

LOSSLESS AND NEAR-LOSSLESS MOTION-COMPENSATED 4D MEDICAL IMAGE COMPRESSION

Pingkun Yan Ashraf Kassim*

Department of Electrical and Computer Engineering,
National University of Singapore, 10 Kent Ridge Crescent, Singapore 119260
*ashraf@nus.edu.sg

ABSTRACT

Four-dimensional (4D) medical images, which are sequences of volume images over time, are represented by a great volume of information. One set of 4D medical images can easily take up hundreds of megabytes of storage volume. In this paper, we present a technique for compressing 4D medical images by combining motion compensation and lossless/near-lossless image encoding. Our technique incorporates a new fast 3D cube match algorithm that effectively exploits the redundancy between frames. The proposed scheme is validated by experiments on 4D cardiac images.

1 Introduction

Four-dimensional (4D) medical images are increasingly used in diagnosis, including 4D MRI and 4D CT. Without efficient compression, large amounts of data due to sequences of 4D medical images, would easily overwhelm storage and transmission systems. Lossless compression is desired by doctors for accurate diagnosis and treatment as well as to conform to legal and regulatory requirements. While there has been extensive work on lossy and lossless compression of still (2D) images [1, 2] and volume (3D) images [3, 4], the problem of 4D image compression is a relatively new research area [5].

There are mainly two categories of lossless image compression schemes [6]. One approach uses transformations while another involves predictive coding. In the former, wavelet transformation has been extensively used [3–5]. Although the wavelet transformation performs quite well in lossy and lossless compression, it results in PSNR fluctuations between decoding neighboring slices when applied on 3D or higher dimensional data in the lossy mode [7, 8]. In addition, transformation schemes normally are associated with heavy computations and so are not used as extensively as prediction based schemes in lossless medical image compression [1, 2, 6]. Currently, most predictive coding schemes work on 2D images, because computational complexity will be increased greatly if extended to 3D. However, this does not influence their applications since medical images are still viewed as 2D slices, although visualization techniques are being

extensively explored. Encoding in 2D enable the doctor to view a single slice at a time without decoding the whole volume. Taking into account these factors, we also adopted the prediction based scheme in our proposed 4D compression scheme.

Lossless data compression schemes often consist of two distinct and independent components: modeling and coding [1, 2]. The former is concerned with the ‘understanding’ of the source data, and is related to other knowledge based areas of computing such as machine learning and categorization techniques. In contrast, coding is a tightly specified task of efficiently representing a single symbol as a code, usually in binary form, given a set of estimated symbol probabilities. Our compression scheme incorporates 3D motion estimation as a preprocessing step to effectively remove the redundancy between frames and so reduce the corresponding prediction error. A new fast 3D cube match technique is used in the motion estimation process [7]. The Low Complexity Lossless Compression for Images (LOCO-I) [1] is a practical and general purpose lossless image coding technique which is efficient, easy to implement and also operates in a near-lossless mode. We applied LOCO-I on residual images obtained as a result of the 3D motion estimation process.

The rest of the paper is organized as follows. Section 2 describes our compression scheme for compressing 4D medical images. In section 3, we describe the fast 3D cube match algorithm. In Section 4, the compression results are reported and compared with the performance of other predictive coding algorithms for lossless and near lossless coding. We provide our conclusions in Section 5.

2 Compression Scheme

In our compression scheme, preprocessing is carried out to reduce the prediction cost and to improve the performance of subsequent operations. Using preprocessing, the voxel values are squeezed into a much smaller range but without loss of information to reduce the prediction cost in coding part. Voxel is the smallest unit in volume image, corresponding to pixel in 2D image. Motion compensation is introduced to exploit the redundancy between continuous frames of 4D medical images and hence to reduce the prediction cost in

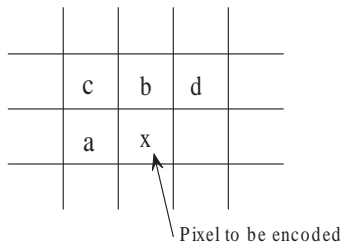


Fig. 1: LOCO-I prediction pattern.

later steps. Since each frame in the image sequence is a volume image, 3D motion may exist. The traditional 2D block match [9, 10] may not be able to exploit the redundancy efficiently. To reduce the computational loads, the fast 3D cube match algorithm [7] is used.

