(H)EXASHRINK: MULTIRESOLUTION COMPRESSION OF LARGE STRUCTURED HEXAHEDRIC MESHES WITH DISCONTINUITIES IN GEOSCIENCES

Jean-Luc Peyrot[1,4], *Laurent Duval*[2,3], *Sébastien Schneider*[1], *Frédéric Payan*[4] *and Marc Antonini*[4]

[1]IFP Energies nouvelles,
Rond-point de l'échangeur de Solaize, BP 3
F-69360 Solaize
{first.lastname}@ifpen.fr

[2]IFP Energies nouvelles,
1 et 4 avenue de Bois-Préau
F-92852 Rueil-Malmaison
{first.lastname}@ifpen.fr

[3]University Paris-Est,
LIGM, ESIEE Paris
F-93162 Noisy-le-Grand

[4]Univ. Nice-Sophia Antipolis, CNRS,
Laboratory I3S, UMR 7271
F-06903 Sophia Antipolis
{fpayan,am}@i3s.unice.fr

ABSTRACT

We propose a new compression method devoted to large structured hexahedric meshes having discontinuities. It is dedicated to applications such as visualization or physical simulations whose management by any workstation or mobile device with limited memory and bandwidth is critical. Our method relies on a multiresolution analysis that generates a hierarchy of meshes at increasing resolutions. Our technique also uses a discontinuity tracking feature for their preservation, whatever the resolutions, and consequently maintains a coherent geometry with respect to the original mesh. Experimental results emphasize the quality of our compression, in terms of both geometrical distortion and compression ratio.

Index Terms— Hexahedric mesh, compression, multiresolution analysis, geometrical discontinuities, upscaling

1. INTRODUCTION

Hexahedric meshes are commonly manipulated in geosciences [1]. They are for instance used by geologists to study flow simulations, and benefit from an increasing interest for geologic modelling [2]. With huge progresses in data acquisition in the past decade, digitization of physical terrains becomes increasingly more accurate, and acquisition devices are now able to produce ultra-high resolution geometric point clouds. The resulting tremendous quantity of data prominently impacts the memory size required for their storage, but also their transmission and their transfer. Consequently, it greatly affects the simulation interactivity that is necessary to exploit such an instance of big data [3]. For all these reasons, compression algorithms [4, 5] are inevitable and should allow a progressive decompression that is adapted to a broad range of applications, starting with visualization for instance. We propose (H)exaShrink as a novel compression scheme dedicated to structured hexahedral meshes. It generates a hierarchy of meshes at increasing levels of resolution, and strives to maintain a geometrical coherency over the resolutions. Hence our scheme also preserves geometrical discontinuities. Despite the growing interest for this type of data, structured hexahedric meshes have not benefited from the same attention as unstructured or surface meshes [6, 7].

The paper is structured as follows: Section 2 presents the specificities of structured hexahedric meshes. Section 3 briefly reviews prior methods for volumetric mesh compression [8], before introducing our structured mesh compression technique in Section 4. Section 5 presents several visual results, and quantitative comparisons with a reference compression technique. Finally, Section 6 summarizes our contributions and proposes future works.

2. CONTEXT

2.1. Representation of data geometry

A volume or volumetric mesh is the numerical representation of a 3D object, area, architecture, etc. This representation is composed of polyhedra (tetrahedra, hexahedra), and is the counterpart of 3D surface representations, described by polygons (triangles or quads). Thereby, the set of polyhedra, called *cells*, defines the grid that tiles the 3D domain. In the case of an hexahedral mesh, each cell (sometimes denoted hex or brick) possesses three different kinds of simplices: 6 *faces* (quads), 12 *edges*, and 8 *vertices*, which may sometimes be degenerated.

2.2. Structured meshes

Structured meshes are generally described with the *Corner-Point Grid* format [9], an hexahedron tessellation of an Euclidean 3D volume. This format consists of a set of hexahedral cells topologically aligned in a Cartesian regular lattice, so that the cells can be indexed by the triplet (i, j, k). It is sometimes referred to as a pillar grid, because it is based on a set of pillars/coordinate lines running from the top to the bottom of the model. Grid cells are defined by eight corners/nodes which lay pairwise on four neighbouring pillars, as presented in Figure 1.

