Image Retrieval with Geometry-Preserving Visual Phrases

Yimeng Zhang Zhaoyin Jia Tsuhan Chen
School of Electrical and Computer Engineering, Cornell University
{yz457,zj32,tsuhan}@cornell.edu

Abstract

The most popular approach to large scale image retrieval is based on the bag-of-visual-word (BoV) representation of images. The spatial information is usually re-introduced as a post-processing step to re-rank the retrieved images, through a spatial verification like RANSAC. Since the spatial verification techniques are computationally expensive, they can be applied only to the top images in the initial ranking. In this paper, we propose an approach that can encode more spatial information into BoV representation and that is efficient enough to be applied to large-scale databases. Other works pursuing the same purpose have proposed exploring the word co-occurrences in the neighborhood areas. Our approach encodes more spatial information through the geometry-preserving visual phrases (GVP). In addition to co-occurrences, the GVP method also captures the local and long-range spatial layouts of the words. Our GVP based searching algorithm increases little memory usage or computational time compared to the BoV method. Moreover, we show that our approach can also be integrated to the min-hash method to improve its retrieval accuracy. The experiment results on Oxford 5K and Flicker 1M dataset show that our approach outperforms the BoV method even following a RANSAC verification.

1. Introduction

Similar image retrieval has attracted increasing interests in recent years. Given a query image or region, the goal is to retrieve the images of the same object or scene from a large database and return a ranked list. Three important factors must be considered in a large-scale retrieval system: retrieval accuracy, memory usage, and efficiency.

Most state of the art retrieval technologies are based on the bag-of-visual-word (BoV) model initially introduced by [17], in which images are represented as histograms of visual words. Image querying is typically accomplished in two steps: *searching* and *post-processing*. During the *searching* step, similar images are retrieved from the large database and an initial ranking is generated. This step must

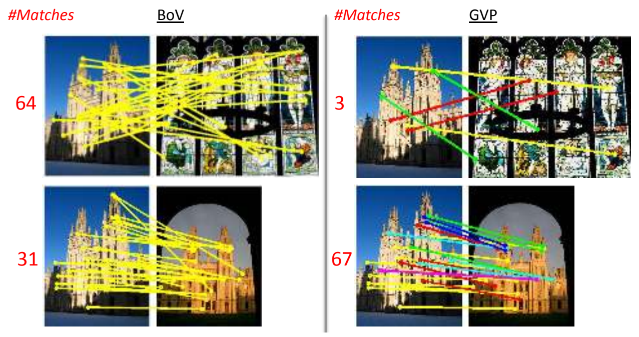


Figure 1. The images on the left show the matched *visual words* of a query image with two different images, and images on the right show the matched *geometry-preserving visual phrases (GVP)*. Each GVP is composed of two words in the same color. With the visual words, we have more matches for images of different objects (top image pair) than images of the same object (bottom image pair). On the contrary, with the GVP, we have much fewer matches for different objects than the same object. If we use the number of matches for ranking, the bag of words model will give incorrect ranking for these two images, while the GVP can generate correct ranking.

be facilitated with an efficient algorithm in order to deal with large scale databases. The most popular approach is to index images with inverted files [17] to facilitate fast access to the images with common words. The *post-processing* step provides a more precise ranking of the retrieved images, usually through spatial verification [15]. Numerous works have been proposed and have successfully improved the retrieval performance and efficiency. The approximate nearest neighbor [15] and tree vocabulary [13] increase the efficiency of building a large vocabulary, while soft matching [16] and hamming embedding [7] address the hard quantization problem of visual words. Spatial verification methods [15] and query expansion [5] have been proposed for re-ranking at the post-processing step, and many methods [9, 22, 8, 14] have been introduced to decrease the memory usage for the inverted files.

In this paper, we are interested in improving the BoV model with spatial information. Despite its simplicity and efficiency, the BoV model discards spatial information, which is crucial for visual representation because of the ambiguity of visual words. Spatial information is usually re-introduced in the post-processing step through a geometry verification, such as RANSAC [15] or neighboring feature consistency [17]. Since geometry verification methods are usually computationally expensive, they are applied only to the top images in the initial ranking. Therefore, efficient algorithms that encode more spatial information into the searching step are beneficial. Lin and Brandt [12], and Lampert [10] rank the images based on the matching scores of the query image with the localized sub-windows in images. These methods encode more spatial information than the the BoV model on the entire image and provide localizations of the query. However, when the query region is large, they are used primarily as a post-processing step because of the memory usage and speed[12]. Spatial Pyramid Matching (SPM)[11] and methods with GIST features [19] encode rigid spatial information by quantizing the image space and lack the invariance to transformations. Spatial-bag-of-features[1] handle variances of SPM by changing the order of the histograms; the spatial histogram of each visual word is rearranged by starting from the position with the maximum frequency, resulting in improvement over both BoV and SPM. However, this rearrangement may not correspond to the true transformation.

