Improved Coding for Image Feature Location Information

Sam S. Tsai*, David Chen*, Gabriel Takacs*, Vijay Chandrasekhar*
Mina Makar*, Radek Grzeszczuk†, and Bernd Girod*

*Department of Electrical Engineering, Stanford University, Stanford, CA 94305, U.S.A.
†Nokia Research Center, Nokia, Sunnyvale, CA 94086, U.S.A

ABSTRACT

In mobile visual search applications, an image-based query is typically sent from a mobile client to the server. Because of the bit-rate limitations, the query should be as small as possible. When performing image-based retrieval with local features, there are two types of information: the descriptors of the image features and the locations of the image features within the image. Location information can be used to check geometric consistency of the set of features and thus improve the retrieval performance. To compress the location information, location histogram coding is an effective solution. We present a location histogram coder that reduces the bitrate by $2.8\times$ when compared to a fixed-rate scheme and $12.5\times$ when compared to a floating point representation of the locations. A drawback is the large context table which can be difficult to store in the coder and requires large training data. We propose a new sum-based context for coding the location histogram map. We show that it can reduce the context up to $200\times$ while being able to perform just as well as or better than previously proposed location histogram coders.

Keywords: content-based image retrieval, mobile visual search, mobile augmented reality, location histogram coding

1. INTRODUCTION

As handheld mobile devices are equipped with greater processing power and higher resolution cameras, a wide range of applications based on mobile visual search[1] and mobile augmented reality[2,3] have emerged. Applications for mobile product recognition enable users to perform comparison shopping, listen to sample music, and read reviews about the product through their mobile devices.[3,4] Image-based localization gives the user the ability to identify their locations where GPS is not available, which can be used to find, e.g., local store information.[2,5] Underlying these applications are image retrieval systems that are based on robust local image features, such as SIFT,[6] SURF,[7] and CHOG.[8] From the query image, these systems extract robust local image features. Then, the feature set is compared to a database of image feature sets to retrieve a match. Depending on the size of the database, the system may perform the matching directly on the mobile device or in the cloud. For applications where the database is large, the query data is extracted on the device and sent over the network to a server to match against the database. When bandwidth is limited, compression of the query data is essential.

Query data of robust local image features comprises of mainly two types of information: the descriptor, which is a summary of the image patch around the keypoint, and the location of the descriptor within the image. Many researchers have looked into the problem of compressing the query data. To reduce the size of the descriptor, Ke and Sukthankar[9] used Principle Component Analsyis (PCA) to reduce the dimensionality of local descriptors. Hua et al.[10] used Linear Discriminant Analysis (LDA). Later, Chandrasekhar et al. designed low bitrate descriptors[8] from the ground up for better performance. To reduce the size of the location data, Tsai et al.[11] developed a histogram coding method for the locations.

The Compact Descriptors for Visual Search (CDVS) group within MPEG[12] aims to standardize technologies that enable efficient and interoperable design between visual search applications.[13] To attain this goal, an evaluation framework for evaluating the visual search technology has been created.[14] The evaluation framework comprises several test image data sets, along with sets of experiments for evaluating image matching and image retrieval performance under various rate constraints. The rate constraints were identified from practical application scenarios. Under the stringent rate constraints, new challenges for the existing technology emerged. In particular, the location histogram coding method needs to deal with a various size of feature set and varying bitrate.

Applications of Digital Image Processing XXXV, edited by Andrew G. Tescher, Proc. of SPIE
Vol. 8499, 84991E · © 2012 SPIE · CCC code: 0277-786/12/$18 · doi: 10.1117/12.935619

Within the CDVS framework we developed an improved method of coding the locations. The method is built upon our previous work of location histogram coding,[11] which uses location histograms to represent the location information. To compress the location histogram, a context-based arithmetic coder is used with a context that is based on the histogram map patterns. A disadvantage of the pattern-based context is that it grows exponentially as the number of reference points increases. To overcome this limitation, we propose to use a sum-based context for encoding the location histogram map. Using the sum-based context, the size of the context grows linearly as the reference points increases. In our experiments, we find that the sum-based context is suitable for a wider range of feature set conditions, and it requires far lesser training data. However, generating the sum-based context from a large set of reference points is time consuming. Thus, we use integral images to speed up the sum-based context generation. Finally, we show that the context size of the sum-based context can be further reduced by sum quantization.

The rest of the paper is organized as follows. In Section 2, we review location histogram coding techniques. In Section 3, we describe the proposed sum-based context, how to calculate the sum-based context in fixed time, and how to reduce context size by sum quantization. We present the evaluation of the proposed method and present the results using the CDVS evaluation framework[14] in Section 4.

2. LOCATION HISTOGRAM CODING REVIEW

Shown in Figure 1 are examples of the SIFT[6] feature locations detected in images. SIFT features are detected using a Difference of Gaussian (DoG)-based blob detector and can be refined to sub-pixel accuracy. Thus, two floating point variables are typically used to represent a feature's location within the image.

