Tensor-fields Visualization using a Fabric like Texture on Arbitrary two-dimensional Surfaces

Ingrid Hotz[1], Louis Feng[1], Bernd Hamann[1], and Kenneth Joy[1]

Institute for Data Analysis and Visualization (IDAV), Department of Computer Science, University of California, Davis, CA 95616, USA;
{ihotz,zfeng,bhamann,kijoy}@ucdavis.edu

Summary. We present a visualization method that for three-dimensional tensor fields based on the idea of a stretched or compressed piece of fabric used as a "texture" for a two-dimensional surfaces. The texture parameters as the fabric density reflect the physical properties of the tensor field. This method is especially appropriate for the visualization of stress and strain tensor fields that play an important role in many application areas including mechanics and solid state physics. To allow an investigation of a three-dimensional field we use a scalar field that defines a one-parameter family of iso-surfaces controlled by their iso-value. This scalar-field can be a "connected" scalar field, for example, pressure or an additional scalar field representing some symmetry or inherent structure of the dataset. Texture generation consists basically of three steps. The first is the transformation of the tensor field into a positive definite metric. The second step is the generation of an input for the final texture generation using line integral convolution (LIC). This input image consists of "bubbles" whose shape and density are controlled by the eigenvalues of the tensor field. This spot image incorporates the entire information content defined by the three eigenvalue fields. Convolving this input texture in direction of the eigenvector fields provides a continuous representation. This method supports an intuitive distinction between positive and negative eigenvalues and supports the additional visualization of a connected scalar field.

1 Introduction

Tensor data play an important role in several mathematical, physical, and technical disciplines. Mathematically, a tensor is a linear function that relates different vectorial quantities. Its high dimensionality makes it very complex and difficult to understand. Since the physical interpretation and significance of its mathematical features is highly application-specific, we focus on symmetric tensor fields of second order that are similar to stress, strain tensor fields. Such fields appear, for example, in geomechanics and solid state physics, which are our major application areas. Here tensors are used, for example, to express the response of a material to applied forces. In contrast to other types

of tensors, like diffusion tensors, these tensor fields are characterized by the property that they have positive and negative eigenvalues. The sign of the eigenvalues indicates regions of expansion and compression, and it is therefore of special interest. To understand field behavior it is important to express this behavior in an intuitive way.

We extend a method interpreting these tensor fields as distortions of a flat metric [9]. A deformation of a fabric-like texture leads to a continuous representation of the main features of the tensor field, regions of compression and tension. Due to occlusion this method is basically restricted to two-dimensional slices of higher-dimensional fields. We now use a similar approach to investigate three-dimensional datasets, visualizing the tensor field on a family of arbitrary two-dimensional surfaces. Defining these surfaces as a one-parameter family of iso-surfaces it is possible the possibility to represent an additional related scalar field, e.g., pressure. Another way to define the surfaces might be the geometry of the problem. The texture on the surfaces is generated by blurring a three-dimensional input texture along the tensor lines of the tensor field, projected onto the surfaces. The result is a fabric-like texture that is dense in regions of compression and sparse in regions of expansion. The input texture consists of three-dimensional "spots", whose size and density reflects the eigenvalues of the tensor field. It is precomputed using a reaction-diffusion method.

2 Related Work

Even though several visualization techniques exist for tensor fields, they only cover a few specific applications. Many of these methods are extensions of vector field visualization methods, which focus on a technical generalization without providing an intuitive physical interpretation of the resulting images. They often concentrate on the representation of eigendirections and neglect the importance of the eigenvalues. Therefore, in many application areas traditional two-dimensional plots are still being used, which represent the interaction of two scalar variables only.

A basic way to represent a tensor field is "icons". They illustrate the characteristics of a field at selected points. One simple example icon that represents a symmetric tensor is the ellipsoid. The principal axes of the ellipsoid are aligned to the eigendirections, scaled according to the corresponding eigenvalues. (See, for example, Kriz et al. [12] or Haber [12].) Ellipsoid-based methods are very common for medical applications to visualize results of diffusion magnetic resonance imaging (MRI). More complex glyphs were constructed by Leeuw et al. [13] showing additional features using flow probes. An improvement of these icon methods using a reaction-diffusion simulation, introduced by Kindlmann et al. [11], generates a pattern that is closely related to ellipsoids. There, the packing of the texture spots are generated automatically by the simulation.

A more advanced but still discrete approach uses hyperstreamlines. This approach is strongly related to streamline methods used for vector field visualization. Hyperstreamlines were introduced by Delmarcelle and Hesselink [3] and were adapted in a geomechanical context by Jeremic et al. [10]. Given a point in the field, one eigenvector field is used to generate a vector field streamline. The other two eigendirections and eigenvalues are represented by the cross section along the streamline. This method extracts more information than icons, but still leaves the problem of choosing appropriate seed points to the user. Thus, both methods have limited value for the exploration of complete data sets and are limited to low-resolution due to cluttering.

An adaptation of hyperstreamlines to diffusion tensors of MRI data was used by Zhukov and Barr [27] with the goal of tracing anatomical fibers. Their method is based on the assumption that there exists one large and two small eigenvalues inside the fibers, and fiber direction corresponds to the dominant eigenvector. An approach that arose in a similar context is an adaptation of direct volume rendering to diffusion tensor fields presented by Kindlmann et al. [21]. After a classification of a field with respect to anisotropy, it is divided into three categories: linear, planar and spherical. This property is then used to define barycentric coordinates of a transfer function over a triangular domain that highlights regions of different anisotropic properties. Approaches like this one are specially designed for the visualization of diffusion tensors that only have positive eigenvalues and thus are not appropriate for stress, strain or gradient tensor fields.

