Mobile Visual Search with Word-HOG Descriptors

Sam S. Tsai, Huizhong Chen, David M. Chen, and Bernd Girod

Department of Electrical Engineering, Stanford University, Stanford, CA, 94305
sstsai@alumni.stanford.edu, {hzchen2,dmchen,bgirod}@stanford.edu

Abstract

Visual text information is a descriptive part of many images that can be used to perform mobile visual search (MVS) with particularly small queries. In this paper, we propose a system that uses word patch descriptors for retrieving images containing visual text. A random sampling method is used to find duplicate word patches in the database and reduce the database size. The system achieves comparable retrieval performance to state-of-the-art image feature-based systems for images of book covers, and performs better than state-of-the-art text-based retrieval systems for images of book pages. Using visual text to provide distinctive features, our system achieves more than 10-to-1 query size reduction for images of book covers and more than 16-to-1 query size reduction for images of book pages.

1 Introduction

In recent years, the number of consumer applications that are built on mobile visual search (MVS) technology has increased substantially. Some well known examples of MVS applications in industry include Google Goggles[1], Amazon Flow[2], and Kooaba[3]. These system are based on robust local image features that permit successful matching of camera phone images with database images despite severe geometric and photometric distortions. For queries that contain primarily visual text information, however, the retrieval performance of conventional image feature-based approach is typically poor. The majority of the interest points detected in such images are at repetitive and non-discriminative locations. When extracting descriptors from these points, we obtain similar descriptors for different text documents, leading to an image representation that is ineffective at image matching.

In this work, we focus on developing a new MVS system that effectively exploits the special properties of visual text. We detect the text regions within the image, and extract a text-oriented feature descriptor called the Word-HOG descriptor [1] to describe the text regions. Using this representation, we developed an MVS system that matches images containing visual text information much more effectively than conventional approaches. The rest of the paper is organized as follows. Section 2 first introduces related work. Then, Section 3 describes of the Word-HOG-based MVS system. In Section 4, experimental results on performing image retrieval on images with visual text are presented. We show that leveraging visual text via the Word-HOG descriptor reduces the query size by an order of magnitude while achieving the same or better retrieval performance than conventional local image features.

[1]Google goggles: http://www.google.com/mobile/goggles
[2]Amazon Flow: http://flow.a9.com
[3]Kooaba: http://www.kooaba.com

(a) Dense-text images (Book Pages) (b) Sparse-text imagse (Book Covers)

Figure 1: Examples of the dense-text and sparse-text images.

2 Background

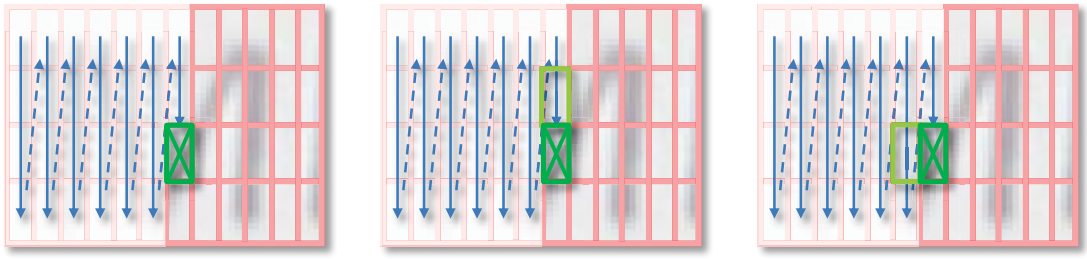
Images with visual text can be categorized into either dense-text images or sparse-text images, as shown in Fig. 1. Most research has focused on retrieval for either dense-text images or sparse-text images, but not both. For dense-text images, features that use the locations of text words [2, 3] have been used for document image retrieval. Another class of algorithms use shape or image information extracted from single characters [4]. Yet another class of algorithms directly recognize text in the dense-text images and use the recognized text for retrieval [5]. For sparse-text images, algorithms have been developed that use image-based features extracted from character locations [6], or use text recognized from detected text locations using optical character recognition (OCR) engines [7]. Note that the text recognition accuracy is lacking when OCR is applied to sparse-text images with severe geometric and photometric distortions. Most OCR algorithms are not designed to handle both dense- or sparse-text images using the same feature extraction pipeline.