Another approach is to search using phrases or collocations generated from visual words. Previous works usually use the phrases to model the co-occurrences of the words, either in the entire images or in local neighborhoods. Co-occurrences in the entire images [18] do not capture spatial information between the words, while co-occurrences in local neighborhoods [21, 25, 20] capture neighboring information, but ignore long-range interactions. Moreover, they ignore the spatial layouts of the words in the neighborhoods [21, 25] or only perform *weak* spatial verification[20]. Another problem with existing methods [18, 21, 25] is that, because the total number of phrases can increase exponentially to the number of words in a phrase, they must select a subset from the entire phrase set. Sophisticated mining or learning algorithms have been proposed for this selection, but it may still be risky to discard a large portion of phrases, some of which may be representative ones for the images.

We use geometry-preserving visual phrases (GVP) to encode more spatial information in the searching step. A GVP is a set of visual words in a particular spatial layout, so different layouts define different GVP. Our searching algorithm with GVP is inspired by a recent algorithm proposed by Zhang and Chen [23] for object categorization. They propose an efficient algorithm which can identify the co-occurring GVP [1] in time linear to the number of local words in an image. The algorithm is used to build a kernel of the SVM for object categorization task, and do not consider large-scale databases. We extend the algorithm so we can perform efficient large-scale image search. Our approach increases little memory usage or runtime of the traditional searching method with the BoV model, while providing a better initial ranking with more spatial information. Figure 1 illustrates the differences of the ranking results using BoV and GVP.

Moreover, our approach can integrate the geometry-preserving visual phrases(GVP) into the popular min-hash method to further improve the efficiency of searching with GVP, because the min-hash method reduces memory usage and increases the search efficiency. The traditional min-hash method [4, 6] is based on the BoV model. Our approach increases its retrieval accuracy by adding spatial information without increasing the computational cost. In this line of work, we are related to [3] and [2]. While they consider local co-occurrences [3] or global co-occurrences [2], we encode more spatial information.

[1]In their paper, they use high order features. We use phrases to make consistency with papers in the image retrieval literature.

2. Inverted Files with GVP

2.1. Inverted File Index

We first review the traditional searching scheme with the inverted file structure. An image I_i is represented as a vector $V(I_i)$, with one component for each visual word in the dictionary. The j^{th} component in the vector $v_j(I_i)$ is the weight of the word j: the tf-idf weighting scheme is usually used. The similarity of two images I_i and $I_{i'}$ is defined as the *cosine similarity* of the two vectors $V(I_i) \cdot V(I_{i'})/(\|V(I_i)\|\|V(I_{i'}\|)$. Ranking with this similarity also gives the same result as ranking with the Euclidean distance of the L2-normalized vectors. With a large vocabulary, this vector representation is very sparse. The inverted file structure [17] utilizes this sparseness to index images and enables fast searching. For each visual word in the vocabulary, this structure stores the list of images in which the word occurs and its term frequency (tf).

Searching Scheme: Given a query image q, the search can be interpreted as a voting scheme [7]: 1) The scores of all images in the database are initialized to 0. 2) For each word j in the query, we retrieve the list of images that contain this word through the inverted files. For each image i in the list, we increment its score by the weight of this word $score(i) += tf_{ij} \times idf_j$. After processing all words in the query, the final score of image i gives the dot product of the vectors of image i and the query. 3) We normalize the scores to obtain the cosine similarities for ranking.

2.2. Geometry-Preserving Visual Phrases

A geometry-preserving visual phrase (GVP) of length k is defined as k visual words in a certain spatial layout. Different words and different spatial layouts both define different phrases. To tolerate shape deformation, the image space is quantized into bins. Figure 2 shows example occurrences

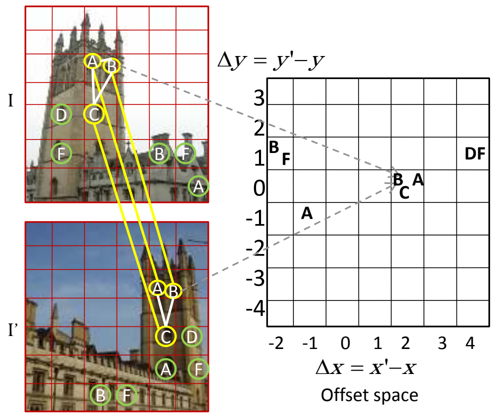


Figure 2. Illustration of the co-occurrences of the same GVP. Different alphabets $(A, B, ...)$ represent different visual words.

of the same GVP of length 3 in the two images. An image will be represented as a vector defined with the GVP. Similar to the BoV model, the vector representation $V^k(I)$ of an image I is defined as the histogram of GVP of length k, with the i^{th} component representing the frequency (tf) of phrase p_i. For simplicity of explanation, we first ignore the idf part of the weighting. This vector can be extremely long even when $k = 2$ while a large vocabulary is used. Therefore, it is impractical to create the vectors directly and perform the search. However, it can be easily proven that the dot product of such vectors of two images equals the total number of co-occurring GVP in these images.

