Four-Dimensional Wavelet Compression of Arbitrarily Sized Echocardiographic Data

Li Zeng*, Christian P. Jansen, Stephan Marsch, Michael Unser, *Fellow, IEEE*, and Patrick R. Hunziker

***Abstract*—Wavelet-based methods have become most popular for the compression of two-dimensional medical images and sequences. The standard implementations consider data sizes that are powers of two. There is also a large body of literature treating issues such as the choice of the "optimal" wavelets and the performance comparison of competing algorithms. With the advent of telemedicine, there is a strong incentive to extend these techniques to higher dimensional data such as dynamic three-dimensional (3-D) echocardiography [four-dimensional (4-D) datasets]. One of the practical difficulties is that the size of this data is often not a multiple of a power of two, which can lead to increased computational complexity and impaired compression power.**

Our contribution in this paper is to present a genuine 4-D extension of the well-known zerotree algorithm for arbitrarily sized data. The key component of our method is a one-dimensional wavelet algorithm that can handle arbitrarily sized input signals. The method uses a pair of symmetric/antisymmetric wavelets (10/6) together with some appropriate midpoint symmetry boundary conditions that reduce border artifacts. The zerotree structure is also adapted so that it can accommodate noneven data splitting. We have applied our method to the compression of real 3-D dynamic sequences from clinical cardiac ultrasound examinations. Our new algorithm compares very favorably with other more ad hoc adaptations (image extension and tiling) of the standard powers-of-two methods, in terms of both compression performance and computational cost. It is vastly superior to slice-by-slice wavelet encoding. This was seen not only in numerical image quality parameters but also in expert ratings, where significant improvement using the new approach could be documented. Our validation experiments show that one can safely compress 4-D data sets at ratios of 128:1 without compromising the diagnostic value of the images. We also display some more extreme compression results at ratios of 2000:1 where some key diagnostically relevant key features are preserved.

***Index Terms*—Echocardiography, image coding, multidimensional images, zerotree algorithm.**

I. Introduction

In medical image processing, there is an abundance of papers dealing with image-compression algorithms, most of which are based on wavelets [1]–[7]. The large majority of them work with two-dimensional (2-D) and three-dimensional (3-D) datasets [4], [5], and usually with power-of-two image size. In practice, however, higher dimensional image datasets, e.g., moving 3-D data, are increasingly used in medical imaging, and often, they come in sizes that are not a multiple of a power of two. The latter point may matter especially in higher dimensions, as four-dimensional (4-D) data extension by some linear factor x may increase the size of the data (and thus the computing time and hardware requirements) by a factor of x^4. This raises the questions of whether compression algorithms optimal for 2-D and 3-D are good enough for 4-D and if simple extension of power-of-two algorithms to arbitrarily sized data is adequate. In a preliminary study, we had already found that genuine 4-D wavelet compression is feasible and yields better results than lower dimensional approaches [8].

The well-known zerotree algorithm [11] is an improvement over simple wavelet compression but is mostly done in 2-D; an extension to volumetric 3-D, termed "octave zerotree," has been described in [9] and [10], although results were only reported for power-of-two data. The goal of this paper is to develop a practical wavelet compression algorithm for (3-D + time) echocardiographic data, and, in particular, to extend Shapiro's well-known embedded zerotree algorithm [11] to a genuine 4-D method. While doing so, the major practical difficulty we had to overcome was to come up with an algorithm that could correctly handle 4-D data sets, irrespective of their size.

Section II describes our new wavelet algorithm, which has the ability to process arbitrarily sized data. It also justifies our design choices and describes the extension of the zerotree algorithm to our particular framework. In Section III, we present experiments using real datasets from dynamic 3-D echocardiography. Initially, we examined different strategies to handle arbitrary-size data in a zerotree with respect to image quality, requirements on hardware, computing time, and software complexity. However, evident problems with these algorithms led us to the development of the genuine 4-D zerotree algorithm that is presented here. Finally, we compared the image-compression performance of this optimized 4-D algorithm to a slice-by-slice or volume-by-volume application of 2-D and 3-D zerotree wavelet encoding.

II. Methods

A. Wavelet Decomposition and Reconstruction in 4-D

The 4-D wavelet transform is implemented in a separable fashion by successive one-dimensional (1-D) transformation along the x, y, z, and t dimensions of the data. The 1-D decomposition amounts to splitting a signal of length N into two subsequences of length $N/2$: the lower resolution signal

Manuscript received November 8, 2001; revised July 6, 2002. The work of L. Zeng and C. P. Jansen was supported by a grant from the Swiss National Science Foundation. The work of P. R. Hunziker was supported by the Swiss Heart Foundation under a Grant. *Asterisk indicates corresponding author.*

*L. Zeng was with the University Hospital of Basel, Switzerland. He is now with the Department of Applied Math, Chongqing University, 400044 Chongqing, China (e-mail: lizeng@cta.cq.cn).

P. R. Hunziker, S. Marsch, and C. P. Jansen are with the University Hospital of Basel, CH 4031 Basel, Switzerland.

M. Unser is with the Biomedical Imaging Group, Swiss Federal Institute of Technology, CH-1015 Lausanne, Switzerland.

Digital Object Identifier 10.1109/TMI.2002.804424

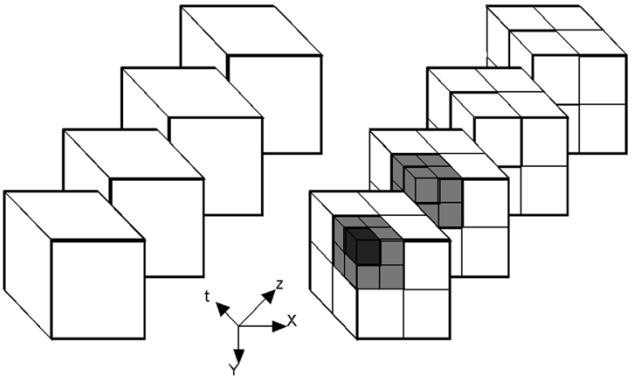

Fig. 1. 4-D (x, y, z, t) dataset after a first 4-D wavelet transform (low-pass elements light gray) and a second wavelet transform (low-pass elements dark gray).

