Spatial Coding for Large Scale Partial-Duplicate Web Image Search

Wengang Zhou[1], Yijuan Lu[2], Houqiang Li[1], Yibing Song[1], Qi Tian[3]
Dept. of EEIS, University of Science and Technology of China[1], Hefei, P.R. China
Dept. of Computer Science, Texas State University at San Marcos[2], Texas, TX 78666
Dept. of Computer Science, University of Texas at San Antonio[3], Texas, TX 78249
zhwg@mail.ustc.edu.cn[1], yl12@txstate.edu[2], lihq@ustc.edu.cn[1], ybsong@mail.ustc.edu.cn[1], qitian@cs.utsa.edu[3]

ABSTRACT
The state-of-the-art image retrieval approaches represent images with a high dimensional vector of visual words by quantizing local features, such as SIFT, in the descriptor space. The geometric clues among visual words in an image is usually ignored or exploited for full geometric verification, which is computationally expensive. In this paper, we focus on partial-duplicate web image retrieval, and propose a novel scheme, spatial coding, to encode the spatial relationships among local features in an image. Our spatial coding is both efficient and effective to discover false matches of local features between images, and can greatly improve retrieval performance. Experiments in partial-duplicate web image search, using a database of one million images, reveal that our approach achieves a 53% improvement in mean average precision and 46% reduction in time cost over the baseline bag-of-words approach.

Categories and Subject Descriptors
I.2.10 [Vision and Scene Understanding]: VISION

General Terms
Algorithms, Experimentation, Verification.

Keywords
Image retrieval, partial-duplicate, large scale, orientation quantization, spatial coding.

1. INTRODUCTION
Given a query image, our target is to find its partial-duplicate versions in a large web image database. There are many applications of such a system, for instance, finding out where an image is derived from and getting more information about it, tracking the appearance of an image online, detecting image copyright violation, discovering modified or edited versions of an image, and so on.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
MM'10, October 25–29, 2010, Firenze, Italy.
Copyright 2010 ACM 978-1-60558-933-6/10/10…$10.00.

Figure 1. Examples of partially duplicated web images.

In image-based object retrieval, the main challenge is image variation due to 3D view-point change, illumination change, or object-class variability [8]. Partial-duplicate web image retrieval differs in that the target images are usually obtained by editing the original image with changes in color, scale, partial occlusion, *etc*. Some instances of partial-duplicate web images are shown in Fig. 1. In partial-duplicate web images, different parts are often cropped from the original image and pasted in the target image with modifications. The result is a partial-duplicate version of the original image with different appearance but still sharing some duplicated patches.

In large scale image retrieval systems, the state-of-the-art approaches [2, 3, 4, 5, 6, 7, 8, 9] leverage scalable textual retrieval techniques for image search. Similar to text words in information retrieval, local SIFT descriptors [1] are quantized to visual words. Inverted file indexing is then applied to index images via the contained visual words [2]. However, the discriminative power of visual words is far less than that of text words due to quantization. And with the increasing size of image database (*e.g.* greater than one million images) to be indexed, the discriminative power of visual words decreases sharply. Visual words usually suffer from the dilemma of discrimination and ambiguity. On one hand, if the size of visual word codebook is large enough, the ambiguity of features is mitigated and different

features can be easily distinguished from each other. However, similar descriptors polluted by noise may be quantized to different visual words. On the other hand, the variation of similar descriptors is diluted when using a small visual codebook. Therefore, different descriptors may be quantized to the same visual word and cannot be discriminated from each other.

Unlike text words in information retrieval [18], the geometric relationship among visual words plays a very important role in identifying images. Geometric verification [1, 4, 6, 8, 10] has become very popular recently as an important post-processing step to improve the retrieval precision. However, due to the expensive computational cost of full geometric verification, it is usually only applied to some top-ranked candidate images. In web image retrieval, however, the number of potential candidates may be very large. Therefore, it may be insufficient to apply full geometric verification to the top-ranked images for sound recall.

Above all, based on the Bag-of-Visual-Words model, image retrieval mainly relies on improving the discrimination of visual words by reducing feature quantization loss and embedding geometric consistency. The expectation of real-time performance on large scale image databases forces researchers to trade off feature quantization and geometric constraints. Quantization of local features in previous work mainly relies on SIFT descriptor, resulting in limited efficiency while geometric verification is too complex to ensure real-time response.

In this paper, we propose to address partial-duplicate image search by using more efficient feature vector quantization and spatial coding strategies. We define two images as partial-duplicate when they share some identical image patches with the same or very similar spatial layout. Our approach is based on the Bag-of-Visual-Words model. To improve the discrimination of visual words, we quantize local features, SIFT, in both 128-D descriptor space and 1-D orientation space. To verify the matched local figures of two images, we propose a novel spatial coding scheme to encode the relative spatial positions of local features in images. Then through spatial verification based on spatial coding, the false matches of local features can be removed effectively and efficiently, resulting in good precision.

2. RELATED WORK

In the past few years, large scale image retrieval [2, 3, 4, 5, 6, 7, 8, 9] has been significantly boosted by two significant works. The first one is the introduction of local invariant SIFT features [1] for image representation. The second one is the scalable image indexing and query based on the Bag-of-Visual-Words model [2]. With visual words for local features, image representation will be more compact. Moreover, by inverted-file index, the number of candidate images is greatly reduced, since only those images sharing common visual words with the query image need to be checked, achieving efficient response.

Scalability of image retrieval system can be achieved by quantizing local features to visual words. However, quantization also reduces the discriminative power of local descriptors since different descriptors quantized to the same visual word are considered to match to each other. Such quantization error will decrease precision and recall in image retrieval.

To reduce the quantization error, soft-quantization [7, 10] quantizes a SIFT descriptor to multiple visual words. Query expansion [5] reissues the highly ranked images from the original query as new queries to boost recall. However, it may fail on queries with poor initial recall. To improve precision, Hamming Embedding [4] enriches the visual word with compact information from its original local descriptor with Hamming codes [4], and feature scale and orientation values are used to filter false matches.

