Physically Based Methods for Tensor Field Visualization

Ingrid Hotz[*], Louis Feng[†]
IDAV, University of California, Davis, USA

Hans Hagen
Technical University of Kaiserslautern, Germany

Bernd Hamann, Kenneth Joy
IDAV, University of California, Davis, USA

Boris Jeremic
Dept. of Civil Eng. University of California, Davis, USA

Abstract

The physical interpretation of mathematical features of tensor fields is highly application-specific. Existing visualization methods for tensor fields only cover a fraction of the broad application areas. We present a visualization method tailored specifically to the class of tensor field exhibiting properties similar to stress and strain tensors, which are commonly encountered in geomechanics.

Our technique is a global method that represents the physical meaning of these tensor fields with their central features: regions of compression or expansion. The method is based on two steps: first, we define a positive definite metric, with the same topological structure as the tensor field; second, we visualize the resulting metric. The eigenvector fields are represented using a texture-based approach resembling line integral convolution (LIC) methods. The eigenvalues of the metric are encoded in free parameters of the texture definition. Our method supports an intuitive distinction between positive and negative eigenvalues.

We have applied our method to synthetic and some standard data sets, and "real" data from Earth science and mechanical engineering application.

CR Categories: I.3.3. [COMPUTER GRAPHICS]: Picture/Image Generation—Line and curve generation; J.2. [Physical Science and engineering]: Engineering

Keywords: tensors field, stress tensor, strain tensor, LIC

1 Introduction

Tensor data play an important role in many disciplines. In geomechanics or solid state physics for example tensors are used to express the response of material to applied forces; in geometry, the curvature and metric tensors describe the fundamental properties of surfaces; the gradient tensor of a flow field provides a characterization of flow structure. A very different application area is medical imaging, where the diffusion tensor describes the directional dependence of molecule motion. These areas are only a few examples. Mathematically, a tensor is a linear function that relates different vectorial quantities. Its high dimensionality makes it very complex and difficult to understand. Thus, there is much need for tensor field visualization that enables easy and intuitive understanding. Because the physical interpretation of mathematical features is highly application-specific it is important that the visualization is closely driven by the special application.

We present a visualization method for symmetric tensor fields of second order that are similar to stress, strain and the symmetrical part of the gradient tensor. Our method provides a continuous representation of the tensor field, which is closely related to the physical meaning of the tensor field and emphasizes regions of expansion and compression. This behavior is strictly connected to the sign of the eigenvalues. A change of sign indicates regions where the material tends to crack. Therefore, we do not focus on the isotropy behavior of tensor fields but on the integration of the sign of the eigenvalues. The underlying idea of our method is to transform the tensor field into a metric, which is visualized. To represent the resulting metric we use a texture that is aligned to the eigenvector fields, similarly to LIC [2, 13]. The metric eigenvalues are included using the free parameters of the convolution filter, like filter length, and free parameters of the input noise texture. Finally, we obtain a dense texture in regions of compression and a sparse texture in regions of expansion. By an animation of these parameters we can enhance the impression of stretching and compression. The representation expresses what is physically happening in the field.

In Section 2, we provide an overview of related work. We review the mathematical basics of tensor fields in Section 3. In Section 4, we explain our method, its motivation and realization. Section 5 discusses results.

2 Related Work

In contrast to the visualization of scalar or vector fields, tensor field visualization is still in its infancy. Even symmetric tensors with six independent values contain so much information at each point that it is not easy to capture at once. Several good visualization techniques exist for tensor fields, but they only cover a few specific applications. Many of the existing visualization methods are extensions from vector field visualization, which focus on the technical generalization without providing an intuitive physical interpretation of the resulting images. These methods often concentrate on the representation of eigendirections and neglect the importance of the eigenvalues. Therefore, in many application areas traditional 2D plots are still used, which represent the interaction of two scalar variables.

A basic way to represent a tensor field is using icons. They illustrate the characteristics of a field at some 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 and scaled according to the corresponding eigenvalues (see for example Kriz et al. or Haber, [6, 12]). More complex glyphs were constructed by Leeuw et al. showing additional features using flow probes [3]. Even though these icons represent the tensor value at one point well they fail to give a global view of the tensor field. The problem of choosing appropriate points to examine is left to the user. Thus, these methods have limited usage in exploration of complete data sets. Furthermore, they hold the problem or cluttering.

A more advanced but still discrete approach uses hyperstreamlines. This approach is strongly related to streamline methods used for vector fields. They were introduced by Delmarcelle and Hesselink [5] and have been utilized in a geomechanical context by Jeremic et al. [11]. Given a point p 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. Even though this method extracts more information than icons, it still leaves the problem of choosing appropriate seed

[*]e-mail: ihotz@ucdavis.edu
[†]e-mail: zfeng@ucdavis.edu

points and is limited to low resolution due to cluttering.

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 [7, 8, 14]. 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 general physical significance. (For example, for stress fields they are points of high symmetry).

A very different approach was chosen by Pang et al. [1, 16]. They considered the tensor field as a force field that deforms an object placed inside the field. The local deformation of these probes, such as planes and spheres, illustrate the tensor field. The deformation is computed by multiplying the local tensor with a user-specified vector in this point. These point-probe techniques can be used at interactive rates. However, the inner product reduces the tensor field to a vector field and thus only displays the tensor information corresponding to one direction. In addition, only a small number of probes can be included in one picture to avoid visual clutter. Zheng et al. extended this method [18] by applying it to light rays that are bended by the local tensor value.

Zhukov and Barr [19] developed a technique to visualize diffusion tensors of magnetic resonance data 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. This approach is an adaptation of hyperstreamlines to diffusion tenors. An approach that arose in a similar context is an adaptation of direct volume rendering to diffusion tensor fields presented by Kindlmann et al. [15]. 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. These approaches 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.

