A near-lossless image compression algorithm using vector quantization

EMIL MARIN POPA, ALINA PITIC, ANTONIU PITIC
Faculty of Science
"Lucian Blaga" University of Sibiu
Ratiu Street, no. 5-7, Sibiu
ROMANIA

Abstract: Lossy image compression methods are based on error measuring at entire image level only. In some areas there is an obvious need for getting an upper bound for the error at pixel level. In the paper we propose a near-lossless compression algorithm based on DPCM simple scheme, followed by a vector quantization. The exit will have a non uniform distribution, so it will be further compressed using an entropy based method. Experimental results obtained and presented in the paper prove that the vector quantization method gives us better results than the scalar quantization and the classic LBG algorithm, in the near-lossless context.

Key-Words: - lossless, near-lossless, differential encoding, scalar quantization, vector quantization, LBG.

1 Introduction

Today, in the age of computers and communication, there is an obvious need for data compression and, especially, for image compression.

In many applications, for example medical imagery or satellite pictures the large amount of data to be stored or transmitted asks for efficient data compression. Lossless image compression usually gives a compression ratio of 2:1 which is, in most cases, unsatisfactory. When higher ratios are needed, lossy coding methods have to be employed.

Because the compression methods where no error is accepted (called lossless) give poor results on images, we have to accept some error in the decompressed image in order to get much higher compression ratios. Those methods (called lossy) have become very popular especially with the multimedia development. Consecrated lossy methods, as the one based on DCT and stated in the JPEG standard, while offering a very good compression ratio (usually over 10:1), without major loss on the subjective visual aspect, do not offer any information on the magnitude and the position in the image where some of the initial information is lost. The main drawback of these schemes is that we cannot have any guarantees about the error at pixel level; therefore there is no control on the possible artifacts introduced by the compression.

In this classic approach the criterion that is minimized by the image compression methods is:

$$RMSE(R,O) = \sum \left(R_{ij} - O_{ij}\right)^2 \qquad (1)$$

where R_{ij} represents the restored (decompressed) image, O_{ij} represents the original (uncompressed) image and the sum is made over the entire image. Here we are interested in the error at entire image level. The error expression corresponds to the classic Euclidean distance measure.

Another class of applications is the one where it is very important to get an upper limit on the error at each pixel. Now the criterion to be minimized is:

$$CHE(R,O) = \max \left|R_{ij} - O_{ij}\right| \qquad (2)$$

where R_{ij} and O_{ij} have the same meanings and the maximum is searched also over the entire image. Here we are interested in the error at pixel level. The error expression to be minimized corresponds to the Chebyshev distance measure.

The first approach (based on *RMSE*) is the classic one and a lot of research has been done during years on that topic. In some applications (satellite images, medical images, etc.), there is a need for some guarantees at pixel level because the artifacts introduced by the compression methods are unacceptable (without an upper limit), especially for further automatic processing. Therefore, in such area image compression was very rarely accepted (or not accepted at all).

In the last years the interest for such compression methods (based on *CHE*) is growing quickly and, therefore, the area requires a lot of research ([5],[6],[8],[9]). Because usually we are interested only in small values for the acceptable error limit (±1, ±2, …, ±10) the method is called near-lossless (NL). Certainly, if the acceptable error level becomes 0 we get the classical lossless compression. It is proven that the task of finding the best model

(and compression) within the acceptable error ±d is a NP-complete task.

2 Differential encoding

When compressing sources with a great deal of correlation of the samples, as images, we can predict current sample based upon previous samples. The difference between the sample and the predicted value is coded and transmitted. A basic differential encoding system, known as the differential pulse code modulation (DPCM) is shown in Fig. 1.

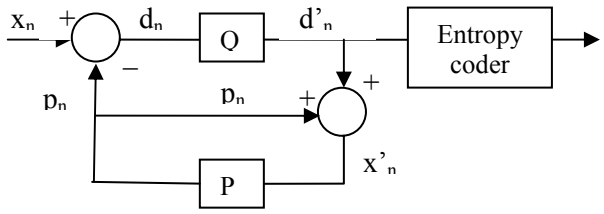

Fig.1 – The basic DPCM scheme

3 Quantization

In many lossy compression applications we are required to represent each source output using one of a small number of codewords. The number of possible distinct source output values is generally much larger than the number of codewords available to represent them. The process of representing a large set of values with a much smaller set is called quantization.

