Volume Graphics (2003)
I. Fujishiro, K. Mueller, A. Kaufman (Editors)

Real-Time Volume Manipulation

V. Singh* D. Silver† N. Cornea‡

Dept. of Electrical and Computer Engineering
Rutgers, The State University of New Jersey
Piscataway, NJ 08855-0909
http://www.caip.rutgers.edu/vizlab.html

Abstract

In this paper, we describe a set of algorithms and an implementation (called VolEdit), for interactively manipulating 3D volumetric objects (datasets). The system utilizes skeletons, which allows users/animators to interactively and intuitively specify the location and type of deformation desired. The skeleton is extracted automatically from the volumetric model and indexes the appropriate part of the volume that needs to be transformed by defining piecewise bounds of the volume. The deformed volume is then reconstructed and rendered using commodity graphics cards. The system performs in real-time with near-interactive speeds. The VolEdit system is demonstrated with two volumes, the Visible Human Dataset and a colon MRI dataset.

Keywords: Skeleton, Deformation, Bounding Boxes, Texture Mapping, Manipulation, Animation, Mid-Plane Geometry

1 Introduction

Three dimensional volume datasets are becoming common in the medical and scientific domains. These datasets are the result of mathematical simulations, such as computational fluid dynamics, or are generated by observational equipment such as CT, MRI, and ultrasound. There is a significant amount of research in techniques to render these datasets quickly and more realistically. Volume rendering API's are available for many different software packages, and new graphics card support hardware accelerated volume rendering [26][37][25][22][7]. However, most of the volume manipulation tools use cutting planes. Manipulating volume models includes deforming, animating, smoothing, reshaping and moving/removing parts of volumes for scientific analysis and computer graphics applications. (The term volume datasets refers to a cubic dataset $N \times M \times P$ with scalar values. An MRI dataset of 512^3 resolution, is one example. The term volumetric model or volume model refers to a region of the dataset segmented from the background. Most of the work presented here assumes that the model can be segmented from the background.) It is hard to pick regions, cull occluding regions, and it is impossible to reshape or rearrange part of a volume image. These are tasks that are potentially important to the researchers/scientists. The few methods that are available are computationally expensive and non-intuitive.

The lack of tools hampers the more widespread uses of volumes. Two obvious examples are virtual reality and computer graphics. Volumes are usually converted to polygons before manipulation in these domains. For virtual reality, either polygons or very small 3D datasets are used since real-time manipulation with force feedback is necessary. For computer graphics, volumes cannot be manipulated by animation packages so polygons are used instead. Volume graphics applications would benefit from more intuitive manipulations [5][6]. An interactive system for manipulating and animating volumes may also have applications in games and education. In visualization, interactive volume manipulation may be useful for presenting new views of the data. For example, cutting planes are currently used in volume datasets to cull parts of the volume and "see inside" the volume more clearly. While cutting planes are an indispensable part of volume visualization, they are very difficult to place since they are not shape based. Using the Visible Human Dataset [41] as an example, one may want to view a particular interior organ from the side (to determine thickness/thinness). However, when viewing the Visible Human Dataset from the side the arms occlude the main body. It would be much easier to “move the arm out of the way”, and then place the cutting plane at a desired location. This should be available even without prior segmentation to determine which voxels constitute the arms. Reshaping the volume may also be useful. One example is colon straightening [1][31]. While radiologists look at the colon as is, pathologists may need to see it “stretched out”. A visualization program that would allow different specialists to view a volumetric dataset in

* viksingh@caip.rutgers.edu
† silver@caip.rutgers.edu
‡ cornea@caip.rutgers.edu

different manners would be very helpful. Furthermore, unrolling a twisted volume may allow a specialist to view the entire contents at once instead of snapshot by snapshot as is done with virtual navigation. Volume deformation may also have applications in laparoscopy [38] and brain modeling.

In [13], a system for creating volume animations was presented. It was based upon generating a "volumetric skeleton". The skeleton could be animated and a new volume was reconstructed about the transformed skeleton. The new volume was then rendered. Although the system was intuitive and allowed an animator to utilize existing commercial animation software, the reconstruction and rendering was slow and could not be used for interactive manipulation. In [36], a methodology was presented which was able to deform volumes using hardware acceleration by warping the hull of the volume, and in [27] the deformation was modeled by warping the volume in texture space. However, there was no way to intuitively define the manipulation and deformation. In this paper, we develop a new algorithm, using components of [13] and [36], to interactively manipulate, deform and animate volumes. We demonstrate how a fast volumetric manipulation system can allow the user to preview new types of visualizations. The algorithms were implemented in a program called VolEdit, which allows the user to manipulate the volume and render it. VolEdit can also be used to read in a set of transformations from a commercial animation package and apply them to a volume in real-time (3fps per new reconstructed volume).

