Tensor Field Visualisation using Adaptive Filtering of Noise Fields combined with Glyph Rendering

Andreas Sigfridsson* Tino Ebbers Einar Heiberg Lars Wigström

Department of Medicine and Care, Linköpings Universitet, Sweden

Abstract

While many methods exist for visualising scalar and vector data, visualisation of tensor data is still troublesome. We present a method for visualising second order tensors in three dimensions using a hybrid between direct volume rendering and glyph rendering.

An overview scalar field is created by using three-dimensional adaptive filtering of a scalar field containing noise. The filtering process is controlled by the tensor field to be visualised, creating patterns that characterise the tensor field. By combining direct volume rendering of the scalar field with standard glyph rendering methods for detailed tensor visualisation, a hybrid solution is created.

A combined volume and glyph renderer was implemented and tested with both synthetic tensors and strain-rate tensors from the human heart muscle, calculated from phase contrast magnetic resonance image data. A comprehensible result could be obtained, giving both an overview of the tensor field as well as detailed information on individual tensors.

CR Categories: I.3.3 [Computer Graphics]: Picture/Image Generation—Viewing algorithms; I.3.8 [Computer Graphics]: Applications; J.2 [Physical Sciences and Engineering]: Engineering; J.3 [Life and Medical Sciences]: Medical Information Systems;

Keywords: Tensor, Visualisation, Volume rendering, Glyph rendering, Hybrid rendering, Strain-rate

1 Introduction

Visualisation of tensor data has become even more important with the ability to acquire tensor metrics. This can be done with, for example, diffusion tensor magnetic resonance imaging (MRI) [1] or strain-rate calculation from phase-contrast MRI [18, 15, 16]. Add to that the various kinds of simulations that output a tensor field. This paper tries to address some of the problems that arise when visualising symmetric second order tensors in three dimensions.

1.1 Previous Work

One common way of visualising tensors is to represent the tensor with an icon or glyph. The glyph can then represent some or all degrees of freedom from the tensor, depending on it's type. When visualising tensor fields with one glyph per tensor, it is difficult to make out continuity, because two adjacent glyphs do not form a continuous shape. This gets even more difficult when a 3D tensor field is studied, where occlusion becomes a problem.

*Correspondence to: Andreas.Sigfridsson@imv.liu.se, Department of Medicine and Care, Clinical Physiology, Linköpings Universitet, SE-581 85 Linköping, SWEDEN

IEEE Visualization 2002 Oct. 27 - Nov. 1, 2002, Boston, MA, USA
0-7803-7498-3/02/$17.00 © 2002 IEEE

Methods for visualisation of tensor fields include various generalisations of streamlines to tensor data, such as hyperstreamlines [4]. Their success is highly dependent on starting points, or seeding points.

Line integral convolution (LIC), originally a method for visualising vector fields, was introduced by Cabral et al. [2] and further improved by Hege et al. [8]. It is a powerful method that shows direction (or, rather, tangent) of flow in a vector field. One of the advantages of the LIC algorithm is that it has a high spatial resolution. In contrast to the streamline family, no seeding points are needed; LIC visualises the vector field in every point, more or less. The key to the success of LIC is that it creates a *continuous* representation. To be able to do that, it assumes that the data itself is continuous, and exploits that fact. Mostly, though, physical data has this property, given sufficient resolution.

LIC is really designed for vector fields, and not tensor fields. Dickinson [5] suggested that tensors could be visualised as vector fields with the vector being the eigenvector corresponding to the largest eigenvalue. Hsu [9] extended this, suggesting two LIC images to be calculated, one for the largest eigenvalue and one for the next largest eigenvalue, where the image corresponding to the second largest eigenvalue uses a shorter filter length. The images are then added together to form the resulting image. Unfortunately, since the ratio between the eigenvalues is not taken into account, it fails to describe the degree of anisotropy of the tensor field, which is often very useful.

Related to LIC is spot noise, introduced by van Wijk [17] and later enhanced by de Leeuw and van Wijk [3]. It can be performed in 3D, but is not adapted to tensor visualisation without modifications.

Laidlaw et al. introduced brush strokes as an alternative [13]. This method can convey the degree of anisotropy, but it lacks continuity. Also, it is oriented towards 2D visualisation.

A volume rendering approach, designed for 3D diffusion tensors, which shares ideas with the work described in this paper, was proposed by Kindlmann et al. [10]. While being good at visualising diffusion tensors, it is unclear whether it would perform well on strain-rate tensors. It uses a reaction/diffusion method to create a scalar 3D texture, later applied to a shaded surface. This scalar texture has similarities to the scalar overview field we propose, though computed in a different way.

2 Methods

We suggest a hybrid visualisation method, consisting of volume rendering of a scalar field for overview visualisation, retaining continuity, and a glyph rendering for detailed tensor analysis with a much smaller region of interest (ROI), showing all degrees of freedom of the tensors while at the same time avoiding occlusion problems. A 3D cursor is then used to position the ROI where the glyph is rendered.

