ 0, & \text{if } f_i \leq \hat{f} \end{cases} \quad (1 \leq i \leq 128), \quad (1)$$

where $\hat{f}$ is a threshold determined by the vector $\mathbf{F}$.

The threshold $\hat{f}$ is an important parameter, which determines the discriminative power of the quantization results. If the discriminative power of SIFT is well kept in scalar quantization, the Hamming distance between scalar vectors $\mathbf{b}$ should be consistent with the $L_2$ distance between original feature vectors $\mathbf{F}$. There may be many methods to choose the threshold $\hat{f}$, such as using the mean value of all bins in vector $\mathbf{b}$ as $\hat{f}$, or learning $\hat{f}$ from a training dataset. In this paper, we select $\hat{f}$ as the median value of vector $\mathbf{F}$. The philosophy behind it is that, the median value is relatively stable to change in some bins of a long vector. Another benefit from the median threshold selection is that, the obtained binary SIFT vector is implicitly normalized.

With each high dimensional feature quantized to a bit-stream vector, the feature comparison is transformed to the binary vector comparison, which can be efficiently accomplished by logical exclusive-OR operation and measured by Hamming distance.

To demonstrate that the discriminative power of SIFT descriptors is well kept in our scalar quantization, we have made a statistical study on $4.08 \times 10^{11}$ SIFT descriptor pairs, which include every SIFT pair extracted from image pairs randomly sampled from the UKBench dataset [9]. For each descriptor pair, its $L_2$ distance before scalar quantization and Hamming distance after scalar quantization are calculated. As shown in Fig. 1(a), the distribution of Hamming distances between these descriptors unsurprisingly exhibits a Gaussian-like distribution. From Fig. 1(b) and Fig. 1(c), it is observed that the Hamming distance between our quantized bit-vectors is consistent with the average $L_2$-distance, with relatively small standard deviation (computed on the unit-normalized descriptors). To further reduce the deviation, we use a variation of Eq. (1) and transform the descriptor vector to a 256-bit vector, which will be discussed at the end of this section.
Fig. 3. Statistics of SIFT descriptors. (a) A typical SIFT descriptor with features. The radius of the red circle centered at the key point is proportional to the SIFT feature’s characteristic scale. (b) top: the 128-D descriptor of the matched SIFT feature in the left image; middle: the 128-D descriptor of the matched SIFT feature in the right image; bottom: the XOR result of the binary SIFT features from the two matched SIFT features. The red horizontal lines in the top and bottom figure denote the median values of the two SIFT descriptors, respectively.

Fig. 2. Example of feature matches. (a) A local match between two images. The endpoints of the green line denote the key point positions of two SIFT features. The radius of the red circle centered at the key point is proportional to the SIFT feature’s characteristic scale. (b) top: the 128-D descriptor of the matched SIFT feature in the left image; middle: the 128-D descriptor of the matched SIFT feature in the right image; bottom: the XOR result of the matched SIFT feature in the left image; middle: the 128-D descriptor of the matched SIFT feature in the right image; bottom: the XOR result of the matched SIFT feature in the right image.

Fig. 4. Examples of local matching results based on 256-bit BSIFT. The Hamming distance threshold is selected as 24. No other geometric verification is involved. The lines endpoints denote the key point positions of two SIFT features. The radius of the red circle centered at the key point is proportional to the SIFT feature’s characteristic scale.

It is notable that our approach is different from the SIFT quantization methods proposed in lattice quantization [34] and Hamming Embedding [12]. In [34], the descriptor space is arbitrarily split along dimension axes into regular lattices. In [12], for each bin/dimension, a median value of all training features on that bin in the low dimensional space is computed for binarizing the corresponding dimension. Both two approaches ignore the unique property of every individual SIFT descriptor.

Fig. 2 shows a real instance of local descriptor match across two images with scalar quantization. From Fig. 2(b), it can be observed that these two SIFT descriptors have similar magnitude in the corresponding bins with some small variations before quantization. After scalar quantization, they differ from each other in nine bins. With a proper threshold, it can be easily determined whether the local match is true or false just by the exclusive-OR (XOR) operation between the quantized bit-vectors. Obviously, the error in the exclusive-OR result is likely to occur in those bins with magnitude around the median value. Intuitively, the median threshold could be increased to some upper level, which can make the Hamming distance between similar SIFT descriptors smaller. However, such modification will also reduce the Hamming distance between irrelevant descriptors and weaken the discriminative power of the BSIFT.