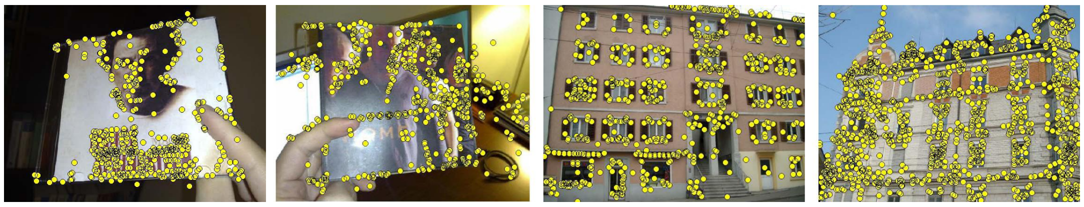


Figure 1. Features in the CDVS dataset are randomly scattered but still cluster near structures.

Location histogram coding exploits the fact that the order in which the features are transmitted in a bitstream is not relevant. Thus, for a set of N features, there are potentially $N!$ equivalent orderings. Location histogram coding orders the features by their locations. We can potentially save $log_2 N!$ by sorting the features in a prescribed manner. This same property has been exploited earlier in Chen's Tree Histogram Coding[15] and later in Chandrasekhar's Digital Search Trees.[16] Additional gains can be achieved if the locations of features in an image are not independent. Features tend to cluster around interesting structures in the image, as seen in Figure 1. To exploit this property, a context-based arithmetic coder or a joint block coder can be used. In the following sections, we describe how to generate and represent the location histogram and provide a brief review of two previous location coding schemes.

2.1 Location Histogram Generation

Given a query image, we first extract the features from the image (Figure 2 (a)). Then, a spatial grid is overlaid (Figure 2 (b)). Square cells are typically used. We refer to each grid as a histogram block and the width of the grid as the histogram block size. Finally, the features within each histogram block are counted, forming the location histogram. We represent the location histogram by using location histogram map and location histogram counts. The location histogram map is a binary map that specifies whether the location histogram block is non-empty or empty, as shown in Figure 2 (c). The location histogram counts record the number of features for the non-empty blocks.

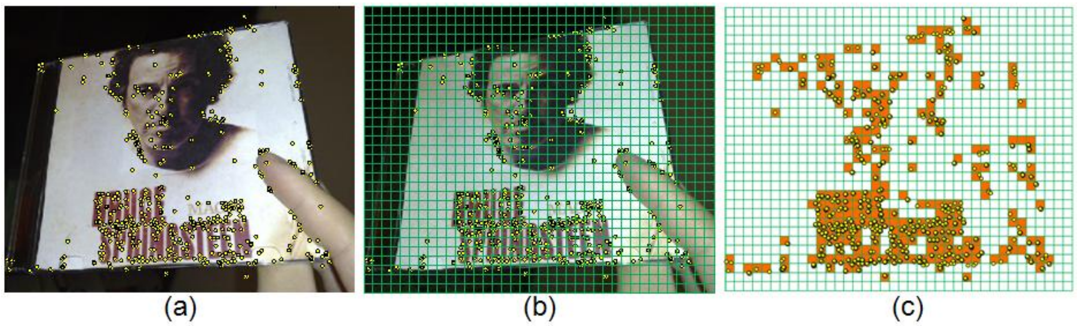


Figure 2. (a) An image with a set of detected interest points. (b) An overlay of a spatial grid over the original image and the set of features. (c) The location histogram map with non-empty blocks colored as orange with the features overlaid on top.

2.2 Pattern-based Location Histogram Coding

The pattern-based location histogram coding was described in our previous work.[11] We code the location histogram map in a way that resembles encoding a binary image. We start from the upper left corner of the histogram map and proceed in raster scan order to encode the information of each histogram block while using previously coded histogram blocks as reference points. We code each block using a context-based arithmetic coder using the context generated by the reference points. Shown in Figure 3 is a histogram map, M. The grayed out area indicates uncoded blocks. When encoding the current block X, we use information of 14 neighboring blocks, as shown in the close-up view, to generate the context. Let (i_m, j_m) be the location of the m-th neighboring block's index in the histogram map. Then, the pattern-based context, p_c, for coding X is generated as follows:

$$p_c = \sum_{m=1}^{14} 2^{m-1} \times M(i_m, j_m) \tag{1}$$

Since each $M(i_m, j_m)$ can take on a value of 0 or 1, $p_c \in [0, 16363]$. The size of the context is 16K, thus, a training set of several hundred images is typically needed.

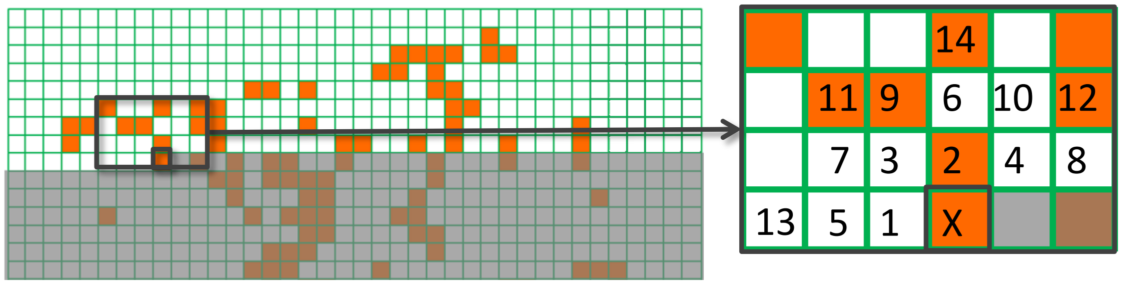


Figure 3. The location histogram map is encoded in a raster scan order. The darker blocks indicate uncoded blocks while the orange blocks indicate non-empty blocks. For each block, nearby previously encoded blocks are used as context as shown in the right.