2.1 Scalar Overview Visualisation

For the overview visualisation, the tensor volume is decomposed into a scalar volume, retaining a few, but important, properties of the tensor volume. The scalar volume is created by iterated adaptive filtering of a noise texture, inspired by image enhancement using adaptive filtering, introduced by Knutsson et al. [7, 12]. The noise field is low pass filtered, thereby smearing the noise spots, and then filtered with directed high-pass filters in directions with weak eigenvalues, to cancel out the smearing, leaving spots smeared in strong eigendirections. The result is then visualised using volume rendering techniques, available in different flavours on different hardware for rapid rendering.

2.1.1 Filter Construction

First, one spherically symmetric low-pass filter, F_{lp}, is constructed. Various kinds of low-pass filters can be used, but we chose a Gaussian low pass filter, defined in the frequency domain as

$$F_{lp}(\rho) = e^{-\frac{\rho^2}{2\sigma^2}} \tag{1}$$

where ρ is the frequency (with $0 < \rho < \pi$) and σ^2 being the variance of the filter, here empirically set to $\sigma^2 = 0.09$.

Then, six spherically separable directional high-pass filters are defined. By standard convention, when $\mathbf{u}$ is the frequency, $\rho = |\mathbf{u}|$ and $\hat{\mathbf{u}} = \frac{\mathbf{u}}{|\mathbf{u}|}$. The k-th filter, F_k is defined by its radial component, $R(\rho)$, and directional component, $D_k(\hat{\mathbf{u}})$, respectively

$$F_k(\mathbf{u}) = R(\rho)D_k(\hat{\mathbf{u}}) \tag{2}$$

The directional part is defined by

$$D_k(\hat{\mathbf{u}}) = (\hat{\mathbf{n}}_k \cdot \hat{\mathbf{u}})^2 \tag{3}$$

where $\hat{\mathbf{n}}_k$ is the direction of the filter, defined as the directions of the vertices of an icosahedron:

$$\begin{cases} \hat{\mathbf{n}}_1 = c(a,0,b)^T & \hat{\mathbf{n}}_2 = c(-a,0,b)^T \\ \hat{\mathbf{n}}_3 = c(b,a,0)^T & \hat{\mathbf{n}}_4 = c(b,-a,0)^T \\ \hat{\mathbf{n}}_5 = c(0,b,a)^T & \hat{\mathbf{n}}_6 = c(0,b,-a)^T \end{cases} \tag{4}$$

with

$$\begin{aligned} a &= 2 \\ b &= (1+\sqrt{5}) \\ c &= (10+2\sqrt{5})^{-1/2} \end{aligned}$$

These six filters, when combined, are enough to create a filter with any direction, being more computationally efficient compared to creating a new filter for every voxel. This is explained by Knutsson et al. [7, 12] along with the parameter choices for a, b and c.

The radial part of the directed high-pass filters are defined as

$$R(\rho) = F_s(\rho) - F_{lp}(\rho) \tag{5}$$

with

$$F_s(\rho) = \begin{cases} 1 & 0 \le \rho < \rho_l \\ \cos^2 \frac{\pi(\rho-\rho_l)}{2(\rho_h-\rho_l)} & \rho_l \le \rho < \rho_h \\ 0 & \rho_h \le \rho \end{cases} \tag{6}$$

The F_s is a spherical filter that gives a smooth transition from 1 where $\rho < \rho_l$ to 0 where $\rho > \rho_h$ to ensure that the high-pass filter does not pass through any frequencies with $\rho > \rho_h$. We have empirically chosen $\rho_l = 0.7\pi$ and $\rho_h = \pi$.

These filters are created only once, and reused for each iterative step. Iso-surfaces of the seven filters are plotted in Figure 1.

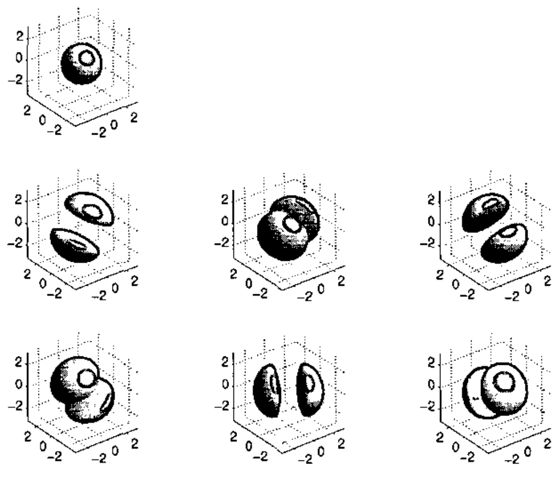


Figure 1: Iso-surface plots of the seven filters in the Fourier domain. The iso-value of the plots is 0.2. The top filter is the low-pass filter F_{lp}, the bottom six filters are the directional high-pass filters F_k.

