

Texturing Deformable Surfaces of Arbitrary Topology

Christoph Lürig, Thomas Ertl

University of Erlangen, Computer Graphics Group
Am Weichselgarten 9, D-91058 Erlangen, Germany
Email: {cpluerig,ertl}@immd9.informatik.uni-erlangen.de

Abstract

In this paper we present a texturing method for deformable surfaces. In general, these surfaces are triangle meshes of arbitrary topology, therefore, special considerations have to be made for the texture mapping procedure, as a reparametrization along a square definition domain is not always possible or stable. The resulting three dimensional texture problem is solved only by employing two dimensional texture mapping hardware.

The deformable surfaces, which are to be textured, are a three dimensional extension of the snake model. This surface model is used to extract boundary information from three dimensional volumetric data sets. Especially tomographic scans have proven to be adequate for the application of this technique.

The three dimensional function values reconstructed from the data set is then used to generate a texture for these surfaces. As an example, we will demonstrate our technique by visualizing the convolutions of the brain and the convergence behavior of the deformable surface method.

1 Introduction

Snakes and their three dimensional equivalent the deformable surfaces have already been used for various segmentation purposes. This technique, first developed for the two dimensional case by Kass et al. [KWT88], simulates a contour, which is attracted by edges and which tends to minimize curvature and length. Intuitively speaking the attraction to edges is caused by external forces and by the minimization of curvature and length by internal forces.

The three dimensional extension of this concept for image segmentation was invented by Ter-

zopoulos et al. [TWK87]. In this case the model simulates a combination of a rubber skin and a thin plate, that is deformed by external forces. First applications of this concept to the visualization of three dimensional scalar data were done by Sardajoen et al. [SP97]. However, they did not use an internal energy term and they approximated an iso-surface instead of arbitrary volume boundaries. A hierarchical approximation method for deformable surfaces including both kinds of energy terms was introduced by Lürig et al. [LKE98] for visualization purposes.

Applying this technique one encounters the problem of extracting further information on the convergence behavior of this method, in order to get hints where problems may occur. Another task is to visualize tiny details, where the surface is too rigid to approximate the surface of the object to be visualized. This problem is also approached by Ge et. al. [GFD⁺96], who used the snake method on slices to approximate the rough shape of the brain in MRI scans. The intensity values along these lines are then projected into the image plane using an orthogonal or a perspective projection.

In contrast, we are using deformable surfaces with a set of parameters, which induce a rigid surface, that cuts into the convolutions of the brain. The sampled intensity values are then used to generate a texture, which is applied to the extracted surface. This fully three dimensional approach avoids expensive recalculation, when the perspective is changing and it additionally accounts for shading effects. The restriction to two dimensional textures, however, causes problems, since the topology of the generated surface may be inadequate for a parameterization over a square which is necessary for texture mapping. In this paper we present an approach to overcome this problem. If texture filters are applied as described in Heckbert [Hec86], further prob-

lems have to be solved in order to find a sufficient parameterization for this problem.

2 Deformable Surfaces

Let the scalar volume data to be analyzed be described by a function $f : R^3 \rightarrow R$, and the surface to be extracted by $\mathbf{v} : \Omega \subset R^2 \rightarrow R^3$. According to the snakes approach the surface \mathbf{v} has to minimize the following functional:

$$\int_{\Omega} \sum_{i=1}^3 \left(\tau (\nabla \mathbf{v}^{(i)})^2 + (1 - \tau) ((\Delta \mathbf{v}^{(i)})^2 - 2H(\mathbf{v}^{(i)})) \right) + P(\mathbf{v}) dA \rightarrow 0$$

where $\mathbf{v}^{(i)}$ denotes the i -th component of \mathbf{v} , $H(\mathbf{v}^{(i)})$ the determinant of the Hessian Matrix of $\mathbf{v}^{(i)}$ and $P : R^3 \rightarrow R$ the potential field induced by the volume data f . The $\nabla \mathbf{v}$ term is a membrane term that tends to minimize the surface area, thus simulating the behavior of a rubber skin. The term $((\Delta \mathbf{v}^{(i)})^2 - 2H(\mathbf{v}^{(i)}))$ denotes the total curvature and simulates a thin plate. The coefficient τ is a balancing factor, which controls the influence of the rubber skin and the thin plate aspect. Both terms stabilize the optimization process. The external energy term P is defined as:

$$P(\mathbf{v}) = -(w_{edge} \|\nabla(G_{\sigma} * f(\mathbf{v}))\| + w_{image} f(\mathbf{v})),$$

where G_{σ} is a Gaussian kernel with variance σ . The kernel is used to generate a smooth potential field from the data, which will improve the convergence of the iterative solver. This makes the approximation of the gradient by finite differences more reliable and enlarges the regions of attraction near sharp boundaries in the volume data set. The second part of the potential field term may be used to detect regions with high intensity values. The coefficient w_{edge} weights the impact of the boundary influence and the coefficient w_{image} describes the direct influence of the intensity value of the data set.