The location histogram counts for the non-empty blocks are encoded as individual symbols. We train a model, with 1 to 64 feature counts, using the same set of images we use to train the histogram map.

2.3 Location Histogram Coding in CDVS Test Model

The CDVS Test Model[17] includes a location information coder that is based on location histogram coding. When encoding the location histogram map, bit codes are used to describe whether a column or a row of the map contains all empty blocks or at least one non-empty block. A raster scan order is used to code the information for the blocks with both row and column marked as non-empty. The information of 8 consecutive blocks is jointly coded using an arithmetic coder. The location histogram counts for each non-empty block are coded in an iterative fashion. Bit codes are used to indicate which blocks have more than 1 feature. Then, for the blocks that have more than 1 feature, a second round is used to indicate which blocks have more than 2. The process is repeated until the largest histogram count value is specified.

3. SUM-BASED LOCATION HISTOGRAM CODING

The proposed sum-based location histogram coder follows the exact architecture as described in Section 2.2. However, instead of using the pattern-based context for encoding the histogram map, we use a new sum-based context. The pattern-based context size grows exponentially as the number of reference points increases. Thus, the number of reference points is typically restricted to a small number. The sum-based context size grows linearly as the number of reference points increases. Thus, we can expand the number of reference points more easily. Furthermore, to generate the sum-based context from a large set of reference points, we introduce a generation method based on integral images. Regardless of the number of reference points, the method generates the sum-based context in a fixed time. Finally, we also introduce a quantization method on the sum-based context that further reduces the context size. We describe these in more detail in the following sections.

3.1 Sum-based Context

When encoding a location histogram map M, we start from the upper left corner and proceed in raster scan order. For each block, we use the number of non-empty blocks in previously encoded neighboring blocks, with a Manhattan distance lesser than a value, as context. Let $N_{i,j}$ be the indices of the set of neighboring blocks which has been previously encoded, and is within a range r_s of the block index (i,j), i.e., for all $(m,n) \in N_{i,j}$, $max(|m-i|,|n-j|) \leq r_s$. r_s is decided empirically. The sum-based context, s_c, for block (i,j) is the number of non-empty blocks in $N_{i,j}$:

$$s_c(i,j) = \sum_{(m,n)\in N_{i,j}} M(m,n). \tag{2}$$

Shown in Figure 4 are two examples of $N_{i,j}$ with different r_s for a block (i,j), which is marked as 'X'. The red dashed line encloses previously encoded blocks that is within a r_s of 2 while the purple dashed line encloses blocks that is with a r_s of 4. The sum-based context, for the two different r_s is 4 and 11, respectively.

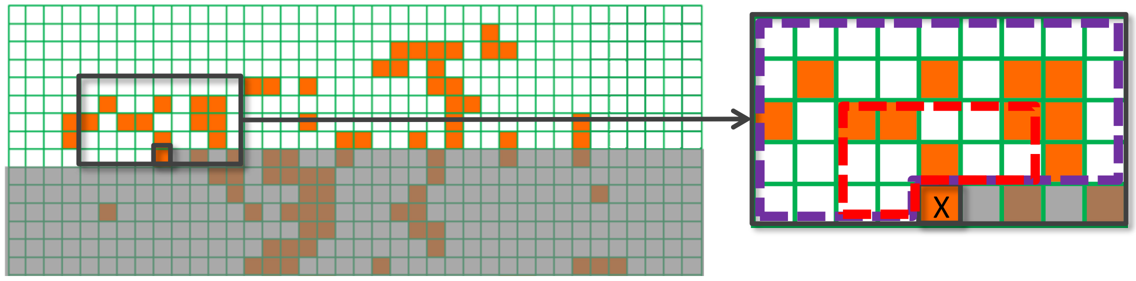


Figure 4. For each block, nearby previously encoded blocks are used to generate the context. In the close up view, the red and purple dashed lines encloses the neighboring blocks used to generate the sum-based context for a range of 2 and 4, respectively.

For a range of r_s, the size of $N_{i,j}$ is $(2r_s^2 + 2r_s)$. Thus, the size of the sum-based context is $(2r_s^2 + 2r_s + 1)$. It grows linearly as the number of reference points increases. Compared to the pattern-based context, the context size is much smaller. Thus, it is easier to train and has a smaller storage requirement.

3.2 Fast Sum Calculation

Sum-based context allows the coder to consider a large amount of neighborhood blocks when generating the context. However, generating the sum can still be time consuming. To overcome this issue, we propose calculating the sum using integral images.[18] When a location histogram map is produced, we generate its integral image:

$$I_M(i,j) = M(i,j) + I_M(i-1,j) + I_M(i,j-1) - I_M(i-1,j-1). \tag{3}$$

To calculate the sum of a rectangular area from the integral image, only four points need to be read. The region containing $N_{i,j}$ can be represented using two rectangles. Thus, the sum-based context of index (i, j) can be calculated using integral images as follows:

$$\begin{aligned} s_c(i,j) =& I(i+r_s, j-1) + I(i-r_s-1, j-r_s-1) - I(i+r_s, j-r_s-1) - I(i-r_s-1, j-1) \\ &+ I(i-1, j) + I(i-r_s-1, j-1) - I(i-r_s-1, j) - I(i-1, j-1) \end{aligned} \tag{4}$$

Noticing two terms cancel out in the equation, the sum-based context is calculated using six values from the integral image. Thus, for any range r_s setting, the complexity of calculating the sum is fixed.

