Voxel-based Modeling with Multi-resolution Wavelet Transform for Layered Manufacturing

Soonkyu Yim and Hae Chang Gea
Department of Mechanical and Aerospace Engineering
Rutgers, The State University of New Jersey, Piscataway, NJ 08854

Abstract

A voxel-based modeling system with multi-resolution for layered manufacturing is presented in this paper. When dealing with discretized data input, voxel-based modeling shows its clear advantages over the conventional geometric modeling methods. To increase the efficiency of voxel data due to its large storage space requirement, multi-resolution method with wavelet transform technique is implemented. Combining with iso-surface generation and lossless polygon reduction, this voxel-based modeling method can easily work with layered manufacturing. To demonstrate these concepts, components with different resolutions are built using Layered Manufacturing and presented in the paper.

Keywords: Layered Manufacturing (LP), Wavelet Transform, Polygon Reduction (PR)

1. Introduction

During the past decade, considerable efforts have been devoted to recreate a meaningful and visible model from three-dimensional scanners and sectional images. In contrast to the conventional geometric modeling that converts 3-D digitized data to many geometric entities for visualization, much research is focused on volumetric modeling that represents digitized data directly from three dimensional discrete units of volume element (voxel). In order to increase the ability of shape control over voxel data, Wu et al. [18] introduced Non-Uniform Rational B-Spline (NURBS) volumetric model with a partial integration between surface and volume graphics. Although volumetric modeling has some clear advantages when dealing with a large set of digitized data, it suffers from its large storage requirement [8]. To reduce the storage size of volumetric data acquired from MRI, Muraki [11] used a wavelet transform that displays models in different resolutions. However, mechanical components modeled by volumetric data still need to be converted to geometric entities before they are fabricated by traditional manufacturing machine tools. Recently, this fabrication bottleneck has been overcome by the advancement of Layered Manufacturing (LM). Because LM fabricates the object layer by layer from the bottom to the top, it is not restricted by the topology of the object or by the orientation of the initial setup. With the development of LM, volumetric data has been suggested as an alternative modeling tool for mechanical components [1][2].

In this paper, a voxel-based modeling with multi-resolution wavelet transform for layered manufacturing is presented. Instead of the Batelle-Lamarie wavelet and the Difference of Gaussian (DOG) wavelet which were implemented in [11][12], the Haar wavelet is chosen in this work. The benefit of the Haar wavelet is that it can be used with integer variables during transform. Therefore, storage size and computation time can be effectively reduced comparing to other wavelets, which use float or double variables. A bit remainder index is applied in this system to accommodate arbitrary data size during the wavelet transform [19]. Primitive voxel editing functions are included for model creation and shape modification. Marching Cube

algorithm and lossless polygon reduction are implemented before building the STereoLithograph (.STL) files. This voxel modeling system is coded in Java and Java3D Application Programming Interface (API) for its portability. To demonstrate these concepts, components with different resolutions built from voxel model creation to LM are presented at the end of this paper.