2.1.2 Tensor Re-mapping

The tensors are re-mapped to enhance the visual experience. We want to exaggerate the tensor shape to create a more easily recognisable field. If the tensor is closest to being linear (one dominating eigenvalue), we exaggerate it to be even more linear, etc. This is accomplished by re-mapping the eigenvalues and then reconstructing the tensor from the eigenvalues and their respective eigenvectors.

First, the eigenvalues are scaled so that the largest eigenvalue, λ_1, becomes 1. The signs of the eigenvalues are discarded, so the scalar overview visualisation does not convey eigenvalue sign. The scaling discards the original norm of the tensors as well. Both of these are visualised using glyph rendering.

Then, the eigenvalues are mapped through the function

$$\mu(x;\alpha,\beta) = \frac{(|x|(1-\alpha))^\beta}{(|x|(1-\alpha))^\beta + (\alpha(1-|x|))^\beta} \tag{7}$$

which is plotted in Figure 2.

The parameters α and β control the shape of the re-mapping function. α is the x where μ crosses 0.5 and β controls the "sharpness" of the function. Higher β means a steeper curve, closer to a step function, lower β gives a function closer to the identity function. We use $\alpha = 0.5$ and $\beta = 2$.

The adaptive filtering described by Knutsson et al. [7, 12] is designed for orientation tensors, and will therefore smear *across* strong eigenvectors instead of *along* them. Therefore, we re-map the eigenvalues once more so that large eigenvalues become small, and small eigenvalues become large, using the mapping function

$$\nu(x;\alpha,\beta) = \frac{2}{1+\mu(x;\alpha,\beta)} - 1 \tag{8}$$

which is in turn plotted in Figure 3.

2.1.3 Iterative Filtering

The noise input field, which is a standard Gaussian noise field with a variance of 1, is filtered by multiplication with the filters in the Fourier domain, creating seven filter responses; one low-pass and

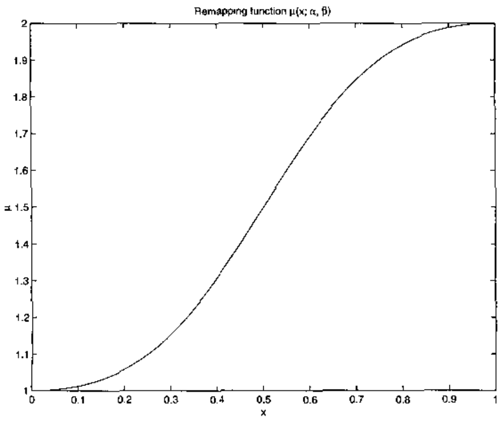


Figure 2: Re-mapping function, $\mu(x;\alpha,\beta)$, attracting eigenvalues to 0 or 1 to exaggerate the shape of the tensor. $\alpha = 0.5$, $\beta = 2$

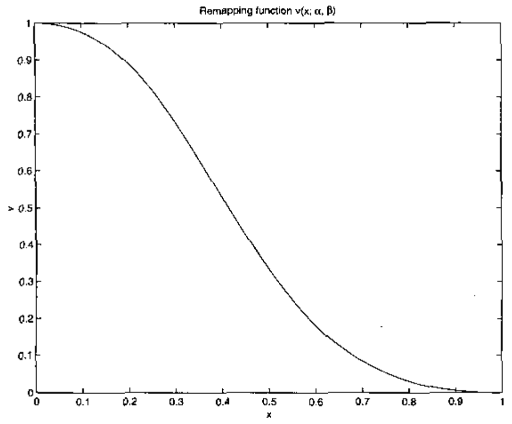


Figure 3: Combined re-mapping function, $\nu(x;\alpha,\beta)$, also mapping low eigenvalues to high and vice versa. $\alpha = 0.5$, $\beta = 2$

six directional high-pass responses. These responses are then recombined to a new image, used as the noise image for the next iteration.

The recombination is done by a simple point-by-point weighting of the filter responses

$$s' = s_{lp} + \sum_k c_k s_k \tag{9}$$

where s_{lp} is low-pass filter response and s_k is the filter response from the high-pass filter F_k.

The coefficients, c_k, are computed for each point as the product sum scalar tensor product

$$c_k = \mathbf{C} \cdot \mathbf{M}_k \tag{10}$$

where $\mathbf{C}$ is the tensor at that point and $\mathbf{M}_k$ are the filter-associated dual tensors, defined by

$$\mathbf{M}_k = \alpha \hat{\mathbf{n}}_k \hat{\mathbf{n}}_k^T - \beta \mathbf{I} \tag{11}$$

where $\mathbf{I}$ is the 3 by 3 identity matrix, $\alpha = \frac{5}{4}$ and $\beta = \frac{1}{4}$ in the three-dimensional case, as described by Knutsson et al. [7, 12]. The c_k are calculated only once and reused for every iteration.

