Adaptive Proxy Geometry for Direct Volume Manipulation

Megumi Nakao[1)], Kei Wai Cecilia Hung [1)], Satoshi Yano[1)], Koji Yoshimura[2)] and Kotaro Minato[1)]

[1)] Graduate School of Information Science, Nara Institute of Science and Technology
[2)] Department of Urology, Kyoto University Hospital

ABSTRACT

This paper introduces a new design to allow interactive, direct manipulation of volume data on volumetrically rendered images. We present an adaptive *volume proxy mesh* which serves not to define surfaces, but to encode the geometry and physical state of the volume. This system performs a modeling-free form of direct volume deformation by adaptively constructing the whole geometric structure on the background while keeping the structure transparent to the user. A complex, high-frequency structure is locally encoded into the mesh on-the-fly based on the user's volume of interest. This paper also presents a scheme for volume shading/shadowing applied in combination with our framework for improving reality in the visualization of time-varying deformable objects. We demonstrate examples of geometry-encoded direct volume manipulation and application to volume exploration and surgical simulation.

KEYWORDS: Direct volume manipulation, Adaptive geometry modeling, GPU-computing and Medical application

INDEX TERMS: I.3.6 [Computer Graphics]: Computational Geometry and Object Modeling—Curve, surface, solid, and object representations, I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Color, shading, shadowing, and texture, I.3.8 [Computer Graphics]: Applications

1 INTRODUCTION

CT, MRI, optical microscopy, and other recent imaging modalities yield very large amounts of volume data (volumetric scalar data). Now that these modalities are widely adopted, volume graphics techniques [17] play important roles in many fields of medicine, biology, and engineering. GPU (Graphical Processing Unit) technology enables the rendering of 512^3 or larger voxels at interactive rates on general-purpose computers [9][15]. Practitioners in many fields have been awaiting the development of new techniques for interactive, direct volume manipulation. In medicine, for example, clinicians and surgeons hope to use such techniques to explore the internal structures of organs [7][23] or simulate surgical processes in rehearsal for actual interventions [26][32]. A number of technical challenges remain to be surmounted before their hopes can be answered.

The surface modeling approaches in the design of volume manipulation environments require explicit segmentation to define complex 3D structures from volume data. This demands time-consuming processing, and it often requires strict physical descriptions between large numbers of complicated triangle meshes. Surface models also run into limitations in the handling of internal structures. In voxel-based volume-editing approaches, the voxels are directly updated to represent the coloring, sculpting, and cutting-out of specific volume areas [1][6][8]. Yet to provide volume-deformation results, the voxels affected by the deformation need to be transformed [32]. This requires abundant computation time and is susceptible to aliasing effects in the deformed volume [23].

Several recent studies have sought to represent volume deformation using point sampling and 3D texture interpolation on GPUs [13][22][28][31][33]. Though few models allow physically strict descriptions, this approach has the potential to enable interactive manipulation while providing visually valid deformation. In the direct manipulation of the volume, control-point-based approaches [13][31] may actually result in indirect manipulation. This is inevitable, as the user manipulates control points rather than volumetric objects. Moreover, a number of technical hurdles continue to hinder the realization of various types of geometry-based manipulation, the representation of elastic behavior, and improved rendering results for time-varying deformable objects.

The long-term goal of this study is to establish intuitive environments where end users can manipulate volumetric objects freely and directly. We approach this goal, in this paper, by introducing a new design to represent interactive, direct manipulation of volume data on volumetrically rendered images. In doing so, we focus on the importance of avoiding three things: non-intuitive manipulation, unfriendly modeling tasks, and time-consuming segmentation processes by the user.

As volume data are originally 3D scalar fields, they have no 3D geometry, surfaces, or physical information as explicit formats. In consideration of this property, we begin our paper by exploring the methods for modeling flexible 3D manipulations and the deformation of volume data. Specifically, we introduce an adaptive *volume proxy mesh* designed to encode structural, physical information as a volumetric geometry format. This structure is laid over the whole volumetric space and used in most parts of our volume-deformation process. By incorporating this framework, we also describe the design of an interactive, direct manipulation environment and a volume-rendering scheme to represent deformation. Towards the end of the paper, we validate the performance of our techniques using case examples.

This work contributes significantly in the following ways.

- We introduce the concept of the *volume proxy mesh* for geometry-based volume manipulation. This mesh is prepared in the background and used not to define surfaces, but to encode the geometry and physical state of all of the volume data. It originally handles global deformation of volumetric space at low-frequency points.
- We present an adaptive design on the proxy geometry for direct volume manipulation. The technique can be performed without relying on the user's manual process for modeling from volume data. The mesh is updated in the background and kept out of view. The high-frequency structure of volume data is locally encoded into the mesh on-the-fly based on the user's volume of interest.
- We present a volume-rendering/shading scheme for visualizing time-varying deformable objects and design a GPU-based computation framework for achieving an

1) 8916-5, Takayama, Ikoma, Nara, 630-0192, JAPAN E-mail : meg@is.naist.jp
2) 54 Kawahara, Shougoin, Kyoto, 606-8357, JAPAN

interactive refresh rate. The rotational components of deformation are volumetrically extracted for each vertex and computed on GPUs.

- We demonstrate several examples of direct volume manipulation during volume exploration and surgical simulation. The proposed framework is applied to several types of medical data as well as public domain data.

2 Related Works

The representation of volume deformation has been studied for at least a decade. The earlier studies focus on the construction of deformed volumes from initial states based on image morphing [16][38] or rigid transformation [20]. Manipulation techniques have been explored by designing more complex strategies for volume transformation. In fields of application such as volume illustration, various types of deformation have been formulated in more detail to explore the complex internal structures of volume data [7][23][31]. One way to represent volume deformation is to pre-computationally prepare for a transformation of the volume. This is categorized as a form of non-interactive manipulation. The approach is still popular in related fields of research, such as transfer functions [10] and displacement maps [11][12].