2. Voxel-based Modeling

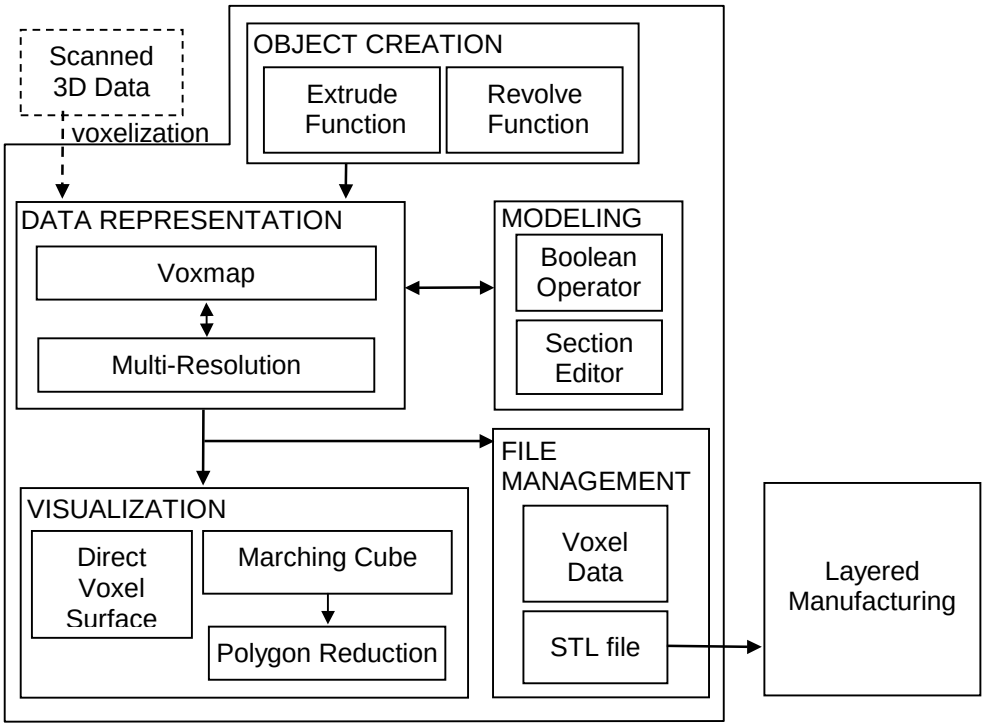


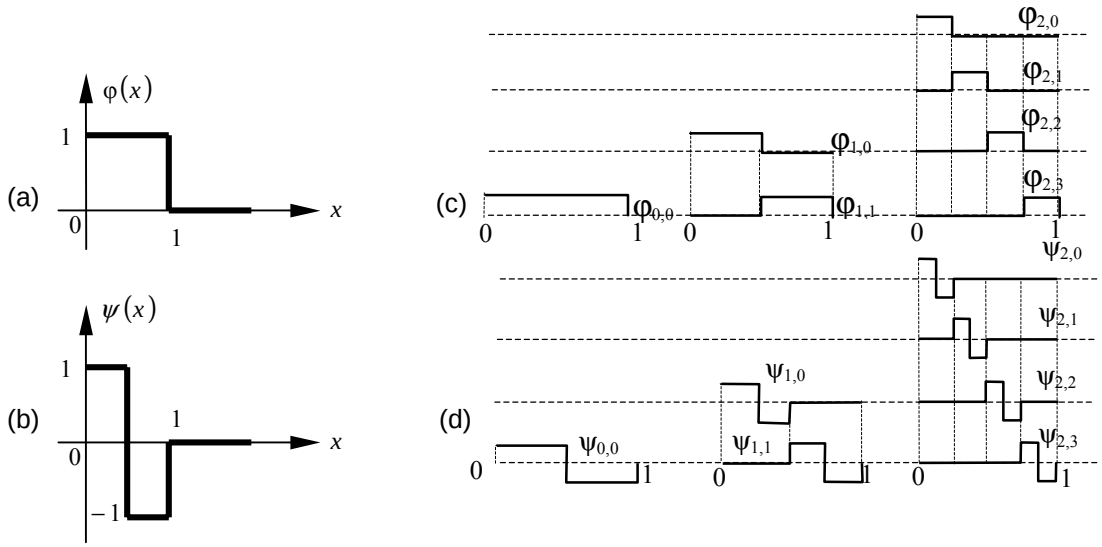
Figure 1: Voxel-based Modeling System.