To generate a more global view, a widely accepted solution for vector fields is the reduction of the field to its topological structure. These methods generate topologically similar regions that lead to a natural separation of a field. The concept of topological segmentation was also applied to two-dimensional tensor fields [5, 6, 18] and has recently been extended to three dimensional tensor fields [26]. The topological skeleton consists of field singularities and connecting separatrices. For tensor fields the vector field singularities are replaced by degenerate points, which are points where the tensor has multiple eigenvalues. Although the tensor field can be reconstructed on the basis of topological structure, physical interpretation is difficult. One reason is the fact that high multiplicity of eigenvalues has no obvious physical significance.

Following an approach of Pang et al. [1, 23], a tensor field can be considered as a force field that deforms an object placed inside it. The local deformation of probes, such as planes and spheres, illustrate the tensor field. Zheng et al. [24] extended this method by applying it to light rays that are bended by the local tensor value.

Another class of visualization methods provides a continuous representation, based on textures. The first one to use a texture to visualize a tensor field in a medical context were Ou and Hsu [15]. A method similar to LIC adapted to tensor fields was proposed by Pang et al. [25]. Here, a white-noise texture is blurred according to the tensor field. In contrast to LIC images, the convolution filter is a two-dimensional area determined by the local tensor

field. This visualization is especially powerful for showing the anisotropy of a tensor field with only positive eigenvalues.

A geometrical approach was followed by Hotz et al. [7]. This approach uses a metric interpretation of a tensor field to emphasize the physical meaning of tensors behaving similarly to stress, strain or gradient tensors. For two-dimensional fields an isometric embedding is used to visualize the resulting metric locally [8]. Using a deformation of a fabric-like texture makes possible a global representation of the metric [9].

3 Mathematical Background and Notation

A tensor is a geometrical entity that generalizes the concept of scalars, vectors and linear operators in a way that is independent of any chosen coordinate system. It is the mathematical idealization of a geometric or physical quantity that expresses a linearized relation between multidimensional variables. For example, the stress, strain or elasticity tensors express the response of a material to an applied force.

The tensors we are interested in are tensors of second order defined over three-dimensional Cartesian space. Using a fixed coordinate basis, each tensor can be expressed by a 3×3 matrix, given by nine independent scalars: $\mathbf{T} = (t_{ij})$. A tensor $\mathbf{T}$ is called symmetric if $t_{ij} = t_{ji}$ for $i, j = 1, .., n$. It is called antisymmetric if $t_{ij} = -t_{ji}$ for $i, j = 1, .., n$. Every tensor can be decomposed in a symmetric part $\mathbf{S}$ and an antisymmetric part $\mathbf{A}$: $\mathbf{T} = \mathbf{S} + \mathbf{A}$, where $s_{ij} = \frac{1}{2}(t_{ij} + t_{ji})$ and $a_{ij} = \frac{1}{2}(t_{ij} - t_{ji})$. In many applications, the tensor fields are already symmetric by definition. We restrict ourselves here to symmetrical tensor fields.

A tensor $\mathbf{S}$ is characterized by its *eigenvalues* λ_1, λ_2 and λ_3 and its corresponding *eigenvectors* $\mathbf{w}_1$, $\mathbf{w}_2$ and $\mathbf{w}_3$, implied by the characteristic equation $\mathbf{S}w_i = \lambda_i w_i$. For symmetric tensors the eigenvalues are always real, and the eigenvectors are mutually orthogonal.

4 Method Overview

To support an intuitive investigation on the entire 3D tensor dataset we define a family of surfaces that move through the volume, controlled by one parameter. The tensor field restricted to these surfaces is represented by deformed fabric-like texture illustrating the forces on the surface. The texture is stretched or compressed and bent according to the tensor field. Positive eigenvalues, which indicate tension, are illustrated by a texture with low density or a stretched piece of fabric. Correspondingly, negative eigenvalues are represented by a dense texture. Our method can be divided into four steps described in more detail in the next sections:

1. Interpretation of the tensor field as distortion of a flat metric:
   This step corresponds to a transformation of the tensor field into a metric. The resulting metric reflects the physical meaning of the tensor field.
2. Definition of a family of surfaces:
   To define the surfaces we support tow different approaches. The first integrates an additional related scalar field (e.g., pressure or the determinant of the tenors field). The second approach uses some underlying geometry of the numerical simulation to define an artificially family of surfaces.
3. Texture generation:
   Using a fabric-like texture we have enough flexibility to integrate all characteristics of the tensor field. The direction of the fibers reflect the eigenvector fields, their density, thickness, and length the eigenvalues. We compute the texture using LIC. The specific definition of the input noise image determines the density of the fabric. The texture is computed in two steps:
   a) Generation of the three-dimensional spot input image, e.g., using a reaction diffusion approach
   b) Final texture generation by blurring the input image on the surfaces according to the eigenvector fields
4. Representing two orthogonal tensor fields by overlaying the LIC images on the surfaces moving through the dataset

5 Metric Definition

Our method requires the definition of a metric representing a tensor field. (For more details we refer to [9].) Considering a stress or strain tensor field or the symmetrical part of the gradient tensor field of a vector field, positive eigenvalues lead to a separation of particles or expansion of a probe. Eigenvalues equal to zero imply no change in distances, and negative eigenvalues indicate a convergence of the particles or compression of the probe. For a tensor field $\mathbf{T}$ defined on a domain D this behavior can be expressed by a time-depending metric $\mathbf{g}$ of the underlying parameter space D with components g_{ik}:

$$\mathrm{d}s^2(t) = \sum_{ik} \underbrace{(a\delta_{ik} + s_{ik} \cdot t)}_{= g_{ik}} \mathrm{d}x_i \, \mathrm{d}x_k, \tag{1}$$

where δ_{ik} is the Kronecker-delta. The constant a plays the role of a unit length, and t is a time variable that can be used as a scaling factor.

