Tensor Field Design in Volumes

Jonathan Palacios* Chongyang Ma‡ Weikai Chen† Li-Yi Wei† Eugene Zhang*
Oregon State University* Tsinghua University‡ University of Hong Kong†

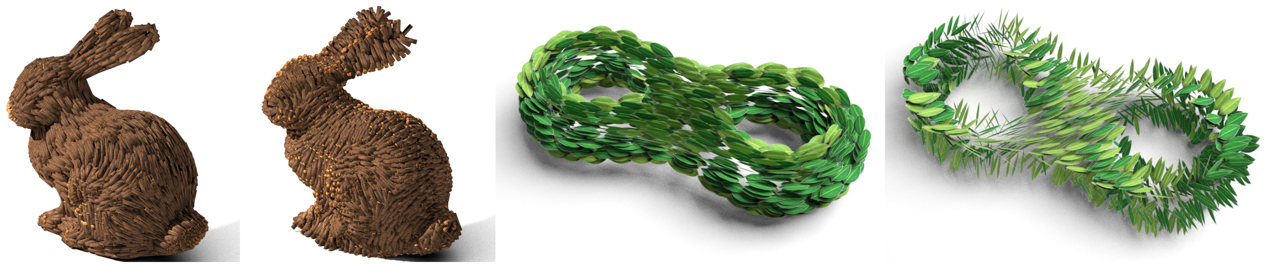


Figure 1: *Different tensor fields can lead to different visual effects when performing geometry synthesis on the same model. While the left wooden bunny looks natural, the right bunny's ears seem twisted. Similarly, while the left leafy torus appears orderly, the right torus provides a sense of restlessness. The fields guiding the placements of wooden sticks and leaves were generated using our tensor field design system.*

Abstract

The design of 3D tensor fields is important in several graphics applications such as procedural noise, solid texturing, and geometry synthesis. Different fields can lead to different visual effects. The topology of a tensor field, such as degenerate tensors, can cause artifacts in these applications. Existing 2D tensor field design systems cannot handle the topology of 3D tensor fields. We present, to our best knowledge, the first 3D tensor field design system. At the core of our system is the ability to specify and control the type, number, location, shape, and connectivity of degenerate tensors. To enable such capability, we have made a number of observations of tensor field topology that were previously unreported. We demonstrate applications of our method in volumetric synthesis of solid and geometry texture as well as anisotropic Gabor noise.

Keywords: solid texture synthesis, 3D tensor fields, tensor field design, topology, texture synthesis, geometry synthesis, element synthesis

Concepts: •**Computing methodologies** → **Shape modeling;**

1 Introduction

Many applications in computer graphics and geometry processing rely on the ability to locally orient 3D entities inside a volume, such as the sinusoidal kernels in Gabor noise [Lagae et al. 2009; Lagae and Drettakis 2011] as well as color and geometry textures in pattern synthesis [Kopf et al. 2007; Takayama et al. 2008; Wei et al. 2009; Ijiri et al. 2008; Ma et al. 2011]. In these applications, the orientations of the 3D entities can be controlled by a continuous, symmetric tensor field, whose eigenvectors provide an *orthonormal* frame everywhere in the domain and whose eigenvalues describe the anisotropy in the 3D entities. Zhang et al. [2011] consider the generation of 3D tensor fields for solid texturing.

Tensor field design can help create different artistic effects by using different tensor fields. Figure 1 illustrates this with element synthesis on the bunny and the double torus, each with two fields. While the first fields for both models (left) give a rather natural appearance of the models, the second fields (right) provide a rather different, unorthodox flavor.

A major challenge facing tensor field design is the existence of *degenerate points*, where the tensor field has repeating eigenvalues and the orthonormal frames have discontinuity. Figure 2 (left) shows the result of geometry synthesis guided by a tensor field with degenerate points (around the center of the image, where the orientations of the slab glyphs change rapidly). Such visual artifacts can be reduced by using a different field without degenerate points (Figure 2 (right)). One of the core capabilities of our system is to specify and control the topology of 3D tensor fields, i.e., degenerate points. Note that degenerate points in 3D fields form curves [Zheng and Pang 2004], and are thus referred to as *degenerate curves* which constitute *tensor field topology*.