After the motion vectors are computed in the motion estimation process, the prediction frame can be constructed by filling with the sub-cubes of the reference frame according to corresponding 3D motion vectors. The residual image is obtained by subtracting the prediction frame from the actual one. The resulting 3D motion vectors and residual images are encoded.

The Huffman encoder is employed to encode 3D motion vectors. The idea behind Huffman coding is simply to use shorter bit patterns for more common characters, and longer bit patterns for less common characters. The residual images are divided into slices and then encoded separately by using LOCO-I [1] for its efficiency and simplicity, which is the algorithm at the core of the new ISO/ITU standard, JPEG-LS, for lossless and near-lossless compression of continuous-tone images.

The LOCO-I compression scheme basically involves: (i) run-length coding; (ii) non-linear prediction; (iii) context-based statistics modeling; and (iv) entropy coding. The modes of run-length coding and prediction-based entropy coding are selected by a template of four neighboring pixels as illustrated in Fig. 1. To reduce the prediction cost, LOCO-I proposed the following three delta values to implement the local texture analysis:

$$\Delta_1 = d - b; \Delta_2 = b - c; \Delta_3 = c - a$$

LOCO-I also provides a near lossless compression mode, in which the information distortion is introduced as a fixed, pre-determined constant value represented by a predefined parameter δ . At the decoder end, the reconstructed pixels have a maximum distortion of δ in comparison with their original values. By defining some acceptable distortion, much higher compression ratios can be achieved.

3 Fast 3D Cube Match Algorithm

The 3D cube match algorithm [7], an extension of the 2D block match [9, 10], which is extensively used in video compression, is used for motion estimation. In the proposed compression scheme, motion compensation is used to exploit the redundancy between 3D

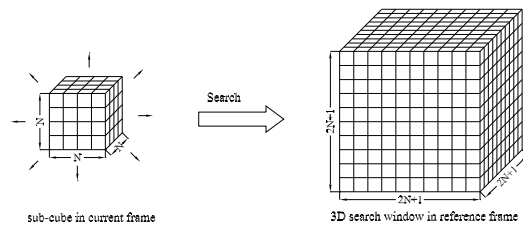


Fig. 2: Illustration of 3D cube match.

frames. To reduce computational complexity, a fast 3D cube match algorithm is used [7]. When doing fast 3D cube match, the current frame is divided into $N \times N \times N$ sub-cubes and a match is found for each sub-cube inside a 3D search window as shown in Fig. 2. Here, the 3D search window is taken from the previous 3D frame, which is the *reference frame*. The first frame in a sequence is selected as *key frame* and encoded directly without motion compensation. Motion estimation is carried out on subsequent frames by selecting their previous 3D frames as reference frames, respectively.

The motion vector (u, v, w) is essentially the difference between the position (i, j, k) of the sub-cube in current frame and the position (i, j, k) of the matched sub-cube in the reference frame. The matched sub-cube is defined as the one that has the minimum Mean Square Error (MSE) with the current sub-cube. The MSE is minimized to obtain the estimation of motion vector (u, v, w) . The 3D predicted frame can be constructed by filling its sub-cube at position (i, j, k) with the sub-cube at position $(i + u, j + v, k + w)$ in the 3D reference frame, where (u, v, w) is the corresponding motion vector. The 3D residual frame is the difference between the 3D predicted frame and the actual 3D intermediate frame.