3.1 Scalar quantization

In the scalar quantization, usually named simply quantization, the inputs are individual numbers. Scalar quantization includes such operations as "rounding to the nearest integer." The possible outputs, in this case the integers, are called quantization levels or reconstruction levels. In general, the quantization levels do not have to be integers, nor evenly spaced. The spacing between the reproduction levels is often called the bin width. To specify a scalar quantizer, one needs the set of possible reproduction levels and a rule for mapping input scalars to reproduction levels.

A probability model for the variations of pixels in an image is almost impossible to obtain because of the great variety of images available. A common approach is to assume that the pixels values are uniformly distributed within [0, N), where N is the number of gray levels of the input image. In this paper we consider N=256 and we will use a uniform quantizer.

However, in some specialized domains, as medical imagery, the images have a known nonuniform pixel distribution. In this case, nonuniform quantizers (as companded quantization) can be used.

3.2 Vector quantization

Vector quantization has found increasing use in data compression, especially in multimedia applications, such as image and speech coding. The technique is an extension of scalar quantization to the vectorial case, and is motivated by the well known result from Shannon's rate distortion theory that superior performance can always be achieved by coding vectors rather than scalars. As with scalar quantization, the objective in vector quantization design is to minimize a distortion criterion (RMSE in most cases). An iterative error minimization algorithm developed by Lloyd for scalar quantization was extended to the vector case by Linde, Buzo, and Gray ([7]). The iterative nature of this algorithm makes it computationally very intensive. A host of other suboptimal but computationally simpler vector quantization techniques have been reported in the literature either as an alternative to the Linde-Buzo-Gray (LBG) algorithm, or as an initial step that may be refined by the LBG technique. Until recently, the general class of vector quantization techniques has not received much attention for practical implementation because of the high computational cost, and the lack of codebook structures that are amenable to hardware implementation.

The LBG algorithm, in the case we have a reconstruction set of vectors available, the algorithm is (similar with k-means algorithm):

Step 1. Start with an initial set of reconstruction values $\{Y_i^{(0)}\}_{i=1}^{M}$ and a set of training vectors $\{X_i\}_{i=1}^{N}$. Set k=0, $D^{(0)}=0$. Select threshold ε.

Step 2. The quantization regions $\{V_i^{(k)}\}_{i=1}^{M}$ are given by

$V_i^{(k)} = \{X_n : d(X_n, Y_i) < d(X_n, Y_j), \forall j \neq i\}$,

i=1,2,…,M

Step 3. Compute the average distortion $D^{(k)}$ between the training vectors and the representative reconstruction value.

Step 4. If $\frac{(D^{(k)} - D^{(k-1)})}{D^{(k)}} < \varepsilon$, stop; otherwise continue.

Step 5. k=k+1. Find new reconstruction values $\{Y_i^{(k)}\}_{i=1}^{M}$ that are the average value of the elements

of each of the quantization regions $V_i^{(k-1)}$. Go to step 2.

A straightforward vector extension of traditional scalar predictive quantization (DPCM) is predictive vector quantization. The encoder makes a prediction of the incoming vector based on previously encoded vectors. The difference between the actual input vector and its prediction is called the residual vector. This residual is vector quantized. Because the encoder only uses the previous outputs in making its prediction, the decoder is able to make the same prediction. After dequantizing the residual vector, the decoder adds the prediction to it to form the reproduction vector. The prediction is, in our case, a simple linear predictor that takes a weighted average of nearby previously encoded coefficients.

4 The algorithm

A simplified scheme of the proposed algorithm is shown in the Fig. 2.

DPCM → VQ → Entropy Coder (optional)

Fig. 2 – The coder scheme

The first step is used because, as we see in Fig. 4, the dynamic range of the differences between pixels (the simplest predictor) is substantially lower than the dynamic range of the initial pixel values.

The major drawback of the LBG algorithm, in the near-lossless context, is the modality of calculus of distortion. There are no guarantees to obtain clusters in which all vectors respects relation (2). In addition, the number M that denotes the codebook size has to be known in advance.

Therefore, as an initial step, we realize a clustering of the input vectors with respect to near-lossless condition.

Initial clustering

Step 1. Start with a set of training vectors $\{X_i\}_{i=1}^{N}$. Initialize M=0 (initial position in the codebook). Set d as the near-lossless error.

Step 2. Select an unallocated vector X_k and made it the current codebook entry $Y_M = X_k$.

Step 3. The quantization region V_M is given by

$$V_M = \{X_i : CHE(X_n, Y_M) < k, X_i \notin V_j,$$

$$\forall j \in [0, M-1]\}, i = 1,2,...,N$$

Step 4. M=M+1.

Step 5. If all training vectors are allocated STOP. Else go to Step 2.