We use a more general mapping that supports more flexibility in the visualization, but still represents the same properties. We use a transformation based on three steps:

1. Diagonalization of the tensor field:

$$\mathbf{T} \mapsto \mathbf{T}' = U \cdot \mathbf{T} \cdot U^T = \mathrm{diag}\,(\lambda_1, \lambda_2, \lambda_3)\,, \tag{2}$$

   where U is the diagonalization matrix.

2. Transformation and scaling of the eigenvalue, to define the metric $\mathbf{g}'$ according to the eigenvector basis:

$$\mathbf{T}' \mapsto \mathbf{g}' = \mathrm{diag}\left(F(\lambda_1), F(\lambda_2), F(\lambda_3)\right), \tag{3}$$

where $F : [-\lambda_{max}, \lambda_{max}] \rightarrow I\!R^+$ is a positive monotone function, with $\lambda_{max} = max\{|\lambda_i(P)|; P \in D, i = 1, 2, 3\}$.

3. Definition of the metric g in the original coordinate system by inverting the diagonalization step defined by Equation (2):

$$\mathbf{g} = U^T \cdot \mathbf{g}' \cdot U. \tag{4}$$

If the mapping F is linear, the three steps can be combined into one step, and F can be applied to the tensor components, independently of the chosen basis.

Since we visualize the metric by a texture and the visual perception of texture attributes is nonlinear, a linear approach is not always a good choice. An alternative approach, for example, is defined by the class of functions: $F[-\lambda_{max}, \lambda_{max}] \rightarrow [\frac{a}{M}, aM]$, where

$$F(-\lambda) = a^2/F(\lambda). \tag{5}$$

The constant a defines the unit, aM the maximum, and $\frac{a}{M}$ the minimum value for F with $M > 1$. Functions with this property can be constructed by using anti-symmetric functions f as exponent:

$$F(\lambda) = a \cdot exp(\sigma \cdot f(\lambda)) \text{ where } f(-\lambda) = -f(\lambda). \tag{6}$$

An example for such a function with $a = 1$ is $F(\lambda; c, \sigma) = exp(\sigma \arctan(c \cdot \lambda))$ is shown in Figure 1. The constant c determines the slope of the function at the origin.

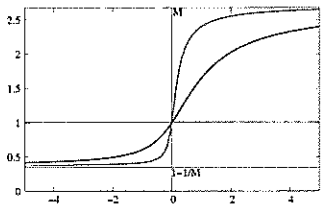


Fig. 1. Example of a non-symmetric transformation function F for two different slopes at the origin.

6 Surface Definition and Tensor Projection

To explore the entire three-dimensional volume D we define a family of surfaces by an additional scalar field S defined over D. We use iso-surfaces defined by the implicit equation

$$S_q(x, y, z) = q, \tag{7}$$

where q is an iso-value moving from a minimal to maximum value. The used scalar field can be an arbitrary additional field. A simple possibility is to move planes through the volume or more complicated surfaces like cylinders or parts of spheres. To compute the textures for the surfaces we project the tensor field

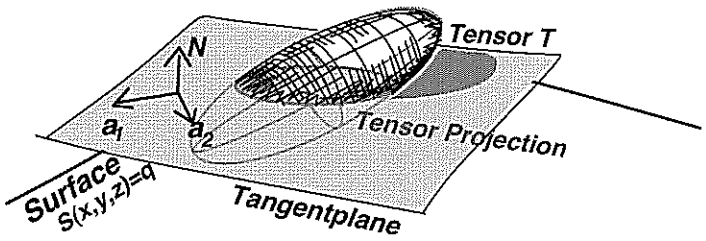


Fig. 2. Projection of the 3D tensor onto the surface S.

onto the surfaces. The surface unit normals are given by the gradient of the scalar field:

$$N = \begin{pmatrix} n_x \\ n_y \\ n_z \end{pmatrix} = \frac{\text{grad}S}{|\text{grad}S|} = \frac{1}{\sqrt{S_x^2 + S_y^2 + S_z^2}} \begin{pmatrix} S_x \\ S_y \\ S_z \end{pmatrix}, \tag{8}$$

where n_x, n_y, n_z are the components of the normal, and $S_x = \frac{\partial S}{\partial x}$, $S_y = \frac{\partial S}{\partial y}$ and $S_z = \frac{\partial S}{\partial z}$ the partial derivatives of the scalar function S. The projection T' of the tensor T onto the surface defined by N is given by

$$T' = P \cdot T \cdot P^T, \tag{9}$$

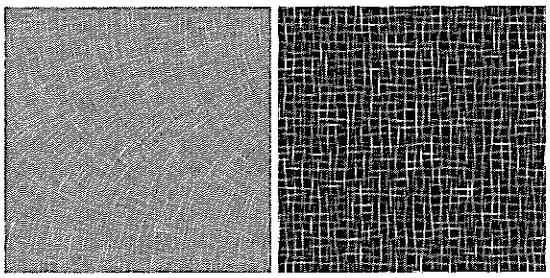
where the projection tensor P to the surface is

$$P = \begin{pmatrix} (1 - n_x^2) & -n_x n_y & -n_x n_z \\ -n_x n_y & (1 - n_y^2) & -n_y n_z \\ -n_x n_z & -n_y n_z & (1 - n_z^2) \end{pmatrix}. \tag{10}$$