3.3 Sum Quantization

To further reduce the context size, we propose to quantize the sum. This is useful for location histograms with small histogram block sizes because they typically require a large number of reference points for better performance. To quantize the sum, we use a quantization step size that is logarithmic. We find that it performs better than using a uniform step size. To generate N quantization levels, we generate a $(N-1)$-long logarithmic sequence from 0.001 to 1. We multiply the numbers by the maximum sum in the sum-based context and round them to the nearest equal or smaller integer. With an additional value of 0, they form the quantized set with N values. When quantizing a sum, we pick the closest number from the quantized set. If two numbers have the same distance, the smaller number is selected.

4. EXPERIMENTAL RESULTS

In this section, we first present the experimental results of the proposed algorithm. Then, we present the evaluation results on the Core Experiment[19] for the CDVS evaluation framework which we will describe in Section 4.2.

4.1 Sum-based Coding of Locations

We present experimental results of different configurations of the sum-based location coder. We also compare it with the pattern-based location coder. We use the Caltech Buildings Dataset* and the Inria Holidays Dataset† for training. The two datasets combined have a total of 1741 images. We use images of Category 1 of Mixed Text and Graphics in the CDVS evaluation framework[14] for testing. The total number of testing images is 2500. From the images, we extract DoG features points using the CHoG implementation‡. We extract two sets; one has an average of 100 features while the other has an average of 500 features.

Range Optimization for Sum-based Context. One advantage of the sum-based context coder is that the context size grows linearly as the number of reference points increases. Furthermore, using integral images for calculating the sum context, only a fixed number of operations are needed. To see how the number of reference points affects the performance, we evaluate the coding performance of the sum-based location coder using different range settings with different histogram block sizes.

Shown in Figure 5 is the coding performance of the sum-based location coder using various range settings. We show the coding performance for a histogram block size of 4, 6, and 8 pixels. We can see that for a histogram block size of 4 pixels, the best performing range is 8 and 5 blocks for the two different feature sets. At 100 features, the distribution of features in the image is much sparser; hence a larger range is needed. At 500 features, a range of 5 blocks is enough to exploit the spatial correlation. As the histogram block size increases, the optimal range decreases. We find the product of the two values, which is the range of the reference region containing the neighboring blocks in pixels, is approximately the same for the same set of feature count setting. For the 100 count feature set, a reference region that covers a range of ~ 32 pixels typically performs better. For the 500 count feature set, the best performing range is ~ 20 pixels.

Quantization of the Sum-based Context. We use sum quantization to further reduce the size of the context. We show the coding performance of the sum-based context without quantization and with quantization under

*http://vision.caltech.edu/malaa/datasets/caltech-buildings/
†http://lear.inrialpes.fr/~dejegou/data.php
‡http://www.stanford.edu/people/vijayc/chog-release.zip

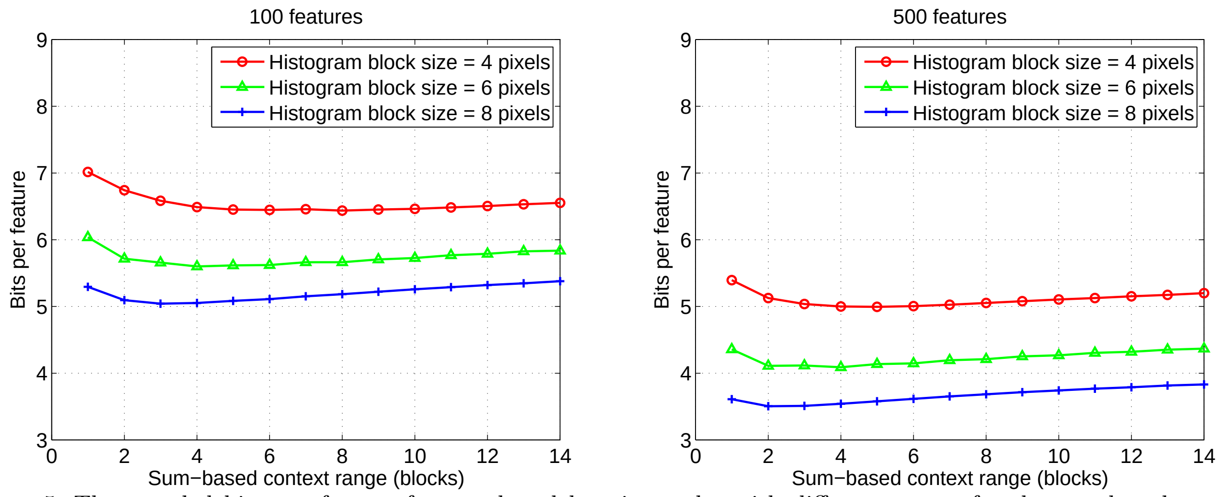


Figure 5. The encoded bits per feature for sum-based location coder with different ranges for the sum-based context location coder. The coding performances for histogram block sizes of 4, 6, 8 pixels are shown.

different histogram block sizes in Figure 6. The coding performance drops as the number of quantization levels decrease. For both the 100 count and the 500 count feature set, the same trend is observed. For the 100 count feature set, when using 5 level quantization, the bitrate difference between using a block size of 4 and using a block size of 3 is smaller than the others. This is due to the rounding used in generating the quantized levels. The effective quantization levels is 5 for block size 4 where as it is 4 for block size 5. To keep performance loss to a minimum, we use a quantization level of 20.

