

DETC2007/CIE-35628

A DYNAMIC 3D GEOMETRY COMPRESSION SCHEME BASED ON THE LIFTED WAVELET TRANSFORM

Malic Dekkar and Yan Wang
Center for e-Design
University of Central Florida
Orlando, FL 32816

ABSTRACT

In distributed environments, efficient visual information sharing is critical for effective communication in real-time engineering collaboration. Methods of geometry compression are needed for high-volume geometry data distribution over networks with limited bandwidths and heterogeneous storage capacities. In this paper, a new compression scheme for time-varying 3D geometry is introduced to support engineering and scientific visualization while showing potential for the audiovisual presentation and entertainment fields. This hybrid approach allows geometric and topological information to be uniformly encoded as volume grid values then compressed based on the lifted wavelet transform. The compression ratio is significantly increased without compromising surface quality due to rescaling and integer-to-integer lifting. This approach also allows for scalability in terms of additional data streams such as color, audio, and other types of concurrent data necessary for the desired customization of this method.

1. INTRODUCTION

3D geometry is one of the most extensively used data types in military, engineering, and multimedia communication. Examples include terrain surface (road, mountain), artifacts (building, machine, device, vehicle), and natural objects (human, animal). In collaborative engineering environments, large amount of 3D data needs to be stored locally and transmitted over networks. This requires speeding up of transmission and reducing storage space given limited resources. Compared to still image, audio, and video, which are widely used in compressed formats, 3D geometry data compression is relatively new.

In the past three decades, digital signal processing has evolved from one-dimensional to three-dimensional data. As illustrated in Figure 1, digital signal processing started from 1D signals (such as audio) in the 1970's, where signals were

digitalized and compressed by software and hardware. In the 1980's, 2D images and videos started being digitalized and processed by computers. In the mid 1990's, research on 3D geometry compression started, part of the driving force being the emergence of Internet. As the amount of data to be processed and transmitted over networks increases, 3D data applications based on the Internet require good performance given limited bandwidth and available storage space.

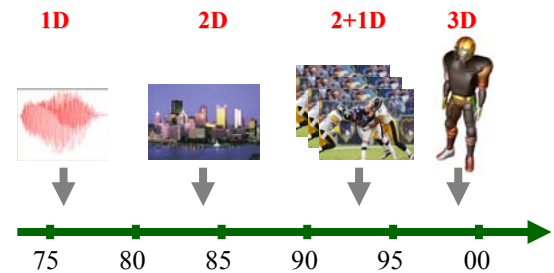


Figure 1. Digital signal processing has evolved from 1D, 2D to 3D

There has been extensive research on 3D static geometry compression in the past decade [1, 2]. Research results have been included in emerging standards such as binary VRML and MPEG-4 [3]. However, only a few focus on dynamic geometry change over time. Comparatively, just as 2D video complements 2D images, time-dependent 3D geometry can be looked as 3D video and has great potential in various applications, including communication, entertainment, scientific and medical computing, computer-aided design and engineering, as well as simulation and visualization. One can expect that 3D videos with compressed formats are standardized in the future and used as commonly as today's audio and video.

In general, there are two approaches in geometry

compression, mesh oriented and image oriented approaches. In the mesh oriented approach, both geometry (vertex coordinates in Euclidean space) and topology (connectivity among vertices) information is compressed. Accurate 3D meshes can be reconstructed, which is ideal for engineering applications. However, the connectivity of a mesh cannot be changed, which restrains it from general applications. In image oriented approaches, topology information is not considered. Volumetric geometry information is represented by voxels or points, and shapes can be compressed based on 2D image alike methods. The dynamics of topology can be captured easily, and the compression methods are general. However, rendering a visually recognizable and appealing surface requires a large amount of data. A balanced approach considering these two ends will possibly introduce a general solution with acceptable performance.

In previous work, we developed a time-varying 3D isosurface compression method based on the 4D lifted wavelet transform (LWT) [4]. This method is based on a generic volumetric compression scheme simultaneously considering spatial and temporal coherence. A rescaling and integer-to-integer transform approach is taken to increase compression ratios. Motion compensation effects are studied. In this paper, we present new mapping and encoding algorithms to enable mesh compression in the generic volumetric compression scheme. This enables an integrated geometry compression framework for both volumetric geometry in scientific visualization and mesh-based engineering analysis. In the rest of the paper, Section 2 gives an overview of related work in dynamic 3D geometry compression. Section 3 presents the LWT-based generic volumetric compression scheme. Section 4 describes a mapping algorithm that maps geometry from 3D Euclidean space to volumetric space while applying LWT compression. A topology encoding method is developed to allow for compression of connectivity information in a similar fashion.

2. BACKGROUND

As computing power grows, 3D animation becomes one of the important components in visual communication. 3D animation compression naturally catches attention. There are two approaches in 3D animation compression. In the first approach, topology is assumed to be static, and there is no or small change in connectivity. In the second approach, topology may change arbitrarily between frames.

2.1 3D Surface Animation with Static Topology

Lengyel [5] first proposed a compression algorithm based on encoding motion parameters. Meshes are clustered into segments and transformation of segments is compressed. The residuals are coded with spatial prediction. Alexa and Müller [6] used principle component analysis (PCA) to define base shapes and interpolation vectors are used to enhance animation. Ahn et al. [7] proposed a segmentation-based algorithm where

the approximation residuals are encoded using Discrete Cosine Transform. Sattler et al. [8] developed a clustered PCA segmentation method considering the time coherence with the analysis of vertex trajectories between frames.