Other studies, meanwhile, have been exploring interactive manipulation techniques which take advantage of advances in GPU technology and programmable shaders. Resk-Salama et al. began by proposing a basic framework for realizing interactive volume deformation [28][29]. In their framework, a 3D deformation is defined based on the transformation of proxy points placed in a segmented volume. The points are placed in a static and regular fashion before the manipulation begins. Masutani et al. have extended this approach in a finite element model (FEM) for simulating the elastic behavior of organs [22]. Georgii et al. presented an efficient scalable pipeline for GPU tetrahedral grid rendering [33]. Yet volume data obtained from actual imaging modalities contain complex topologies (see Figure 1a). Even if a specific anatomical area is segmented, the omission of surrounding tissues in the rendering precludes a realistic representation, as shown in Figure 1b. This can be solved by accurately simulating interactions with multiple high-quality surface models [34]. Yet to do so requires a data-specific mesh-modeling process of a type impractical for most end users and practitioners.

Correa et al. recently presented an efficient design for volume deformation using scattered control points, a method which enables volume deformation without applying the complex process of mesh modeling [13]. In their framework, the user manually places control points on cross sections of the volume and manipulates them to goal positions. The deformation is rendered via inverse displacement maps [11]. This approach succeeds in controlling some types of deformation, but the method of manipulation is still indirect and independent from the structure of the volume. And as an added limitation, the degree-of-freedom of the deformation is restricted by a small number of manually placed 2D control points.

In spite of the earlier efforts heretofore described, several issues remain to be solved before direct volume manipulation becomes a reality. The foremost technical challenge is the on-the-fly, direct handling of implicit geometry stored in the volume data. The friendliest environment will allow the user to freely and directly manipulate the volume of interest while observing the rendered image. Although Chen et al. have focused on the context preserving deformation of 2D anatomical illustration [39], no methods for direct volume manipulation have yet been formulated manipulation techniques. Specifically, we extend the point-sampling approaches [13][28] to a more generalized format, that from this viewpoint. This paper introduces a set of direct volume

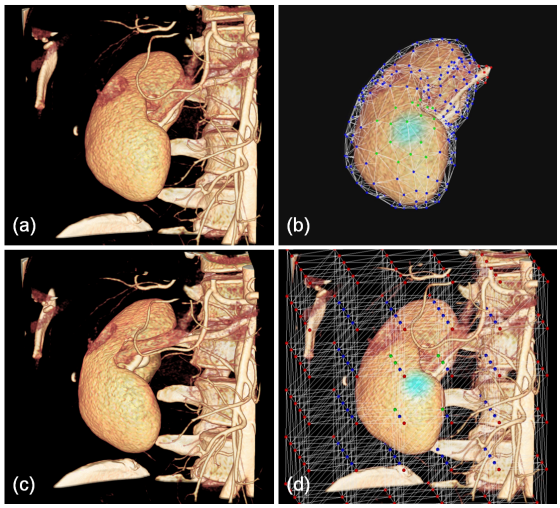

Figure 1 (a) A portion of an abdominal CT volume with complex anatomical structures. (b) Conventional meshing approach to model the shape of a single organ for deformation. (c) Global, uniform image-warping can generate visual artifacts on the local fine structure of the volume. (d) Our volume proxy mesh encodes the local geometry and physical behavior of the volume for interactive direct volume manipulation.

of the *adaptive proxy volume mesh*, and present a new framework for handling geometry and rendering for direct volume manipulation.

3 Direct Volume Manipulation Framework

3.1 Volume Proxy Mesh

First, we assume volume data $\mathbf{I}$ is a 3D scalar field composed of discrete voxels with intensity $I_{xyz} = f(x, y, z)$ in a three-dimensional space. As we mentioned earlier, per-voxel computation has too high a computational cost to feasibly apply. On the other hand, the sampling of volumetric space with discrete points affects both the accuracy and degree-of-freedom of local deformation. Figure 1c shows an example of image warping with distortion of the kidney and unnatural deformation of the bones. One of our research targets is to improve this situation by modeling implicit geometry with minimal user effort. In many situations, low-frequency sampling and interpolation with 3D texture work well for visually representing valid deformation while maintaining interactive performance [13][28][30]. Here we take advantage of this capability and improve the point sampling by handling generalized aspects of the integration of the geometry into the volume data.

To sample volume intensity $\mathbf{I}$, we use regularly or irregularly placed vertices in the volumetric space. Point sampling representation has been well explored in the fields of point-based graphics and fluid simulation [25]. In this paper we introduce the *adaptive volume proxy mesh* (see Figure 1d) into the volume manipulation field. This mesh encodes volumetric geometry and physical information (i.e., elasticity and boundary conditions) between sampled vertices. Rather than being visualized directly, the mesh is placed into the entire volumetric area as a background data structure. This enables a direct, interactive manipulation of the volume with elastic deformation.

The volume proxy mesh is formed with a standard tetrahedral mesh format $(\mathbf{V},\mathbf{E})$, where $\mathbf{V}$ is sampled vertices and $\mathbf{E}$ is a set of tetrahedral elements. Three-dimensional manipulation and deformation of the target volume are supported by vertices placed in the volumetric space. The small image in Figure 2a shows an

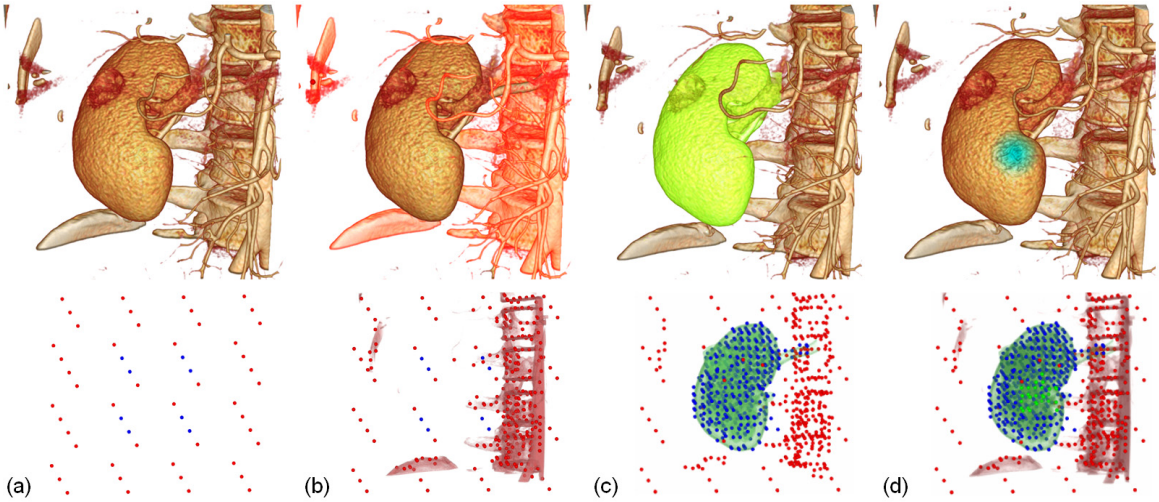

Figure 2 Configuration of the direct volume manipulation and the geometry-encoding process. (a) Rendered volume and vertices placed in a regular configuration for low-frequency sampling of global structures. (b) Physically fixed volume area and newly created vertices with a fixed condition, (c) A manipulation target and the corresponding geometry encoded to the mesh. (d) Interaction area directly manipulated by the user. The positions of the vertices in this area are used as constraints in the deformation models.

