Modeling and Visualization Approaches for Time-Varying Volumetric Data

Kenneth Weiss[1] and Leila De Floriani[2]

[1] Department of Computer Science, University of Maryland, College Park, MD ⋆⋆⋆
[2] Department of Computer Science, University of Genova, Genova, Italy †

Abstract. Time-varying volumetric data arise in a variety of application domains, and thus several techniques for dealing with such data have been proposed in the literature. A time-varying dataset is typically modeled either as a collection of discrete snapshots of volumetric data, or as a four-dimensional dataset. This choice influences the operations that can be efficiently performed on such data. Here, we classify the various approaches to modeling time-varying scalar fields, and briefly describe them. Since most models of time-varying data have been abstracted from well-known approaches to volumetric data, we review models of volumetric data as well as schemes to accelerate isosurface extraction and discuss how these approaches have been applied to time-varying datasets. Finally, we discuss multi-resolution approaches which allow interactive processing and visualization of large time varying datasets.

1 Introduction

In the last few years, there has been an increasing trend in scientific visualization toward the generation and interaction with very large time-varying volume data sets. Such data sets are used as basic tools for analyzing the dynamics and the evolution of phenomena in a variety of application domains, including medicine, meteorology, astrophysics and engineering.

Time-varying volume data sets are sets of points in the 3D Euclidean space describing one or more scalar quantity (e.g., pressure, temperature, strength of an electric or magnetic field) at different instances of time. Approaches to dealing with time-varying data differ in their treatment of the temporal dimension. Values in local regions tend to change slowly over short intervals of time. This temporal coherence can be exploited in interactive applications by efficiently encoding these small changes, thus minimizing storage and retrieval costs. A common metaphor for this approach is that of a video, where users can gain insight into the dataset by 'playing' the discrete snapshots of the volume over time. Decoupling the spatial and temporal components can be advantageous when the spatial and temporal resolutions differ greatly or where there is a higher degree of spatial coherence in a local region than there is temporal coherence, or

⋆⋆⋆ *kweiss@cs.umd.edu*
† *deflo@disi.unige.it*

vice versa. An alternative approach is to treat the temporal and spatial dimensions in a homogeneous way. Although this approach can be more complex to conceptualize, it offers several advantages. Since time is assumed to be continuous, smoother animations can be performed by interpolating the field values. Additionally, the correspondence between time steps can be exploited to track features over time.

Most of the techniques for visualizing time-varying volume data are based on direct volume rendering (DVR) or isosurface and interval volume extraction. Specific algorithms have been developed for performing such tasks also on 4D hypercubic grids, or 4D simplicial meshes.

The huge size and complexity of available time-varying data sets poses interesting challenges for inspecting, analyzing and visualizing such data, as the underlying domain is typically at a much higher resolution than one which could be interactively processed or meaningfully analyzed. This naturally leads to the investigation of hierarchical methods to control and adjust the level of detail of a given data set, the so-called *multi-resolution models*. A multi-resolution model encompasses several representations of the same shape at a virtually continuous range of resolutions and facilitates the efficient retrieval of variable-resolution representations of the shape. Multi-resolution approaches have successfully been applied to large data sets describing 3D shapes, as well as 2D and 3D scalar fields discretized through simplicial meshes (see [1, 2] for surveys). Reducing the geometry in less 'interesting' areas allow users to focus on a region of interest, thus achieving lower storage costs and better performance.

The remainder of the paper is organized as follows. We review some background notions in Section 2. Representations for efficient isosurfaces and interval volume extraction from time-varying volume data are discussed in Section 3. Indexing techniques proposed for speeding up isosurface extraction and rendering are reviewed in Section 4. In Section 5, we provide an overview of multi-resolution approaches for both 3D and time-varying scalar fields. In Section 6, we summarize the presented techniques and discuss directions for future work.