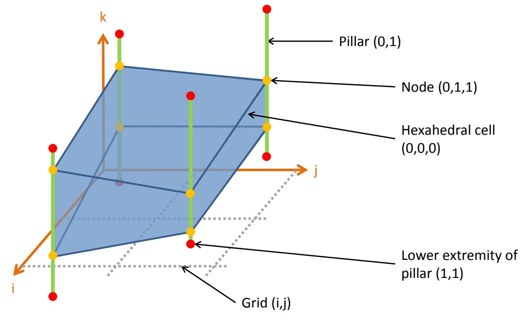
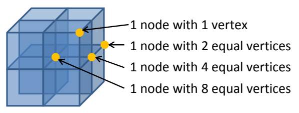
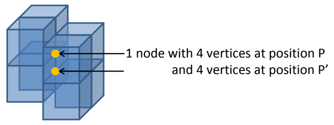


Fig. 1. Simplest structured hexahedral volume mesh, described by pillars, with dimensions $[1, 1, 1]$.

Geo-scientific data often contains geometrical discontinuities that physically correspond to geological faults. These faults highlight a vertex disparity in space at the same node. Indeed, in a free-fault area, one node is associated to at most 8 identical vertices, whereas, in a faulty area, faults stretch the geometry vertically along pillars, and therefore the node has different 3D vertices coordinates, as depicted in Figures 2(a) and 2(b), respectively. A real hexahedric model with $128 \times 128 \times 128$ cells, which is composed of a geological fault is presented in Figure 3.


1 node with 1 vertex
1 node with 2 equal vertices
1 node with 4 equal vertices
1 node with 8 equal vertices

(a) Free-fault area.


1 node with 4 vertices at position P
and 4 vertices at position P'

(b) Area with a vertical fault.

Fig. 2. Illustration of a free-fault and a faulty area.

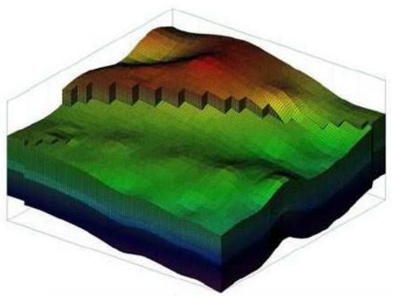


Fig. 3. Rendering of a model with dimensions $[128, 128, 128]$.

3. STATE OF THE ART

Without loss of generality, the compression techniques can be separated according to the type of structure they are able to handle, and which are briefly presented below.

3.1. Unstructured mesh compression

3.1.1. Progressive algorithms

Progressive algorithms aim at representing the original surface with several resolution meshes, from coarse to fine. Thus, a hierarchy of meshes at different resolutions is created, which is well-suited to transmission or visualization for instance. To the best of our knowledge, the literature only covers the case of progressive compression for tetrahedra [10, 11].

3.1.2. Streaming algorithms

Another type of compression schemes appeared to alleviate the issue of gigantic volume meshes, which cannot entirely fit into the core memory [12]. The streaming process represents the mesh in an ordered interleaved set of vertices, hexahedra, and *finalization tags*. Basically, a finalization tag explicitly indicates to the coder/compressor that a given vertex will not be referenced anymore in the stream, and thereby the allocated structure for this vertex can be deallocated, to efficiently minimize the memory footprint.

3.2. Structured mesh compression

To the best of our knowledge, the only method that manages the compression of structured meshes is the one proposed by L. Chizat [13]. It realizes a multiresolution analysis to compress, in a lossless manner, pillar grid structured hexahedral meshes, while ensuring the preservation of geometrical discontinuities over the resolutions. It is mainly based on an instance of a morphological wavelet transform [14] which is dedicated to the preservation of discontinuities, and uses a gzip entropy encoder to produce the compressed mesh.

Although this technique provides reliable and consistent representations, it is restricted to meshes with simple fault networks (*i.e.* non intersecting, isolated or linear faults).

4. METHODOLOGY: 3D HEXAHEDRIC MESH COMPRESSION

Our work on the geometry compression focuses on the Z coordinates of the cell nodes along the pillars in a structured hexahedral mesh.

The key idea consists in representing the tri-dimensional matrix of Z coordinates with less coefficients, while preserving the coherency of representation and reconstruction. First of all, an approximation coefficient is computed for each group of dimensions $[N, N, N]$ of original coefficients, whereas the $N^3 - 1$ other coefficients within the original group are updated to represent the lost details. Figuratively speaking, the original group of coefficients at resolution L is split into two subgroups, as illustrated on top of Figure 4: the approximation coefficient, and a set of updated coefficients called detail coefficients. Note that only the approximation coefficient is part of the Z coordinates matrix at the resolution $L-1$, as it represents an approximation of the original group of coefficients. On bottom of Figure 4 is depicted the inverse operation, which allows to reconstruct the original group of coefficients at resolution L using the approximation coefficient at resolution $L-1$, plus the subgroup of detail coefficients.