To clarify the algorithm, a two-dimensional example iteration sequence is shown in Figure 4.

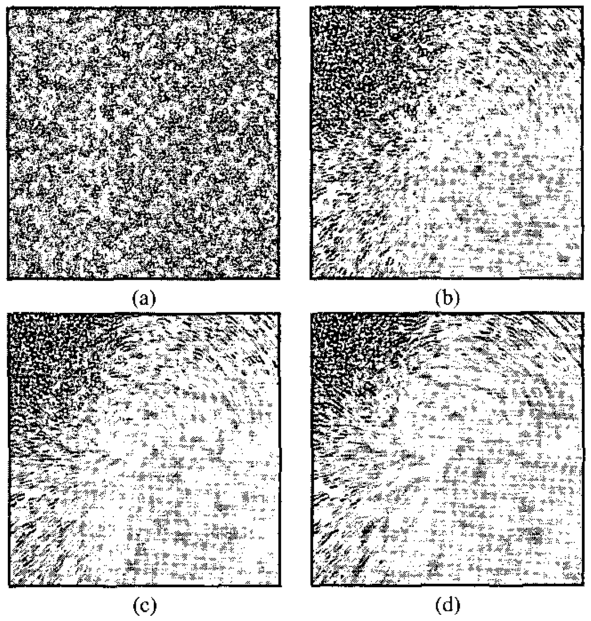


Figure 4: 2D example of the scalar field showing initial noise image (a), after 3 iterations of adaptive filtering (b), after 6 iterations (c), and after 12 iterations (d).

Since the seven filterings are independent, they were performed in parallel with good speedup using seven processors. Also, most of the filtering time is spent in the Fourier transforms. Since the multidimensional Fourier transforms are separable, they can too be parallelised, if enough processors are available.

With large volumes, the memory requirements for the filters and their respective responses are large. Memory requirements were cut in half by exploiting the fact that the filters are real and even in the Fourier domain and the noise field is real in the signal domain. The noise is therefore complex Hermitian in the Fourier domain. Consequently the filter responses are also complex Hermitian in the Fourier domain and thus real in the signal domain.

2.1.4 Eigenvalue Distribution Colour Coding

The scalar field obtained from the adaptive filtering can be further enhanced by adding colours. One way of adding colour is to encode the eigenvalue distribution, thus showing the degree of anisotropy of the tensor field. This is done with

$$R = \frac{\lambda_1 - \lambda_2}{\lambda_1} \tag{12}$$

$$G = \frac{\lambda_2 - \lambda_3}{\lambda_1} \tag{13}$$

$$B = \frac{\lambda_3}{\lambda_1} \tag{14}$$

R, G and B are the components of red, green and blue, respectively. λ_i are absolute values of the eigenvalues, sorted so that λ_1

is the largest. This gives the following properties for R, G and B:

$$0 < R < 1 \tag{15}$$
$$0 < G < 1 \tag{16}$$
$$0 < B < 1 \tag{17}$$
$$R + G + B = 1 \tag{18}$$

The R, G and B components then show the "probability" for the tensor being linear, planar or isotropic, respectively. This does not really add more information to the visualisation process, but it helps classification of the tensor. Furthermore, it aids depth perception when combined with interaction.

2.2 Glyph Tensor Visualisation

The most common way of visualising all degrees of freedom of a tensor is to use glyph rendering. One tensor is represented with a geometric object that can visualise all properties of the tensor.

Most glyph-based tensor field visualisation techniques show a glyph for every tensor in the field. In contrast to this, we suggest displaying only one or a few tensors, selected by interaction with a 3D cursor. Since we also provide an overview visualisation, not every tensor needs to be displayed with a glyph.

There are several different glyphs to consider, and some of them have certain constraints on the tensors to visualise, regarding, for instance, symmetry or sign of eigenvalues. The method suggested in this paper is invariant of the tensor glyph chosen.

For symmetric, real, positive semidefinite tensors (all eigenvalues are positive), ellipsoids provide an easy glyph. Let the eigenvectors be the principal axes and the eigenvalues be the radii of the ellipsoid. The different cases (linear, planar and isotropic) will be represented with appropriate shapes, shown in Figure 5.

Ellipsoids can also be drawn with wire-frame rendering, that is, only draw the edges of the polygons without filling the surface. This has the advantage of not obscuring the scalar volume when doing hybrid rendering. On the other hand, one has to sacrifice shading when drawing in wire-frame mode.

An alternative to ellipsoids is to use a line-based glyph. The tensor is simply represented with lines in the eigenvector directions with the length being proportional to the magnitude of the eigenvalue. The sign of the eigenvalue controls the colour of the line; a red line for positive eigenvalues and a blue line for negative eigenvalues. Sometimes, though, the line-based glyph can be hard to spot among the structure of the scalar overview field. Examples of line-based glyphs are also shown in Figure 5.

