ISSN (Print) : 0974-6846
Indian Journal of Science and Technology, *Vol 10(26), DOI: 10.17485/ijst/2017/v10i26/113242, July 2017* ISSN (Online) : 0974-5645

Event Image Archive using Codebook Generation

S. Regina Lourdhu Suganthi* **and M. Hanumanthappa**

Department of Computer Science and Applications, Bangalore University, Bangalore – 560056, Karnataka, India; reginalsuganthi@gmail.com, hanu6572@bub.ernet.in

Abstract

Objective: To develop a system that aims at optimal storage for voluminous image data sets that are acquired during various events, by generating codebook for image clusters by exploiting the fact that these images are generally multiple shots of similar scenes at frequent intervals. **Methods:** Images taken during the events on various occasions in organizations are increasing exponentially and pose tremendous challenge in terms of storage and retrieval. The volume, veracity and variety of features present in image data induce complexity in computation. Optimal storage and efficient tagging will ease the retrieval process of these image data. Among various lossy image compression techniques, vector quantization yields desirable compression ratio in many applications and is one of the efficient approaches for image compression. In this research work, vector quantization technique is explored to optimize the storage space required to maintain an archive of event image data set, by generating code book for a cluster of images. **Findings:** The similarity in the images that are acquired during a short span of time induces redundancy. This fact has been exploited by organizing the image data set into clusters that are similar. For each cluster, vector quantization technique is used to generate code book. The codebook generated has been used to encode the image by creating an index table for each image. The codebook of the cluster and the index table of each image is further used for decoding. The compression ratio and the peak signal to noise ratio of this method are above eighty percent and thirty DB respectively. **Applications:** The codebook generation technique described in this work could be applied for creating image archives. Archival images are generally voluminous and consume huge storage space. The code book generated for image clusters would reduce the storage requirement. This work also finds its application in transmitting video and image clusters in a network.

Keywords: Codebook, Compression, Vector Quantization, GLA, LBG

1. Introduction

Multimedia data in the form of images and videos lock-up enormous storage space in devices as opposed to text data. To reduce the storage requirement various algorithms have been developed. The objectives of these algorithms are to minimize the storage space and to ensure that the distortions induced through these techniques are less significant to human perception. Image processing techniques are largely domain specific and the application domain under focus here is, image sets taken during various events in organizations. In this view, the primary purpose of captured images is for documentation and thus a minimal loss in the information would not effect and compromise on the print quality. Vector quantization is most powerful technique for image transmission. Though vector quantization reduces the bit rate for transmission, it retains the image quality. The generation of an appropriate code book is the center for this technique. An attempt is made in this work to apply vector quantization technique to generate code book for image clusters that are similar. The three major steps in this process are namely, vector formation, code book generation, image encoding. The code books generated for different image clusters can further be used to create an image archive that will reduce the storage space required for maintaining these archives.

The paper is organized as follows: The following section presents the literature survey of the techniques applied in the present work. Section 2 presents the proposed scheme for code book generation. Section 3 discusses the experimental results of an organizational data set and Section 4 concludes with scope of the work.

The complex features present in the image data could be categorized as low-level and high-level features.

* *Author for correspondence*

Different applications demand distinct features to be used for image representation, image understanding and image retrieval. The aim of this work is to create an archive of enterprise image collection for future retrieval and reference. Present days, Events organized in any Enterprise are documented using images and videos. This generates huge volume of image data. These data need to be archived for need based surfing.

The two main objectives of creating an image archive for the event domain is to reduce the storage space and to organize the collection efficiently for future reference. One of the important factors that will address efficient storage is to eliminate redundancy and to reduce the space for storage. Most common technique for reducing the storage space is compression. Two classes of image compression are lossy compression and lossless compression[1]. The Lossless compression algorithms reconstruct the original image in its entirety. On the other hand the lossy compression algorithms are irreversible. Lossless compression is essential in applications such as medical imaging, technical drawing and many other such fields. Integer to Integer multi-wavelet transform[2] based on lifting scheme is used for lossless image compression. Integer to Integer multi-wavelets transform is based on multi-scaling and multi-wavelets function. The important part of multi-wavelet transform is lifting scheme where integer operation is used to achieve the expected transform.

In the process of development of image compression techniques, to attain perpetual quality, visual redundancy is combined with statistical redundancy[3,4]. These procedures compress the image by using image features. Instead of pixel values, Scale Invariant Feature Transform (SIFT) descriptors[5] are used to compress the image. A down sampled image with key SIFT descriptors are generated as one component and the rest are simplified to form a differential set. The groups formed are compressed separately. During the process of reconstruction, the decoder uses the visual correlation through SIFT based matching.