2 Background Notions

Time-varying volumetric data sets are given as sets of points in 3D space with which one or more scalar values are associated at different instants in time. The points are either at the vertices of a regular grid, or they can be irregularly distributed. These data sets can be either interpreted as collections of 3D data sets or as four-dimensional ones. In the latter case, each point has four coordinates and one or more scalar values are associated with each point in 4D space.

Time-varying data sets are modeled through a decomposition of the 3D (or 4D) domain into a collection of non-overlapping *cells* with vertices at the data points, forming a mesh. The mesh can be a *regular* grid, and thus the 3D (4D) cells are cubes (hyper-cubes), or a *simplicial mesh*. In the first case, a tri-linear (quadri-linear) interpolant is used over the 3D (4D) cells, while in the second case the field values are linearly interpolated.

A k-simplex is the locus of the points in $\mathbb{E}^d$ that can be expressed as the convex combination of $k+1$ affinely independent points. A *simplicial mesh* is a collection of k-simplexes where the intersection between two simplices is empty or a face of both simplices.

In isosurfacing applications, a surface passes through a cell of a mesh if the field values at some of its vertices are greater than the isovalue, and below at others. Such a cell is considered *active*. For high-quality meshes, care must be taken to ensure that the extracted mesh components agree at cell junctions. An *interval volume* is a generalization of an isosurface and is the sub-volume enclosed between two isosurfaces.

3 Isosurface and Interval Volume Extraction Techniques

Several algorithms have been developed for extracting isosurfaces and interval volumes from 3D scalar fields. These algorithms typically assume a decomposition of the domain of the field into a set of disjoint cells and compute a conforming mesh by extracting *patches* of the isosurface or of the interval volume from each active cell. Patch vertices are computed along the intersected edges of the cell using linear interpolation, and the patch is triangulated via a lookup table based on the relative values of the field at the vertices of the cell with respect to the isovalue. When generating the table, symmetry and topology considerations can be used to reduce the table size and to ensure correctness of the algorithm. The Marching Cubes (MC) algorithm [3] and its variants (e.g. [4] for interval volumes) operate on cubic grids, while the Marching Tetrahedra approach [5] and its variants (e.g. [6] for interval volume) operate on tetrahedral meshes. These latter can be generated from irregular grids or from regular grids by decomposing cubic cells into five or six tetrahedra.

For the 4D case, marching methods have been extended to regular grids using cell-based lookup tables [7, 8]. Roberts and Hill [7] reduce the exponential growth in the number of cases by decomposing isosurface cases into a set of patches containing a single connected component. Each case in their lookup table is then composed of a set of 'sub-cases' containing a single connected component.

Bhaniramka et al. [8] propose an alternative case-based isosurfacing method for hypercubic as well as convex cells in arbitrary dimension. They reduce the computational and storage costs for infrequent cases through a lazy evaluation scheme that creates cases on the fly and caches the more common cases.

Weigle and Banks [9] provide an isosurfacing technique for simplicial meshes. They decompose hypercubic cells into *pentatopes* (i.e. 4-simplexes) and then apply a constraint-based scheme to extract $(d-1)$-dimensional simplices from d-simplices. Thus, by specifying an isovalue for the 4D dataset, they extract a tetrahedral mesh containing the *envelope* of all isosurfaces swept by this isovalue. By further specifying a time value, they extract a specific isosurface from the tetrahedral mesh. Although working in the higher dimensional space enables smoother interpolation, their initial decomposition of each hypercube into 192 pentatopes leads to a significant overhead.

Algorithms have also been proposed to extract interval volumes from 4D meshes. In [10, 11], interval volumes are used to segment a volume data set and several new interval volume rendering techniques are presented. They generate interval volume cases via a projection from $(d+1)$-dimensional isosurface cases onto d-dimensional interval volume cases. This is accomplished by treating the range of the scalar field as an extra dimension when generating the cases. For instance, they compute interval volumes for 3D cubic cells by isosurfacing 4D cells. However, when this dimension-lifting approach is applied to simplicial meshes, cases for d-simplices are generated from prisms in $(d+1)$-space rather than $(d+1)$-simplices. Ji et. al. [12] apply a similar dimension-lifting technique to track the evolution of isosurfaces, such as the creation, splitting and merging of surfaces, in volumetric and time-varying datasets. They generate hypercubic interval volume cases from 5-dimensional hypercubes.