We utilize the algorithm proposed in [23] to identify the co-occurring GVP in two images. The algorithm is illustrated in figure 2. For each pair of the same word j in images I and I', we calculate their offset $(\triangle x_j, \triangle y_j)$, which is the location of the word in image I' substracts that in image I. Then a vote is generated on the offset space at $(\triangle x_j, \triangle y_j)$. On the offset space, k votes locating at the same place correspond to a co-occurring GVP of length k. In the example of figure 2, word A, B, C corresponds to a co-occurring GVP of length 3. Thus, by utilizing the offset space, we can efficiently count the number of co-occurring GVP. For example, the number of co-occurring GVP of length 2 in figure 2 is 1 (for bin with D, F) + 1 (for bin with B, F) + $\binom{3}{2}$ (3 choose 2, for bin with A, B, C). After obtaining the dot product, the similarity of the two images is the dot product dividing the L2-norms. The L2-norm of an image I can be calculated by counting the co-occurring GVP with itself, since $\|V^k(I)\| = \sqrt{V^k(I) \cdot V^k(I)}$.

2.3. Searching with GVP

The previous section presented the algorithm for calculating the GVP similarity score of two images. In order to search a large database with this similarity, we integrate the algorithm into the inverted file structure. Suppose for each visual word in the inverted files, other than the images that contain this word, we have also stored the location in which the word occurs. Multiple entries are created if a word occurs multiple times in one image. We discuss about the storage strategy in the later section.

The key idea is to calculate the number of votes in each offset bin for all images in the database during the process of searching. We modify the searching scheme introduced in Section 2.1. Rather than keeping one bin for each image for accumulating the scores, we keep M bins for each image, where M is the number of possible offsets. The voting procedure for obtaining the similarity scores of length k-GVP is as follows.

1. Initialize M bins for each image in the database to 0. Each bin represents an offset value.
2. For each word j in the query image, retrieve the image IDs and locations of the occurrences of j through the inverted files. For each retrieved word occurrence d in image i, we calculate the offset of d and j and increment the corresponding offset bin of image i.

$$S_{i,x_d - x_j, y_d - y_j} += 1 \tag{1}$$

   where, (x_d, y_d) and (x_j, y_j) are the x and y axis of the locations (in the quantized image space) of d and j, S_i are the scores for image i.
3. Calculate the number of co-occurring GVP of length k for each image i by traversing each bin m.

$$\widehat{S}_i = \sum_{m >= k} \binom{S_{i,m}}{k} \tag{2}$$

4. Obtain the final score S_i^* of each image by normalizing $\widehat{S}_i$ with its L2-norm.

2.4. IDF Weighting for Phrases

We further consider the idf weights of visual words. The idf value for a word w_j is calculated as $log(N_j/N)$, where N_j is the number of images that contain w_j and N is the total number of images. For the sake of efficiency, we define the idf weight of a GVP p as the summation of the idf weights of the words composing it: $idf(p) = \sum_{w_j \in p} idf(w_j)$

When the idf weights considered, the dot product of the vector representations of two images is equal to the summation of the idf weights of the co-occurring GVP. Suppose an offset bin of two images has m votes generated from words $w_1, ..., w_m$ (for example, the bin with words A, B, C in figure 2); the summation of the idf weights of GVPs of length k in this bin can calculated as follows:

$$\sum_{p \in \{w_1,...,w_m\}^k} idf(p) = \binom{m-1}{k-1} \sum_{i=1}^{m} idf(w_i) \tag{3}$$

To calculate the similarity scores of all images, we modify the voting procedure in the previous section as follows.

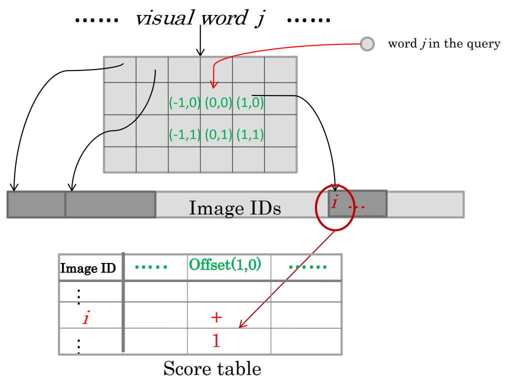


Figure 3. Inverted file structure and the illustration for updating the scores of images with this structure. Green numbers are the offsets of word j and the word occurrences in the database, which are calculated online using the location of j.