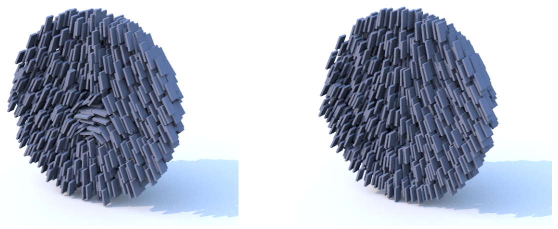


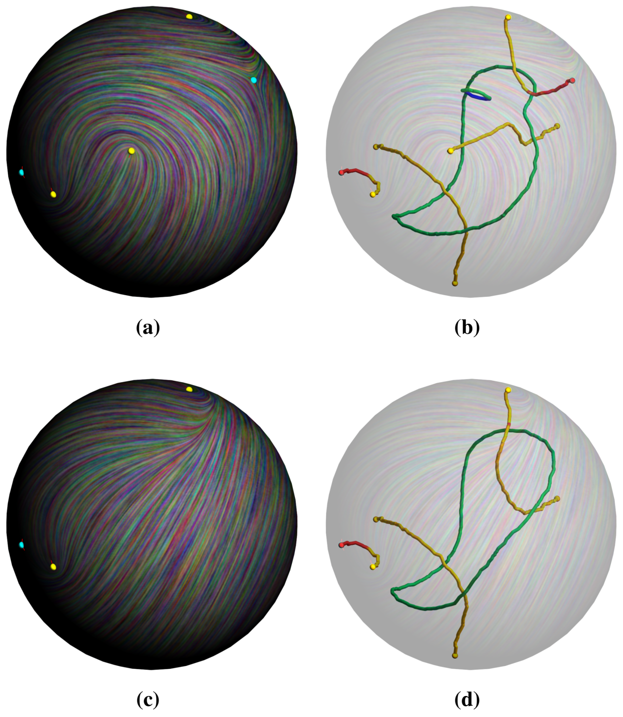
Figure 2: *Degenerate points in the tensor field can lead to visual artifacts such as the unnaturally rapid change in slab orientations (left). Such artifacts can be alleviated when using a field without degenerate points (right).*

While it is conceivable to control the topology of a volume tensor field by controlling its behavior on the boundary surface, this strategy can lead to undesired effects due to the following reason. The topology of the surface tensor field does not provide a complete picture of the topology of the corresponding 3D tensor field. As an example, the 2D tensor field in Figure 3a contains eight degenerate points on the boundary surface (yellow and cyan points). However, as a 3D tensor field in Figure 3b there are two degenerate loops completely in the interior of the volume and thus not visible from the surface field. When simplifying the topology of the 3D tensor

‡Chongyang Ma is currently at Activision.

SA '16 Technical Briefs ,, December 05-08, 2016, , Macao
ISBN: 978-1-4503-4541-5/16/12 $15.00
DOI: http://dx.doi.org/10.1145/3005358.3005369

field by reusing 2D tensor field topological editing [Zhang et al. 2007], unpredicted changes may occur. In Figure 3c, two degenerate points in the surface are cancelled, which not only leads to the loss of an internal degenerate loop and shape change for the other loop but also the reconnection of two previously unrelated degenerate curves (Figure 3d). Identifying possible, fundamental operations to modify the topology of 3D tensor fields, i.e., only targeted degenerate curves are impacted, is a challenging yet important problem that any 3D tensor field design system faces.



(a) **(b)** **(c)** **(d)**

Figure 3: *The topology of a surface tensor field does not provide a complete picture of the corresponding volume tensor field. Notice that it is difficult to predict which surface degenerate points belong to the same degenerate curve. Also, there can be degenerate loops which do not intersect the boundary surface. In addition, topological editing operations applied on the boundary tensor field can lead to unpredicted behaviors in the topology of the corresponding volume tensor field. For example, when removing a singularity pair in the surface tensor field (from (a) to (c)), two degenerate curves in the volume tensor field are joined, one of the degenerate loops is eliminated, and the shape of the other degenerate loop is altered (compare (b) to (d)).*

Here, we present to our knowledge the first interactive 3D tensor field design system that provides control over the topology of the tensor fields. The user can create a tensor field by specifying desired tensor values and local patterns inside the volume, or on the boundary surface, or in both places. The field can be made boundary-conforming, i.e., the surface normal direction is aligned with one of the eigenvectors of the tensor field everywhere on the surface. At the core of our system is the ability to *locally* edit the topology of a tensor field. We identify topological editing operations that are designed to impact only the intended degenerate curves, such as removing one or two degenerate curves from the field, switching the connection between two degenerate curves, and deforming a degenerate curve. Other degenerate curves are left unaltered. We also provide efficient algorithms to realize these operations. We demonstrate the efficacy of our system by applying it to the control of anisotropic Gabor noise, geometry element synthesis, and solid texturing.