An approach that does attempt to provide a continuous global view is the adaptation of LIC-similar texture approaches to tensor fields by Pang et al. [17]. Here a white noise textures 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 good at showing the anisotropy of a tensor field. However, one problem of this method is the need for integration of the sign of the eigenvalues. Points with eigenvalues with opposite sign are illustrated as isotropic.

A geometrical approach was followed by Hotz et al. [9, 10]. 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. An isometric embedding is used to visualize the resulting abstract metric. The problem with this approach is that, in general, no global Euclidean embedding exists for arbitrary metrics, and several "patches" must be used to cover the entire domain [9].

3 Mathematical basics

A tensor is a type of 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. The curvature tensor used in geometry, for example, relates a direction to the change of the surface normal. The physical stress, strain or elasticity tensors express the response of material to an applied force. We now review some required basics.

Let V be a vector space of dimension n, and let V^* be its dual space. In typical applications, V is the tangent space at a point of a manifold; the elements of V may represent velocities or forces. A *tensor of order* (q,p) is a multilinear mapping from the tensor product of p copies of V and q copies of the dual vector space V^* into the space of real numbers, i.e.,

$$\mathbf{T} : \underbrace{v^* \otimes \ldots \otimes V^*}_{q-times} \otimes \underbrace{V \otimes \ldots \otimes V}_{p-times} \rightarrow \mathbb{R}. \qquad (1)$$

According to a basis of V and V^*, $\mathbf{T}$ can be expressed by a n^{p+q}-dimensional array of real numbers. It must be emphasized that the tensor $\mathbf{T}$ and the representing array are not the same. Tensors of order zero $(p = q = 0)$ are scalars, tensors of first order $(q = 1, p = 0)$ are vectors. A *tensor field* over some domain D assigns to every point $P \in D$ a tensor $\mathbf{T}(P)$.

We restrict ourselves to tensors of second order $(q = 2, p = 0)$ defined over three-dimensional Cartesian space. Using a fixed coordinate basis, each tensor can be expressed as a 3×3 matrix, given by nine independent scalars:

$$\mathbf{T} = (t_{ij}) = \begin{pmatrix} t_{11} & t_{12} & t_{13} \\ t_{21} & t_{22} & t_{23} \\ t_{31} & t_{32} & t_{33} \end{pmatrix}. \qquad (2)$$

A tensor $\mathbf{T}$ is called symmetric if for any coordinate basis, the corresponding array of scalars is symmetric, i.e., $t_{ij} = t_{ji}$ 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 $\mathbf{S}$ and an antisymmetric $\mathbf{A}$ part:

$$\mathbf{T} = \mathbf{S} + \mathbf{A} \qquad (3)$$

where

$$s_{ij} = \frac{1}{2}(t_{ij} + t_{ji}) \text{ and } a_{ij} = \frac{1}{2}(t_{ij} - t_{ji}). \qquad (4)$$

The antisymmetric part is defined by three independent scalars and can be interpreted as a rotation around the three-dimensional vector $a = (a_{23}, -a_{13}, a_{12})$. The symmetric part of the tensor $\mathbf{S}$, is defined by six independent scalars and is represented by a symmetric matrix:

$$\mathbf{S} = (s_{ij}) = \begin{pmatrix} s_{11} & s_{12} & s_{13} \\ s_{12} & s_{22} & s_{23} \\ s_{13} & s_{23} & s_{33} \end{pmatrix}. \qquad (5)$$

The tensor $\mathbf{S}$ is characterized by its *eigenvalues* λ_1, λ_2 and λ_3 and corresponding *eigenvectors* $\mathbf{w}_1$, $\mathbf{w}_2$ and $\mathbf{w}_3$, implied by the eigenvalue equation:

$$\mathbf{S}w_i = \lambda_i w_i \qquad (6)$$

For symmetric tensors the eigenvalues are always real and the eigenvectors mutually orthogonal. The transformation of the basis to the eigenvectors basis is given by the orthogonal matrix

$$U = (\mathbf{w}_1, \mathbf{w}_2, \mathbf{w}_3). \qquad (7)$$

If U^T defines the transposed matrix of U, then

$$U \cdot \mathbf{S} \cdot U^T = \begin{pmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{pmatrix} = \mathrm{diag}(\lambda_1, \lambda_2, \lambda_3) \qquad (8)$$

4 The Method

4.1 Basic Idea

Because the physical interpretation of mathematical features of tensor fields strongly depends on the application, we want to restrict ourselves to a class of related tensor fields. These are stress and strain tensor fields and the symmetrical part of the gradient tensor field of a vector field. To motivate our approach we start with an exemplary analysis of the symmetrical part of a gradient tensor **S**, given by:

$$s_{ij} = \frac{1}{2}\left(v_{i,j} + v_{j,i}\right), \tag{9}$$

where $\mathbf{v} = (v_1, v_2, v_3)$ is the basic flow field. The variable $v_{i,j}$ denotes the partial derivative of the ith component of $\mathbf{v}$ with respect to the coordinate x_j. The tensor **S** describes the separation of neighboring particles in the flow field. This behavior is expressed by the following equation:

$$\frac{\mathrm{d}}{\mathrm{d}t}(\mathrm{d}s^2) = \sum_{i,k=1}^{3} s_{ik}\,\mathrm{d}x_i\,\mathrm{d}x_k = \sum_{j=1}^{3} \lambda_j\,\mathrm{d}u_j^2. \tag{10}$$