example of an initial setup of vertices for a volume proxy mesh. A small number of vertices are placed in a regular configuration within the whole volumetric area. Though earlier approaches use the pre-computed segmented volume to construct the mesh structure [22][28], we do not segment the volume in this stage. The mesh is kept in the background, unperceived by the user. Through user interactions with volumetrically rendered objects (see the upper part of Figure 2), the mesh is adaptively improved and self-organized by the extraction of the local geometry from the volume data. Figure 2d shows a locally refined mesh with boundary conditions for improving manipulation. The vertices densely placed around the user's volume of interest help achieve fine deformation even on high-frequency structures and edges.

After constructing the tetrahedral mesh of the encoded geometry, we can rely on a number of well-established mesh deformation algorithms, including physically based models [3][24][25] and feature-preserving approaches [5][21]. Here we use a linear finite element model (FEM) for computing mesh deformation. The deformation results are obtained at an interactive refresh rate (> 100 frame/sec) even on thousands of vertices. Our interest in this paper is the design of the *adaptive proxy geometry* for direct volume manipulation. As such, we refrain from touching upon the details of physics-based computation methods. The details of linear FEM computation, meanwhile, have been introduced in two earlier papers [25][3]. For details on the technique to support large deformation with rotation, refer to the stiffness-warping approach introduced by Muller [24] and the co-rotational finite element modeling recently introduced by Georgii [35].

In our framework, all of the model transformations with deformation are discretely modeled by displacement of vertices of the volume proxy mesh. First, the vertices are categorized into three groups based on the physical conditions: *fixed* vertices (red), *constrained* vertices (green), and *free* vertices (blue) (see Figure 1d). The fixed vertices represent physically fixed structures that will not deform or move through manipulation, such as bones. The constrained vertices represent selected areas that can be directly controlled by the user. The free vertices express other soft areas that will deform through manipulation.

When the user manipulates the volume, the initial vertices $\mathbf{V}$ are displaced to $\mathbf{V}'$ through mesh deformation. In the computation of deformation, the positions of the fixed vertices and constrained vertices are imposed as physical constraints (or boundary conditions) that must be satisfied by the deformation model. Concurrently, free vertices are updated at every simulation step based on the computation results. The constrained vertices and free vertices represent model deformations. Next, our framework renders deformed volume using the new coordinates of $\mathbf{V}'$ incorporating the initial volume intensity $\mathbf{I}$. If no manipulation is given, $\mathbf{V}'$ is equivalent to the initial vertices $\mathbf{V}$. Note, again, that none of the vertices of the mesh are displayed to the user. This is the key point for enabling direct volume manipulation in our framework. The next section goes into detail on the design.

3.2 Adaptive Geometry Encoding

This section describes how we refine the volume proxy mesh on the background during the user's interaction with the rendered volume. In the pre-computation stage, regularly sampled vertices are placed into the whole volumetric space. If necessary, boundary conditions are initially assigned to the specific vertices. In Figure 2a, the vertices placed at the boundary of the volume are fixed due to the physical constraints of the internal structures of the abdomen. This setup alone may serve to represent simple, global deformation, but the uniformly placed grid fails to express the local geometry and accurate physical state of the volumes [33] of organs such as the kidney, blood vessels, and bones. To overcome this limitation, our framework extracts local information on the physical condition through the user's interaction with the volume. This information is stored as volume labels and used to refine the volume proxy mesh.

The user begins by setting a fixed area (colored red in Figure 2b) on the rendered image with help from an interactive volume-selection interface. Here we prepare a drawing-based interface for capturing the volume of interest through 2D manipulation on the rendered image. When the user indicates a pixel on the rendered image, our framework estimates the corresponding voxel in the volumetric space by accumulating alpha values at sample points in the eye direction. This process is similar to the ray-casting protocol [19]. By integrating this pointing interface with the region-growing process, the user can extract the volume of interest interactively. Two-dimensional drawing on the

volumetrically rendered image is also allowed as a means of restricting the labeling area. Details on related volume-editing and segmentation techniques can be found in the papers by Bruckner et al. [6] and Burger et al. [8]. The selected volume area is stored as a volumetric fixed label $\mathbf{L}_{fixed}$. The magnitude of its partial derivatives, that is $|\nabla \mathbf{L}_{fixed}|$, represents the boundary of the fixed area defined by the user. We can place additional vertices in a fixed condition by adequately sampling $|\nabla \mathbf{L}_{fixed}|$. The reference image in Figure 2b shows $|\nabla \mathbf{L}_{fixed}|$ and newly created vertices. The fixed vertices refine the volume proxy mesh, which helps to localize the deformation of the volume.

Next, a series of manipulation targets, the user's volumes of interest, are locally indicated based on a process similar to that used to set the fixed area. The deformable area consists of not only the manipulation targets, but also the surrounding structures and free spaces. In Figure 2c, the kidney is one manipulation target. The visually open spaces, blood vessels, and soft tissues neighboring or connecting to the kidney compose the deformable area. The regularly placed vertices are effective for roughly interpolating the deformation of this surrounding deformable area. In this step, user-defined elastic parameters such as Young's modulus and Poisson's ratio are stored in a volumetric deformable label $|\nabla \mathbf{L}_{free}|$. As shown in Figure 5, an heterogeneous elastic state is modeled by repeating this selection process. Four different elastic parameters are set to the corresponding sphere-shaped volume area. We note that a small Young's modulus is used to visually open spaces. Thus, we model the whole volumetric space as continuous elastic media by virtually inserting extremely soft objects onto the free spaces. This approximation visually allows discontinuous deformation, such as deformation by an incision. The details are explained in the following section.