A voxel-based modeling starts from constructing a voxmap of the object. The voxmap is a three-dimensional data array that is generated through voxelization of an object. To voxelize an object, we first divide the volumetric object into small cubes (voxel) with equal sizes. Then an integer value is assigned to each voxel to describe its property and stored in the corresponding volume buffer (voxmap). Because voxel is a basic atomic unit of a volumetric object, voxel-based model is convenient when representing objects from three-dimensional scanned data, when modeling activity is based on digitized operations, and when the output consists of only discrete values [2][3][7][16][17]. In this paper, a voxel-based modeling system is presented as shown in Fig. 1. First, the object is created from pixel extrusion and revolving operations. It is stored in a voxmap and can be modified by a built-in modeler consisting of various Boolean operations and section editing functions. Direct conversions from 3D scanned image data to voxmap is not implemented here because most scanner vendors support these functions with their hardware. To reduce the storage requirement of the voxmap, a multi-resolution wavelet transform is implemented in the system. Before visualizing the model, inner surfaces are removed and outer surfaces are smoothened by a Marching Cube (MC) algorithm. However, the MC usually generates too many small triangle elements on the surfaces. Therefore, a simple, but loseless Polygon Reduction (PR) is applied to the iso-surfaces before converting to a STL file for LM. Detailed discussion of the system is given in the next few sections.

3. Multi-resolution Wavelet Transform

Although voxmap provides a very simple data structure for digitized models, it suffers from the limitations on resolution and storage. For digitized models, resolution is expressed in terms of the number of pixels (voxels) used in the x, y and z directions. Image with high resolution needs more pixels than that of low resolution. For instance, doubling the image resolution requires eight times more storage space in three dimension. On the other hand, resolution requirements are not the same for all applications. In practice, some applications need higher resolution and others trade higher resolution with faster access speed. To satisfy both demands, a multi-resolution wavelet transform is implemented in our system. The basic idea of multi-resolution wavelet transform is to express functions with a collection of coefficients, each of which has some limited information in both data size and resolution [14][15]. Using multi-resolution wavelet transform, users can easily retrieve both coarse and detail information of the model without heavy computation penalty. The Haar wavelet is chosen in this work. The benefit of the Haar wavelet is that it can be used with integer variables during transform. Therefore, storage size and computation time can be effectively reduced comparing to other wavelets using float or double variables. Next, the Haar wavelet and bit-remainder algorithm used in this system will be discussed.

3.1 Haar Wavelet



(a) scale function (b)wavelet function (c) scale functions for different resolution when n=2 (d) wavelet functions for different resolution when n=3.

Figure 2: Haar wavelet functions

Wavelet transform has two basic functions, scale(φ) and wavelet (ψ) functions as shown in Fig. 2. Before wavelet transform, one dimensional data with 2^n samples are mapped into initial scale functions ($\varphi_{n,j}$), where j varies from 0 to 2^n-1. From wavelet transform, samples are divided into two different regions, coarse and detail data by averaging and subtracting two neighboring samples. This process can be done recursively until there remains one element in the coarse data. Therefore, wavelet transform can be done for maximum n times if the data set

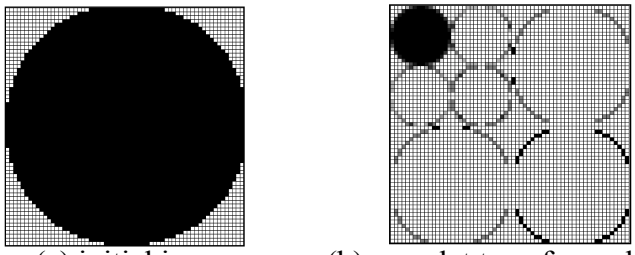
has 2^n samples. During transformation, wavelet and scale functions follow these relations in Eq. (1) through Eq. (3) and their shapes are represented as shown in Fig. 2(c) and (d).

$$\varphi_{i,j} = \varphi(2^i x - j) \tag{1}$$

$$\psi_{i,j} = \psi(2^i x - j) \tag{2}$$

$$\psi_{i,j} = \varphi_{i+1,2j+1} - \varphi_{i+1,2j} \tag{3}$$

where $\varphi(x) = 1$ for $(0 < x \leq 1)$ and 0 otherwise and $\psi(x) = 1$ for $(0 < x \leq 0.5)$, -1 for $(0.5 < x \leq 1.0)$, and 0 otherwise; i=0,1,...,n-1 and j=0,1,…,2^i-1. If a data set consists of two-dimensional array such as a 2-D image, wavelet transform is applied to both row and column directions. Then, the size of the coarse data ($\varphi(y)\ \varphi(x)$) becomes one quarter of the previous coarse image after one transform. Fig. 3 shows an example when wavelet transform is applied twice in both row and column directions. The black circle in Fig. 3(a) is reduced to its 1/16 size as shown in the top left corner of Fig. 3(b) from two times wavelet transform and rest of the region in Fig. 3(b) indicates detail data