Here, $\mathrm{d}s = (\mathrm{d}x_1, \mathrm{d}x_2, \mathrm{d}x_3)$ and $\mathrm{d}s^2$ is the quadratic distance of two neighboring points. $\lambda_j, j = 1,2,3$ are the eigenvalues of **T**, and $\mathrm{d}u_j$ are the components of $\mathrm{d}x$ corresponding to the eigenvector basis $\{\mathbf{w}_j, j = 1,2,3\}$. If we reduce the observation to one eigendirection $\mathbf{w}_i$ the change of $\mathrm{d}s^2$ is defined by the corresponding eigenvalue λ_i. We must consider these cases:

$$\begin{array}{lcl} \lambda_i > 0 & \longrightarrow & \frac{\mathrm{d}}{\mathrm{d}t}\,\mathrm{d}s^2 > 0, \\ \lambda_i = 0 & \longrightarrow & \frac{\mathrm{d}}{\mathrm{d}t}\,\mathrm{d}s^2 = 0, and \\ \lambda_i < 0 & \longrightarrow & \frac{\mathrm{d}}{\mathrm{d}t}\,\mathrm{d}s^2 < 0. \end{array} \tag{11}$$

A similar behavior can be observed for the deformation of a probe in a stress field (see Fig. 1).

To summarize we observe that for a gradient field as well as for stress and strain tensors 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.

Considering a time-independent vector field, the integration of the separation Equation (10) results in the following expression for $\mathrm{d}s^2$:

$$\mathrm{d}s^2(t) = \mathrm{d}s^2(0) + \sum_{ik}(s_{ik} \cdot t)\,\mathrm{d}x_i\,\mathrm{d}x_k. \tag{12}$$

Using

$$\mathrm{d}s^2(0) = a \cdot \sum_i \mathrm{d}x_i\,\mathrm{d}x_i \tag{13}$$

we obtain:

$$\boxed{\mathrm{d}s^2(t) = \sum_{ik} \underbrace{(a\delta_{ik} + s_{ik} \cdot t)}_{=:\,g_{ik}}\,\mathrm{d}x_i\,\mathrm{d}x_k} \tag{14}$$

Where δ_{ik} is the Kronecker-delta. The tensor **g** with components $g_{ik} = a\delta_{ik} + s_{ik} \cdot t$ can be interpreted as metric of the underlying parameter space D. The constant a plays the role of a unit length, and t is a time variable that can be used as a scaling factor. This metric definition is the basis of our tensor field visualization method. The transformation of the tensor field into a metric is described in the next section and the visualization of the resulting metric is discussed in Section 4.3.

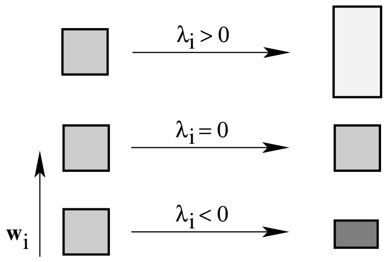


Figure 1: Deformation of a unit probe under influence of a stress tensor in direction of eigenvector $\mathbf{w}_i$. Eigenvalues larger than zero correspond to a tensile, and eigenvalues smaller than zero to a compressive force in the direction of the eigenvector.

4.2 The Transformation

Based on the observations made in Section (4.1), we define a transformation of the tensor field into a metric. We do not exactly follow the motivating Equation (14) but use a more flexible approach.

Let **T** be a tensor field defined on a domain D. The tensor in a point $P \in D$ is given by $\mathbf{T}(P)$. For each point P the tensor $\mathbf{T}(P)$ is mapped to a metric tensor $\mathbf{g}(P)$ describing the metric in P. In the most general form, the assignment is achieved by the following three steps:

1. Diagonalization of the tensor field:
   Switching from the original coordinate basis to the eigenvector basis $\{\mathbf{w}_1, \mathbf{w}_2, \mathbf{w}_3\}$, we obtain a diagonal tensor $\mathbf{T}'$ with the eigenvalues of **T** on its diagonal:
   $$\mathbf{T} \mapsto \mathbf{T}' = U \cdot \mathbf{T} \cdot U^T = \mathrm{diag}(\lambda_1, \lambda_2, \lambda_3). \tag{15}$$
   U is the diagonalization matrix defined in Equation (7).
2. Transformation and scaling of the eigenvalue, to define the metric $\mathbf{g}'$ according to the eigenvector basis:
   $$\mathbf{T}' \mapsto \mathbf{g}' = \mathrm{diag}(F(\lambda_1), F(\lambda_2), F(\lambda_3)), \tag{16}$$
   where $F : [-\lambda_{max}, \lambda_{max}] \to I\!R^+$ is a positive monotone function, where
   $$\lambda_{max} = max\{|\lambda_i(P)|; P \in D, i = 1,2,3\} \tag{17}$$
   is the largest eigenvalue according to absolute value.
3. We obtain the metric g in the original coordinate system by inverting the diagonalization defined in Equation (15):
   $$\mathbf{g} = U^T \cdot \mathbf{g}' \cdot U. \tag{18}$$

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.

We now consider the function F, which defines the metric. To underline the motivation defined by (14), we can rewrite Equation (16) in the following way:

$$\mathbf{g}' = a \cdot \mathbf{I} + \sigma \cdot \mathrm{diag}(f(\lambda_1), f(\lambda_2), f(\lambda_3)). \tag{19}$$

Here **I** is the unit matrix, $a = F(0)$ defines the unit length, and $\sigma \neq 0$ is an appropriate scaling factor, that guarantees that the resulting metric is positive definite. The function $f : I\!R \to I\!R$ is a monotone function with $f(0) = 0$.

The resulting metric **g** has the following properties:

- It is positive definite and symmetric.
- Its eigenvector field is the same as the eigenvector field of the original tensor field **T**, which means the tensor field topology in the sense of Delmarcelle et al. [4] is preserved.