3.3 Direct Manipulation via 2D Interface

After creating fixed vertices and free vertices on the boundary of the user-defined labels, the user inputs an interaction area on the rendered image (see Figure 2d). In the case of a simple point-based pushing/pulling manipulation, for example, the nearest vertex of the volume proxy mesh is selected and marked as the constrained vertex. To support a large-area interaction, a bounding sphere or 2D drawing is used to impose the constraints to multiple vertices. The user can manipulate volume in 3D via 2D displacement of the constrained vertices. In our interface, the 2D displacement is obtained by a drag of the mouse and then transferred to 3D displacement on the projected screen. This displacement information is used for physical constraints, the forced displacement of the constrained vertices of the deformable model. The displacement of all free vertices of the volume proxy mesh is computed by solving the FEM model [25]. To represent discontinuous deformation shown in Figure 3, we create two constraint surfaces in a face-to-face orientation by inserting a set of constrained vertices on the interaction area. The distance between the constraint surfaces is specified by the user and imposed as a boundary condition to compute the deformation of the incision. This process is based on the tetrahedral subdivision scheme presented in the paper by Bielser et al. [36]. As the volume data originally take the form of a continuous image, earlier studies have attempted to represent discontinuous deformation by methods such as inverse displacement mapping [11], control points [13], or incision templates [37]. Our scheme shows the potential for robust modeling of an incision by adaptively updating the background proxy geometry through the cutting operation of the user on the rendered image.

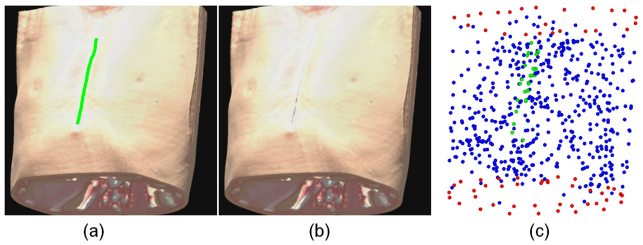

Figure 3 Direct sketching of incision on rendered breast data. The adaptive proxy geometry supports the discontinuous geometry and deformation of the incision.

4 Rendering and Lighting of the Deformed Volume

4.1 Rendering of the Deformed Volume

Figure 4 summarizes the rendering process of our framework. The gradient of the volume data $\mathbf{G}$ is pre-computed as partial derivatives of $\mathbf{I}$, that is, $\mathbf{G} = (\partial f/\partial x, \partial f/\partial y, \partial f/\partial z)$. By slicing the displaced volume proxy mesh $\mathbf{V}'$ at every deformation step, we obtain view-aligned slices of the deformed volume area as discrete samples to incorporate into a texture-based volume-rendering scheme [9]. The voxel values are stored in a 3D texture and translated to RGBA color values via a user-defined transfer function. The volume deformation can be rendered by integrating the proxy geometries in back-to-front order with corresponding color and alpha values at the positions of the initial vertices $\mathbf{V}$ of the tetrahedral mesh. The color values between the vertices are linearly interpolated by referring to the 3D texture. As most parts of this process are executed on GPUs, volume deformation is rendered with a sufficient refresh rate for interactive manipulation.

This scheme works well for generating deformed results, but only without shading. To work with deformed volumes with shading, we need to model the reflection and attenuation of light. We add self-shadowing to the deformed volume here by applying a volume-shadowing model [2] to simulate direct lighting. Preliminarily, we modify the model slightly by recovering attenuation in order to simulate soft shadow. The shadow volume is rendered on GPUs to the fragment buffer object (FBO) and used in the final rendering process. The following section goes into detail on the technique for volume shading.

4.2 Per-vertex gradient computation

To add shading effects to deformed volume models, we have to handle the transformation of the 3D gradient field $\mathbf{G}$ (i.e., the volume gradient). The volume gradient can be computed by basic gradient models [14][19]. These models define the gradient for each voxel, and most usually prepare it in the pre-computation stage. If the volume model is static, volume shading can be achieved well using the pre-computed gradient $\mathbf{G}$. When the volume is to be manipulated, however, the gradient must be changed in real time. Correa et al. presents a volumetric displacement map for GPU-based computation [11]. This map also supports discontinuous deformation of the volume, but the per-voxel inverse matrix computation precludes smooth performance in real time. In our framework, the displacement is formulated for each vertex of the volume proxy mesh. By developing a model for the fast computation of the volume gradient $\mathbf{G}'$ from time-varying sampled vertices, we achieve faster volume shading.

In this section we describe a new volume-shading model for interactive volume manipulation. To approximate transformation of the volume gradient, our model handles local rotation of the gradient field $\mathbf{G}$ not for each voxel, but for each vertex of the pre-defined volume proxy mesh $(\mathbf{V}, \mathbf{E})$. The local coordinates of each vertex are defined based on the positions of the neighboring vertices. Next, a rotation matrix is computed based on the corresponding coordinates before and after deformation. The

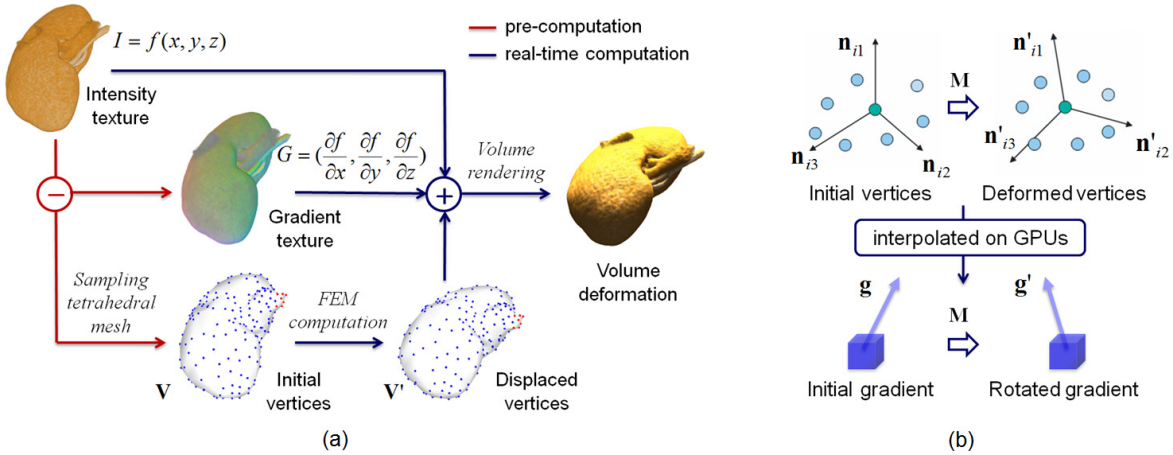

Figure 4 GPU-based computation framework for volume rendering/shading of deformed volume. (a) Overall flow (b) Gradient computation by extracting rotational components using a local frame. The per-vertex gradient is interpolated on GPUs in parallel.

rotation between vertices is linearly interpolated and then used to determine the locally rotated volume gradient. The key advantage of this approach is its ability to interpolate on GPUs. The approach requires no additional displacement textures, inverse matrix computation, or any updates of the initial volume data.