4 Indexing Approaches

Methods that exploit the spatial and temporal coherence within the data can lead to significantly more efficient isosurface extraction algorithms. For a given isovalue, only a small subset of the domain contains active cells. Thus, precomputed indexes on the field values within cells enable a more efficient isosurface extraction than a brute force search.

Indexing schemes have been developed for 3D scalar fields. There are two predominant indexing techniques: *hierarchical spatial indexing* and *value indexing*. The most widely-used spatial indexing technique is Wilhelms and Van Gelder's *Branch-On-Need Octree (BONO)* [13]. This method applies a hierarchical spatial partitioning of the domain into an octree and associates with each node of the octree the maximum and minimum field values of its children. BONO efficiently handles datasets whose dimension are not powers of 2, and uses a caching mechanism to minimize field value lookups of neighboring cells. In contrast, the value-based partitioning approaches [14–16] attempt to minimize isosurface extraction time by reorganizing the data into a two-dimensional *span space* on the range of the field. In span-space techniques, each cell is projected into a two-dimensional point whose coordinates are the minimum and maximum field values spanned by the cell. Efficiency is achieved through optimized data structures on the span space rather than the spatial coordinates of the cells. The NOISE algorithm [14] uses a kD tree to efficiently extract an isosurface from a 3D volumetric data set in $O(\sqrt{n}+k)$ time, where n is the total number of cells, and k is the number of active cells. ISSUE [15] improves the performance of the NOISE algorithm and provides a parallel algorithm by subdividing the span space into an axis-aligned lattice where each tile contains approximately the same number of cells. Interval trees [17] are another efficient method for performing range queries, and have been used by van Kreveld [18] to extract iso-contours from triangulated terrain data. Cignoni et al. [16] use interval trees to index cells in span space, and prove that their iso-surface extraction algorithm operates in optimal $O(\log n+k)$ time.

Models for time-varying data often use the indexing schemes described above, but differ in their method of exploiting the spatial and temporal coherence within the data. The *Temporal Branch-on-Need octree (T-BON)* [19] exploits the spatial coherence of the data by creating a BON octree for each time step. By using the same branching structure for all time steps, the T-BON amortizes the storage overhead of the octree structure. It stores the data for each time step independently, and does not incorporate any optimization for exploiting the temporal coherence. More recently, the *Persistent Octree (POT)* [20] uses a compact 3D or 4D octree to store the active cells for all isosurfaces. It treats time as a spatial dimension, and extracts all cells that contain the isovalue and intersect a temporal hyperplane. It has slightly quicker rendering times than those of BONO but exhibits a significant memory overhead.

The *Temporal Hierarchical Index Tree (THIT)* [21] projects the cells from each time step into span space and coalesces cells with high temporal coherence using a binary tree. The cells in each node of the binary tree are then indexed in span space using a variation of the ISSUE [15] and interval tree [16] algorithms. Chiang's approach [22] is similar to THIT but it is catered for out-of-core data access. This approach uses a cache-oblivious time tree whose nodes are stored out-of-core. In an attempt to improve I/O efficiency, spatially coherent cells are clustered into *meta-cells*, containing the field values for all time-steps. Vrolijk et. al. [23] also use an out-of-core variation of the THIT to quickly render approximate isosurfaces. They note that accessing the field values from disk can be a significant bottleneck in isosurface rendering algorithms and develop a point splatting algorithm that uses only the min-max range of a cell. Waters et. al. [24] use the video metaphor as well as a modified THIT to efficiently encode the difference between time steps. This technique enables quick random access into the dataset through appropriately placed key frames.