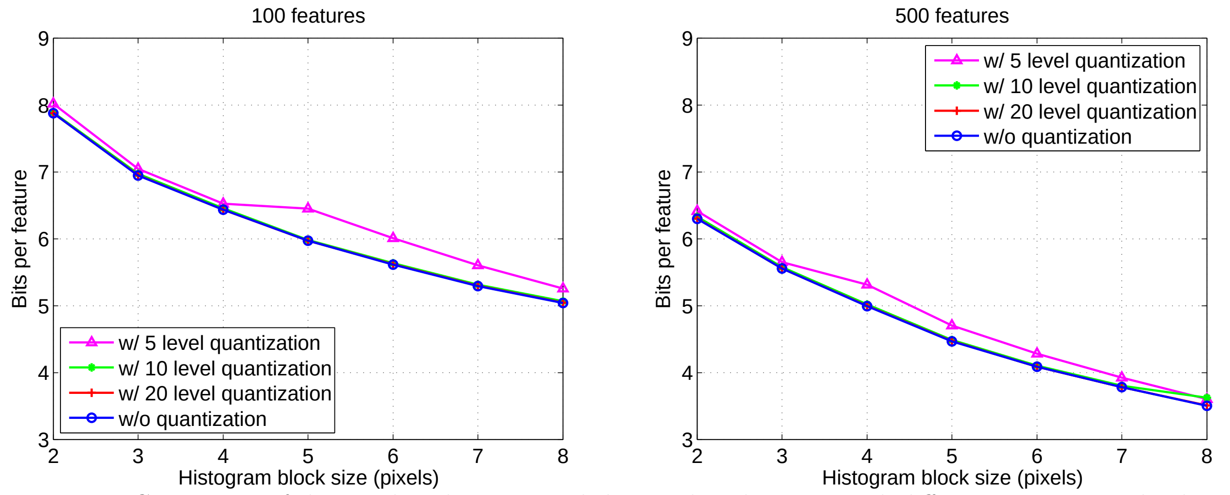
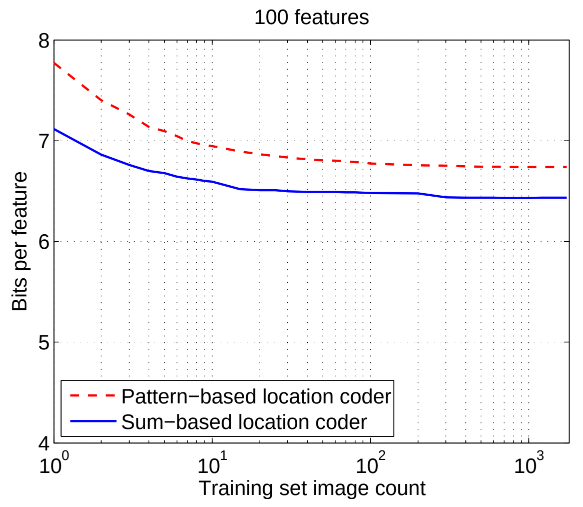
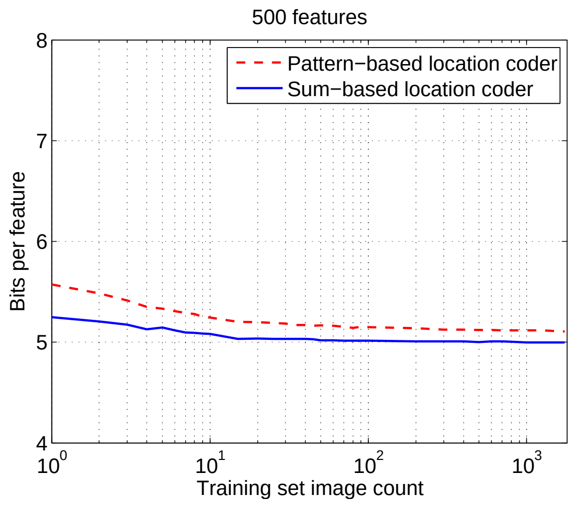


Figure 6. Comparison of the sum-based context and the sum-based context with different quantization levels.

Comparison with Pattern-based Context Location Coder. We compare the sum-based location coder with the pattern-based location coder. The sum-based location coder has the advantage of having a smaller context size given the same number of reference points. The smaller context means that it requires a smaller set of training images. We test the coding performance of the sum-based location coder and the pattern-based location coder with different training set image counts. We use a location histogram block size of 4 pixels for both schemes and use the optimal range of 8 and 5 blocks for the 100 count feature set and the 500 count feature set, respectively. We quantize the sum-based context to 20 levels resulting in a context size of 20 for the sum-based location coder. The pattern-based context is configured to use a total of 12 neighboring blocks resulting in a context size of 4096.