If the full search method is used, the entire search window consisting of $(2N + 1)^3$ points need to be checked. To reduce the number of points to be searched and therefore the computational complexity, a fast 3D cube match strategy is used [7], which is extended from 2D center-biased search algorithms [9, 10]. Fig. 3 illustrates the three-step fast 3D cube match algorithm based on a $15 \times 15 \times 15$ 3D search window. For this instance, the full search will compute CDM at 3375 points, while the fast cube match algorithm checks only 105 points under the worst situation, which leads to a speed-up more than 32.

4 Experimental Results

In our experiments, we use a sequence of 16-bit CT cardiac volume images [11]. The size of each 3D frame is $128 \times 128 \times 107$ and there are a total of 15 frames in the sequence. Fig. 4(a) shows a 3D frame of the image.

Motion compensation is performed on the dataset first. The fast 2D block match [10] and the fast 3D cube match [7] algorithms are applied on the image sequence separately for comparison purposes. The 2D

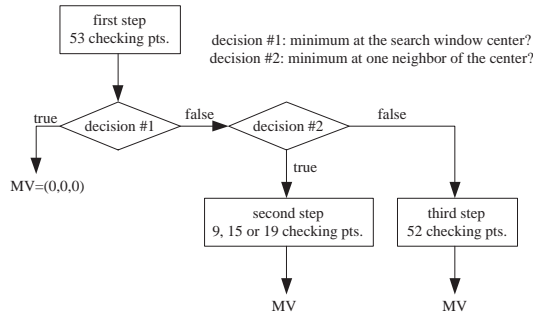


Fig. 3: The block diagram of the three-step 3D cube match algorithm.

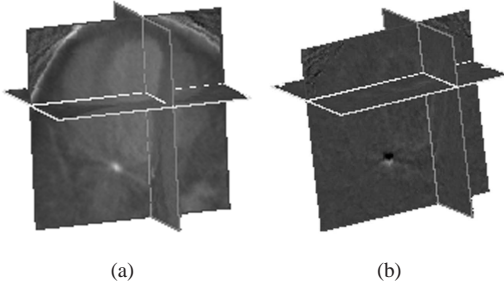


Fig. 4: One frame of the (a) original image and (b) residual image.

block match method is applied on slices of a 3D frame, while 3D cube match is used on a whole 3D frame directly. Prediction frames are reconstructed by using reference frame and motion vectors. *Peak signal-to-noise ratio* (PSNR) is employed to measure the quality of the results, which is defined as follows

$$\text{PSNR} = 10 \log_{10} \frac{M^2}{D}$$

where M is the maximum voxel value in the image. In addition, D is mean square error (MSE), which is defined as

$$D = \frac{1}{N} \sum_{n=0}^{N-1} |x_i - x'_i|^2$$

where N is the number of voxels in the frame, x_i and x'_i are the original and the predicted values at the i^{th} voxel respectively.

Fig. 5 shows that the PSNR of the reconstructed volume using 3D cube match is normally 2–4dB higher than that of those reconstructed by 2D block match. This naturally leads to the conclusion that 3D cube match algorithm is more suitable for motion compensation in this work than 2D block match.

Fig. 6 shows the histograms of original and residual images. After the motion compensation, the residual images are quite smooth (Fig. 6(b)). As shown in Fig. 4(b), textural information is present in residual images. This will facilitate the prediction process of the LOCO-I [1] and get more efficient coding results.

