EFFICIENT 4D MOTION COMPENSATED LOSSLESS COMPRESSION OF DYNAMIC VOLUMETRIC MEDICAL IMAGE DATA

V. Sanchez, P. Nasiopoulos, and R. Abugharbieh

Department of Electrical and Computer Engineering, University of British Columbia, Vancouver, Canada

ABSTRACT

Dynamic volumetric (four dimensional- 4D) medical images are typically huge in file size and require a vast amount of resources for storage and transmission purposes. In this paper, we propose an efficient lossless compression method for 4D medical images that is based on a multi-frame motion compensation process employing a 4D search, variable block-sizes and bi-directional prediction. Data redundancies are reduced by recursively applying multi-frame motion compensation in the spatial and temporal dimensions. The proposed method also uses a novel differential coding algorithm to reduce redundancies in motion vectors and a new context-based adaptive binary arithmetic coder (CABAC) for compression of the residual data. Performance evaluations on real medical images of varying modality resulted in lossless compression ratios of up to 16:1.

Index Terms— 4D medical images, motion compensation, lossless compression, fMRI, CABAC.

1. INTRODUCTION

Recent advances in medical image technology have resulted in an increasing use of dynamic volumetric (four dimensional-4D) medical images for diagnosis and research purposes. Examples of such technologies include dynamic magnetic resonance imaging (4D-MRI), functional MRI (fMRI) and positron emission tomography (PET). 4D medical images consist of 3D images (volumes) captured at different time points, with each volume comprised of 2D images (slices). These data are usually huge in file size and pose a big burden on the resources needed to store them for future study and follow up. Furthermore, with the increasing use of telemedicine and the picture archiving and communications system (PACS), there is also a need to quickly transmit these data over limited bandwidth channels. Hence, it has become essential to design efficient lossless compression methods for storage and transmission of 4D medical images.

Since lossless compression of 4D medical images is still a relatively new area of research, 3D and 2D lossless compression algorithms are often used to compress volumes or slices independently. Current state-of-the-art 2D and 3D compression methods use mainly wavelet transforms or prediction coding to decorrelate the data and improve the compression performance [1,2]. However, these compression methods fail to exploit redundancies in all four dimensions. Few methods that exploit redundancies in all four dimensions have been proposed using either 4D wavelet transforms or 3D motion compensation [3,4]. However, the lossless compression ratios achieved by these approaches remain comparable to those achieved by state-of-the-art 3D compression techniques, such as 3D-JPEG2000, thus leaving room for much needed improvement. In this paper, we extend our initial ideas presented in [5] by proposing a novel lossless compression method that fully exploits redundancies in all dimensions of 4D medical images. Our new method decorrelates the data in the spatial and temporal dimensions by recursively applying a multi-frame motion compensation process that employs a full 4D search with variable block-sizes and bi-directional prediction. Redundancies in the resulting motion vectors are also reduced using a novel differential coding algorithm, while the residual data is losslessly compressed using a new context-based adaptive binary arithmetic coder. Performance evaluations show that our proposed method provides a significant improvement in compression ratio compared to current stat-of-the-art such as JPEG2000, 3D-JPEG2000 and H.264/AVC.

2. PROPOSED COMPRESSION METHOD

Our proposed compression method encodes a 4D medical image $I(x,y,z,t)$ in *four stages*, as illustrated in Fig. 1, where x and y denote the dimensions of a slice, z denotes the dimension of a volume and t denotes the temporal dimension. These four stages are described in detail next.

Stage I– first multi-frame motion compensation process: Redundancies between slices in the z and t dimensions are reduced by using multi-frame motion compensation (MF-MC) with a 4D search area. To achieve this, we divide an image $I(x,y,z,t)$ of dimensions $x=X$, $y=Y$, $z=S$ and $t=V$ into sub-images of dimensions $x=X$, $y=Y$, $z=S$ and $t=c$, where $c<V$, and encode the slices using bi-directional prediction. We process each sub-image separately by first scanning it in a raster order to create a single sequence of ($c\times S$) slices (see Fig. 2(a)), which are grouped into groups of slices (GOS) of size g=9 and coded as I- or P-frames or bi-directionally as B-frames in the coding order illustrated in Fig. 2(c). Our experiments on a large set of 4D medical images have shown that applying this coding order on GOS of size g=9 slices provides a superior compression performance. This coding order is repeated in each GOS. Only the first slice of each sub-image is coded as the reference slice,