A statistical study on the distribution of median value of SIFT descriptor has been performed. 100 million SIFT descriptors are sampled from a large dataset, and the median value of each 128-D descriptor vector is computed. As shown in Fig. 3, the median value of most SIFT descriptors is relatively small, around 9, but the maximum magnitude in some bins still can reach more than 140. This may incur potential quantization loss since those bins with magnitude above the median are not well distinguished. To address this issue, the same scalar quantization strategy could be conducted again on those bins with magnitude above the median. Intuitively, such operation can be performed recursively. However, it will cause additional storage cost. In our implementation, we only perform the scalar quantization twice, i.e., first on the whole 128 elements, and then on those elements with magnitude above the median value. Consequently, a SIFT descriptor $F = (f_1, f_2, \ldots, f_{128})^T$ is quantized to a 256-bit vector $b = (b_1, b_2, \ldots, b_{256})^T$, as follows:

$$
(b_i, b_{i+128}) = \begin{cases} 
(1, 1), & \text{if } f_i > \hat{f}_2 \\
(1, 0), & \text{if } \hat{f}_1 < f_i \leq \hat{f}_2 \quad (1 \leq i \leq 128), \\
(0, 0), & \text{if } f_i \leq \hat{f}_1 
\end{cases}
$$

(2)

where $\hat{f}_1 = \frac{g_1 + g_{128}}{2}$, $\hat{f}_2 = \frac{g_1 + g_{128}}{2}$, $(g_1, g_2, \ldots, g_{128})$ is the sorted vector from $(f_1, f_2, \ldots, f_{128})$ in the descending order. With Eq. (2), each dimension of SIFT descriptor is divided into three parts, and two bits are used to encode each part. It should be noted that there is still some redundancy with such representation. According to Shannon’s information theory [48], the bit rate of binary SIFT by Eq. (2) is 1.5 bits per bin. Therefore, it may only take 196 bits to represent each BSIFT if compression is required.

Some sample results of image local matching based on BSIFT by Eq. (2) are shown in Fig. 4, from which we
can observe that those true local matches are satisfactorily identified even without introducing any false match.

With scalar quantization by Eq. (2), the comparison of SIFT descriptors by $L_2$-distance is captured by the Hamming distance of the corresponding 256-bit vectors. Since our target is large-scale image search, how to adapt our scalar quantization result to the classic inverted file structure for scalable image search needs to be explored.

C. Indexing With Inverted File Structure

In image search, the problem of feature matching between images can be regarded as finding feature’s nearest or approximately nearest neighbors within a certain range. When the feature amount becomes very large, say, over one billion, it is too computationally expensive to find the nearest neighbors by linearly comparing all features’ binary vectors. To address this issue, inverted file structure, leveraged from text retrieval, can be used for scalable indexing of large-scale image dataset.

In traditional inverted file structure for image search, a group of visual words are pre-trained. And each visual word is followed with an entry list of image features, which are quantized to this followed visual word. Each indexed feature in the list records its image ID and some other clues.

To adapt to the classic inverted file structure to index image features, we define code word by the first $t$ bits of the binary code generated by scalar quantization. Then, the rest bits of features are recorded in the entry list of the corresponding code word for later verification. A toy example is shown in Fig. 5. Intuitively, if a code word is represented with $t$ bits, the total number of code words could be amounted up to $2^t$. However, it is found from experiments that, when $t$ increases to 20 and larger, the amount of non-empty code words becomes much smaller than $2^t$, as shown in Fig. 6(a). For example, when $t$ increases to 32, the total number of code words could be up to $2^{32} \approx 4 \times 10^9$ (4 billion). However, the number of unique code words generated by scalar quantization (on one million image database) is even much less than $10^8$.

Generally, the more code words are generated, the shorter the average length of indexed feature list becomes, and the less the time cost is needed to query a new feature. However, in our method, we will introduce a soft quantization scheme (Section III-D2) to expand more code words for each query feature. And the number of expanded indexed feature lists is polynomial to $t$. To make a tradeoff, in our experiments, we select $t = 32$, and 46.5 million code words are obtained.

Fig. 5. A toy example of image feature indexed with inverted file structure. The scalar quantization result of the indexed feature is an 8-bit vector (1001 0101). The first three bits denote its code word ID (100), and the remaining 5 bits (10101) are stored in the inverted file list.

Fig. 6. (a) The amount of unique code words (top bits from 256-bit vector) for different on 1-million image database. (b) Frequency of code words among one million images before application of a stop list.

Fig. 6(b) shows the distribution of code word occurrence on a one-million-image database. It can be observed that, of the 46.5 million code words, only the top few thousand code words exhibit very high frequency. Those code words are prevalent in many images, and their distinctive power is weak. As suggested by [2], we apply a stop-list to ignore those code words that frequently occur in the database images. Experiments reveal that a proper stop-list may not affect the search accuracy, but helps avoid checking many long inverted lists and achieve gain in efficiency.