3 Word-HOG-based Mobile Visual Search

The proposed Word-HOG-based MVS system operates at two levels. The lower word level comprises extracting Word-HOG descriptors and using the Word-HOG descriptors to perform word patch matching. The algorithms used have been presented recently in our paper [1], but will be summarized in Sec. 3.1 for the convenience of the reader, along with some previously unpublished experimental results on the compression of Word-HOGs. The higher image level processing uses the Word-HOG matching results to perform image retrieval. This includes merging multiple results and using the locations of the Word-HOG descriptors to perform geometric verification and is explained in Sec. 3.2. For large databases, we perform Word-HOG de-duplication to reduce the database size. To do this efficiently, we use a random sampling scheme which we present in Sec. 3.3.

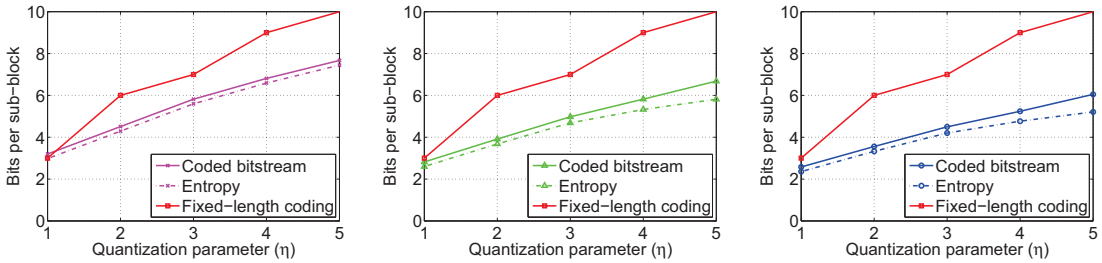
3.1 Overview of Word-HOG Descriptor

A Word-HOG descriptor is formed by extracting gradient orientation histograms from sub-blocks within a word patch. Using the sub-block histograms, WSIFT descriptors can be assembled to perform vocabulary tree-based word patch matching from a database of word patches. To geometrically verify two word patches, the two sets of



(a) Scan-order (b) Vertical context (c) Horizontal context

Figure 2: Context-based arithmetic coding for compression of Word-HOG descriptors..



(a) Scan-order (b) Vertical context (c) Horizontal context

Figure 3: Average bits used per sub-block within the Word-HOG descriptor for different arithmetic coding contexts, for the word patch databases presented in [1]. The quantization parameter η controls the coarseness of an A_7 lattice on the probability manifold, see [1] for details.

WSIFT descriptors are paired with one another and a geometric transformation is found between the two sets.

To compress the Word-HOG descriptor, lattice quantization is used to quantize the gradient histogram counts in each spatial sub-block. The lattice indices are then compressed by arithmetic coding in the order shown in Fig. 2(a). We experimented with using the vertical and horizontal sub-block's lattice index as context (Fig. 2(b) and 2(c), respectively) and found that using horizontal context provides the best compression gain, see Fig. 3.

Additional to the compressed Word-HOG descriptor, the geometry information of the text is also used in the image retrieval stage, including the top-left corner of the visual text box, the width and height of the box, and the orientation of the text direction. This information is encoded in the packet using 10 bits for all except for the orientation, which is encoded using 8 bits.

3.2 Image Retrieval with Word-HOGs

Word-HOG descriptors are used as the basic elements to perform image retrieval in the proposed MVS system. An image is represented as a bag-of-Word-HOGs. Figure 4 illustrates the training pipeline. First, text detection [8] is used to find the

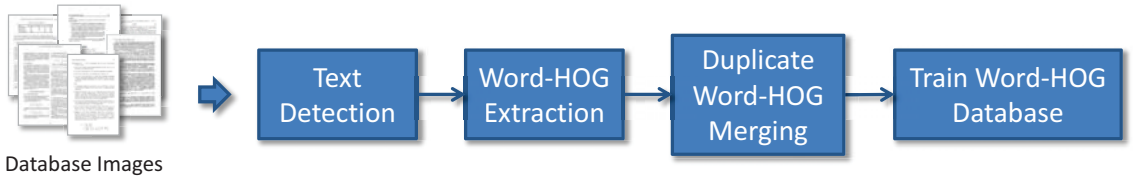


Figure 4: Overview of the training process for the proposed retrieval framework with duplicate Word-HOG detection.