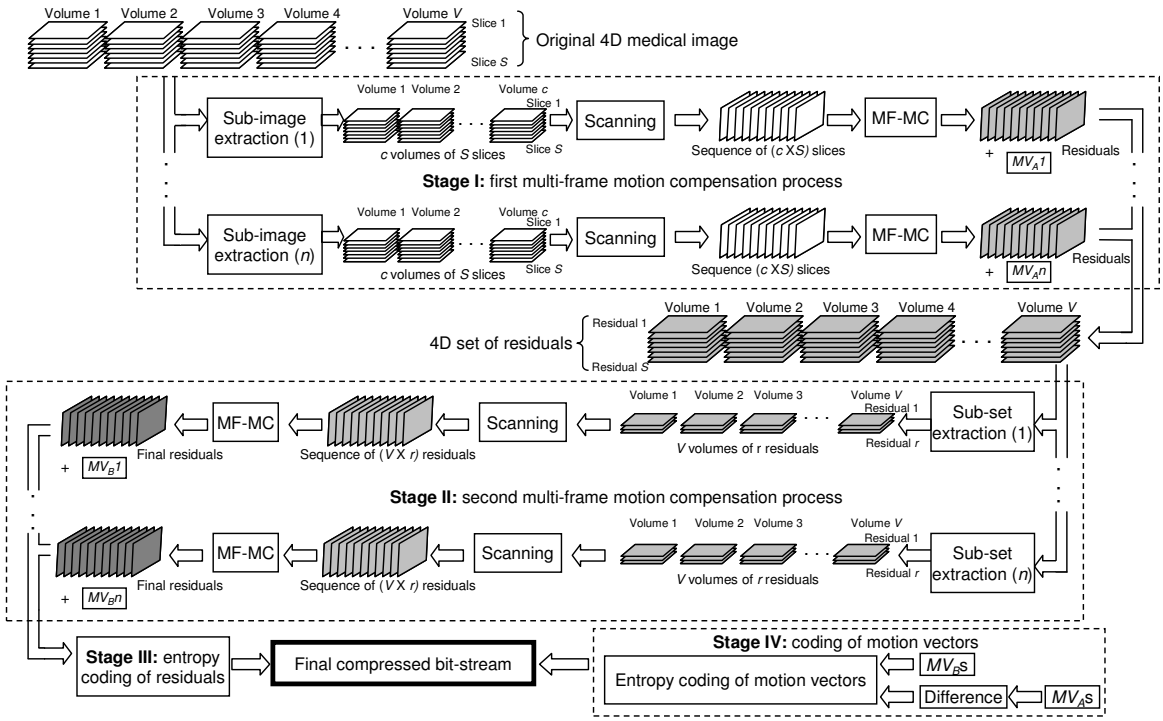


Fig. 1. Block diagram of the proposed lossless compression method. *MF-MC*: multi-frame motion compensation. MV_As: sets of motion vectors produced after MF-MC on sub-images. MV_Bs: sets of motion vectors produced after MF-MC on subsets of residuals.

or I-frame. Slices coded as P- and B-frames are predicted using MF-MC with macroblocks of 16×16 pixels, which may be further partitioned into smaller blocks of 16×8, 8×16 and 8×8 pixels. The block that yields the minimum sum of absolute differences with the current block is the one used for prediction. This first stage results in I-frames, residuals and their corresponding motion vectors. The I-frames and residuals are re-arranged back to the original order to create a 4D set of residuals, denoted as $R(x,y,z,t)$, of dimensions $x=X$, $y=Y$, $z=S$ and $t=V$.

Stage II– second multi-frame motion compensation process: Here, we reduce any remaining redundancies between the residuals generated in *Stage I* by using a second MF-MC process with a 4D search and variable block sizes. We first divide the set of residuals $R(x,y,z,t)$ into subsets of dimensions $x=X$, $y=Y$, $z=r$ and $t=V$, where $r<S$. We then process each subset separately by first scanning it in a raster order to create a single sequence of $(V\times r)$ residuals (see Fig. 2(b)), which are grouped into GOS's of size g=9 and coded as I-, P- or B-frames in the coding order illustrated in Fig. 2(c). This coding order is repeated in each GOS. This second stage results in the final residuals and their corresponding motion vectors.