The above methods focus on improving the discriminative power of visual words. Geometric relationship among local features is ignored. In fact, geometric information of local features plays a key role in image identification. Although exploiting geometric relationships with full geometric verification (RANSAC) [1, 4, 6, 14] can greatly improve retrieval precision, full geometric verification is computationally expensive. In [2, 16], local spatial consistency from some spatial nearest neighbors is used to filter false visual-word matches. However, the spatial nearest neighbors of local features may be sensitive to the image noise incurred by editing. In [8], Bundled-feature groups features in local MSER [12] regions into a local group to increase the discriminative power of local features. The matching score of bundled feature sets are used to weight the visual word vote for image similarity. Since false feature matches between bundles still exist, the bundle weight will be degraded by such false matches.

In [9] [13], min-Hash is proposed for fast indexing via locality sensitive hashing in the context of near-duplicate image detection. Min-Hash represents an image as a visual-word set and defines the image similarity as a set overlap (ratio intersection over union) of their set representation. It works well for duplicate images with high similarity, or, in other words, sharing a large percentage of visual words. But in the partial-duplicate web images, the overlapped visual words may be only a very small portion of image's whole visual word set, resulting in low image-similarity and making it difficult for min-Hash to detect.

3. OUR APPROACH

In our approach, we adopt SIFT features [1] for image representation. Generally, the SIFT feature is characterized with several property values: a 128-D descriptor, a 1-D orientation value (ranging for $-\pi$ to π), a 1-D scale value and the (x, y) coordinates of the key point. In Section 3.1, we will apply the SIFT descriptor and orientation value for SIFT quantization. The locations of SIFT key points will be exploited for generation of spatial maps, as discussed in Section 3.2.

3.1 Vector Quantization of SIFT Feature

To build a large scale image indexing and retrieval system, we need to quantize local descriptors into visual words. Our quantization contains two parts [17]: *descriptor quantization* and *orientation quantization*. Assuming that the duplicated patch enjoys similar spatial layout in both the target and query images, a pair of true matched features should share similar descriptor vector and similar orientation value. Therefore, the features should be quantized in both descriptor space and orientation space.

Since the descriptor and orientation value of SIFT feature are independent to each other, the quantization can be performed in sequential order. Intuitively, we can quantize a SIFT feature first in the descriptor space and then in the orientation space, or in reverse order. Since the orientation value is one-dimensional and it is easy to perform soft quantization, we first quantize SIFT

feature in the descriptor space in a hard manner and then in the orientation space in a soft mode.

3.1.1 *Descriptor quantization*

For descriptor quantization, the bag-of-words approach [2] is adopted. A descriptor quantizer is defined to map a descriptor to an integer index. The quantizer is often obtained by performing *k*-means clustering on a sampling SIFT descriptor set and the resulting descriptor cluster centroids are defined as descriptor visual words. In descriptor quantization, the quantizer assigns the index of the closest centroid to the descriptor. To perform the quantization more efficiently, a hierarchical vocabulary tree [3] is adopted and the resulting leaf nodes are considered as descriptor visual-words.

3.1.2 *Orientation quantization*

For each descriptor visual word, quantization is further performed in the orientation space. To mitigate the quantization error, a soft quantization strategy is applied. Assuming that the quantization number of orientation space is t, when a query SIFT feature is given, we first find the corresponding descriptor visual word using the descriptor quantizer, as discussed in Section 3.1.1. Then, any SIFT feature assigned to the same leaf node will be considered as a valid match when its orientation difference with the query feature is less than π/t. With orientation space quantization, many false positive matches will be removed.

Orientation quantization of SIFT features is based on the assumption that the duplicated patches in both query and target images share the same or similar spatial layout. In fact, such orientation constraint can be relaxed by rotating the query image by some pre-defined angles to generate new queries for query expansion, as discussed in detail in section 5.1. The retrieval results of all rotated queries can be aggregated to obtain the final results.

In [4], SIFT orientation value is used to filter potential false matches via checking the histogram of orientation difference of the matched feature pairs. But in the case that false matches are dominant, the orientation difference histogram may fail to discover genuine matches.

3.2 Spatial Coding

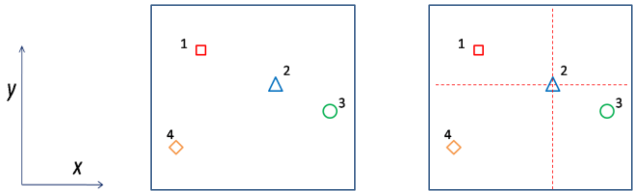
The spatial relationships among visual words in an image are critical in identifying special duplicate image patches. After SIFT quantization, matching pairs of local features between two images can be obtained. However, the matching results are usually polluted by some false matches. Generally, geometric verification [1, 6] can be adopted to refine the matching results by discovering the transformation and filtering false positives. Since full geometric verification with RANSAC [14] is computationally expensive, it is usually only adopted as a post-processing stage. Some more efficient schemes to encode the spatial relationships of visual words are desired. Motivated by this problem, we propose the spatial coding scheme.

Spatial coding encodes the relative positions between each pair of features in an image. Two binary spatial maps, called *X*-map and *Y*-map, are generated. The *X*-map and *Y*-map describes the relative spatial positions between each feature pair along the horizontal (*X*-axis) and vertical (*Y*-axis) directions, respectively. For instance, given an image I with K features $\{v_i\}$, $(i = 1,2,\ldots,K)$, its *X*-map and *Y*-map are both $K \times K$ binary matrix defined as follows,

$$Xmap(i,j) = \begin{cases} 0 & \text{if } x_i < x_j \\ 1 & \text{if } x_i \geq x_j \end{cases} \quad (1)$$

$$Ymap(i,j) = \begin{cases} 0 & \text{if } y_i < y_j \\ 1 & \text{if } y_i \geq y_j \end{cases} \quad (2)$$

where (x_i, y_i) and (x_j, y_j) are the coordinates of feature v_i and v_j, respectively.



(a) (b)

Figure 2. An illustration of spatial coding for image features. (a) shows an image with four features; (b) shows the image plane division with feature 2 as the origin point.