Yang et al. [9] proposed a two-stage vertex-wise motion vector prediction algorithm to spatio-temporally predict both motion and the error for triangular mesh sequences. Ibarria and Rossignac [10] developed spatio-temporal geometry prediction algorithms with parallelogram-like extended Lorenzo predictor [11] and replica predictor. Karni and Gotsman [12] presented a coding scheme such that PCA captures the spatial redundancy of vertices and a second-order linear predictor exploits temporal coherence of PCA coefficients. Zhang and Owen [13] proposed an octree-based motion coding, in which each octree cell has eight motion vectors associated with its corners, and the motion of each vertex in the cell is interpolated. Müller et al. [14] combined the spatial clustering algorithm with context-adaptive binary arithmetic coding.

Briceño et al. [15] proposed a coding scheme based on geometry image [16], in which original mesh is parameterized, cut, remeshed, and fit into completely regular grids. The frames thus can be encoded by MPEG-like standard video compression methods. Guskov and Khodakovsky [17] applied wavelet coding on top of a progressive mesh hierarchy, in which I- and P-frames are differentiated. Payan et al. [18] used a temporal lifting wavelet scheme to exploit high temporal coherence of mesh sequences. Mathur et al. [19] proposed a measure of information loss based on curvature change and dynamic joint of skeleton to enhance compression performance.

The above research concentrated on dynamic mesh compression. The mesh models are assumed to have no or little topological change between frames. In general situations, 3D geometry with arbitrary topology changes may occur during animation.

2.2 3D Volume Animation with Dynamic Topology

Ma et al. [20] encoded quantized voxels with the consideration of time coherence to achieve compression. Shamir et al. [21, 22] constructed a multi-resolution compression data structure known as a time-direct acyclic graph (TDAG), to code time dependent polygonal meshes with arbitrary deformations. Guthe and Strasser [23] proposed a volume animation compression method based on the 3D wavelet transform for I-frames and motion prediction and compensation similar to MPEG for P- and B-frames. Lum et al. [24, 25] used temporal encoding of indexed volume data that can be quickly decoded in graphics hardware. Sohn et al. [26, 27] extracted and compressed isosurface-related volumes and volume differences with the 3D wavelet transform. Recently, Fout et al. [28] studied vector quantization and encoding of multi-variant volume information. Wang et al. [29] introduced a hierarchical wavelet-based time-space partitioning tree to organize time-varying volume data.

Different from the above coding methods, we propose a new 3D geometry animation compression scheme that integrates volumetric data and mesh data in a generic framework. 4D integer-to-integer LWT is used to compress heterogeneous data that is converted to volumetric space. It simultaneously considers spatial and temporal coherence for geometry and topology in coding. The aim is to explore the potential of a generic 3D video coding mechanism to support a diverse range of applications.

3. GENERIC VOLUMETRIC COMPRESSION SCHEME

The generic volumetric compression scheme proposed here is to convert different types of data such as geometry coordinates, normal vectors, topological connectivity, colors, isovalues, etc. for engineering and scientific visualization to a regularized volumetric data. Then these volumetric data as well as their changes along time are uniformly compressed and decompressed with LWT. This general framework is illustrated in Figure 2. 4D volumetric data is divided into groups of frames (GOFs). Each GOF is decomposed and compressed with 4D LWT at the server side. When received by the client, the compressed data is decoded and decompressed. The advantages of this wavelet-based approach include scalability and simultaneous redundancy reduction for spatial and temporal domains. With the inherent scalable representation, wavelets provide a multi-resolution solution without extra costs. Wavelets capture coherence locality of spatial and temporal domains. There is no need to divide intraframe data into blocks, as in MPEG-2 standards based on discrete cosine transform (DCT). In this section, the integer-to-integer LWT is introduced and illustrated with native volumetric data such as scientific simulation results. In Section 4, the mapping methods to convert geometric coordinates and topology to volumetric data will be described.

3.1 4D Lifted Wavelet Transform

The lifting scheme [30] is also known as the second generation wavelet transform. It is a two-step filtering process: prediction and update. In the prediction step, even sequences are used to predict odd sequences. The prediction error forms

the corresponding high-pass subband. In the update step, an approximation subband is obtained by updating even sequences with the scaled high-subband samples, which forms a low-pass subband. The main advantage of lifting is its memory efficiency in computation. Different from traditional traversal discrete wavelet transform (DWT), the wavelet coefficient calculation in the lifting scheme can be embedded in-place. The backward transform is easy to find and has the same complexity as the forward transform. A lossless integer-to-integer transform [31] can also be achieved.

The two-step lifting transform can be generally described as

$$h_k[x] = f_{2k+1}[x] + \sum_i p_i f_{2(k-i)}[x] \tag{1}$$

$$l_k[x] = f_{2k}[x] + \sum_j u_j h_{k-j}[x]$$

where $f_k[x]$ is the sequence of input data to be processed, h_k and l_k are resulting high-pass and low-pass sequences respectively, p_i and u_j are prediction and update coefficients of filters respectively. For example, the well studied Haar and 5/3 wavelet kernel are listed in Table 1.

Table 1. Examples of lifting wavelet kernels

Haar	5/3
p: $p_0 = -1.0$	p: $p_{-1} = -0.5$ $p_0 = -0.5$
u: $u_0 = 0.5$	u: $u_{-1} = 0.25$ $u_0 = 0.25$

An important decision associated with spatio-temporal decomposition is the choice of filters. Different filters exhibit varied signal characteristics in terms of energy compactness in the transform domain and coding gains. Long filters tend to explore coherences of large regions or long periods of time. However, they may blur boundaries of occupation or movement.