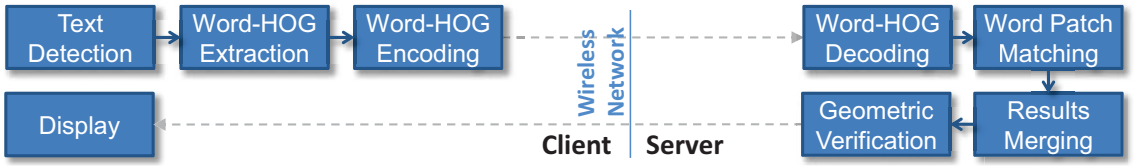
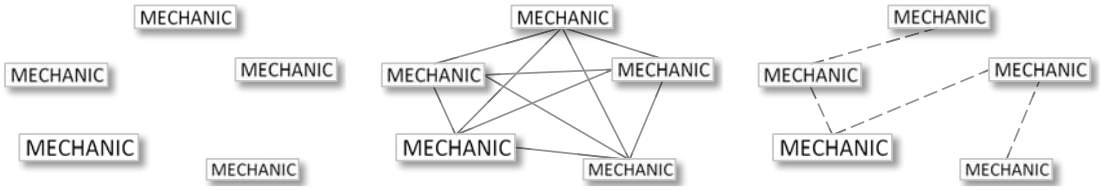


Figure 5: Overview of the querying process of the image-based retrieval system that uses visual text information.

text locations within database images. Database Word-HOG descriptors are then calculated from these locations. A de-duplication stage is used to reduce the number of duplicates within the set of database Word-HOG descriptors. The remaining ones are used to generate a word patch matching database.

When querying the image retrieval pipeline, the query goes through a system as shown in Fig. 5. On the mobile device, text is detected and Word-HOG descriptors are extracted and encoded. Then, the query data are sent to the server for image retrieval. The encoded query Word-HOG descriptors is decoded and matched against the word patch matching database. Each query Word-HOG would obtain a matching candidate list. To merge then into a single list, we use an online *idf*-scoring and the *tf*-scoring method. For each query result list, the geometric matching scores of entries in the result list are normalized by the highest score. Normalized scores lower than a cut-off threshold, θ_c, are discarded. Then, *idf* weighting for the query is generated using the remaining matching documents. Let ω_j^{idf} be the weighting for the j^{th} query. For each database image, only a single score from a query Word-HOG is used. Let $s_{j,D}$ be the maximum score that the j^{th} query contributes to database image D. Then, the score of the D^{th} database image is $s_D = \sum_j \omega_j^{idf} \cdot s_{j,D}$. The merged list uses these score to rank the database images.

Following the merging stage, an image-level geometric verification method is used to find the correct matching database image from the merged list. The image-level geometric verification is a two-step process, similar to past approaches [9] wherein a descriptor pairing step is followed by a geometric transformation model estimation step. The paired candidate Word-HOG descriptor of a query Word-HOG descriptor is the Word-HOG in the database image that gives the highest geometric matching score. Then, the geometric model estimation step uses the locations of the matched WSIFT descriptors to estimate a consistent geometric model. By using the locations of the WSIFT descriptors instead of a whole text box geometry, the geometric verification process is able to use matches that come from partially matched words.



(a) Duplicate Word-HOGs (b) Pairwise Connections (c) Edges for Connected Set

Figure 6: Duplicate Word-HOG descriptors do not need to be fully compared to find a connected set. In fact only a small number of connections are needed to make the full set connected.

3.3 Database Word-HOG De-duplication

For large databases, near-duplicate Word-HOG descriptors are very common. To find duplicate descriptors, one can compute the geometric matching score for every pair of database Word-HOG descriptors. For large databases, this would be computationally expensive. Fortunately, in a duplicate set of Word-HOG descriptors, only a small set of pairwise comparisons are needed to connect a duplicate set. As shown in the toy example in Figure 6, a database of five duplicates requires a total of 4 connections where as the total number of pairwise connections is 20.

For the Word-HOG database, it is not possible to know a priori which Word-HOG descriptors are duplicates. Thus, a random sampling approach is used to find and connect the duplicate Word-HOG descriptors. Random pairs are selected from the set of database Word-HOG descriptors and the pairwise geometric matching score is calculated between the two. If the score is greater than a threshold, the two Word-HOG descriptors are considered a duplicate.