(a) initial image (b) wavelet transformed domain

Figure 3: wavelet transform for an image

3.2 Bit-remainder

After one wavelet transform, data is divided into two regions: coarse and detail data. Both must have exactly the same number of samples. Therefore, it is necessary to have data with 2^n samples in order to keep even numbers for every transform. If an image has an arbitrary size that does not match 2^n, artificial inflation to the nearest 2^n must be done before transforms. For example, an object with 144×128-pixel size image must be expanded to the size of 256×256 before wavelet transformation. Obviously, this artificial inflation increases the size of an image and wastes processing time. This problem is much more serious when we apply wavelet transform to voxmap since it is a three-dimensional array. To reduce this problem, a bit-remainder index is added to the wavelet transform [19]. With the bit-remainder index, images with arbitrary size can be transformed without artificial inflation. The basic idea of bit-remainder index is to keep the size of data in even for every wavelet transform. If the initial data set needs r times wavelet transform, a binary index array with r elements is created first; each index element corresponds to one wavelet transform. For wavelet transform with even array size, '0' is assigned to the index; for wavelet transform with non-even array size, '1' is assigned to the index and one null element is appended to the initial data set. In this way, data set maintains even for wavelet transform and bit-remainder index keeps track how many null elements are added. This process is reversible, i.e. the initial data set can be recovered from a reverse transform. A simple 2-D example can illustrate this method. Suppose a 2-D data set with 140×128 samples needs to have wavelet transforms for five times. Using this method, the

data size after each wavelet transform are: (70×64), (35×32→36×32), (18×16), (9×8→10×8), and (5×4). The second, fourth and fifth transforms produce non-even data size in *x* direction and null elements are added to the sample in *x* direction. The *x*- and *y*- bit-remainder indices become {0,0,1,0,1} and {0,0,0,0,0}. With the traditional wavelet transform, this data set need to be inflated to 256×256, while this method only adds ten index bits and two null elements (142×128).