In the next section, we describe related research. This is followed by the algorithm overview and implementation. Section 5 demonstrates some resulting animations and manipulations applied to 3D volumetric datasets.

2 Related Work

The research related to the work presented here falls into four broad categories, namely, hardware acceleration of volume rendering, volume deformation, volume animation, and volume editing. There has been a lot of recent work in hardware acceleration of volume rendering [26][37][25][22][7]. The hardware accelerated volume rendering techniques rely on the texture mapping capability of current graphics cards [42]. These techniques sample the volume using polygon slices along the viewing direction (viewport aligned). The polygons get textured with the 3D volume using the 3D interpolation capabilities of the hardware. These polygons are then composited to generate the final volume rendered image. There have been many recent attempts to map other graphics and geometric algorithms to hardware. For example, in [25] a method was presented that demonstrated the speed up of ray tracing, in [16] a method was presented for rendering unstructured grids, in [20] a method was presented to compute a 3D voronoi diagram.

Most of the research in volume deformation involves applying a transformation to every voxel in the object (free-form deformation) or defining a physical model for the full object (physically-based deformation). The computations can include spring-like models, continuum models [9], finite element methods [4] or landmark deformations [8]. Gibson and Mirtich [9] have presented a survey of work done in modeling deformable objects. Physically-based animation is used for realistic modeling of collision and deformation. In [4], a system is presented using a volumetric mass spring model and an FEM model for animating volume objects. In [10], a 3D Chain Mail algorithm is used to propagate deformation through a volume rapidly. These are sophisticated techniques requiring specification of material properties like elasticity and are sometimes an over-kill for simple manipulation. In [23], volumes are indirectly deformed by deforming the rendering rays. However, the method is not intuitive for users and the computational time is proportional to the number of deflectors (which can be large). In [21], a method was presented to compute 3D model deformation in hardware with pre-computed modal vibrations. The method does not directly deform volumetric objects (the volumes are used for the pre-computation). Another form of volume animation is targeted deformation, or volume morphing [12][19][18][24]. These methods cannot be used in the context of manipulation or deformation since a target model may not available. Furthermore, all of these methods could benefit from an intuitive way to specify *feature-based* morphing, i.e., which part of the source object should morph to which part of the target object. The method described in this paper could be used to interactively determine and render the morph.

A hardware accelerated volume deformation algorithm is presented in [36]. The volumetric data is first preprocessed to extract an isosurface. The isosurface is sliced parallel to the viewing direction and these slices are composited using hardware support (the isosurfacing and slicing is done in software). The advantage of using this system is that the isosurface coordinates can be deformed (for example, by "pulling" a vertex) and the resulting image will look like a warped volume. However, there is no intuitive way to specify manipulations and the isosurfaces must be convex, further limiting the type of manipulation that could be specified.
In [13], a volume animation system has been presented which allows an animator to specify an animation for a volume in the same way that they would specify an animation for a polygonal model. A volumetric skeleton is computed directly from the volume using the distance transform. The skeleton can then be deformed using a standard animation package. A volume is reconstructed about the skeleton using the distance field. This method

was used to create realistic animations of the Visible Human Dataset [41]. In [14], a method was also presented to animate the Visible Human Dataset. This method subdivided the volume by hand into logical blocks. Each block could be transformed and the volume copied into the new location. Discretization errors resulted at the block bounds.

In [33][34][44], volume-sculpting systems with force-feedback devices are presented. These systems allow the user to sculpt a 3D volume from scratch, or use a sculpting tool to take away parts of a volumetric model. They do not permit editing or moving (in the kinematic sense) parts of an existing volumetric model. There has also been work on developing new interfaces for volume visualization and in particular for medical visualization (see [17] for one example). In this paper, we present another technique that could supplement the existing methodologies and aid in facilitating the manipulation of volumetric datasets.

3 Interactive Volume Manipulation

Our goal is to build a system that enables interactive manipulation of volumetric datasets. While the methodology described in [13] is powerful for animation, it is very slow since each reconstructed voxel has to be shaded from the original volume. The reconstruction process could take 15 minutes to 1 hour depending upon size and transformations. Furthermore, breaking at the joint resulted for large angular rotations. In this paper, we present an interactive method to perform intuitive volume manipulation and animation. The new methodology is also based upon using a skeleton. However, the skeleton is only used to help specify deformation and logically subdivide the volume into components. The skeleton can be deformed interactively, through the VolEdit interface, or off-line, for example, using a standard animation package such as Character Studio [40] or Maya [43].

To render the deformed volume, a bounding rectangle is placed about each skeleton component. The rectangular boxes are sliced, and the resulting polygons are mapped back to the original volume to determine the texture on that polygon [36][37]. The polygons are then composited to render the image. The bounding cubes are a simple geometry that can support interactivity.