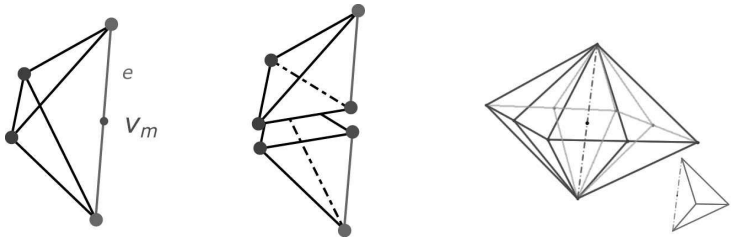
It is important to note that the speedup associated with indexing techniques depends heavily on the dataset due to their assumption of coherence in the dataset and in data access patterns.

5 Multi-resolution Approaches

Multi-resolution approaches can be subdivided into two categories: *mesh-based* approaches, where the approximation is guided by the mesh refinement, and *wavelet-based* approaches, where the multi-resolution behavior is determined by the space of functions.

A class of multi-resolution models which has been extensively studied in the literature is those based on a nested decomposition of the field domain. A *nested mesh* is a mesh in which the cells are defined by the uniform subdivision of a d-dimensional cell into scaled copies. The most common examples of nested meshes are quadtrees, octrees and their d-dimensional generalization. The major problem with such models is the intrinsic difficulty in extracting highly-adaptive representations [1, 2]. Topologically consistent meshes are extracted by constraining two adjacent cells to differ by at most one level in the subdivision.

Nested multi-resolution models generated through longest edge simplicial bisection provide more flexibility. The *bisection rule* for a d-simplex σ in a d-dimensional mesh Σ consists of replacing σ with the two d-simplexes obtained by splitting σ at the middle point $\mathbf{v}_m$ of its longest edge $\mathbf{e}$ and by the hyperplane defined by $\mathbf{v}_m$ and the vertices of σ which are not endpoints of $\mathbf{e}$ (see Figure 1(a)). When this rule is applied recursively to an initial decomposition



(a) Longest edge bisection (b) Diamond in 3D

Fig. 1. (a) A simplex is bisected along the hyperplane containing the midpoint $\mathbf{v}_m$ of its longest edge $\mathbf{e}$ and all vertices not adjacent to $\mathbf{e}$. (b) The set of simplices sharing a common longest edge (dashed) of a simplex (lower right) form a *diamond*.

of the d-dimensional hyper-cubic domain into $d!$ simplexes, it generates a nested mesh, denoted as a *hierarchy of d-simplexes*. Hierarchies of triangles are used as the basis for terrain visualization [25, 26], and hierarchies of tetrahedral meshes have been applied to volume data visualization (see [2] for a survey). When bisecting a d-simplex σ, all d-simplexes sharing the common longest edge with σ must be split at the same time to ensure a conformal mesh. The set of d-simplexes which share their split edge form a *diamond* (see Figure 1(b)). In [26, 27], efficient data structures are proposed for hierarchies of triangles and tetrahedra, respectively, based on the implicit encoding of the diamonds and their dependency relations as a Directed Acyclic Graph (DAG). In [28] a technique is proposed for extracting nested tetrahedral meshes without cracks based on an $O(1)$ neighbor finding algorithm applied to the hierarchy of tetrahedra.

To the best of our knowledge, there have not been any 4D octree approaches treating the temporal dimension in the same manner as the spatial dimensions. Shen et al. [29] indicate that such approaches would suffer by not being able to properly exploit the coherence among each component. They instead propose the *Time-Space Partitioning (TSP) tree* [29], which exploits the spatial and temporal coherences within the dataset. The TSP tree creates an octree partitioning over the 3D domain whose nodes are binary trees containing the temporal values. This organization allows efficient incremental DVR of the dataset since cells whose values do not change across time steps do not need to be updated.

