Visual Text Features for Image Matching

Sam S. Tsai[1], Huizhong Chen[1], David Chen[1], Vasu Parameswaran[2], Radek Grzeszczuk[2], Bernd Girod[1]
[1] *Department of Electrical Engineering, Stanford University, Stanford, CA, U.S.A.*
[2] *Nokia Research Center, Sunnyvale, CA, U.S.A.*
[1] {*sstsai,hchen2,dchen,bgirod*}*@stanford.edu,* [2] {*vasu.parameswaran,radek.grzeszczuk*}*@nokia.com*

***Abstract*—We present a new class of visual text features that are based on text in cameraphone images. A robust text detection algorithm locates individual text lines and feeds them to a recognition engine. From the recognized characters, we generate the visual text features in a way that resembles image features. We calculate their location, scale, orientation, and a descriptor that describes the character and word information. We apply visual text features to image matching. To disambiguate false matches, we developed a word-distance matching method. Our experiments with image that contain text show that the new visual text feature based image matching pipeline performs on par or better than a conventional image feature based pipeline while requiring less than 10 bits per feature. This is $4.5\times$ smaller than state-of-the-art visual feature descriptors.**

I. Introduction

The cameraphone has evolved substantially over the recent years, being equipped with higher resolution cameras and more processing power. These hardware improvements make possible many applications that employ image processing and computer vision techniques, such as location recognition [1], product recognition [2], or document search [3]. Most of these applications rely on compact forms of local image features, such as SIFT [4], or CHoG [5], to reduce the data sent over the network. Demonstrations of these technologies compelled MPEG to initiate the Compact Descriptor for Visual Search (CDVS) standardization effort [6]. All of these advances are based on descriptors that summarize the visual appearance of image patches. When text appears in an image, it is simply treated like any other visual feature without exploiting its special properties. We may think of local image feature algorithms, such as SIFT or CHoG, as illiterate.

Researchers have attempted to exploit the special properites of visual text in image search applications. We use the term "visual text" to refer to text pictured in an image. For example, Yeh and Katz [7] show that by recognizing text and extracting image features in onscreen captured images, the accuracy of finding help documents can be improved. Tsai et al. [8] successfully apply a hybrid approach that utilizes both image features and recognized text on book spine images for book spine recognition on mobile phones. Tsai et al. [3] use recognized text to perform web-scaled document search while using image features for verification.

In this work, we propose a new class of features based on visual text called Visual Text Features (VTFs). Visual text is located within the image using robust text detection algorithms such as [9], [10]. Then, an Optical Character Recognition (OCR) engine is used to recognize the characters and their locations. From the extracted visual text information, we generate VTFs in a way that resembles image features. Thus, we can use VTFs analogously to, and in combination with, image features to perform image matching.

The main benefits of VTFs are as follow. First, the recognized visual text can be readily used for text-based web search [3] or hybrid search [7], [8]. Second, VTFs takes on values of characters, potentially resulting in a much smaller representation compared to image features. This helps in reducing the data size during transmission and storage. Third, VTFs are well suited for and more efficient in matching objects with text. However, to reap these benefits, we need to be able to robustly extract the visual text. We address this by using EMSERs [10] to robustly find the text regions within the image. An open source OCR engine is used to recognize the characters in the text regions. We also need to develop an algorithm that can compress the VTFs. We study different ways of compressing the VTFs and propose a joint coding scheme which greatly reduces the feature size. Finally, we need to improve the discriminability of individual VTFs due to the small set of allowable descriptor values when performing image matching. We show that we can achieve this by using word distances when performing VTF matching. We describe the details of our findings in this paper.

The rest of the paper is organized as follows. We review prior research related to our work in Sec. II. Then, in Sec. III, we describe our VTF extraction pipeline. In particular, Sec. III-A describes our visual text extraction pipeline and Sec. III-B presents how to generate and code the VTFs. How to use the VTFs for image matching is described in Sec. IV. Finally, we present the experimental results in Sec. V.

