ENHANCING THE ENTROPY ENCODER OF A 3D-FWT FOR HIGH-QUALITY COMPRESSION OF MEDICAL VIDEO

Gregorio Bernabé, José González, José M. García

Universidad de Murcia
Dpto. Ingeniería y Tecnología de Computadores
30071 MURCIA (SPAIN)

José Duato

Universidad Politécnica de Valencia
Dpt. Inf. de Sistemes i Computadors
46071 VALENCIA (SPAIN)

ABSTRACT

Medical video is characterized by small interframe variations and the representation in gray scale. Traditional compression techniques such as MPEG2 cannot fully exploit these issues and makes other approaches, such as Wavelet-based coders much more interesting. We present in this paper several improvements in the entropy encoder of a 3D-FWT encoder: 3D-Conscious Run-Length, which exploits the high presence of zeros in the decorrelated sequence, an hexadecimal coding of the chains of zeros generated in the previous step and the application of arithmetic coding instead of Huffman. We have evaluated these enhancements improving the compression ratio of the original 3D-FWT encoder between 40% and 70%. We have also compared our scheme with MPEG2 and EZW (which is also based on the FWT), obtaining better compression ratios, up to 119% and 46% respectively, for the same PSNR.

1. INTRODUCTION

In the last few years, there has been a considerable increase in the volume of medical images and video generated in hospitals. In the average sized hospital, many tera-bytes (10^{15}) of medical data are generated every year which must be kept and stored since legislation requires all captured healthcare information to be preserved for a certain period of time before it can be deleted. Medical video has special features such as its representation in gray scale, the small amount of interframe variations, and the quality requirements of the reconstructed video to allow for a correct diagnostic. Therefore, compression techniques are needed to reduce the amount of information that needs to be handled.

Nowadays, three dimensional (3D) compression techniques seem to offer better results than two dimensional (2D) compression techniques which operate in each frame independently. MPEG2 [14] is the most outstanding compression scheme and has emerged as a widely accepted industry standard. However, MPEG2 presents several drawbacks. The MPEG coding scheme works with colors and divide the images into YUV components, whereas medical video is mainly represented in gray scale. In addition, some features of the MPEG2 encoding, as for instance the division of the image in blocks, makes this standard not suitable for achieving a good trade-off between quality and compression ratio for coding medical video [2].

This work has been partially supported by the Spanish CICYT under grant TIC2000-1151-C07-03. The video sequences have been donated by the Hospital Recoletas (Albacete, Spain).

On the other hand, the wavelet transform [5] has been mainly applied to image compression. Several coders have been developed using 2D wavelet transform [1][9][13]. The 2D wavelet transform has been used for compressing video [6] as well. Muraki introduced the idea of using 3D wavelet transform to efficiently approximate 3D volumetric data [10][11]. Since one of the three spatial dimensions can be considered similar to time, a 3D subband coding using the zerotree method (EZW) was presented to code video sequences [4] and posteriorly improved with an embedded wavelet video coder using 3D set partitioning in hierarchical trees (SPHIT) [7][8].

In this work, we present and evaluate several improvements over a new lossy video compression scheme, based on the use of the Fast Wavelet Transform (FWT) [3]. This method achieves high compression ratios with an excellent quality, so that medical doctors cannot find differences between the original and the reconstructed video.

In a previous work [3], we present an initial implementation of a lossy encoder for medical video based on the 3D wavelet transform. Figure 1 shows the key processing steps involved in this compression method.

First of all, the 3D-FWT is computed by succesively applying the 1D wavelet transform to the value of the pixels in each dimension. The thresholding step uses the x-percentile method to discard those coefficients whose value does not provide enough information. The quantizer transforms the floating points coefficients into unsigned integer coefficients. Thus, it is necessary to specify how many bits are needed to encode each of the wavelets layers in each independent frame. In the entropy encoder, a run-length compression is applied to the binary representation of the integer coefficients, exploiting the presence of chains of zeros. Finally, a 128-symbol Huffman encoder is applied.

The rest of this paper is organized as follows. Section 3 explains the improvements to improve our original encoder. Experimental results with some test medical video are analyzed in Section 3 and comparisons of our encoder with MPEG2 and EZW in coding of two medical video sequences. Section 4 summarizes the work and concludes the paper.