2.3 Combining Scalar Volume Rendering with Glyphs

Combining scalar volume rendering and glyph rendering imposes a problem; to display both glyphs and volume rendering at the same time. We solved this by using an OpenGL texture based volume renderer with a depth buffer and opaque glyphs. The opaque surfaces are first drawn in an arbitrary order, updating the depth buffer at every pixel. Then, the semi-transparent volume slices are drawn in a back-to-front order. This was implemented using both 2D or 3D textures in mono and stereo on a variety of platforms, ranging from consumer to high end hardware.

2.4 Obtaining Strain-Rate Tensors

One application for tensor visualisation is investigating the strain-rate (rate of deformation) in the heart musculature to study cardiac dysfunction and also to help understand the mechanics and physiology of the heart. We calculate the strain-rate tensors from phase-contrast MRI by first computing the velocity Jacobian $\mathbf{J}$ according to

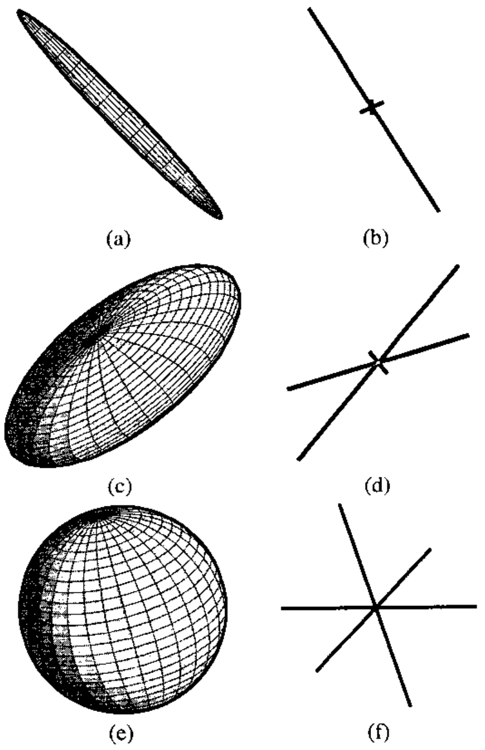


Figure 5: 3D glyph shapes shown for a linear tensor ($\lambda_1 \gg \lambda_2 \simeq \lambda_3$) (a, b), a planar tensor ($\lambda_1 \simeq \lambda_2 \gg \lambda_3$) (c, d) and an isotropic tensor ($\lambda_1 \simeq \lambda_2 \simeq \lambda_3$) (e, f). Figures (a, c, e) show ellipsoid glyphs, while (b, d, f) show line-based glyphs, showing positive eigenvalues in red and negative eigenvalues in blue.

$$\mathbf{J} = \begin{pmatrix} \frac{\partial u}{\partial x} & \frac{\partial v}{\partial x} & \frac{\partial w}{\partial x} \\ \frac{\partial u}{\partial y} & \frac{\partial v}{\partial y} & \frac{\partial w}{\partial y} \\ \frac{\partial u}{\partial z} & \frac{\partial v}{\partial z} & \frac{\partial w}{\partial z} \end{pmatrix} \tag{19}$$

where $(x\ y\ z)^T$ is the coordinate vector and $(u\ v\ w)^T$ is the velocity vector, obtained using 3D phase-contrast MRI [14, 19].

The rigid body rotation should not contribute to the deformation rate, so only the symmetric part needs to be considered. Thus, to obtain a strain-rate tensor $\mathbf{T}$, $\mathbf{J}$ is forced symmetric by

$$\mathbf{T} = \frac{1}{2}\left(\mathbf{J} + \mathbf{J}^T\right) \tag{20}$$

The velocity data used for strain-rate tensor calculation was acquired with sampling distances of 4 by 4 by 1.2 mm. At this resolution, the strain-rate tensor is not particularly continuous. The adaptive filter technique depends on highly continuous data with low frequency to perform adequately. To get that kind of data, it was low-pass filtered, reducing the accuracy of the tensors, and resampled to a higher resolution (0.375 mm in all three directions) using cubic interpolation. Only the tensor field used to create the scalar field is low-pass filtered; for glyph rendering, the original tensor field is used.

When performing low-pass filtering, or smoothing, one must be careful not to blend tensors from inside the myocardium (heart muscle) with tensors from blood inside the heart or the lungs outside. This is solved using normalised averaging [7], with a myocardial probability as the certainty function. We use a method suggested by Ebbers [6], which finds a probability function for tissue from time-resolved 3D phase contrast MRI data, by discarding voxels that have a high probability of being air or blood. After the normalised averaging, the data inside the myocardium is smooth and not contaminated with data from outside. The border, however, contains tensors that change rapidly.

3 Results

A combined volume and glyph renderer using OpenGL was implemented on a variety of platforms, using both 2D and 3D textures, in stereo where available. Where texture mapping was hardware accelerated, interactive frame rates (greater than 15 fps, depending on screen size) were obtained. It was found that stereo rendering and the interactivity improved the depth perception significantly. The images presented below were created using 3D textures on a SGI Onyx2 InfiniteReality running IRIX 6.5.