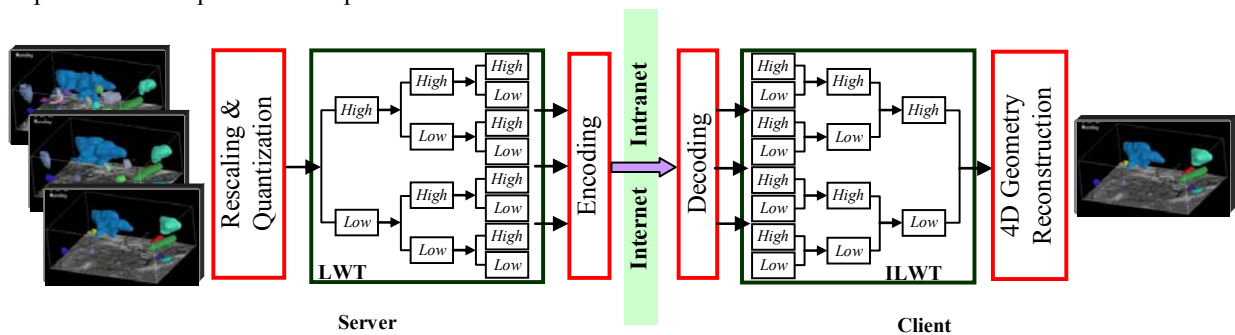


Figure 2. Volume animation scheme based on 4D lifted wavelet transform

Two decomposition approaches can be taken for 4D wavelet transform. In dyadic decomposition, wavelet transform is applied in spatial X , Y , Z , and time T dimensions alternatively. In decoupled decomposition, the transform is first cascadingly applied along the T dimension, then in X , Y , and Z dimensions. A dyadic decomposition approach is taken in this paper since it simultaneously considers the coherence relation between time and space.

A computational fluid dynamic simulation example is used to illustrate the visual effect of volumetric data compression. Figure 3-(a) shows a few frames of the original 3D animation of “wind” speed simulation. A 2-level decomposition process is applied with Haar lifting scheme. Compression is achieved by setting the transformed coefficients to zero with the original values less than a threshold. If the highest coefficient magnitude is M and the threshold is t , then all transformed coefficients which absolute values are less than $t \times M$ are set to zeros. Figure 3-(b), -(c), and -(d) show the compressed animations with the threshold values of 0.1%, 1%, and 5% respectively. In the compressed files, 4D arrays of transformed coefficients are concatenated, and zero values are ignored. Meta information such as data type flags and dimensions of arrays is also included.

3.2 Rescaling and Integer-to-Integer Transform

Compared to classical wavelet transform, in which transformed wavelet coefficients are floating point numbers even if the original data are integers, lifting scheme supports lossless integer-to-integer transform. It transforms integer data to integer coefficients. With an inverse transform, the original integer data can be reconstructed.

If the compression scheme is applied to isosurface visualization, the integer-to-integer LWT shows the advantage of compression ratios, since the isosurface construction is not sensitive to the number of bits used in coding if the range of grid values is large. As a result, floating-point grid values can be rounded to integers and the compression ratio is significantly increased with little impact on the isosurface quality. If the range of grid values is too small, the grid values can be rescaled before the rounding. A good rescaling strategy is to rescale the isosurface values as close to zero as possible and the overall grid values to be evenly distributed between the positive and negative sides. This will reduce the number of bits to code values. Figure 4 shows the result of applying integer-to-integer LWT to the previous wind animation example in Figure 3. The original grid values are multiplied by 10000

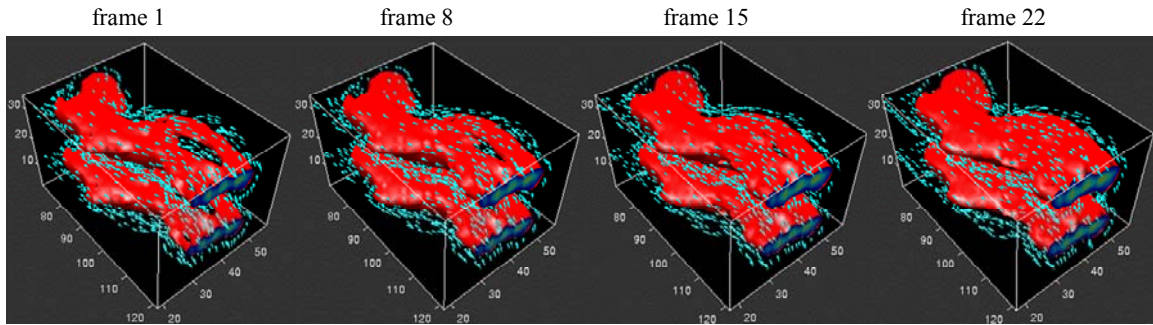
before rounding. To reduce distortion, floor operation that rounds towards negative infinity is used. Then the integer lifting is applied. Compressed data size is reduced to 1,924KB with little visual difference from the original animation in Figure 3-(a).

The reconstructed surfaces generated using the integer to integer transform maintain a high fidelity compared to the original surface. The wavelet transform inherently supports multi-resolution representation. That is, if only a portion of data is transferred and decompressed via multi-level encoding, models with different resolutions can be reconstructed.