This process can be summarized by four basic steps: skeletonize the volumetric object (thin the volume and determine the bones and joints), manipulate the skeleton (either interactively or using an animation tool), compute the bounding rectangles (determine planar slices and texture), and render using compositing. The texture mapping and compositing is done by the hardware. An outline of this process is given in Figure 1. In the next sections, the various steps in the process are explained in detail.

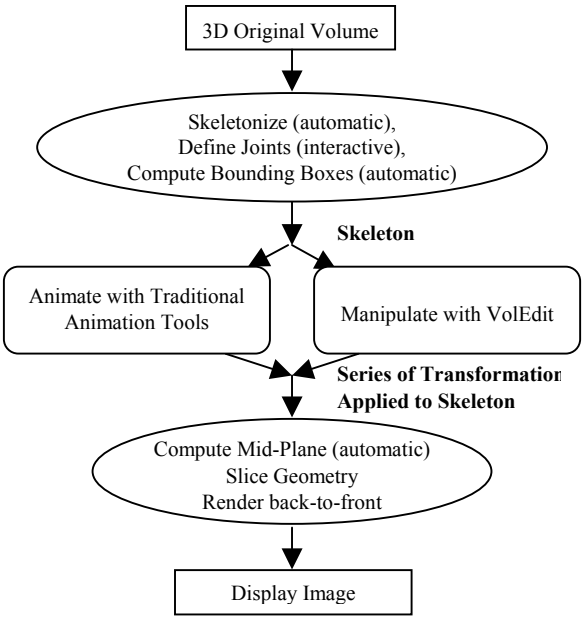


Figure 1. Volume manipulation pipeline.

3.1 Skeleton Extraction

The first step of the process is to compute a skeleton from the volumetric object. The skeleton is a useful shape abstraction that captures the essential topology of an object. It is related to the medial-axis, which is the set of points centered with respect to the boundaries of an object. For three-dimensional objects, the medial-axis is not just a curve, but a surface, often called a medial-surface. The skeleton necessary here is a centerline representation of the medial-surface for 3D shapes. Since the skeleton is a reduced representation, an intuitive method for shape deformation involves manipulating the skeleton that causes corresponding deformations in the object. The skeleton of a volume object can be extracted by using a variety of methods (see [13] for a review). For this work, we use the approach described in [15] where a parameter-controlled volume thinning algorithm is presented which computes skeletal voxels of different densities. The algorithm first computes the distance field and then progressively thins the volume until a desired thinness is achieved. The density of the graph is determined using the thinness parameter and can be further reduced using clustering. The algorithm is automatic and results in a thin set of "skeletal voxels".

For animations, joints of revolution must be chosen from the skeletal voxels. Unfortunately, these cannot be detected automatically without knowledge of the biology of the object in the dataset. Joints and bones from the graph are chosen by the user/animator interactively to correspond to appropriate features of the underlying

volumetric model. For example, for the legs given in Figure 4, the joint at the knee is chosen for motion as opposed to one in the middle of the shin. For the colon in Figure 6, the joints can be specified arbitrarily. Because a distance field calculation is used, the skeleton retains information about the boundaries of the volumetric object. Other line-skeleton algorithms can also be used [32] [29] with appropriately defined bounding boxes. The bounding rectangular regions are automatically computed from the skeletal segments.

3.2 Manipulation

Once the skeleton is defined, different components of the volumetric object can be earmarked for transformation. Our volume animation pipeline supports two modes of manipulation: offline manipulation and interactive manipulation. For offline manipulation the skeleton is imported into a standard computer graphics animation package such as Character Studio or Maya. Figure 4 shows a running sequence implemented using Character Studio and motion capture data. The transformations applied to the skeleton are forwarded to the next stage of the pipeline for rendering. Alternately, the skeleton can be manipulated interactively by the VolEdit GUI. A picture of the VolEdit interface is shown in Figure 3. It allows a user to pick a "joint" and specify rotation about a joint and/or scale about a bone.

3.3 Geometry and Rendering

For each skeletal segment a rectangular-bounding box is automatically defined, enclosing a logical segment of the volume. The box length is determined from the length of the bone and the width from the distance field values of the skeletal voxels (these are saved with the nodes when the skeleton is computed). In this work, the underlying concept is to use rectangular boxes to represent the shape of the different parts of the volume. This geometry, though a coarse approximation to the actual shape, suffices for reconstructing most shapes and provides a simple geometry for fast rendering. The boxes are also good for fuzzy bounded volumes, which may not have a definite boundary, and can be used for other volume graphics applications [5]. More exact representation, such as isosurfaces, can also be used. Because the cubes are rigid, breaking can occur at the joint. An example of this can be seen in Figure 5(a) where the motion at the knee joint breaks the volume. This also occurred in [13][14]. There are different techniques from standard polygonal animation that can be used to overcome the breakings such as modeling NURBS at the joints, drawing spheres and using meta-objects [30]. A simple method for boxes is the mid-plane algorithm [3]. The mid-plane algorithm maintains connected rectangular regions during animation by adding planar polygons at the joints. Interpolation is done between two end planes of the adjoining bounding boxes to determine a "mid-plane". The mid-plane is then connected to the two end planes forming two new sheared bounding boxes. The advantage of using mid-planes is that it requires a minimum of geometric processing and so it is very fast. We are currently investigating smoother geometries while still maintaining interactivity. Once the geometry is defined, slices along the viewport are computed (the spacing of the slicing can be given as a parameter to the system). We model the deformation through a warped geometry mapped back to the original texture space. Each sliced polygon is mapped back to the original texture to determine the appropriate color value at each polygon vertex. The texture-mapped polygons are then composited back-to-front which effectively volume renders the deformed volume [26][37][25][22]. If the volume is rotated, the underlying geometry has to be resliced and composited.