Gregorski et al. [30] use the video metaphor for time-varying data but represent the spatial component of the data as a hierarchy of tetrahedra. This allows them to have level-of-detail control over the extracted mesh in each frame, but

limited frame to frame coherence. To advance to the next frame, they need to refine/coarsen each node of the previous frame's mesh. Instead of refining neighboring geometry to ensure a conformal mesh, they use a lookup table to locally subdivide tetrahedra that do not align with their neighbors. They claim that this improves their run-time performance due to the increased parallelizability of the algorithm.

Lee et al. [31] extend their tetrahedral neighbor finding algorithm [28] to 4D hierarchical datasets, which they denote as a Hierarchy of Pentatopes. This allows constant time computation of diamonds in 4D and thus enables the extraction of conforming adaptive meshes in 4D.

Recently, Ponchio and Hormann [32] introduced an interactive multiresolution volume rendering algorithm for irregular time-varying datasets. They first apply the algorithm by Bhaniramka et al. [10] to extract a 4D tetrahedral isosurface from the dataset at full resolution. Then they apply the *Batched Multi-Tessellation (BMT)* algorithm [33] to create a multi-resolution structure on the isosurface and convert the faces of each tetrahedron in 4D space into so-called *dynamic triangles*. The dynamic triangles in each patch are then projected into span space and rendered on the GPU. Since this method first extracts the tetrahedral isosurface at full resolution and then computes the multi-resolution structure on this mesh, the generation time can be quite inefficient. For instance, a $374^3 \times 600$ dataset takes more than 13 days to pre-process. However after pre-processing has completed, their algorithm can render at a rate of 20M triangles/sec regardless of the input dataset.

Wavelets offer another multi-resolution approach to scalar field modeling, which can be applied directly to the scalar field [34, 35]. Although wavelet techniques such as [36] can be applied to an extracted surface, this requires generation and storage of the mesh at full resolution as well as a calculation of the wavelet coefficients before the mesh can be simplified. Westermann [34] proposed a lossy wavelet projection scheme to speed up the volume rendering integral calculation while maintaining accurate error bounds and reducing memory requirements. If separable bases are used, this method can be directly applied to higher dimensional scalar fields. Linsen et. al. [35] present a hierarchy of B-spline filters for n-variate multi-resolution analysis, which they call $\sqrt[n]{2}$-subdivision, where each finer level of resolution only doubles the size of the mesh (an approach formalized in [37] and related to the diamonds introduced above). They use the lifting scheme [38] as an in-place, lossless family of bases with narrow support to achieve an adaptable level-of-detail representation of the data.

Wang et al.'s *Wavelet-based Time-Space Partitioning Tree (WTSP)* [39] extends the TSP tree [29] with wavelet coefficients. They use a distributed algorithm that reduces the dependency between processors for wavelet reconstruction by redundantly storing copies of the reconstructed values at deeper levels (in a manner similar to keyframes in videos).

Wavelet methods offer an advantage in the fidelity of the data at higher scales, but they may require expensive runtime reconstruction and loss of adaptability.

Since they downsample using low-pass filters, they can avoid aliasing effects introduced by other multi-resolution methods.

The choice of basis can have dramatic effects on reconstruction time and accuracy. A narrow filter requires fewer data samples to be resident in memory, but might introduce reconstruction artifacts. Linsen et al. [35] suggest that bilinear B-spline wavelets offer the best space-time tradeoff.

6 Concluding Remarks

Due to the vast size of time varying volume datasets, most of the existing data structures are optimized for specific operations such as isosurface extraction or direct volume rendering. Such methods typically involve reorganizing the layout of the data, as in value-based indexing schemes or augmenting the data with additional information, as in the spatial indexing schemes. These approaches achieve efficiency by exploiting the spatial and temporal coherences within the data, but their utility can vary depending on how well the data fits the expected distribution.