Compression ratio is generally high when lossy compression[6] is applied to the image data sets. Vector quantization is one of the efficient approaches that is used in speech recognition, pattern recognition and video encoding. The foundation for vector quantization technique is Linde-Buzo-Gray (LBG) algorithm[7], and it is based on minimization of the squared-error distortion measure. The objective of vector quantization is to divide the image matrix into small blocks and represent each block as a k-dimensional vector[8] and represent the image matrix $X \subseteq R^k$ mapped to the codebook. The primary challenge here is the codebook generation and its convergence. LBG suffers local optimality. The other approaches for codebook generation[9] include Directed Search Binary Splitting (DSBS), Mean distance ordered Partial Search (MPS), Centroid Neural Network Adaptive Resonance Theory (CNN-ART), Evolution based Tab Search Approach (ETSA), Codebook Generation Algorithm Using Codeword Displacement (CGAUCD), and many other hybrid schemes[10]. Once the codebook is generated the test images are mapped to the codebook and its corresponding index tables are generated. The size of the codebook is the next challenge in terms of its computational cost.

2. Proposed Work

The process architecture for codebook generation and mapping of Enterprise event image collection is shown in Figure 1.

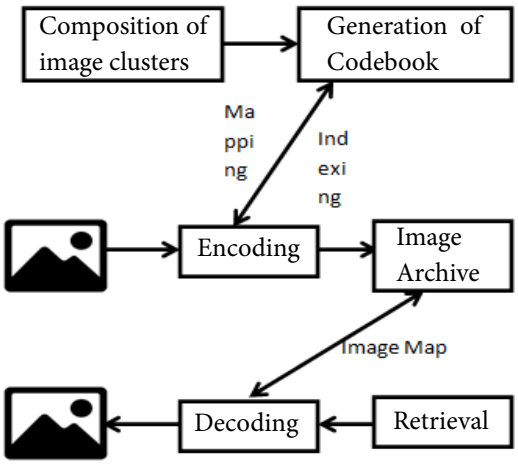

Figure 1. Process architecture.

2.1. Composition of Image Clusters

Images that are captured during events can be grouped into clusters based on structural similarity. Images in each cluster will have many matching blocks of gray levels. This observation is well exploited in codebook generation and to minimize the size of the codebook.

2.2. Generation of Codebook

An image I is a matrix of gray levels of 8-bit depth. It can

be viewed as a set of blocks of size b, hence each original image is resized to a nearest multiple of b. Each image in the cluster is of size M by N where M and N are the multiples of b. Once the images are resized, each image is divided into blocks of size b and each block is reshaped into a vector of size b^2. This forms a reshaped matrix of gray levels of size m by n where $m=M*N/b^2$ and $n=b^2$. The temporary codebook is generated by comparing each vector with the remaining vectors and eliminating those vectors which are closely redundant. Closely redundant vectors are identified by computing the distance between the two vectors. If the maximum of the absolute difference between each of the b^2 gray level pairs is less than a threshold then one of the two vectors could be used to represent both the blocks as these are closely redundant. This process is depicted in Figure 2 and Figure 3. The procedure for the elimination of closely redundant vectors/code words is given in Algorithm 1.

Algorithm 1. Closely Redundant Codeword Elimination

Algorithm Redundant (CBTemp[1..m, 1..b^2] , T)

```
for i = 1 to m - 1
  for j = i+1 to m
    diff = abs ( v_i - v_j )
    maxi = max( diff )
    if (maxi < T )
      copy (CBTemp[j..m-1, 1..b^2], CBTemp[j+1..m, 1..b^2])
      m=m-1
    end
  end
end
```

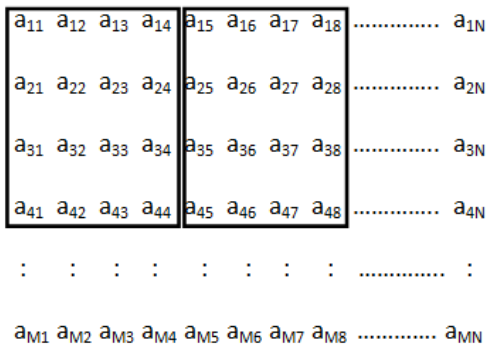

Figure 2. Image I with block size b= 4.