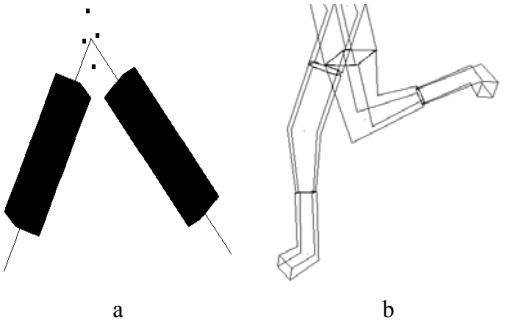


Figure 2. (a) The Mid-Plane geometry (b) a geometric model used for visible human animation using mid-plane geometry

4 The VolEdit User Interface

The VolEdit interface allows a user to interactively edit a skeleton and then render the resulting (animated/manipulated) volume. The user can pick a bone to rotate and/or scale. The user can also disable a bone (effectively cutting out a segment of the volume). Once the desired manipulation is determined, the volume can be rendered. The VolEdit interface also allows the user to rotate the image to a different viewpoint. In the offline mode, VolEdit will read in a previously determined sequence of transformations. These are then applied to the volume to create a real-time volume animation. The animated volumes can also be rotated while viewing. A picture of the VolEdit interface is shown in Figure 3. In this figure, the skeleton geometry is seen with the volume rendered model. The right shin was "moved away" by the user. Currently sliders are used to rotate the entire volume while the pick feature is used to rotate individually selected limbs. A more expressive interface could also be built similar to commercial animation packages, or specifically for medical applications.

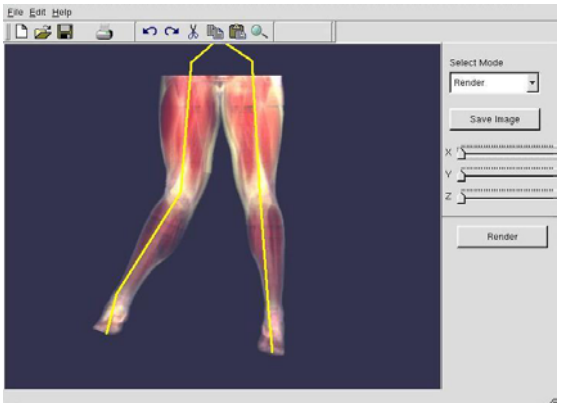


Figure 3. The VolEdit User Interface (the "skeleton lines" are shown in yellow). The volume is displayed after the user rotated the right shin.

5 Results and Discussion

The system was implemented using OpenGL on a Pentium 4, 1.7GHz processor, and an Nvidia GeForce3 video card. Two different volumetric datasets were used to demonstrate this system. The first is a 256x512x256 portion of the Visible Male Dataset (the legs). The second is an MRI of a colon, also 256x512x256. (The colon is courtesy of the Mayo Clinic). Figure 3 shows an image of the deformed legs with the skeleton superimposed in yellow. In Figure 4, four frames (out of a total of 24 frames) of a jogging sequence are shown. This animation used motion capture data and the skeleton transformation was done in Character Studio. The skeleton was then imported into VolEdit, and the volumes were rendered in the appropriate animated poses. A frame rate of 3fps (frames per second) was achieved compared to a reconstruction time of about 20 minutes per frame in [14]. This does not include the time to read in the texture. (This time was averaged over all of the different animations/rotations performed on both volumes.)

During the playback, the volume can be rotated for an alternate view (Figure 4). Note how the actual bones can be seen during the animation. An mpeg version of the animation is available at [39] (a full color software version of the animation is also available). Figure 5 shows a closeup with the joint, with and without the mid-plane geometry. The advantage of the mid-plane geometry "box paradigm" is that it is simple, convex, and allows cutting planes to be calculated quickly. It can also be used in a general volume graphics environment [5]. However, the bounding box is not always an accurate and tight fit and can cause visible artifacts. In the animation, when the knee is bent completely (at 90 degrees), artifacts are visible because of the boxes and mid-plane geometry. More complex surfaces (such as isosurfaces) and geometric/animation techniques such as NURBS, bone weighting [2], or matrix palette thinning [42] may smooth out the artifacts. We are currently investigating these options in terms of accuracy and performance cost.