For each image i, other than the number of words $S_{i,m}$ in each offset bin m, we also keep the summation of the idf weights of these words $D_{i,m}$. At step 2, for each word j, we update both these values:

$$S_{i,x_d-x_j,y_d-y_j} += 1 \tag{4}$$

$$D_{i,x_d-x_j,y_d-y_j} += idf(j) \tag{5}$$

Equation 2 for calculating the score of image i is changed to:

$$\widehat{S}_i = \sum_{m>=k} D_{i,m} \binom{S_{i,m}-1}{k-1} \tag{6}$$

2.5. Index Structure

We discuss about the storage strategy of the inverted files. For storing the images that contain a visual word, there are two standard strategies for the inverted files [15, 7]. The first one is to keep one entry for each image, and store its ID and the term frequency of this word in this image. The second one is to keep one entry for each word occurrence, and avoid the storage for the term frequencies. With a large vocabulary, the memory usage is almost equivalent for the two strategies, since the same word rarely occurs multiple times in one image [7][3].

To store the locations of the words, one simple way is to adopt the second strategy and associate each word occurrence with the location in which it occurs. Since our algorithm uses only the quantized locations, an alternative storage method is illustrated in figure 3. For each visual word j and each quantized image location (x, y), we store the list of images that contain word j at location (x, y). This data structure only increases the memory by the pointers of the locations. Supposing we have 1M images with 2 billion features and the image space is quantized to 10 by 10 grids, the first method costs additional 2G bytes of memory to store the locations, while the second one only increases the memory by 100M bytes. Another advantage of this structure is that we do not need to calculate the offsets for each word occurrences in step 2. The offset can be calculated before referring to the image lists at a location (x, y) (Figure 3).

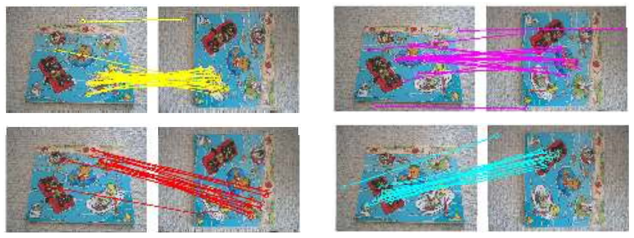


Figure 4. Four example sets of words that have the same offset. The object in the right image is a 90 degree rotation from the same object in the left image.

2.6. Discussion about Invariance

The GVP presented above only captures the translation invariance. More invariance, such as scale or rotation invariance, can be added by increasing the dimension of the offset space (Section 2.2) as in paper [24]. The searching algorithms proposed in this paper will remain the same. However, since adding more dimension increases both the memory usage and the runtime, we prefer to only encode translation invariance into the GVP. In figure 4, we show what will happen when we match two images with a rotation difference. Because of the quantization of the image space, the algorithm matches the words in local neighborhoods. In this case, our approach degenerates to model the local histograms as in spatial pyramid matching [11] and its extension [1]. However, we consider the interactions of all local histogram pairs in contrast to the exact one-to-one matching in [11] and [1].

3. Min-hash with GVP

GVP can also be used to improve the retrieval accuracy of the min-hash method. The min-hash method [4, 6] is one popular dimension reduction technique that reduces the memory usage of inverted files and increases the searching efficiency, and is originally designed based on the bag-of-visual-words model. It is particularly suitable for near-duplicate image retrieval. We briefly review the min-hash algorithm based on BoV.

3.1. Min-hash Review

A number of independent random hash functions are generated. Each hash function f_j randomly assigns a real number to each visual word, and thus defines a permutation of the words. An image I is represented as a set A_I consisting of the words that occur in I. The min-hash value of the image A_I under function f_j is defined as the smallest word in A_I: $mf_j(A_I) = argmin_{w \in A_I} f_j(w)$. We call this mf_j a min-hash function, which has the property that the probability of two sets A_I and $A_{I'}$ having the same min-hash value

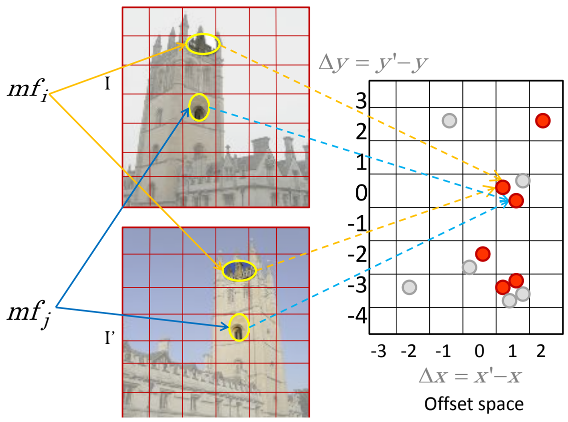


Figure 5. Illustration of the min-hash method with GVP.

is equal to their set overlap. The similarity of two images is defined as their set overlap.

$$p(mf_j(A_I) = mf_j(A_{I'})) = \frac{|A_I \cap A_{I'}|}{|A_I \cup A_{I'}|} = sim(A_I, A_{I'}) \tag{7}$$

If l is the number of times $mf_j(A_I) = mf_j(A_{I'})$ among the N min-hash functions we defined, the similarity of images A_I and $A_{I'}$ can be estimated as l/N.