Due to the page limit, we refer to the readers to the supplementary materials for the mathematical background on 3D tensor fields, including tensor field topology.

2 3D Tensor Field Specification

Our 3D tensor field design system consists of two steps. First, the user generates a tensor field by providing constraints inside the volume and/or on the boundary surface. Second, the user edits individual degenerate curves of the tensor field.

The computational setup of our system is as follows. The input domain is a tetrahedral mesh $M \subset \mathbb{R}^3$. A 3D tensor field is represented as a set of tensor values, one per each vertex of M. Linear interpolation is used to extend the tensor values from a set of points (vertices of M) to a continuous tensor field over M. When performing tensor field design and editing, we modify the tensor field by modifying its values on the vertices of M. After each modification to the tensor field, the degenerate curves are extracted based on the method of Zheng and Pang [2004]. The **LP**-type as well as wedge/trisector classification (please see the supplementary material) along degenerate curves are also performed and color-coded in order to facilitate the user during design. As shown in Figure 3 (right), an **L**-type wedge degenerate point is colored in green, while an **L**-type trisector point is colored in blue. A **P**-type degenerate point is colored in yellow if a wedge, or red if a trisector.

Our specification system is based on a number of requirements. First, it is important to be able to specify the tensor value at a given point in space. Second, if the desired tensor value is degenerate, it is usually important to also specify the degenerate tensor pattern near the point of interest. Third, we would like the interface to be both intuitive and easy-to-use for the user. Given these requirements, our specification system consists of the following steps. First, the user places constraints using our design system, each of which is a desired tensor value at a given vertex in the mesh. Second, the constraints are used as the boundary condition of a tensor-valued Laplace system of equations. This constrained optimization method has been successfully used in 2D tensor field processing [Zhang et al. 2007]. Next, we focus on the details of the first step, i.e., generating constraints based on user specification.

A constraint can be either degenerate or non-degenerate. In the latter, it is in the form of a non-degenerate tensor T_0. A degenerate constraint is in the form of $T_0 + (x - x_0)T_x + (y - y_0)T_y + (z - z_0)T_z$. Here (x_0, y_0, z_0) is the 3D coordinates of the vertex, T_0 is a degenerate tensor, while T_x, T_y, and T_z are tensors that may be degenerate or non-degenerate. Together, the coefficients of T_x, T_y, and T_z form the *Jacobian* of the tensor field and are responsible for the local tensor degenerate patterns (wedges, trisectors) around the point where the constraint is placed.

We have found that it is easier and often more intuitive to specify not a single constraint, but a set of constraints along a curve. This is especially true when specifying degenerate tensor constraints, which naturally form curves. Given a user-specified curve, our system first generates a spline that best captures the sketched curve. The spline curve is then subsampled at a set of evenly spaced points on the curve including both end points. At each sample point p, we compute a frame based on the method of Bergou et al. [2008]. The first vector in the frame, i.e., the curve tangent, is assumed to be a non-degenerate eigenvector (please see the supplementary material for the definition).

For a non-degenerate tensor constraint, the other two vectors in the frame are assumed to be the remaining eigenvectors. The user can change the eigenvectors by freely rotating the frame in 3D. The default maximal eigenvalue is 1 if **L**, and the default minimal eigenvalue is -1 for **P** type of constraints. The user also provides the desired mode (see supplementary material) of the tensors along the curve through the interface. The desired tensor mode combined with the given eigenvalue can uniquely determine the other two eigenvalues.