Stage III– entropy coding of residuals: In order to efficiently compress the final residuals generated in *Stage II*, we have developed a new context-based adaptive binary arithmetic coder (CABAC) for lossless compression of medical images. In CABAC, the magnitudes of the residual values are first represented by a unique binary code. Each bit of these codes is then encoded using a binary arithmetic coding engine and a probability model. CABAC is optimized for lossy video

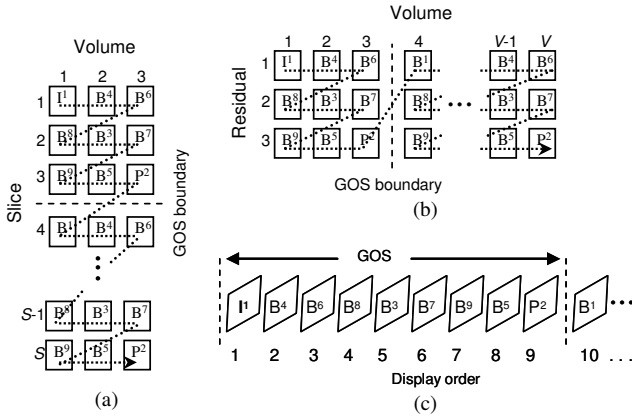


Fig. 2. (a) Scanning order of a sub-image of c=3 volumes of S slices and of (b) a subset of residuals of V volumes of r=3 residuals. (c) Sequence after scanning with GOS (Group of Slices) of g=9 slices. I: I-frame. P: P-frame B: B-frame. Superscript in each type of slice indicates coding order.

compression applications, where after quantization most of the magnitudes of the quantized residual values are smaller than 15 [6]. In our proposed lossless compression method no quantization is employed (i.e., the compression is fully lossless) and the magnitudes of the residuals values are thus likely to be larger than 15, for which CABAC has not been fully optimized. In order to improve its coding performance, we separate the residual values into two sets (type A and type B) according to their magnitude, R, as indicated below:

$$Type(residual\ value) = \begin{cases} A, & if\ R < 15 \\ B, & otherwise \end{cases} \quad (1)$$

We assign different binary codes to type-A and B residual values, and introduce an extra bit ('type-bit', hereafter) to identify the type of residual value, as illustrated in Table 1. The bits of the binary codes of type-A residual values are arithmetic coded using the probabilities models employed in [6]. Since the bits of the binary codes of type-B residual values are uniformly distributed (i.e., the probability of encountering a "1" or a "0" bit is roughly the same), we use a binary arithmetic coder designed to work with a uniformly distributed source to encode these bits.

Table 1. Binary codes for residual magnitudes in the proposed CABAC

Magnitude	Type		Binary code
			Unary code: applied to (magnitude – 1)
1	A	0	0
2	A	0	1 0
…	…	…	… … … … …
13	A	0	1 1 1 1 1 1 1 1 1 1 1 1 1 0
14	A	0	1 1 1 1 1 1 1 1 1 1 1 1 1 1 0
			2^{nd} *order Exp-Golomb code:* applied to (magnitude – 15)
15	B	1	0 0 0
16	B	1	0 0 1
17	B	1	0 1 0
…	…	…	… … …
Bit index		0	1 2 3 4 5 6 7 8 9 10 11 12 13 14 …

Bit with index 0, highlighted in grey, indicates the type of residual value.

In order to compress the type-bit, we use one of three probability models. These probability models exploit the correlation of the residual values, as residual values of the same type tend to appear in clusters. We employ the previously coded neighboring residual value to the left and on top of the current residual value to determine which model to use. When a neighbor is not available or is equal to zero, a type-A is assumed. Thus, the probability model for encoding the type-bit is selected as follows:

$$Model(type\text{-}bit) = \begin{cases} 1, & if\ Type(L) = A\ and\ Type(U) = A \\ 2, & if\ Type(L) \neq Type(U) \\ 3, & if\ Type(L) = B\ and\ Type(U) = B \end{cases} \quad (2)$$

where L is the immediate residual value to the left of the current residual value and U is the immediate residual value on top of the current residual value.