For efficient retrieval, the min-hashes are grouped into k-tuples $F = (mf_1(A_I), mf_2(A_I), ..., mf_k(A_I))$ called sketches. The probability that two images A_I and $A_{I'}$ has identical k-tuples is $sim(A_I, A_{I'})^k$. The typical retrieval procedure estimates the similarity of the query with only images that have h identical sketches out of S randomly generated sketches.

3.2. Min-hash with GVP

To integrate the GVP to the min-hash algorithm, we first force one-to-one mapping from a min-hash function to a feature (a word occurrence) for each image. A min-hash function mf_j selects the word w in image I, which has the minimum value under function f_j. If the word occurs only once in the image, we set the function value $mf_j(I)$ to this occurrence. If the word occurs multiple times, we randomly select one of them and use the selected one as the function value. In practice, with a large vocabulary, most words has only one occurrence in the same image (more than 95% reported in [3]).

Thus, k min-hash functions will locate k features, a length-k-GVP, in an image. The similarity score $sim^k(I, I')$ of two images I and I' is defined as the probability that the GVP located by a set of k min-hash functions on these images are the same GVP. The calculation of this similarity is illustrated in figure 5. For each min-hash function mf_j, we locate the corresponding feature in each image. If the two features are the occurrences of the same word, that is, the min-hash value of the two images is the same, we calculate their offset and generate a vote on the offset space. k votes at the same bin on the offset space correspond to a co-occurring GVP located by a set of k min-hash functions. Let m_s be the number of votes in bin s of the offset space and N be the total number of min-hash functions used; the similarity score $sim^k(I, I') = \sum_s \binom{m_s}{k} / \binom{N}{k}$. We use this similarity for ranking.

Collision probability

We look into the defined similarity score $sim^k(I, I')$, which is also the probability that a GVP collision occurs. This also equals to the probability that for k min-hash functions, 1) the min-hash values of all functions are the same for the two images; 2) the located feature pairs generate votes in the same bin on the offset space. Suppose the two images have $\rho^1(I, I')$ number of same word pairs. This also indicates that if we generate one vote for each same word pair (gray and red circles in figure 5), we have $\rho^1(I, I')$ total number of votes on the offset space. Because of its randomness, when a min-hash function has the same min-hash values for images I and I', it will randomly generate one vote among the $\rho^1(I, I')$ votes (red circles). Therefore, let M_s be the number of votes (gray and red circles) in offset bin s, and $sim(A_I, A_{I'})$ be the word set overlap defined in equation 7; the similarity score is:

$$sim^k(I, I') = sim(A_I, A_{I'})^k \frac{\sum_s M_s^k}{(\sum_s M_s)^k} \tag{8}$$

$$= sim(A_I, A_{I'})^k \frac{\rho^k(I, I')}{\rho^1(I, I')^k} \tag{9}$$

where $\rho^k(I, I')$ is the total number of co-occurring GVP of length k[2].

Note that the left part of equation 9 is the probability of a sketch collision. By using GVP, we further decrease the collision probability. Table 1 gives typical probabilities that a pair of relevant images has at least one sketch or GVP collision among S number of sketches or GVP[3]. Even for length 2, the probability is quite low for at least one GVP collision among 1024 GVPs. The proposed algorithm works because with N number of min-hash functions, we have already considered $\binom{N}{k}$ number of GVP. Therefore, when we have 512 min-hash functions, we can ensure to have at least one GVP collision with probability 1.0 for both $k = 2$ and $k = 3$.

4. Experiments

We evaluate our approach based on the three important factors in image retrieval: retrieval accuracy, memory usage and searching time.

[2]The GVP here include the phrases generated using the same word multiple times

[3]The probability is calculated as the median of the probabilities of the relevant image pairs on the University of Kentucky dataset [13] with a 100K vocabulary

method	S=512		S=1024	
	k=2	k=3	k=2	k=3
sketch	0.61	0.04	0.85	0.08
GVP	0.075	0.002	0.15	0.004

Table 1. The probabilities that at least one sketch or GVP collision for relevant image pairs among S number of sketches or GVP. k is length of the sketch or GVP.

Datasets: The experiments are conducted on three publicly available image retrieval datasets: *Oxford 5K Dataset*[15], *Flicker 1M dataset*[7], and *University of Kentucky dataset*[13]. *Oxford 5K dataset* contains 5062 images with more than 16M features. It also provides 55 test queries of 11 Oxford landmarks with their ground truth retrieval results. *Flicker 1M dataset* contains 1M images with more than 2 billion features. This dataset is added as distractors to the Oxford dataset to test the scalability of our system. *University of Kentucky dataset* contains 10,200 images of 2550 objects where each object has four images. Using each image to query the database should return the four images of the same object. To ensure the compatibility of the experiments, we use the public available descriptors (SIFT features on hessian affine regions or maximally stable extremal regions) on the dataset web pages. To create the vocabulary, we adopt the approximate k-means [15] whose code is published by the authors.