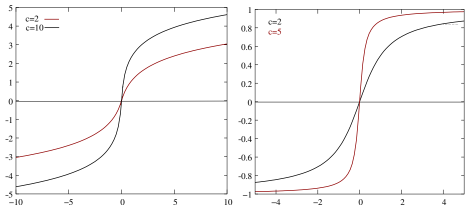


Figure 2: Two examples for a anti-symmetric transformation function f. Left: Logarithmic function; right: arc-tangent for two different slopes for $\lambda = 0$.

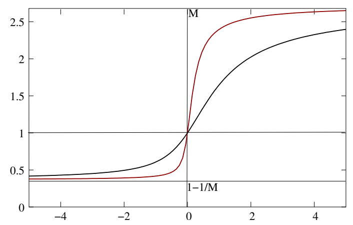


Figure 3: Example for a non-symmetric transformation function F for two different slopes in the origin.

- The zero tensor is mapped to a multiple of the unit matrix $a \cdot I$. Its eigenvalues γ_j are given by $F(\lambda_j)$. Positive eigenvalues are mapped to eigenvalues greater than a, negative eigenvalues are mapped to eigenvalues smaller than a but larger then zero.
- Since the transformation is invertible, a one-to-one correspondence of the metric and the tensor field is given.

Examples for the transformation function F:

We can define the function F by using the scalars a, σ, and the function f as given in Equation (19): $F(\lambda) = a + \sigma f(\lambda)$.

- Identity: $f = id$, $f(\lambda) = \lambda$
  Since f is linear, the metric g is defined as:
  $$g_{ij} = F(t_{ij}) = a + \sigma \cdot t_{ij}. \tag{20}$$
  This equation corresponds exactly to our motivating Equation (14), where σ plays the role of the time variable t. With $\sigma < a/\lambda_{max}$ we can guarantee that the metric is positive definite. The value λ_{max} is defined by (17).
- Anti-symmetric functions f: $f(-\lambda) = -f(\lambda)$:
  To empathize regions where the eigenvalues change sign it makes sense to choose a function f with a larger slope in the neighborhood of zero. From the large class of functions satisfying this condition we have considered two examples:

  The first example is an anti-symmetric logarithmic function,
  $$f(\lambda;c) = \begin{cases} \log(c \cdot \lambda + 1) & \text{for } \lambda \geq 0 \\ - \log(1 - c \cdot \lambda) & \text{for } \lambda < 0 \end{cases}. \tag{21}$$
  If we require $\sigma < a/\log(c \cdot \lambda_{max} + 1)$ the resulting metric is positive definite.

  The second example is an asymptotic function f:
  $$f(\lambda;c) = \arctan(c \cdot \lambda). \tag{22}$$
  Here, the limitation of $\sigma < 2a/\pi$ is independent of λ_{max}.

  For both functions, the constant c controls the "sharpness" at the zero crossing. For hight values of c the function becomes steeper, see Figure 2.

As the visual perception of texture attributes is nonlinear, this anti-symmetric approach is not always a good choice. An alternative that takes care of this aspect is defined by the class of functions $F[-\lambda_{max}, \lambda_{max}] \rightarrow [\frac{a}{M}, a \cdot M]$, with
$$F(-\lambda) = \frac{a^2}{F(\lambda)}. \tag{23}$$
The constant a defines again the unit, $a\dot{M}$ the maximum, and $\frac{a}{M}$ the minimum value for F satisfying $M > 1$. Functions with this property can be obtained using anti-symmetric functions f as exponent:
$$F(\lambda) = a \cdot exp(\sigma \cdot f(\lambda)) \text{ where } f(-\lambda) = -f(\lambda). \tag{24}$$
An example for such a function with $a = 1$ is $F(\lambda;c,\sigma) = exp(\sigma \arctan(c \cdot \lambda))$. c determines the slope of the function in the origin, see Figure 3. The resulting metric is always positive definite. Therefore, the scaling factor σ is not limited.

4.3 Visualization

The problem of visualizing a tensor field has become a problem of visualizing an abstract metric. One way to solve this problem is an isometric embedding of the metric [9]. The disadvantage of this approach is that it is restricted to two dimensions, and its existence is only guaranteed locally. In general, several patches are needed to cover the entire domain. Because we want to provide a global representation of a field we decided to follow a different approach: Our basic idea is to use a texture that resembles a piece of fabric to express the characteristic properties of the metric. The texture is stretched or compressed and bended according to the metric. Large values of the metric, which indicate large distances, are illustrated by a texture with low density or a stretched piece of fabric. We use a dense texture for small values of the metric. One can also think of a texture as probe inserted into the tensor field.

We generate the texture using LIC, a very popular method for vector field visualization. LIC blurs a noise image along the vector field or integral curves. Blurring results in a high correlation of the pixel along field lines, whereas perpendicular to them almost no correlation appears. The resulting image leads to a very effective depiction or flow direction everywhere, even in a dense vector field. LIC was introduced in 1993 by Cabral and Leedom [2]. Since the method was introduced there have been many publications proposing extensions and improvements to make it faster [13] and more flexible.

We compute a LIC-image for every eigenvector field to illustrate the eigendirections of the tensor field. For the integration of the integral curves we use Runge-Kutta of 4th order, the LIC image is computed using Fast-LIC as proposed in [13]. In each LIC image the eigenvalues of every eigenvector field are integrated using the free parameters of the underlaying noise image and the convolution. At the end we overlay all resulting LIC images to get the fabric-like texture.