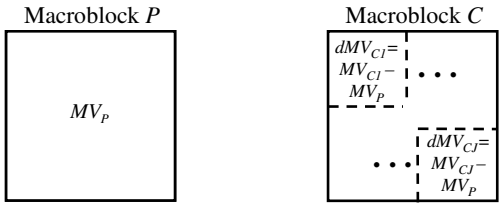
Stage IV– coding of motion vectors: The sets of motion vectors of two consecutive sub-images (see *MVA* in Fig. 1) as generated in *Stage I* are often highly correlated, since the sub-images usually depict the same anatomical ROI in time undergoing some structural or functional changes. We therefore propose a differential coding algorithm to exploit this correlation by calculating the difference between two consecutive sets of *MVA* motion vectors. Let C be the current macroblock of slice i of volume n of sub-image k, and let P be the previous macroblock in the same spatial position in slice i of volume n of sub-image $k-1$. For simplicity reasons, consider that only C is further partitioned in smaller blocks (Fig. 3(a)). Then, the differential motion vector of the *jth* partition of C (dMV_{Cj}) is the difference between the motion vector of the *jth* partition of C (MV_{Cj}) and the motion vector of P (MV_P):

$$dMV_{Cj} = MV_{Cj} - MV_P \quad (3)$$

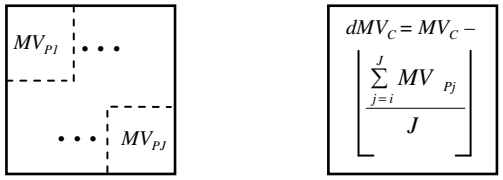
If only P is further partitioned in smaller blocks (Fig. 3(b)), dMV_C is then taken as the difference between MV_C and the average value of the motion vectors of all partitions of P:

$$dMV_c = MV_c - \left\lfloor \sum_{j=1}^{J} MV_{Pj} \Big/ J \right\rfloor \quad (4)$$

where MV_{Pj} is the motion vector of the *jth* partition of P located in the same spatial region as C, J is the total number of partitions in P and $\lfloor x \rfloor$ denotes the largest integer $\leq$ x.



a) P contains no partitions. C contains J partitions



b) P contains J partitions. C contains no partitions

Fig. 3. C: current macroblock in slice i of volume n of sub-image k. P: previous macroblock in the same spatial region as C in slice i of volume n of sub-image $k-1$. MV_C: motion vector of C. MV_P: motion vector of P. dMV_C: differential motion vector of C. $\lfloor x \rfloor$: largest integer $\leq x$.

The resulting dMV_Cs as well as the sets of *MVB* motion vectors (see Fig. 1) are then entropy coded using CABAC for motion vectors as described in [6].

3. RESULTS AND DISCUSSION

We tested our proposed compression method on real 4D medical images consisting of six fMRI, two 4D-MRI and two PET sequences. For comparison purposes, we have also losslessly encoded the 4D sequences using JPEG2000, 3D-JPEG2000 and H.264/AVC [7,8]. JPEG2000 and 3D-JPEG2000 are wavelet-based compression algorithms, while H.264/AVC is based on MF-MC. For the case of JPEG2000, we compressed each slice independently with two levels of decomposition. Since the highest redundancies are usually found in the temporal dimension t, we encoded the 4D sequences using 3D-JPEG2000 and H.264/AVC by first grouping together slices across the t dimension into sets and then processing each set as an individual 3D image in a manner

similar to our work in [5]. Therefore, the first set is comprised of all the first slices of each volume whereas the last set is comprised of all the last slices of each volume. For the case of 3D-JPEG2000 we first applied a one dimensional discrete wavelet transform (1D-DWT) across the *t* dimension of each set with the resulting transform slices being encoded using JPEG2000 with two levels of decomposition as suggested in [7]. For the case of H.264/AVC, we losslessly encoded each set as a monochrome video sequence using an IPPP coding structure and no quantization for residuals [8]. In our proposed compression method, we divided the 4D medical images into sub-images of c=3 volumes of S slices each (as described in *Stage I*), and the 4D set of residuals into subsets of V volumes of r=3 slices each (as described in *Stage II*). Our performance evaluations over a large set of 4D medical images have shown that sub-images and subsets of these sizes yield the best trade-off between coding efficiency and computational cost.

Table 2 shows the compression performance of our proposed method as well as that of JPEG2000, 3D-JPEG2000 and H.264/AVC. JPEG2000 achieves the lowest compression ratios as slices are encoded independently and the correlations in the *z* and *t* dimensions are not exploited. In 3D-JPEG2000, after applying the first 1D-DWT, most of the energy is concentrated in the first few slices, leaving the last slices with mostly zero coefficients which are usually better compressed using JPEG2000. H.264/AVC achieves lossless compression ratios comparable to those achieved by 3D-JPEG2000. Our proposed compression method outperforms all other compression schemes by a large margin, especially for the fMRI sequences. fMRI sequences are comprised of smooth slices with a high correlation in the *t* dimension. PET sequences, on the other hand, have little well defined structures, which makes them difficult to compress using block-based motion compensation. Structural data such as 4D-MRI depict changes in the structure of an anatomical ROI. If these changes are large and the number of volumes is low, the correlation in the *t* dimension decreases, affecting the compression ratio. The improvement on compression ratio achieved by our proposed method is constant across all three imaging modalities tested and is up to three times better compared to 3D-JPEG2000 and H.264/AVC.