2. THE PROPOSED METHOD

In order to increase the compression ratio, maintaining the video quality we have developed several improvements in the quantization and the entropy encoder.

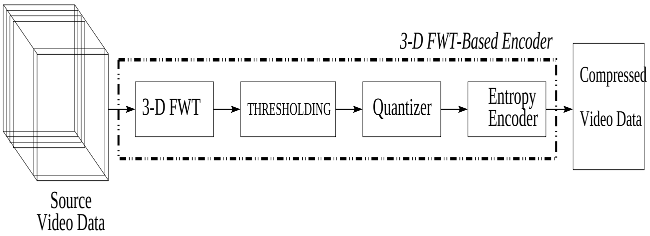

Fig. 1. 3D-FWT-based encoder processing steps

2.1. 3D-Conscious Run-Length

In order to the binary run-length be more effective, the highest possible number of zeros from the decorrelated image must be placed in the row order. In the previous approach, the coefficients of each independent frame are processed in a similar way as JPEG aligns coefficients before applying the entropy encoder. However, since the 3D-FWT is applied, the original sequence is also decorrelated in the time axis. Therefore there is a higher likelihood of having longer chains of zeros if coefficients are processed taking into account the time dimension, particularly for the H bands. We refer to this technique as 3D-Conscious Run-Length.

Moreover, the quantizer proposed in this work takes into account the number of times that the 3D-FWT has been applied, assigning different number of bits depending on the subcube. For instance, for a video of 64 frames of 512x512 pixels, the first 3D-FWT is applied over the whole sequence. The second application is just performed to the subcube consisting of 32 frames of 256x256 pixels (video reference). This subcube will generate higher wavelets coefficients, needing more bits in the quantization process.

2.2. Hexadecimal coding

In the former approach, the run-length compressed chains of up to 128 zeros since the Hufman code used 128 symbols. This limits the enhanced approach, since chains of thousands of zeros may appear after traversing the video taking into account the time dimension. We propose to represent the chains of zeros using the hexadecimal representation of their length in the following way: A first symbol (hexadecimal number) indicates the number of hexadecimal symbols used to code the length of the chain. Then, the following hexadecimal symbols code this length. Therefore, unlimited chains of zeros can be represented, and Huffman code will be reduced to 16 symbols. For example, if the decorrelated video has 15503 zeros in a row, the former run-length code needs 366 bytes to encode it. Now, this length is represented as $43C8F$ (0x3C8F = 15503), drastically reducing the number of bytes needed to encode it (2.5 bytes). This hexadecimal coding can be improved with some extensions:

First, chains of up to seven zeros in a row need one byte to be coded: the first 4-bit symbol is always 1 and the second 4-bit symbol ranges from 1 to 7. We have verified that the largest chain of zeros does not need more than seven hexadecimal symbols. If we fix this bound, chains of more 268 millions of zeros can be compressed in 4 bytes. The improvement is based on representing only with one symbol (1/2 byte) the chains of zeros ranging from one to seven. If the first bit is zero, the symbol indicates the number of hexadecimal symbols that code the chain of zeros, whereas if the first bit is one, the following three bits directly indicate a length between 1 and 7.

Second, the free 8 hexadecimal symbol appears to indicate a chain of ones, then an hexadecimal symbol is used to determine the number of hexadecimal symbols that follow and finally, the symbols representing the number of ones in a similar way as performed with the chains of zeros.