4 Model Creation

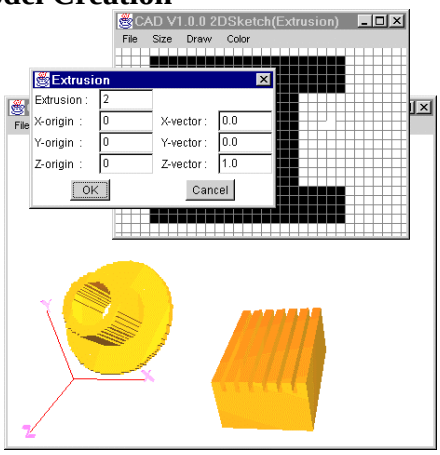


Figure 4: Object Creation

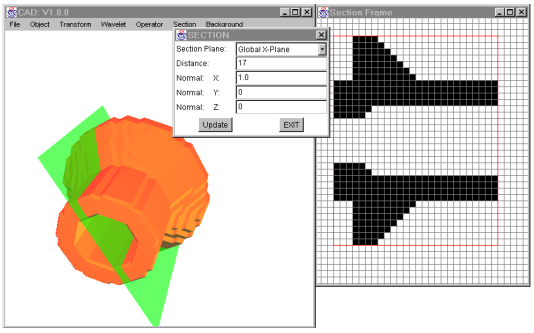


Figure 5: Section view and its local editing

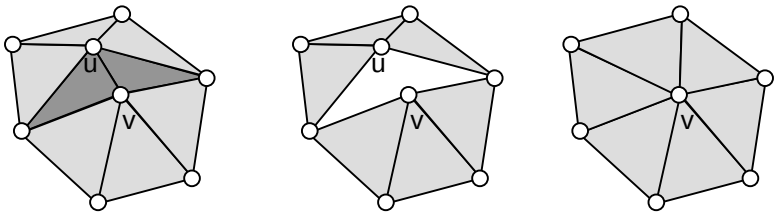
Two basic image operators, extrusion and revolution, are included in this voxel-based modeling system for model creation. A two dimensional canvas is called when designers need to create or modify an object. After designers paint the section shape in the canvas, the Minkowski addition operator is used to extrude the section with a given distance and direction [6][13]. Fig. 4 shows a two-dimensional canvas window, extrusion input dialog and parts created by extrusion and revolution. Model can be modified with the same way: voxels on the intersection plane are extracted to the two-dimensional editing canvas (Fig. 5).

5 Model Visualization

In order to visualize the voxel data, standard MC [9] is used for iso-surface extraction. Because one voxel has eight vertices, the number of different cases of occupied/non-occupied vertices is $2^8 = 256$. With the help of symmetric conditions, they are reduced to fifteen distinct cases. Based on this look-up table, one to four triangles are created in one voxel for the feature's boundary surface. With the MC algorithm, iso-surface generation from voxmap is simple and straightforward. But this algorithm tends to create so many triangles that enormous triangle entities reduce the over-all system performance and increase the time needed for generating StereoLithograph Contour file (SLC) for LM. To reduce the number of polygons from a set of triangles, a simple Polygon Reduction (PR) scheme is implemented in the system. There are many PR algorithms reported in the literature [4][5][10]. Cost of the edges are determined by the normal vectors of the surfaces as defined in Eq. (4) and the less the cost, the little there has geometric changes. Therefore, PR with zero cost is equal to the loseless polygon simplification.

$$\text{cost}(u,v) = \max_{n_u \in T_u} \left(\min_{n_{uv} \in T_{uv}} \left[(1 - n_u \cdot n_{uv}) \right] \right) \qquad (4)$$

where T_u is the set of triangles surrounding vertex u and T_{uv} is the set of two triangles share vertices u and v. n_u and n_{uv} are the normal vectors of the triangles in T_u and T_{uv}. Cost functions, cost(u,v) and cost(v,u), are different from the formulation. Since our goal is to merge unnecessary triangles without altering the shape, we only merge vertices when cost is zero. All the vertices in the voxmap will pass through this polygon reduction test until zero cost exists. Once the PR is completed, STL file can be generated easily because the whole surface of the model is covered with triangles, which matches the native format of STL files. At this point, the part is ready for LM.



(a) detect an edge (b) remove surfaces (c) update vertex

Figure 6: Surface Removal from each edge's cost function

6. Example

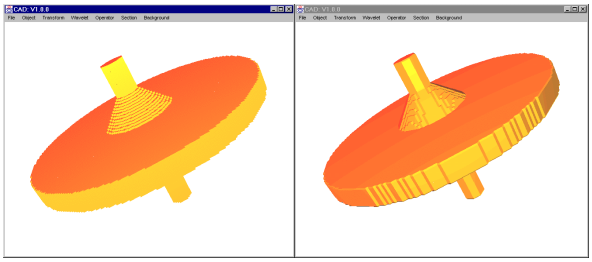


Figure 7: Initial voxel model (117×117×80)

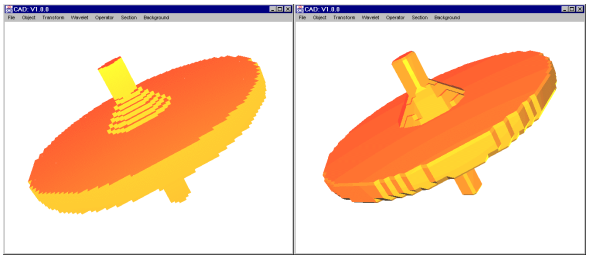


Figure 8: Wavelet transformed voxels(59×59×40)