(a)

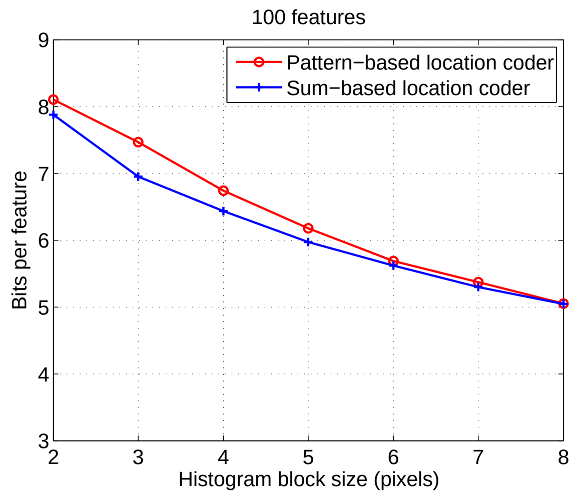


(b)

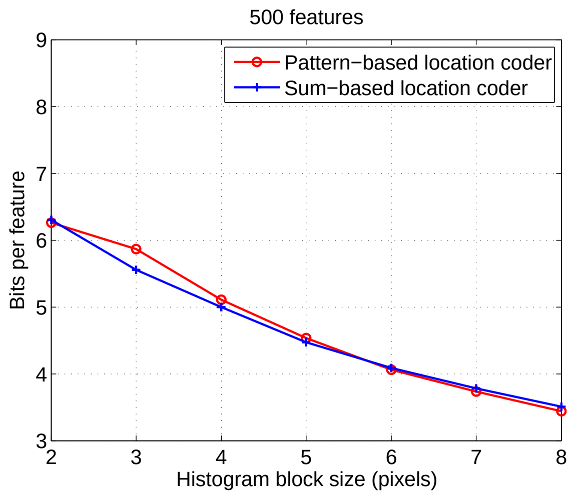
Figure 7. The encoded bits per feature for two coders under different sizes of training set images.

The coding performance of the two schemes under different training image counts are shown in Figure 7. We can see clearly that the sum-based location coder performs significantly better than the pattern-based location coder when only a small set of images is used for training. Using a single training image, the sum-based location coder has a $\sim 10\%$ bitrate reduction over the pattern-based location coder. At ten training images, the sum-based location coder is already on par or outperforms the best performance of the pattern-based location coder. With the smaller context size, the sum-based location coder typically requires a training set size that is an order smaller than that of the pattern-based location coder.

We also compare the two location coders under different histogram block sizes, as shown in Figure 8. The sum-based location coder, being able to cover a larger number of reference points, performs particularly better than the pattern-based location coder for the 100 count feature set. At a histogram block size of 3 pixels, we observe the largest bitrate improvement of $\sim 8\%$. For larger histogram block sizes, the two perform equally well. This is because the range of the sum-based context covers the same set of reference points that the pattern-based context covers. In this case, the pattern-based context has the advantage of being able to provide finer descriptions of the neighboring blocks.



(a)



(b)

Figure 8. The encoded bits per feature for the two coder under different histogram block size settings.

4.2 CDVS Evaluation

In the 99th MPEG meeting, evidence was brought to the CDVS group to show that the location coding method in the Test Model[17] can be improved.[20] A bitrate reduction of 10% was observed for multiple operating points using various location histogram block sizes. This evidence motivated the group to set up a core experiment to re-evaluate methods to code the locations.

As described in the Core Experiment description,[19] the location coding core experiment is performed using the Test Model. Parties that wish to participate and contribute to the experiment should integrate their solution into the Test Model and evaluate the performance for the following configurations. At six operating points, 512, 1K, 2K, 4K, 8K, and 16K bytes, defined in the evaluation framework,[14] the matching performance of keypoint location rates at 4, 6, and 8 bits per feature will be evaluated. The bitrate need not be exact, but should be as close as possible. Image matching experiments will be tested on matching and non-matching image pairs for all categories set forth in the evaluation framework.[14] The results of the True Positive Rate (TPR) and the False Positive Rate (FPR) are to be reported. For Category 1, in particular, the localization score is also measured and reported. The localization score is the Jaccard index of the ground truth bounding box and the detected bounding box from the matching results.

We submitted our sum-based location coder as the one of the input contributions to the core experiment. To achieve the bitrate of 4, 6, and 8 bits per feature, different configurations of the coder were tested and the closest matching configurations were selected. We use location histogram block sizes ranging from 2 to 17 pixels. The smaller sized is used for the higher rates and the larger sized is used for the lower rates. The number of reference points for generating the sum-based context varies as the histogram block size changes. The distribution of the features produced by the Test Model is different from the DoG interesting points. We find that the best performing configurations uses a reference region that covers ~ 45 pixels. We show our reported TPR results in Figure 9. We see that the TPR varies for different categories. However, for all categories a better performance is observed for a higher rate operating point. As the rate of the keypoint location decreases, the TPR drops slightly. The drop is more pronounced in the 512 byte operating point in Category 1.

In Figure 10, we show the localization score of the Category 1 matching experiment for the sum-based location coder and the Test Model location coder. We use the default configuration of the Test Model location coder to generate a single rate point. Unlike the TPR results, the localization score drops more significantly as the rate of the keypoint location decreases. The localization score difference between the highest rate and the lowest rate is largest at the 512 byte operating point. At the 512 byte operating point, the sum-based location coder achieves a bitrate reduction of $\sim 10\%$ over the Test Model location coder. The bitrate reduction increases for higher rate operating points. At 16K operating point, the bitrate reduction is $\sim 20\%$.