Once all features of an image dataset have been indexed with the inverted file structure, given a new SIFT descriptor, it will be first quantized to a 256-bit vector with scalar quantization. Then through the top 32 bits, the corresponding code word can be located. And only the indexed features following the matched code word will be checked. Therefore, the searching space is greatly reduced. Finally, the exclusive-OR operation is performed between the remained 224 bits of the query vector and those of indexed features recorded in the entry list of the matched code word. A threshold $\kappa$ on the Hamming distance between the 256-bit vectors needs to be set for true-match judgment, such that those matches with Hamming distance no larger than $\kappa$ will be accepted as true matches. The impact of $\kappa$ will be studied in Section IV-A.

D. Reduction of Quantization Error

Since our scalar quantization is based on a hard-decision strategy for thresholding, it will unavoidably incur quantization error. To address the quantization loss, three schemes are presented. Firstly, we propose to filter unreliable SIFT features by checking the median value. Secondly, we propose a code word expansion strategy to include more candidate words for verification. Last but not least, we propose a query sensitive masking scheme to shield those unreliable bins.

1) SIFT Filtering by Median Value: In visual matching based on BSIFT, we frequently observe that, SIFT features with key points along edges can easily cause local matches across images. This is because edge patches are prevalent in many images and many SIFT features are detected along edges, but the discriminative power of those SIFT features are relatively weak. In SIFT key point detection, the difference-of-Gaussian function generates strong response along edges, even though the location along the edge is poorly determined and therefore is unstable to small amount of noise [1].
In [1], Lowe tried to remove such SIFT key points by checking the ratio of principal curvatures computed at the key points. Although many key points can be eliminated in this way, some key points along the edges are still kept and the corresponding SIFT features harass the visual matching, which degrade the image search accuracy.

Generally, for SIFT features with key point along edges, the energy of SIFT descriptor vectors is concentrated on few bins. In other words, the coefficients in most bins are of low magnitude. Accordingly, the median values of those descriptors are expected to be small. Based on this observation, we define a criterion to filter those SIFT features: the median of the feature vector \( \mathbf{v} \) is no larger than a threshold \( h \), i.e.,

\[
\text{median}(\mathbf{v}) \leq h. \tag{3}
\]

The impact of parameter \( h \) will be studied in Section IV-A.

It should be noted that, in image search, our feature filtering strategy can be applied not only to query feature but also to those database features. In other words, we can even perform such feature filtering in the image indexing stage. That is, we will ignore those features satisfying Eq. (3). In this way, a large amount of local features are free from indexing, saving considerable memory cost. In Section IV-A, an experimental study reveals that the optimal value of \( h \) is selected as 5.5, which greatly improves both the retrieval accuracy and efficiency. And correspondingly, about 27.3% features of database images will be filtered.

2) Code Word Expansion: In Section III-C, we define the code word by the top 32 bits of the BSIFT from scalar quantization. However, such simple processing will exclude some candidate features that have some flipping bits among the top 32 bits (e.g., 0 changes to 1) due to noise. To address this issue, we propose a soft-decision strategy to reduce such quantization loss. Assuming such flipping happens only to very few dimensions, features before and after the flipping should be still very similar, i.e., small Hamming distance. To identify these candidate features, it is desired to quickly enumerate all of its possible nearest neighbors (code words) within a predefined Hamming distance \( d \), just by alternatively flipping some bits [49]. This is equivalent to a tolerant expansion of the original code word. The impact of expansion-bit number \( d \) will be studied in Section IV-A.

As shown in the toy example in Fig. 7, the code word of a new query feature is a bit-vector 100, i.e., CW4 in pink color. To identify all of candidate features, its possible nearest neighbors (e.g., Hamming distance \( d = 1 \)) will be obtained by flipping one bit in turn, which generates three additional code words (in green color): CW0 (000), CW5 (101) and CW6 (110). These code words are nearest neighbors of CW4 in the Hamming space. Then, besides CW4, the indexed feature lists of these three expanded code words will be also considered as candidate matches, and all features in these expanded lists will be further compared on their rest bit-codes.

3) Query Sensitive Mask Generation: From the scalar quantization function Eq. (2), we can see that the binary bits are sensitive in those bins sorted close to the thresholds. Even small noise can cause the magnitude to fluctuate across those thresholds and reverse the corresponding bits. In other words, those bits with magnitude close to the thresholds are not reliable and should be shielded. With such motivation, we define a mask vector \( \mathbf{M} = (m_1, m_2, \ldots, m_{256}) \) for a query BSIFT vector \( \mathbf{b}^q = (v_1, v_2, \ldots, v_{256})^T \) to shield those bits. Before that, we first check the absolute distance from all bin values to the median threshold \( f_1 \) defined in Eq. (1), and sort those distance values accordingly. Those bins corresponding to the top \( \gamma \) smallest values are assembled into a set \( S_1 \). Similarly, we do similar operation with threshold \( f_2 \), and obtain the set \( S_2 \), which corresponds to those \( \lceil \frac{\gamma}{2} \rceil \) bins with the smallest distance values. Then, with \( S_1 \) and \( S_2 \), we formulate the mask vector as follows:

\[
(m_i, m_{i+128}) = \begin{cases} 
(0, 1), & \text{if } i \in S_1 \\
(1, 0), & \text{if } i \in S_2 \\
(1, 1), & \text{otherwise}
\end{cases} \tag{4}
\]

