Four-Dimensional Wavelet Compression of 4-D Medical Images Using Scalable 4-D SBHP

Ying Liu and William A. Pearlman
Center for Image Processing Research,
Electrical Computer and Systems Engineering Department,
Rensselaer Polytechnic Institute, Troy, NY

Abstract

This paper proposes a low-complexity wavelet-based method for progressive lossy-to-lossless compression of four dimensional (4-D) medical images. The Subband Block Hierarchial Partitioning (SBHP) algorithm is modified and extended to four dimensions, and applied to every code block independently. The resultant algorithm, 4D-SBHP, efficiently encodes 4D image data by the exploitation of the dependencies in all dimensions, while enabling progressive SNR and resolution decompression. The resolution scalable and lossy-to-lossless performances are empirically investigated. The experimental results show that our 4-D scheme achieves better compression performance on 4-D medical images when compared with 3-D volumetric compression schemes.

I. Introduction

Four-dimensional (4-D) data sets, such as images generated by computer tomography (CT) and functional Magnetic Resonance (fMRI) are increasingly used in diagnosis. Three-dimensional (3-D) volumetric images are two-dimensional (2-D) image slices that represent cross sections of a subject. Four dimensional (4-D) medical images, which can be seen as a time series of 3-D images, represent the live action of human anatomy and consume even larger amounts of resources for transmission and storage than 3-D image data. For example, a few seconds of volumetric CT image sequences require a few hundred mega-byte memory. Therefore, for modern multimedia applications, particularly in the Internet environment, efficient compression techniques are necessary to reduce storage and transmission bandwidth. Furthermore, it is highly desirable to have properties of SNR and resolution scalability with a single embedded bitstream per data set in many applications. SNR scalability gives the user an option of lossless decoding, which is important for analysis and diagnosis, and also allows the user to reconstruct image data at lower rate or quality to get rapid browsing through a large image data set. Resolution scalability can provide image browsing with low memory cost and computational resources.

Since 4-D image data can be represented as multiple 2-D slices or 3-D volumes, it is possible to code these 2-D slices or 3-D volumes independently. Many wavelet-based 2-D [5], [6], [4], [7] and 3-D [1], [8], [2], [9], [14] image compression algorithms have been proposed and applied on medical images. However, those 2-D and 3-D methods do not exploit the dependency among pixels in different volumes. Since the 4-D medical data is normally temporally smooth, the high correlation between volumes makes an algorithm based on four-dimensional coding a better choice.

Very little work has been done in the field of 4-D medical image compression. [10] use 4-D discrete wavelet transform and extended EZW to 4-D for lossy compression

the echocardiographic data. SPIHT was extended to 4-D and tested on fMRI and 4-D ultrasound images in [13]. These two algorithms are zerotree codecs and use symmetric tree structure. In [11], 4-D wavelet transform is applied on fMRI data, and the transformed slices are compression by JPEG2000 separately. [12] proposed a lossy-to-lossless compression method for 4-D medical images by using a combination of 3-D integer wavelet transform and 3-D motion compensation.

In this paper, we propose a low-complexity progressive lossy-to-lossless compression algorithm that exploits dependencies in all four dimension by using a 4-D discrete wavelet transform and 4-D coder. We extend SBHP [3], originally proposed as a low complexity alternative to JPEG2000 [7], to four dimensions. We have already reported on extension of SBHP to three dimensions and shown that this 3D SBHP is about 6 times faster in lossless encoding and 6 to 10 times faster in lossless decoding than Asymmetric Tree 3D-SPIHT [14]. This block-based algorithm has better scalability and random accessibility than zerotree coders. The 4D-SBHP is based on coding 4-D subblocks of 4-D wavelet subbands and can provide scalability and fast encoding and decoding.

In this paper, we will investigate the lossy-to-lossless performance and resolution scalability of 4D-SBHP in detail.

The rest of this paper is organized as follows. We present the scalable 4D-SBHP algorithm in Section 2. Experimental results of scalable coding are given in Section 3. Section 4 will conclude this study.