5. CONCLUSIONS

We propose an improved location information coder using location histogram coding with sum-based context. Previous location information coders use location histogram coding with pattern-based context that has a size that grows exponentially as the number of reference points increases. Thus, the number of reference points is typically limited to a small number. The sum-based context has a context size that grows linearly as the number of reference points increases. Thus, it can be used for a larger set of reference points while still maintaining a small context size. Because of the smaller context size and the ability to use a larger set of reference points, we find that the sum-based location coder can be trained more easily and also perform better than the pattern-based location coder. Furthermore, the sum-based location coder is both fast and memory efficient. To speed up the generation of the sum-based context, we introduce a method based on integral images. Using integral images, the sum-based context can be generated in a fixed time regardless of the number of reference points used. To further reduce the context size, we show that we can quantize the sum to a smaller set of numbers. We can reduce the context size by > 200 times compared to previous location coders while being able to perform better.

The proposed location coder has been submitted as an input contribution to the Compact Descriptors for Visual Search group in MPEG and is being considered for adoption. Compared to the Test Model location coder, the new location coder provides a bitrate reduction of up to $\sim 20\%$.

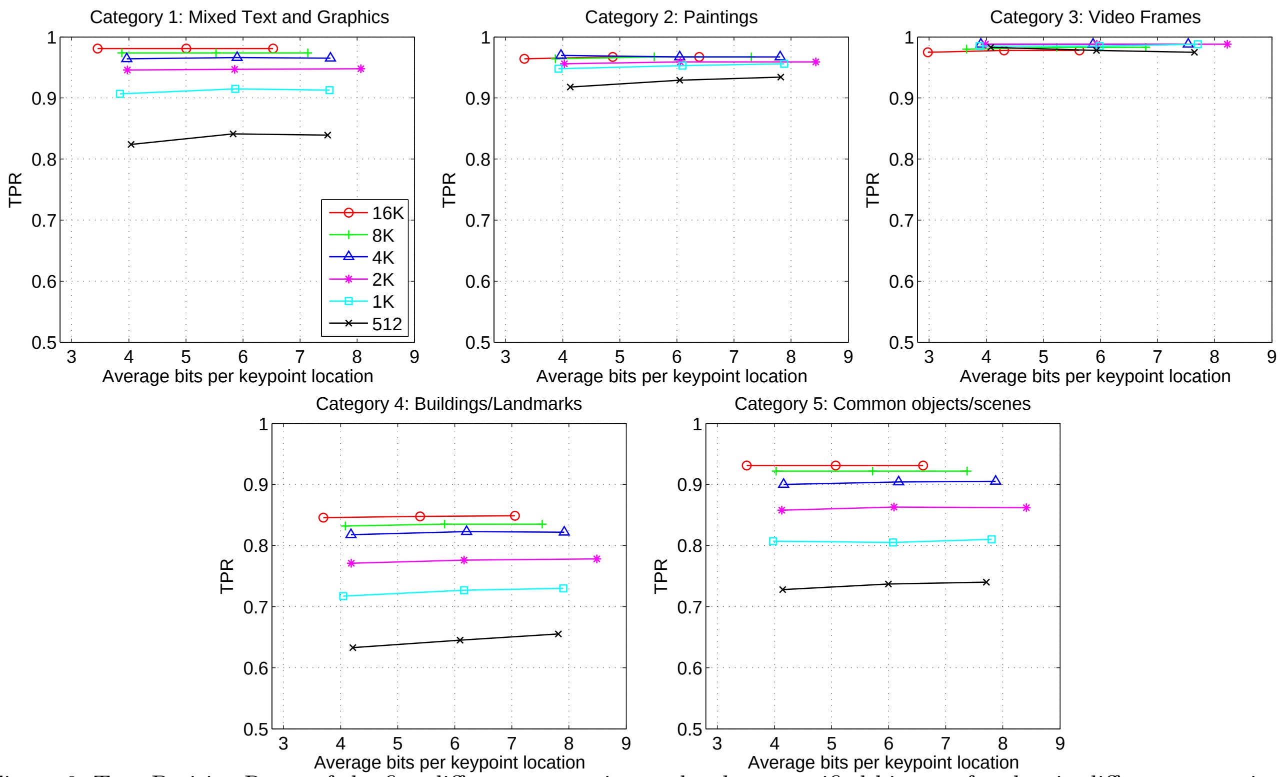


Figure 9. True Positive Rates of the five different categories at the three specified bitrates for the six different operating points. All points have a FPR that is smaller than 0.01.