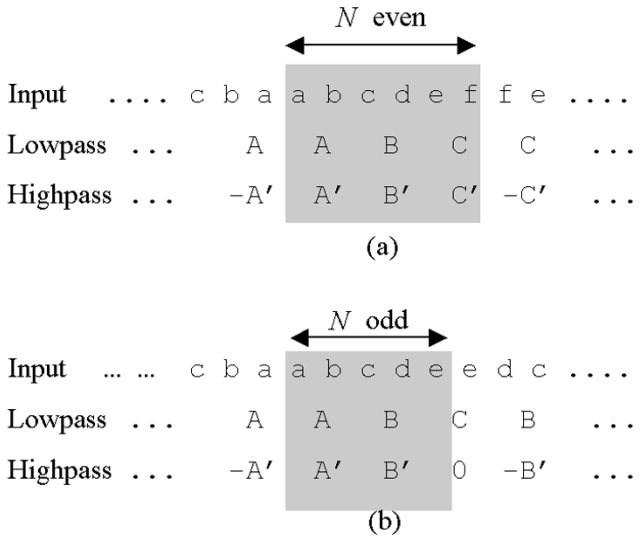

Fig. 2. Illustration of the signal extension rules for wavelet decomposition using Haar-type even-length filters. In the case of the Haar transform, one has that $A = a + b$, $C = c + d$, etc., in the low-pass channel and $A' = a - b$, $B' = c - d$, etc., in the high-pass channel. The input and the subsampled low-pass channel are extended symmetrically, while the high-pass channel is extended in an antisymmetric fashion. The nonstandard case is when the size of the input is odd (b), which results in a center- (anti-) symmetric extension on the right-hand side, as opposed to midpoint symmetries in all other cases.

approximation (low-pass component) and the wavelet (or high-pass) component. The splitting process is repeated on the lower resolution version of the signal until a predefined decomposition level is reached. In our implementation, the level of decomposition (number of scales) is specified independently for each dimension according to the data length available. The basic structure of the 4-D wavelet transform is shown in Fig. 1. For the inverse algorithm, the splitting is replaced by a 1-D merging process and the same sequence of operations applied but in the reverse order, making the decomposition and reconstruction algorithms flowgraph transposes of each other.

The difficulty in this application is that the 4-D echocardiograms do usually not come in sizes that are powers of two. This led us to develop a 1-D wavelet algorithm that could handle input signals of arbitrary length without any additional data storage. Thanks to this method, we were able to correctly decompose our data with any desired number of scales. Specifically, we considered data sets of size $240 \times 216 \times 18 \times 18$, applying four levels of decomposition in each of the dimensions.

B. 1-D Algorithm for Arbitrary Sized Data

A wavelet transform uses a pair of analysis filters $\tilde{H}(z)$ (low-pass) and $\tilde{G}(z)$ (high-pass). The reconstruction algorithm uses the complementary filters $H(z)$ (refinement) and $G(z)$ (wavelet filter). These four filters define a perfect reconstruction filterbank. In the biorthogonal case, the system is entirely specified by the low-pass filters $\tilde{H}(z)$ and $H(z)$, which form a biorthogonal pair; the high-pass operators are obtained by simple shift and modulation: $\tilde{G}(z) = zH(-z)$ and $G(z) = z^{-1}\tilde{H}(-z)$.

Our approach works for even-length filters that are either symmetric or antisymmetric with respect to their center (midpoint symmetry). Since the filters must include a minimum number of regularity factors, we can assume without loss of generality that the analysis filters are such that

$$\tilde{H}(z) = \tilde{H}_s(z) \cdot \frac{1+z}{\sqrt{2}} \quad \tilde{G}(z) = \tilde{G}_s(z) \cdot \frac{1-z}{\sqrt{2}} \qquad (1)$$

where the odd-length factors $\tilde{H}_s(z)$ and $\tilde{G}_s(z)$ are symmetric with respect to the origin; i.e., $\tilde{H}_s(z) = \tilde{H}_s(z^{-1})$ (central point symmetry). In other words, we have expressed the decomposition process as a symmetric filtering operation followed by a Haar transform, which is the simplest transform to provide the required midpoint symmetry; i.e., symmetry for the low-pass and anti-symmetry for the high-pass.

When applying this type of filter to finite-length signals, one usually implements midpoint mirror symmetric boundary extensions [12], as shown in Fig. 2(a). This has the desirable feature of reducing border artifacts. In the standard setup where the length of the input signal is even, it also yields boundary conditions that are consistent across all scales. The approach is entirely reversible because there is a corresponding symmetry rule that allows one to extend the subsampled low-pass and high-pass components based on the values within the interval [see Fig. 2(a)]. Note that the boundary conditions for the Haar transform are directly transposable to the more general case because the symmetric factors $\tilde{H}_s(z)$ and $\tilde{G}_s(z)$ in (1) preserve the boundary conditions of the input.

Here, we have extended this approach to the case where the size of the input signal is odd. This is possible because in the odd case, the very last wavelet coefficient happens to be zero always, as shown in Fig. 2(b). Thus, we are able to split a sequence of length $2N+1$ into $N+1$ low-pass coefficients plus N wavelet coefficients, thus fitting the transformed data into the available storage space while preserving perfect reconstruction (one-to-one transform). It is then easy to reconstruct the original signal exactly by applying the inverse wavelet transform to the low-pass and high-pass coefficients after extending them outside the interval using the symmetry rules in Fig. 2(b).

Another more standard solution would be data expansion to the next multiple of a power-of-two, but this incurs additional cost in terms of memory and computing time, especially in higher dimensions.