II. 4-D SBHP

A. Wavelet Decomposition in 4-D

For 4D datasets, the variances along the axial and temporal direction are determined by the thickness of slices and imaging speed. The variances among four dimensions may be very different. In general, the similarity of pixel values along the temporal direction is expected to be closer than along the other three directions, and similarity along the X and Y directions is very close. This asymmetric similarity has been shown in [11] for 4-D fMRI image data sets. Therefore, it is reasonable to apply transforms along the axial and temporal directions in different ways from the transforms along the X and Y directions in the 4D wavelet transform.

In our method, 2D spatial transformation, 1D axial transformation (along image slices) and 1D temporal transformation are done separately by first performing 1D dyadic wavelet decomposition in the temporal domain, followed by 1D dyadic wavelet decomposition along the axial direction and then 2D dyadic spatial decomposition in the XY planes. A heterogeneous selection of filter types and different amounts of decomposition levels for each spatial direction (x, y, z or t direction) are supported by this separable wavelet decomposition module. This allows for adapting the size of the wavelet pyramid in each direction in case the resolution is limited. Figure 1 shows a 4-D (x, y, z, t) data set after 2 levels of 4-D wavelet transform.

Because the number of volumes and slices in a typical 4-D data set can be quite large, it is impractical to buffer all volumes and slices for the temporal and axial transform. In our scheme, F consecutive slices in T consecutive volumes are collected into a Group Of Blocks (GOB) . For example, slices $S_{k,l}$, $k \in (0, F-1)$, $l \in (0, T-1)$ are in the same GOB. Each GOB is independently transformed and coded.

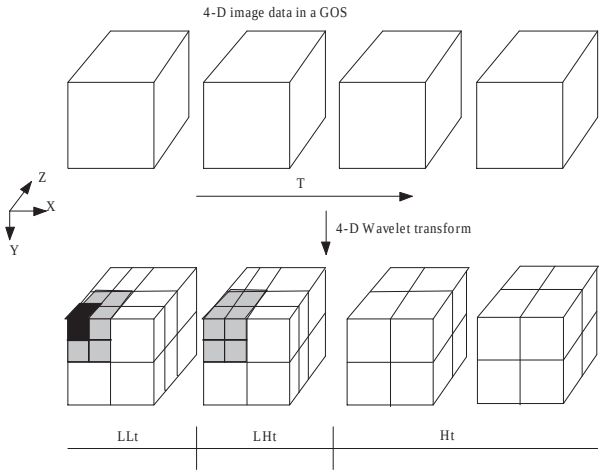


Fig. 1. Wavelet decomposition structure with 2 levels of 1D temporal transform followed by 2 levels of 1D axial transform and 2D spatial transform. The black block is the lowest frequency subband and the gray part marks the next higher frequency subbands.

B. Coding Algorithm

The 2-D SBHP algorithm is a SPECK[4] variant which was originally designed as a low complexity alternative to JPEG2000 [3]. 4-D SBHP is a modification and extension of 2-D SBHP to four dimensions. In 4-D SBHP, each subband is partitioned into 4-D code-blocks. All 4-D code-blocks have the same size. 4-D partitioning is applied to every code-block independently and generates a highly scalable bit-stream for each code-block by using the same form of progressive bit-plane coding as in SPIHT[5].

Consider a 4-D image data set that has been transformed using a discrete wavelet transform. The image sequence is represented by an indexed set of wavelet transformed coefficients $c_{i,j,k,l}$ located at the position (i, j, k, l) in the transformed image sequence. Following the idea in[6], for a given bit plane n and a given set τ of coefficients, we define the significance function:

$$S_n(\tau) = \begin{cases} 1, \text{ if } 2^n \leq \max\limits_{(i,j,k,l)\in\tau} |c_{i,j,k,l}| \leq 2^{(n+1)}, \\ 0, \text{ otherwise.} \end{cases} \tag{1}$$

Following this definition, we say that set τ is significant with respect to bit plane n if $S_n(\tau) = 1$. Otherwise, we say that set τ is insignificant.