To quantify the number of random samples needed, we consider Word-HOG descriptors as nodes and their pairwise similarity as edges in a graph. Let M be the number of nodes. For each Word-HOG, we randomly sample k other Word-HOG descriptors to form edges. Then, the probability of an edge in the graph being selected is $p_c = k/M$.

Suppose we have a duplicate Word-HOG group of size α. The maximum number of edges in the graph of the group is $\alpha \cdot (\alpha - 1)$. We can calculate the probability that there will be γ number of edges in the group using the Binomial distribution:

$$P_e(\gamma) = \begin{pmatrix} \alpha \cdot (\alpha - 1) \\ \gamma \end{pmatrix} \times p_c^{\gamma} \times (1 - p_c)^{\alpha \cdot (\alpha - 1) - \gamma}. \quad (1)$$

For large α, we can approximate the distribution using a Gaussian distribution according to the central limit theorem. Then, the approximated distribution is $\mathcal{N}(\mu, \sigma)$, with $\mu = \alpha \cdot (\alpha - 1) \cdot p_c, \sigma = \sqrt{\alpha \cdot (\alpha - 1) \cdot p_c \cdot (1 - p_c)}$. Thus, with probability greater than 0.99, we can determine the number of edges found in the duplicate Word-HOG group greater than N_c using the Gaussian distribution CDF:

$$N_c = \mu - 3\sigma = \alpha \cdot (\alpha - 1) \cdot p_c - 3 \cdot \sqrt{\alpha \cdot (\alpha - 1) \cdot p_c \cdot (1 - p_c)}.$$

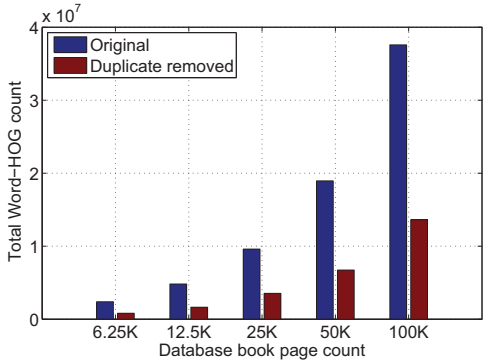


Figure 7: The number of Word-HOGs in the original vs. duplicates-removed database. Deduplication was performed on 2M batches and pairwise comparisons were made only when two word patches produce WSIFT counts that differ at most by one.

In the random sampling, we target large groups so $\alpha \gg 0$. Then, N_c can be approximated as follows:

$$N_c \cong \alpha \cdot (\alpha - 1) \cdot p_c = \alpha \cdot (\alpha - 1) \cdot k/M. \tag{2}$$

In [10], Erdős derived the asymptotic probability of a random graph being completely connected as:

$$\lim_{\alpha \longrightarrow \infty} P_c(\alpha, N_e) = e^{-e^{-2x}}, \tag{3}$$

where α is the number of nodes, N_e is the number edges, and $x \in R$. The three variables have the following relation: $N_e = \lfloor 0.5 \cdot \alpha \cdot \log \alpha + x \cdot \alpha \rfloor$. Thus, assuming that we wish to attain a probability of the graph being connected of $1 - 10^{-8}$, we can set an operating point at $x = 3$ for (3). Then, with (2), the amount of sampling we need is the following:

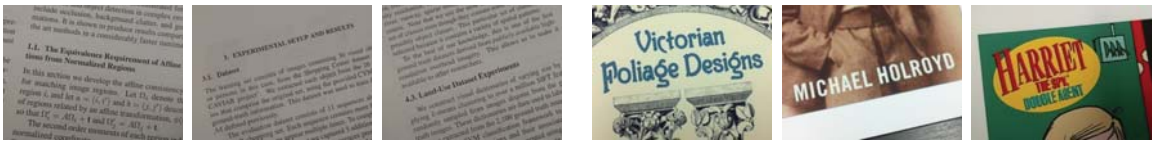
$$\begin{aligned} N_c &> N_e & (4)\\ \alpha \cdot (\alpha - 1) \cdot k/M &> \lfloor 0.5 \cdot \alpha \cdot \log \alpha + 3 \cdot \alpha \rfloor & (5)\\ k &> \frac{M \cdot \lfloor 0.5 \cdot \alpha \cdot \log \alpha + 3 \cdot \alpha \rfloor}{\alpha \cdot (\alpha - 1)}. & (6) \end{aligned}$$

Thus, we need a sampling rate of k that is greater than that in (6).