Fig. 2 shows an illustration of spatial coding on an image with four features. The resulting *X*-map and *Y*-map are:

$$Xmap = \begin{pmatrix} 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix}; \quad Ymap = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

In *X*-map and *Y*-map, row *i* records the feature v_i 's spatial relationships with other features in the image. For example, $Xmap(1,2) = 0$ and $Ymap(1,2) = 1$ means feature v_1 is on the left side of feature v_2 and above it. We also can understand the map as follows. In row *i*, feature v_i is selected as the origin, and the image plane is divided into four quadrants along horizontal and vertical directions. *X*-map and *Y*-map then show which quadrant other features are located in (Fig.2 (b)). Therefore, one bit either 0 or 1 can encode the relative spatial position of one feature to another in one coordinate. In total, we use $\log(4) = 2$ bits, one bit for the *X*-map and one bit for the *Y*-map.

In fact, *X*-map and *Y*-map impose loose geometric constraints among local features. Further, we advance our spatial coding to more general formulations, so as to impose stricter geometric constraints. The image plane can be evenly divided into $4 \cdot r$ parts, with each quadrant uniformly divided into r parts. Correspondingly, two relative spatial maps *GX* and *GY* are desired to encode the relative spatial positions of feature pairs. Intuitively, it will take at least $\lceil \log(4 \cdot r) \rceil$ bits to encode relative spatial position of feature v_i to feature v_j ($\lceil \cdot \rceil$ denotes the least integer), by exactly determining which fan region v_i is located in. Instead, we propose to use a more efficient approach to generate the spatial maps.

For an image plane divided uniformly into $4 \cdot r$ fan regions with one feature as the reference origin point as discussed above, we decompose the division into r independent sub-divisions, each uniformly dividing the image plane into four parts. Each sub-division is then encoded independently and their combination leads to the final spatial coding maps. Fig. 3 illustrates the decomposition of image plane division with $r = 2$ and feature v_2 as the reference origin. As shown in Fig. 3(a), the image plane is divided into 8 fan regions. We decompose it into two sub-divisions: Fig. 3(b) and Fig. 3(c). The spatial maps of Fig. 3(b) can be generated by Eq. (1) and Eq. (2). The sub-division in Fig. 3(c) can be encoded in a similar way. It just needs to rotate all the feature coordinates and the division lines counterclockwise, until the two division lines become horizontal and vertical, respectively, as shown in Fig. 3(d). After that, the spatial maps can be easily generated by Eq. (1) and Eq. (2).

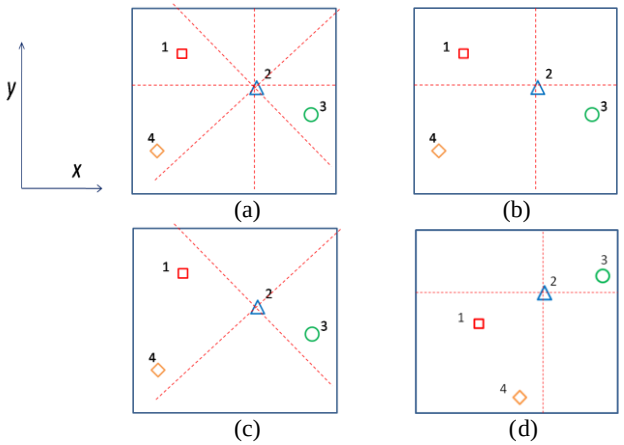


Figure 3. An illustration of spatial coding with *r* = 2 for image features. (a) shows the image plane division with feature 2 as the origin point; (a) can be decomposed into (b) and (c); (c) rotates $\pi / 4$ counterclockwise yields (d).

Consequently, the general spatial maps *GX* and *GY* are both 3-D matrix and can be generated as follows. Specially, the location (x_i, y_i) of feature v_i is rotated counterclockwise by $\theta = \frac{k \cdot \pi}{2 \cdot r}$ degree $(k = 0,1,\ldots,r-1)$ according to the image origin point, yielding the new location $(x_i^{\,k}, y_i^{\,k})$ as,

$$\begin{pmatrix} x_i^{\,k} \\ y_i^{\,k} \end{pmatrix} = \begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix} \cdot \begin{pmatrix} x_i \\ y_i \end{pmatrix} \quad (3)$$

Then GX and GY are defined as,

$$GX(i, j, k) = \begin{cases} 0 & \text{if } x_i^{\,k} < x_j^{\,k} \\ 1 & \text{if } x_i^{\,k} \geq x_j^{\,k} \end{cases} \quad (4)$$

$$GY(i, j, k) = \begin{cases} 0 & \text{if } y_i^{\,k} < y_j^{\,k} \\ 1 & \text{if } y_i^{\,k} \geq y_j^{\,k} \end{cases} \quad (5)$$

With the generalized spatial maps *GX* and *GY*, the relative spatial positions between each pair of features can be more strictly defined.

From the discussion above, it can be seen that, spatial coding can be very efficiently performed. But the whole spatial maps of all features in an image will cost considerable memory. Fortunately, there is no need to store these maps. Instead, we only need to save the sorting orders of *x*- and *y*-coordinate of each feature, respectively. When checking the feature matching of two images, we only need the sorting orders of the coordinates of these matched features, which will be used to generate the spatial maps for spatial verification in real time. The details are discussed in the next subsection.

3.3 Spatial Verification

Spatial coding plays an important role in spatial verification. Since the problem that we focus on is partial-duplicate image retrieval, there is an underlying requirement that the target image must share some duplicated patches, or in other words, share the same or very similar spatial configuration of matched feature points. Due to the unavoidable quantization error, false feature matches are usually incurred. To more accurately define the similarity between images, it is desired to remove such false matches. Our spatial verification with spatial coding can perform this task.