In on-line query stage, we compare the bit vectors of a query feature \( \mathbf{b}^q \) and a database feature \( \mathbf{b}^d \) as follows:

\[
\text{dist}(\mathbf{v}^q, \mathbf{v}^d) = H(\mathbf{M} \& (\mathbf{b}^q \oplus \mathbf{b}^d)),
\]

where & denotes bit-wise logical AND operation, \( \oplus \) denotes bit-wise exclusive OR operation, and \( H(\mathbf{x}) \) denotes the amount of logic “1” in bit vector \( \mathbf{x} \). By Eq. (5), those bins with magnitude close to the two thresholds are shielded and therefore are not involved in the Hamming distance computation. In our experiments, we set \( \gamma = 6 \).

E. Algorithm Summary

Overall, the proposed scalar quantization for large scale image search consists of two stages: off-line indexing and on-line querying. The general steps of the off-line indexing are described as follows:

1) Given a 128-D SIFT descriptor from an image to be indexed, if it does not satisfy Eq. (3), convert it to a 256-bit BSIFT vector by Eq. (2);
2) Identify code word ID \( V_d \) by the first 32 bits of BSIFT vector.
We formulate the image retrieval as a voting problem. Each matching result between the query image and target images is ranked by their similarity scores and returned to the users as retrieval results.

For each feature in the inverted image list linked to $V_q$, compare its indexed 224-bit vector with the query feature. If the total Hamming distance in 256-bit by Eq. (5) is no greater than $\kappa$, accept the indexed feature as true match.

Expand $V_q$ to include its nearest code words $V_q^i$, $(i = 1, 2, \cdots)$ with Hamming distance no greater than $d$ by Eq. (5). For each indexed feature in the inverted image list linked to each $V_q^i$, compare its indexed 224-bit vector with the query feature. If the total Hamming distance in 256-bit by Eq. (5) is no greater than $\kappa$, accept the indexed feature as true match.

IV. EXPERIMENTS

We evaluate our approach on two public datasets, i.e., DupImage dataset [13] and UKBench dataset [9]. We firstly take the DupImage dataset as the ground-truth dataset, which contains 1104 images collected from 33 groups, including Mona Lisa, KFC logo, American Gothic Painting, Seven-eleven logo, etc. We mix the DupImage dataset with a basic database, which is built by crawling one million images from the Web. Then, we study the impact of three parameters in terms of accuracy and efficiency in our approach: Hamming distance threshold $\kappa$, expansion-bit number $d$, and median threshold $h$. We first ignore $h$ (set $h < 0$) and just investigate the mAP performance and efficiency under different parameter settings of $\kappa$ and $d$ on the one-million image database. After selecting the optimal values of $\kappa$ and $d$, we then investigate the impact of $h$ on retrieval accuracy and efficiency.

The impact results of $\kappa$ and $d$ are shown in Fig. 8. From Fig. 8(a), it can be observed that when the Hamming distance threshold $\kappa$ increases, the mAP performance first increases and then keeps stable and gradually drops a little after it reaches the peak, where $\kappa = 24$. This is intuitive, since increasing $\kappa$ always includes more candidate true matches, but when $\kappa$ is too large, many noisy matches are also included and therefore degrade the search performance. On the other hand, when expansion-bit number $d$ increases, the mAP gradually increases. This is due to the fact that more candidate inverted file lists are involved in matching verification, and more true matches will be kept.

In terms of efficiency, as shown in Fig. 8(b), the average time cost per query increases as $\kappa$ increases. This is due to the fact that, when $\kappa$ is larger, we have to make more exclusive-OR operations, until the Hamming distance between two 224-bit vectors is above a threshold. As $d$ increases, the querying time cost rises significantly. This is because the number of expanded code words or inverted file lists is exponential to the expansion-bit number $d$. Considering the tradeoff between mAP performance and time cost, $\kappa$ is set as 24 and $d$ is set as 2 in the rest experiments.