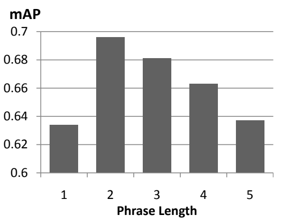
Baseline: We posit our approach as an efficient algorithm that improves the initial ranking at the searching step. Therefore, we consider the searching with bag-of-visual-words (BoV) as our baseline. We also compare favorably with other works that encode spatial information to BoV at the searching step. Methods that improve the visual word quantization or methods performed at the post-processing step are complimentary to our method.

4.1. Inverted Files with GVP

We evaluate our searching method with the inverted file structure (method in Section 2) on the Oxford 5K and Flicker 1M dataset. We use the same experiment settings as previous papers [15]. The retrieval accuracy is measured with the mean average precision (mAP) score generated as follows: the precision-recall curve is created for each query to calculate the average precision (AP). The mAP is defined as the mean of the APs of all queries.

Parameter effects: We first examine the effect of the main parameters in our approach on the Oxford 5K dataset. All the experiments here are conducted with 1M vocabulary size. Table 2(a) shows the mAP scores of different lengths of the geometry-preserving phrases phrases (GVP). We quantize the image space with 10 by 10 steps[4]. The

[4]To lessen the problem of hard quantization of the image space, and we also quantize the offset space by a factor of 2, that is, we merge the neighboring offset bins. The left sides of Equation 4 and 5 will be changed to $S_{i,\lfloor(x_d-x_j)/2\rfloor,\lfloor(y_d-y_j)/2\rfloor}$ and $D_{i,\lfloor(x_d-x_j)/2\rfloor,\lfloor(y_d-y_j)/2\rfloor}$



# steps	mAP	# r. Imgs
2	0.657	3949
5	0.682	3334
10	0.696	2597
15	0.698	1510
20	0.692	1354

(a) (b)

Table 2. The effect of parameter changes on Oxford 5K dataset. (a): mean average precisions with GVP of different lengths. Length 1 corresponds to the BoV model. (b): the change of mAP and average number of retrieved images when changing the number of quantization steps of the image space.

BoV model (length 1) has a mAP score of 0.634, and using GVP of length 2 improves the mAP score to 0.696. The searching algorithm with a longer GVP corresponds to a more rigorous geometry modeling. Length 2 is also the optimal length among the lengths from 1 to 5. Since we only retrieve images with common GVP, a longer GVP may also leads to a lower recall when the shape deformation of the relevant images is large. Table 2(b) shows the effect of different number of steps for quantizing the image space. The same number of steps is used for x and y axis of the image space. We use GVP of length 2 in these experiments. A larger number of steps corresponds to a more rigorous spatial modeling. Increasing the number of steps also increases the memory usage for storing the accumulated scores in the offset bins. Therefore, we choose to use 10 steps with GVP of length 2 in our retrieval system.

Comparison: Table 3 compares the retrieval accuracy (mean average precision) of our approach with the other methods under different vocabulary sizes on the Oxford 5K dataset. The results showed that encoding spatial information (GVP) to the BoV model (BoV) can significantly improve the retrieval accuracy. More significant improvement is made on smaller vocabulary sizes, because the visual words are more ambiguous. Searching with GVP also performs better than the BoV model plus a RANSAC post-processing (BoV+RANSAC, for this method, we cite the results reported in [15]), especially on larger vocabularies. The reason is that our approach can provide spatial examination for the whole database, while the RANSAC only runs on the top images resulted by BoV and thus will be affected a lot when the precision of the top images is low. On the smaller dictionary, BoV+RANSAC performs better than GVP. The reason is that when the visual words are very ambiguous, the GVP will also be affected, while the RANSAC is less influenced since it is running on raw features. We further apply RANSAC on the top 400 images ranked with the GVP (GVP+RANSAC), which provides the best results among these methods. The improvement from GVP to GVP+RANSAC is resulted from the spatial information the GVP failed to capture, such as scale or affine

Vocab	BoV	GVP	BoV + RANSAC	GVP + RANSAC
50K	0.490	0.594	0.599	0.614
100K	0.536	0.607	0.597	0.622
250K	0.598	0.672	0.633	0.688
500K	0.626	0.683	0.642	0.699
1M	0.634	0.696	0.653	0.713

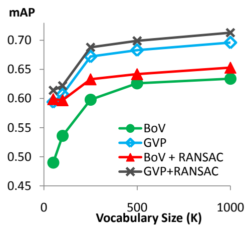


Table 3. Comparison of the performances of GVP and BoV with different vocabulary size for the Oxford 5K dataset.