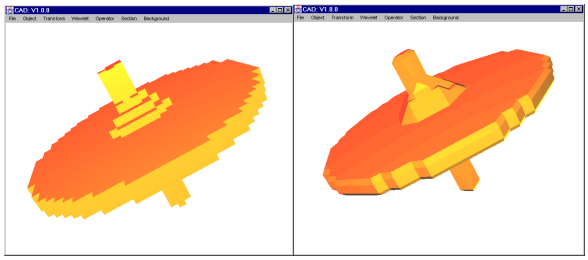


Figure 9: Wavelet transformed voxels (30×30×20)

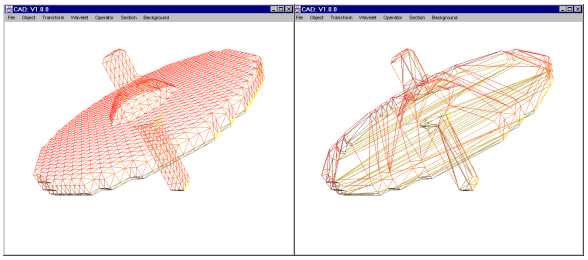


Figure 10: polygon reduction

In this section, simple models created from a voxel-based modeling system and fabricated by SLA-190 are presented. The initial model is created using revolving operator and stored in a 117×117×80 voxmap. Multi-resolution wavelet transform with bit-remainder indices is used twice to reduce the model size to 59×59×40 and 30×30×20. Models with different resolutions are displayed in two ways: voxel surface and iso-surface in Fig. 7 to Fig. 9. In the left hand side, voxel surface models with stair case effect in the center support region are displayed. In the right hand side, iso-surface models with smoothened surfaces are shown. Comparison among different resolutions is listed in Table 1. Due to the multi-resolution characteristics, the number of voxels, vertices, edges and triangles are decreased significantly after wavelet transform. To fabricate this component using LM, we applied polygon reduction to reduce the number of

triangles. As an example, we took the final shape from Fig. 9 and applied the polygon reduction scheme. The results are shown in Fig. 10. In the last row of Table 1, numbers of triangles after polygon reduction for different resolutions are listed for comparison. From triangles and normal vectors generated from PR, STL files are created and parts are built in the SLA-190 machine as shown in Figure 11. The sample ratio among these three objects with different resolution is 1:1, 1:8 and 1:64 due to multi-resolution wavelet transforms. For comparing the difference in resolution, parts are scaled up so that all features have the same dimension as shown in Fig. 11(b).

Table 1. Models with different resolutions

voxmap size	117×117×80	59×59×40	30×30×20
total voxels	1,095,120	139,240	18,000
vertices	29,530	7,600	1,830
edges	88,584	22,794	5,484
triangles	59,056	15,196	3,656
triangles with PR	1,940	1,100	446

(a) (b)

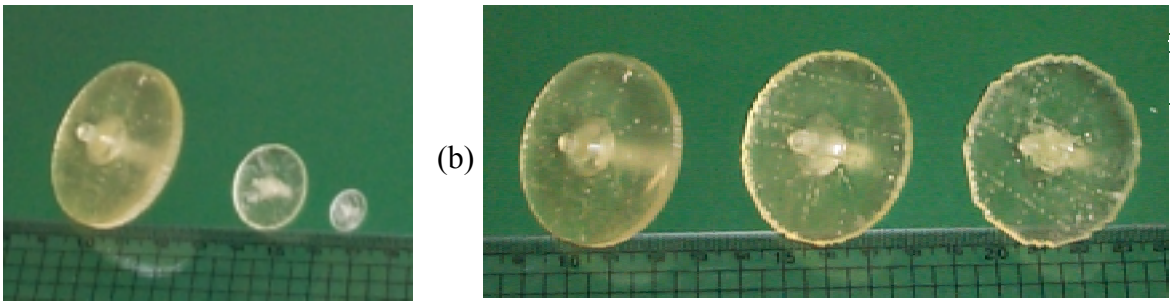


Figure 11: Parts from SLA-190. (a) parts built with same voxel size (b) parts scaled-up to the initial size.