Figure 6 displays a stretched out colon with its corresponding stretched out skeleton. Three different stages of the stretch are shown. The stretch was performed interactively, though an automatic alternative can be developed using a line mapping algorithm. A faster frame rate can be achieved by further optimizing the software component and or sampling the bounding rectangles at a lower rate (the figures presented here used a sample rate based upon the dimension of the volume along the viewing direction). The available texture memory on today's graphics cards limits the volume size.

While the actual physics of the animation is not accurate (i.e., no muscle deformation such as contraction and expansion is modeled), the methodology presented does allow the user to get a fast preview of what the animation/deformation would look like. We are also investigating more realistic muscle deformations [11]. (There are numerous research results for modeling tissue and muscle deformations, for example, [28] which uses the mid-plane geometry, or [21] which uses hardware acceleration and finite element modeling.) Simple physics can be added to the skeleton model described in this paper to create more realistic animations or provide fast feedback for virtual reality applications.

In this paper, we have demonstrated the feasibility of interactive manipulation of volumetric objects. The manipulation is unrestricted, i.e., the user is able to rotate and deform parts of the volume intuitively. The manipulation is based upon a volumetric skeleton that supports rotation about joints, scaling along bones and culling. The skeleton and associated joints decompose the volume into a set of components, which can be sliced and rendered using hardware acceleration. The system is simple and allows a user to manipulate a volume and see the results in real-time.

Acknowledgements

We would like to thank Nikhil Gagvani for his help in this project. We gratefully acknowledge the support from NSF (0118760).

References

[1] A. Bartroli, R. Wegenkittl, A. Konig and E. Groller, Nonlinear Virtual Colon Unfolding, *IEEE Visualization 2001 Proceedings*, October 2001.

[2] J. Bloomenthal, Medial Based Vertex Deformation, *Symposium on Computer Animation, ACM SIGGRAPH, 2002.*

[3] J. Chadwick, D. Haumann, and R. Parent. Layered Construction for Deformable Animated Characters. In *Proceedings of SIGGRAPH 89*, 23(3), pages 243-252 (July 1989, Boston, Mass.)

[4] Y. Chen, Q. Zhu and A. Kaufman. Physically-based Animation of Volumetric Objects. In *Proceedings of IEEE Computer Animation '98,* pages 154-160, 1998

[5] M. Chen, D. Silver, A. Winter, V. Singh, and N. Cornea, Spatial Transfer Functions – A Unified Approach to Specifying Deformations in Volume Modeling and Animation, V*olume Graphics 03, July 2003.*

[6] M. Chen and J. Tucker, Constructive Volume Geometry, Computer Graphics Forum, 19(4), 2000.

[7] K. Engel, M. Kraus and T. Ertl, High-Quality Pre-Integrated Volume Rendering Using Hardware-Accelerated Pixel Shading. *In SIGGRAPH/EUROGRAPHICS Graphics Hardware Workshop, 2001*

[8] S. Fang and R. Srinivasan. Volumetric CSG – A Model-Based Volume Visualization Approach. In *Proceedings of 6th International Conference in Central Europe on Computer Graphics and Visualization,* pages 88-95, 1998.

[9] S. F. Gibson and B. Mirtich. A Survey of Deformable Modeling in Computer Graphics. Technical Report TR97-19, MERL, November 1997. URL: *http://www.merl.com/reports/TR97-19/TR97-19.ps.gz*

[10] S. F. Gibson. 3D Chain Mail: A Fast Algorithm for Deforming Volumetric Objects. In *Proceedings of 1997 Symposium on Interactive 3D Graphics*, April 1997.

[11] J. Gourret, N. Magnenat-Thalmann and D. Thalmann, Simulation of Object and Human Skin Deformations in a Grasping Task, *Computer Graphics (SIGGRAPH '89)*, Volume 23, 1989.

[12] S. Fang, R. Srinivasan, R. Raghavan, and Joan Richtsmeier. Volume Morphing and Rendering – An Integrated Approach. *Computer Aided Geometric Design*, 1998.

[13] N. Gagvani and D. Silver. Animating Volumetric Models. *Graphical Models and Image Processing, Academic Press*, March 2002.

[14] N. Gagvani and D. Silver. Animating the Visible Human Dataset. *Proceedings for The Visible Human Project Conference*, 2000.

[15] N. Gagvani, and D. Silver. Parameter Controlled Volume Thinning. *Graphical Models and Image Processing, Academic Press,* vol. 61, no. 3, 1999.

[16] S. Guthe, S. Roettger, A. Schieber, W. Strasser and T. Ertl. High-Quality Unstructured Volume Rendering on the PC Platform. In *Proceedings EG/SIGGRAPH Graphics Hardware Workshop '02 http://www.graphicshardware.org/*, 2002.