In 4D-SBHP, each subband is partitioned into 4-D code-blocks with the same size. The 4D-SBHP algorithm makes use of sets referred to as sets of type S which can be of varying dimensions. The dimension of a set S depends on the dimension of the 4-D code-block and the partitioning rules. Because of the limited number of volumes and slices in a GOB, the dimensions along the temporal and axial directions of the 4-D code-block might be much shorter than dimensions along x and y directions. In our method, we set the 4-D code-block size to be $2^M \times 2^M \times 2^N \times 2^N$, $(M > N > 0)$, i.e., code-block size has equal dimensions along x and y directions, and equal dimensions along z and t directions. With these dimensions, the initial stages of partitioning result in some S sets that are 2D sets, i.e., temporal and axial dimension are both 1. We define *Max2D* to be the maximum 2D S set that can be generated. For a $2^M \times 2^M \times 2^N \times 2^N$ code-block,

the *Max2D* is the $2^{M-N} \times 2^{M-N} \times 1 \times 1$ set. 4D-SBHP always has S sets with at least $2 \times 2 \times 1 \times 1$ coefficients.

The size of a set is defined to be the cardinality C of the sets, i.e., the number of coefficients in the set. During the course of the algorithm, sets of various sizes will be formed, depending on the characteristics of coefficients in the code-block.

$$size(S) = C(S) \equiv |S| \tag{2}$$

4-D SBHP is based on a set-partitioning strategy. Below we explain in detail the 4-D partition rules by using a $64 \times 64 \times 4 \times 4$ 4-D code-block X as an example. Here size(*Max2D*) $= 16 \times 16 \times 1 \times 1$.