The video metaphor [30, 24] is appropriate when the goal is to understand the evolution of features within the dataset. However, since the correspondence between the time steps is not kept, it can be difficult to quantify the changes. In contrast, 4D approaches [9, 7, 8] directly encode these correspondences. As such, tetrahedral isosurfaces cover the envelope of triangular isosurfaces over all time steps within the dataset. Triangular isosurfaces can then be extracted and interpolated from these envelopes along cutting planes in the temporal (or other) dimension.

Since active cells are usually distributed throughout the dataset, value-based indexing schemes tend to be among the most efficient ones for isosurface rendering. These approaches are essentially equivalent to hashing, and as such, the spatial relationships between cells of the data are lost. Thus, while these approaches are very efficient at rendering the data, they are less appropriate when adjacency relationships are required in downstream applications which involve navigation.

A multi-resolution model offers an exciting approach to time varying datasets. It allows not only a reduction of the size of the underlying representation but also provides the possibility of inspecting a field at variable resolutions in the domain, according to specific application needs. Bertram et al. [40] note that while thresholded and quantized wavelet compression algorithms are effective in minimizing the memory requirements and field error, they offer no guarantees of maintaining the topology of extracted isosurfaces. Thus, wavelet methods are generally superior in direct volume rendering applications, while mesh based multi-resolution techniques are more appropriate for isosurfacing. Mesh-based approaches allow morphological analysis to be incorporated into the adaptive refinement. Furthermore, the aliasing artifacts introduced by mesh-based techniques from downsampling tend to reduce at higher resolutions. It is often more

efficient and more accurate to retrieve the next level of resolution from a hierarchical mesh than to reconstruct the wavelet coefficients at the current level.

We conclude with a taxonomy of the approaches presented in this paper. In Tables 1 we classify the approaches that distinguish between the temporal and spatial components, first by their primary organization (space or time) and then by their other differences. Although all such spatial indexing approaches are based on the BONO [13], the T-BON [19] does not utilize any temporal coherence, while approaches based on the TSP [29] exploit this coherence using a binary tree on the temporal dimension. The remaining approaches either use the video metaphor and represent a linear sequence of 3D samples, or a binary tree approach, such as those based on the THIT [21]. The approaches that treat the temporal and spatial components equally are presented in Table 2. Here, we distinguish between the cell type (hypercubes and pentatopes) as well as whether the technique exploits spatial coherence or uses a multi-resolution approach.

	Primary	Secondary	Approach
Spatial	Octree	None	T-Bon [19]
		Binary Tree	TSP [29] WTSP [39]
Temporal	Linear Sequence	Keyframes	Waters et al. [24]
		Diamonds	Gregorski et al. [30]
	Binary Time Tree	Span Space	THIT [21] Chiang [22] Vrolijk et al. [23]

Table 1. Taxonomy of approaches that differentiate in their treatment of spatial and temporal dimensions. The spatial approaches are all based on BONO [13] while the temporal approaches are either organized as a series of 3D volumes or a tree of coherent cells in span space.

	Cell Type	
	Hypercube	Simplicial
None	Roberts and Hill [7] Bhaniramka et al. [8]	Weigle and Banks [9]
Spatial	POT[20]	–
Multi-resolution	Westermann [34]	Lee et al. [31] Linsen et al. [35] Ponchio and Hormann [32]

Table 2. Taxonomy of approaches that treat spatial and temporal dimensions equally in terms of the coherence exploited (None, Spatial and Multi-resolution) as well as the cell type (hypercube and simplicial).