Denote that a query image I_q and a matched image I_m are found to share N pairs of matched features through SIFT quantization. Then the corresponding sub-spatial-maps of these matched features for both I_q and I_m can be generated and denoted as (GX_q, GY_q) and (GX_m, GY_m). For efficient comparison, we perform logical Exclusive-OR (XOR) operation on GX_q and GX_m, GY_q and GY_m, respectively, as follows,

$$V_x(i, j, k) = GX_q(i, j, k) \oplus GX_m(i, j, k) \quad (6)$$

$$V_y(i, j, k) = GY_q(i, j, k) \oplus GY_m(i, j, k) \quad (7)$$

Ideally, if all N matched pairs are true, V_x and V_y will be zero for all their entries. If some false matches exist, the entries of these false matches on GX_q and GX_m may be inconsistent, and so is that on GY_q and GY_m. Those inconsistencies will cause the corresponding exclusive-OR result of V_x and V_y to be 1. Denote

$$S_x(i) = \sum_{j=1}^{N} \bigcup_{k=1}^{r} V_x(i, j, k) \ , \ S_y(i) = \sum_{j=1}^{N} \bigcup_{k=1}^{r} V_y(i, j, k) \quad (8)$$

Consequently, by checking value of S_x and S_y, the false matches can be identified and removed. The corresponding entry values of the remained true matches in V_x and V_y will be all zeros.

From the discussion above, it can be found that the spatial coding factor r controls the strictness of spatial constraints and will impact verification performance. We will study it in Section 4.1.3.

Fig. 4 shows two instances of the spatial verification with spatial coding on a relevant image pair and an irrelevant image pair. Both image pairs have many matches of local features after quantization. For the left "Mona Lisa" instance, after spatial verification via spatial coding, those false matches are discovered

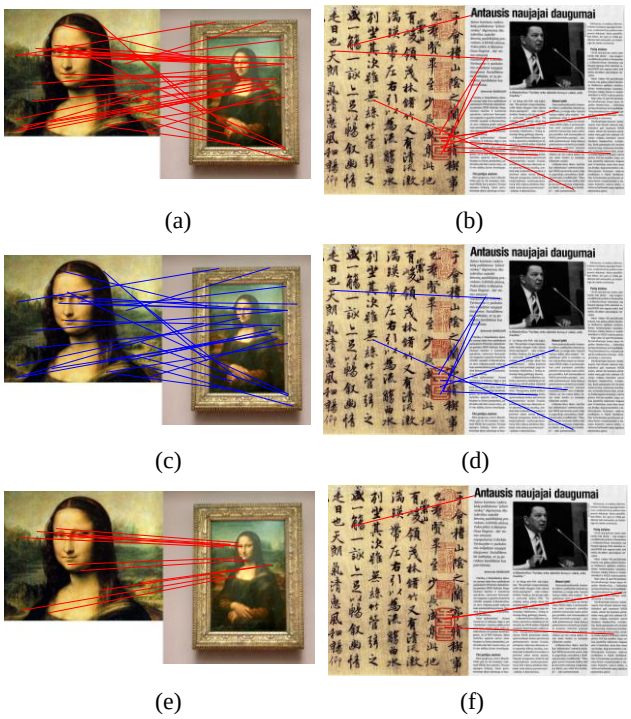


Figure 4. An illustration of spatial verification with spatial coding on a relevant pair (left column) and an irrelevant pair (right column). (a) (b)Initial matched feature pair after quantization; (c) (d) False matches detected by spatial verification; (e) (f) True matches that pass the spatial verification.

and removed, while true matches are satisfactorily kept. For the right instance, although they are irrelevant in content, 12 matched feature pairs are still found after quantization. However, by doing spatial verification, most of the mismatching pairs are removed and only 3 pairs of matches are kept. Moreover, it can be observed that those 3 pairs of features do share high geometric similarity.

The detailed algorithm for spatial verification with spatial coding is shown in Fig. 5. In spatial verification, the main computation operations are logical Exclusive-OR and addition. Therefore, unlike full geometric verification with RANSAC [1, 6, 14], the computational cost is very low.

3.4 Indexing and retrieval

An inverted-file index structure is used for large-scale indexing and retrieval, as illustrated in Fig. 6. Each visual word has an entry in the index that contains the list of images in which the visual word appears. As discussed in Section 3.2 and 3.3, we do not need to know the accurate location of local SIFT features. Instead, we only need to record the relative spatial positions of local features. Therefore, it suffices to store the sorting order of the *x*-coordinate and *y*-coordinate of each feature, which will be used to generate the spatial coding maps during query time. Consequently, for each indexed feature, we store its image ID, SIFT orientation value and the corresponding sorting order for its *x*- and *y*- coordinate.

We formulate the image retrieval as a voting problem. Each visual

Spatial Verification with Spatial Coding

1) Find those matching pairs $P = \{(q_i, m_i)\}$, $(i = 1 \sim N)\}$, where q_i and m_i are feature points in the query image and matched image, respectively, N is the total number of matching pairs. Let $Q = \{ q_i \}$ and $M = \{ m_i \}$;
2) Generate the spatial maps *GX_q* and *GY_q* for *Q*, *GX_m* and *GY_m* for *M*, by Eq.(4) and Eq. (5).
3) Compute the inconsistency matrix V_x and V_y, according to Eq. (6) and Eq. (7).
4) Compute the inconsistency sum S_x and S_y, by Eq. (8).
5) By check value of S_x and S_y, identify and remove the false matches.

Figure 5. The general steps of spatial coding verification

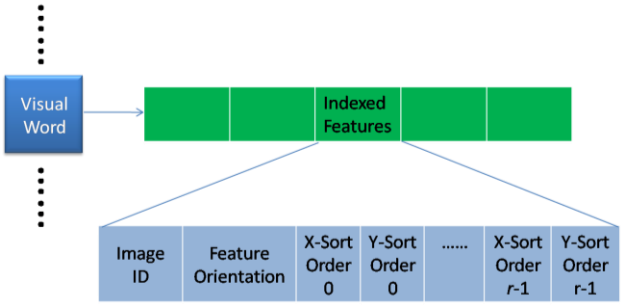


Figure 6. Inverted file structure.

word in the query image votes on its matched images. Intuitively, the *tf-idf* weight [2] can be used to distinguish different matched features. However, from our experiments, we find that simply counting the number of matched features yields similar or better results. Further, to distinguish images with the same number of true positive matches but different false positive matches or different feature number, a penalty term from spatial verification is also defined. Suppose a query image *q* and a matched image *p* share *a* pairs of visual words from both descriptor and orientation quantization, and only *b* pairs pass the spatial verification, we define the similarity between the two images as:

$$S(q, p) = b - \frac{a - b + 1}{a} \cdot \frac{N(p)}{N_{\max}} \quad (9)$$

where $N(p)$ denotes the feature number in the matched image *p* and $N_{\max}$ denotes the maximum number of SIFT features in an image.