7. Conclusion

Voxel-based modeling system that supports multi-resolution was presented in this paper. This system is programmed with Java and Java3D API. It is portable to any system as long as a Java compiler is presented. To control the size of the data set, wavelet transform and bit-reminder index are applied. We found that multi-resolution function is very useful when the balance between model quality and process efficiency is needed. Simple Boolean operators and section editing functions are implemented in the system but much more need to be done. In addition, model after PR can be easily converted to STL files for LM. To further accuracy improvement to the wavelet transformed model, combination of voxels with different resolution is under development.

8. References

[1] Chandru, V., Manohar, S., and Prakash, G., Voxel-based modeling for layered manufacturing, IEEE Computer Graphics and Applications, pp. 42-47, November, 1995.

[2] Chandru, V., Manivannan, M., and Manohar, S., Minkowski operator and features of voxel models, DETC-99/DAC-8629, September 12-15, Las Vegas, 1999.

[3] Chen, Y., Zhu, Q., Kaufman, A., Physically-based animation of volumetric objects, IEEE Computer Animation '98, pp. 154-160, 1998.

[4] Cohen, J., Olano, M., Manocha, D., Appearance-Preserving Simplification, SIGGRAPH 98, 115-122, Orlando, Florida, July 19-24, 1998.
[5] Hoppe H., Progressive Meshes, SIGGRAPH 96, 99-108, Louisiana, August 4-9, 1996.
[6] Ghosh, P., A unified computational framework for Minkowski Operators, Computer & Graphics, 17(4):357-378, 1993.
[7] Gibson., S., Fyock, C., Grimson, E., Kanade, T., Kinis, R., Lauer, H., McKenzie, N., Mor, A., Nakajima, S., Ohkami, H., Osborne, R., Samosky, J., Sawada, A., Simulating Surgery using Volumetric Object Representations, Real-Time Rendering and Haptic Feedback, MERL technical Report, TR96-16, 1996.
[8] Kaufman, A., Cohen, D., and Yagel, R., Volume Graphics, IEEE Computer, pp. 51-64, July 1993.
[9] Lorensen, W., Harvey, E., Marching Cubes: A High Resolution 3D Surface Construction Algorithm, ACM Computer Graphics, 21(24):163-169, July 1987.
[10] Melax, S., A Simple, Fast, and Effective Polygon Reduction Algorithm, Game Developer, pp. 44-49, November, 1998.
[11] Muraki, S., Volume Data and Wavelet Transforms, IEEE Computer Graphics & Applications, pp. 50-56, 1993.
[12] Muraki, S., Multiscale Volume Representation by a DOG Wavelet, IEEE Transactions on Visualization and Computer Graphics, 1(2):109-116, June, 1995.
[13] Sethia, S., Manohar, S., Minkowski Operator for Voxel-based Sculpting, Computer and Graphics, 22(5):593-600, 1998.
[14] Stollinitz, E., DeRose, T., Salesin, D., Wavelets for Computer Graphics: A Primer Part I, IEEE Computer Graphics and Applications, 15(3):76-84, May, 1995.
[15] Stollinitz, E., Derose, T., Salesin, D., Wavelets for Computer Graphics: Theory and Applications, Morgan Kaufmann Publishers, Inc., 1996, ISBN;1-55860-375-1.
[16] Wan, M., Qu, H., Kaufman, A., Virtual flythrough over a voxel-based terrain, IEEE Virtual Reality Conference, pp. 53-60, March, 1999.
[17] Wright, J., Hsieh, J., A Voxel-based forward projection algorithm for rendering surface and volumetric data, IEEE Visualization ’92, pp. 340-348, Los Alamitos, CA, October, 1992.
[18] Wu, Z., Seah, H., and Lin, F., NURBS Volume for Modelling Complex Objects, chapter 9, pp. 159-167, Chen, M., Kaufman, A., and Yagel, R. (Eds), Volume Graphics, Springer 2000, ISBN:1-85233-192-5.
[19] Yim, S., and Gea, H., Development of an Image-based CAD using Wavelet Transform, Proceeding of ASME, DETC00/ DAC-14262, September 10-13, 2000.