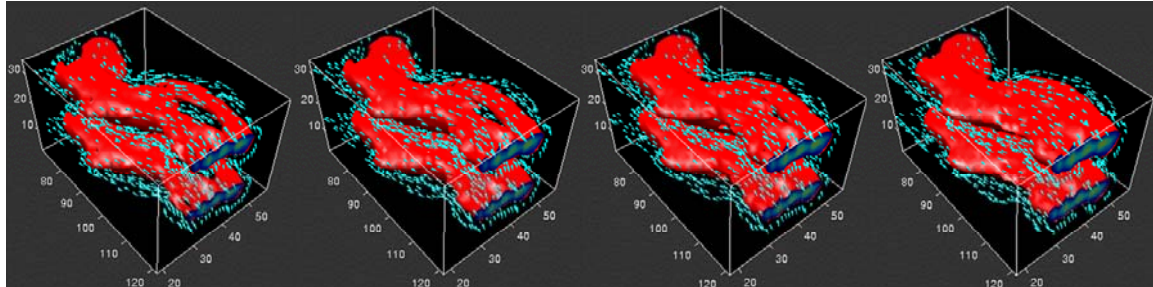
4. VOLUME MESH CODING

In data compression applications, it is important to emphasize the extensibility of algorithms. Different data types such as color, audio, or scientific values should be compressed as needed. In order to apply compression methods to general time-varying volume meshes, dynamic connectivity and geometry information must be encoded. In 4D compression, both spatial (within each frame) and temporal correlation (in between frames) should be considered.

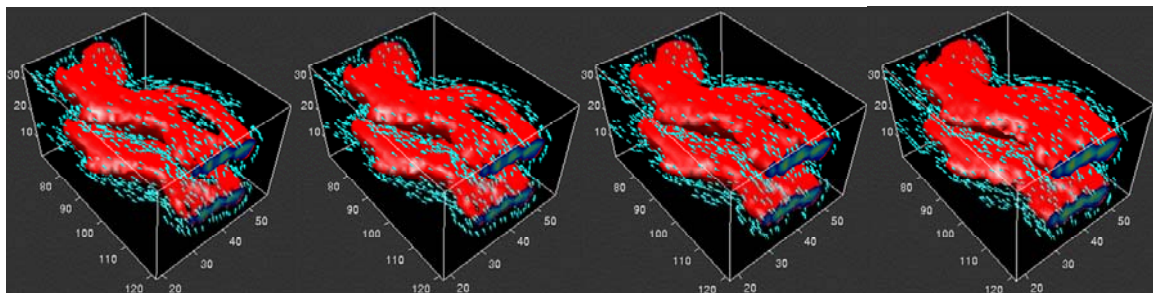
We developed a compression scheme capable of retaining both the geometry and connectivity information of a volume mesh while maintaining temporal synchronization. From a general volume mesh model, geometry information is mapped from the 3D Euclidean space to a volumetric space based on a *spherical hashing algorithm*. A volumetric space is a collection of regularized 3D matrices with elements of data that contain information to be compressed with the LWT. Topology information is also encoded and stored in the same volumetric space. Figure 5 depicts the idea of the generic compression scheme. The x , y , z , *isovalue*(*iso*), and *connectivity information*(*con*) along with an attribute placeholder for future expansion, denoted as *other*, are stored separately in respective 3D matrices. All of the matrices containing the necessary attributes, including Cartesian coordinate values, connectivity, isovalues, and others, at a point in time contribute to a single frame. Temporal synchronization is maintained across matrices, where all the matrices at T_1 are part of Time Frame 1, T_2 represents Time Frame 2 and so forth. The spherical hashing algorithm for geometry mapping is described in Section 4.1. The topological encoding is explained in Section 4.2.



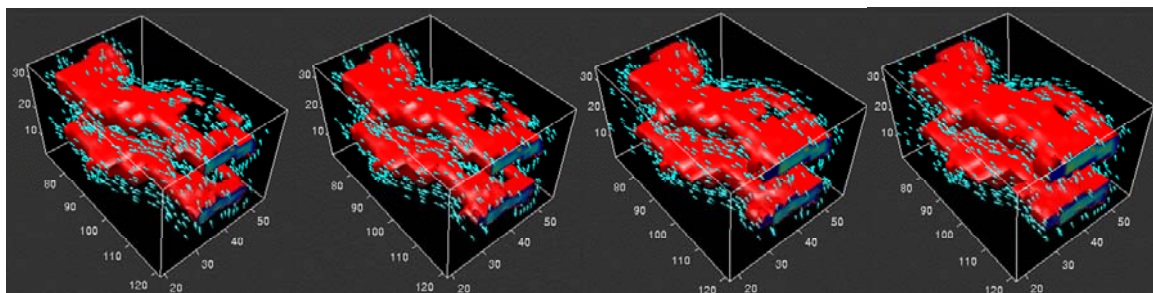
(a) original volume stream with a size of 6,474KB



(b) compressed volume stream with the threshold of 0.1% and the file size of 4,568KB



(c) compressed volume stream with the threshold of 1% and the file size of 2,585KB



(d) compressed volume stream with the threshold of 5% and the file size of 891KB

Figure 3. Compressed animation with a threshold of 0.1% with the file size of 4,568KB