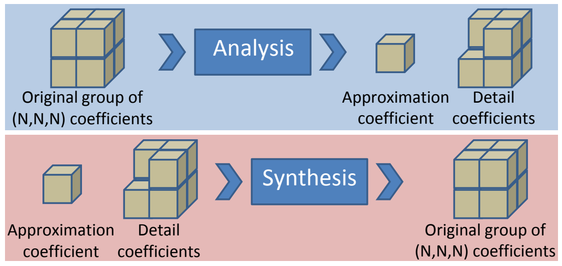
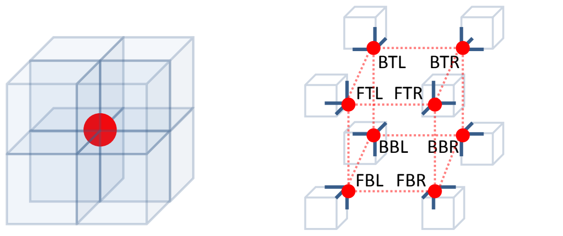

Analysis
Original group of (N,N,N) coefficients
Approximation coefficient
Detail coefficients
Synthesis
Approximation coefficient
Detail coefficients
Original group of (N,N,N) coefficients

Fig. 4. Multiresolution analysis principle applied on a group of coefficients with dimensions $[2, 2, 2]$.

Basically, the geometry compression algorithm follows the wavelet analysis [15] principle given in Figure 4, where the approximation coefficient is computed using the three-stage process described below:

1. **Removing the redundancy from the Z coordinates matrix**. Using the vertex naming convention presented in Figures 5(a) and 5(b), the 8 vertices of a given node can be differentiated according to their positions [Back/Front Bottom/Top Left/Right] with respect to this node. Given that there is no vertical fault within a mesh (*i.e.* there is no vertical gap between any two adjacent layers of cells), the following equation logically turns out:

$\forall$ node $N_{i,j,k}$, $\forall\,(X, Z) \in$ ([Back/Front] $\times$ [Left/Right]), vertex $XBZ_{i,j,k}$ = vertex $XTZ_{i,j,k}$.
In other words, each of the four top vertices for every node has the same Z coordinate as its counterpart bottom vertex. Consequently the quantity of information in the matrix Z can be halved without any loss of information, due to this redundancy. So from now on, every node $N_{i,j,k}$ is only represented by the Z coordinates of its bottom vertices $BBR_{i,j,k}$, $FBR_{i,j,k}$, $BBL_{i,j,k}$ and $FBL_{i,j,k}$, as its top vertices Z coordinates are removed from the Z matrix.



(a) A node and its 8 surrounding cells. (b) Splitting view of the node into its 8 vertices.

Fig. 5. Vertex naming convention with [Back/Front Bottom/Top Left/Right].

2. **Fault segmentation within the original mesh**. This phase uses the four bottom vertices $BBR_{i,j,k}$, $FBR_{i,j,k}$, $BBL_{i,j,k}$ and $FBL_{i,j,k}$ of each node contained in the Z matrix to detect the fault node configuration within the dozen of possible configurations (free-fault, straight, corner, T or cross), which are presented in a 2D schematic top view in Figure 6. So, each configuration consists of the four orientations of the cardinal axes: north, south, east and west, which are either active or inactive. For instance, the T-north configuration has its south axis inactive, the three remaining ones are active.

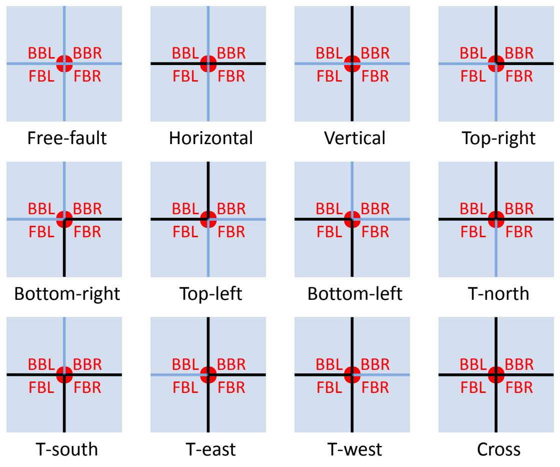


Fig. 6. 2D top view presentation of the 12 possible oriented fault configurations at a given node in black (black and light blue lines represent, respectively, faults and cell borders).