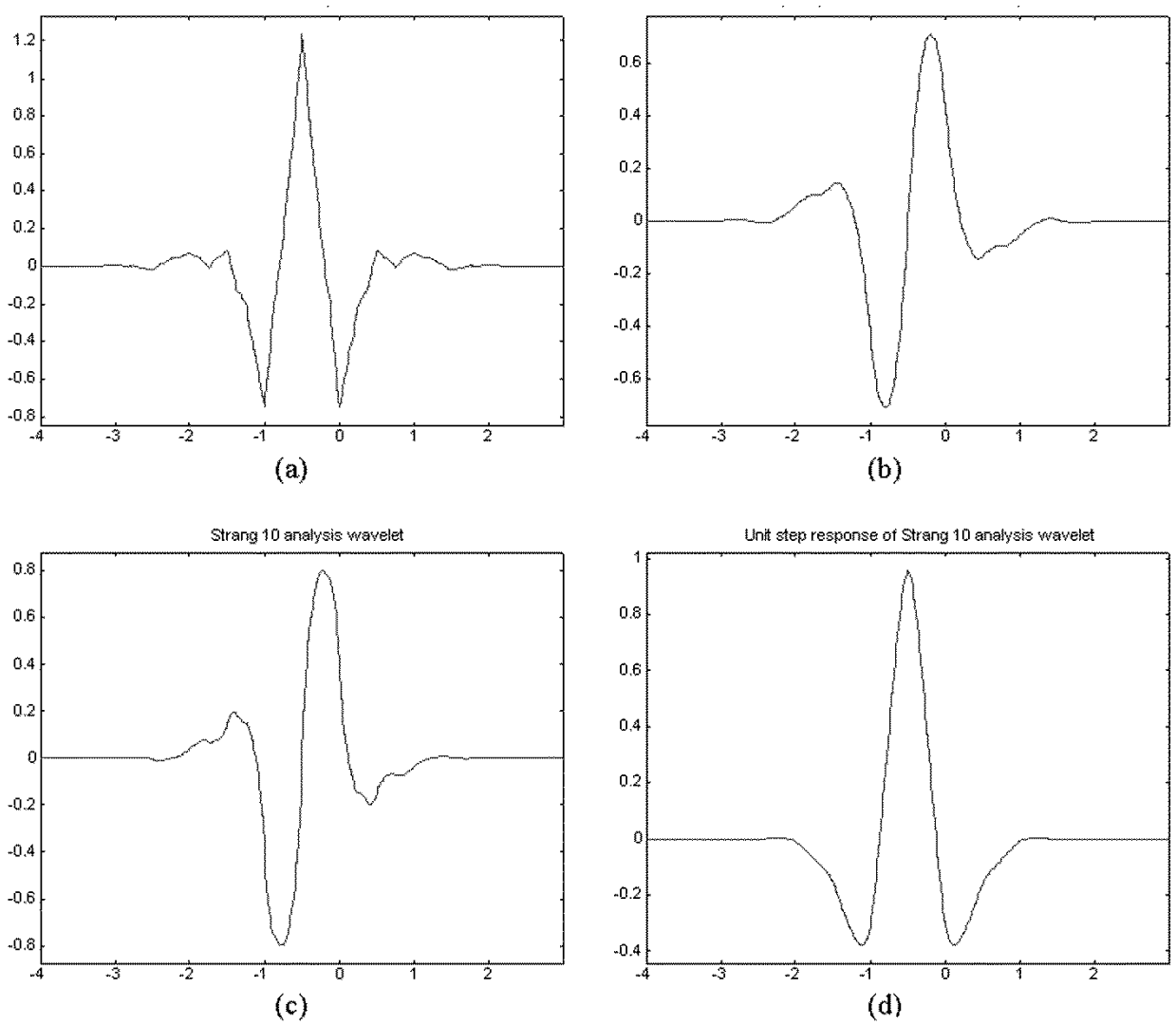

Fig. 3. Comparison of the 9/7 and 10/6 wavelets and their response to a unit step (ideal edge): (a) 9/7 analysis wavelet, (b) 9/7 step response, (c) 10/6 analysis wavelet, (d) 10/6 step response. The step response of a symmetric wavelet is antisymmetric and vice versa. A response of the type (d) may be more advantageous than (b) for zerotree wavelet encoding because an edge will produce a rather narrow response consisting of highly correlated wavelet maxima across scale.

C. Choice of the Wavelet Filters

Our implementation requires the use of even-length wavelet filters where the low-pass operator is symmetric and the high-pass antisymmetric. We compared several sets of filters and finally retained the 10/6 filter pair described in [12, p. 127]. The corresponding symmetric components in (1) are

$$\tilde{H}_s(z) = \frac{z^4 - 8z^2 + 16z + 46 + 16z^{-1} - 8z^{-2} + z^{-4}}{64} \quad (2)$$

$$H_s(z) = \frac{-z^2 + 2z + 6 + 2z^{-1} - z^{-2}}{8}. \quad (3)$$

The 10/6 filters are both of order three, yielding wavelets with three vanishing moments. Their frame bounds—analysis (0.925, 1.065) and synthesis $(A, B) = (0.943, 1.083)$—are rather close to one, which makes them particularly attractive in our application. We recall that the constants A and B represent, respectively, the minimum and maximum possible ratios between the norms in the signal and wavelet domains. In our case, the condition number of the transform B/A is close to one, which means that the filterbank is very close to being orthogonal, a property that it shares with the popular 9/7 pair that is now part of the JPEG2000 standard [13]–[15]. Near-orthogonality is a highly desirable feature in image compression because it justifies the use of uniform quantization and a ranking of the wavelet coefficients according to their magnitude (energy preservation property in the transformed domain).

In addition to our algorithm requirements, there are other good reasons for using such even-length filters. Villasenor identified a 6/10 pair (which is not as close to orthogonality as the one used here) as an excellent candidate for image compression [16]. He singled it out based on its superior shift-invariance properties. What we feel is perhaps an even stronger argument is that the corresponding wavelets, which are antisymmetric, tend to have a better response to step edges in images. They give rise to one clearly defined maximum at the location of the singularity, instead of a positive plus a negative alternation, as is the case with symmetric wavelets. This behavior is illustrated in Fig. 3. Practically, this means that an antisymmetric

wavelet should be better suited for zerotree wavelet encoding than a symmetric one, because the wavelet coefficient amplitudes will tend to be strongly correlated within a cone-like region that closely matches the zerotree. In contrast, a symmetric wavelet will also produce small coefficients at the location of the singularity; these will be difficult to code efficiently because they are isolated. Another way to put it is that the midpoint point symmetry is the one that best matches the structure of a binary tree—think of the Haar transform, which is even computable within the tree structure.