This analytical formulation of the problem is approximated by a multi-level finite difference solver (details are given in [LKE98]). The technique has shown to be especially useful for visualizing tomographic data sets containing distinct objects.

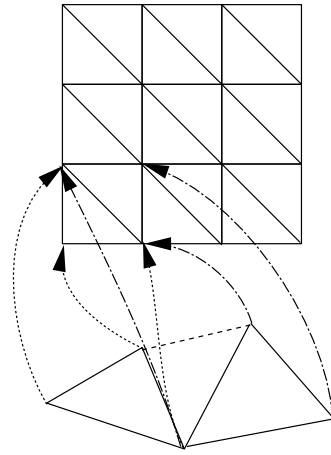


Figure 1: Correspondence of geometric triangles and texture triangles

3 Texturing

In order to provide more information in the visualization process than it is given by the surface itself, we would like to generate a texture that represents the value of f in the neighborhood of the surface. In fact, what we do is to simulate a three dimensional texture with one exception: we would like to have the same amount of texels for each triangle, independent of its size. This way we make use of the adaptivity information of the underlying grid, as the triangles have been refined adaptively.

The generation of the texture for a given surface is done by tessellating the definition domain of the texture into triangles as shown in Fig. 1. Each geometric triangle corresponds to one of the texture triangles. As the triangles which are adjacent in the geometric representation are not necessarily adjacent in the texture, every geometric vertex might have different texture coordinates for each triangle it belongs to. In order to keep texture mapping stable later on, the tessellation of the texture domain should be approximately the same size in both dimensions.

The triangles are scan-converted in texture space. During scan conversion the u, v -coordinates of the positions are computed. These local coordinates are then transformed to global coordinates in the three dimensional volume using the corresponding geometric triangle. This is illustrated in Fig. 2. The function value is sampled by trilinear interpolation and the texture color is computed based on a transfer function.

First experiments have shown, that under cer-

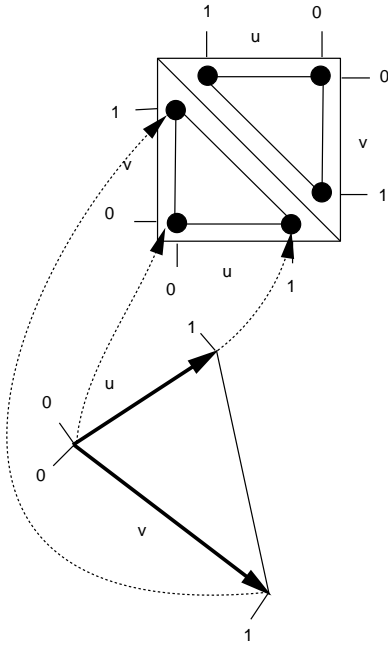


Figure 2: Mapping a triangle to texture space

tain circumstances texture artifacts may occur at the triangle edges. This phenomenon is caused by special texture filtering algorithms or MIP mapping techniques. We deal with this problem by under-scanning the triangles and by using u,v -coordinates, which are smaller than zero and larger than one. In this way the edges of triangles adjacent in geometric space are colored correctly if these triangles lie in the same plane. This technique is also illustrated in Fig. 2. Increasing the padding space around the triangles suppresses artifacts even if more complicated filters are applied, but it also increases texture memory consumption for the same image quality.

4 Results

We have applied the presented texture mapping process to verify the convergence behavior of the deformable surface approach and to provide additional information about the data set.

For the verification of convergence behavior, we extracted a deformable surface from a data set, which contains a cubic area in the center, with function values set to one inside and to zero outside. An initial surface has been layed around the cube as a starting solution for the iteration process. The mesh resulting from the shrinking process is shown in Fig. 3(d). The adaptive refinement at the edges of the box can clearly be

seen. Triangles are refined, if they are influenced by a strong external force and an equally strong internal force, which nearly compensates it. The texture mapped cube in Fig. 3(b) shows that the resulting surface cuts through the edges of the original volume cube, and that it converges from the outside of the cube towards its surface going from an edge towards the center of a cube face. This phenomenon is caused by the internal forces, that tend to minimize the surface curvature. They are pushing the surface into the cube at its edges and are causing the adaptive refinement in these regions.