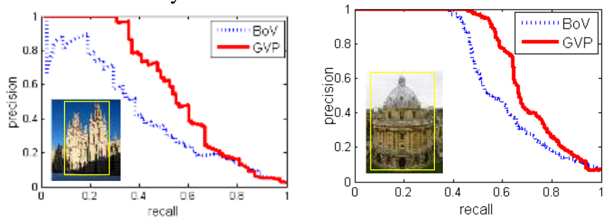


Figure 6. The precision-recall curve for example queries on the Oxford 5K dataset. The left one is for query *all_souls_1*, and the right one is for *radcliffe_camera_3*.

transformation, or the errors in GVP caused by visual word quantization.

We also compare the proposed GVP with the Spatial Bag-of-Feature method (SBOF) [1], which also integrate spatial information into the BoV model. The SBOF uses the concatenated histogram of the histograms in local bins and introduces the transformation invariance by reordering the histogram so that it starts from the bin with maximum number of words. We encode more spatial information by considering the interaction between the local bins, and our method is more robust to invariance since we do not fix one-to-one matching between bins of two images. They report 0.651 mAP with the SBOF on the 1M vocabulary [1], while the proposed GVP achieved 0.696 mAP.

Analysis: Figure 6 presents the precision-recall curves of the BoV model and the proposed GVP for two example queries. These curves are also typical among the curves for other queries. Our approach with GVP improve the precision of the BoV model at lower recalls. Close precision is shown when the recall reaches a certain point. We found the reason of this phenomenon is as follows. There are some relevant images which have few matched visual words with the queries. For these images, we can neither find co-occurring GVP with the queries. Therefore, our method with GVP fails to improve the ranks of these images.

Flicker 1M: We show the retrieval accuracy when adding the Flicker 1M images in figure 7. The proposed method with the GVP consistently improve the ranking results of the BoV model on different number of images. The GVP outperforms the BoV by 12% on the 1M images.

Computational cost: We run our experiments on a *single CPU* of a 2.26G Quad-Core Intel Xeon server with 12G memory. The Flicker 1M + Oxford 5K dataset contains around 2G features and therefore, the size of the inverted files is around 8G. Compared with the BoV model, the GVP method needs additional memory to store the relative pointers of the locations as presented in figure 3 and the scores of offset bins. The additional memory required is around 500M, insignificant comparing to the inverted files. Table 4 summarizes the memory usage as well as the running time. Feature extraction time is not included. The proposed GVP achieves a significant improvement in retrieval accuracy with little speed penalty. In comparison, the RANSAC spatial verification on top 400 images takes more than 4 seconds per query. The increased runtime (around 0.1 seconds) is mainly because the scores of the offset bins cannot be saved to the Cache (8M), and the update of them requires random access to the memory.

# Images	BoV	GVP
105K	0.509	0.604
205K	0.479	0.581
505K	0.443	0.554
1M+5K	0.413	0.532

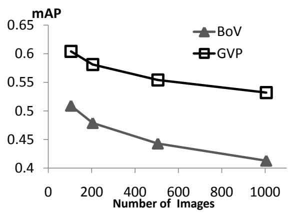


Figure 7. The mean average precision of Oxford 5K dataset combined with Flicker 1M images as distractors.

Method	Memory	Runtime	
		Quantization	Search
BoV	8.1G	0.89s	0.137s
GVP	8.6G		0.248s

Table 4. The memory usage and average runtime per query on the Flicker 1M dataset.

4.2. Min-hash with GVP

We evaluate the proposed min-hash method with the geometry-preserving visual phrases (GVP) on the University of Kentucky dataset [13] which is also used in the previous min-hash paper [6]. We use the same experiment setting as in [13, 6]. The retrieval accuracy is measured as the average number of relevant images in the top four retrieved images. We choose to integrate the GVP into the min-hash method with the similarity function using histogram intersection [6], since the method with histogram intersection reports the best results in [6]. We use sketches of length 2 in the same way as the traditional min-hash, and compute similarity scores with GVP for images with at least one sketch collision with the query image. We adopt the same number of sketches with which the best performance is achieved for each vocabulary size in [6]. We build a 100K vocabulary using the same features as [4].

Table 5 presents the average number of relevant images in the top 4 images for the min-hash method with BoV and GVP. The results using different number of min-hash functions are presented. By encoding the spatial information using the proposed GVP, our approach outperforms the min-hash method with BoV. The improvement is less significant

method	# of min-hash functions				
	268	512	640	768	896
BoV	2.88	3.07	3.12	3.14	3.17
GVP	2.96	3.14	3.20	3.24	3.26

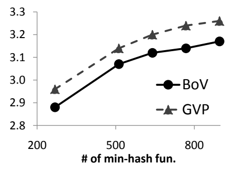


Table 5. The number of relevant images in top 4 images on the University of Kentucky dataset for the min-hash methods with BoV and GVP (length 2).

when fewer min-hash functions are used, because the probability of a GVP collision is low in this case.