The projection tensor is symmetric ($P^T = P$). The resulting tensor has one eigenvector in direction of the surface normal N with eigenvalue zero and two orthogonal eigenvectors, lying in the tangent plane.

7 Texture Generation

To visualize the properties of the resulting metric we use a texture that resembles a piece of fabric. The texture is stretched or compressed and bent according to the metric. To generate the texture we use LIC, a method for vector field visualization [2, 17]. LIC blurs a noise image along the vector field or integral curves. Blurring results in a high correlation of the pixels along field lines, whereas perpendicular to them almost no correlation appears. The resulting image leads to a very effective depiction of flow direction everywhere, even in a dense vector field. We compute a LIC image for both eigendirection fields on the surface. Finally we overlay these images, choosing everywhere the pixel value with the larger intensity, to obtain the fabric-like texture, see Figure 3.



(a) (b)

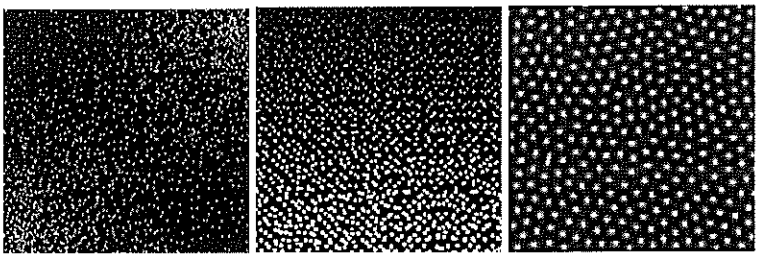
Fig. 3. Overlay of two LIC images to illustrate two direction fields, without integrating the eigenvalues, constant input image and constant convolution length. (a) White noise image; (b) sparse noise image

7.1 Input Texture Definition

Besides the direction field we need for each LIC image a specific noise input. The parameters of this input image determine the properties of the texture. The standard white noise input is the input that supports the highest resolution, but it is not flexible enough to represent a stretched or compressed structure. For this reason, we use sparse input images with lower density and larger spot size even if we obtain a lower resolution. A regular, homogeneous input spot image results in a piece of fabric with constant density and fiber width. An impression of a stretching or compressing can be achieved by changing density, width and length of the fibers. Figures 4 and 5 show examples of different input images and their impact on the fiber structure.

Fiber Density and Fiber Width - Forces Orthogonal to the Fibers:

Stretching and compressing forces orthogonal to the fiber direction are directly related to the eigenvalues of the orthogonal eigenvector field. The change of the fiber density and fiber width can be controlled by the density and spot-size of the input spot texture. For each direction field $\mathbf{w}_i$, $i = 1, 2$, on the surface,



(a) (b) (c)

Fig. 4. Different input images. (a) Spot noise image with changing density; (b) spot noise image with changing spot size; (c) spot noise image generated with isotropic reaction.

we define a specific density d_i and spot size r_i, $i = 1, 2$:

$$\begin{aligned} d_i(x,y,z) &= d_0 \cdot \tfrac{1}{F(\lambda_j)} \\ r_i(x,y,z) &= r_0 \cdot F(\lambda_j) \end{aligned}, \text{ with } j = \begin{cases} 2 \text{ if } i = 1 \\ 1 \text{ if } i = 2, \end{cases} \tag{11}$$

where the function F is defined by Equation (3), λ_1 and λ_2 are the two eigenvalues of the projected tensor field, d_0 and r_0 define the "unit density" and respectively, the "unit-radius" of the circular spots. The density and spot size are spatially varying and depend on the definition of the surfaces. Since the change of the texture parameters in direction of the integration is hardly noticeable we can combine the two textures by using ellipses instead of circles and a direction-dependent density. The resulting texture still depends on the definition of the surface. This means that every time we define a new set of surfaces we have to recompute all input textures.

An alternative construction of the spot textures on the surfaces is the generation of a three-dimensional input image, where the spots are replaced by three-dimensional ellipsoids whose principal axes and radii $r_i = r_0 F(\lambda_i)$ are defined by the metric $\mathbf{g}$ given by Equation (4). The texture on the surface then results from an intersection of the surface with the three-dimensional texture. The direction dependent radius can be expressed by the tensor $\mathbf{r}$, i.e.,

$$\mathrm{r} = r_0 \cdot \mathrm{g} = r_0 \cdot U^T \cdot \begin{pmatrix} F(\lambda_1) & 0 & 0 \\ 0 & F(\lambda_2) & 0 \\ 0 & 0 & F(\lambda_3) \end{pmatrix} \cdot U. \tag{12}$$