Assuming that the fault configuration is Z-invariant (*i.e.* the nodes belonging to the same pillar present the same fault configuration), a single 2D configuration map is sufficient to represent the fault configuration of the whole mesh. Figure 7 illustrates the segmentation realized on the mesh, by studying the Z coordinates of the four vertices $BBR_{i,j,k}$, $FBR_{i,j,k}$, $BBL_{i,j,k}$ and $FBL_{i,j,k}$ for every node $N_{i,j,k}$.

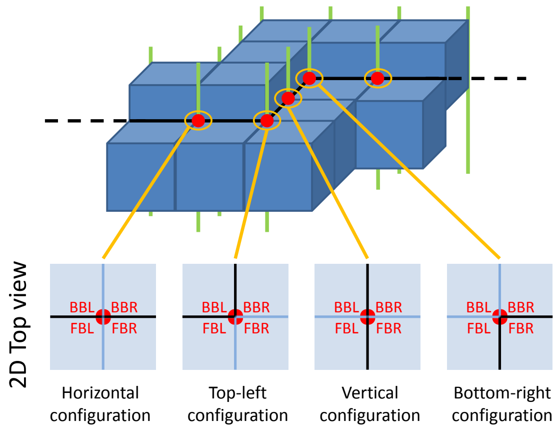


Fig. 7. Fault segmentation within the original mesh (pillars are green-colored).

3. **Morphological wavelet transform**. The previous segmentation guides the analysis to preserve faults all over the process. To predict the configuration of the resulting node at the resolution $L-1$, the fault configuration of the four nodes at resolution L is studied. The resulting configuration contains the north axis if the configurations of the top nodes contain at least one north axis. By repeating the procedure for the south, east and west axes, the fault configuration of the resulting node is predicted as illustrated in the example of Figure 8. Finally, from this prediction, the node whose configuration minimizes its distance with the predicted one, corresponds to the aforementioned approximation coefficient, which will be part of the Z matrix at resolution $L-1$.

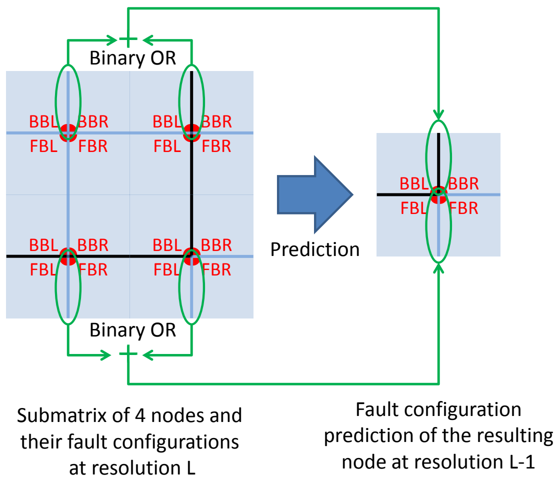
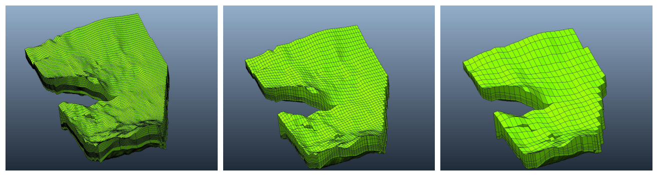
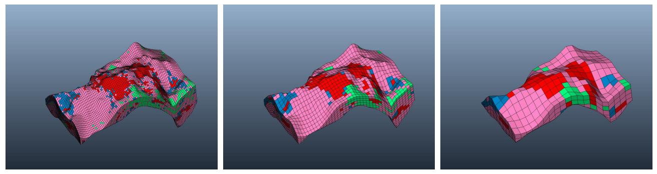

Binary OR
BBL BBR FBL FBR
Prediction
Binary OR
Submatrix of 4 nodes and their fault configurations at resolution L
Fault configuration prediction of the resulting node at resolution L-1

Fig. 8. Prediction of the fault node configuration at resolution $L-1$ from the fault node configurations of 4 nodes at resolution L (black and light blue lines represent, respectively, faults and cell borders).

5. RESULTS

Figure 9 presents results obtained from three real geoscience meshes unpacked at three resolution levels. The two last meshes are decorated with categorial properties associated to their geometry, depicted with different colors. Because the proposed hierarchical mesh coder is lossless, the reconstruction at Res. 0 corresponds to the original mesh.