3.1 Synthetic Tensor Data

The renderer was tested on three synthetic tensor volumes. The visualisation of the volumes are shown in Figure 6.

Figures 6a and 6b show a field where all tensors are linear (one dominating eigenvalue). The eigenvalues were set as $\lambda_1 = 1, \lambda_2 = \lambda_3 = 0.1$. In Figure 6a, the direction of the eigenvector corresponding to the largest eigenvalue is radial, whereas it is cylindrical in Figure 6b.

Figure 6c shows a field where all tensors are planar (two dominating eigenvalues). The eigenvalues were set as $\lambda_1 = \lambda_2 = 1, \lambda_3 = 0.1$. The direction of the eigenvector corresponding to the *smallest* eigenvalue is radial. Note that this direction corresponds to the normal of the scalar field surface structure.

The synthetic volumes are 64^3 in size and the adaptive filtering was iterated 20 times.

The images from synthetic tensor data do not use eigenvalue colour coding, so they are in gray scale, and have been inverted for printing purposes. Note especially that the glyph resembles the local structure of the scalar field.

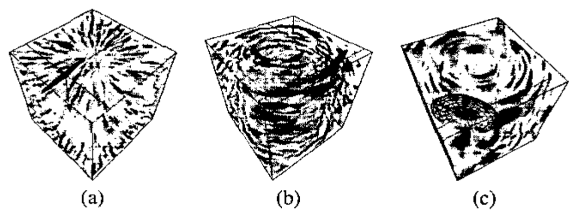


Figure 6: Synthetic tensor data visualisation showing a field with radial linear tensors (a), cylindrical linear tensors (b) and cylindrical planar tensors (c).

3.2 Human Strain-Rate Tensor Data

The method was also tested on strain-rate data from the heart of a healthy human volunteer. The volume was cut to only include the left ventricle and re-sampled to 256^3 in size. The scalar field was generated by 40 iterations of adaptive filtering and retrospectively masked so that only the tissue, as found using a threshold of a tissue probability function suggested by Ebbers [6], remained. The filtering time on a Sun Fire-6800 with 8 UltraSPARC-III CPUs of 900 MHz each was 10 minutes and 20 seconds.

Figure 7 shows the scene, without the glyph rendering and with high opacity to aid general orientation in the images. The apex (tip), base, left ventricle (LV) and right ventricle (RV) of the heart are marked. The cut plane is arranged in a short-axis manner.

Figure 8 shows a short-axis slab of the scalar data. The glyph is located in the left heart wall and the volume is viewed from two locations, to better convey the depth of the field. In Figure 8a, the heart wall of the left ventricle can clearly be seen (like a ring) and the tensor glyph is clearly planar, also shown with the green colour in the scalar field. In Figure 8b, the heart wall ring is seen from the side, showing the orientation of the planar glyph. Note that the shape of the scalar field in the neighbourhood of the glyph is shown as a planar green structure.

Figure 9 shows a long-axis slab, orthogonal to a short-axis slab. The left ventricle is shown from the side. Near and above the apex (tip), the chest can be seen. The glyph is located in the apex of the heart, and shown from two angles. Also here, the tensor is planar, conveyed by the glyph, the colour and the local structure of the scalar field.

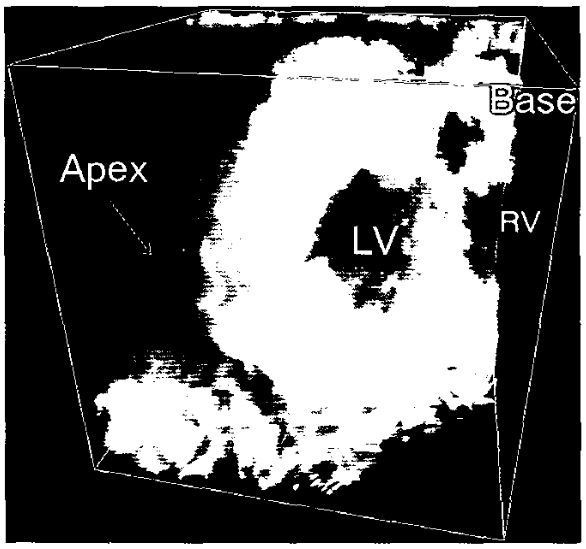
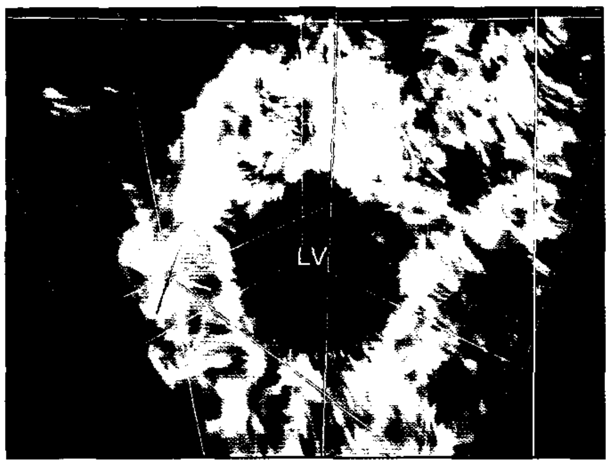
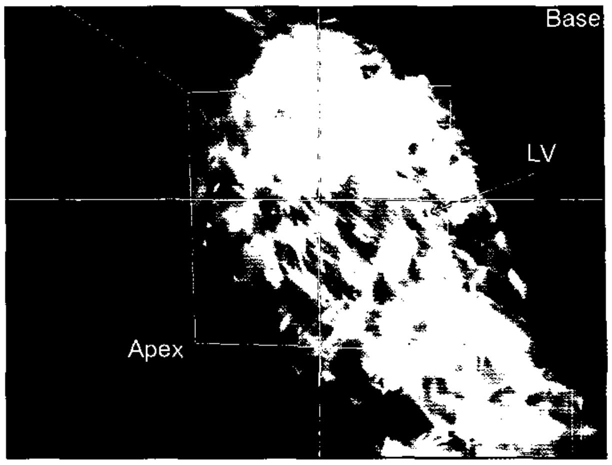