D. Zerotree Algorithm in 4-D

A zerotree is a tree with zero-children ending nodes. In 1-D, a zerotree corresponds to a binary tree with two children elements per parent node; in 2-D, a zerotree corresponds to a quadtree with four children per parent. "Binary" 3-D and 4-D trees can therefore be designed that consist of 8 and 16 children per parent, respectively.

E. Extension of Zerotree Algorithm to Arbitrarily Sized Data

When the size is a power-of-two, the "binary" design of the tree fits the wavelet decomposition structure well, but in other image sizes, this tree design matches the wavelet transform only if data expansion or cropping to a multiple of a power-of-two is performed. Basically, four strategies are possible to handle this problem, as depicted for the 1-D case in Fig. 4

1) *Expanding the dataset (Fig. 4, left upper panel):* When performing m wavelet decomposition levels, data need to be expanded to the next multiple of 2^m. While this is an adequate solution in 1-D, it becomes less viable in higher dimensions (n) where the number of additional pixels grows with the nth power of the padding size. In our example datasets, the data size would grow from $240 \times 216 \times 18 \times 18$ (64 MB when float datatype is used) to $240 \times 224 \times 32 \times 32$ (210 MB), thus leading to a large increase in computation time and hardware requirements. In addition, image padding adds data that need to be encoded, thus potentially reducing compression efficiency.
2) *Tiling the dataset to the next lower multiple of power-of-two (Fig. 2, left lower panel):* This is less demanding in terms of hardware requirements and computing time but has nevertheless a number of disadvantages: it forfeits a key advantage of the wavelet transform by opening the door to "block" or "tile" border artifacts. This is a well-known problem of the widely used JPEG standard that becomes predominant at high compression rates and that we would like to avoid as much as possible. In addition, the computer algorithm becomes rather complex, expecially in multidimensions, where a single "tiling" step per dimension leads in the 4-D case to no less than 16 different "hypercube tiles," all of which may have different sizes.
3) *Apply a conventional zerotree to the wavelet transform adapted to arbitrary size data:* Due to the mismatch of wavelet structure and zerotree structure, the children of one zerotree parent node may belong to different wavelet subbands. Because different subbands often contain a different amount of image energy, this may lead to grouping of coefficients with largely different values, making later entropy coding less successful.
4) *Use a tree structure with a variable number of children per parent* that matches the wavelet transform adapted to arbitrary size data. This appears appealing because it avoids the mentioned disadvantages of expanding, tiling, and mismatch between wavelet and zerotree structure. The modified tree structure is characterized as being "nonbinary." It features a variable number of children per parent, ranging from one to three in 1-D and 1^N to 3^N in N dimensions. The fact that all children belong to the same decomposition level facilitates their compression by entropy coding methods.

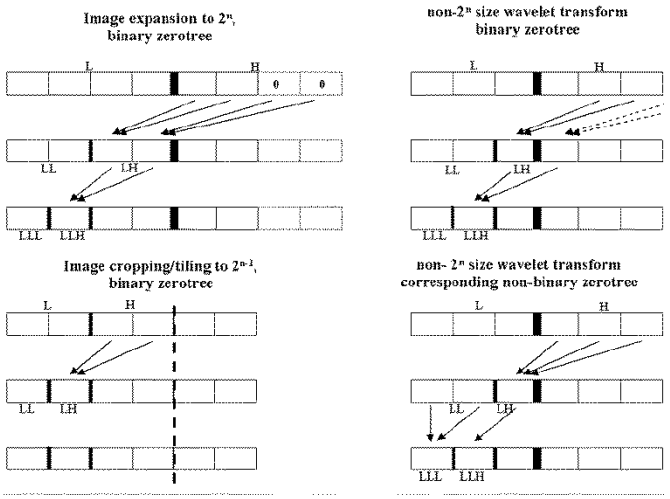

Fig. 4. Different apporaches to zerotree decomposition of non-power-of-2 data, displayed for the 1-D case. Boxes show creation of subbands in consecutive wavelet decomposition steps. Arrows show connectivity of children to their parents in the zerotree structure. Note that in our proposed solution (lower right), wavelet subbands and tree connectivity correspond also in arbitrary length data.

```
FOR level: = Firstlevel DOWNTO Lastlevel
 DO
 LengthH: = SignalLength DIV 2;
 LengthL: = (SignalLength + 1) DIV 2;
 LengthLH: = LengthL DIV 2;
 LengthLL: = (LengthL + 1) DIV 2;
 FOR child: = 0 TO LengthH – 1 DO
  parent: = child DIV 2;
  IF parent >= LengthLH THEN
   parent: = LengthLH – 1
  END;
  parent: = LengthLL + parent;
  ConnectParentToChild (parent, LengthL +
 child)
 END;
 Signallength: = LengthL
END;
```

Pseudocode for building a zerotree for arbitrary size data; multiple dimensions can be handled separately.

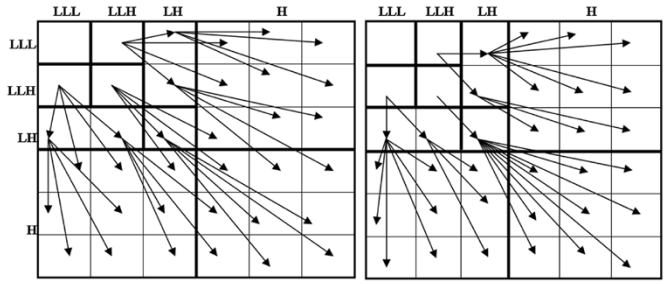

Fig. 5. Zerotree structures for wavelet transform not size 2^n. (a) Conventional zerotree with four children per parent. Wavelet decomposition structure and zerotree structure do not match. (b) Zerotree with variable number of children, with zerotree structure matching the structure of wavelet transform. Note that in (b), a parent may have from one (e.g., LLH/LLH) to nine (e.g., LH/LH) children L, LH, LLH, LLL: subbands of wavelet transform. Arrows: zerotree structure.