The third parameter $h$ in Section III-D1 works as a threshold to filter unreliable features. The impact results of $h$ are shown in Fig. 9. From Fig. 9(a), we can see that, when $h$ increases, the corresponding mAP performance first increases and then reaches the peak when $h$ is equal to 5.5. After that, the mAP decreases sharply. This is due to the fact that, when $h$ increases from a small value, fewer and fewer unreliable features are remained, which benefits the image search accuracy. However, when $h$ increases, the impact of $h$ on retrieval accuracy and efficiency will be shown in the next section.

A. Parameter Analysis

We study three parameters in terms of accuracy and efficiency in our approach: Hamming distance threshold $\kappa$, expansion-bit number $d$, and median threshold $h$. We first ignore $h$ (set $h < 0$) and just investigate the mAP performance and efficiency under different parameter settings of $\kappa$ and $d$ on the one-million image database. After selecting the optimal values of $\kappa$ and $d$, we then investigate the impact of $h$ on retrieval accuracy and efficiency.

In terms of efficiency, as shown in Fig. 8(b), the average time cost per query increases as $\kappa$ increases. This is due to the fact that, when $\kappa$ is larger, we have to make more exclusive-OR operations, until the Hamming distance between two 224-bit vectors is above a threshold. As $d$ increases, the querying time cost rises significantly. This is because the number of expanded code words or inverted file lists is exponential to the expansion-bit number $d$. Considering the tradeoff between mAP performance and time cost, $\kappa$ is set as 24 and $d$ is set as 2 in the rest experiments.

The third parameter $h$ in Section III-D1 works as a threshold to filter unreliable features. The impact results of $h$ are shown in Fig. 9. From Fig. 9(a), we can see that, when $h$ increases, the corresponding mAP performance first increases and then reaches the peak when $h$ is equal to 5.5. After that, the mAP decreases sharply. This is due to the fact that, when $h$ increases from a small value, fewer and fewer unreliable features are remained, which benefits the image search accuracy. However, when $h$ increases, the impact of $h$ on retrieval accuracy and efficiency will be shown in the next section.

1 http://vis.uky.edu/~stewe/ukbench/data/
is too large, some reliable features are also filtered, which degrades the mAP performance. On the other hand, the average time cost per query gradually decreases when $h$ increases (Fig. 9(b)). To make a tradeoff in accuracy and efficiency, we select $h$ as 5.5, which is used in the following experiments.

Since the database features with median less than $h$ can be filtered, we can save the memory cost on those features. To quantitatively study how much memory can be saved, we make a statistical study on the median value of 100 million original SIFT descriptors (128-D). The probability density distribution of the median magnitude is shown in Fig. 3(b), and the probability that the median of a feature is below a magnitude is shown in Fig. 9(c). From Fig. 9(c), we can observe that, as much as 27.3% index memory can be saved if those features with magnitude no larger than 5.5 are discarded.

**B. Evaluation on DupImage Dataset**

1) **Comparison Algorithms:** We compare our approach with several related feature quantization algorithms in large scale image search based on the original SIFT feature. The BoW approach with the visual vocabulary tree [9] is selected as the baseline method. We test various sizes of visual codebook, and the one-million visual codebook (a vocabulary tree with branch factor as 10 and level number as 6) gives the best overall performance. As suggested in [2] and [9], the stop-list strategy is also adopted to improve efficiency. To enhance the baseline, soft assignment [33] is also compared.

Soft assignment [33] identifies a local feature with a weighted combination of three nearby visual words. We select the same one-million visual words (leaf nodes of the visual vocabulary tree) as used in the baseline approach. We use the default parameters setting in [33]. The “nearby” visual words for a given feature are found by the approximate nearest neighbor search algorithm $k$-d tree [31], [50] with a public library.

In our approach, it is enhanced with three schemes discussed in Sec. III-D. Of our three enhanced schemes, the code word expansion aims to improve the recall, while the other two schemes target on improving the precision. To investigate their contribution and impact on the retrieval framework, we also evaluate a simplified version of our approach denoted as “BSIFT”, which is equipped with only code word expansion but omits the median-value based feature filtering and query-sensitive mask strategy.

On the other hand, several local binary features, such as ORB [27], FREAK [28], and BRISK [29], have been proposed recently with promising performance in image matching. To compare their potential with BSIFT in image search, we replace BSIFT with each of the above three local binary features in our index and retrieval framework and denote the corresponding method as ORB, FREAK, and BRISK. For each local binary feature, we take the first 32 bits as code words for indexing and the remained bits are stored in the inverted list entries. In the on-line retrieval stage, expansion-bit number $d$ is selected based on the tradeoff between accuracy and time efficiency. For each binary feature, we test a series of values on the Hamming distance threshold $\kappa$ and select the one which achieves the best performance. According to the experimental study, the optimal value of $\kappa$ for ORB, FREAK, and Brisk is selected as 44, 60, and 100, respectively.