For a degenerate tensor constraint, the other two vectors in the frame give the degenerate plane. Again, the user can change the non-degenerate eigenvector and the degenerate plane by rotating the frame. The default eigenvalues are $2, -1, -1$ for an **L** constraint and $1, 1, -2$ for a **P** constraint. The default values for the Jacobian are $T_x = \begin{pmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 0 \end{pmatrix}$, $T_y = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$ and $T_z = 0$ for a wedge constraint. A default Jacobian for a trisector constraint simply negates T_y. The user can change the 2D tensor pattern inside the degenerate plane by changing the 2×2 sub-blocks in T_x and T_y. The user can also rotate the non-degenerate eigenvector direction freely in 3D, which will lead to the rotation of the 3D frame but the Jacobian coefficients do not change with respect to the new local coordinate system. This can lead to the change in the angle between the non-degenerate eigenvector and the tangent of the degenerate curve.

3 Topological Editing Operations

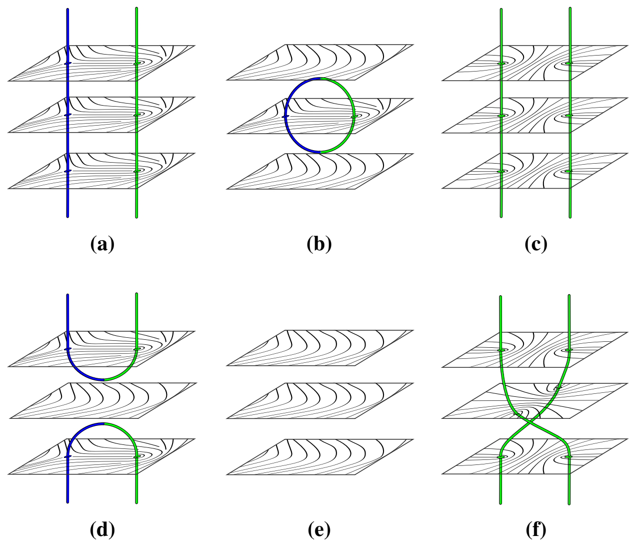


Figure 4: *Degenerate curve removal and reconnection can be seen as performing simultaneous bifurcations on a family of 2D tensor fields: a wedge and trisector curve pair reconnection (left), a mixed loop removal (middle), and two wedge curve reconnection (right). Note that (a) to (e) corresponds to wedge and trisector curve removal.*

The tensor fields generated in the first step often contain degenerate curves in addition to the ones specified by the user. As mentioned earlier, excessive and misbehaving degenerate curves can lead to visual artifacts in the applications. Here, we introduce a number of *fundamental* topological editing operations that we have identified as part of this research. These operations are designed to impact the least number of degenerate curves, while together can provide enough flexibility to modify tensor field topology.

The first topological editing operation is *degenerate curve removal*, which refers to removing one or two degenerate curves. In the former, a degenerate curve with both wedge and trisector segments is eliminated (Figure 4b to Figure 4e). Notice the difference in the tensor field along a number of cutting planes before (Figure 4b) and after (Figure 4e) the one degenerate curve removal operation. The two degenerate curve removal operation is illustrated also in Figure 4 ((a) to (e)). In this case two degenerate curves (one wedge and one trisector) are cancelled. The degenerate curve removal operations are designed to reduce the number of degenerate curves in the field.

The second topological editing operation is *degenerate curve reconnection*, which refers to cutting open two degenerate curve segments and stitching together pieces from different segments together, thus resulting in two new degenerate curve segments. There are three fundamental scenarios. First, one wedge curve and one trisector curve are reconnected (Figure 4a to Figure 4d). The second scenario is the inverse of first scenario, i.e., two mixed curves (with both wedge and trisector segments) are reconnected so that two pure degenerate curves result (one wedge and one trisector). The last scenario refers to reconnecting two pure degenerate curves with the same type (Figure 4c to Figure 4f). In this case two wedge curves are reconnected, resulting in two wedge curves. The degenerate curve reconnection operation does not change the number of degenerate curves. Rather, it changes how these curves connect. This operation can help untangle linked degenerate curves and thus facilitate subsequential topological editing operations such as removal. In addition, when mixed degenerate curves are undesirable, this operation can help reduce the number of such curves.

The last topological editing operation is *degenerate curve deformation*. In this case, part of a degenerate curve is deformed. This operation does not reduce the number of the degenerate curves or change their connectivity. Instead, it can control the shape (and location) of degenerate curves in the field.