F. Building a Zerotree With a Variable Number of Children

The latter method requires a modification of the conventional algorithm for tree building; however, the tree structure is entirely determined by the size of the data in each dimension, so that no additional storage is needed for the description of the modified tree structure. The algorithm given in the text insert allows the construction of parent–child relationships in a zerotree for arbitrarily sized data, whereby the zerotree topography exactly matches that of the wavelet decomposition performed as described above. In this zerotree structure, all children of one tree branch thus belong to the same wavelet subband. The algorithm does not imply data extension, which means economic use of memory and computation. It is sufficiently simple for straightforward implementation.

Given the separability of multiple dimensions as to the parent–child relationship, this algorithm is easily extended to multiple dimensions. Fig. 5 depicts the structure of wavelet decomposition and zerotree parent–child relation in 2-D; in 3-D and 4-D, tree building is analogous.

Coefficients are scanned in the Z-raster sequence typical for zerotrees, starting in the very low-pass subband. In contrast to 2-D zerotrees, where there are three high-pass subband has three high-pass subbands per decomposition level, there are seven high-pass subbands in 3-D and 15 high-pass subbands in 4-D, requiring somewhat more bookkeeping. A zerotree node was assumed if a parent and all its children were zero, although other published approaches like stopping after a single zero-parent coefficient or scanning all details before assuming a zerotree would be equally applicable in our algorithm.

Coefficient scanning was done in the classical bit-plane manner [11], with decreasing threshold until a predefined data length was reached. More recent positional coding strategies, e.g., as described by Said and Perlman [18], are easy to plug in at this stage and may further improve compression ability.

G. Coding

The resulting file was further compressed using arithmetic coding [] (experimentally, Huffmann coding yielded inferior results). *Adaptive* frequency arithmetic coding was rather slow, while a general fixed frequency model did not perform so well for the compression of our quantized wavelet decomposition coefficients. We therefore first got a frequency model (requiring about 512 bytes storage space) of the 4-D zerotree. Then, we used these data for fixed frequency arithmetic coding to compress the 4-D zerotree. In this way, we were able to achieve an average storage space per coefficient that corresponds to the lower limit predicted by the entropy of the 4-D zerotree.

H. Error Measures

$$\max \texttt{error} = \max \left| f(x,y,z,t) - g(x,y,z,t) \right|, \quad (x,y,z,t) \in \texttt{hypercube}$$

$$\texttt{average error} = \sum_{(x,y,z,t)\in \text{hypercube}} \times \frac{\left| f(x,y,z,t) - g(x,y,z,t) \right|}{\texttt{NrofVoxels}}$$

$$\texttt{mean square error} = \sum_{(x,y,z,t)\in \text{hypercube}} \times \frac{\left| f(x,y,z,t) - g(x,y,z,t) \right|^2}{\texttt{NrofVoxels}}$$

$$\texttt{PSNR} = 10 \log_{10} \left(\frac{255^2}{\texttt{mean square error}} \right).$$

The image quality of compressed and decompressed image files was quantitatively compared using maximum error, average error, mean square error, and peak signal-to-noise ratio as parameters, as defined in the insert.

III. Experimental Results

A. Test Data

Dynamic 3-D echocardiographic data (yielding 4-D image datasets) were chosen for our tests, as this imaging modality is increasingly used in clinical cardiology. Echocardiograms were from consecutive patients undergoing echocardiograms for clinical indications and were not selected for image quality.

B. Impact of Handling of Non-Power-of-Two Sized Data

With regard to data size, our 3-D echo system (SONOS, Philips Medical Systems), which is widely available, features frame width and height of $240 * 216$ pixels, a fraction of 360 for the Z dimension, and a T dimension corresponding to the duration of the heart beat; i.e., possibly different for each data acquisition. For example, one of the datasets used for our experiments had a size of $240 \times 216 \times 18 \times 18$ for x, y, z, t, as determined by imaging hardware and patient heart rate, with 256 gray levels, accounting for a storage size of 16 MB when using 1 byte/pixel.

First, we implemented the simplest possible version of the wavelet coder where the image is expanded to the next power-of-two. In our 4-D data, this led to a marked increase in hardware requirements and computing time, as shown in Table I. In addition, a degradation of compression efficency due to the need to encode additional data ("zeros scattered in the tree") was observed.

TABLE I
DATA EXTENSION TO THE NEXT POWER-OF-TWO MAY, DEPENDING ON ORIGINAL DATA SIZE, HAVE A HIGH COST IN TERMS OF RAM REQUIREMENT AND COMPUTING TIME (PENTIUM-III, 550 MHz). IN ADDITION, COMPRESSION EFFICIENCY DEGRADES BECAUSE ADDITIONAL COEFFICIENTS HAVE TO BE ENCODED

Method for handling non-power-of-2 data	data size (MB)	wavelet decomposition time	zerotree encoding time	zerotree decoding time	wavelet reconstruction time	PSNR	compression ratio
non-binary zerotree	64MB	55s	9.1s	3.3s	61.94s	36.79	98:1
data expansion, binary zerotree	210MB	195s	71.1s	9.1s	221.4s	36.79	64:1

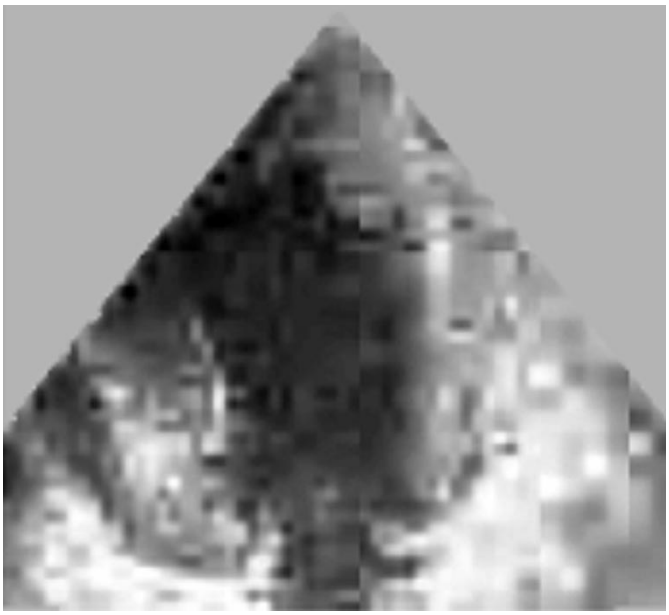

Fig. 6. Tiling of data to a power-of-two leading to linear artifacts.