As a second example we have applied the deformable surface method to extract the brain surface in a T1-weighted MRI scan of a head. A result where the convolutions of the brain surface are approximated is shown in Fig. 3(a). Texturing this surface did not reveal much extra information. However, if the surface is constructed with stronger inner forces, then it does not approximate each convolution, but more or less cuts them. In this case, the texturing of the surface reveals extra detailed information about the brain surface as shown in Fig. 3(c). The idea is to provide a detailed visual impression, even if the geometric complexity of the surface is relatively low.

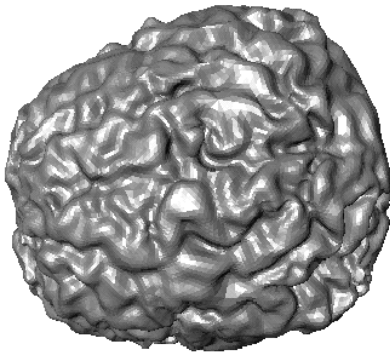
5 Conclusions and future work

In this paper we have demonstrated a new method for visualizing additional information on deformable surfaces by means of texture mapping. This approach can be used to monitor the convergence behavior as well as displaying additional features of the volumetric data set. Texturing allows to reduce the geometric complexity of the surface without completely losing detail information, which greatly increases rendering speed on displays with 2D texture mapping hardware.

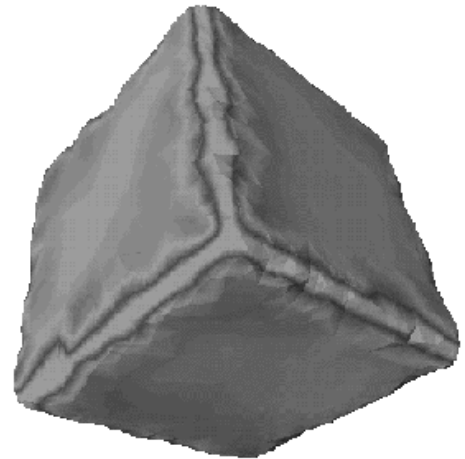
As future work we plan to include morphological operations to preprocess the data set. This could possibly improve the convergence behavior of the deformable surface method.

References

- [GFD⁺96] Yarong Ge, Michael J. Fitzpatrick, Benoit M.



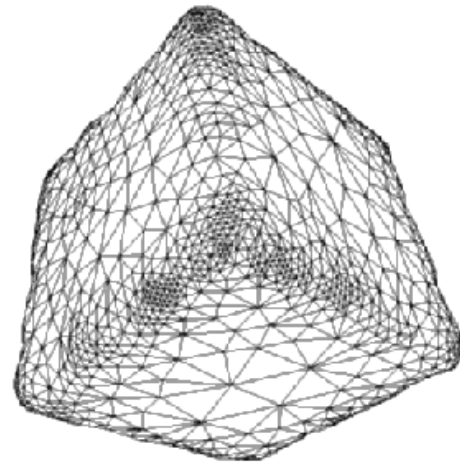
(a) Original Brain



(b) Textured Cube



(c) Flat Brain with Texture



(d) Cube with hidden lines

Figure 3: Applications of textured deformable surfaces

- Dawant, Jun Bao, Robert M. Kessler, and Richard A. Margolin. Accurate Localization of Cortical Convolution in MR Brain Images. *IEEE Transactions on Medical Imaging*, 15(4):418–428, 1996.
- [Hec86] Paul S. Heckbert. Survey of texture mapping. *IEEE Computer Graphics and Applications*, pages 321–332, 1986.
- [KWT88] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active Contour Models. *International Journal of Computer Vision*, pages 321–331, 1988.
- [LKE98] Christoph Lürig, Leif Kobbelt, and Thomas Ertl. Deformable Surfaces for Feature Based Indirect Volume Rendering. In *Proceedings CGI'98 (submitted)*, 1998.
- [SP97] I. A. Sardajoen and F. H. Post. Deformable Surface Techniques for Field Visualization. In *Eurographics '97*, pages 109–116, 1997.
- [TWK87] D. Terzopoulos, A. Witkin, and M. Kass. Symmetry-Seeking Models and 3D Object Construction. *International Journal of computer Vision*, 1:211–221, 1987.