In our system, the three topological editing operations are implemented under a unified framework. This framework consists of two steps. First, a region enclosing only the targeted degenerate curves (or segments of them) is constructed. Second, the field inside the region is modified to strive for the desired effects (removal, reconnection, or deformation). The first step is achieved by finding all the tetrahedra containing the desired degenerate curves (or segments on them) and iteratively adding more tets until the region is a topological ball (or a topological torus when removing two degenerate loops). Tets containing other degenerate curves are not added, thus ensuring that the topological editing operations only modify targeted degenerate curves.

The second step, i.e., field modification inside the region, depends on the type of the operation. For degenerate curve deformation, we construct a re-parameterization of the region (self-homeomorphism) such that the homeomorphism is the identity map on the region's boundary and maps the original curve segment to its deformed shape. The self-homeomorphism naturally leads to a modified tensor field with the desired shape of the targetted degenerate curve. For degenerate curve reconnection and removal, we perform tensor-valued Laplacian smoothing inside the volume with the tensor values fixed at the region's boundary and some interior vertices.

Figure 5 shows a sequence of topological editing operations performed on a tensor field on the bunny.

4 Applications

We demonstrate applications of our method in anisotropic Gabor noise (Figure 6d), solid texturing (Figures 6a to 6c), and geometry element placement (Figures 1, 2 and 6e). The fields used in all these examples are created with our tensor field design system.

References

BERGOU, M., WARDETZKY, M., ROBINSON, S., AUDOLY, B.,

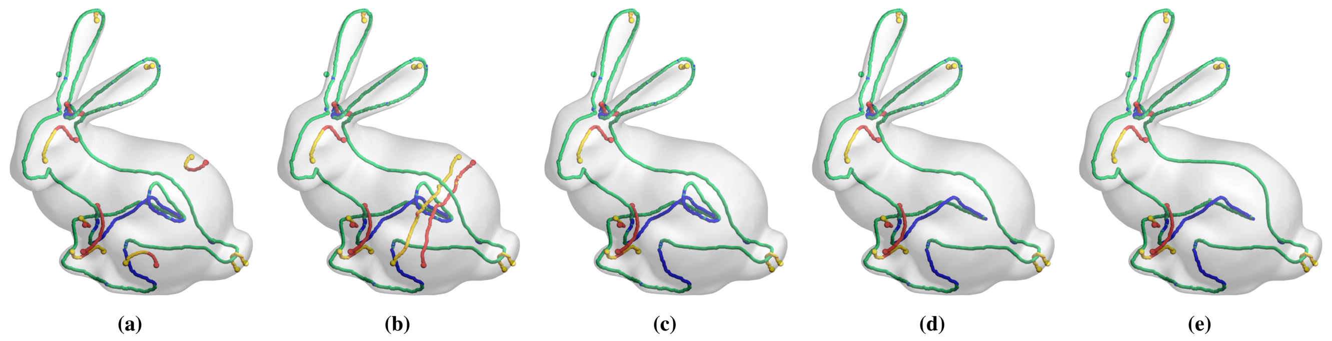


Figure 5: *A sequence of topological editing operations are applied to an input tensor field on the bunny. From left to right are: (a) the input tensor field, (b) two degenerate curves are reconnected, resulting in one pure wedge curve (yellow) and one pure trisector curve (red), (c) the two degenerate curves are removed simultaneously, (d) a mixed degenerate loop is removed (the green/blue loop), and (e) a section of the degenerate curve along the bunny's back (green) is deformed. Notice that our editing operations is able to perform degenerate curve reconnection even when they are relatively far away and there are additional degenerate curves in the middle.*