Next, we considered the possiblity of *tiling of the data* in blocks with a power-of-two size followed by a conventional zerotree algorithm. We observed the following problems in implementing this approach: Especially at higher compression rates, visible blocking artifacts arise, as shown in Fig. 6. Tiling in multiple dimensions also complicates the software substantially. Some of the 4-D tiles are wide but of little depth, others are narrow but deep, and so on, thus effectively prohibiting a uniform approach for wavelet transform and zerotree building in different dimensions. In addition, image coherence across tile borders cannot be exploited by the compression algorithm, and tile borders are numerous in 4-D. The resulting cost in compression efficiency and reconstructed image quality for this approach was about 3 dB in peak SNR (PSNR) at several compression levels, compared to the best strategy described below. An additional disadvantage is that the "embeddedness" of the zerotree, i.e., the possibility of progressive transmission and successive refinement, is forfeited by tiling.

When the *wavelet decomposition is adapted to arbitrary size data*, as discussed in Section II, one can either choose a *conventional zerotree with* 2^N *children* per parents (N = number of dimensions) or the strategy proposed in this paper; i.e., constructing a zerotree with a variable number of children, which matches the wavelet decomposition structure exactly. Using a conventional "binary" zerotree leads to a number of parent nodes without children, as visible in Fig. 4, upper right panel. While data extension is not necessary in the algorithm, these nodes have to be coded as terminal nodes, which leads to some additional cost in compression efficiency. The resulting loss in image quality was about 1 dB at moderate compression rates in the 2-D case and increased with higher dimensionality of the data.

Using a *zerotree structure adapted to the wavelet decomposition and featuring a variable number of children* (1^N to 3^N per dimension) emerged as the strategy of choice: it could be performed on data of arbitrary size and yielded the best compression efficiency. This could all be achieved with a rather modest increase in algorithm complexity. The advantage of this approach is that the tree matches the structure of the wavelet transform in that the children of one parent node always belong to the same wavelet decomposition subband (thus sharing image energy), a property that is important for optimal performance of entropy-based coding methods.

C. Influence of the Dimensionality of the Zerotree on Compression Ability and Image Quality

The genuine 4-D zerotree approach with a variable number of children, as described above, was then compared to a compression strategy that separates the data into separate 2-D slices that are handled individually. Also tested was a 3-D version of the algorithm, after the original 4-D data hypercube was cut into separate 3-D cubes. Obviously, cutting the 4-D data into lower dimensional parts will destroy any "interframe" or "intercube" coherence that would be amenable to the compression algorithm, and a reduced compression power of the lower dimensional approach is therefore expected. Detailed compression results for one representative data set are given in Table II. Note that compression efficiency is best for the genuine 4-D approach at all compression levels, but the difference is especially marked at very high compression rates, suggesting that exploitation of data coherence in 3-D and 4-D is especially important in applications with low-bandwidth requirements.

In addition, reconstructed data were displayed as movies and printed as still frames shown in Fig. 7. Note that at least some key image features (left ventricular shape and global contractile function) remain visible even in 2000-fold compressed data with the genuine 4-D zerotree approach. Comparing Table II and Fig. 7, it is also interesting to note that peak SNR is not a very good surrogate marker for visual image quality: to achieve a compression rate of 128:1 in 2-D, it was necessary to sacrifice most high-pass information, leading to a loss of contours and a blurred image. In the 4-D 2048 : 1 compression example, however, important high-pass details that are strongly coherent

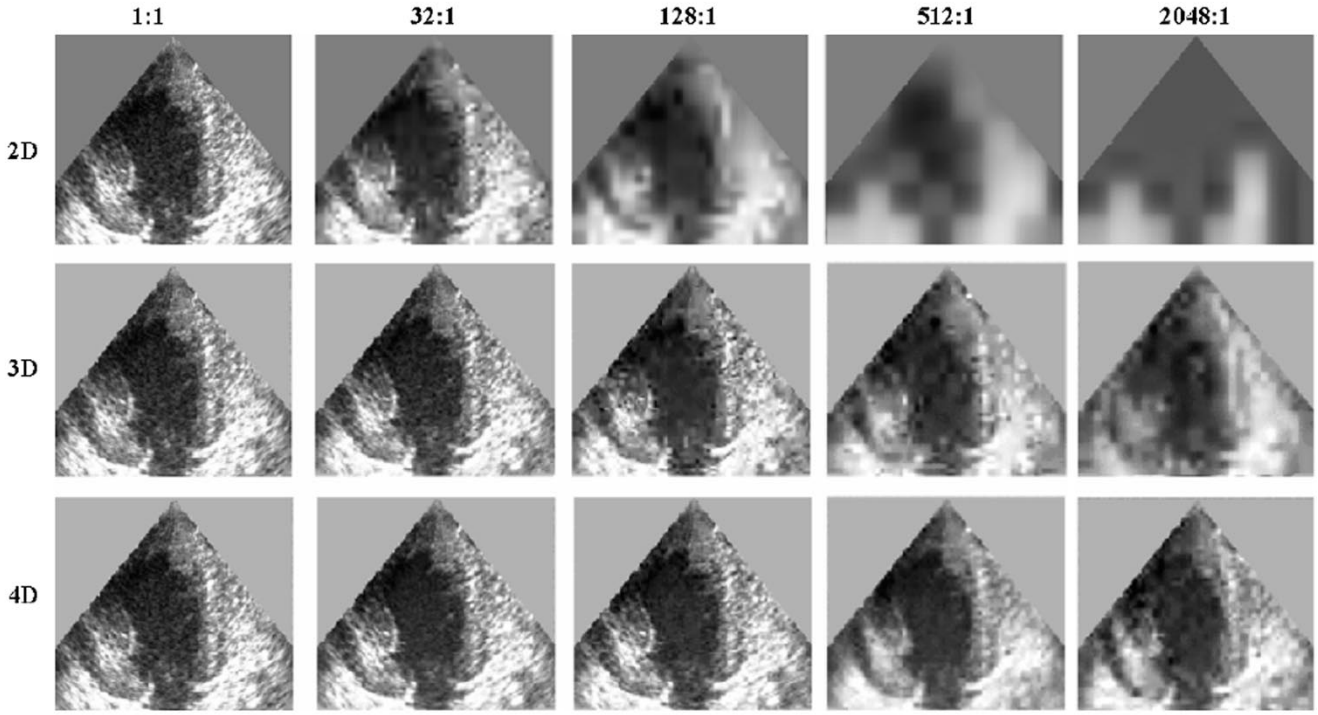

Fig. 7. 4-D dataset compressed with 2-D, 3-D, and 4-D zerotree. There is a marked improvement in image quality with the genuine 4-D approach, especially at high compression ratios, reflecting the impact of exploiting spatial data coherence in all dimensions.