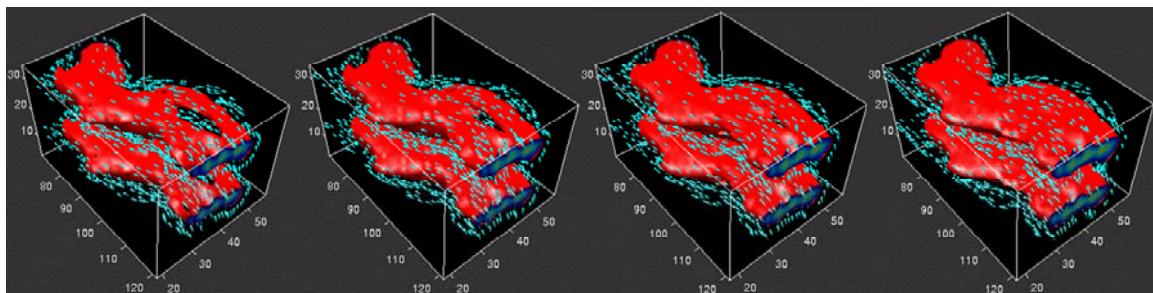


Figure 4. Compressed "wind" animation with integer-to-integer LWT with the file size of 1,924KB

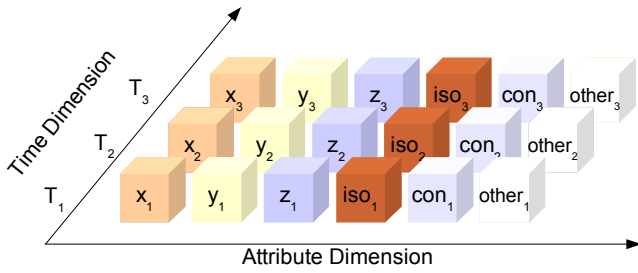


Figure 5. Time-varying matrix-based attributes in volumetric space

4.1 Spherical Hashing Algorithm

The mapping of geometry from Euclidean space to volumetric space is based on a *Spherical Hashing Algorithm*. As illustrated in Figure 6, for each vertex in the mesh model, the corresponding Cartesian coordinates are first converted to spherical coordinates where one Cartesian coordinate yields exactly one spherical coordinate: $(x, y, z) \Rightarrow (r, \theta, \phi)$. Then, the value of x , y , and z coordinates of each vertex is mapped to a cell of the respective 3D matrices in volumetric space. Each attribute (x , y , z , isovalues, connectivity, etc.) forms a 3D matrix. Based on certain threshold values, a quantization procedure is applied to r , θ , and ϕ values so that the maximum, minimum, and the number of distinct values of each are derived. The number of distinct values of r , θ , and ϕ determine the size of the matrices. Since each vertex consists of the same number of coordinates (x, y, z) and attributes (depending on the desired additional attributes) all matrices have the same size. The length of any matrix is the number of distinct values of r . The depth of any matrix is the number of distinct values of

θ . The height of any matrix is the number of distinct values of ϕ .

For any given vertex in Euclidean space, its associated attributes (x, y, z, iso , etc.) are inserted into the corresponding cells of the matrices with the exact same indices (i, j, k) with respect to r, θ , and ϕ , as shown in Figure 7. For instance, if there is a vertex $\mathbf{p} = (x_p, y_p, z_p)$ in Euclidean space with an isovalue attribute of iso_p , its corresponding spherical coordinates are (r_p, θ_p, ϕ_p) . The discretized spherical coordinate values $(\lfloor r_p \rfloor, \lfloor \theta_p \rfloor, \lfloor \phi_p \rfloor)$ determine the matrix cell address (i, j, k) to insert the values. If we designate $X_{i,j,k,t}$, $Y_{i,j,k,t}$, $Z_{i,j,k,t}$, and $ISO_{i,j,k,t}$ as the values of x, y, z , and iso 3D matrices in cell (i, j, k) respectively, at a point in time t , then we store the attributes of vertex $\mathbf{p} = (x_p, y_p, z_p)$ in each matrix such that: $X_{i,j,k,t} = x_p$, $Y_{i,j,k,t} = y_p$, $Z_{i,j,k,t} = z_p$, $ISO_{i,j,k,t} = iso_p$. A negative big integer is inserted to identify an empty cell if there is no vertex in the volumetric space.

4.2 Topology encoding

Once the geometry coordinates have been mapped into their respective cells in the volumetric space, the topology can also be encoded in matrices with the same size as geometry 3D matrices X, Y , or Z . The connectivity between one vertex and other neighboring vertices in the volumetric space instead of Euclidean space is recorded. This information is encoded according to a pre-defined connectivity pattern and represented by a 26 bit integer.