The principal challenge in developing this approach was to find a strategy for extracting a local rotational component for each sampled vertex of the volume proxy mesh. While normal vectors work well for shading deformed surface models, they cannot be directly applied to volume models with internal structures. If we define the rotation components for the internal vertices of the tetrahedral mesh in addition to the mesh surfaces, shading effects can be added even for internal structures of the volume model. To approximate rotation around all vertices of the tetrahedral mesh, we define a local frame for each vertex. The local frame is defined with a local coordinate matrix $\mathbf{N}_i$ by selecting normalized orthogonal bases using the neighboring vertices:

$$\mathbf{N}_i = (\mathbf{n}_{i1}, \mathbf{n}_{i2}, \mathbf{n}_{i3}) \quad (1)$$

where $\mathbf{N}_i \in \mathbf{R}^{3\times3}$ is the orthogonal matrix defining the local coordinate for $\mathbf{v}_i$, and $\mathbf{n}_{ij} \in \mathbf{R}^3 \ (j = 0, 1, 2)$ is the normalized orthogonal basis. A new orthogonal matrix $\mathbf{N}'_i$ for the deformed volume model is defined in the same way based on the corresponding neighboring vertices.

$$\mathbf{N}'_i = (\mathbf{n}'_{i1}, \mathbf{n}'_{i2}, \mathbf{n}'_{i3}) \quad (2)$$

The relationship between these two coordinates is expressed by the following equation:

$$\mathbf{N}'_i = \mathbf{R}_i \mathbf{N}_i \quad (3)$$

where $\mathbf{R}_i \in \mathbf{R}^{3\times3}$ is a rotation matrix for $\mathbf{v}_i$. As $\mathbf{N}_i$ is orthogonal, the inverse matrix $\mathbf{N}^{-1}{}_i$ can be replaced by $\mathbf{N}^{T}{}_i$. Solving this equation for $\mathbf{R}_i$, we obtain

$$\mathbf{R}_i = \mathbf{N}'_i \, \mathbf{N}^{-1}{}_i = \mathbf{N}'_i \, \mathbf{N}^{T}{}_i \quad (4)$$

Consequently, $\mathbf{R}_i$ can be rapidly computed without the inverse matrix calculation. Once the rotational components are obtained for each vertex, we linearly interpolate the rotation between vertices of the proxy geometry. Parallel processing of the interpolation on GPUs enables fast rendering. By multiplying the per-voxel rotation matrix by a pre-computed gradient on GPUs, we finally obtain the rotated volume gradient. With this information, we go on to compute the shaded result based on the Phong-Blinn shading model [4][27].

This computation method is one of several efficient approaches based on Gram-Schmidt orthonormalization for extracting rotational components from displacements of sampled points. Muller et al. introduced this method as a means of eliminating rotational artifacts in deformable modeling [24]. We follow this strategy and describe a new GPU-based volume-shading framework for deformable bodies. The rotational components can also be extracted by polar decomposition, but only at a high computational cost. Thus, we select the former approach for describing the rotation of the volume gradient for each sampled vertex in our volume manipulation framework.

5 Results and Applications

Our direct volume manipulation framework is implemented using C++ and GLSL. Tetrahedral elements are created from adaptively placed vertices using the tetgen library (published at http://tetgen.berlios.de/). Here we examine the quality and performance using both public domain data and clinical CT data. Table 1 summarizes the details of the test data and computation times for both deformation and rendering. The test is performed on a standard PC (CPU: Intel Core2 Duo 3.0 GHz, Memory: 3 GB) with a general-purpose graphic card (nVidia GeForce GTX260 896 MB). See the supplemental movies to confirm the interactive performance of the volume manipulation.

5.1 Direct Volume Manipulation Examples

Several results have been obtained from attempts to apply direct volume manipulation for volume exploration and surgical simulation. To begin, Figure 5 shows the volume-deformation results of test sphere data with inhomogeneous elasticity. We select the sphere-shaped volume areas and directly assign the

Data	Volume size	Vertices	FPS
Spheres	128 × 128 × 128	243	27.6
Engine	256 × 256 × 128	319	21.8
Kidney	256 × 256 × 256	463	16.3
Breast	256 × 256 × 256	873	9.4
Abdomen	512 × 512 × 320	512	8.9

Table 1. Details on test data with computation time for both deformation and rendering.

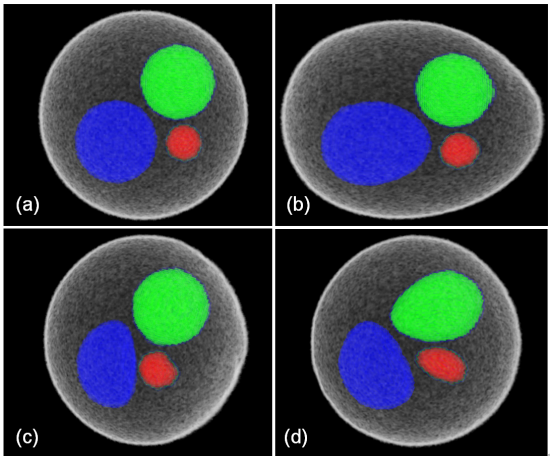

Figure 5 Volume deformation of sphere data when heterogeneous elasticity is set on the rendered image. (a) Initial state, (b)(c) direct, heterogeneous deformation, and (d) homogeneous deformation without adapting the proxy geometry to the volume for comparison.