II. Related Work

Related to features based on text are many different descriptors and features in the research literature of document image retrieval. Examples of descriptors that are based on the spatial distibrution of word locations includes LLAH [11], zigzag and spiral coding [12], and point-based signatures [13]. Binary codes that are generated from text

lines [14] are also used for matching documents on mobile devices. Descriptors that use character information without explicit character recognition include n-grams [15] and word shape coding [16]. Most of these techniqes work best on documents with dense text distributions. In contrast, we work with natural scene images where text are embedded in complex backgrounds. Closer to our work, [17] extract descriptors from text regions detected from natural scenes and use them for matching. Different from [17], we use an OCR engine to recognize the characters from the detected text regions. Using the recognized characters, we generated VTFs which are analogous to image features. We show that VTFs can be used for image matching and its performance is comparable to image descriptors while at a much lower rate. In addition, OCR engines, which have been previously considered to be costly, are more accessible now that devices have better compute power. It has also been shown to work realtime on mobile devices[1].

III. Visual Text Features

In Fig. 1, we show the block diagram of the VTF processing pipeline. In the extraction stage, we detect text in the image and perform recognition on the detected text patches. In the coding stage, we generate the VTFs from the recognized characters and compress the data. We describe these two stages in more detail in the following sections.

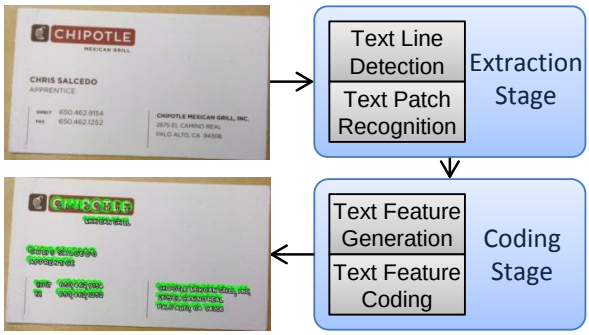


Figure 1. The visual text feature extraction and coding pipeline.

A. Visual Text Extraction

In the first step of the extraction, we use an edge-enhanced MSER-based text detection algorithm [10] to find text regions in a given image, see Fig. 2 (a). We fix the input size of the given image to have a maximum dimension of 640 pixels to match the size of typical visual search input images and the resolution of typical video capturing devices. However, we rescale the image to a size of t_d pixels using bicubic interpolation to a larger size before extracting MSERs to reduce sampling effects.

After text lines are detected from the image, we extract image patches of these text lines. We scale the patch to a fixed size t_p, and use an OCR engine to recognize the text[2], see Fig. 2 (b). Since we detect text lines of multiple orientations, we typically do not know which direction is upwards. Thus, we perform recognition for both the original direction and a 180 degree rotated version to deal with upside-down text. The results with the highest confidence score given by the OCR engine is selected.

In scenes where complex backgrounds appear in the image, the OCR engine typically fails to find the correct threshold for binarizing the text. Thus, to improve the recognition performance, we use results of the text detection MSER masks to binarize the text patches. Along the borders of the text detection mask, we calculate the average intensity of a region that is 1/5 the width of the character stroke width. This threshold is then used to binarize the text patch.

Finally, after the characters and their bounding boxes have been recognized in the text patch, we project the bounding boxes back to the original image, as shown in Fig. 2 (c). We describe how we generate the VTFs using the characters and their bounding boxes in the next section.