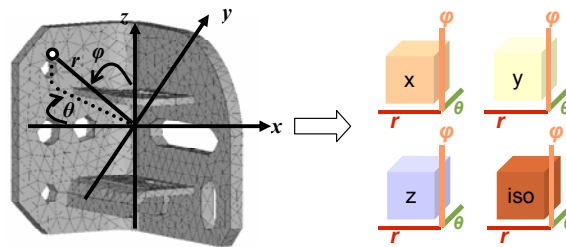


Figure 6. Spherical hashing mapping of a volumetric mesh

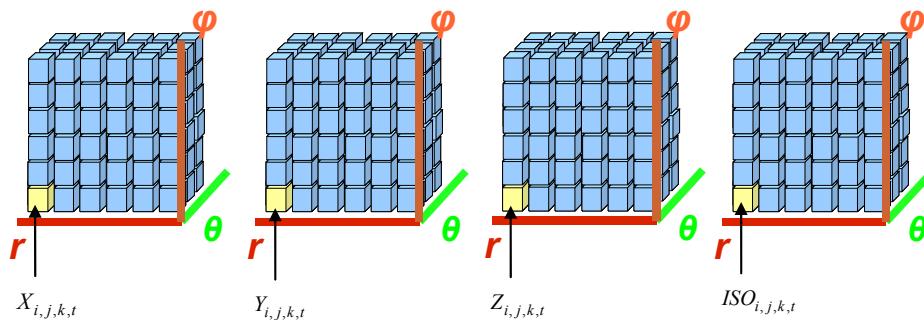


Figure 7. Coordinate & Data Attribute Correspondence

This method only takes into account the connectivity information between one cell and its neighbors, and its neighbors' neighbors. The assumption here is that the mesh models are regular or semi-regular so that the probability that a vertex is connected to another one beyond its neighbors' neighbors is very low. Thus, as shown in Figure 8, a cubic sub-matrix with $3 \times 3 \times 3$ cells is chosen to analyze and encode topology. To encode the connectivity of a vertex $\mathbf{p} = (x_p, y_p, z_p)$ in Euclidean space, the indice of the corresponding cell in the topology matrix, P_f (referred herein as the focal point), can be determined based on X , Y , or Z matrices in Figure 9. In other words, P_f is the point in volumetric space with respect to whom the connectivity pattern of the members of the $3 \times 3 \times 3$ subsection will be analyzed. Due to the fact that cell correspondence is maintained, the topology information can be encoded using any one of the X , Y , or Z matrices as a reference since connectivity is based on \mathbf{p} and any of its coordinates leads to the others by association. In Figure 10, P_f is shown in yellow, P_f 's neighbors are shown in red, and the neighbors' neighbors are green. As P_f advances sequentially within the 3D matrix, so does the relative $3 \times 3 \times 3$ subsection.

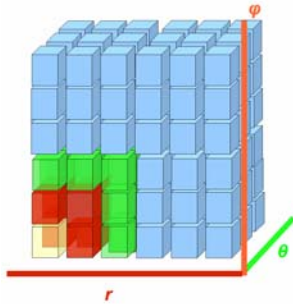


Figure 8. Selection of a $3 \times 3 \times 3$ subsection for the Topology Coding.

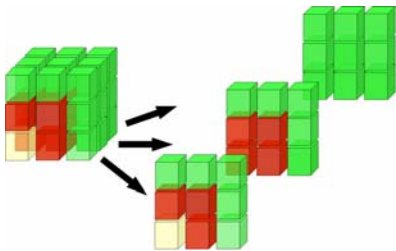


Figure 9. Exploded view of the vertical subsection showing the neighbors (red) and neighbors' neighbors (green).

The naming convention used in the topology encoding assumes that any shift towards a single cell's direction involves a Δ in that direction. Thus a shift along a single unit of (r, θ, ϕ) directions would yield a $(\Delta r, \Delta \theta, \Delta \phi)$, similarly a progress along any two units in the ϕ , for example, would yield $(2\Delta \phi)$.

The values of the 3D topology matrix are determined as follows: if a connection exists between a vertex \mathbf{p} and the vertices associated with the focal point's neighbors or neighbors' neighbors (depending on the bit being examined), then a 1 is assigned to the corresponding bit, otherwise a 0 is assigned. Once all of the 26 cells of the sub-matrix have been verified for connectivity, a 26-bit integer (defined in Table 2) designated as the *topological code* is formed where the most significant bit (MSB) is the rightmost bit position. The topological code is placed in P_f 's relational location in the *con* volumetric space (shown in Figure 11) and labeled as con_p , where $CON_{i,j,k,t} = con_p$. Therefore every value of the *con* box shall contain the connectivity information of the corresponding vertex \mathbf{p} , its neighbors and its neighbors' neighbors. The focal point, P_f , sequentially shifts from left to right, from front to back, and from bottom to top across the volumetric space such that all the points in the space will be checked for connectivity according to the coding scheme in Table 2. A periodic boundary condition is applied during the checking of each cell, in which the neighboring vertices are wrapped to the left (front, bottom) if they are out of the right (back, top) boundaries. Each bit-wise iteration of the topological coding follows the order in Figure 11 such that the neighbors and neighbors' neighbors are noted as $P_{f+1}, P_{f+2}, P_{f+3} \dots P_{f+n}$ where $n = 26$.

Similar to geometry, isovalue, and other attributes, once all the points of the generic volume mesh have been analyzed the attribute and *CON* matrices are formed such that all cells contain either a value or an empty cell identifier, the topological matrices then can be compressed with the LWT and decompressed to regenerate the connectivity information.

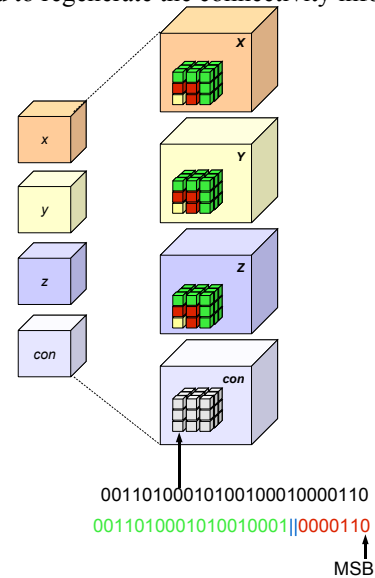


Figure 10. The connectivity of P_f is encoded in a 26-bit topological code where the neighbor's bit values are red and the neighbors' neighbors are green..