Input Noise Image

For each LIC image we need besides the direction field a specific noise image input. We use the free parameters of this input image to encode properties of the metric. Three basic parameters are changed according to the eigenvalues. They are: density, spot size, and color intensity of the spots. Considering these parameters, the standard white noise image is the noise image with maximum density, minimal spot size and constant color intensity. It allows one to obtain a very good overall impression of the field; its resolution is only limited by the pixel size. But it is not flexible enough to integrate the eigenvalues which represent fundamental field properties besides the directions. For this reason we use sparse input images,

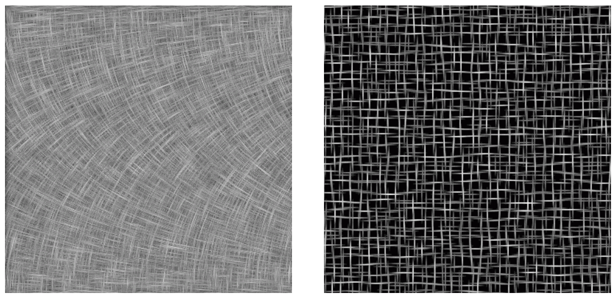

(a) (b)

Figure 4: Overlay of two LIC images to illustrate to direction fields, without integrating the eigenvalues, constant input image and constant convolution length (a) white noise image (b) sparse noise image

with lower density and larger spot size even if we obtain a lower resolution. Some examples for different input images with changing density and spot size are provided in Figure 5. The connection of these parameters to the eigenvalues are explained in more detail in the next sections.

Density

To define the density depending on the eigenvalues we consider the change of the density of a fiber texture under the influence of a stress tensor field. Only the change of density orthogonal to the fiber structure is directly visible. A compression orthogonal to the fibers leads to an increasing density, an expansion to a decreasing density. For each direction field $\mathbf{w}_i$ we define a specific density d_i depending on the orthogonal eigenvalues. For two dimensional textures this leads to the following definition of a one-dimensional density d_i [spots$/cm$]:

$$d_i(\lambda) = d_0 \cdot \frac{1}{F(\lambda_j)}, \text{ with } j = \begin{cases} 2 & \text{if} \quad i=1 \\ 1 & \text{if} \quad i=2 \end{cases} . \tag{25}$$

F is defined by (16). d_0 defines the "unit-density", $d(0) = d_0/F(0)$. If we use F defined in Equation (23) we obtain as minimum density value $\frac{d_0}{aM}$ and as maximum value $\frac{d_0 M}{a}$. An example for a density function is shown in Figure 6.

When we switch to three dimensions we have two orthogonal eigenvalues and thus a direction-dependent density:

$$\begin{aligned} d_{i,j}(\lambda) &= d_0 \cdot \frac{1}{F(\lambda_j)}, \text{ in direction of } \mathbf{w}_j, \\ d_{i,k}(\lambda) &= d_0 \cdot \frac{1}{F(\lambda_k)}, \text{ in direction of } \mathbf{w}_k, \end{aligned} \tag{26}$$

where $j=2, k=3$ if $i=1$; $j=3, k=1$ if $i=2$: and $j=1, k=2$ if $i=3$.

Spot Size

The spot size of the noise image affects the width of the fiber. Increasing the radius of the underlying noise image leads to thicker, decreasing the radius leads to thinner fibers. Similar to the definition of the density this value is controlled by the orthogonal eigenvalues. The radius of the spots changes proportionally to the eigenvalues. Because a change of the spot size in direction of the integral lines does not affect the resulting image much, we define for two-dimensional textures circular spots specified by a radius r_i. For three dimensions, we define ellipsoids with three different diameters according to the three eigenvalues:

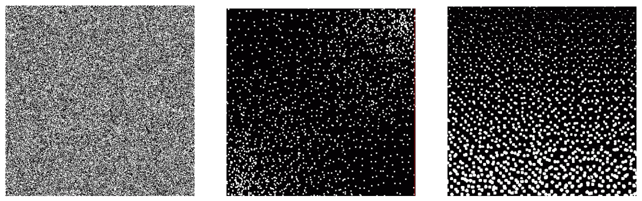

(a) (b) (c)

Figure 5: Example for different input images. (a) white noise image with maximum resolution, (b) spot noise image with changing density, (c) spot noise image with changing spot size

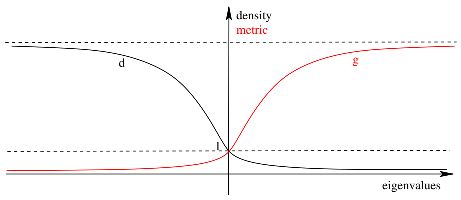


Figure 6: Example of a density function used to encode a metric

$$r_{i,j} = \frac{r_0}{d_{i,j}}. \tag{27}$$

Convolution Length

The defined noise image only uses the eigenvalues orthogonal to the actual eigendirection field. A stretching or compressing in the direction of the integral lines changes the length of the fibers. Fiber length is correlated directly with the length of the convolution filter l_i. This insight leads to the following definition of a filter length proportional to the eigenvalues of the metric:

$$l_i = l_0 \cdot F(\lambda_i). \tag{28}$$

Color and Color Intensity

Since the influence of the density and the spot radius on the total brightness is the same (quadratic for two and cubic for three dimensions), the total brightness of the fibers is constant for this spot size and density definition. If we want to enforce the impression of stretching and compressing along the fibers, we can in addition change their color intensity I_i:

$$I_i = I_0 \cdot \frac{1}{\lambda_i} \tag{29}$$

Another way to identify the regions of compression and expansion is to use a color code. Engineers are used to red for compression and green for tension. Therefore we use this color definition. We apply a continuous color mapping from red for the smallest negative eigenvalues, white for zero eigenvalues and green for positive eigenvalues. The definition of the different parameters for two dimensional field are summarized in Table 1, for three dimensions in 2.