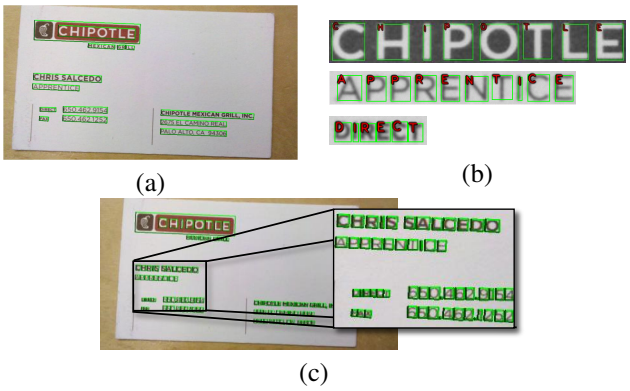


Figure 2. Visual text extraction example: (a) Text lines are detected, (b) Text patches are extracted and recognized, and (c) bounding boxes of the text are reprojected back to original image.

B. Visual Text Feature Generation and Coding

From the recognized characters and their bounding boxes, we generate VTFs that's analogous to image features. The location of a VTF is calculated as the center of the recognized character's bounding box. The scale is the radius of the minimal bounding circle of the bounding box. The orientation is the direction of the text line the recognized character came from. Note that scale and orientation are typically not needed for image matching. The descriptor of the VTF comprises the recognized character, the word from which the recognized character came from, and the position of the character in the word. Word information is very useful in the image matching stage which we will discuss in Sec. IV. Additionally, we use only recognized characters that have a OCR confidence score of t_s.

[1]Word Lens, http://questvisual.com

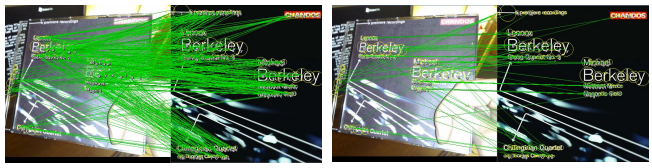
[2]We use the Tesseract v3.01 (http://code.google.com/p/tesseract-ocr/) for our experiments and limit the recognized character set to {'a-z','A-Z','0-9',',','.','-'}.

Figure 3. Examples of visual text features. Each visual text feature is represented by a circle with a radius corresponding to the scale.

To transmit these features, or store them in a database on a mobile device, we wish to compress them. We use a joint coding scheme to code the visual text features in a image. We create a word table containing all recognized words. Then, we use an index to the table and a position to the character within the word to code each VTF. We provide further details of the compression scheme in Sec. V-C.

IV. Image Matching with Visual Text Features

When using image features for image matching, features from two images are paired based on descriptor similarity. A ratio test typically is used to rule out ambiguous matches [4]. Then, RANSAC is used to estimate an affine model from the locations of the matching features [2]. Since VTFs are akin to image features, the conventional image matching pipeline can be adapted easily. One challenge is that the recognized characters have only a very limited set of values, and hence multiple matches are common (Fig. 4 (a)). The ratio test cannot be used because the distance of the recognized characters are either 1 for being the same or 0 for being different. The large percentage of invalid feature correspondences can easily confuse RANSAC. Thus, we propose to use the surrounding word as part of the VTF to disambiguate character-based feature matches.



(a) (b)

Figure 4. (a) Pairwise feature matches using the recognized character. (b) Pairwise feature matches using word distances.

To determine the distance between two VTFs, we calculate a word-distance which we define as the sum of the editing distance between the strings preceding the examined character and between the strings following the examined character. For example, suppose we are matching the two '**t**'s from 'arithme**t**ic' and 'mathema**t**ic'. The word-distance is 6, which is the sum of the editing distance of ('arithme', 'mathema') and ('ic','ic'). Furthermore, we require a feature match to have a distance that is smaller than half of either word's length. By using the word-distance for feature pairing, the VTFs becomes much more discriminative and avoid irrelevant feature matches, as shown in Fig. 4 (b).

V. Experimental Results

In this section, we first present results on training the visual text extraction pipeline. Then, we show the image matching performance using VTFs. We evaluated different configurations and also compare with image feature based matching approach. Finally, we show results on compressing the VTFs.