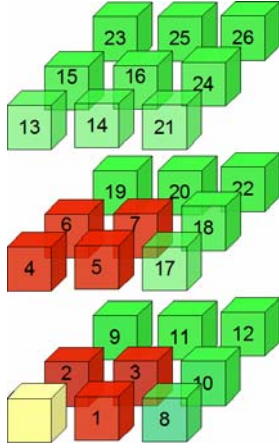


Figure 11. Graphical representation of topology coding order presented in Table 2. Each numeral represents the topological code's corresponding bit

Table 2. Bit pattern of topology coding scheme.

bit	r	θ	φ
1	Δr	0	0
2	0	$\Delta\theta$	0
3	Δr	0	$\Delta\varphi$
4	0	0	$\Delta\varphi$
5	Δr	0	$\Delta\varphi$
6	0	$\Delta\theta$	$\Delta\varphi$
7	Δr	$\Delta\theta$	$\Delta\varphi$
8	$2\Delta r$	0	0
9	0	$2\Delta\theta$	0
10	$2\Delta r$	$\Delta\theta$	0
11	Δr	$2\Delta\theta$	0
12	$2\Delta r$	$2\Delta\theta$	0
13	0	0	$2\Delta\varphi$
14	Δr	0	$2\Delta\varphi$
15	0	$\Delta\theta$	$2\Delta\varphi$
16	Δr	$2\Delta\theta$	$\Delta\varphi$
17	$2\Delta r$	0	$\Delta\varphi$
18	$2\Delta r$	$\Delta\theta$	$\Delta\varphi$
19	0	$2\Delta\theta$	0
20	Δr	$2\Delta\theta$	$\Delta\varphi$
21	$2\Delta r$	0	$2\Delta\varphi$
22	$2\Delta r$	$2\Delta\theta$	$\Delta\varphi$
23	0	$2\Delta\theta$	$2\Delta\varphi$
24	$2\Delta r$	$\Delta\theta$	$2\Delta\varphi$
25	Δr	$2\Delta\theta$	$2\Delta\varphi$
26	$2\Delta r$	$2\Delta\theta$	$2\Delta\varphi$

4.3 An Mesh Model Example

The deformation animation of a fringe plate shown in Figure 12 is encoded. The model contains 136 vertices and 280 triangles. After the spherical hashing of the vertices, the part is transformed to matrices in volumetric space containing 136 distinct values of r , 85 distinct values of θ , and 136 distinct values of φ . This signifies that 37.5 % of the yielded θ values were identical. Therefore the resulting cell count of the 3D

matrices in volumetric space is 136 x 85 x 136, or 1,572,160 cells.



Figure 12. The deformation of a fringe plate model as an example

All x_p, y_p, z_p values as well as the topological information of the fringe plate model are able to be retrieved losslessly from the 4D matrices encoding. The topology encoding algorithm enabled full point retrieval as compared in Figure 13. The compressed 4D matrix of eight frames containing x coordinates has a size of 155 KB, while the size of the original matrix is 507 KB. The sizes of matrices containing y, z , and topology information are similar.



Figure 13. Before and after topological coding algorithm.

5. CONCLUDING REMARKS

In this paper, a new time-varying 3D geometry compression scheme based on 4D lifted wavelet transform is presented. Both geometric information and topological connectivity are encoded as volumetric grid values and compressed uniformly with LWT. Thus the compression supports both volumetric data and volume mesh geometry. Rescaling and integer-to-integer lifting show significant improvement on compression ratios without affecting surface

quality.

The generic geometric and topological coding scheme, based on a Spherical Hashing method, is used to store mesh information in 4D matrices. The method creates vertex correspondence between the spatial and volumetric domains. The addition of attributes is enabled by adding extra 4D matrices to the volumetric data set. The creation of the topological code in the same volumetric domain allows for the storage of connectivity in a similar way, prior to being compressed via the LWT.

This 4D geometry compression can be used in general applications such as scientific computing and visualization, collaborative engineering, modeling and simulation, teleconferencing, and entertainment. The future work includes performance analysis of the spherical hashing algorithm for different data sets, optimization of topological encoding and neighbor sequencing combination, as well as wavelet transform filter selection.

ACKNOWLEDGEMENT

The authors appreciate the support from industry partners at the NSF I/UCRC Center for e-Design.