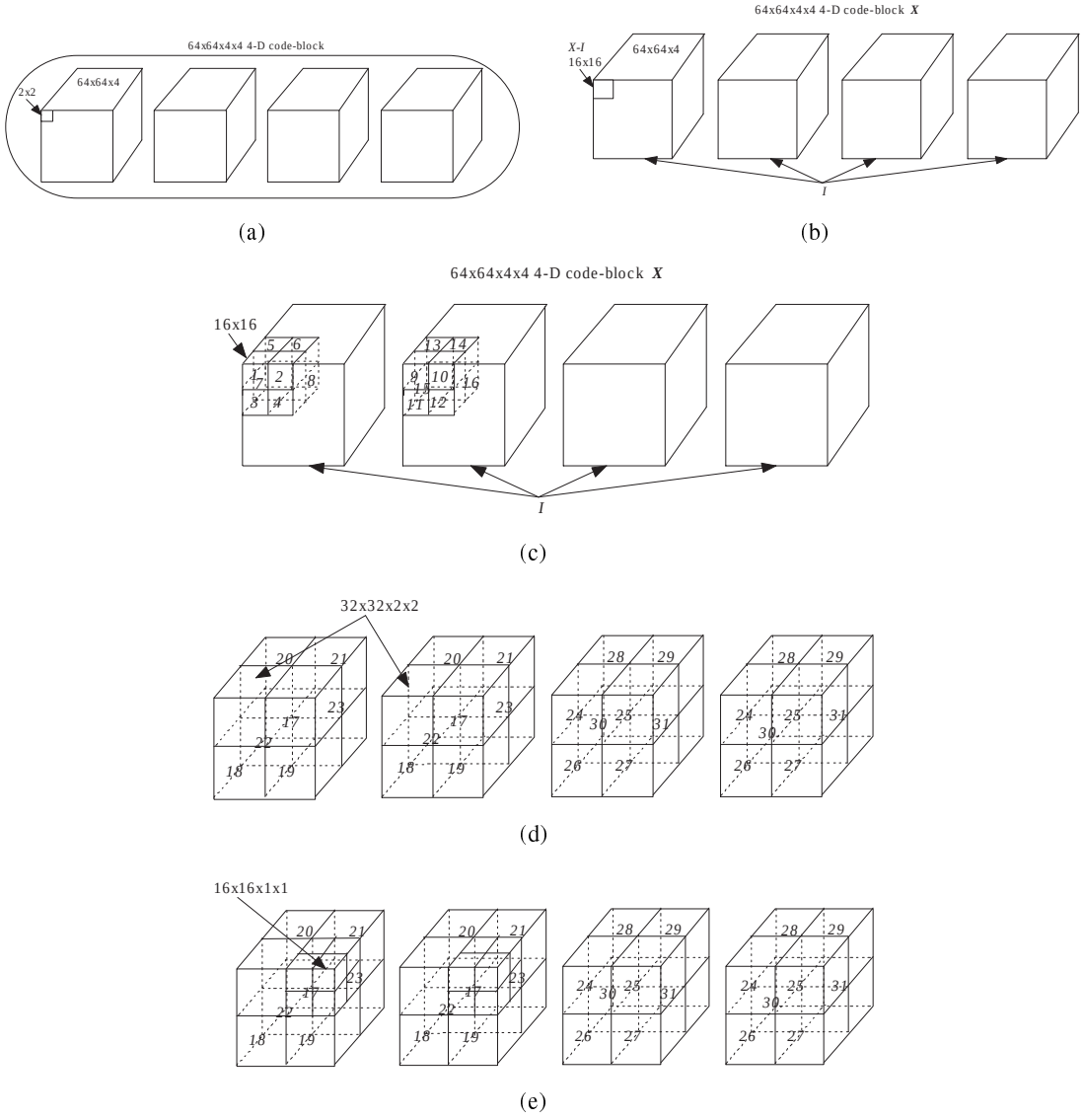


Fig. 2. Set partitioning rules used by 4-D SBHP.

The algorithm starts with two sets, as shown in Figure 2(a). One is composed of the S set: $2 \times 2 \times 1 \times 1$ top-left wavelet coefficients at the (0,0,0,0) position in X, and the

other is the I set which contains the remaining coefficients, $I = X - S$. The 4-D SBHP will work on a 2-D domain and follow exactly the octave band partition rules as in 2-D SBHP [3] until the size$(X - I)$ equal to size(*Max2D*), i.e. $16 \times 16 \times 1 \times 1$, as shown in Figure 2(b). In the next stage, the upper left $16 \times 16 \times 1 \times 1$ set at the (0,0,0,0) position in X will follow the 2-D quadrisection partition rules until all the significant coefficients be located. And the remaining I set will be partitioned into fifteen $16 \times 16 \times 1 \times 1$ S sets (labeled $2 - 16$ in Figure 2(c)) and one I set. The 2-D SBHP partition rule is applied on these $16 \times 16 \times 1 \times 1$ S sets to locate significant coefficients. At the following stage, the remaining I set is partitioned into fifteen $32 \times 32 \times 2 \times 2$ S sets, labeled $17 - 31$ in Figure Figure 2(d). The two $32 \times 32 \times 2$ 3-D blocks with the same label make one $32 \times 32 \times 2 \times 2$ 4-D block. At the next step, each $32 \times 32 \times 2 \times 2$ 4-D block will be partitioned into sixteen $16 \times 16 \times 1 \times 1$ blocks and 2-D SBHP partition rules will be applied on those 2-D blocks until all sets are partitioned to individual coefficients. Figure 2(e) shows this partition on block 17.

During the coding process a set is partitioned following the above rules when at least one of its subsets is significant. To minimize the number of significant tests for a given bit-plane, 3-D SBHP maintains three lists:

- LIS(List of Insignificant Sets) - all the sets(with more than one coefficient) that are insignificant but do not belong to a larger insignificant set.
- LIP(List of Insignificant Pixels) - pixels that are insignificant and do not belong to insignificant set.
- LSP(List of Significant Pixels) - all pixels found to be significant in previous passes.