Table 2. Lossless compression ratios of different 4D medical images of various modalities

4D Sequence (*volumes: slices per volume: pixels per slice: bits per pixel*)	Lossless compression method			
	JPEG2000	3D-JPEG2000	H.264/AVC	**Proposed method**
1. fMRI (150:36:128×12:12)	4.13:1	5.00:1	5.59:1	**16.52:1**
2. fMRI (33:60:256X256:12)	5.57:1	5.71:1	5.66:1	**14.59:1**
3. fMRI (126:36:128X128:12)	4.81:1	6.36:1	4.85:1	**13.12:1**
4. fMRI (195:36:128X12:12)	4.09:1	7.25:1	5.33:1	**15.60:1**
5. fMRI (33:60:256X256:12)	5.60:1	5.74:1	5.68:1	**15.03:1**
6. fMRI (33:60:256X256:12)	5.54:1	5.68:1	5.62:1	**14.67:1**
7. 4D-MRI (6:87:512X352:8)	2.09:1	2.12:1	2.41:1	**4.90:1**
8. 4D-MRI (6:87:512X352:16)	2.36:1	2.86:1	2.98:1	**4.24:1**
9. PET (6:93:128X128:8)	1.32:1	1.65:1	1.76:1	**2.84:1**
10. PET (9:93:128X12:16)	1.28: 1	1.47:1	2.05:1	**2.85:1**

4. CONCLUSIONS

We proposed a novel efficient lossless compression method for 4D medical images. Our method is based on a multi-frame motion compensation process that employs 4D search, variable block-sizes and bi-directional prediction. Correlations between slices in the spatial and temporal dimensions are exploited by recursively applying motion compensation. Correlations between motion vectors are exploited by employing a new differential coding algorithm. Residual data are compressed using a new context-based adaptive binary arithmetic coder specifically designed for lossless compression of medical images. Experimental results show compression ratios of up to three times those of state-of-the-art lossless compression techniques, such as 3D-JPEG2000 and H.264/AVC.

5. REFERENCES

[1] Z. Xiong, X. Wu, S. Cheng and J. Hua, "Lossy-to-lossless compression of medical volumetric images using three-dimensional integer wavelet transforms," *IEEE Trans. on Medical Imaging,* vol. 22, no. 3, pp. 459-470, March 2003.

[2] M. Benetiere, V. Bottreau, A. Collet-Billon and T. Deschamps, "Scalable compression of 3D medical datasets using a (2D+T) wavelet video coding scheme," *Sixth Int. Symposium on Signal Processing and its Applications,* vol. 2, pp. 537 - 540, Aug. 2001.

[3] L. Zeng, C.P. Jansen, S. Marsch, M. Unser, and P.R. Hunziker, "Four-dimensional wavelet compression of arbitrarily sized echocardiographic data," *IEEE Trans. on Medical Imaging,* vol. 21, no. 9, pp. 1179-1187, Sept. 2002.

[4] A. Kassim, P. Yan, and W. S. Lee, "Motion compensated lossy-to-lossless compression of 4-D medical images using integer wavelet transforms," *IEEE Trans. on Inf. Technology in Biomedicine,* vol. 9, no. 1, pp. 132-138, March 2005.

[5] V. Sanchez, P. Nasiopoulos and R. Abugharbieh, "Lossless compression of 4D medical images using H.264/AVC," *Int. Conf. of Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1116-1119, May 2006, Toulouse, France.

[6] D. Marpe, H. Schwarz and T. Wiegand, "Context-based adaptive binary arithmetic coding in the H.264/AVC video compression standard", *IEEE Trans. on Circuits and Systems for Video Technology,* vol. 1, no. 7, pp. 620-623, July 2003.

[7] Information Technology—JPEG 2000 Image Coding System—Part 2: Extensions, ISO/IEC 15 444-2, 2002.

[8] G. Sullivan, P. Topiwala and A. Luthra, "The H.264/AVC advanced video coding standard: overview and introduction to the fidelity range extensions", *Proc. SPIE Int. Soc. Opt. Eng.*, vol. 5558, pp. 454-474, August 2004.