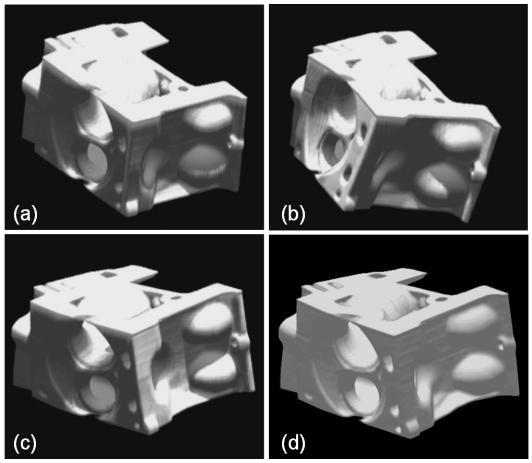

Figure 6 Volume lighting for a deformed engine. (a) Initial state. The light source is placed above the volume. (b)(c) The shade and shadow correctly change as the volume structure is deformed. (d) deformation without shadow or shade update for comparison.

following elastic parameters: 100.0 MPa for the green sphere (hard), 0.1 MPa for the blue sphere (soft), 10.0 MPa for the red sphere and the background spaces, and 1.0 MPa for the white area.

The left side of the white sphere is fixed throughout the test. In Figure 5b, the right side of the white sphere is set as the interaction area and pulled. In Figure 5c, the red sphere is directly grasped and moved toward the blue area. The four spheres are naturally deformed based on their sphere-shaped geometries. The blue sphere is extensively deformed, while the green sphere behaves like a hard object. For comparison to past works, we set the objects in Figure 5d as homogeneous. As in the case previously demonstrated by Georgi et al. [33], three objects are uniformly deformed based on regular tetrahedral grids placed over the whole image space. Adaptation of the proxy geometry to volume data makes it possible to simulate inhomogeneous deformation without manually preparing the mesh. As the results show, our framework can represent visually valid deformation of the volume by reflecting 3D geometry and inhomogeneous elasticity. Perceiving no mesh structures, the user feels that he or she can interact directly with the volume.

Next, we confirm the volume lighting results when manipulating data representing an engine (see Figure 6). The volume of the engine is selected as a manipulation target, and 1.0 MPa is set for the Young's modulus directly on the rendered image. The shade and shadow correctly change as the volume structure deforms. In addition, the gradient computation on the GPUs successfully maintains the interactive performance, as the supplemental movie demonstrates. Here we test volume cutting and deformation using volume data from a breast CT. As shown in Figure 7, we simulate two different approaches for preoperative planning of an actual surgery by a cardiovascular surgeon. The incision is interactively input on the rendered image and deformed by mouse manipulation. The internal structure can be partly seen through the created space. The limited view provides useful information for planning the surgical approach. This demonstrates how our framework can support a discontinuous geometry and deformation.

5.2 Applications for Surgical Simulation

Figure 8 shows clinical application of our technique for geometry-encoded volume deformation. The overall configuration of the kidney CT volume is rendered by the method explained in section 3. A practitioner in urology manipulates the kidney to visualize the actual surgical process and check the 3D structures of the blood vessels. He begins by grasping and deforming the blue interaction area to check how the blood vessels connect to the kidney. The manipulation leaves the bone structure unaffected, while surrounding tissues such as blood vessels move together with the kidney. Next, the practitioner selects a part of the kidney artery and changes its elasticity (Figure 8c), softening and deforming the artery selectively as he sees appropriate. Unlike the related approaches for volume manipulation [13][28], this approach enables direct interaction with the rendered image for all of the procedures. The volume proxy mesh with the local geometry undergoes a visually valid deformation without manual mesh modeling. These features are our main contribution to the field of interactive volume manipulation.

Finally, we tested the direct volume manipulation of complex abdominal structures. Abdominal data have complex internal structures such as those shown in Figure 9a. Yet in this clinical situation, it is impractical to mesh all of the meaningful objects from patient-specific data. In this case, on-the-fly encoding of local geometries works well. Effective transfer functions and partial clipping of the volume allow us to directly select some of the internal structures. The manipulation targets are a liver (Figure 9b-c), an aortic aneurysm (Figure 9e), and a kidney (Figure 9f). The deformation of the liver is useful for visualizing hidden blood vessels. As the aortic aneurysm is selectively manipulated, the spine behind the aorta is unaffected.

6 Discussion and Limitation

The advantage of the technique we propose is its geometry-based approach for direct manipulation of the volume without any manual mesh modeling. This approach at its current stage of implementation has limitations, however, as the adaptive proxy geometry framework is still unable to support extremely large deformations such as twists and rotations of the target objects. If the kidney is rotated more than 180 degrees, for example the tetrahedral elements can reverse and yield visual artifacts. While this may pose no immediate problem in the current application to surgical simulation, adaptive mesh refinement based on large manipulations by users will remain a challenging issue in the volume-deformation literature. Inhomogeneous elasticity modeling is also still restricted to very simple objects, as maintain

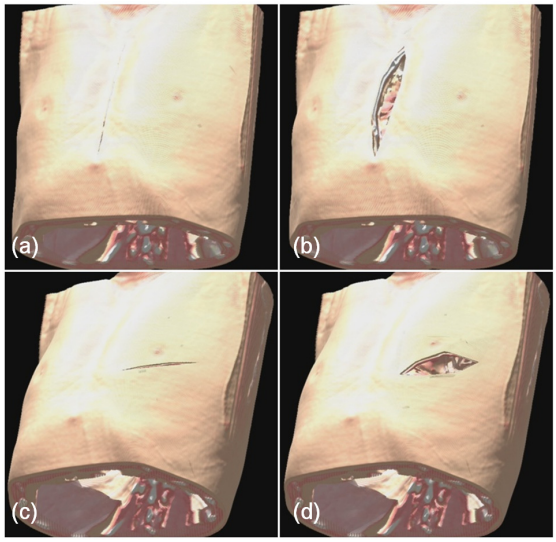

Figure 7 Volume cutting and deformation of an incision. (a) The incision can be interactively inputted on the rendered image, and (b) deformed by mouse manipulation.(c) Another incision and (d) discontinuous deformation.