![Fig. 10. Performance (mAP) comparison of different methods with different database sizes. “BSIFT” corresponds to the simplified version of our approach enhanced only with code word expansion but without filtering by median value or query sensitive mask.](image)

<table>
<thead>
<tr>
<th>Binary Feature</th>
<th>ORB</th>
<th>FREAK</th>
<th>BRISK</th>
<th>BSIFT</th>
<th>our approach</th>
</tr>
</thead>
<tbody>
<tr>
<td>mAP</td>
<td>0.28</td>
<td>0.10</td>
<td>0.18</td>
<td>0.34</td>
<td>0.39</td>
</tr>
<tr>
<td>Time (second)</td>
<td>0.11</td>
<td>0.58</td>
<td>0.51</td>
<td>0.30</td>
<td>0.31</td>
</tr>
</tbody>
</table>
TABLE II
TIME COST TO INDEX ONE MILLION SIFT FEATURES FOR ALL APPROACHES IN OFF-LINE STAGE

<table>
<thead>
<tr>
<th>Method</th>
<th>baseline</th>
<th>soft assignment</th>
<th>ORB</th>
<th>FREAK</th>
<th>BRISK</th>
<th>BSIFT</th>
<th>our approach</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time (second)</td>
<td>53.72</td>
<td>771.09</td>
<td>5.89</td>
<td>10.80</td>
<td>10.24</td>
<td>18.86</td>
<td>18.86</td>
</tr>
</tbody>
</table>

2) Accuracy: From Fig. 10, it can be observed that our approach outperforms the other three retrieval methods on large image databases. On the 1-million dataset, the mAP of the baseline is 0.38. Our approach hits 0.59, a relatively 55.2% improvement, while the mAP of BSIFT still reaches 0.54. The soft assignment approach reaches an mAP of 0.48. Compared with soft assignment, our approach still enjoys a relatively 22.9% improvement. It is interesting to note that, when the database size decreases to 50K, the accuracy improvement of our approach over soft assignment becomes smaller. This is due to our tradeoff selection on the expansion bit number and efficiency. Besides, we also compare the average precision of each query on the one-million image database. As revealed in Fig. 11, our approach achieves steady accuracy improvement over the baseline approach. Compared with the soft assignment approach, our method achieves better performance on the majority of the query images.

Besides, we also compare different local binary features based on our framework for image search. As shown in Table I, all the other three binary features, i.e., ORB, FREAK and BRISK, perform much worse than our approach in mAP. That is mainly due to two reasons. Firstly, those binary local features are based on the FAST corner detector [22] and extracted directly from local patch by simply comparing sampling pixels. As a result, their discriminative power is not as good as SIFT and its variant feature BSIFT. Secondly, the suitable Hamming distances for those binary features to distinguish true matches from false matches are relatively large. As a result, with our framework to index each binary feature with the first 32 bits, the recall rate of true matches is very low with the constraint of expansion-bit number $d$ for efficiency consideration.

3) Efficiency: The experiments are performed on a server with 3.4 GHz CPU. We compare efficiency in both off-line indexing and on-line query. From Table II, it is observed that our approach is very efficient in indexing image features. It takes our approach 18.86 seconds to index one million SIFT features, which is 2 times and 40 times faster than the baseline and soft assignment approach, respectively. The other three local binary features, i.e. ORB, FREAK and BRISK, are even more efficient than our approach based on BSIFT. This is mainly due to that those binary features do not involve any quantization step.

Table III shows the average time cost per query of all retrieval approaches. It should be noted that the time cost of SIFT feature extraction is not included for all approaches. It takes the baseline 0.12 second in average to perform one query. Soft assignment is the most time-consuming approach, consuming 0.52 second in average per query. Although our approach costs more time than the baseline approach, it may still meet users’ expectation of fast response time (average 0.31 second per query) but with much higher search accuracy. Our approach is more efficient than the soft assignment approach, with 0.21 second less in average per query. This is partly due to the fact that lots of features are filtered without verification in our approach. The retrieval time cost of all query images is illustrated in Fig. 12. While our approach is more time-consuming than the baseline, it is much more efficient than the soft assignment approach. Although our approach involves with both feature filtering and query-sensitive mask enhancement, its time efficiency is still comparable with the BSIFT approach. This is due to the fact that a considerable portion of SIFT features are filtered out during the indexing stage and don’t need to be verified in the retrieval stage, which compensates the time consumption for query-sensitive mask based verification.