Instead of using a single large LIS that has sets of varying sizes, we use an array of smaller lists of type LIS, each containing sets of a fixed size. All the lists and list arrays are updated in the most efficient list management method - FIFO. Since the total number of sets that are formed during the coding process remain the same, using an array of lists does not increase the memory requirement for the coder. Use of multiple lists completely eliminates the need for any sorting mechanism for processing sets in increasing order of their size and speeds up the encoding/decoding process. For each new bit plane, significance of coefficients in the LIP are tested first, then the sets in the LIS in increasing order of their sizes, and lastly the code refinement bits for coefficients in LSP. The idea behind processing LIP and LIS in increasing size can be seen as a way to achieve the same effect as the fractional bitplane coding used in JPEG2000 without having to go through several passes through the bitplane.

The way 4D-SBHP entropy codes the comparison results is an important factor that reduces the coding complexity. Instead of using adaptive arithmetic or Huffman coding, 4D-SBHP uses only three fixed Huffman codes in some special conditions. Since there are only four subsets or pixels after most sets are partitioned, they can be coded together. In 4D-SBHP, we choose a Huffman code with 15 symbols, corresponding to all the possible outcomes. The largest Huffman code is of length 6 bits. To speed up decoding, we can use lookup tables instead of binary trees to decode. No entropy coding is used to code the sign or the refinement bits. For these large datasets, this light use of entropy coding is a major factor in the low complexity and speed of the 4D-SBHP algorithm.

C. Scalable Coding

4D-SBHP is applied independently to every 4-D code-block inside a subband. An embedded bit stream is generated by bitplane coding and the method has the same effect as fractional bitplane coding [15]. To enable SNR scalability, bit stream boundaries are maintained for every bit plane, as shown in Figure 3. To get SNR scalability, bits belonging to the same fraction of the same bit planes in the the different code-blocks can be extracted for decoding.

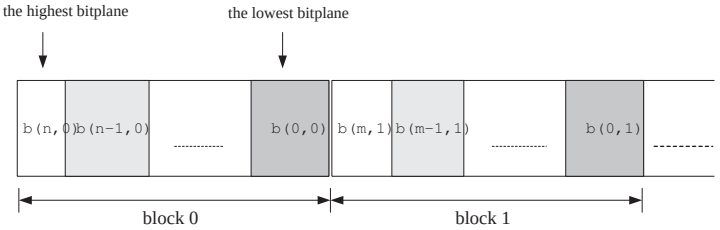


Fig. 3. Bitstream structure generated by 4D-SBHP. Each bitplane α in block β is notated as $b(\alpha, \beta)$.

In a wavelet coding system, resolution scalability enables increase of resolution when bits in higher frequency subbands are decoded. As shown in Figure 1, 4D-SBHP codes code-blocks in the black part first, then code-blocks in the gray part followed by code-blocks in the white subbands. After N levels of wavelet decomposition, the image has N resolution scales. As shown in Figure 4, 4D-SBHP codes code-blocks from lowest to the highest frequency subbands. The algorithm generates a progressive bit stream for each code-block, and the whole bit stream is resolution scalable. If a user wants to decode up to resolution n, bits belonging to the same fraction of the same bit planes in the code-blocks related to resolution n can be extracted for decoding.

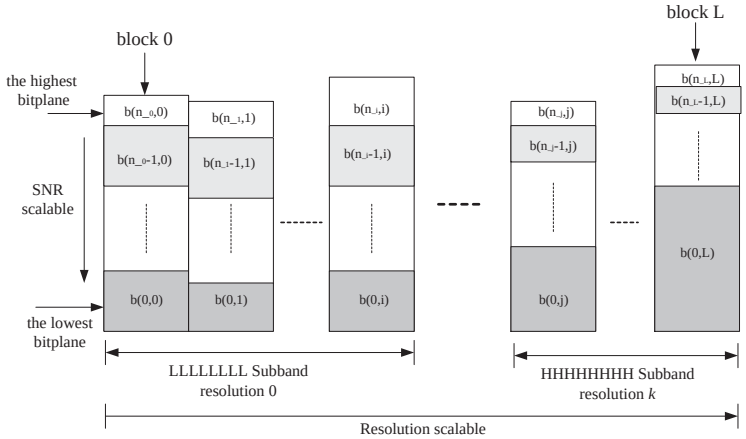


Fig. 4. An example of 4D-SBHP SNR and resolution scalable coding. Each bitplane α in block β is notated as $b(\alpha, \beta)$. Code-blocks are encoded and indexed from the lowest subband to the highest subband.