To speed up the steps 2 and 3 we use an additional N dimensional vector.

In the end, M will be the dimension of the resulting codebook. Some results for Lena and Peppers test image are shown in Table 1. If the algorithm stops now and we consider a fixed size representation of the dictionary entries we achieve results around 5 bpp.

The codebook will be used as a start for successive refinements. After sorting, the number of vectors associated with each codebook entry look like in Fig. 5. In the presented case (Lena, vector dimension=4, near-lossless distance=2), most of the codebook entries have less than 3 corresponding vectors (64% with 1 vector, 13 % with 2, 8% with 3). Other test images and parameter values lead us to the same results, typically 80-90% of the codebook entries having less than 3 vectors in their quantization region.

The next step is to reduce the number of codebook entries, by successive search in the vectors that are part of small quantization region (in our experiments we consider t=3 as minimum vector count). The number of training vectors is, in this case, considerable lower, typically 9%-16% from the initial number of the training vectors, this making a more exhaustive search possible.

Refinement

Step 1. Set a minimum quantization region size t and MM=0 (additional codebook entries). Construct initial training vectors:

$$\{T_i\}_{i=1}^{NN} = \{X_n \in V_j : card(V_j) <= t\}$$

Step 2: Sort the initial vectors $\{T_i\}_{i=1}^{NN}$ ascending by

$$R_{i,j} = \{i <= j \mid \max(|T_i[k] - T_j[k]|) < 2d,$$

$$\forall k \in [0, Dim-1], i, j \in [0, NN-1]\}$$

v= $\sum_{i=0}^{i<Dim} (T_i * x^i)$, x=512 (lexicographic order), using radix sort.

Step 3. Find the largest continue region of unallocated vectors:

Step 4. Set a reconstruction vector Y_{MM} corresponding to the vectors from the previous selected region (the average value of the elements).

Step 5. MM=MM+1

Step 6. If all training vectors are allocated STOP. Else go to Step 3.

The final codebook is constructed by replacing the entries corresponding to $\{T_i\}_{i=1}^{NN}$ with the new

entries $\{Y_i\}_{i=1}^{MM}$, this final step reducing the codebook size by 20%-30%. As a direct result, even if we use a fixed size representation, the compression is around 4 bpp. In this point all the initial training vectors obtained from the original image have a reconstruction vector with respect to the near-lossless criterion.

Even after the last step of our algorithm, most of the quantization regions have only a few corresponding training vectors. Therefore, a fixed size representation of the codebook entries is clearly suboptimal. The problem is solved by associating variable-length codes with the codebook entries, as Huffman or Golomb codes. Using Huffman codes the results are presented in the last column of the Table 2.

5 Results and conclusions

The input images in our experiments are Lena and Pepper images (512x512, 256 level gray scale).

Fig. 3 –Test images: Lena (left), Peppers (right)

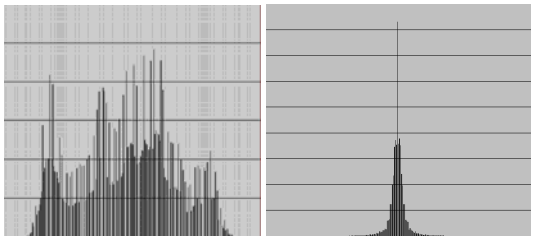

Fig 4. – Lena histogram (left) and differences histogram (right)

v. d.	d	M	v. d.	d	M	v. d.	d	M
2	1	1738	4	1	16159	8	1	28699
2	2	907	**4**	**2**	**9125**	8	2	19563
2	4	383	4	4	3966	8	4	10272

v. d.	d	M	v. d.	d	M	v. d.	d	M
2	1	2497	4	1	20005	8	1	32051
2	2	1413	4	2	10903	8	2	25753
2	4	658	4	4	5041	8	4	13653

Table 1 – Resulting codebook sizes for Lena (up) and Peppers (down) test images

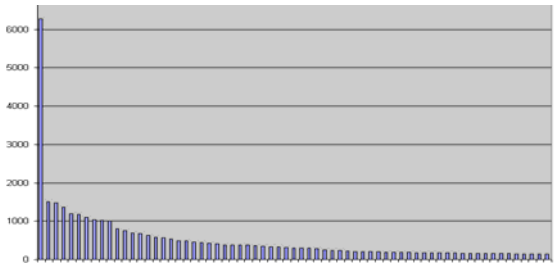

Fig 5. – A partial histogram of clusters sizes (Lena, vector dimension=4, d=2)

d	SQ	DPCM+SQ	S1	S2	Final
1	7	5.76	5.12	4.85	3.94
2	6	5.14	4.88	4.12	3.22
4	5	4.04	4.01	3.45	2.64

d	SQ	DPCM+SQ	S1	S2	Final
1	7	6.22	5.53	5.24	4.15
2	6	5.55	5.27	4.45	3.26
4	5	4.36	4.33	3.73	2.74