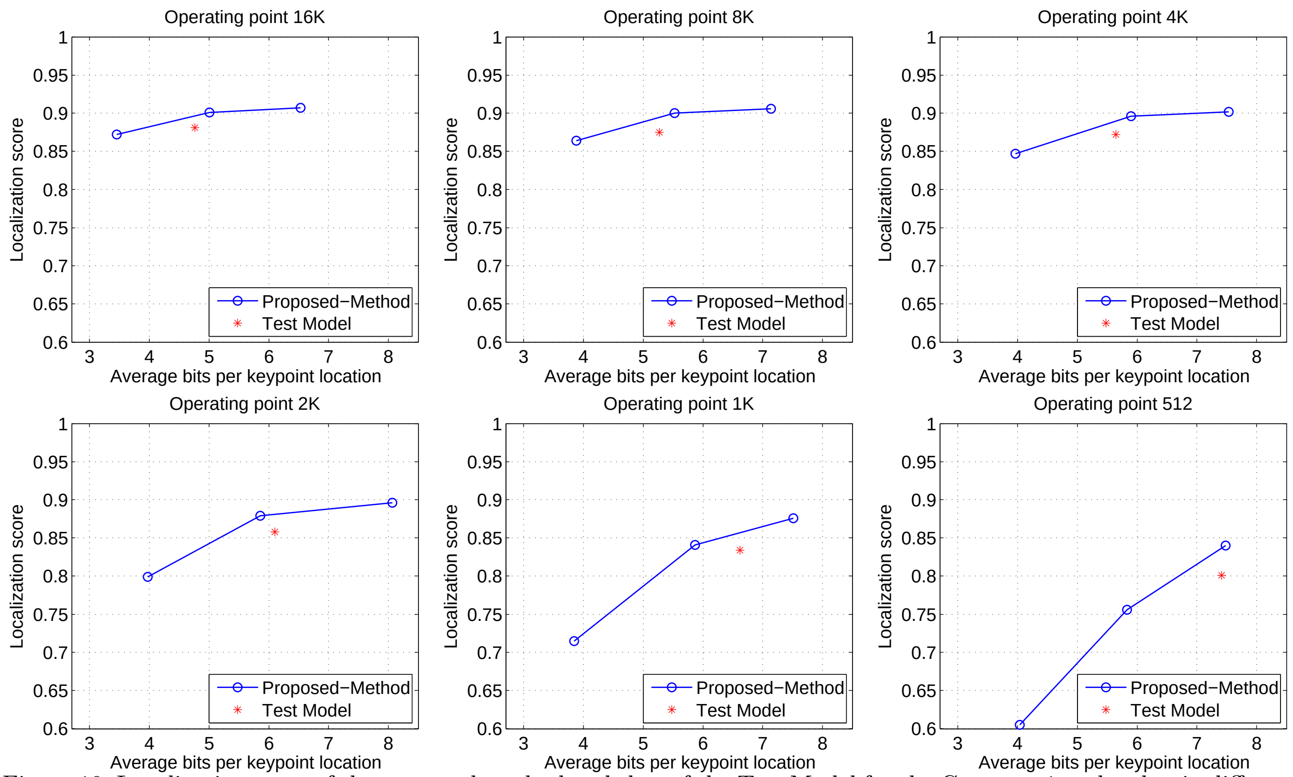


Figure 10. Localization score of the proposed method and that of the Test Model for the Category 1 under the six different operating points.

REFERENCES

[1] Girod, B., Chandrasekhar, V., Chen, D. M., Cheung, N.-M., Grzeszczuk, R., Reznik, Y. A., Takacs, G., Tsai, S. S., and Vedantham, R., "Mobile visual search," *IEEE Signal Processing Magazine* (2011).

[2] Takacs, G., Chandrasekhar, V., Gelfand, N., Xiong, Y., Chen, W., Bismpigiannis, T., Grzeszczuk, R., Pulli, K., and Girod, B., "Outdoors augmented reality on mobile phone using loxel-based visual feature organization," in [*ACM International Conference on Multimedia Information Retrieval*], (October 2008).

[3] Chen, D., Tsai, S. S., Vedantham, R., Grzeszczuk, R., and Girod, B., "Streaming mobile augmented reality on mobile phones," in [*International Symposium on Mixed and Augmented Reality*], (October 2009).

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

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

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

[7] Bay, H., Tuytelaars, T., and Gool, L. V., "SURF: speeded up robust features," in [*European Conference on Computer Vision*], (2006).

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

[9] Ke, Y. and Sukthankar, R., "PCA-SIFT: a more distinctive representation for local image descriptors," in [*Conference on Computer Vision and Pattern Recognition*], 511–517 (June 2004).

[10] Hua, G., Brown, M., and Winder, S., "Discriminant embedding for local image descriptors," in [*International Conference on Computer Vision*], 1–8 (October 2007).

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

[12] "Compact descriptors for visual search: Context and objectives," in [*ISO/IEC/JTC1/SC29/WG11/N11530*], (2010).

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

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

[15] Chen, D., Tsai, S. S., Chandrasekhar, V., Takacs, G., Singh, J., and Girod, B., "Tree histogram coding for mobile image matching," in [*Data Compression Conference*], (March 2009).

[16] Chandrasekhar, V., Reznik, Y. A., Takacs, G., Chen, D. M., Tsai, S. S., Grzeszczuk, R., and Girod, B., "Compressing feature sets with digital search trees," in [*International Workshop on Mobile Vision*], (2011).

[17] "Description of test model under consideration for cdvs," in [*ISO/IEC/JTC1/SC29/WG11/N12367*], (2011).

[18] Crow, F. C., "Summed-area tables for texture mapping," *Computer Graphics (SIGGRAPH '84 Proceedings)* **18**, 207–212 (July 1984).

[19] "Description of core experiments on compact descriptors for visual search," in [*ISO/IEC/JTC1/SC29/WG11/N12551*], (2012).

[20] Tsai, S., Chen, D., Chandrasekhar, V., Takacs, G., Makar, M., Grzeszczuk, R., and Girod, B., "Improvements to the location coder in the TMUC," in [*ISO/IEC/JTC1/SC29/WG11/M23579*], (2011).