References

1. De Floriani, L., Magillo, P.: Multiresolution mesh representation: models and data structures. In Floater, M., Iske, A., Quak, E., eds.: Principles of Multi-resolution Geometric Modeling. Lecture Notes in Mathematics, Berlin, Springer Verlag (2002) 364–418
2. De Floriani, L., Danovaro, E.: Generating, representing and querying level-of-detail tetrahedral meshes. In Bonneau, P., Nielson, G.M., Ertl, T., eds.: Scientific Visualization: Extracting Information and Knowledge from Scientific Data Sets. Springer Verlag (2005)
3. Lorensen, W., Cline, H.: Marching cubes: A high resolution 3d surface construction algorithm. Proceedings of the 14th annual conference on Computer graphics and interactive techniques (1987) 163–169
4. Fujishiro, I., Maeda, Y., Sato, H., Takeshima, Y.: Volumetric data exploration using interval volume. Trans. Visualization and Computer Graphics (1996)
5. Treece, G.M., Prager, R.W., Gee, A.H.: Regularised marching tetrahedra: improved iso-surface extraction. Computers and Graphics **23** (1999) 583–598
6. Nielson, G.M., Sung, J.: Interval volume tetrahedralization. In: Proceedings IEEE Visualization '97. (1997) 221–228
7. Roberts, J.C., Hill, S.: Piecewise-linear hypersurfaces using the marching cube algorithm. In Erbacher, R., Pang, A., eds.: Visual Data Exploration and Analysis VI, Proceedings of SPIE Visualization 2000. SPIE (1999) 170–181
8. Bhaniramka, P., Wenger, R., Crawfis, R.: Isosurface construction in any dimension using convex hulls. Trans. Visualization and Computer Graphics **10** (2004) 130–141
9. Weigle, C., Banks, D.: Extracting iso-valued features in 4-dimensional scalar fields. In: Proceedings IEEE Visualization 1998, IEEE Computer Society (1998) 103–110
10. Bhaniramka, P., Zhang, C., Xue, D., Crawfis, R., Wenger, R.: Volume interval segmentation and rendering. In: Proc. Volume Visualization Symposium. (2004)
11. Zhang, C., Xue, D., Crawfis, R., Wenger, R.: Time-varying interval volumes. Workshop on Volume Graphics (2005) 99–232
12. Ji, G., Shen, H.W., Wenger, R.: Volume tracking using higher dimensional isosurfacing. In Turk, G., van Wijk, J., Moorhead, R., eds.: Proceedings IEEE Visualization 2003, IEEE Computer Society (2003) 209–216
13. Wilhelms, J., Gelder, A.V.: Octrees for faster isosurface generation. ACM Trans. Graph. **11** (1992) 201–227
14. Livnat, Y., Shen, H., Johnson, C.: A Near Optimal Isosurface Extraction Algorithm Using the Span Space. Trans. Visualization and Computer Graphics **2** (1996) 73–84
15. Shen, H., Hansen, C., Livnat, Y., Johnson, C.: Isosurfacing in span space with utmost efficiency (ISSUE). Proceedings IEEE Visualization (1996)
16. Cignoni, P., Marino, P., Montani, C., Puppo, E., Scopigno, R.: Speeding up isosurface extraction using interval trees. IEEE Transactions on Visualization and Computer Graphics **3** (1997) 158–170
17. Edelsbrunner, H.: Dynamic Data Structures for Orthogonal Intersection Queries. Technical report, Institut für Informationsverarbeitung, Tech. Univ. Graz (1980)
18. Van Kreveld, M.: Efficient methods for isoline extraction from a tin. Geographical Information Systems **10** (1996) 523–540
19. Sutton, P., Hansen, C.: Isosurface extraction in time-varying fields using a temporal branch-on-need tree (T-BON). Proceedings IEEE Visualization (1999) 147–153
20. Shi, Q., JaJa, J.: Isosurface extraction and spatial filtering using persistent octree. IEEE Trans. Visualization and Computer Graphics **12** (2006) 1283–1290