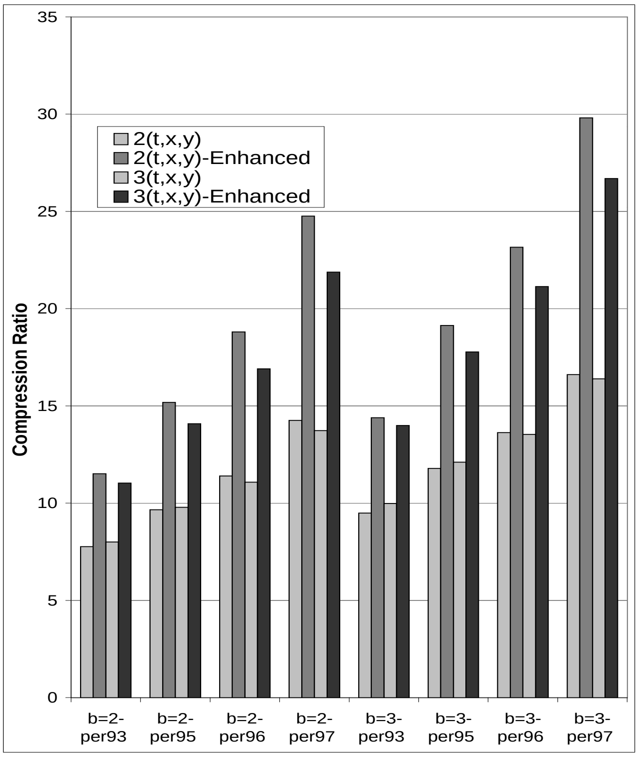

Fig. 2. Compression Ratios for Heart

2.3. Applying Arithmetic coding

In the previous approach, we have used the Huffman code in the entropy encoder. Arithmetic coding appears to be much more effective to complete our encoder, since it bypasses the idea of replacing an input symbol with a specific code. Instead, it replaces a stream of input symbols with a single floating-point output numbers [12]. In our encoder, the arithmetic coding has been implemented to work with only sixteen symbols in hexadecimal.

3. EXPERIMENTAL RESULTS

The evaluation has been carried out on a Intel Pentium-III 450 MHz bi-processor with 256 Mbytes of RAM, but we have only used one processor. The operating system was Linux 2.2.12-20smp. The entire video compressor and decompressor have been written in the C programming language.

We have compared the original 3D-FWT lossy compression method [3] with the proposal of this work, referred to as the enhanced method, on a $heart$ video medical sequence of 256 frames of $512x512$ pixels and a $hand$ video medical sequence of 256