TABLE II
INFLUENCE OF DIMENSIONALITY OF ZEROTREE AND OF TARGET COMPRESSION RATE ON IMAGE QUALITY. A GENUINE 4-D APPROACH IS CONSISTENTLY BETTER, EVEN MORE SO AT HIGH COMPRESSION RATIOS

compression method	compression ratio	max error	average error	mean square error	PSNR
2D zerotree	32:1	28	2.26	13.87	36.71
	64:1	30	2.72	19.85	35.15
	128:1	34	3.20	27.76	33.70
	256:1	34	3.72	36.48	32.51
	512:1	39	4.29	48.00	31.32
	1024:1	46	5.24	70.06	29.68
	2048:1	48	5.67	80.70	29.06
3D zerotree	32:1	28	2.00	11.07	37.69
	64:1	28	2.35	14.88	36.41
	128:1	32	2.87	22.45	34.62
	256:1	35	3.21	27.59	33.72
	512:1	34	3.72	36.83	32.47
	1024:1	42	4.86	66.77	29.88
	2048:1	47	5.19	74.93	29.39
4D zero tree	32:1	29	1.94	10.62	37.87
	64:1	28	2.05	11.59	37.49
	128:1	30	2.36	15.00	36.37
	256:1	31	2.81	21.35	34.84
	512:1	34	3.32	30.34	33.31
	1024:1	33	3.44	32.52	33.01
	2048:1	38	3.85	40.10	32.10

in space and/or time like cardiac borders could be preserved, resulting in an improved visual quality despite a lower PSNR, that is due to suppression of image information without coherence in space and time by the algorithm. This incoherent information probably contains an important amount of noise, and noise filtering is a classical example of an image operation that improves visual image quality but may reduce the PSNR in comparison to the noisy original image.

D. Clinical Validation

As numerical image quality parameters do not always reliably reflect clinically relevant visual image quality, compression quality was also rated by an expert in a series of nonselected echocardiograms that were acquired in clinical routine. We performed 156 individual compression experiments using various compression strategies and multiple compression rates. Image quality was judged by an experienced cardiologist through review of the original and compressed-reconstructed image data side-by-side. Each compressed-reconstructed image was displayed as a moving 2-D loop and was scored using a quality scale ranging from 1 to 10 (with 10 = identical to original image, 8 = discrete difference visually detectable without loss of diagnostic information, 6 = moderate image difference detectable, at most minor loss of diagnostic information, 4 = marked image degradation, significant loss of diagnostic information, and 2 = most diagnostic information lost). Overall differences in image quality were analyzed by analysis of variance, and posthoc testing between individual compression strategies was done by paired t-tests, using a two-sided p value below 0.05 as the criterion for significance. Results shown in Table III demonstrate that while image quality is (as expected) reduced with increasing compression, the use of a higher dimensional compression strategy consistently leads to improved images at all compression levels. For many current applications, the results in the $32–128x$ compression range are interesting, as they document that higher dimensional approaches allow significant savings in bandwith and storage with similar or superior image quality.

The very high compression range—while certainly not sufficient for exhaustive diagnosis—may be of interest in certain

TABLE III
CLINICAL VALIDATION OF COMPRESSION STRATEGIES IN 156 INDIVIDUAL COMPRESSION EXPERIMENTS ON NONSELECTED CLINICAL ECHOCARDIOGRAMS: INFLUENCE OF DIMENSIONALITY OF ZEROTREE AND OF TARGET COMPRESSION RATE ON SUBJECTIVE IMAGE QUALITY. RATING BY AN EXPERIENCED CARDIOLOGIST, USING A SEMIQUANTITATIVE QUALITY SCALE FROM 1 TO 10 (10 = NO DIFFERENCE TO ORIGINAL DETECTABLE). THE EXPERT RATED IMAGES COMPRESSED BY HIGHER DIMENSIONAL ALGORITHMS CONSISTENTLY BETTER AT ALL TARGET COMPRESSION RATIOS

	compression method	**2D zerotree**	**3D zerotree**	**4D zerotree**	*p (ANOVA)*	*p 2D vs 3D (t-test)*	*p 3D vs 4D (t-test)*
compression ratio	**32:1**	7.3±0.4	8.5±0.7	9.0±0.6	<0.001	<0.001	0.014
	64:1	5.8±0.6	7.2±0.6	7.9±0.9	<0.001	<0.001	0.001
	128:1	5.0±0.8	6.5±1.1	7.0±1.4	<0.001	<0.001	0.014
	300:1	3.7±1.1	5.1±1.0	6.0±1.1	<0.001	0.025	0.019
	2000:1	1±0	3.5±0.8	4.3±0.7	<0.001	<0.001	<0.001

situations where network limitations are severe, e.g., in areas where no more than a phone line is available. Our findings indicate notably that global left ventricular function, an important variable in myocardial infarction, remains visible even in the very highly (2000) compressed loops using the 4-D algorithm, while in 2-D at 2000 : 1 compression no cardiac structures were discernible in any patient.

IV. DISCUSSION

The increased availability of medical imaging technologies that yield 4-D data, combined with the low-bandwidth requirements of telemedicine, pose new demands for image-compression methods. In this paper, we have shown that genuine 3-D and especially 4-D algorithms dramatically outperform the standard lower dimensional zerotree wavelet encoders, mainly because these cannot fully exploit the coherence of the data in all dimensions. With datasets whose dimensions are not dyadic, this can be handled optimally through an adaptation of the zerotree algorithm that allows a variable number of children per parent, corresponding to the nonexpansive wavelet decomposition strategy performed on these data.