It should be noted that a distinctive characteristic of our approach from other three comparison methods is that,
no visual codebook is needed to be trained before feature quantization, which could save a lot of computational efforts. In contrast, the comparison algorithms have to train a large visual codebook containing as many as one million visual words, which usually costs tens of hours. In order to train a visual codebook of one million in size, usually about more than 10 million SIFT descriptors are needed as training samples. However, even with so many training samples, it is still unclear whether these training samples are enough to generate desired visual words to capture the sample distribution in the so large 128-D descriptor space. Moreover, when more and more new features are indexed, it may be necessary to update the visual codebook accordingly (e.g., re-cluster all the features), which is always time consuming and computationally expensive. On the contrary, our approach just needs to incrementally add new code words to the existing code word set.

4) Memory Cost: We compare memory cost of all approaches on both indexed feature and quantizer, as listed in Table IV. In terms of memory cost per indexed feature, for each feature, the baseline approach needs 4 bytes to store image ID and another 4 bytes to store the TF-IDF weight. The soft assignment has to store each indexed feature in three visual word lists, therefore it costs 24 bytes, three times the memory cost of the baseline approach. Compared with the above methods, our approach consumes more memory. It takes 4 bytes to store image ID and additional 28 bytes to store another 224 bits from quantization results. The three local binary features, i.e., ORB, FREAK, and BRISK, take the similar index structure as our BSIFT and differ only in the bit length. Considering that the bit length values of ORB, FREAK and BRISK are 256, 512, and 512, their memory cost for each index feature is 32 bytes, 64 bytes and 64 bytes, respectively.

Besides indexed feature, all the three comparison retrieval methods have to load a large quantizer into main memory. A hierarchical visual vocabulary tree (about 142M bytes) is required for the baseline. As for soft assignment approach, besides the visual words (leaf nodes of the vocabulary tree, 128M bytes), it also needs to generate a k-d tree (about 378M bytes) to quantize features. In contrast, our approach based on BSIFT or the other three local binary features does not need memory cost on quantizer.

As revealed by Fig. 9(c), when filtering SIFT features with median value no larger than 5.5, about 27.3% features will be filtered. Those filtered features will be ignored when constructing the index file and therefore, considerable memory will be saved.

C. Evaluation on UKBench Dataset

We also evaluate our approach on the public UKBench dataset [9]. This dataset contains 10200 images from 2550 object or scene groups. Each group consists of four images taken in different views or imaging conditions. In terms of accuracy measurement, mean average precision (mAP) is used as evaluation metric.

We use the same parameters setting as selected in Section IV-A. The retrieval results in terms of mAP and efficiency are compared in Table V. The index efficiency and memory cost can be referred to Table 1 and Table 2. From Table 3, we can observe that the mAP of our approach is 0.811, which is better than the baseline approach. It is interesting to note that the “soft assignment” approach achieves the best mAP performance on this dataset. Such observation is different from that on the DupImage dataset in Section IV-B. This is because the size of this dataset is relatively small, which favors codebook-training based approach. Besides, unlike the DupImage dataset, many images in the UKBench dataset suffer from large view changes. Such image distortion from view changes will cause a lot of severe variations on SIFT coefficients, which consequently deteriorate the loss in our scalar quantization. The three binary local features based on our framework perform even worse than the baseline approach, which suggests that those binary features are not as discriminative as BSIFT and may not be suitable to be directly applied to the image retrieval task on our framework.

In terms of efficiency on this dataset, soft assignment approach costs 0.63 second per query, which is about one order more than the other comparison approaches. This is because soft assignment approach has to spend considerable time cost in searching the approximate nearest neighbors for each query feature in a large k-d tree, which is independent of the index image database size.

V. CONCLUSION

In this paper, a novel scalar quantization scheme is proposed on SIFT descriptor for large scale image search. Scalar quantization quantizes a SIFT descriptor to BSIFT,
which can be easily adapted to the classic inverted file structure for indexing. Distinguished from the traditional approaches, the proposed method involves no visual codebook training. The quantizer is defined by an individual feature itself and is independent of collections of images. Further, three schemes are discussed for reduction of quantization error. Experiments on large scale image search demonstrate the effectiveness of scalar quantization on two public image datasets.

Our proposed BSIFT may also be combined with the traditional codebook based approaches for large scale image search. On the other hand, since feature comparison in BSIFT is much more efficient than SIFT, BSIFT can also be used to feed the hierarchical $k$-means to build a flatter hierarchical visual vocabulary tree, so as to reduce the quantization loss.

In the future, investigation will be performed on developing more compact bit-vector representation in scalar quantization. Moreover, the flipping behavior of bit-vectors of similar SIFT descriptors will be explored. Some insights are expected to be obtained from this study, which may be beneficial to narrow search scope in soft quantization step and consequently improve retrieval efficiency. Further, various choices for threshold selection in scalar quantization will be studied.