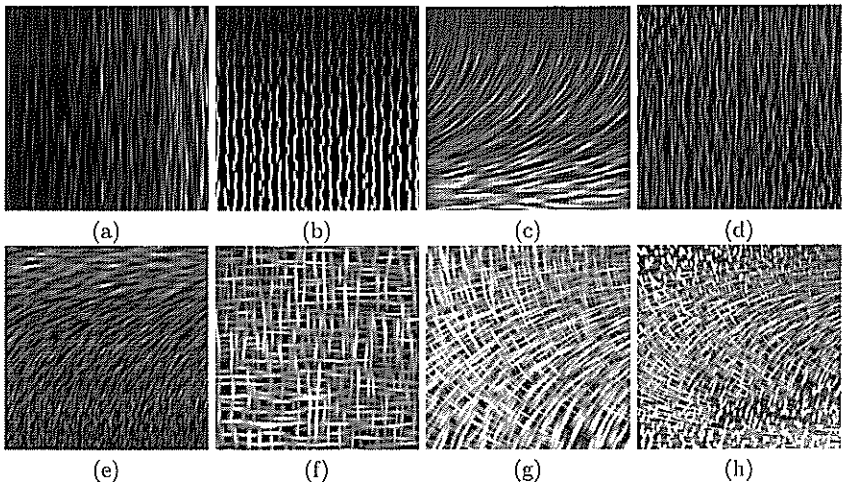


Fig. 5. Effect of changing image parameters; one eigenvector field of a simple synthetic tensor field. In (a)-(c), only the input image is changed corresponding to the eigenvalues of the orthogonal eigenvector field; (a) change of density; (b) change of spot size; (c) change of density and spot size. Image (d) illustrates the effect of changing the convolution length, where the parameters of the input noise image are constant. Image (e) shows a combination of the three parameters (density, spot size, and convolution length). Images (f)-(h) show a combination of both eigenvector fields. In images (g) and (h), only density and spot size are changed; (i) shows a combination of the three parameters.

The direction-dependent density is defined by the tensor $\mathbf{d}$ with the same principal directions but inverse eigenvalues, i.e.,

$$\mathbf{d} = d_0 \cdot U^T \cdot \begin{pmatrix} 1/F(\lambda_1) & 0 & 0 \\ 0 & 1/F(\lambda_2) & 0 \\ 0 & 0 & 1/F(\lambda_3) \end{pmatrix} \cdot U. \tag{13}$$

The parameters r_0 and d_0 define a unit radius and unit density. Using this approach the input texture only has to be computed once, independently of the surface definition, and can be done in a pre-processing step.

Fiber Length - Forces along the Fibers

Stretching and compressing forces parallel to the fibers change length. This property can most easily be controlled by the filter length used for the convolution, and it is merely influenced by the parameters of the input spot texture. We define for each direction field a spatially varying convolution length

$$l_i = l_0 \cdot F(\lambda_i). \tag{14}$$

Color and Color Intensity

In addition to these three "structure" parameters, color intensity can be used to enhance the impression of compression and stretching. We use red for compression and green for tension. We apply a continuous color map ranging from from red for the smallest negative eigenvalues, white for zero eigenvalues, and to green for positive eigenvalues.

7.2 Input Texture Computation

To synthesize the input texture we need an algorithm that places ellipsoids with varying radii and direction-dependent density in the three-dimensional domain. "Reaction diffusion" is a method that generates a texture with the desired properties automatically. Using reaction diffusion a large variety of patterns arising from local nonlinear interactions of two chemicals can be generated. This mechanism was first discussed by Alan Turing [19] to describe the biological process of morphogenesis in biological cells. The basic assumption is that two concurrently operating processes build the biological patterns. The two processes are: diffusion, which transports chemicals form points of higher concentration to points of lower concentration, and reaction, which produces or destroys a chemical.

Reaction diffusion has been used in many different computer graphics applications, including for examples, the generation of natural texture patterns [22, 20, 14] and for vector field visualization [16]. An application very similar to ours was described by Kindlmann et al. [11] who used reaction diffusion to visualize diffusion tensor fields. We provide a short overview of the method.

Reaction Diffusion:

Reaction diffusion can be described by a set of two nonlinear partial differential equations describing the concentrations of two chemicals as function of time:

$$\frac{\partial c_l}{\partial t} = R_l(c_1, c_2) + \nabla d_l \nabla c_l, \qquad l = 1, 2, \tag{15}$$

where c_1 and c_2 are the concentrations of the two chemicals. The functions R_l, $l = 1, 2$, are the functions controlling the reaction of the two chemicals. d_l, $l = 1, 2$, are the diffusion rates for the chemicals. $\nabla^2 c_l$ is the Laplacian of the concentrations c_l, $l = 1, 2$. The resulting "Turing patterns" depend on the specification of the functions R_l. We choose a set of functions proposed by Turing and used by Kindlmann et al. [11] for the visualization of diffusion tensor fields.

$$R_1 = k(16 - c_1 c_2) \tag{16}$$

$$R_2 = k(c_1 c_2 - c_2 - 12 + \beta) \tag{17}$$

Here, k is the reaction rate relative to the diffusion. The value β is the decay rate of c_2; it is a random value in a small interval around zero. This value is the source of slight irregularities in the concentrations. These initially small variations can cause the system to be at first unstable moving over time to a stable state, in which the concentrations vary across the surface and form characteristic patterns. As initial condition for the concentrations are $c_1 = c_2 = 4$ everywhere. Further, we apply periodic boundary conditions.