III. Numerical Results

We tested our algorithm on four fMRI , two 4D CT and one 4D ultrasound imaging datasets. Table I gives a brief description of these image datasets.

File Name	Image Type	Dimension (x,y,z,t)	Bit Depth (bit/pixel)	Resolution in 'xy' and z respectively (mm)
mb01	fMRI	$64 \times 64 \times 20 \times 100$	13	3.75 and 5
siem	fMRI	$64 \times 64 \times 16 \times 120$	8	unknown
feeds	fMRI	$64 \times 64 \times 20 \times 180$	13	3.75 and 5
3T	fMRI	$64 \times 64 \times 28 \times 244$	13	3.75 and 5
heart	ultrasound	$128 \times 128 \times 128 \times 12$	8	unknown
ct4d	CT	$256 \times 256 \times 256 \times 16$	8	unknown
4D_CT	CT	$512 \times 512 \times 108 \times 8$	13	0.787109 and 2.5

TABLE I

DESCRIPTION OF THE IMAGE VOLUMES

In this section, we provide simulation results and compare the proposed 4-D codec with 3-D volumetric algorithms.

A. Comparison of Lossless performance with 3-D and 4-D schemes

For 4-D datasets (x, y, z, t), we can employ 3D wavelet compression on either xyz cube or xyt cube. Table II compares the lossless compression performance of 4-D SBHP with 3-D SBHP applied on xyz cube and xyt cube. We get considerable compression improvement for 4D-SBHP as compared to 3D-SBHP on xyz and a small improvement as compared to 3D-SBHP on xyt. All results were obtained using three-level I(2,2) reversible transforms in the xy domain and one-level S transforms along the temporal and axial directions. The 4-D code-block size of $(32 \times 32 \times 2 \times 2)$ and GOS size $GOS(z,t) = (4,4)$ are chosen here. For 3D-SBHP, code-block size $(32 \times 32 \times 2)$ and GOS size $GOS = 4$ are chosen.

File Name	3D-SBHP on 'xyz' cube	3D-SBHP on 'xyt' cube	4D-SBHP
mb01	7.4732	6.8829	6.8196
siem	4.9947	4.7234	4.6187
feeds	5.4952	4.5045	4.4923
3T	10.3150	9.6163	9.4972
heart	2.0719	2.1728	2.0272
ct4d	3.2870	3.0830	2.8394
4D_CT	5.1953	5.0137	4.8667

TABLE II

LOSSLESS COMPRESSION PERFORMANCE USING 4D-SBHP AND 3D-SBHP (BITS/PIXEL)

Table III compares the lossless compression performance of 4D SBHP with 4D JPEG2000 [11], 4D EZW and 4D SPIHT [13]. Instead of applying wavelet transform on every GOB independently, these works apply 4D JPEG2000, 4D EZW and 4D SPIHT apply on 4D wavelet transform of the whole 4D dataset. For these three 4D methods [11], [13], the compression parameter settings such as code-block size and wavelet decomposition level are not mentioned. To get 4D SBHP results, $GOS(z,t) = (16,32)$, and wavelet decomposition level $(xy, z, t) = (3, 2, 5)$ are used for *mb01* and *siem* datasets. For *heart* image data, we use $GOS(z,t) = (32,8)$ and wavelet decomposition level$(xy, z, t) = (3, 2, 2)$. Code-block size $(64 \times 64 \times 4 \times 4)$ is used on all three datasets. As shown in the table, 4D-SBHP is roughly comparable to 4D JPEG2000, 4D-EZW and 4D-SPIHT in lossless performance, while having lower memory requirement and complexity.