4. EXPERIMENTS

We build our basic dataset by crawling one million images that are most frequently clicked on a commercial image-search engine. Since there is no public dataset for evaluation of partial-duplicate image retrieval, following the Tineye search demo results (http://www.tineye.com/cool_searches), we collected and manually labeled 1100 partially duplicated web images of 23 groups from both Tineye [11] and Google Image search. The images in each group are partial duplicates of each other and there are very near-exact duplicates in these images. Some typical examples are shown in both Fig. 1 and Fig. 16.

Since the basic 1M dataset also contains additional partial duplicates of our ground truth data, for evaluation purpose we identify and remove these partially duplicated images from the basic dataset by querying the database with every image from our ground-truth dataset. We then add these ground truth images into the basic dataset to construct an evaluation dataset.

To evaluate the performance with respect to the size of dataset, three smaller datasets (50K, 200K, and 500K) are built by sampling the basic dataset. In our evaluation, 100 representative query images are selected from the ground truth dataset. Mean average precision (mAP) [6] is adopted as our evaluation metric.

4.1 Impact of parameters

The performance of our approach is related with three parameters: orientation quantization size, SIFT descriptor codebook size and spatial coding map factor r. In the following, we will study their impacts respectively and select the optimal values.

4.1.1 Orientation quantization size

To study the impact of the orientation quantization, we experiment with different quantization sizes on the 1M image dataset. The performance of mAP for different orientation quantization sizes with $r = 1$ is shown in Fig. 7. Orientation quantization size with value equal to 1 means that no orientation quantization is performed. For each dataset, when the quantization size increases, the performance first increases and then keeps stable with a little drop, while the time cost first decreases sharply and then stays stable. The maximal mAP value is obtained with orientation size as 11 and the corresponding time cost is 0.48s per query. Hence, in the following experiments, we select the orientation quantization size as 11.

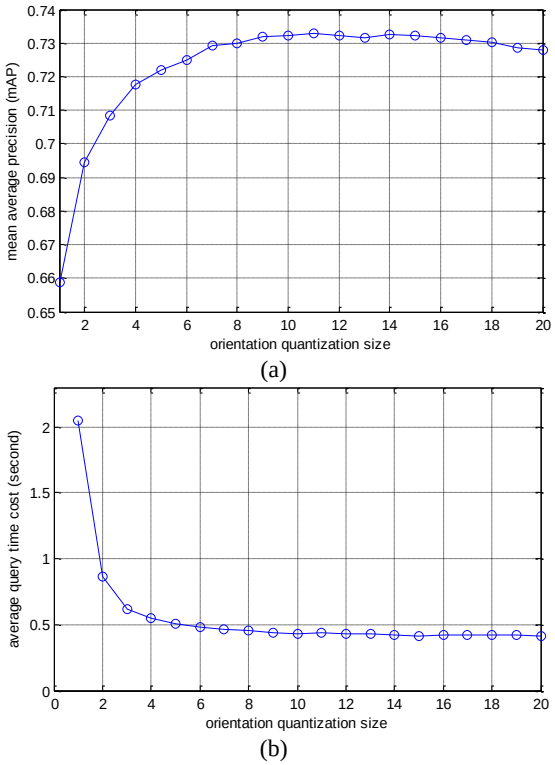


Figure 7. (a)Mean average precision and (b)average query time cost with different orientation quantization sizes and a visual vocabulary tree of 130K descriptor visual words. The size of testing image database is 1 million. The maximal mAP is obtained with orientation quantization size as 11.

4.1.2 SIFT descriptor codebook size

The size of SIFT descriptor codebook size describes the extent of the descriptor space division. Since our spatial coding can effectively discover and remove false matches, a comparatively smaller codebook can be adopted, with the SIFT descriptor space coarsely divided.

We test four different sizes of visual codebooks on the 1M image database. From Fig. 8(a), it can be observed that when the size of descriptor visual codebook increases from 12K to 500K, the mAP decreases gradually. As can be seen in Fig. 8(b), the time cost when using the small codebook is very high, but reduces sharply and then keeps stable when the codebook size increases to 130K and larger. Codebook with 130K descriptor visual words gives the best tradeoff between mAP and time cost. In our later experiments, we use the visual codebook with 130K- SIFT descriptor.

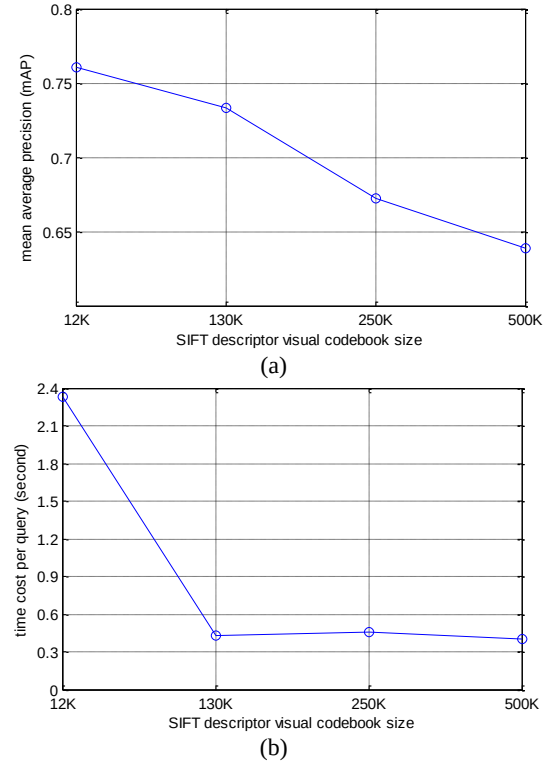
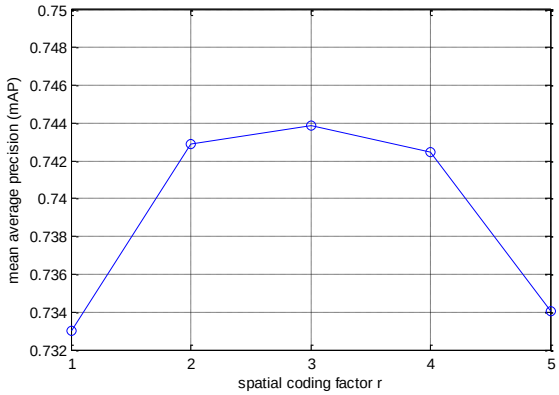