The proposed compression scheme has two compression modes. One mode is lossless compression, under which the compressed image can be reconstructed perfectly. The other mode is near-lossless

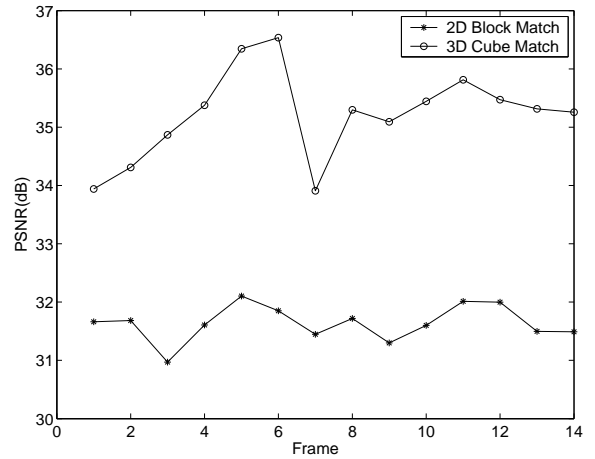


Fig. 5: PSNR of reconstructed images by using 2D block match and 3D cube match respectively.

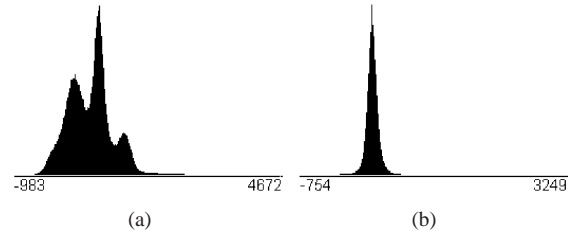


Fig. 6: Histogram of (a) the original image and (b) the residual image.

compression, under which every sample value in a reconstructed image is guaranteed to differ from the corresponding value in the original image by up to a small predefined amount δ . In fact, the lossless compression mode is just a special case of near-lossless compression, where $\delta = 0$. The near-lossless compression is useful in that the bits rate can decrease significantly with only slight loss of quality. The compression results for these two modes are discussed in the following subsections.

4.1 Lossless Compression Mode

In the proposed compression scheme, the compression results to be stored or transferred are encoded motion vectors and bit streams of the compressed residual images. The compression results under lossless mode are shown in Table 1 together with the compression results of several other algorithms for comparison. In Table 1, column 2 lists the actual bits/voxel obtained by using our compression scheme. Column 3 lists the compression result of LOCO-I. Columns 4 and 5 list the compression results obtained by the other two publicly available lossless image compression codecs: the context-based adaptive lossless image coding (CALIC) [2] and JPEG2000-LS [12]. The latter is an implementation of the JPEG2000 in lossless mode. Clearly the proposed compression scheme is more efficient than other compression schemes listed in Table 1 by decreasing bit rates about 20% to 30%.

The compression result of the first frame is the same

Table 1: Lossless compression results (bits/voxel)

Frame	Our Algorithm	LOCO-I	CALIC	JPEG-2000(LS)
1	8.286	8.286	9.589	8.993
2	5.970	8.235	9.512	9.463
3	5.807	8.249	9.415	9.369
4	5.830	8.261	9.370	9.223
5	5.817	8.239	9.396	9.310
6	5.804	8.240	9.415	9.299
7	5.750	8.225	9.483	10.655
8	5.813	8.245	9.435	9.373
9	5.826	8.246	9.435	9.274
10	5.813	8.271	9.447	9.338
11	5.822	8.271	9.432	9.227
12	5.802	8.250	9.333	9.119
13	5.812	8.267	9.428	9.022
14	5.791	8.266	9.473	9.029
15	5.818	8.261	9.526	8.998
Avg.	5.984	8.254	9.446	9.308

for our scheme and that of LOCO-I because motion compensation is applied from the second frame onwards.

4.2 Near-lossless Compression Mode

Table 2: Near-lossless compression results (bits/voxel)

Frame	$\delta=5$	$\delta=20$	$\delta=50$	$\delta=200$
1	6.919	4.738	3.016	0.995
2	5.134	3.327	2.123	0.794
3	5.115	3.289	2.111	0.632
4	5.126	3.305	2.113	0.786
5	5.103	3.276	2.089	0.716
6	5.089	3.275	2.089	0.783
7	5.092	3.272	2.088	0.771
8	5.126	3.305	2.112	0.764
9	5.119	3.284	2.110	0.760
10	5.094	3.277	2.103	0.763
11	5.110	3.291	2.103	0.780
12	5.097	3.274	2.093	0.788
13	5.102	3.287	2.106	0.718
14	5.113	3.290	2.103	0.796
15	5.118	3.301	2.105	0.605
Avg.	5.230	3.366	2.163	0.763