(a) Mesh#1 — Res. 0. (b) Res. −1. (c) Res. −2.


(d) Mesh#2 — Res. 0. (e) Res. −1. (f) Res. −2.

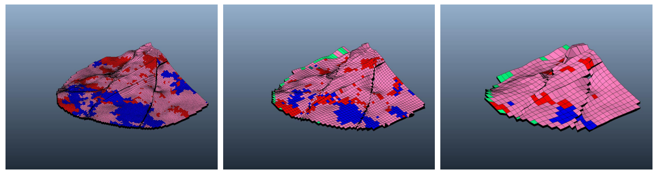
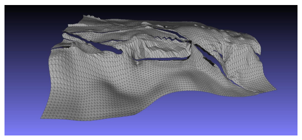

(g) Mesh#3 — Res. 0. (h) Res. −1. (i) Res. −2.

Fig. 9. Three levels of resolution generated with (H)exaShrink on three geological meshes with respective dimensions (from top to bottom) $[100, 100, 21]$, $[80, 45, 26]$, $[100, 95, 100]$.

We observe that the geometry is preserved and is coherent over the resolutions. To assess this observation objectively, Root Mean Square Error (RMSE) levels between our (lossy) compressed meshes and the original ones are computed, and compared to those obtained with regard to resulting meshes from the reference compression technique JPEG2000 3D [16] using the open source OpenJPEG codec [17] and with lossless settings. In order to be as fair as possible, only the top layer surface meshes have been compared (see Figure 10), because JPEG2000 3D is not dedicated to deal with geological data. Figure 11 summarizes the error measures and clearly shows that our meshes are first of all consistent with respect to the original meshes, and second of all, outperform those generated by the prior JPEG2000 3D compression scheme. Indeed, our method was dedicated to preserve faults over the resolutions.

To further illustrate the benefit of our method, Table 1 presents compression ratios between losslessly compacted and original mesh data. It conveys an idea of the memory usage reduction, comparing a direct encoding of the geometry with gzip or bzip2 entropy coders and with our proposed wavelet pre-processing. It clearly demonstrates that the wavelet transform drastically sparsifies the mesh structure and reduces the amount of physically stored information.


(a) Original top layer surface.

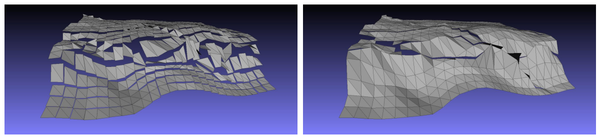

(b) Res. −2 JPEG2000 3D. (c) Res. −2 (H)exaShrink.

Fig. 10. Mesh#3 (portion): Original top layer surface (a), third resolution level from JPEG2000 3D [16] (b) and (H)exaShrink (c).

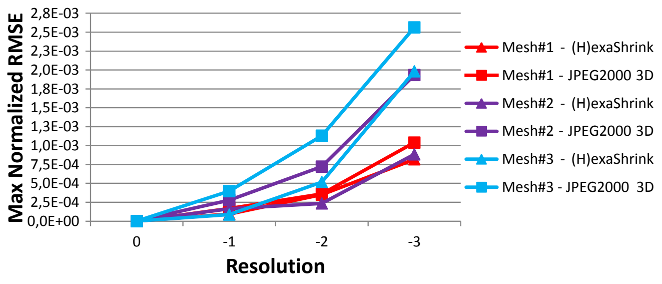


Fig. 11. RMSE computed on the top layer surfaces with respect to the original ones, and compared to the prior JPEG2000 3D compression scheme [16].

EE / WT	gzip	bzip2 with size block of		
		0.10 MB	0.50 MB	0.90 MB
None	3.09	2.33	3.28	3.51
Our wavelet	3.11	4.05	4.19	4.25

Table 1. Compression ratios for different entropy encoding (EE) techniques, without or with wavelet (WT) pre-processing.

6. CONCLUSION AND PERSPECTIVES

We presented a lossless progressive compression technique that handles large structured hexahedric meshes having discontinuities, by creating a hierarchy of meshes at different resolutions. Besides the entropy encoding, the resolution meshes can easily be generated and loaded at user's convenience, thanks to the use of wavelet transform. Moreover, numerous applications are still nowadays not robust enough to manipulate large amounts of data, and our compression technique circumvents several issues such as in **mesh upscaling** for instance. Indeed, this method is commonly used in fluid flow computation [18] to limit simulation runtime, which might take days or even weeks in case the original mesh is too dense, and consists in building a lower resolution mesh, from a denser one.