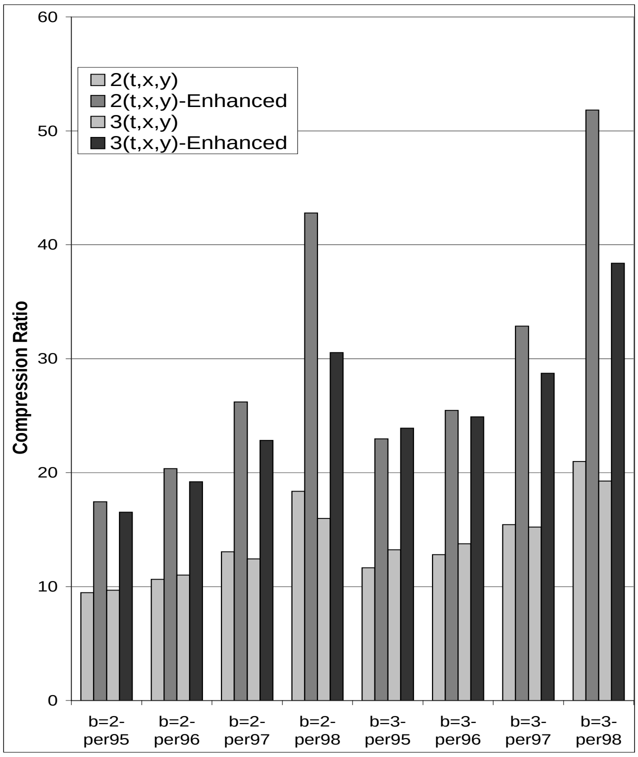

Fig. 3. Compression Ratios for Hand

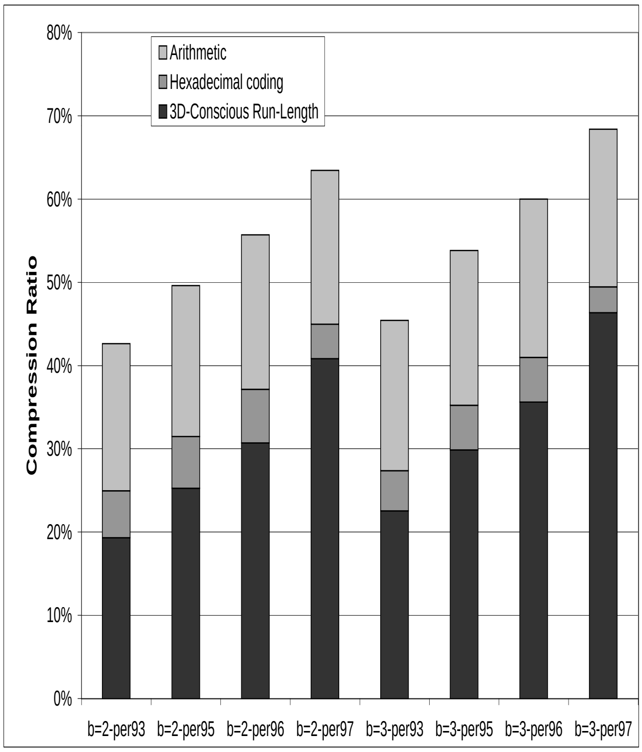

Fig. 4. Contribution of each Enhancement

frames of $256x256$ pixels. Both coded in gray scale (8 bit per pixel).

Figures 2 and 3 show the compression ratio achieved by the two wavelet-based encoders. Results are presented for two and three applications of the 3D-FWT, and considering percentiles ranging from 93 to 98 in the thresholding phase. PSNR (Peak Signal to Noise Ratio) is the same for both the original and the enhanced techniques for each configuration since the enhancements proposed in this work optimize the entropy encoder [1]. We can observe that the enhanced method clearly improves the compression ratio for all configurations. Also, as the number of discarded coefficients and the percentile increase, the difference between the original and the enhanced method increases significantly because the improvements aim to exploit the high presence of zeros. For instance, in the $heart$ sequence, choosing a 95-percentile, discarding 3 bits and applying two times the FWT, the compression ratio is increased 62%, whereas for a 97-percentile, discarding 3 bits and applying three times the FWT, the compression ratio is increased 79%. For the $hand$ sequence, the enhanced method obtains better results than the $heart$ sequence because the $hand$ sequence experience less interframe variations, which confirms the potential of the enhanced method with sequences that hardly have movement.

In Figure 4, we present the contribution to the overall performance improvement of each of the enhancements presented in this work for the $heart$ sequence. We can observe that the 3D-Conscious Run-Length is responsible for most of the achieved improvement. In fact, applying just this technique, the original compression ratio is increased between 20% and 40%. This supposes a contribution between 45% and 68% on the overall performance. This confirms that, if coefficients are processed taking into account the time dimension, there will be many long-length chains of zeros that will be effectively compressed, although Huffman is used as entropy encoder. The Hexadecimal coding provides a contribution that ranges from 5% to 13% on the overall improvement. Despite of the fact that this technique does not seem to provide significant benefits by itself, the absolute contribution (around 5%) is maintained in all configurations and it does not depend on the length of the chains of zeros. In addition, this enhancement benefits the subsequent application of Arithmetic code instead of Huffman in the entropy encoder. Finally, the use of Arithmetic code provides an absolute improvement of around 20%, better than some previous reported results, which show improvements ranging from 5% to 10% when Huffman code is replaced by Arithmetic code [15]. This is due to the synergistic effect obtained by previously applying Hexadecimal coding.

3.1. Comparison with MPEG2 and EZW

The video sequences have also been compressed using EZW [13], in the quantization process, and MPEG2. We force EZW, the original 3D-FWT coder and MPEG2 to obtain the same quality as the proposed enhanced 3D-FWT. Figure 5 shows the compression ratio for the $heart$ and $hand$ sequences for the four methods. For each video, two set of bars are shown. First set is obtained when the quality is forced to be excellent (PSNR around 41 for all frames). This implies that there are no differences between the original and the reconstructed sequences. On the other hand, the

[1] PSNR ranges from 41 to 35 for the $heart$ sequence and from 44 to 38 for the $hand$ sequence