Table 2 lists the compression results under near-lossless mode with different δ values. The value of δ starts from 5 as shown in Table 2, because the compression ratio is very near to that of lossless compression if $\delta < 5$. Clearly, the compression ratio increases significantly when larger δ is used. It is noted, when $\delta=20$, the resulting bit rate is lower than half of that of lossless compression as shown in Table 1. This distortion may be acceptable because the image is stored in 16bits/voxel and its intensity varies between 0 and 65535. Therefore, it could be useful for compressing noisy images like ultrasound data sets, where lossless compression is not required [5, 8].

5 Conclusions

By applying 3D motion compensation before coding, we compress 4D medical images under lossless and near-lossless constraints efficiently. The fast 3D cube match algorithm results in reduced computations and thus faster compression. Experiments are carried out on 4D medical images in both lossless and near-lossless compression modes of the scheme. The experimental results show that our compression scheme is successful.

References

- [1] M. J. Weinberger, G. Seroussi, and G. Shapiro, "The LOCO-I lossless image compression algorithm: principles and standardization into JPEG-LS," *IEEE Trans. Image Processing*, Vol. 9, pp. 1309-1324, 2000.
- [2] X. Wu and N. D. Memon, "Context-based adaptive lossless image coding," *IEEE Trans. Commun.*, vol. 45, pp.437-444, 1997.
- [3] Y. S. Kim and W. A. Pearlman, "Lossless volumetric medical image compression," *Proc. of SPIE, App. of Digital Image Proc.*, vol. 3808, pp. 305-312, 1999.
- [4] J. Wang and H. K. Huang, "Medical image compression by using three-dimensional wavelet transformation," *IEEE Trans. Medical Imaging*, Vol. 15, pp. 547-554, 1996.
- [5] L. Zeng, C. P. Jansen, S. Marsch, M. Unser, and P. R. Hunziker, "Four-dimensional wavelet compression of arbitrarily sized echocardiographic data," *IEEE Trans. Medical Imaging*, Vol. 21, pp. 1179-1187, 2002.
- [6] N. D. Memon and X. Wu, "Recent developments in context-based predictive techniques for lossless image compression," *The Computer Journal*, Vol. 40, No. 2/3, pp. 127-136, 1997.
- [7] A. A. Kassim, P. Yan, W. S. Lee, and K. Sengupta, "Motion Compensated Lossy-to-Lossless Compression of 4D Medical Images Using Integer Wavelet Transforms," to appear in *IEEE Trans. Information Technology In Biomedicine*.
- [8] J. Reichel and G. Menegaz and M. Nadenau and M. Kunt, "Integer wavelet transform for embedded lossy to lossless image compression," *IEEE Trans. Image Processing*, vol. 10, pp. 383-392, Mar., 2001.
- [9] J. Y. Tham, S. Ranganath, M. Ranganath, A. A. Kassim, "A Novel Unrestricted Center-Biased Diamond Search for Block Motion Estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 8, pp 369-377, Aug. 1998.
- [10] R. Li, B. Zeng, and M. L. Liou, "A new three-step search algorithm for block motion estimation," *IEEE Trans. Circuits Syst. Video Technol.*, Vol. 4, pp. 438-442, 1994.
- [11] E. L. Ritman, R. A. Robb, and L. D. Harris, *Imaging Physiologic Functions: Experience with the Dynamic Spatial Reconstructor*. New York: Praeger, 1985.
- [12] ISO/IEC JTC1/SC29/WG1. *WG1N1523 JPEG 2000 Part I Committee Draft Version 1.0*. 1999.