[17] K. Hinckley, R. Pausch, D. Proffitt, N. Kassell, Two Handed Virtual Manipulation, *ACM Transactions on Computer-Human Interaction, 5:3*, September 1998.

[18] J. F. Hughes. Scheduled Fourier Volume Morphing. *Computer Graphics,* 26(2):43-46, July 1992.

[19] T. He, S. Wang, and A. Kaufman. Wavelet-Based Volume Morphing. In *Proceedings of Visualization 94,* pages 85 – 91, Los Alamitos, CA, 1994, IEEE Comuter Society Press.

[20] K. Hoff III, T. Culver, J. Keyser, M. Lin and D. Manocha. Fast Computation of Generalized Voronoi Diagrams using Graphics Hardware. In *Proceedings of the 26th annual conference on Computer Graphics and Interactive Techniques*, pages 277-286, 1999.

[21] D. James and D. Pai, DyRT: Dynamic Response Textures for Real Time Deformation Simulation with Graphics Hardware, *Proceedings of ACM SIGGRAPH*, July 2002.

[22] S. Krishnan, C. Silva, and B. Wei. A Hardware-Assisted Visibility-Ordering Algorithm with Applications to Volume Rendering. *Data Visualization 2001,* pages 233-242, 2001.

[23] Y. Kurzion and R. Yagel. Space Deformation using Ray Deflectors. *6th EuroGraphics Workshop on Rendering 95,* pages 21-32, 1995.

[24] A. Lerios, C. Garfinkle and M. Levoy. Feature-Based Volume Metamorphosis. In *Proceedings of SIGGRAPH 95,* August 1995.

[25] T. Purcell, I. Buck, W. Mark and P. Hanrahan. Ray Tracing on Programmable Graphics Hardware. In *Proceedings of SIGGRAPH 2002*, 2002.

[26] C. Rezk-Salama, K. Engel, M. Bauer, G. Greiner, and Ertl. T. Interactive Volume Rendering on Standard PC Graphics Hardware Using Multi-Textures and Multi-State Rasterization. In *SIGGRAPH/EUROGRAPHICS Workshop on Graphics Hardware, 2000.*

[27] C. Rezk-Salama, M. Scheuering, G. Soza, G. Greiner, Fast Volumetric Deformation on General Purpose Hardware. *In Proceedings of SIGGRAPH/EUROGRAPHICS Graphics Hardware Workshop 2001,* pages: 17 – 24, 2001.

[28] F. Scheepers, R. Parent, W. Carlson and S. May, Anatomy-Based Modeling of the Human Musculature, Proceedings of SIGGRAPH 97, August 1997.

[29] L. Serra, N. Hern B. Chua and T. Poston. Interactive Vessel Tracing in Volume Data, *ACM 1997 Symposium on Interactive 3D Graphics, 1997.*

[30] J. Shen and D. Thalmann. Interactive shape design using metaballs and splines. In *Implicit Surfaces '95*, pages 187--193, 1995

[31] D. Silver and N. Gagvani. Unwinding the Colon. *Medicine Meets Virtual Reality (MMVR)*, 2002.

[32] L. Wade and R. Parent, Automated Generation of Control skeletons for use in Animation*, The Visual Computer,* Vol 18, No 2, March 2002.

[33] S. Wang and A. Kaufman. Volume Sculpting. In *Symposium on Interactive 3D Graphics,* ACM Press, April 1995, pp.151-156.

[34] S. Wang and A. Kaufman. A Haptic Interaction Method for Volume Visualization. In *Symposium on Interactive 3D Graphics,* ACM Press, April 1995, pp.151-156.

[35] T. Welsh, M. Ashikhmin and K. Mueller, Transferring Color to Grayscale Images. In *Proceedings of ACM SIGGRAPH 2002, San Antonio*, 2002.

[36] R. Westermann and C. Salama. Real-Time Volume Deformations. In *Computer Graphics Forum (Eurographics)*. 2001.

[37] R. Westermann and T. Ertl. Efficiently using Graphics Hardware in Volume Rendering Applications. In *Computer Graphics (SIGGRAPH 98 Proceedings),* pages 291-294, 1998.

[38] Q. Zhu, Y. Chen and A. Kaufman. Real-time Biomechanically-based Muscle Volume Deformation using FEM. In *Proceedings of Eurographics '98,* 1998.