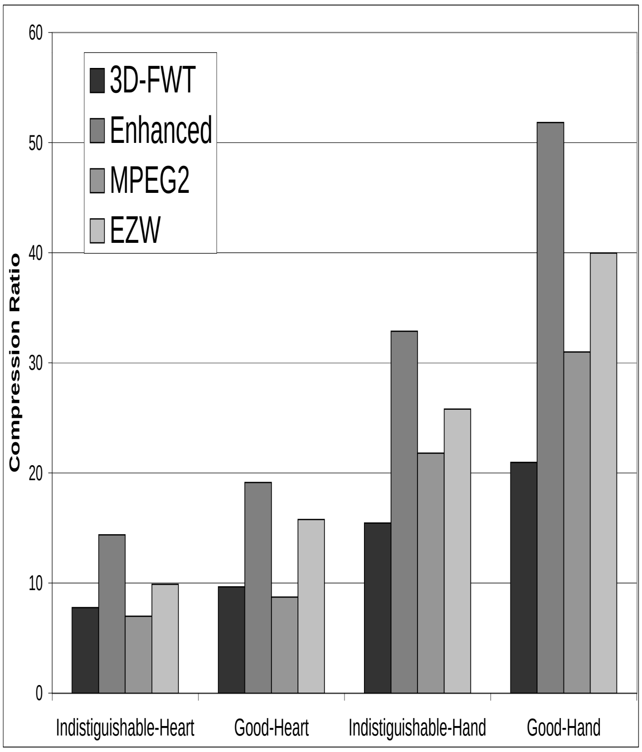

Fig. 5. Compression Ratios of 3D-FWT, Enhanced, MPEG2 and EZW for both sequences

second set is forced to have a good quality (PSNR of 38). First, we can observe that EZW outperforms both the original 3D-FWT coder and MPEG2 for both sequences and for both qualities. However, the enhanced 3D-FWT clearly outperforms the EZW, and thus MPEG2 for all configurations as well. We can observe that the 3D-FWT coder proposed in this work clearly outperforms previous coders, achieving speedups ranging from 51% to 119% for MPEG2 and 21% to 46% for EZW.

4. CONCLUSIONS

In this work, we have presented and evaluated several improvements to a compression scheme based on applying the 3D-FWT, focused on coding medical video. These improvements achieve better compression ratio, ranging from 40% to 70%, maintaining the good quality obtained in the original encoder. The trade-off achieved between compression ratio and quality is excellent, especially when compared with the compression ratio and quality achieved by the standard MPEG2 and the quantizer EZW, with no extra cost in computation time.

5. REFERENCES

[1] M. Antonini and M. Barlaud. Image coding using wavelet transform. *IEEE Transactions on Image Processing*, 1(2):205–220, April 1992.

[2] S. A. Basith and S. R. Done. Digital video, mpeg and associated artifacts. Technical report, Department of Computing and Electrical Engineering, Imperial College, June 1996.

[3] G. Bernabé, J. González, J. M. García, and J. Duato. A new lossy 3-d wavelet transform for high-quality compression of medical video. *Proc. of IEEE EMBS International Conference on Information Technology Applications in Biomedicine*, pages 226–231, November 2000.

[4] Y. Chen and W. A. Pearlman. Three-dimensional subband coding of video using the zero-tree method. *Proc. of SPIE-Visual Communications and Image Processing*, pages 1302–1310, March 1996.

[5] I. Daubechies. *Ten Lectures on Wavelets*. Society for Industrial and Applied Mathematics, 1992.

[6] M. L. Hilton, B. D. Jawerth, and A. Sengupta. Compressing still and moving images with wavelets. *Multimedia Systems*, 2(3), 1994.

[7] B.-J. Kim and W. A. Pearlman. An embedded wavelet video coder using three-dimensional set partitioning in hierarchical trees (sphit). *Proceedings of Data Compression Conference*, 1997.

[8] Y. Kim and W. A. Pearlman. Stripe-based sphit lossy compression of volumetric medical images for low memory usage and uniform reconstruction quality. *Proceedings of International Conference on Acoustics, Speech and Signal Processing*, 2000.

[9] A. S. Lewis and G. Knowles. Image compression using the 2-d wavelet transform. *IEEE Transactions on Image Processing*, 1(2):244–256, April 1992.

[10] S. Muraki. Approximation and rendering of volume data using wavelet transforms. *Proceedings of Visualization*, pages 21–28, October 1992.

[11] S. Muraki. Multiscale volume representation by a dog wavelet. *IEEE Transactions on Visualization and Computer Graphics*, 1(2):109–116, June 1995.

[12] M. Nelson and J.-L. Gailly. *The Data Compression Book*. M&T Books, 2 edition, 1996.

[13] J. M. Shapiro. Embedded image coding using zerotrees of wavelets coefficients. *IEEE Transcations on Signal Processing*, 41(12):3445–3462, December 1993.

[14] T. Sikora. *MPEG Digital Video Coding Standars*. McGraw Hill Company, 1997.

[15] S. W. Smith. *The Scientist and Engineer's Guide to Digital Signal Processing*. California Technical Publishing, 1997.