A. Training the Visual Text Extraction Pipeline

We train the visual text extraction pipeline using an image set with annotations of text characters. Well known image databases satisfying such criteria includes [18] and [19]. However, their text content for each image is scarce. Thus, we create a new image database of books, magazines, and business cards taken using a cameraphone. Text in the images is annotated semi-automatically. We use the text extraction pipeline to find possible text and manually correct the results. We generate a image set with 20 images and 6500 annotated characters[3]. Two examples of the training images are shown in Fig. 5.

Figure 5. Two example images from our training set and the close up view with annotations.

We perform text extraction on the annotated image set and calculate the correct matches. A match is correct when the detected character is the same as an annotated character and the distance of the two character centers is within a half of the detected character's scale (as defined in Sec. III-B). Only single matches for each annotated character are allowed. We calculate the precision p by dividing the number of correct matches by the number of detected characters, and recall r by dividing the number of correct matches by the number of annotated characters. Similar to [18], we calculate a single score $f = 1/(\alpha/p + (1-\alpha)/r)$ as the final reference. We use a $\alpha = 0.5$ for equal weighting between p and r. We aim to maximize f in the training process.

[3]http://msw3.stanford.edu/~sstsai/VisualTextDatabase

We optimize the visual text extraction pipeline by sweeping the different parameters and finding the best performance in the following order: (1) text recognition patch size t_p, (2) text detection image size t_d, and (3) text detection MSER parameters (we use MSER implementation based on VLFeat[4]). In each step we optimize one dimension, and iteratively go through the each dimension until the score converges. After the fourth step, the recognition converges to $f = 0.88$ with the following settings: $t_p = 70, t_d = 1536$. Comparing to before optimization, the visual text extraction pipeline has a recognition score of only $f = 0.62$.

B. Image Matching using Visual Text Features

To evaluate the performance of the image matching algorithm we have developed, we use the pairwise matching evaluation framework and the image dataset provided by the MPEG CDVS standardization group [20]. We test the image matching pipeline using pairs of matching images and pairs of non-matching images from the category of business cards. We show two examples of matching image pairs and their VTF correspondences after RANSAC in Fig. 6.

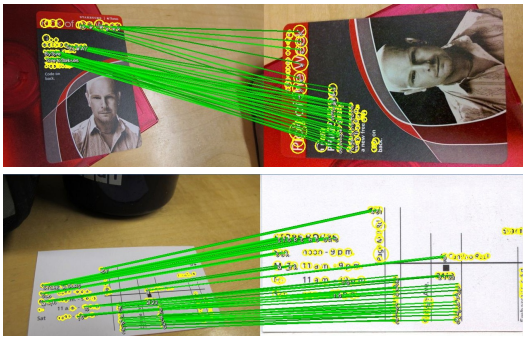


Figure 6. Two matching image pairs from the CDVS dataset and their VTF correspondences after RANSAC.

To evaluate the performance, we calculate the True Positive Rate (TPR) and False Positive Rate (FPR) from the image matching results. By varying the feature matches decision threshold, we can plot the Receiver Operating Characteristic (ROC) curve of the pipeline. To see how VTFs perform on images with text, we limit the test set to have at least 60 VTFs detected. On average, 163 VTFs are extracted from each image and used to perform matching. To compare the performance with the image feature based approach, we extract CHoG low-bitrate image features [5] with a total of 163 image features and perform the same test.

We compare the performance of the VTF image matching pipeline to the CHoG image matching pipeline in Fig. 7. The VTF image matching pipeline with word-distances enhancements has approximately the same performance as the CHoG image matching pipeline that uses the same number of features when the FPR is < 0.01. Examination of the false negatives revealed that the images includes background text and the object of interest did not have enough visual text features. At the same performance, the size of the VTFs are much smaller than the CHoG, which we will show in the next section.

Fig. 7 also shows the performance of the VTF image matching pipeline without using the word-distances. We can see that the ambiguous character matches result in a much higher FPR. Using the word-distances, we achieve a TPR of ~ 0.9 while still maintaining an FPR at 0.01. We additionally include the performance of the CHoG image matching pipeline with using only 56 CHoG features. This number is chosen so that the query size of the VTF matches the query size of the CHoG. With the same rate constraint, the VTF image matching pipeline outperforms the CHoG image matching pipeline by more than 0.3 in TPR.