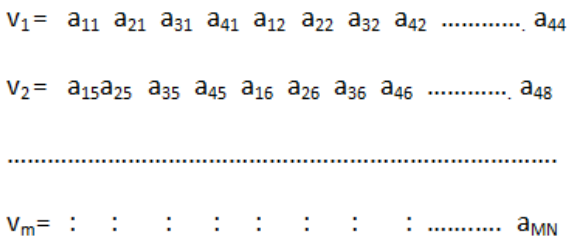

Figure 3. Block vector Matrix of size m by b^2 where $m = M*N/b^2$.

The elimination of redundant codeword's reducing the size of the temporary codebook. The procedure is repeated for all the images. This results in temporary codebooks of varying size representing each image in a cluster. These temporary codebooks are further processed for removal of redundant codeword's. This completes the process of generating a codebook for a cluster.

2.3. Encoding

An image in the cluster is encoded by dividing the image matrix into blocks of size b and reshaping each block into a vector of length b^2. Each vector is searched for a nearest codeword in the codebook that was generated for the cluster. The nearest codeword is identified by choosing that codeword whose distance is minimum among all the matching codeword's. The index of that matching codeword in the codebook is assigned to the corresponding vector of the image matrix resulting in an encoded image map matrix Imap of size m by X where X is the number of images in the cluster.

2.4. Image Archive

The image archive is created by storing the codebook of each image cluster and the URL of the images in the cluster coupled with the encoded index vectors. If a cluster has a collection of X images the space requirement would be X * M * N bytes. On the other hand the lossy compression using codebook generation and encoding technique would require the space for the codebook and for X image maps each of size m * 2 bytes where $m = M*N/b^2$.

2.5. Decoding

The requested image URL during image retrieval is searched in the image archive. The cluster to which this URL is associated is identified and the corresponding codebook and the image map is extracted. Using the

image map the respective vectors are derived. As each vector represents a block, these vectors are reshaped into an image matrix of size M by N.

3. Experimental Results

An image set of two hundred images captured during an event organized in an Institution has been considered. The image set consists of images that were taken during a seminar. These images are divided into clusters based on time stamp and structural similarity index. One cluster comprising of five similar images has been taken for codebook generation, encoding and decoding. MATLAB software has been used for processing.

To generate the codebook, an uncompressed image I shown in Figure 4, from the chosen cluster is randomly taken to obtain a temporary codebook. The block size can be chosen as per the application domain and the permissible distortion due to the result of lossy compression. Here the block size is assumed to be b=4. A portion of the image matrix of image I and the respective reshaped vectors of size b^2=16 are shown in Table 1a and Table 1b.

Figure 4. Original Image I.

Table 1a. Portion of original image matrix I[101..112, 1..8] of size 400 x 300

80	81	81	82	82	81	81	80
88	89	89	90	90	89	89	88
91	92	92	93	93	92	92	91
96	97	98	98	98	98	97	96
100	101	101	101	100	99	98	97
101	102	102	102	102	101	100	99
103	103	104	104	104	103	102	101
105	105	106	106	106	105	105	104
106	106	107	108	108	107	107	106
106	107	108	108	109	109	108	108
106	107	108	109	109	109	109	108
106	106	108	109	109	109	109	109

Table 1b. Vectors of the blocks 2501, 2502, 2601, 2602

v_{2501}	80	88	91	96	81	89	…	98
v_{2502}	82	90	93	98	81	89	…	96
v_{2601}	100	101	103	105	101	102	…	106
v_{2602}	100	102	104	106	99	101	…	104

The matrix I is reshaped to the matrix CBTemp of size m by b^2 with m vectors of size b^2 as per the block size where m = M * N / b^2 = 400 * 300 /16 = 7500. By applying the algorithm 1 to the matrix CBTemp, the closely redundant codeword's as explained in Table 2 are eliminated, thus resulting in only 2589 codewords of length 16 each. The threshold T is assumed to be 10.

Table 2. Diff = abs (V_{2501} – V_{2502})

2	2	2	2	0	0	0	1	0	0	0	1	2	2	2	2

Since max(Diff) < T one of the two codewords is eliminated. The five images in the cluster resulted in CBTemp matrices of sizes [2589 x 16, 2239 x 16, 2244 x 16, 1866 x 16, 2250 x 16]. The matrices are concatenated vertically into a single matrix of size 11488 x 16. This concatenated matrix is again processed to eliminate closely redundant codewords. The final codebook CBBook is resulted in size 9295 x 16.