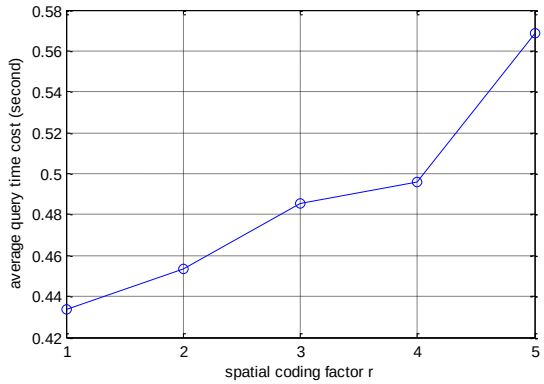
Figure 8. (a) Mean average precision and (b) time cost per query on different sizes of SIFT-descriptor visual codebook.

4.1.3 Impact of r

The r value of the spatial coding factor determines the division extent of image plane to check the relative spatial positions of features. We test the performance of our spatial coding using different r value on the 1M dataset, with orientation quantization size equal to 11. The performance and time cost are shown in Fig. 9. Intuitively, higher r value defines stricter relative spatial relationships and better performance is expected. However, due to the minor SIFT drifting error as discussed in section 5.2, higher r value will not necessarily obtain better performance. As illustrated in Fig. 9, r = 3 gives the best results and is used in our report results.



(a)



(b)

Figure 9. (a) Mean average precision and (b) average query time cost on different values of spatial coding factor *r*.

4.2 Evaluation

We use a Bag-of-Visual-Words approach with vocabulary tree [3] as the "baseline" approach. A visual vocabulary of 1M visual words is adopted. In fact, we have experimented with different visual codebook sizes, and have found the 1M vocabulary yields the best overall performance for the baseline.

Two approaches are adopted to enhance the baseline method for comparison. The first one is Hamming Embedding [4] by adding a hamming code to filter out matched features that have the same number of quantized visual words but have a large hamming distance from the query feature. We denote this method as "HE." The second one is re-ranking via geometric verification, which is based on the estimation of an affine transformation with our implementation of [6]. As post-processing, the re-ranking is performed on only the top 300 initial results of the baseline. We call this method "reranking".

From Fig. 10, it can be observed that our approach outperforms all the other three methods. On the 1M dataset, The mAP of the baseline is 0.486. Our approach increases it to 0.744, a 53% improvement. Since re-ranking based on full geometric verification is only applied on the top-300 initial returned results, its performance is highly determined by the recall of initial top-300 results. As can be seen, the performance of reranking the results of baseline is 0.61, lower than our approach.

Fig. 11 illustrates the mAP performance of the baseline and our approach on all the 100 testing query images. It can be observed

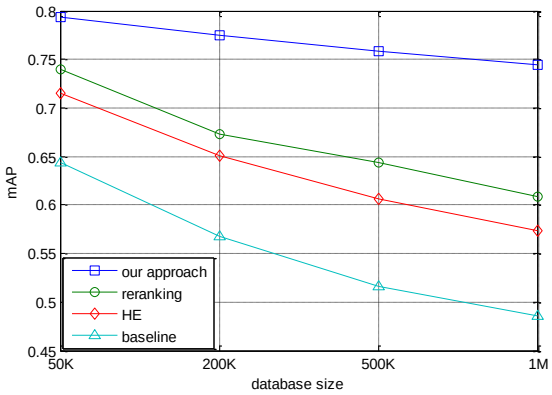


Figure 10. Performance comparison of different methods with different database sizes.

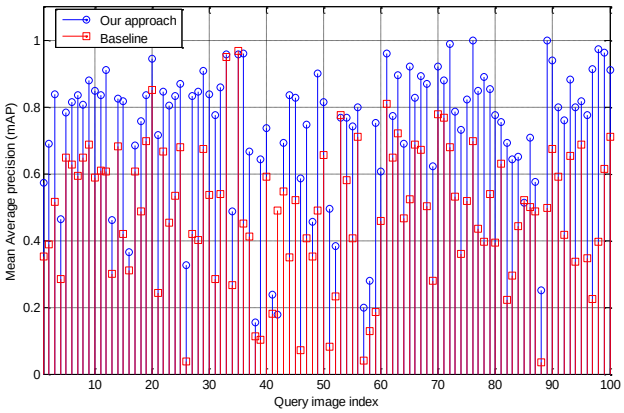


Figure 11. The mAP performance of the baseline and our approach on all query images

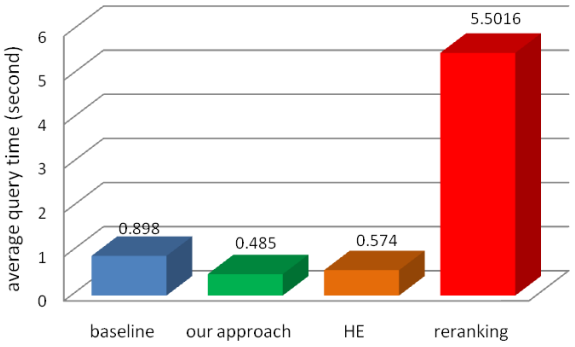
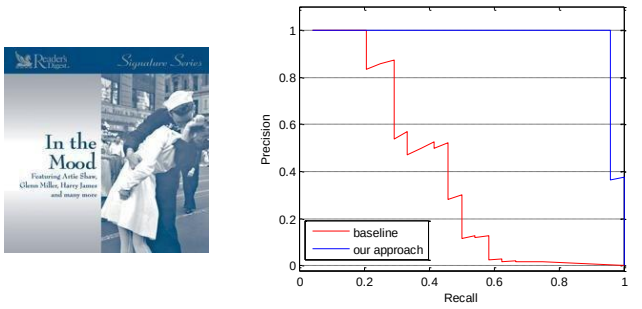


Figure 12. Comparison of average query time cost for different methods on the 1M database.