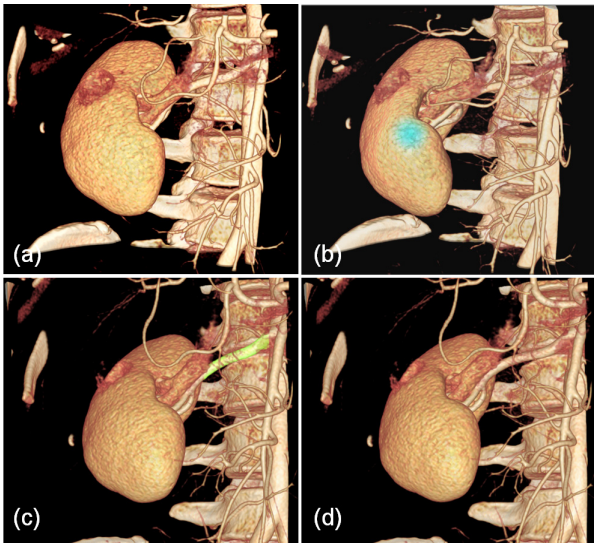

Figure 8 Medical application for preoperative planning in a laparoscopic kidney surgery: (a) Initial volume data. (b) Local deformation of the kidney. (c) Volume selection of the kidney artery. (d) The artery is selectively softened and deformed easily.

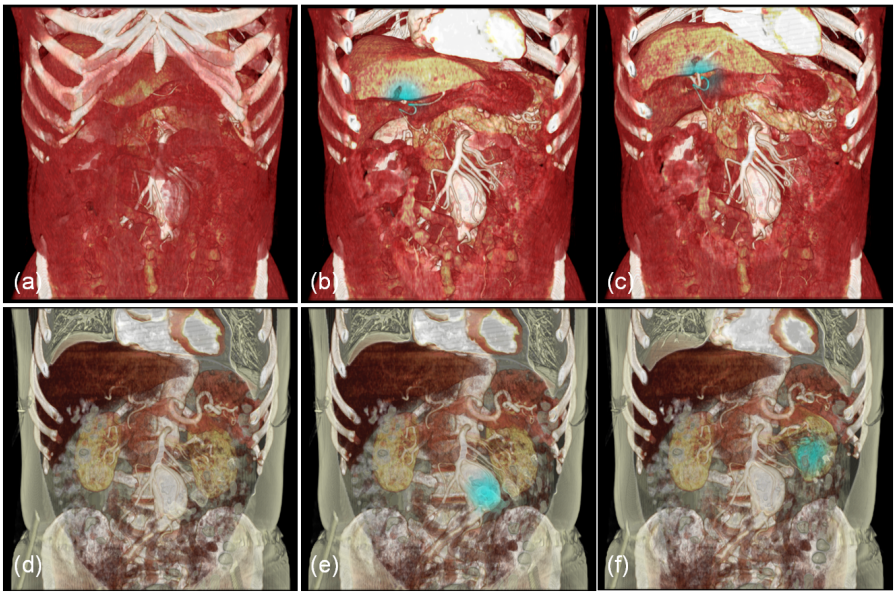

Figure 9 Volume manipulation of a complex abdominal structure: (a) Volume rendering result of a whole abdominal CT. (b) Direct selection of internal structure on the clipping plane. The blue area shows the interaction area. (c) Deformation of liver for exploring blood vessels by the method applied in actual surgery. (d) Another volume rendering with clipping via a different transfer function. (e) Direct, selective volume deformation of an aortic aneurysm with no deformation of rigid objects such as the spine. (f) A kidney deformation result.

interactive manipulation can only be maintained by sampling the target volume with hundreds of vertices. This is basically a trade-off problem between complex topology modeling and the interactive performance of overall algorithms. As time passes, however, on-the-fly, local geometry modeling is likely to acquire the required accuracy and interactivity.

7 Conclusions

This paper has proposed a new design to represent interactive, direct manipulation of volume data on volumetrically rendered images. The key concept is to encode the local geometry through the user's interaction with the volume. The captured geometry is stored into the volume proxy mesh on-the-fly without being shown to the user. We have also presented and implemented a volume-shading/shadowing scheme in this framework on GPUs for visualizing time-varying deformable objects. The paper has shown several examples of manipulation for preoperative planning and medical illustration. We hope that the techniques presented here will be applied to more complex volume data and theoretically contribute to volume-visualization research. Our goals in future research will be to improve the framework and develop a practical medical application for preoperative planning

and medical illustration. Our future work will be to enhance the reality of both deformable models and real-time rendering.

ACKNOWLEDGEMENTS

This research has been carried out as a project entitled ”Development of tailor-made surgical navigation system" under contract with the Innovation Plaza Kyoto, Japan Science and Technology Agency (JST). A part of this study was supported by a Grant-in-Aid for Scientific Research for Young Scientists (A) (21680044) from The Ministry of Education, Culture, Sports, Science and Technology, Japan.

REFERENCES

[1] M. Agus, A. Giachetti, E. Gobbetti, G. Zanetti and A. Zor-colo, "Adaptive techniques for real-time haptic and visual simulation of bone dissection", Proc. of IEEE VR, pp. 102-109, 2003.

[2] U. Behrens and R. Ratering, "Adding shadows to a texture-based volume renderer", Proc. Volume Visualization Symposium, pp. 39-46, 1998.

[3] J. Berkley, G. Turkiyyah, D. Berg, M. Ganter and S. Weghorst, "Real-time finite element modeling for surgery simulation: an application to virtual suturing", IEEE Trans. on Visualization and Computer Graphics, Vol. 10, No. 3, pp.314-325, 2004.

[4] J. Blinn, "Models of light reflection for computer synthesized pictures", Computer Graphics, Vol. 11, No. 2, pp.192-198, 1977.

[5] M.~Botsch and O.~Sorkine, "On linear variational surface deformation methods", IEEE Trans. on Visualization and Computer Graphics, Vol. 14, No. 1, pp.213--230, 2008.

[6] S. Bruckner and M. E. Groller, "Volumeshop: An interactive system for direct volume illustration", Proc. of IEEE Visualization, pp.671-678, 2005.

[7] S. Bruckner and M. E. Groller, "Exploded views for volume data", IEEE Trans. on Visualization and Computer Graphics, Vol. 12, No. 5, pp. 1077-1084, 2006.

[8] K. Burger, J. Kruger and R. Westermann, "Direct volume editing", IEEE Trans. on Visualization and Computer Graphics, Vol. 14, No. 6, pp.1388-1395, 2008.

[9] B. Cabral, N. Cam and J. Foran, "Accelerated volume rendering and tomographic reconstruction using texture mapping hardware", Proc. Volume Visualization Symposium, pp. 91-98, 1994.