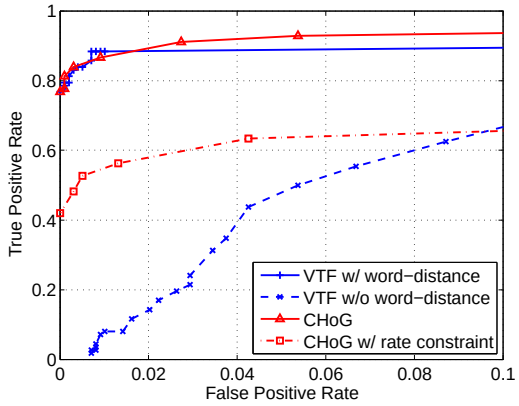


Figure 7. The ROC curve comparison for the VTF image matching pipeline and the CHoG image matching pipeline.

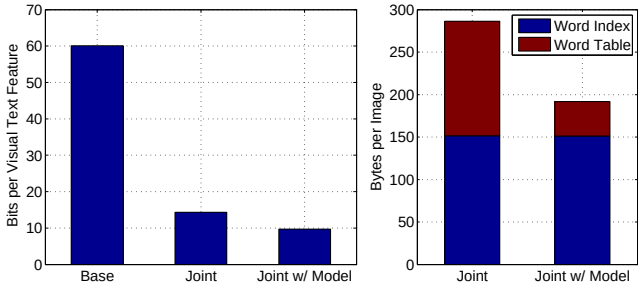
C. Coding of Visual Text Features

We evaluate the coding performance of the joint coding scheme for VTFs using the same set of images we used for pairwise image matching experiments.

In Fig. 8 (a), we show the average size of the descriptor of the VTFs using three different coding schemes. The base method, which encodes the VTFs independently, uses a fixed rate coding scheme. Since word information is included in every descriptor, a word of length n is redundantly encoded for n times, resulting in a high rate. Using the joint coding scheme, we avoid redundantly coding the words and reduce the bitrate by more than $5\times$. Further rate reduction can be achieved using a character frequency model [21] to entropy code the word table. We show in Fig. 8 (b) that using the model reduces the word table size by 60%. Using the joint coding scheme with the model enhancement, a VTF is only ~ 9.7 bits per feature. Compared to the CHoG low-bitrate descriptor with a rate of 44 bits per descriptor, VTFs are $> 4.5\times$ smaller.

Additionally, we use location histogram coding [22] to code the locations and achieve a rate of ~ 8.1 bits per location. With an average of 163 features, the total query

[4]http://www.vlfeat.org



(a) (b)

Figure 8. (a) Bitrate comparison of three schemes. (b) Bitrate constitution of the two joint coding schemes.

size per image is ~ 2900 bits, or ~ 360 bytes. For CHoG features, the bits per feature is ~ 52 bits. Thus, given the same bitrate constraint, only 56 CHoG features would be allowed.

VI. Conclusions

We present a new type of visual text features based on recognized text from an image. To the best of our knowledge, this is a first attempt in incorporating text in a way that resembles image features. To have useful visual text features, a reliable visual text extraction pipeline needs to be built. By using the edge-enhanced MSER text detection and an OCR engine, we are able to get good visual text extraction performance on our image set. Additionally, cues from the OCR engine, such as the confidence score, is used to select useful features.

We apply the visual text features to the image matching problem. We found that the discriminability of just the characters were not enough to disambiguate false feature matches. Hence, we developed a word-distance based matching approach which uses word information to determine if two visual text features are similar. The method significantly reduces the number of ambiguous feature matches and greatly improves the image matching performance. We compare the visual text feature image matching pipeline to a low-bitrate image descriptor image matching pipeline and found that our approach is on par or even better while having a smaller data size. Visual text features can be compressed very efficiently using a joint coding scheme. It requires only $\sim$9.7 bits per descriptor which is $> 4.5\times$ smaller than the state-of-the-art low-bitrate descriptors.