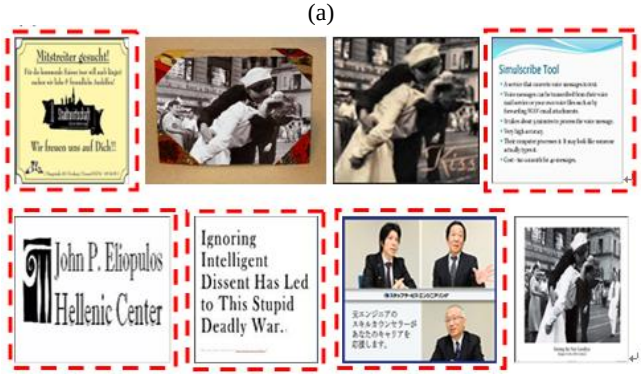
that, except for comparative performance on some queries, our approach outperforms over the baseline on most of the queries.

We perform the experiments on a server with 2.0 GHz CPU and 16 GB memory. Fig. 12 shows the average time cost of all four approaches. The time for SIFT feature extraction is not included.

Compared with the baseline, our approach is much less time-consuming. It takes the baseline 0.90 second to perform one image query on average, while for our approach the average query time cost is 0.49 second, 54% of the baseline cost. Hamming Embedding is also very efficient, but still with 18%



(a)



(b)

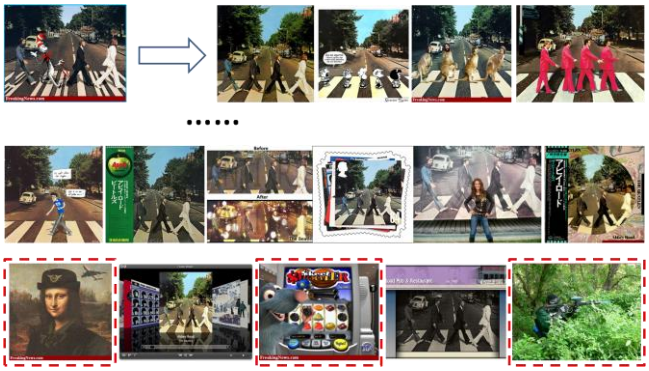
(c)

Figure 13. Sample results comparing our approach with the baseline approach. (a) Query image and a comparison of the precision-recall curves. (b) The top images returned by the baseline approach. (c) The top images returned by our approach. The false positives are shown with red dashed bounding boxes.

more time cost than our approach. Full geometric re-ranking is the most time-consuming, introducing an additional 4.6 seconds over the baseline on the top 300 candidate images.

4.3 Sample results

In Fig. 13, we give examples of our results on the 1M dataset on the "sailor kiss" query image. For this query, compared with the baseline approach, our approach improves the mAP from 0.40 to0.97, with 142% improvement. Fig. 13 (b) and (c) show the top images returned by the baseline approach and our approach, respectively. Since the top 5 images of both approaches are all correct, we show results starting from the 6th returned image. The false positives are marked by red dashed bounding boxes. Due to the additions of text in the query image, the top-returned results are greatly polluted by irrelevant images, which contain similar visual words related with text patch. As for our approach, such false positives are effectively removed.



(a)

(b) (c)

Figure 14. (a): Top-ranked images returned from an "Abbey Road" image query. The query image is shown on top left and false positives with red dashed bounding boxes. (b) (c): The feature matching between the query image and two false positives. The red lines across two images denote the true matches, while the blue lines denote the matches that fail to pass the spatial coding verification.

Fig. 14 shows the results for a query image "Abbey Road". We select those true positives before the first three false positives. The first three positives are highlighted by red dashed bounding boxes. Fig. 14 (b) and (c) illustrate the SIFT matching results between the query image and the first and third false positive. The first false positive shares duplicate regions with the query image on the bottom containing text addition "Freakingnews.com". It should be noted that there is a true matched feature pair (blue line) that fails to pass the spatial coding verification. This is due to the SIFT drifting error, as discussed in section 5.2. The second false positive in the middle of the third row in Fig. 14(a) is also due to the same reason as the first false positive. As for the third false positive, although no duplicate patches are shared, the remained four pairs of matched features do share almost the same geometric configuration. To remove such mismatching pairs, some other information, need to be incorporated.

Fig. 16 shows more example results using our spatial coding approach. It can be observed that, the retrieved images are not only diverse but also contain large changes in contrast, scale, or significant editing.

5. DISCUSSION

5.1 Orientation Rotation

Our orientation quantization is based on the assumption that the duplicated patches in query image and matched image share the same or very similar spatial layout. This is reasonable in most application cases. In fact, this orientation constraint can be easily relaxed to adapt to our framework. In other words, we can rotate the query image by a few pre-defined angles so that all possible orientation changes from the original image will be covered. Each rotated version is used as query and the aggregation results are merged as the final retrieval results. In fact, the query image does not need to be rotated, since the SIFT features of each rotated query share the same descriptors as the original query but only differ in orientation value. Therefore, we just need to change the features' orientation value and compute the new spatial location for each query feature. The remaining processing is the same as the case of no orientation rotation. It should be noted that the quantization in descriptor space needs to be performed only once.