[39] Vizlab Web Site, Rutgers, The State University of NJ, *http://www.caip.rutgers.edu/vizlab.html*

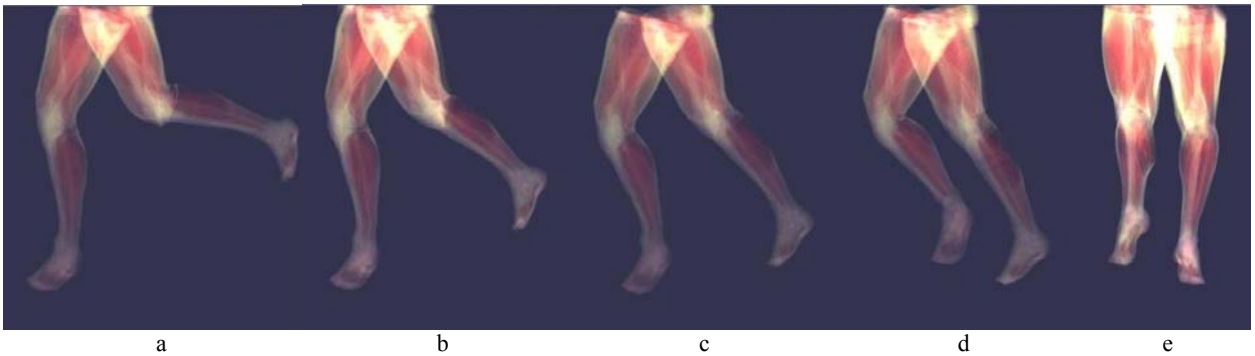
[40] Character Studio Animation Toolkit *http://www.discreet.com/products/cs/*

[41] National Library of Medicine, Visible Human Project, *http://www.nlm.nih.gov/*

[42] Nvidia Corporation, OpenGL Extension Specifications, *http://www.nvidia.com*

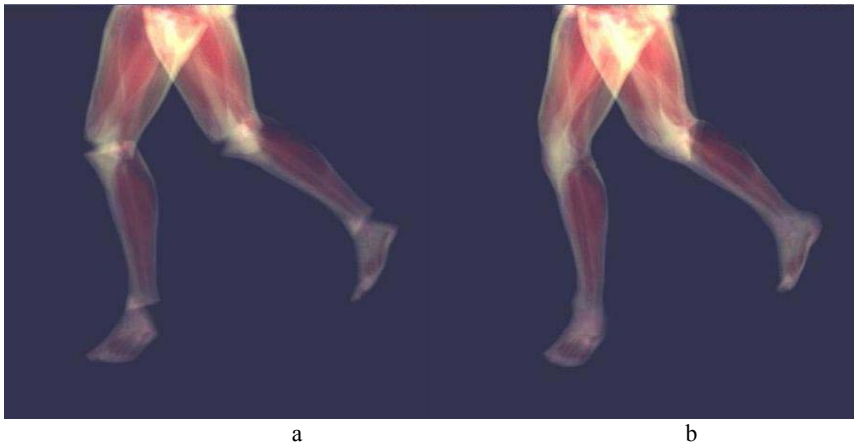
[43] Maya : A 3D Animation and Visual Effects Tool *http://www.aliaswavefront.com/en/products/maya/index.shtml*

[44] SensAble Technologies, Freeform Modeling System, *http://www.sensable.com*



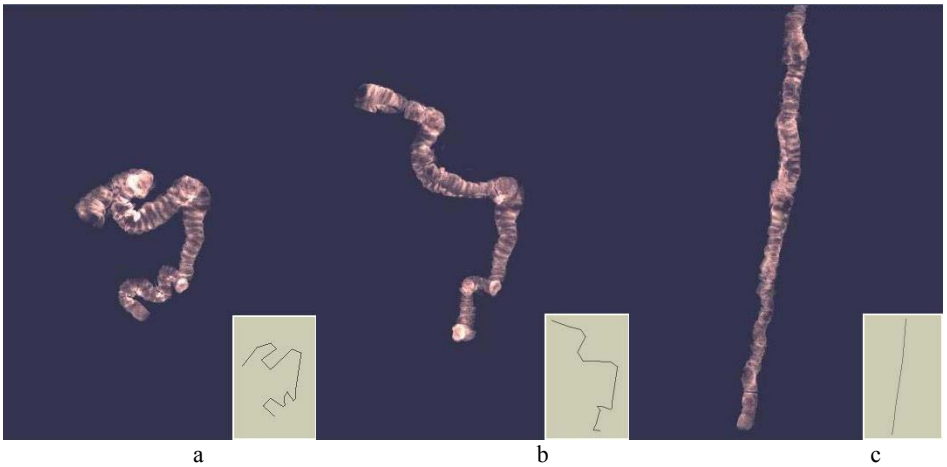
a b c d e

Figure 4. Images from an animation sequence for the legs of the visible human dataset. (a), (b), (c) and (d) Side Views, (e) Front view. The images are rotated as the running progresses. The full movie can be seen at [39].