File Name	4D JPEG2000	4D EZW	4D SPIHT	4DSBHP
mb01	5.962	5.984	5.749	6.0585
siem	4.056	4.331	3.933	4.1580
heart	1.68	2.013	1.735	1.9905

TABLE III

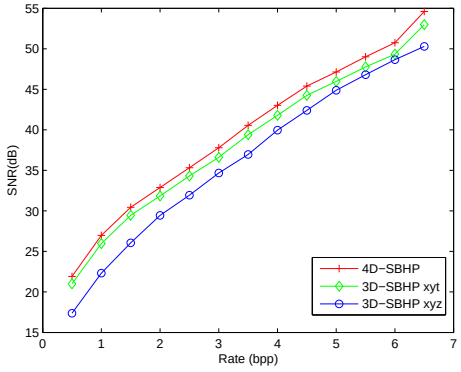
LOSSLESS COMPRESSION PERFORMANCE USING 4D METHODS (BITS/PIXEL)

B. Comparison of Lossy performance with 3-D schemes

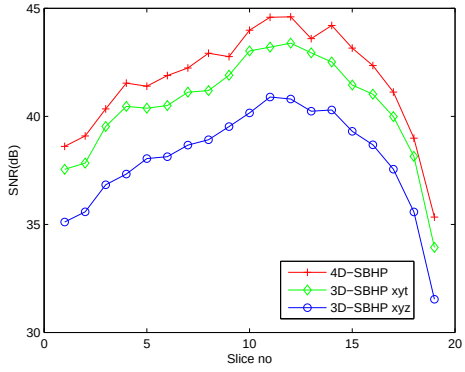
In this section, we show performance of lossy reconstruction from lossless compressed file. The quality of reconstruction is measured by Signal-to-Noise Ratio (SNR) over the whole 4-D image dataset. SNR is defined by

$$SNR = 10 \log_{10} \frac{P_x^2}{MSE} dB \tag{3}$$

where P_x^2 is the average squared value of the original medical images and MSE denotes the mean squared-error between the original and reconstructed slice. Figure 5(a) shows plots of Signal-to-Noise Ratio (SNR) versus bitrate over the whole $mb01$ image data. We also evaluated the lossy performance of our algorithm in 2-D as medical images are usually viewed slice by slice. Figure 5(b) shows the lossy results of the (x, y) slices at every z of the 4-D block at time $t = 8$, i.e., $(x, y, z, 8)$ for every z of $mb01$ at 3.5 bits/pixel. Figure 5 clearly shows that our 4-D scheme exploits the redundancy in all four dimensions and is superior to 3-D coding schemes.



(a) Lossy performance on mb01 image data



(b) Lossy performance of every slice in eighth time sequence of mb01 image data

Fig. 5. Comparison of lossy performance.

C. Resolution scalable results

The fMRI medical sequence *siem* , three-level I(2,2) reversible transform in xy domain and two-level S transform along temporal and axial directions are selected for this comparison. And GOS size $GOS(z, t) = (16, 16)$. Figure 6 shows the reconstructed *semi* images decoded from a single scalable code stream at a variety of resolutions at

0.25 bpp. The SNR values listed in Table IV for low resolution image sequences are calculated with respect to the lossless reconstruction of the corresponding resolution. Table IV shows that the SNR values increase from one resolution to the next lower one.