Compared to performing full pairwise distance computation, we reduce computation by a factor of

$$\begin{aligned} r &= \frac{(M-1) \cdot M}{k \cdot M} = \frac{M-1}{k} = \frac{(M-1) \cdot \alpha \cdot (\alpha - 1)}{M \cdot \lfloor 0.5 \cdot \alpha \cdot \log \alpha + 3 \cdot \alpha \rfloor} & (7)\\ &\cong \frac{\alpha \cdot (\alpha - 1)}{\lfloor 0.5 \cdot \alpha \cdot \log \alpha + 3 \cdot \alpha \rfloor}. & (8) \end{aligned}$$

For example, with $\alpha = 1,000$, we speed up computation by a factor of $r = 154.8$. From a database of dense-text images (see Sec. 4), we were able to reduce the number of total Word-HOG counts by $\sim 3\times$, as shown in Fig. 3.3.



(a) Dense-text images (Book Page) (b) Sparse-text images (Book Cover)

Figure 8: Sample query images of the two datasets.

4 Experimental Results

To evaluate the proposed MVS system, we use two different datasets that represent two types of images with visual text. One dataset is a dense-text image dataset which we call the **Book Page** dataset that consists of printed text documents consisting of online electronic books and conference proceedings. The other dataset is a sparse-text image dataset that consists of book covers from the Open Library archive[4] which is referred to as the **Book Cover** dataset.

To generate the query images, images from the database images are randomly selected and printed. Mobile devices are used to take pictures of the text regions in the printed image with a resolution of 640×480. For the Book Page dataset, we gathered 100,000 database images and 350 query images. For the Book Cover dataset, we gathered one million database images and 420 query images. Sample images of the database and query images are shown in Fig. 1 and Fig. 8, respectively. For both query datasets, a fifth of the query images are used for training while the rest are used for testing.

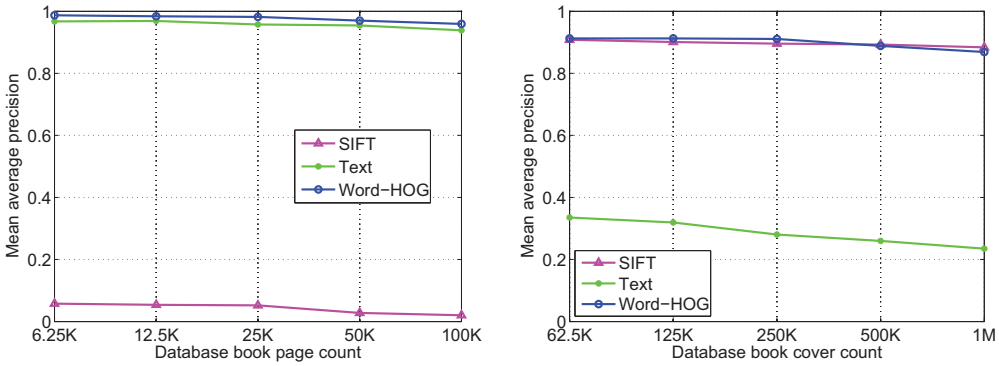
In the following, Section 4.1 first evaluates the image retrieval performance of the Word-HOG-based approach without compression. Then, Section 4.2 will present the results on performing retrieval with various rates. To evaluate the system quantitatively, we use the mean average precision (MAP), which considers the performance of the whole list.

4.1 Retrieval Performance with Uncompressed Queries

To test the image retrieval performance of the system, we vary the database size and test the retrieval performance. We compare the retrieval performance to two other types of system: (1) an image feature-based retrieval system, and (2) a text-based retrieval system. For the image feature-based retrieval system, difference of Gaussian interest points and SIFT descriptors are extracted from the images. For retrieval, vocabulary tree and geometric verification as described in [9] is used to find a correct match. The text-based retrieval system uses text extracted from the images to perform retrieval. Text is extracted from images by using Tesseract OCR engine[5] to recognize words from the detected text location in images. Each database image is represented as a text document containing the recognized words from the image, and recognized text from the query image is used to search for a matching database image using text-based search.