Table 2 – Results in bit/pixel for Lena (up) and Pepper (down) for vector dimension=4

The state-of-the-art near-lossless algorithms, JPEG2000, SPIHT, CALIC, have a better performance than our algorithm, especially for d<=10. If admitting a large error at pixel level (d>10) the results are similar to CALIC (differences are less than 0.2 bpp for the test images). However, the way we obtain the initial training vectors from the start image (i.e. the way we decide to scan the image) and the initial clustering strategy have a great impact over the achieved compression ratio.

The vector selections in the second step of the initial clustering, for the results shown in Table 2, is made by simply selecting the first unallocated vector and consider it the current codebook entry. By using a random selection of the unallocated vectors or by trying to select a vector with a higher probability of generating large quantization region, the result could be quite different from the presented results. The best results are presented in Table 3.

d	JPEG2000	SPIHT	CALIC	Proposed
1	2.58	2.58	2.60	3.22
3	1.50	1.50	1.57	1.96
7	0.73	0.73	0.83	0.95

d	JPEG2000	SPIHT	CALIC	Proposed
1	2.90	2.90	2.84	3.42
3	1.78	1.78	1.78	2.15
7	0.93	0.93	0.95	1.12

Table 3 – Results in bit/pixel for Lena (up) and Pepper (down)

As conclusions:

- The LBG and other similar algorithms give us no guarantees that the near-lossless criterion can be achieved for a given codebook size.
- The first step that realize the initial clustering, can be successfully used as an input into the classic LBG algorithm, by sorting the codebook entries by the size of quantization regions and then considering the first N entries as the initial codebook. This gives us better visual results compared with the random generation of the initial codebook, but no assurance of respecting the near-lossless criterion. However, the number of pixels where the near-lossless criterion is not respected is rather small (especially for d>=4), so the algorithm could be used in a lossy + lossless scheme.
- Using a lossy+lossless scheme, our algorithm performs better than the predictive vector quantization
- The performance of the proposed method (in term of achieved compression) is poor as a lossless scheme (d=0). For d>=2 the results are promising.
- Empirical results show that it is highly improbable to construct a codebook that works for a set of images and respects the near-lossless criterion of compression; therefore the codebook has to be transmitted to the decoder.
- If we use a simple vector selection strategy the results are much worse that the one achieved by the state of the art near-lossless algorithms
- By selecting the best vector selection strategy (highly dependent by the image being compressed), the results are comparable with the results given by CALIC, especially for large values of d.

References:

[1] Khalid Sayood, *Introduction to data compression*, Morgan Kaufman publishers, 2000

[2] Guy E. Blelloch: *Introduction to Data Compression* Guy E. Blelloch, Computer Science Department, Carnegie Mellon University, October 16, 2001

[3] G. Patane, M. Russo, The enhanced LBG algorithm, *Neural Networks* 14:1219-1237, 2001

[4] Cosman, P., Gray, R., & Vetterli, M.: Vector Quantization of Images. Subbands: A Survey. *IEEE Transactions on Image Processing*, 1996

[5] Breazu M., Pitic T., Volovici D., Brad R., Near-lossless LZW image compression, *SINTES 10 Conference*, Craiova, Romania, 2003

[6] Breazu M., Pitic T., Volovici D., Dictionary clearing vs. freezing in near-lossless LZW image compression, *CONTI Conference*, *Craiova, Romania,* 2004

[7] Y. Linde, A. Buzo, and R. M. Gray. An algorithm for vector quantizer design. *IEEE Transactions on Communication, COM-28:84-95*, 1980.

[8] Vleuten R. J., - Low-Complexity Lossless and Fine-Granularity Scalable Near-Lossless Compression of Color Images, *Data Compression Conference (DCC '02), Snowbird, Utah*, USA, 2001

[9] Yamauchi M., Wakatani A., A New Lossless Compression Scheme for Medical Images by Hierarchical Segmentation, *Data Compression Conference (DCC '01), Snowbird, Utah, USA*, 2001

[10] X. Wu, P. Bao, L_∞ Constrained High-Fidelity Image Compression via Adaptive Context Modeling, *IEEE Transactions on Image Processing,* 2000.