This model assumes that the diffusion occurs at a uniform rate in all directions and at all positions. It generates a texture with spherical bubbles. A generalization using anisotropic diffusion can be achieved by replacing the scalar diffusion rate d by a diffusion tensor matrix D:

$$\frac{\partial c}{\partial t} = R + \nabla(D\nabla c). \tag{18}$$

The pattern generated with a diffusion tensor D is an ellipsoid whose axes are proportional to the square root of the eigenvalues of D. To generate ellipsoids with the aspect ratio radii as defined in Equation (12) we use the following diffusion:

$$D = r_0^2 \cdot U^T \cdot \begin{pmatrix} F^2(\lambda_1) & 0 & 0 \\ 0 & F^2(\lambda_2) & 0 \\ 0 & 0 & F^2(\lambda_3) \end{pmatrix} \cdot U. \tag{19}$$

To obtain spots of the desired size one has to adjust the parameter k in Equation (16).

Discretization

In our implementation we represent the concentration c as a tree-dimensional array of discrete samples $C_{i,j,k}$. The partial differential equations used ti model the reaction diffusion process may be simulated with a variety of techniques. We use forward Euler integration of the finite-difference equations that one obtains by spatial discretization of the Laplacian:

$$\frac{\partial^2 c}{\partial x^2} \simeq \frac{C_{i+1,j,k} + Ci - 1, j, k - 2C_{i,j,k}}{h^2}. \tag{20}$$

The discrete Laplacian for three-dimensions can be expressed as the convolution of the concentration array with a $3 \times 3 \times 3$ mask. Together with the anisotropic diffusion tensor D the convolution mask M has the following form:

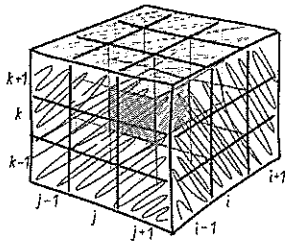


Fig. 6. The 27 voxels involved in the convolution with mask M

$$
\begin{aligned}
z = k+1: &\quad \begin{pmatrix} 0 & D_{yz}/2 & 0 \\ -D_{xz}/2 & D_{zz} & D_{xz}/2 \\ 0 & -D_{yz}/2 & 0 \end{pmatrix}, \\
z = k: &\quad \begin{pmatrix} -D_{xy}/2 & D_{yy} & D_{xy}/2 \\ D_{xx} & -2\,\mathrm{trace}(D) & D_{xx} \\ D_{xy}/2 & D_{yy}/2 & -D_{xy}/2 \end{pmatrix}, \\
z = k-1: &\quad \begin{pmatrix} 0 & -D_{yz}/2 & 0 \\ D_{xz}/2 & D_{zz} & -D_{xz}/2 \\ 0 & D_{yz}/2 & 0 \end{pmatrix}.
\end{aligned} \tag{21}
$$

The discretized reaction diffusion equation is finally given as

$$\Delta C_{t+\Delta t} = C_t + \Delta t(M * C + R(C)). \tag{22}$$

A convolution with the mask M is a weighted sum of the intensity values of the neighboring voxels, see Figure (6).

$$(M * C)_{i,j,k} = \sum_{\substack{x=i-1,i,i+1 \\ y=j-1,j,j+1 \\ z=k-1,k,k+1}} M_{xyz}\, C_{x,y,z}.$$

7.3 Convolution

The last step for final fabric texture creation is the computation of two line convolution images. The two direction fields are defined by the tensor field projected onto the set of surfaces defined by Equation (9). For the integration of the integral curves we use Runge-Kutta of fourth order. The LIC image is computed using Fast-LIC as proposed by Hege et al. [17]. The convolution length depends on the eigenvalues of the tensor field. It is defined by Equation (14). For integration we do not require an explicit surface definition, since this information is already part of the definition of the projected tensor field.

7.4 Visualization

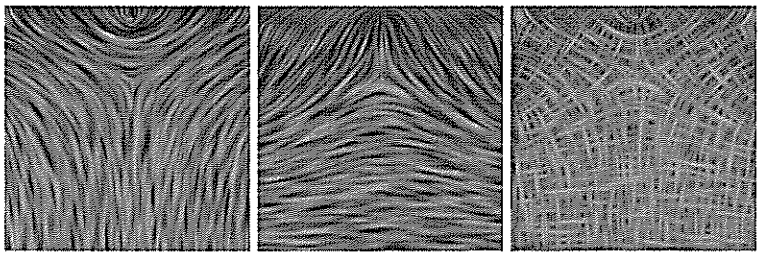
Representing an entire texture leads to occlusion problems. Therefore, we only show the texture on one surface that is moved through the volume, defined by an iso-value. We use two approaches to represent the surfaces:

- Volume rendering transfer function illustrating the volume only in an ϵ-neighborhood of the chosen iso-value.
- explicitly extraction and visualization of the surfaces.

8 Results and Conclusions

We have used two data sets of stress fields being the results of applying different load combinations to a solid block. These datasets are well-understood and therefore appropriate to evaluate our method. The tensor field resulting from the numerical simulation is continuous inside each cell, but not on cell boundaries. In Figures 7 and 8, we show two-dimensional planar slices through a one-point and two-point load dataset. These images were generated using a specific input noise texture for each eigenvector field. The input textures consist of circular spots of varying size, density and (Figure 8) also color. Figure 9 shows results for the same dataset when using only one input texture generated with reaction diffusion. These images provide also a good representation of the tenor field, but the features are not as clearly visible when compared with the corresponding results based on the spot noise input. Image 10 show examples of textures on curved surfaces for the two-point load dataset. All images provide a good visual segmentation of regions of compression and expansion.