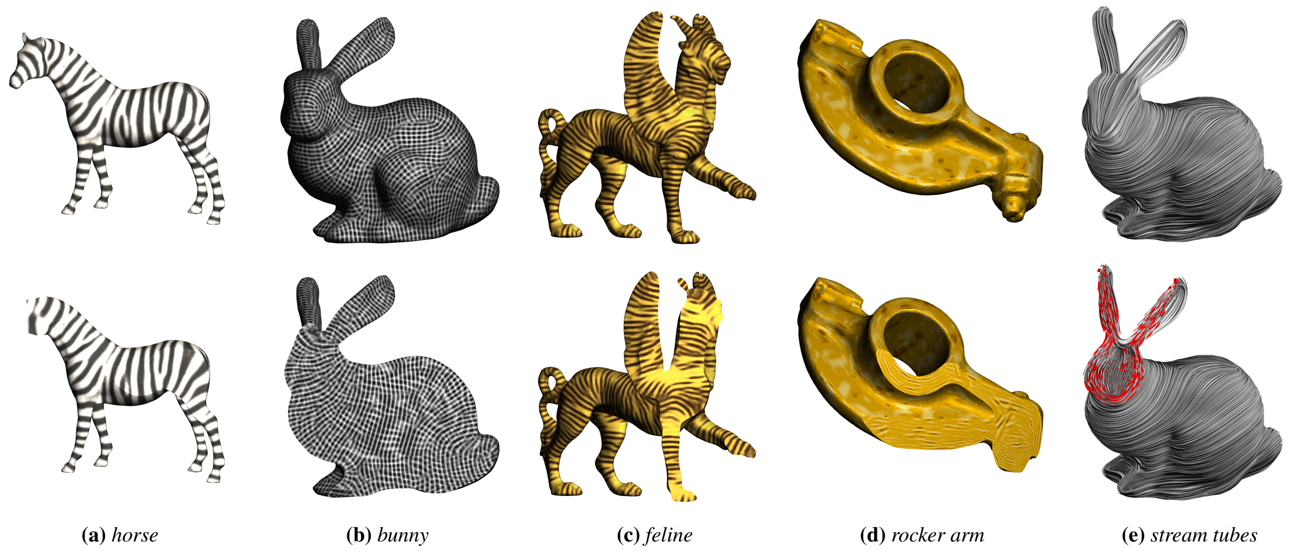


Figure 6: *Solid texture synthesis based on examples ((a), (b), (c)) and procedures ((d), (e)). The outputs are rendered through different cross sections to illustrate their volumetric nature.*

AND GRINSPUN, E. 2008. Discrete Elastic Rods. In *SIGGRAPH '08*, 1–12.

ENNIS, D. B., AND KINDLMANN, G. 2006. Orthogonal tensor invariants and the analysis of diffusion tensor magnetic resonance images. *Magnetic Resonance in Medicine 55*, 1, 136–146.

IJIRI, T., MECH, R., IGARASHI, T., AND MILLER, G. 2008. An example-based procedural system for element arrangement. *EG '08 27*, 2, 429–436.

KOPF, J., FU, C.-W., COHEN-OR, D., DEUSSEN, O., LISCHINSKI, D., AND WONG, T.-T. 2007. Solid texture synthesis from 2d exemplars. In *SIGGRAPH '07*.

LAGAE, A., AND DRETTAKIS, G. 2011. Filtering solid gabor noise. In *SIGGRAPH '11*, 51:1–6.

LAGAE, A., LEFEBVRE, S., DRETTAKIS, G., AND DUTRÉ, P. 2009. Procedural noise using sparse gabor convolution. In *SIGGRAPH '09*, 54:1–10.

MA, C., WEI, L.-Y., AND TONG, X. 2011. Discrete element textures. In *SIGGRAPH '11*, 62:1–10.

TAKAYAMA, K., OKABE, M., IJIRI, T., AND IGARASHI, T. 2008. Lapped solid textures: filling a model with anisotropic textures. In *SIGGRAPH '08*, 1–9.

WEI, L.-Y., LEFEBVRE, S., KWATRA, V., AND TURK, G. 2009. State of the art in example-based texture synthesis. In *EuroGraphics STAR*, 93–117.

ZHANG, E., HAYS, J., AND TURK, G. 2007. Interactive tensor field design and visualization on surfaces. *IEEE Transactions on Visualization and Computer Graphics 13*, 1, 94–107.

ZHANG, G.-X., DU, S.-P., LAI, Y.-K., NI, T., AND HU, S.-M. 2011. Sketch guided solid texturing. *Graphical Models 73*, 3, 59–73.

ZHENG, X., AND PANG, A. 2004. Topological lines in 3D tensor fields. In *Proceedings IEEE Visualization '04*, 313–320.

ZHENG, X., PARLETT, B., AND PANG, A. 2005. Topological structures of 3D tensor fields. In *Proceedings IEEE Visualization 2005*, 551–558.