4.4 Animation

According to Equation 14 we use the time variable t or the scaling factor σ defined by Equation 19 or 24 to generate an animation

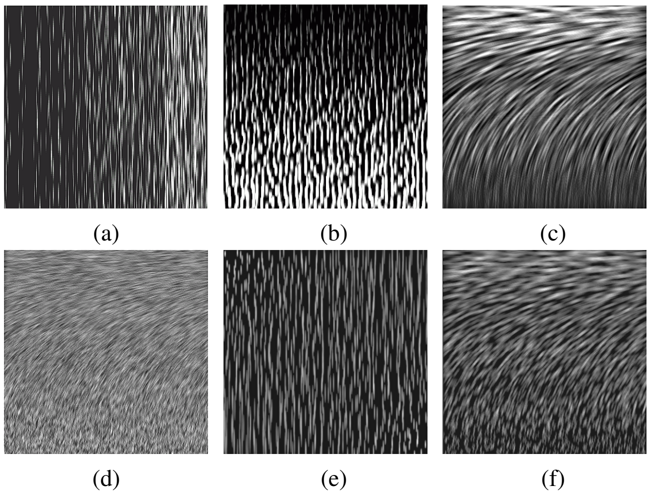


Figure 7: These images illustrate the effect of the change of different combinations of the image parameters on the LIC image of one eigenvector field for a simple synthetic tensor field. In (a)-(c) only the input image is changed corresponding to the eigenvalues of the orthogonal eigenvector field; (a) change of density; (b) change of spot size; (c) change of density and spot size. In images (d)-(f) we illustrate the effect of changing the convolution length. In images (d) and (e) the parameters of the input noise image are constant; (d) uses a white noise; (e) shows a sparse noise image. Image (f) shows a combination of the three parameters density, spot size, and convolution length.

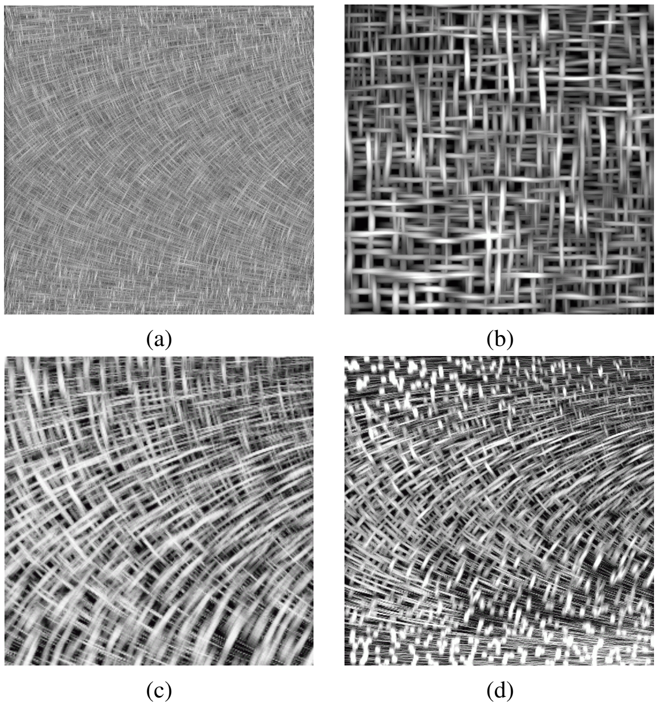


Figure 8: The images show the combination of two eigenvector field, each representing both eigenvalues; (a) is based on a white noise input image varying the convolution length. The other three images use sparse noise. In (b) and (c) density and spot size are changed. (d) shows a combination of the three parameters.

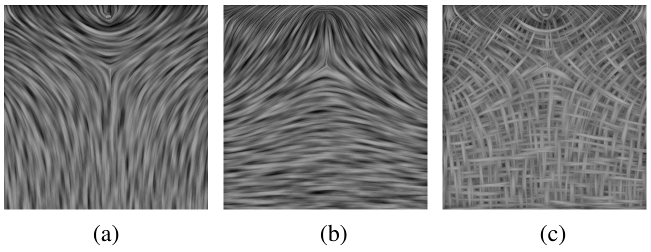


Figure 9: These images are the result of a single top-load data set, where the force is applied in z-direction. A yz-plane slice is shown; (a) and (b) illustrate the two eigenvector fields separately, in (c) they are overlaid. In all images, spot size and density are changed according to eigenvalues.

of the compression or expansion process. We start with an image where only the eigendirections and no eigenvalues are illustrated (pure LIC image with constant density, spot size, color intensity and convolution length). We can then scale the factor σ slowly to intensify the impression of expansion or compression, which are the most important features to capture the data.

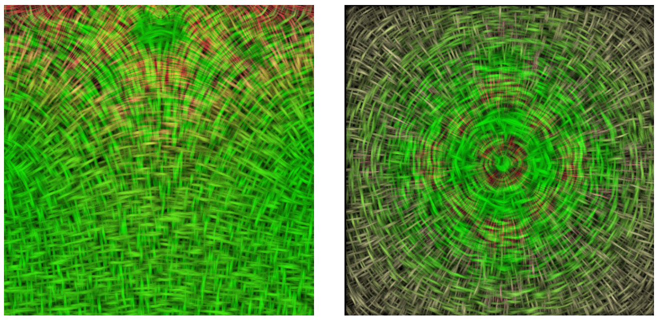
5 Results

We have evaluated our method using synthetic and real data sets. We started with a simple tensor field where the eigenvector fields are aligned to the coordinate axes. These examples allow us to validate the effect of the change of the texture parameters depending on the eigenvalues. In a second step, we used a datasets where we changed the eigenvector fields by rotating them by 90 degrees. Figure 7 shows some results for one eigenvector field for these datasets using different texture parameters. In the first three images (a) - (c) only the input texture is changed. In images (d) and (e) only the convolution length is changed. The last image (f) shows a result of changing all three texture parameters. Images for the same datasets showing both eigendirections are shown in Figure 8. Again, different input textures and parameter mappings were applied.