To efficiently resolve this problem and avoid the research using some tree structures that might require a huge amount of core memory, a wavelet decomposition of the finest resolution mesh, would generate several meshes at different resolutions, from coarse to fine.

The authors thank L. Chizat for initial work on mesh compression.

7. REFERENCES

[1] M. Perrin and Rainaud J.-F., Eds., *Shared Earth Modeling. Knowledge Driven Solutions for Building and Managing Subsurface 3D Geological Models*, Editions Technip, 2013.

[2] G. Caumon, G. Gray, C. Antoine, and M.-O. Titeux, "Three-dimensional implicit stratigraphic model building from remote sensing data on tetrahedral meshes: Theory and application to a regional model of La Popa basin, NE Mexico," *IEEE Trans. Geosci. Remote Sens.*, vol. 51, no. 3, pp. 1613–1621, Mar. 2013.

[3] R. K. Perrons and J. W. Jensen, "Data as an asset: What the oil and gas sector can learn from other industries about "big data"," *Energy Pol.*, vol. 81, pp. 117–121, Jun. 2015.

[4] M. Nelson and J.-L. Gailly, *The Data Compression Book*, Wiley, 1995.

[5] D. Salomon and G. Motta, *Handbook of Data Compression*, Springer, 2009.

[6] F. Payan and M. Antonini, "Compression of surface meshes," in *Biomedical Diagnostics and Clinical Technologies: Applying High-Performance Cluster and Grid Computing*, chapter 5, pp. 163–180. IGI Global, 2010.

[7] C. Mazetto Mendes, K. Apaza-Aguero, L. Silva, and O. R. P. Bellon, "Data-driven progressive compression of colored 3D mesh," in *Proc. Int. Conf. Image Process.* sep 2015, Institute of Electrical & Electronics Engineers (IEEE).

[8] F. Dupont, G. Lavoué, and M. Antonini, "3D mesh compression," in *3D Video from Capture to Diffusion*, L. Lucas, C. Loscos, and Y. Remion, Eds. Wiley-ISTE, 2013.

[9] K.-A. Lie, *An Introduction to Reservoir Simulation Using MATLAB - User Guide for the Matlab Reservoir Simulation Toolbox (MRST)*, Sintef ICT, Department of Applied Mathematics, 2014.

[10] R. Pajarola, J. Rossignac, and A. Szymczak, "Implant sprays: Compression of progressive tetrahedral mesh connectivity," in *Proc. IEEE Visualization Conf.*, San Francisco, CA, USA, Oct. 29, 1999, pp. 299–305.

[11] E. Danovaro, L. De Floriani, M. T. Lee, and H. Samet, "Multiresolution tetrahedral meshes: An analysis and a comparison," in *Proc. Shape Modeling International*, Banff, Canada, May 17-22, 2002, pp. 83–91.

[12] C. Courbet and M. Isenburg, "Streaming compression of hexahedral meshes," *Vis. Comput.*, vol. 26, no. 6-8, pp. 1113–1122, 2010.

[13] L. Chizat, "Multiresolution signal compression: Exploration and application," M.S. thesis, ENS Cachan, Jul. 2014.

[14] F. Arandiga, A. Cohen, R. Donat, N. Dyn, and B. Matei, "Approximation of piecewise smooth functions and images by edge-adapted (ENO-EA) nonlinear multiresolution techniques," *Appl. Comput. Harmon. Analysis*, vol. 24, no. 2, pp. 225–250, 2008, Special Issue on Mathematical Imaging — Part II.

[15] L. Jacques, L. Duval, C. Chaux, and G. Peyré, "A panorama on multiscale geometric representations, intertwining spatial, directional and frequency selectivity," *Signal Process.*, vol. 91, no. 12, pp. 2699–2730, Dec. 2011.

[16] ITU-T T.809, "JPEG2000 image coding system: Extensions for three-dimensional data," May 2011, ISO/IEC 15444-10:2011.

[17] "OpenJPEG: An open-source JPEG 2000 codec written in C," Version 2.1.0.

[18] T. Arbogast, "Numerical subgrid upscaling of two-phase flow in porous media," in *Numerical Treatment of Multiphase Flows in Porous Media*, Z. Chen, R. E. Ewing, and Z.-C. Shi (16), Eds., pp. 35–49. Springer, 2000.