Bit Rate	SNR (dB)	
	1/2 resolution	FULL
0.0625	23.722	12.347
0.125	28.817	13.356
0.25	37.80	15.187
0.5	lossless	18.716
1	lossless	21.985
2	lossless	25.883

TABLE IV

SNR FOR DECODING SIEM AT A VARIETY OF RESOLUTIONS AND BIT RATES

Fig. 6. A visual example of resolution scalable decoding. Full resolution and 1/2 resolution of one slice at 0.25 bpp

IV. SUMMARY AND CONCLUSIONS

In this paper, we proposed an image coding algorithm, 4D-SBHP, for lossy-to-lossless compression of 4-D medical images using 4-IWT and 4-D set-partition. This block-based algorithm supports resolution scalability. Fixed Huffman coding and one coding pass per bit plane are used to reduce the coding time. The experimental results show that 4D-SBHP exploits the redundancy in all four dimensions and has higher compression ratio than 3-D compression schemes. Furthermore, 4D-SBHP is low in complexity and exhibits fast encoding and decoding times.

ACKNOWLEDGMENTS

We gratefully acknowledge that this work was jointly sponsored by the Office of Naval Research and DARPA as part of the Dynamic 3-D Digital Structure Program. under Grant No. N00014-05-1-0507. We also thank Professor Ali Bilgin for providing 4D datasets used in this work.

REFERENCES

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

[2] A. Bilgin, G.Zweig, and M.W. Marcellin, "Three-dimensional image compression with integer wavelet transform", *Applied Optics*, Vol. 39, No.11, April. 2000.

[3] C. Chysafis, A. Said, A. Drukarev, A. Islam, and W.A Pearlman, "SBHP - A Low complexity wavelet coder", *IEEE Int. Conf. Acoust., Speech and Sig. Proc. (ICASSP2000)*, vol. 4, PP. 2035-2038, June 2000.

[4] A. Islam and W.A. Pearlman, "An embedded and efficient low-complexity hierarchical image coder", *in Proc. SPIE Visual Comm. and Image Processing*, Vol. 3653, pp. 294-305, 1999.

[5] J.M. Shapiro, "Embedded image coding using zerotrees of wavelet coefficients", *IEEE Trans. Image Processing*, Vol. 41, pp. 3445-3462, Dec. 1993.

[6] A. Said and W.A. Pearlman, "A new, fast and efficient image codec based on set-partitioning in hierarchical trees", *IEEE Trans. on Circuits and Systems for Video Technology*, Vol. 6, pp. 243-250, June 1996.

[7] D. Taubman, "High performance scalable image compression with EBCOT", *IEEE Trans. on Image Processing*, Vol. 9, pp. 1158-1170, July 2000.

[8] S. Cho, D. Kim, and W. A. Pearlman, "Lossless compression of volumetric medical images with improved 3-D SPIHT algorithm", *Journal of Digital Imaging*, Vol. 17, No. 1, pp. 57-63, March 2004.

[9] P. Schelkens, J. Barbarien, and J. Cornelis, "Compression of volumetric medical data based on cube-splitting", *in Applications of Digital Image Processing XXIII*, Proc. of SPIE 4115, pp. 91-101, San Diego, CA, July 2000.

[10] 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, Sep. 2002.

[11] H.G.Lalgudi, A.Bilgin, M.W.Marcelllin, A.Tabesh, M.S.Nadar, and T.P.Trouard, "Four-dimensional compression of fMRI using JPEG2000", in *Proc. SPIE International Symposium on Medical Imaging*, Feb, 2005.

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

[13] H.G.Lalgudi, A.Bilgin, M.W.Marcelllin, A.Tabesh, M.S.Nadar, and T.P.Trouard, "Compression of fMRI and Ultrasound images using 4D SPIHT," in *Proceedings of 2005 International Conference on Image Processing*, Genova, Italy, September 2005.

[14] Y. Liu and W. A. Pearlman, "Scalable three-dimensional SBHP algorithm with region of interest access and low complexity," *Applications of Digital Image Processing XXIX* , Proc. SPIE Vol. 6312, pp. 631209-1–11, Aug. 2006.

[15] W.A. Pearlman, A. Islam, N. Nagaraj, and A. Said, "Efficient, low-complexity image coding with a set-partitioning embedded block coder", *IEEE Trans. on Ciruits and Systems for Video Technology*, Vol. 14, No. 11, pp. 1219-1235, Nov. 2004.