Figure 7: Strain-rate data, scene overview from posterior (back). The apex (tip), base, left ventricle (LV) and right ventricle (RV) of the heart are marked. The cut plane is arranged in a short-axis manner.

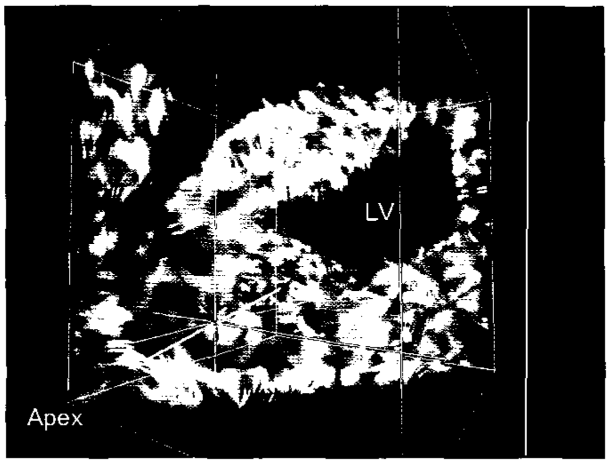


(a)

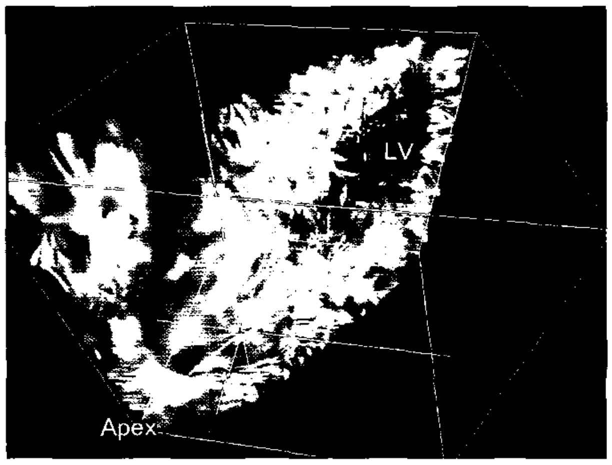


(b)

Figure 8: Myocardial strain-rate data, short-axis slab. The glyph, showing a planar tensor (two dominating eigenvalues) is located in the left heart wall, and the heart is viewed from its base (a), and from the side (b). Note that the shape of the scalar field in the neighbourhood of the glyph is shown as a planar structure.



(a)



(b)

Figure 9: Myocardial strain-rate data, long-axis slab. The glyph, showing a planar tensor (two dominating eigenvalues) is located in the apex. Note that the shape of the scalar field in the neighbourhood of the glyph is shown as a planar structure.

4 Discussion

We have presented a method for visualising second order tensor fields in three dimensions. The method combines adaptive filtered noise field for overview visualisation with standard glyph rendering for detail visualisation. The scalar field was visualised using a hybrid volume renderer, rendering both a semi-transparent scalar overview volume using 2D or 3D textures and the glyphs for detail visualisation using standard polygon and line geometry.

Many tensor visualisation techniques that are generalised from two-dimensional to three-dimensional have a problem with occlusion and the problem of continuity becomes even harder. Visualisation techniques generalised from vector to tensor fields usually have problems representing the degree of anisotropy of the tensor field.

The main advantage of this method is that it gives an accurate detail visualisation while still providing continuity, something that is lacking in most other techniques.