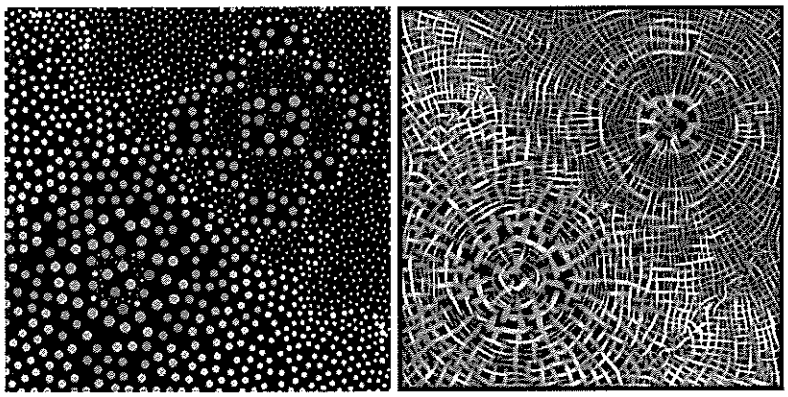
The interpretation of a tensor field as a distortion of a flat metric can be used to produce visualizations based on the real physical effect of the tensor



(a) (b) (c)

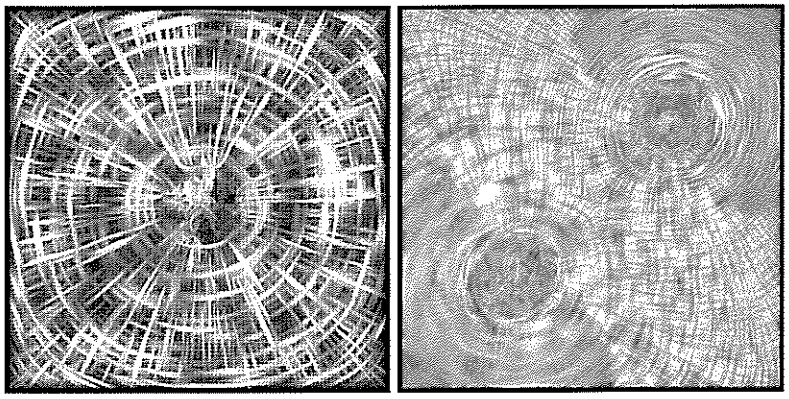
Fig. 7. Images showing a yz-plane slice of single top-load data set, where a force is applied in z-direction. Images (a) and (b) illustrate the two eigenvector fields separately, in (c) they are overlaid. Spot size and density are changed according to eigenvalues.

field. Using a three-dimensional input texture simplifies texture generation for arbitrary surfaces substantially. Having a pre-computed of the input texture, the three dimensional volume can easily be examined using different surfaces. The advantage of using reaction diffusion to generate input texture is the methods capability of automatic placement and packing of the spots according to the desired density and spot size, but is also has several disadvantages. The generated features are not as "crisp" as they are for spot noise input. The determination of appropriate parameters in the reaction diffusion equation is not intuitive and its numerical computation is time-consuming. To improve our method we intend to use a more efficient algorithm that can achieve a much higher performance, considering, for example, the approach proposed by Witkin and Kass [22]. We also plan to investigate other ways to generate the three-dimensional input textures, ways that are less expensive and produce crisp features. Other possible extensions are approaches for a better visualization of the resulting surfaces.



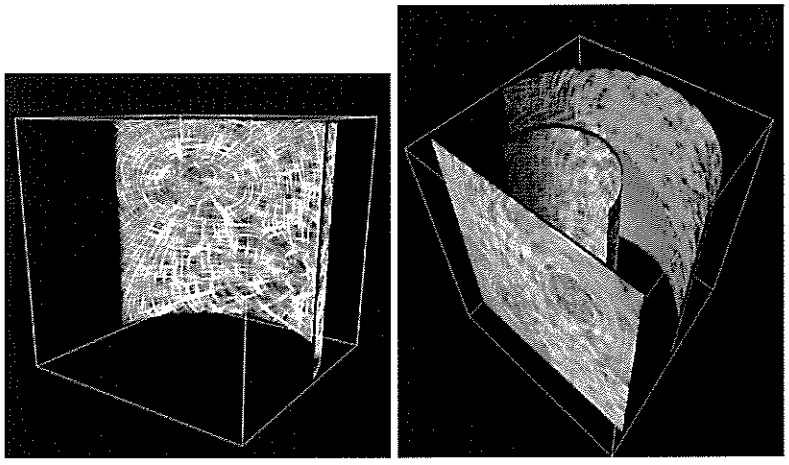
(a) (b)

Fig. 8. The images represent a xz-plane slice of a two-force dataset. (a) represents one of the two spot noise input textures for the final image. (b) The lower left circle corresponds to the pushing and the right to the pulling force.



(a) (b)

Fig. 9. These images represent a xz-plane slice of (a) a one-force and (b) a two-force dataset using three-dimensional spot noise input generated with diffusion reaction.



(a) (b)

Fig. 10. These images are examples for the representation of several curved surfaces to explore the three-dimensional domain for the two-point load dataset.