REFERENCES

- [1] P. Alliez and C. Gotsman, "Recent advances in compression of 3D meshes," in *Proc. Symp. on Multiresolution in Geometric Modeling*, 2003
- [2] J. Peng, C.-S. Kim, and C.-C. J. Kuo, "Technologies for 3D mesh compression: A survey," *J. Visual Communication & Image Representation*, vol.16, pp.688-733, 2005
- [3] G. Taubin, W. Horn, F. Lazarus, and J. Rossignac, "Geometry coding and VRML," *Proc. the IEEE*, vol.96, no.6, pp.1228-1243, 1998
- [4] Y. Wang and H. Hamza, "Time-varying volume geometry compression with 4D lifting wavelet transform," *Lecture Notes in Computer Science*, vol.4077, pp.670-676, 2006
- [5] J. Lengyel, "Compression of time dependent geometry," *Proc. 1999 ACM Symp. on Interactive 3D Graphics*, pp.89-95, 1999
- [6] M. Alexa and W. Müller, "Representing animations by principle components," *Computer Graphics Forum*, vol.19, no.3, pp.411-418, 2000
- [7] J.-H. Ahn, C.-S. Kim, C.-C. J. Kuo, and Y.-S. Ho, "Motion-compensated compression of 3D animation models," *IEE Electronics Letters*, vol.37, no.24, pp.1445-1446, 2001
- [8] M. Sattler, R. Sarlette, and R. Klein, "Simple and efficient compression of animation sequences," *Proc. EUROGRAPHICS 2005 / ACM SIGGRAPH Symp. on Computer Animation*, pp.209-217
- [9] J.-H. Yang, C.-S. Kim, and S.-U. Lee, "Compression of 3-D triangle mesh sequences based on vertex-wise motion vector prediction," *IEEE Trans. Circuits and Systems for Video Technology*, vol.12, no.12, pp.1178-1184, 2002
- [10] L. Ibarria and J. Rossignac, "Dynapack: space-time compression of the 3D animations of triangle meshes with fixed connectivity," *Proc. EUROGRAPHICS 2003 / ACM SIGGRAPH Symp. on Computer Animation*, pp.126-135, 2003
- [11] L. Ibarria, P. Lindstrom, J. Rossignac, and A. Szymczak, "Out-of-core compression and decompression of large n-dimensional scalar fields," *Proc. EUROGRAPHICS 2003 Computer Graphics Forum*
- [12] Z. Karni and C. Gotsman, "Compression of soft-body animation sequences," *Computers & Graphics*, vol.28, pp.25-34, 2004
- [13] J. Zhang and C.B. Owen, "Octree-based animated geometry compression," *Proc. 2004 IEEE Data Compression Conf.*, pp.508-517, 2004
- [14] K. Müller, A. Smolic, M. Kautzner, P. Eisert, and T. Wiegand, "Predictive compression of dynamic 3D meshes," *Proc. 2005 Int. Conf. on Image Processing*, Genova, Italy, 2005
- [15] H.M. Briceño, P.V. Sander, L. McMillan, S. Gortler, and H. Hoppe, "Geometry videos: A new representation for 3D animations," *Proc. EUROGRAPHICS 2003 / ACM SIGGRAPH Symp. on Computer Animation*, pp.136-146, 2003
- [16] X. Gu, S. Gortler, and H. Hoppe, "Geometry images," *Proc. ACM SIGGRAPH*, pp. 355-361, 2002
- [17] I. Guskov and A. Khodakovsky, "Wavelet compression of parametrically coherent mesh sequences," *Proc. EUROGRAPHICS 2004 / ACM SIGGRAPH Symp. on Computer Animation*, pp.183-192, 2004
- [18] F. Payan, Y. Bouffani, and M. Antonini, "Temporal lifting scheme for the compression of animated sequences of meshes," *Proc. IEEE Int. Workshop VLBV 2005*, Sardinia, Italy, 2005
- [19] P. Mathur, C. Upadhyay, P. Chaudhuri, and P. Kalra, "A measure for mesh compression of time-variant geometry," *Comp. Anim. Virtual Worlds*, vol.15, pp.289-296, 2004
- [20] K.-L. Ma, D. Smith, M.-Y. Shih, and H.-W. Shen, "Efficient encoding and rendering of time-varying volume data," *NASA/CR-1998-208424 ICASE Report NO.98-22*, 1998
- [21] A. Shamir, V. Pascucci, and C. Bajaj, "Multi-resolution dynamic meshes with arbitrary deformations," *Proc. IEEE Visualization 2000*, pp.423-430, 2000
- [22] A. Shamir and V. Pascucci, "Temporal and spatial level of details for dynamic meshes," *Proc. Virtual Reality Systems and Techniques*, pp.423-430, 2001
- [23] S. Guthe and W. Strasser, "Real-time decompression and visualization of animated volume data," *Proc. IEEE Visualization 2001*, pp.349-356, 2001
- [24] E.B. Lum, K.-L. Ma, and J. Clyne, "Texture hardware assisted rendering of time-varying volume data," *Proc. IEEE Visualization 2001*, pp.263-270, 2001

- [25] E.B. Lum, K.-L. Ma, and J. Clyne, "A hardware-assisted scalable solution for interactive volume rendering of time-varying data," *IEEE Tran. on Visualization and Computer Graphics*, vol.8, no.3, pp.286-301, 2002
- [26] B.-S. Sohn, C. Bajaj, and V. Siddavanahalli, "Feature based volumetric video compression for interactive playback," *Proc. IEEE/SIGGRAPH Symp. on Volume Visualization and Graphics 2002*, pp.89-96, 2002
- [27] B.-S. Sohn, C. Bajaj, and V. Siddavanahalli, "Volumetric video compression for interactive playback," *Computer Vision and Image Understanding*, vol.96, pp.435-452, 2004
- [28] N. Fout, K.-L. Ma, and J. Ahrens, "Time-varying, multivariate volume data reduction," *Proc. 2005 ACM Symp. on Applied Computing*, pp.1224-1230, 2005
- [29] C. Wang, J. Gao, L. Li, and H.-W. Shen, "A multiresolution volume rendering framework for large-scale time-varying data visualization," *Proc. EUROGRAPHICS Int. Workshop on Volume Graphics 2005*
- [30] W. Sweldens, "The lifting scheme: A custom-design construction of biorthogonal wavelets," *Applied and Computational Harmonic Analysis*, vol.3, no.2, pp.186-200, 1996
- [31] A.R. Calderbank, I. Daubechies, W. Sweldens, and B.-L., Yeo, "Lossless image compression using integer to integer wavelet transforms," *Proc. IEEE Int. Conf. Image Processing*, pp.596-599, 1997