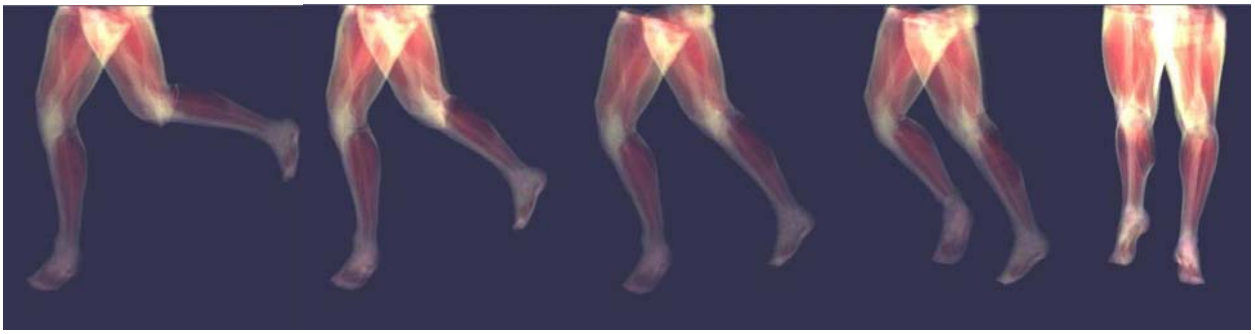
a b

Figure 5. Volume rendered images with different geometry. In (a) breaks occur at the joints because the bounding boxes are not connected. The image in (b) uses mid-plane geometry.



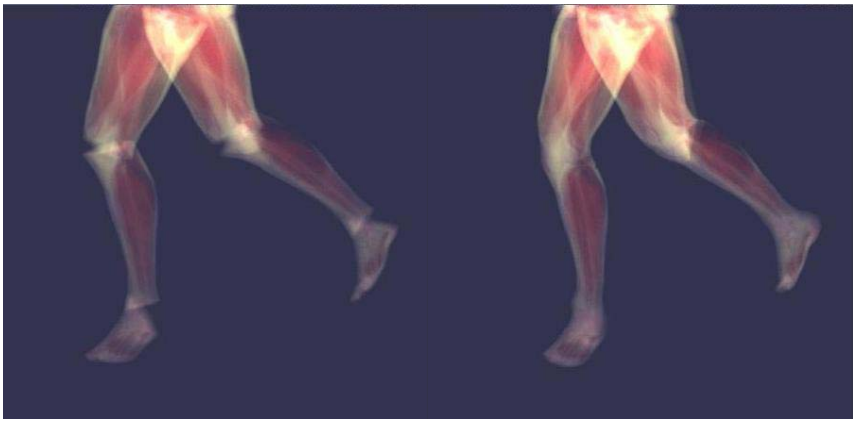
a b c

Figure 6. Unwinding a colon. The dataset is 256x512x256. (a) The original dataset, (b) later reconstructed frame while unwinding, (c) stretched out colon. The corresponding skeletons are shown. This was done by a user interacting with the volume through the VolEdit interface.



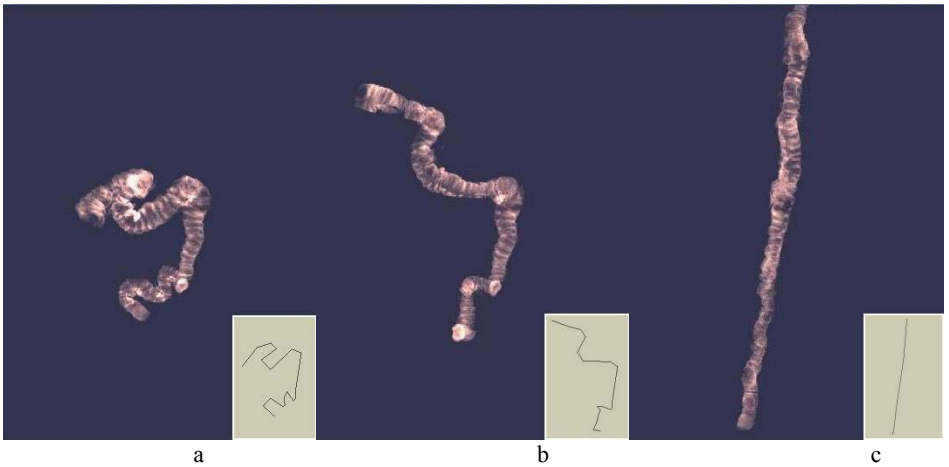
a b c d e

Figure 4. Images from an animation sequence for the legs of the visible human dataset. (a), (b), (c) and (d) Side Views, (e) Front view. The images are rotated as the running progresses. The full movie can be seen at [39].



a b

Figure 5. Volume rendered images with different geometry. In (a) breaks occur at the joints because the bounding boxes are not connected. The image in (b) uses mid-plane geometry.



a b c

Figure 6. Unwinding a colon. The dataset is 256x512x256. (a) The original dataset, (b) later reconstructed frame while unwinding, (c) stretched out colon. The corresponding skeletons are shown. This was done by a user interacting with the volume through the VolEdit interface.