4.3. Discussion

We discuss the limitations of our approach in this section. First, like most other image retrieval systems, our approach is built upon the visual word representation. As already discussed, if few matched visual words are found for the relevant images, our algorithm cannot correct them. This problem may be solved with a query expansion [5] on raw features in the post-processing step. Second, since we model the spatial interaction among the visual words, our current algorithm can not be applied to some dimension reduction methods that conduct linear transformations on the histogram of visual words [9, 22, 8]. However, as we have shown, as long as the dimension reduction method retains the word representation as the min-hash method [4], our approach is still applicable.

5. Conclusion

We proposed an approach that encodes more spatial information into the bag-of-visual-words (BoV) model at the searching step of a retrieval system. The spatial information is encoded with the geometry-preserving visual phrases that models local and long-range spatial interactions between the visual words. Our approach can deal with all possible phrases without a learning step in which a set of phrases are selected. The experiment results showed that our approach outperforms the BOV model plus a RANSAC post-processing while requiring similar memory usage and computational time compared as those of the BoV model.

References

[1] Y. Cao, C. Wang, Z. Li, L. Zhang, and L. Zhang. Spatial-bag-of-features. In *CVPR*, 2010. 810, 812, 815

[2] O. Chum and J. Matas. Unsupervised discovery of co-occurrence in sparse high dimensional data. In *CVPR*, 2010. 810

[3] O. Chum, M. Perdoch, and J. Matas. Geometric min-hashing: finding a (trick) needle in a haystack. In *CVPR*, 2009. 810, 812, 813

[4] O. Chum, J. Philbin, M. Isard, and A. Zisserman. Scalable near identical image and shot detection. In *CIVR*, 2007. 810, 812, 815, 816

[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 *ICCV*, 2007. 809, 816

[6] O. Chum, J. Philbin, and A. Zisserman. Near duplicate image detection: min-hash and tf-idf weighting. In *BMVC*, 2008. 810, 812, 815

[7] H. Jegou, M. Douze, and C. Schmid. Hamming embedding and weak geometric consistency for lage scale image search. In *ECCV*, 2008. 809, 810, 812, 814

[8] H. Jegou, M. Douze, and C. Schmid. Packing bag-of-features. In *ICCV*, 2009. 809, 816

[9] H. Jegou, M. Douze, C. Schmid, and P. Perez. Aggregating local descriptors into a compact image representation. In *CVPR*, 2010. 809, 816

[10] C. H. Lampert. Detecting objecs in large image collections and videos by efficient subimage retrieval. In *ICCV*, 2009. 810

[11] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: spatial pyramid matching for recognizing natural scene categories. In *CVPR*, 2006. 810, 812

[12] Z. Lin and J. Brandt. A local bag-of-features model for large scale object retrieval. In *ECCV*, 2010. 810

[13] D. Nister and H. Stewenius. Scalable recognition with a vocabulary tree. In *CVPR*, 2006. 809, 813, 814, 815

[14] M. Perdoch, O. Chum, and J. Matas. Efficient representation of local geometry for large scale object retrieval. In *CVPR*, 2009. 809

[15] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *CVPR*, 2007. 809, 810, 812, 814

[16] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Lost in quantization: improving particular object retrieval in large scale image databases. In *CVPR*, 2008. 809

[17] J. Sivic and A. Zisserman. Video google: a text retrieval approach to object matching in videos. In *ICCV*, 2003. 809, 810

[18] L. Torresani, M. Szummer, and A. Fitzgibbon. Learning query-dependent prefilters for scalable image retrieval. In *CVPR*, 2009. 810

[19] Y. Weiss, A. Torralba, and R. Fergus. Spectral hashing. In *NIPS*, 2009. 810

[20] Z. Wu, Q. Ke, M. Isard, and J. Sun. Bundling features for large scale partial-duplicate web image search. In *CVPR*, 2009. 810

[21] J. Yuan, Y. Wu, and M. Yang. Discovery of collocation patterns: from visual words to visual phrases. In *CVPR*, 2007. 810

[22] X. Zhang, Z. Li, L. Zhang, W. Ma, and H.-Y. Shum. Efficient indexing for large scale visual search. In *ICCV*, 2009. 809, 816

[23] Y. Zhang and T. Chen. Efficient kernels for identifying unbounded-order spatial features. In *CVPR*, 2009. 810, 811

[24] Y. Zhang and T. Chen. Weakly supervised object recognition and localization with invariant high order features. In *BMVC*, 2010. 812

[25] C. L. Zitnick, J. Sun, R. Szeliski, and S. Winder. Object instance recognition using triplets of feature symbols. *Tech. Report, Microsoft Research*, 2007. 810