The next examples are results from numerical finite element simulations of the stress field applying different load combinations on a solid block. These datasets are well-studied and therefore appropriate to evaluate our method. For the simulation, a 10x10x10 gird, each cell consisting of $3^3 = 27$ Gauss-points had been used. The tensor field resulting from the simulation is continuous inside each cell, but not on cell boundaries. This fact can also be observed in our images. Figure 9 and Figure 10 show different slices of the three-dimensional dataset from a single point load. Figure 11, 12 and 13 represent a block with a couple of forces with opposite sign applied to the block. These images make possible a good visual segmentation of regions of compression and expansion. Red indicates compression, white reflects for no change, and green means expansion.

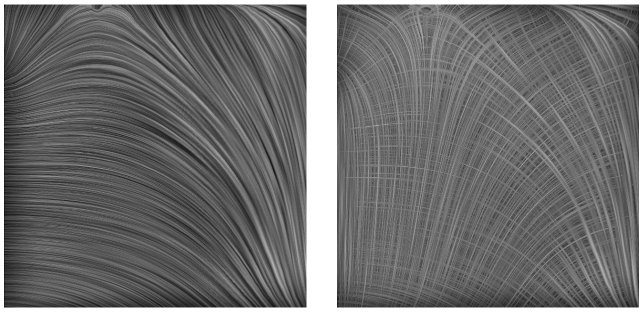
6 Conclusions and Future Work

We have discussed an intuitive method to visualize symmetric second order tensor fields with similar behavior as stress and strain fields. The interpretation of a tensor field as distortion of a flat metric results in a visualization based on the real physical effect of the tensor field. The distortion of the texture according to the metric supports a flexible representation of the tensor field, which is easy to understand. We evaluated our method for two-dimensional data sets and two-dimensional slices of three-dimensional tensor fields. The foundation for an extension to three dimensions now exists.



(a) (b)

Figure 10: This figure again shows a single-top-load. Spot size and density of the input images are adapted to the corresponding eigenvectors. Red shows regions of compression, green expansion according the respective eigenvector field: (a) shows a yz-plane slice parallel to the applied force; (b) shows a xy-plane slice orthogonal to the force.



(a) (b)

Figure 11: This image shows a yz-plane slice of a two-force data set: (a) shows the minor eigenvector field; (b) both eigenvector fields.

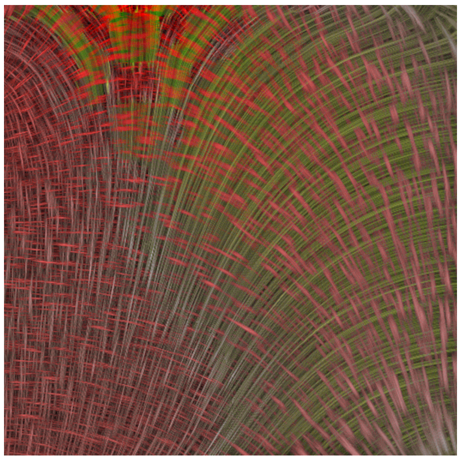


Figure 12: The image represents a yz-plane slice of a two-force dataset. In the lower-left corner we see a region of compression, a result mainly of the left pushing force; in the upper-right corner expansion dominates as a result of the right pulling force.

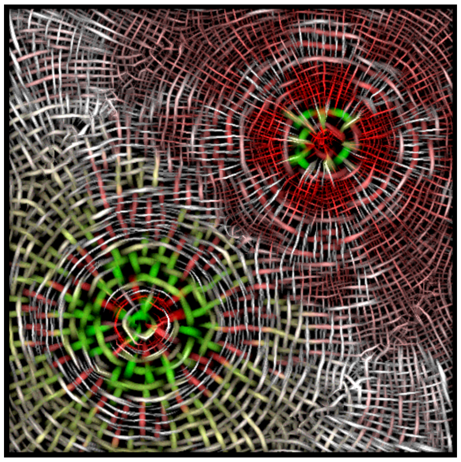


Figure 13: The image represents a xz-plane slice of a two-force dataset. The left circle corresponds to the pushing and the right to the pulling force. The fluctuation of the color is a result of the low resolution of the simulation.

		eigenvector field	
free parameters		$i = 1$	$i = 2$
density value	d_i	$\frac{1}{\lambda_2}$	$\frac{1}{\lambda_1}$
color intensity	I_i	$\frac{1}{\lambda_1}$	$\frac{1}{\lambda_2}$
convolution length	l_i	λ_1	λ_2
spot size r_i		λ_2	λ_1

Table 1: Assignment of eigenvalues to free parameters for a two-dimensional texture.

The transformation to a metric and the definition of the texture parameters has already been discussed in this paper. For visualization purpose we plan to use volume rendering methods, where we have opacity as an additional parameter. It will be used to enhance regions of interest like for example, regions of compression.