5.2 Error Analysis for Spatial Maps

Our spatial coding is based on an assumption that the key point location of the SIFT features in the object of interest is invariant to general 2D editing. However, due to the unavoidable digital error in the detection of key points, some SIFT features exhibit some small drifting, less than one pixel. On such drifting, the comparable location of features with the same *x*- or *y*-coordinate will be inverse, causing some true matches to fail our spatial coding verification. Such phenomenon is prevalent in the case when two duplicate images share many matched feature pairs in a cluster. Small affine transformation of images will exacerbate such error. Moreover, the error will also be worsened by too large spatial coding factor r , as demonstrated in Section 4.1.3. As illustrated in Fig. 15, although the duplicate image pair shares 12 true matches, 5 of them are discovered as false matches that fail to pass the spatial coding verification. Since still many true matches are remained, the matched images will be assigned a comparatively high similarity score and consequently the effect of those false negatives on the retrieval performance is small.

5.3 Query Expansion

The spatial coding verification result is very suitable for query expansion. Unlike previous works that define the similarity of a query image to the matched image by the L1, L2 or cosine distance, our approach formulates the image similarity just by the number of matched feature pairs. Image pairs with highly duplicated patches usually share many (>10) matched feature pairs, while unrelated images only have few (<5) matched features pairs, as shown in Fig. 4. Therefore, the number of matched feature pairs naturally lends itself as a criterion to select top-ranked images as seeds for query expansion.

6. CONCLUSION

In this paper, we propose a novel scheme of spatial coding for large scale partial-duplicate image search. The spatial coding efficiently encodes the relative spatial locations among features in an image and effectively discovers false feature matches between images. As for partial-duplicate image retrieval, spatial coding achieves even better performance than geometric verification on the baseline and consumes much less computational time.

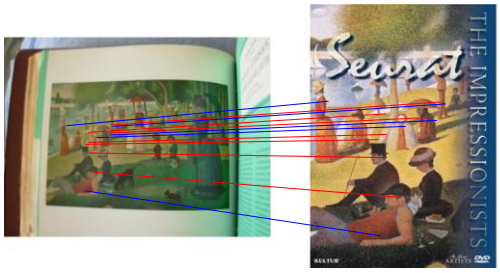


Figure 15. An instance of matching error due to SIFT drifting. Each line across two images denotes a match of two local features. The red lines denote the true matches that pass the spatial coding verification, while the blue lines denote those that fail to.

In our approach, we adopt SIFT feature for image representation. It should be noted that our method is not constrained to SIFT. Some other local features, such as SURF [15], can also be substituted for SIFT.

Our spatial coding aims to identify images sharing some duplicated patches. As demonstrated in the experiments, our approach is very effective and efficient for large scale partial-duplicate image retrieval. However, it may not work as well on general object retrieval, such as searching for different style cars.

In the future, we will experiment on orientation rotation and query expansion, as discussed in Section 5.2 and 5.3, respectively. Besides, we will also focus on better quantization strategy to generate more discriminative visual words and test other local affine-invariant features.

7. ACKNOWLEDGMENTS

This work was supported in part by NSFC under contract No. 60632040 and 60672161, Program for New Century Excellent Talents in University (NCET), Research Enhancement Program (REP) and start-up funding from the Texas State University, and Akiira Media Systems, Inc. for Dr. Qi Tian.

8. REFERENCES

[1] D. Lowe. Distinctive image features from scale-invariant key points. *IJCV*, 60(2):91-110, 2004.

[2] J. Sivic and A. Zisserman. Video Google: A text retrieval approach to object matching in videos. In *Proc. ICCV*, 2003.

[3] D. Nister and H. Stewenius. Scalable recognition with a vocabulary tree. In Proc. CVPR, pages 2161-2168, 2006.

[4] H. Jegou, M. Douze, and C. Schmid. Hamming embedding and weak geometric consistency for large scale image search. In *Proc. ECCV*, 2008.

[5] O. Chum, J. Philbin, J. Sivic, M. Isard, and A. Zisserman. Total recall: Automatic query expansion with a generative feature model for object retrieval. In *Proc. ICCV*, 2007.

[6] J. Philbin, O. Chum, M. Isard, J. Sivic and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *Proc. CVPR*, 2007.

[7] J. Philbin, O. Chum, M. Isard, J. Sivic and A. Zisserman. Lost in quantization: Improving particular object retrieval in large scale image databases. In *Proc. CVPR*, 2008.

[8] Z. Wu, Q. Ke, M. Isard, and J. Sun. Bundling Features for Large Scale Partial-*Duplicate* Web Image Search. In *Proc. CVPR*, 2009.

[9] O. Chum, J. Philbin, M. Isard, and A. Zisserman. Scalable near identical image and shot detection. In *Proc. CIVR*, 2007.

[10] H. Jegou, H. Harzallah, and C. Schmid. A contextual dissimilarity measure for accurate and efficient image search. In *Proc. CVPR*, 2007.

[11] http://www. Tineye.com

[12] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide baseline stereo from maximally stable extremal regions. In Proc. BMVC, 2002.

[13] Ondřej Chum, James Philbin, and Andrew Zisserman. Near duplicate image detection: min-Hash and tf-idf weighting. In *Proc. BMVC*, 2008.

[14] M. A. Fischler and R. C. Bolles. Random Sample Consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Comm. of the ACM*, 24: 381–395, 1981.

[15] H. Bay, T. Tuytelaars, L. V. Gool. SURF: Speeded up robust features. In *Proc. ECCV*, 2006.

[16] T. Li, T. Mei, I. –S. Kweon, X. –S, Hua, Contextual bag-of-words for visual categorization. *IEEE Transactions on Circuits and Systems for Video Technology*. 2010.

[17] W. Zhou, H. Li, Y. Lu, and Q. Tian, "Large Scale Partial-Duplicate Image Retrieval with Bi-Space Quantization and Geometric Consistency," In *Proc. ICASSP*, 2010.

[18] D. Liu, X. Hua, L. Yang, M. Wang, and H. Zhang. Tag Ranking, In *Proc. WWW*, 2009.

Figure 16. Example results. Queries are shown on first left column of each row, and highly-ranked images (selected from those before the first false positive) from the query results are shown on the right.