[4]Open Library: http://openlibrary.org

[5]Tesseract OCR engine: http://code.google.com/p/tesseract-ocr



(a) Book Page Dataset (b) Book Cover Dataset

Figure 9: Comparison of retrieval performances for three types of systems.

Figure 9(a) shows the MAP retrieval performances of the three systems for database sizes ranging from 6,250 to 100,000 for the Book Page dataset. The best performing system is the Word-HOG-based system. As the database size increases, the MAP starts to decrease. At a database size of 100,000, an MAP of 0.96 is achieved. The text-based system performs similarly and achieves a MAP of 0.94 for the same database size. In contrast, the retrieval performance of the SIFT-based system is much worse. This poor performance is because interest points detected are located at small scales, and descriptors extracted from these points are not distinctive and helpful for retrieval.

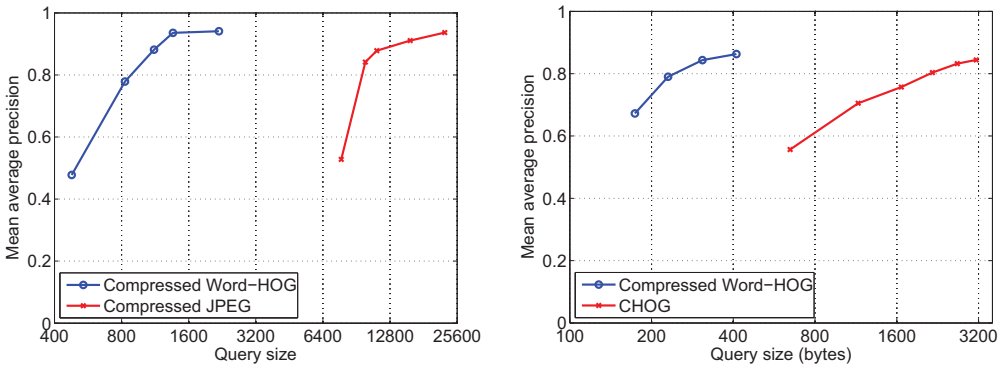
The same comparison is made for the Book Cover dataset in Figure 9(b). For this comparison, the database size is varied from 62,500 to one million. We can see that the retrieval performances are different from those of the Book Page dataset. The Word-HOG-based and SIFT-based system achieves the best retrieval performance. At a database size of 62,500, the MAP of the two systems are both ~ 0.91. At a database size of one million, the MAP drops slightly to 0.88 (SIFT) and 0.87 (Word-HOG). The text-based system performs poorly because of the great variations of font styles.

Text-based systems perform well when performing retrieval on images that consist of printed text but fails noticeably when dealing with text that has more artistic designs and is placed on cluttered backgrounds. In contrast, SIFT-based systems are unable to find database matches for document images because the interest point detector fails in finding interest points that produce useful image features, but excels when the text is designed more artistically and embedded in cluttered background. Only the Word-HOG-based system works well on both dense and sparse images.

4.2 Retrieval Performance with Different Query Sizes

In this section, we present the retrieval performance with different query sizes. To construct compressed queries with different sizes, we vary the number of Word-HOG packets that are in the query. To choose which packets to send, we use a selection scheme that is based on the intuition that longer words are more distinctive.

The results are compared with two other types of systems: (1) a compressed image-based system, and (2) a compressed image feature-based system. For the



(a) Book Page Dataset (b) Book Cover Dataset

Figure 10: Retrieval performance comparison of the MVS systems.

compressed image-based systems, we extract text from the image and use the text to perform retrieval. No image processing is required on the client side and a JPEG compressed image is sent as a query. For the compressed image feature-based system, we extracted CHOG features [11] from the query image on the mobile device and use the CHOG features as queries. This approach produces query data that are much smaller in size when compared to that of the compressed image-based system.

From the comparisons described in the Section 4.1, we learned that text-based methods do not work well for retrieving Book Cover images and image feature-based methods do not work well for retrieving Book Page images. Therefore, to make the comparisons simpler, only the results of the two better performing systems are compared. For the Book Page and the Book Cover dataset, we use a database of 100,000 and one million images respectively.