7 Acknowledgments

This work was supported by the National Science Foundation under contracts ACI 9624034 (CAREER Award) and ACI 0222909, through the Large Scientific and Software Data Set Visualization (LSSDSV) program under contract ACI 9982251, through the National Partnership for Advanced Computational Infrastructure (NPACI), and a large Information Technology Research (ITR)

		eigenvector field		
free parameters		$i = 1$	$i = 2$	$i = 3$
density value	$d_{i,j}$	$\frac{1}{\lambda_2}$	$\frac{1}{\lambda_1}$	$\frac{1}{\lambda_1}$
	$d_{i,k}$	$\frac{1}{\lambda_2}$	$\frac{1}{\lambda_1}$	$\frac{1}{\lambda_1}$
color intensity	I_i	$\frac{1}{\lambda_1}$	$\frac{1}{\lambda_2}$	$\frac{1}{\lambda_3}$
convolution length	l_i	λ_1	λ_2	λ_3
spot diameter	$r_{i,j}$	λ_2	λ_3	λ_1
	$r_{i,k}$	λ_3	λ_1	λ_2

Table 2: Assignment of eigenvalues to free parameters for three dimensions.

grant. We thank the members of the Visualization and Graphics Research Group at the Center for Image Processing and Integrated Computing (CIPIC) at the University of California, Davis. Further we thank the members of the Department of Civil and Environmental Engineering, University of California, Davis who provided us with the data sets.

REFERENCES

[1] Ed Boring and Alex Pang. Interactive deformations from tensor fields. In David Ebert, Hans Hagen, and Holly Rushmeier, editors, *IEEE Visualization '98*, pages 297–304. IEEE, 1998.

[2] Brian Cabral and Leith (Casey) Leedom. Imaging vector fields using line integral convolution. In James T. Kajiya, editor, *Computer Graphics (SIGGRAPH '93 Proceedings)*, volume 27, pages 263–272, August 1993.

[3] W. C. de Leeuw and J. J. van Wijk. A probe for local flow field visualization. In Gregory M. Nielson and Dan Bergeron, editors, *Proceedings of the Visualization '93 Conference*, pages 39–45, San Jose, CA, October 1993. IEEE Computer Society Press.

[4] T. Delmarcelle and L. Hesselink. The topology of second-order tensor fields. In R. Daniel Bergeron and Arie E. Kaufman, editors, *Proceedings of the Conference on Visualization*, pages 140–148, Los Alamitos, CA, USA, October 1994. IEEE Computer Society Press.

[5] Thierry Delmarcelle and Lambertus Hesselink. Visualizing second-order tensor fields with hyperstream lines. *IEEE Computer Graphics and Applications*, 13(4):25–33, July 1993.

[6] Robert B. Haber. Visualization techiques for engineering mechanics. *Computing Systems in Engineering*, 1:37–50, 1990.

[7] Lambertus Hesselink, Thierry Delmarcelle, and James L. Helman. Topology of second-order tensor fields. *Computers in Physics*, 9(3):304–311, May/June 1995.

[8] Lambertus Hesselink, Yuval Levy, and Yingmei Lavin. The topology of symmetric, second-order 3D tensor fields:. *IEEE Transactions on Visualization and Computer Graphics*, 3(1):1–11, January/March 1997.

[9] Ingrid Hotz. Isometric embedding by surface reconstruction from distances. In Robert Moorhead, Markus Gross, and Kenneth I. Joy, editors, *Proceedings of the 13th IEEE Visualization 2002 Conference (VIS-02)*, pages 251–258, Piscataway, NJ, October 27– November 1 2002. IEEE Computer Society.

[10] Ingrid Hotz. *Geometrische Algorithmen zur Visualisierung diskreter und kontinuierlicher Tensorfelder*. PhD thesis, University of Kaiserslautern, Germany, January 2003.

[11] B. Jeremic, Gerik Scheuermann, Jan Frey, Zhaohui Yang, Bernd Hamann, Kenneth I. Joy, and Hans Hagen. Tensor visualization in computational geomechanics. *International Journal for Numerical and Analytical Methods in Geomechanics*, 26:925–944, 2002.

[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*, 15(6):10–13, November 1995.

[13] Detlev Stalling and Hans-Christian Hege. Fast and resolution independent line integral convolution. In Robert Cook, editor, *SIGGRAPH 95 Conference Proceedings*, Annual Conference Series, pages 249–256. ACM SIGGRAPH, Addison Wesley, August 1995. held in Los Angeles, California, 06-11 August 1995.

[14] X. Tricoche, G. Scheuermann, and Hans Hagen. Tensor topology tracking: A visualization method for time-dependent 2D symmetric tensor fields. In A. Chalmers and T.-M. Rhyne, editors, *EG 2001 Proceedings*, volume 20(3) of *Computer Graphics Forum*, pages 461–470. Blackwell Publishing, 2001.

[15] David M. Weinstein, Gordon L. Kindlmann, and Eric C. Lundberg. Tensorlines: Advection-diffusion based propagation through diffusion tensor fields. In David Ebert, Markus Gross, and Bernd Hamann, editors, *IEEE Visualization '99*, pages 249–254, San Francisco, 1999. IEEE.

[16] Xiaoqiang Zheng and Alex Pang. Volume deformation for tensor visualization. In Robert Moorhead, Markus Gross, and Kenneth I. Joy, editors, *Proceedings of the 13th IEEE Visualization 2002 Conference (VIS-02)*, pages 379–386, Piscataway, NJ, October 27– November 1 2002. IEEE Computer Society.

[17] Xiaoqiang Zheng and Alex Pang. Hyperlic. In *IEEE Visualization '03 (Vis '03)*, pages ?–? IEEE, October 2003.

[18] Xiaoqiang Zheng and Alex Pang. Interaction of light and tensor field. In *Proceedings of VISSYM '03*, pages 157–166, 2003.

[19] Leonid Zhukov and Alan H. Barr. Oriented tensor reconstruction: Tracing neural pathways from diffusion tensor MRI. In Robert Moorhead, Markus Gross, and Kenneth I. Joy, editors, *Proceedings of the 13th IEEE Visualization 2002 Conference (VIS-02)*, pages 387–394, Piscataway, NJ, October 27– November 1 2002. IEEE Computer Society.