The images are encoded using CBBook and the index map of the cluster IMap of size m x X where m = 7500, X=5 (Number of images in the cluster) is obtained. The IMap of the first image is shown in the Table 3.The image archive is created by storing the cluster numbers, codebooks of each cluster, Image URLs of each image in the clusters and the respective Imaps. This procedure is a lossy compression technique.

Table 3. IMap[1..10, 1]

2	2	2	2	466	466	2	1097	2	2

To retrieve an image, the cluster to which the URL of the queried image is associated is searched. The codebook CBBook of that cluster and the IMap are retrieved. The image is reconstructed by locating the codewords from the codebook using the IMap of that image and reshaping the vectors to the corresponding blocks of the image. The decoded and reconstructed image is presented as the query result. Figure 5 is the reconstruction of the original image I of Figure 4.

Figure 5. Reconstructed Image I.

The Peak Signal to Noise Ratio (PSNR) of the original and reconstructed image is computed for all the images in the cluster as given below :

```
I = double(I);
IRe = double(reconstructed_image);
MSE = sum ( ( I(:) - IRe(:) ).^2 ) / prod ( size(I) );
PSNR = 10 * log10 ( 255*255 / MSE );
```

The PSNR for the five images in the cluster are respectively: 33.8648, 30.4659, 30.0014, 29.6607, 30.4003. For an 8-bit depth image the typical PSNR values of a lossy compression lie between 30 and 50 DB.

The experimental results show that the storage requirement for the defined cluster of five images without compression require M * N * 5 = 400 * 300 * 5 = 600000 bytes, but, using the codebook CBBook and the Imap, the storage space required for the same cluster of five images would be :

Size of CBBook = 9295 * 2 = 18590 bytes and Size of IMap = 7500 * 5 * 2 = 75000 bytes

Totaling just 93590 bytes. The compression ratio is: (600000-93590) / 600000 = 0.844 * 100 = 84.4%

It is evident that, for this application domain the proposed lossy compression procedure achieves greater compression rate. It can be observed that distortion induced through the codebook generation totally depend on the choice of the block size b and the threshold T.

4. Conclusion

The distortion is directly proportional to the block size b and the threshold T. The time-complexity for generating the codebook and the IMap is high and rely on the size of the codebook. With the support of multiple processors, threads can be created to perform independent tasks of the procedure. The same procedure could be extended to color images by considering the three planes R-plane, G-plane and B-plane independently. The codebooks for each of the three planes can be generated for each cluster and the images could be decoded and reconstructed using the same procedure.

5. References

1. Chen T, Chuang K. A Pseudo lossless Image Compression Method. IEEE Congress on Image and Signal Processing. 2010; 2: 610–5.
2. Mohsin S, Sajjad S. Codebook generation for Image Compression with Simple and Ordain GA. International Journal Of Computers And Communications. 2007; 1(2): 1–6.
3. Nobuhara H, Pedrycz W, Hirota K. Fast solving method of Fuzzy relational equation and its application to lossy image compression/reconstruction. IEEE, Transactions on Fuzzy Systems. 2000 June; 8(3):325–34.
4. Gupta K, Sharma M, Baweja N. Three different KG version for Image Compression. International Conference on Issues and Challenges in Intelligent Computing Techniques (ICICT) IEEE, Gazihabad, India. 2014.
5. Yue H, Sun X, Yang J, SIFT-Based Image Compression. IEEE, International Conference on Multimedia and Expo, Melbourne, Australia. 2012.
6. Shen Y, Gao X, Liu L, Cao Q. Integer to Integer Multiwavelets for Lossless Image Compression. Proceedings of IEEE IC-BNMT, Shenzhen, China. 2011. p.217–21.
7. Asmita AB, Tijare PA. A Review on LBG Algorithm for Image Compression. International Journal of Computer Science and Information Technologies. 2011; 2(6): 2584–89.
8. Lu TC, Chang CY. A Survey of VQ Codebook Generatio. Journal of Information Hiding and Multimedia Signal Processing. 2010 July; 1(3): 190–203.
9. Tsai CW, Lee CY, Chiang MC, Yang CS. A Fast VQ Codebook Generation Algorithm via Pattern Reduction. Pattern Recognition Letters. 2009. 30: 653–60.
10. Mittal M, Lamba R. Image Compression Using Vector Quantization Algorithms: A Review. International Journal of Advanced Research in Computer Science and Software Engineering. 2013 June; 3(6): 354–8.