References

[1] D. M. Chen, G. Baatz, K. Köser, S. S. Tsai, R. Vedantham, T. Pylvänäinen, K. Roimela, X. Chen, J. Bach, M. Pollefeys, B. Girod, and R. Grzeszczuk, "City-scale landmark identification on mobile devices," in *Conference on Computer Vision and Pattern Recognition*, 2011.

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

[3] S. S. Tsai, H. Chen, D. M. Chen, R. Vedantham, R. Grzeszczuk, and B. Girod, "Mobile visual search using image and text features," in *Asilomar Conference on Signals, Systems, Computers*, 2011.

[4] D. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, 2004.

[5] V. Chandrasekhar, G. Takacs, D. Chen, S. S. Tsai, R. Grzeszczuk, and B. Girod, "CHoG: Compressed Histogram of Gradients," in *In Proceedings of Conference on Computer Vision and Pattern Recognition*, 2009.

[6] "Call for proposals for compact descriptors for visual search," in *ISO/IEC/JTC1/SC29/WG11/N12201*, 2011.

[7] T. Yeh and B. Katz, "Searching documentation using text, ocr, and image," in *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, New York, NY, USA, 2009.

[8] S. Tsai, D. Chen, H. Chen, C.-H. Hsu, K.-H. Kim, J. P. Singh, and B. Girod, "Combining image and text features: a hybrid approach to mobile book spine recognition," in *ACM International Conference on Multimedia*, 2011.

[9] B. Epshtein, E. Ofek, and Y. Wexler, "Detecting text in natural scenes with stroke width transform." in *Conference on Computer Vision and Pattern Recognition*, 2010.

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

[11] T. Nakai, K. Kise, and M. Iwamura, "Use of affine invariants in locally likely arrangement hashing for camera-based document image retrieval," in *Document Analysis Systems*, 2006.

[12] J. Moraleda and J. J. Hull, "Toward massive scalability in image matching," in *International Conference on Pattern Recognition*, 2010.

[13] N. Spasojevic, G. Poncin, and D. S. Bloomberg, "Discrete point based signatures and applications to document matching," in *International Conference on Image Analysis and Processing*, 2011.

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

[15] J. Hull, B. Erol, J. Graham, Q. Ke, H. Kishi, J. Moraleda, and D. Van Olst, "Paper-based augmented reality," in *International Conference on Artificial Reality and Telexistence*, November 2007.

[16] S. J. Lu, L. L. Li, and C. L. Tan, "Document image retrieval through word shape coding," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2008.

[17] G. Schroth, S. Hilsenbeck, R. Huitl, F. Schweiger, and E. Steinbach, "Exploiting text-related features for content-based image retrieval," in *IEEE International Symposium on Multimedia (ISM)*, Dana Point, CA, USA, Dec 2011.

[18] S. Lucas, A. Panaretos, L. Sosa, A. Tang, S. Wong, and R. Young, "ICDAR 2003 robust reading competitions," in *International Conference on Document Analysis and Recognition*, vol. 2, 2003.

[19] K. Wang and S. Belongie, "Word spotting in the wild," in *European Conference on Computer Vision*, 2010.

[20] "Evaluation framework for compact descriptors for visual search," in *ISO/IEC/JTC1/SC29/WG11/N12202*, 2011.

[21] R. E. Lewand, *Cryptological Mathematics*. Mathematical Association of America, 2000.

[22] S. S. Tsai, D. M. Chen, G. Takacs, V. Chandrasekhar, R. Vedantham, R. Grzeszczuk, and B. Girod, "Location coding for mobile image retrieval," in *Proc. 5th International Mobile Multimedia Communications Conference*, 2009.