REFERENCES


Richang Hong (M’14) is currently a Professor with the Hefei University of Technology, Hefei, China. He received the Ph.D. degree from the University of Science and Technology of China, Hefei, in 2008. He was a Research Fellow with the School of Computing, National University of Singapore, Singapore, from 2008 to 2010. His research interests include multimedia content analysis and multimedia question answering. He has co-authored over 60 publications in these areas. He was a recipient of the Best Paper Award in the Association for Computing Machinery (ACM) Multimedia 2010 Conference. He is a member of the ACM.

Yijuan Lu is currently an Associate Professor with the Department of Computer Science, Texas State University, San Marcos, TX, USA. She received the Ph.D. degree in computer science from the University of Texas at San Antonio, San Antonio, TX, USA, in 2008. Her current research includes multimedia information retrieval, computer vision, and machine learning. She has authored extensively and has served on Program and Organizing Committee for many international conferences. She was a recipient of the 2013 ICME Best Paper Award, the 2012 ICMCS Best Paper Award, and one of the top winners of the 2013 Eurographics SHERC competitions in large-scale sketch-based 3D retrieval track, range scan track, and low-cost depth-sensing camera track. She received the 2014 College Achievement Award, the 2012 Dean Nominee for the Texas State Presidential Award for Excellence in Scholarly/Creative Activities, and the Nominee for the 2008 Microsoft Research Faculty Summit. Her research has been supported by the National Science Foundation, Texas Department of Transportation, the Department of Defense, Army Research, and Texas State.

Qi Tian (M’96–SM’03) received the B.E. degree in electronic engineering from Tsinghua University, Beijing, China, in 1992, the M.S. degree in electrical and computer engineering from Drexel University, Philadelphia, PA, USA, in 1996, and the Ph.D. degree in electrical and computer engineering from the University of Illinois at Urbana—Champaign, Champaign, IL, USA, in 2002. He is currently a Professor with the Department of Computer Science, University of Texas at San Antonio (UTSA), San Antonio, TX, USA. His research interests include multimedia information retrieval and computer vision. He has authored over 260 refereed journal and conference papers. His research projects were supported by the National Science Foundation, the Army Research Office, the Directorate of Health Services, the San Antonio Life Sciences Institute, the Center for Integrated Agricultural Systems, and UTSA. He received the Faculty Research Awards from Google, Mountain View, CA, USA, NEC Laboratories of America, Princeton, NJ, USA, FX Palo Alto Laboratory, Inc., Palo Alto, CA, USA, Akira Media Systems, Palo Alto, and HP Labs, Palo Alto. He took a one-year faculty leave with Microsoft Research Asia, from 2008 to 2009. He co-authored the papers which received the ACM ICMCS 2012 Best Paper Award, the MMM 2013 Best Paper Award, and the PCM 2013 Best Paper Award, the Top 10% Paper Award in MMSP 2011, the Best Student Paper Award in ICASSP 2006, and the co-author of the Best Paper Candidate Award in PCM 2007. He received the 2010 ACM Service Award.

Dr. Tian has served as the Program Chair, an Organization Committee Member, and a Technical Committee Member for numerous IEEE and ACM Conferences, including ACM Multimedia, SIGIR, ICCV, and ICME. He is on the Editorial Board of the IEEE TRANSACTIONS ON MULTIMEDIA, IEEE TRANSACTIONS ON CIRCUIT AND SYSTEMS FOR VIDEO TECHNOLOGY, Multimedia System journal, the Journal of Multimedia, and the Journal of Machine Visions and Applications. He is also the Guest Editor of IEEE TRANSACTIONS ON MULTIMEDIA, Journal of Computer Vision and Image Understanding, Pattern Recognition Letters, EURASIP Journal on Advances in Signal Processing, and the Journal of Visual Communication and Image Representation.

Houqiang Li (S’12) received the B.S., M.Eng., and Ph.D. degrees in electronic engineering from the University of Science and Technology of China (USTC), Hefei, China, in 2011. He was a Research Intern with the Internet Media Group, Microsoft Research Asia, Beijing, China, from 2008 to 2009. From 2011 to 2013, he was a Post-Doctoral Researcher with the Department of Computer Science, University of Texas at San Antonio, San Antonio, TX, USA. He is currently an Associate Professor with the Department of Electronic Engineering and Information Science, USTC. His research interest is mainly focused on multimedia information retrieval.

Wengang Zhou received the Ph.D. degree from the Department of Electronic Engineering and Information Science, University of Science and Technology of China (USTC), Hefei, China, in 2011. He was a Research Intern with the Internet Media Group, Microsoft Research Asia, Beijing, China, from 2008 to 2009. From 2011 to 2013, he was a Post-Doctoral Researcher with the Department of Computer Science, University of Texas at San Antonio, San Antonio, TX, USA. He is currently an Associate Professor with the Department of Electronic Engineering and Information Science, USTC. His research interest is mainly focused on multimedia information retrieval.