[10] M. Chen, D. Silver, A. Winter, V. Singh and N. Cornea, " Spatial transfer functions: A unified approach to specifying deformation in volume modeling and animation", Proc.of Volume Graphics, pp. 35-44, 2003.

[11] C. D. Correa, D. Silver and M. Chen, "Discontinuous displacement mapping for volume graphics", Eurographics / IEEE VGTC Workshop on Volume Graphics, pp. 9-16, 2006.

[12] C. D. Correa, D. Silver and M. Chen, "Feature aligned volume manipulation for volume illustration", IEEE Transactions on Visualization and Computer Graphics, Vol. 12, No. 5, pp.1069-1076, 2006.

[13] C. D. Correa, D. Silver and M.~Chen, " Volume deformation via scattered data interpolation", Proc. of the Sixth Eurographics / IEEE VGTC Workshop on Volume Graphics, pp. 91-98, 2007.

[14] R. A. Drebin, L. Carpenter, and P. Hanrahan, "Volume rendering", Proc. ACM SIGGRAPH, pp. 65-74, 1988.

[15] K. Engel and T. Ertl, "Interactive high-quality volume rendering with flexible consumer graphics hardware", Eurographics State of The Art Report, 2002.

[16] S. Fang, S. Huang, R. Srinivasan and R. Raghavan, "Deformable volume rendering by 3D texture mapping and octree encoding", Proc.of IEEE Visualization, p73, 1996.

[17] M. Hadwiger, J. M. Kniss, C. Rezk-salama, D. Weiskopf, and K. Engel, "Real-time Volume Graphics", A K Peters, 1st edition, 2006.

[18] M. Levoy, "Efficient ray-tracing of volume data", ACM Trans. on Graphics, Vol. 9, No. 3, pp.256-261, 1990.

[19] J. Kniss, S. Premoze, C. Hansen, P. Shirley and A. McPherson, " A Model for Volume Lighting and Modeling.", Vol. 9, No. 2, pp.150-162, 2003.

[20] W. Lin and R. A. Robb, " Dynamic volume texture mapping and model deformation for visually realistic surgical simulation, Proc. Medicine Meets Virtual Reality}, pp. 198-204, 1998.

[21] Y. Lipman, O. Sorkine, D. Levin and D. Cohen-Or, "Linear rotation-invariant coordinates for meshes", Proc. ACM SIGGRAPH, pp. 479-487, 2005.

[22] Y. Masutani, Y. Inoue, K. Ishii, N. Kumai, F. Kimura and I. Sakuma, "Development of surgical simulator based on fem and deformable volume rendering", Proc. SPIE, pp.500-507, 2004.

[23] M. J. Mcguffin, L. Tancau and R. Balakrishnan, "Using deformations for browsing volumetric data", Proc. of IEEE Visualization, pp.401-408, 2003.

[24] M. Muller, J. Dorsey, L. McMillan, R. Jagnow, and B. Cutler, "Stable real-time deformations", Proc. of ACM SIGGRAPH/Eurographics Symposium on Computer Animation, pp. 49-54, 2002.

[25] A. Nealen, M. Muller, R. Keiser, E. Boxerman and M. Carlson, "Physically based deformable models in computer graphics", Proc. of Eurographics, pp. 809-836, 2005.

[26] B. Pflesser, R. Leuwer, U. Tiede and K. Hohne, "Planning and rehearsal of surgical interventions in the volume model", Stud. Health Tech. Inform, Vol. 70, pp. 259-264, 2000.

[27] B. T. Phong, "Illumination for computer generated pictures", Communications of the ACM, Vol. 18, No. 6, pp.311-317, 1975.

[28] C. Rezk-Salama, M. Scheuering, G.~Soza and G. Greiner, " Fast volumetric deformation on general purpose hardware", Proc.of The ACM SIGGRAPH / EUROGRAPHICS workshop on Graphics hardware, pp.17-24, 2001.

[29] R. Westermann and C. Rezk-Salama. Real-time volume deformations, Computer Graphics Forum, Vol. 20, No. 3, pp.443-451, 2001.

[30] S. Schaefer, T. Mcphail and J. Warren, "Image deformation using moving least squares", ACM Transaction on Graphics, Vol. 25, No. 3, pp.533-540, 2006.

[31] V. Singh, D. Silver and N. Cornea, "Real-time volume manipulation", Proc. of Eurographics/IEEE TVCG Workshop on Volume graphics, pp. 45-51, 2003.

[32] H. Takiuchi, Y. Mori, H. Shima, M. Tanooka, S. Hirayama, and N. Nakao, "Kidney displacement simulator for retroperitoneal laparoscopic nephrectomy, Journal of Urology,. Vol. 174, No. 6, pp.2111-2114, 2005.

[33] J. Georgii and R. Westermann, "A Generic and Scalable Pipeline for GPU Tetrahedral Grid Rendering", Proc. IEEE Visualization, pp.1345-1352, 2006.

[34] J. Georgii and R. Westermann, "Interactive Simulation and Rendering of Heterogeneous Deformable Bodies", Proc. of Vision, Modeling and Visualization, pp.383-390, 2005.

[35] J. Georgii and R. Westermann, “Corotated Finite Elements Made Fast and Stable", In Proc. of the 5th Workshop on Virtual Reality Interaction and Physical Simulation", 2008.

[36] D. Bielser, P. Glardon, M. Teschner, M. Gross, A State Machine for Real-time Cutting of Tetrahedral Meshes, Graphical Models, Vol. 66, No. 6, pp. 398-417, 2004.

[37] J. Mensmann, T. Ropinski, K. H. Hinrichs, "Interactive Cutting Operations for Generating Anatomical Illustrations from Volumetric Data Sets", Journal of WSCG, Vol. 16, No. 1-3, page 89-96, 2008.

[38] T-Y. Lee, Y-C Lin, Y.N. Sun & L. Lin, "Fast Fast Feature-Based Metamorphosis and Operator Design", Computer Graphics Forum, Vol. 17, No. 3, pp. 15-22, 1998.

[39] W. Chen, X. Liang, R. Maciejewski and D. S. Ebert, "Shape Context Preserving Deformation of 2D Anatomical Illustrations", Computer Graphics Forum, Vol. 28, No. 1., pp. 114-126, 2008.