21. Shen, H.: Isosurface extraction in time-varying fields using a temporal hierarchical index tree. IEEE Visualization'98 (1998) 159–166
22. Chiang, Y.: Out-of-core isosurface extraction of time-varying fields over irregular grids. Proceedings IEEE Visualization (2003) 217–224
23. Vrolijk, B., Post, F.: Interactive out-of-core isosurface visualization in time-varying data sets. Computers & Graphics **30** (2006) 265–276
24. Waters, K.W., Co, C.S., Joy, K.I.: Using difference intervals for time-varying isosurface visualization. IEEE Trans. Visualization and Computer Graphics **12** (2006) 1275–1282
25. Duchaineau, M., Wolinsky, M., Sigeti, D.E., Miller, M.C., Aldrich, C., Mineev-Weinstein, M.B.: ROAMing terrain: real-time optimally adapting meshes. In Yagel, R., Hagen, H., eds.: Proceedings IEEE Visualization'97, Phoenix, AZ, IEEE Computer Society (1997) 81–88
26. Lindstrom, P., Pascucci, V.: Terrain simplification simplified: a general framework for view-dependent out-of-core visualization. IEEE Transactions on Visualization and Computer Graphics **8** (2002) 239–254
27. Gregorski, B., Duchaineau, M., Lindstrom, P., Pascucci, V., Joy, K.: Interactive view-dependent rendering of large isosurfaces. In: Proc. IEEE Visualization, San Diego, CA, IEEE Computer Society (2002)
28. Lee, M., De Floriani, L., Samet, H.: Constant-time neighbor finding in hierarchical tetrahedral meshes. In: Proceedings International Conference on Shape Modeling, Genova, Italy, IEEE Computer Society (2001) 286–295
29. Shen, H., Chiang, L., Ma, K.: A fast volume rendering algorithm for time-varying fields using a TSP tree. Proc. IEEE Visualization (1999) 371–377
30. Gregorski, B., Senecal, J., Duchaineau, M., Joy, K.: Adaptive extraction of time-varying isosurfaces. IEEE Trans. Visualization and Computer Graphics (2004)
31. Lee, M., De Floriani, L., Samet, H.: Constant-time navigation in four-dimensional nested simplicial meshes. In: Proceedings Shape Modeling International 2004, IEEE Computer Society (2004) 221–230
32. Ponchio, F., Hormann, K.: Interactive rendering of dynamic geometry. IEEE Transactions on Visualization and Computer Graphics **14** (2008) 914–925
33. Cignoni, P., Ganovelli, F., Gobbetti, E., Marton, F., Ponchio, F., Scopigno, R.: Batched multi triangulation. Proceeding IEEE Visualization (2005) 27
34. Westermann, R.: A multiresolution framework for volume rendering. In: Symposium on Volume visualization, New York, NY, USA, ACM Press (1994) 51–58
35. Linsen, L., Pascucci, V., Duchaineau, M., Hamann, B., Joy, K.: Wavelet-based multiresolution with n-th-root-of-2 Subdivision. Conference: Presented at: Dagstuhl Seminar on Geometric Modeling, Dagstuhl (DE), 05/01/2002–05/06/2002 (2004)
36. Eck, M., DeRose, T., Duchamp, T., Hoppe, H., Lounsbery, M., Stuetzle, W.: Multiresolution analysis of arbitrary meshes. In: SIGGRAPH, New York, NY, USA, ACM Press (1995) 173–182
37. Pascucci, V.: Slow-Growing Subdivisions (SGSs) in any dimension: towards removing the curse of dimensionality. Computer Graphics Forum **21** (2002)
38. Sweldens, W.: The lifting scheme: A custom-design construction of biorthogonal wavelets. Appl. Comput. Harmon. Anal **3** (1996) 186–200
39. Wang, C., Gao, J., Li, L., Shen, H.: A Multiresolution Volume Rendering Framework for Large-Scale Time-Varying Data Visualization. Volume Graphics, 2005. Fourth International Workshop on (2005) 11–223
40. Bertram, M., Laney, D., Duchaineau, M., Hansen, C., Hamann, B., Joy, K.: Wavelet representation of contour sets. In: Proceedings IEEE Visualization. (2001) 303–566