A potential disadvantage of the 4-D approach compared to a standard storage of 2-D slices is the fact that if only a single but highly resolved slice is needed, the entire dataset needs to be handled anyway. However, this is more important in radiology than in echocardiography, where single frames alone are rarely used for diagnostic purposes.

Most current wavelet compression algorithms, including JPEG2000, use odd-length filters with a central symmetry—the standard example being the Daubechies' 9/7 pair. In our case, we designed our method to work with even-length filters with midpoint symmetry, which is less standard. There are two important reasons that support this choice. First, we showed that these filters were applicable to the decomposition of signals of arbitrary length, without any additional data storage. Second, we also show that the midpoint antisymmetry of the corresponding wavelet (same type as the Haar transform) was best matched to the structure of the zerotree.

The proposed method is appealing because it improves the compression ability of conventional approaches, thus on one side improving image quality at low to moderate compression rates and or alternatively allowing very high compression rate while retaining diagnostically important image information; in addition, it relaxes the requirements for image dimensions but does not put additional demands on computer memory and computation time. Still, the algorithm is reasonably simple to implement and flexible enough to allow for the integration of recent refinements in wavelet coding [17].

With the significant improvement seen not only in numerical image parameters but also in the expert grading of the reconstructed loops, the described approach offers itself for a number of potential applications in medical imaging, ranging from the space-efficient storage of the typically very large moving 3-D ("4-D") datasets, to improved high-quality compression of image loops that may reduce bandwith requirements (and costs) for image transfer, and finally to some very low-bandwith applications, e.g., telemedicine diagnosis of myocardial infarction when only limited bandwidth is available but rapid recognition of major contractility abnormalities may be more urgent than transmission of the finest structural details of the heart.

However, there remains much to do in clinical validation of this approach. In an ongoing study, diagnostic accuracy and interobserver agreement using various approaches are tested in a larger range of patients using several experts, focusing on the low–moderate compression range. Another planned analysis is a more extended comparison of our even-length midpoint-symmetric wavelet approach with the standard Daubechies' 9/7 and other wavelet filters.

ACKNOWLEDGMENT

The authors would like to thank Dr. T. Blu for computing the frame bounds of Strang's 10/6 wavelets and for producing the graphs in Fig. 2.

REFERENCES

[1] M. Unser and A. Aldroubi, "A review of wavelets in biomedical applications," *Proc. IEEE*, vol. 84, pp. 626–638, Apr. 1996.
[2] K. R. Persons, P. M. Palisson, A. Manduca, W. J. Charboneau, E. M. James, N. T. Charboneau, N. J. Hangiandreou, and B. J. Erickson, "Ultrasound grayscale image compression with JPEG and wavelet techniques," *J. Digital Imag.*, vol. 13, no. 1, pp. 25–32, 2000.
[3] A. Munteanu, J. Cornelis, and P. Cristea, "Wavelet-based lossless compression of coronary angiographic images," *IEEE Trans. Med. Imag.*, vol. 18, pp. 272–281, Mar. 1999.
[4] J. J. Wang and H. K. Huang, "Medical image compression by using three-dimensional wavelet transformation," *IEEE Trans. Med. Imag.*, vol. 15, pp. 547–554, Aug. 1996.

[5] J. B. Luo, "Coherently three-dimensional wavelet-based approach to volumetric image compression," *J. Digital Imag.*, vol. 7, no. 3, pp. 474–485, 1998.

[6] A. Kalyanpur, V. P. Neklesa, C. R. Taylor, A. R. Daftary, and J. A. Brink, "Evaluation of JPEG and wavelet compression of body CT images for direct digital teleradiologic transmission," *Radiology*, vol. 217, no. 3, pp. 772–779, 2000.

[7] M. A. Losada, G. Tohumoglu, D. Fraile, and A. Artes, "Multi-iteration wavelet zerotree coding for image compression," *Signal Process.*, vol. 80, no. 7, pp. 1281–1287, 2000.

[8] L. Zeng, C. Jansen, M. Unser, and P. Hunziker, "Extension of wavelet compression algorithms to 3D and 4D image data: Exploitation of data coherence in higher dimensions allows very high compression ratios," *Proc. SPIE*, vol. 4478, no. 55, 2001.

[9] L. Jiebo, W. Xiaohui, C. W. Chang, and P. J. Kevin, "Volumetric medical image compression with three-dimensional wavelet transform and octave zerotree coding," *Proc. SPIE*, vol. 2727, pp. 579–590, 1996.

[10] L. Lin, W. Yun-Nan, L. Jin, and Z. Ya-Qin, "Compression of concentric mosaic scenery with alignment and 3D wavelet transform," *Proc. SPIE*, vol. 3974, pp. 89–100, 2000.

[11] J. M. Shapiro, "Embedded image coding using zerotrees of wavelet coefficients," *IEEE Trans. Signal Processing*, vol. 41, pp. 3445–3462, Dec. 1993.

[12] G. Strang and T. Nguyen, *Wavelets and Filter Banks*. Wellesley, MA: Cambridge Univ. Press, 1997, pp. 365–383.

[13] M. Antonini, M. Barlaud, P. Mathieu, and I. Daubechies, "Image coding using wavelet transform," *IEEE Trans. Image Processing*, vol. 1, pp. 205–220, 1992.

[14] C. Christopoulos, A. Skodras, and T. Ebrahimi, "The JPEG2000 still image coding system: An overview," *IEEE Trans. Consumer Electron.*, vol. 46, pp. 1103–1127, 2000.

[15] M. Unser and T. Blu, *Mathematical Properties of the JEGP2000 Wavelets*, 2000.

[16] J. D. Villasenor, B. Belzer, and J. Liao, "Wavelet filteer evaluation for image compression," *IEEE Trans. Image Processing*, vol. 4, pp. 1053–1060, Aug. 1995.

[17] A. Said and W. A. Pearlman, "A new, fast and efficient image codec based on set partitioning in hierarchical trees," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 6, pp. 243–250, June 1996.

[18] I. H. Witten, R. M. Neal, and J. G. Cleary, *Commun. ACM*, vol. 30, pp. 520–540, 1987.