Figure 10(a) shows the average compressed query size versus the retrieval performance for the Book Page dataset. For the compressed JPEG-based system, since the whole image is sent over the network, the query size is typically much larger. When the image is compressed with a size of 22.5K bytes, the MAP is 0.94. At a query size of 7.7K bytes, the MAP is only 0.53. For the compressed Word-HOG-based system, the retrieval performance is better with smaller queries. With a query size of 1.4K bytes, the retrieval performance is high, with an MAP of 0.94. At a query size of 1.1K bytes, the MAP is at 0.88. At an MAP of 0.94, the query data size of the Word-HOG-based system is 16.6 times smaller than that of the compressed JPEG-based.

Figure 10(b) shows the average compressed query size versus the retrieval performance for the Book Cover dataset. For the CHOG-based system, the number of features per query is varied within 50 to 300, resulting in query data sizes from $\sim$ 647 to $\sim$ 3.1K bytes. With a query of $\sim$ 3.1K bytes, the retrieval performance MAP is only 0.84. For the compressed Word-HOG-based system, the retrieval performance is high with much lower rates. With a query size of 411 bytes, the retrieval performance MAP is 0.86. At a query size of 230 bytes, the MAP drops to 0.79. If we compare the two systems at an MAP of 0.84, the query data size of the compressed Word-HOG-based system has a query size that is 10.2 times smaller than that of the CHOG-based system.

5 Conclusions

A new MVS system that exploits visual text information has been proposed. The system uses a highly compressible word patch descriptor called Word-HOG to describe visual text in images, and uses the Word-HOG descriptor to perform image retrieval. Matching database images are ranked according to the Word-HOG matching scores with an adapted *tf-idf* weighting scheme, and passed through an image-level geometric verification method to improve the accuracy. To reduce the database size, a de-duplication stage that randomly samples pairs of Word-HOG descriptors is used to find the duplicates. Without compression, the image retrieval method is shown to perform as well as image feature-based retrieval methods for sparse-text image datasets, and perform as well as text-based systems for dense-text image datasets. With compression, the proposed system performs much better than the other two system. For dense test and sparse-text image datasets, the proposed system is capable of performing as well as other approaches with queries that are 16× and 10× smaller, respectively.

References

[1] S. S. Tsai, H. Chen, D. M. Chen, and B. Girod, "WORD-HOGs: Word histogram of oriented gradients for mobile visual search," in *IEEE International Conference on Image Processing*, October 2014, pp. 451–452.

[2] K. Takeda, K. Kise, and M. Iwamura, "Real-time document image retrieval for a 10 million pages database with a memory efficient and stability improved LLAH," in *International Conference on Document Analysis and Recognition*, September 2011.

[3] B. Erol, E. Antúnez, and J. J. Hull, "HOTPAPER: Multimedia interaction with paper using mobile phones," in *ACM International Conference on Multimedia*, October 2008.

[4] A. F. Smeaton and A. L. Spitz, "Using character shape coding for information retrieval," in *Int. Conference on Document Analysis and Recognition*, August 1997.

[5] T. Yeh, B. White, J. S. Pedro, B. Katz, and L. S. Davis, "A case for query by image and text content: Searching computer help using screenshots and keywords," in *International Conference on World Wide Web*, April 2011.

[6] G. Schroth, S. Hilsenbeck, R. Huitl, F. Schweiger, and E. Steinbach, "Exploiting text-related features for content-based image retrieval," in *International Symposium on Multimedia*, December 2011.

[7] S. Karaoglu, J. C. van Gemert, and T. Gevers, "Object reading: Text recognition for object recognition," in *European Conference on Computer Vision Workshops and Demonstrations*, October 2012.

[8] H. Chen, S. S. Tsai, G. Schroth, D. M. Chen, R. Grzeszczuk, and B. Girod, "Robust text detection in natural images with edge-enhanced maximally stable extremal regions," in *IEEE International Conference on Image Processing*, September 2011.

[9] S. S. Tsai, D. Chen, V. Chandrasekhar, G. Takacs, N. Cheung, R. Vedantham, R. Grzeszczuk, and B. Girod, "Mobile product recognition," in *ACM International Conference on Multimedia*, October 2010.

[10] P. Erdős and A. Rényi, "On random graphs i." *Publicationes Mathematicae Debrecen*, vol. 6, 1959.

[11] V. Chandrasekhar, G. Takacs, D. M. Chen, S. S. Tsai, Y. Reznik, R. Grzeszczuk, and B. Girod, "Compressed histogram of gradients: A low-bitrate descriptor," *International Journal of Computer Vision*, 2012.