References

1. Ed Boring and Alex Pang. Interactive Deformations from Tensor Fields. Proceedings of the Visualization '98 Conference, 1998, pages 297-304.
2. Brian Cabral and Leith Leedom. Imaging Vector Fields Using Line Integral Convolution. In SIGGRAPH '93 Conference Proceedings, 1993, pages 263-272.
3. Thierry Delmarcelle and Lambertus Hesselink. Visualization of second order tensor fields and matrix data. Proceedings of the Visualization '92 Conference, 1992, pages 316-323.
4. Robert B. Haber. Visualization Techniques for Engineering Mechanics. In Computing Systems in Engineering, 1990, pages 37-50.
5. Lambertus Hesselink, Thierry Delmarcelle and James L. Helman . Topology of Second-Order Tensor Fields. In Computers in Physics, 1995, pages 304-311.
6. Lambertus Hesselink, Yuval Levy and Yingmei Lavin. The Topology of Symmetric, Second-Order 3D Tensor Fields. IEEE Transactions on Visualization and Computer Graphics, vol 3(1), 1997, pages 1-11.
7. Ingrid Hotz. Visualizing second order symmetric tensor fields by metric surfaces. Work in Progress of the Conference IEEE Visualization, 2001.
8. Ingrid Hotz. Isometric Embedding by Surface Reconstruction from Distances. Proceedings of the IEEE Visualization '02 Conference, 2002, pages 251-258.
9. Ingrid Hotz, Louis Feng, Hans Hagen, Bernd Hamann, Boris Jeremic, and Kenneth Joy. Physically Based Methods for Tensor Field Visualization. Proceedings of the IEEE Visualization '04 Conference, 2004, pages 123-130.
10. B. Jeremic, Gerik Scheuermann and Jan Frey et al. Tensor Visualization in Computational Geomechanics. In International Journal for Numerical and Analytical Methods in Geomechanics, 2002, pages 925-944.
11. Gordon Kindlmann, David Weinstein and David Hart. Strategies for Direct Volume Rendering of Diffusion Tensor Fields. In IEEE Transactions on Visualization and Computer Graphics, vol. 6(2), pages 124-138, 2000.
12. Ron D. Kriz, Edward H. Glaessgen and J.D. MacRae. Visualization Blackboard: Visualizing gradients in composite design and fabrication. IEEE Computer Graphics and Applications, vol 15, 1995, pages 10-13.
13. W. C. de Leeuw and J. J. van Wijk. A Probe for Local Flow Field Visualization. Proceedings of the Visualization '93 Conference, 1993, pages 39-45.
14. James D. Murray. Mathematical Biology I. An introduction, 3rd edition in 2 volumes, Springer, ISBN: 0-387-95223-3, 2002.
15. J. Ou and E. Hsu". Generalized Line Integral Convolution Rendering of Diffusion Tensor Fields. Proceedings of the 9th Scientific Meeting and Exhibition of the International Society for Magnetic Resonance in Medicine (ISMRM), page 790, 2001.
16. Allen R. Sanderson and Chris R. Johnson and Robert M. Kirby. Display of Vector Fields Using a Reaction-Diffusion Model. In Proceedings of the IEEE Visualization 2004, pages 115-122, 2004
17. Detlev Stalling and Hans-Christian Hege. Fast and Resolution Independent Line Integral Convolution. In SIGGRAPH '95 Conference Proceedings, 1995, pages 149-256.
18. Xavier Tricoche, Gerik Scheuermann and Hans Hagen. Tensor Topology Tracking: A Visualization Method for Time-Dependent 2D Symmetric Tensor Fields. A. Chalmers and T.-M. Rhyne, editors, EG 2001 Proceedings vol 20(3) of Computer Graphics Forum. 2001, pages 461-470.

19. Alan Turing. The chemical basis of morphogenesis. In Philosophical Transactions of the Royal Society London, B237:37-72, 1952
20. Greg Turk. Generating textures on arbitrary surfaces using reaction diffusion. In proceedings SIGGRAPH '91, volume 25, pages 289-298. Addison Wesley, 1991
21. David M. Weinstein, Gordon L. Kindlmann and Eric C. Lundberg. Tensorlines: Advection-Diffusion based Propagation through Diffusion Tensor Fields. Proceedings IEEE Visualization '99 conference,1999, pages = 249-254.
22. Andrew Witkin and Michael Kass. Reaction-diffusion textures. In Proceedings SIGGRAPH'91, volume 25, pages 299-308. Addison Wesley, 1991.
23. Xiaoqiang Zheng and Alex Pang. Volume Deformation For Tensor Visualization. Proceedings of the Visualization '02 Conference, 2002, pages 379-386.
24. Xiaoqiang Zheng and Alex Pang. Interaction of Light and Tensor Field. Proceedings of VISSYM '03, 2003, pages 157-166.
25. Xiaoqiang Zheng and Alex Pang. HyperLIC, Proceedings of the IEEE Visualization '03 Conference, 2003, pages 249–256.
26. Xiaoqiang Zheng and Alex Pang. Topological Lines in 3D Tensor Fields. Proceedings of the IEEE Visualization '04 Conference, 2004, pages .
27. Leonid Zhukov and Alan H. Barr. Oriented Tensor Reconstruction: Tracing Neural Pathways from Diffusion Tensor MRI. Proceedings of the IEEE Visualization '02 Conference, 2002, pages 387-394.