In the case of tensor imaging of the heart wall, it was possible to make the voxels not belonging to the heart wall totally transparent, using the tissue probability function suggested by Ebbers [6]. However, this is not generally possible for tensor imaging in other domains. Therefore, it should be investigated how to make parts of the volume (semi-)transparent. This could be done by reducing opacity in areas depending on anisotropy, for example, in isotropic areas, if that is of less interest. Another possibility is to control the opacity according to the variation of the tensor field, thus reducing the opacity in areas with low or high variation in the tensor field.

This method emphasises a lot on visualising the anisotropy of the tensor field, as opposed to visualising the magnitude. This might not always be desirable. The colour coding step of the method could be changed to show the magnitude of the tensor, instead of redundantly showing the degree of anisotropy. Perhaps a hue/intensity combination could show both. The tensor norm can also be used to modulate the opacity of the field.

4.1 Strain-Rate Adaption

The method described in this paper is aimed to visualise general symmetric second order tensors. For simplicity, the scalar field is generated without regard to the sign of the eigenvalues. It is application dependent whether this is a big disadvantage or not. For strain-rate tensors, the scalar overview is adequate, because it is just used as an overview, and the glyph rendering displays the sign of the eigenvalues.

The result could be improved further if strain-rate tensors are assumed. Strain-rate tensors are in a way a general symmetric second order tensor, and would mathematically encompass also positive semidefinite tensors, such as diffusion tensors. Taking the underlying physics into account, strain-rate tensors and diffusion tensors are quite different and, hence, ought to be visualised differently. Preferably, the visualisation of the tensors would mimic the underlying physics.

The sign of the eigenvalues can be added to the scalar field, if the eigenvalue re-mapping function is altered. This re-mapping could be done with a simple sigmoid, approaching 1 for large negative eigenvalues and 0 for large positive eigenvalues. This would cause areas with large positive eigenvalues to only have low pass content (large noise spheres, planes or lines) and areas with large *negative* eigenvalues to be high-pass filtered (small noise spheres, planes or lines). Eigenvalues close to zero would be re-mapped to 0.5, being somewhere in between. The resulting scalar field would then be as if the original noise function was medium size spheres, compressed or expanded in the directions of negative or positive eigenvalues, respectively.

Ellipsoid type glyphs can also be made to show the sign of the eigenvalues. Kirby et al. deformed unit circles according to the local strain-rate tensor in 2D [11]. The circle was expanded in directions of positive eigenvalues and compressed in directions of negative eigenvalues. Eigenvalues of zero neither compressed nor expanded the circle. When generalising this to 3D and only a single glyph, one needs to add a reference sphere, to show both the original unit sphere and the deformed ellipsoid to be able to see what is compression and what is expansion. With multiple glyphs in 2D this is not needed nor possible, because the deformed ellipses can be compared to each other and adding a reference circle to every ellipse would clutter the view. The amount of compression or expansion should be controlled by the magnitude of the eigenvalue, but how this should be done is not obvious. Kirby uses an exponential function to map eigenvalue to radius.

Preliminary tests of combining the strain-rate adapted scalar field with the reference/deformed sphere type glyph show promising results.

5 Conclusion

This paper presented a method for visualising second order tensor fields in three dimensions, using an adaptive filtered noise field for overview visualisation combined with standard glyph rendering for detail visualisation. The glyph and scalar field was shown to be congruent, meaning that the overview scalar field can be used to show some properties of the tensor. The scalar field was also shown to visualise both direction and degree of anisotropy.

Acknowledgements

This work has been supported by the Swedish Research Council and the Swedish Heart and Lung foundation. Special thanks to Anders Ynnerman, Matts Karlsson and Hans Knutsson for assistance in matters of scientific visualisation, biomechanics and multidimensional signal processing respectively.

References

[1] Peter J. Basser, James Mattiello, and Denis LeBihan. MR Diffusion Tensor Spectroscopy and Imaging. *Biophysical Journal*, 66:259–267, 1994.

[2] Brian Cabral and Leith Casey Leedom. Imaging Vector Fields Using Line Integral Convolution. *Computer Graphics*, 27(Annual Conference Series):263–270, 1993.

[3] Wim C. de Leeuw and Jack J. van Wijk. Enhanced Spot Noise for Vector Field Visualization. In *IEEE Visualization 95*, pages 233–239, 1995.

[4] Thierry Delmarcelle and Lambertus Hesselink. Visualization of Second Order Tensor Fields and Matrix Data. In *IEEE Visualization 92 Proceedings*, pages 316–323, 1992.

[5] R. R. Dickinson. A Unified Approach to the Design of Visualization Software for the Analysis of Field Problems. In *Three-dimensional Visualization and Display Technologies (Proceedings of SPIE)*, volume 1083, pages 173–180, 1989.

[6] Tino Ebbers. *Cardiovascular Fluid Dynamisc: Methods for Flow and Pressure Field Analysis from Magnetic Resonance Imaging*. PhD dissertation, Linköpings Universitet, Department of Biomedical Engineering, 2001. Dissertation No. 690, ISBN 91-7373-021-1.