Asymmetric Tensor Field Visualization for Surfaces

Guoning Chen, *Member, IEEE*, Darrel Palke, Zhongzang Lin, Harry Yeh, Paul Vincent, Robert S. Laramee, *Member, IEEE*, and Eugene Zhang, *Member, IEEE*

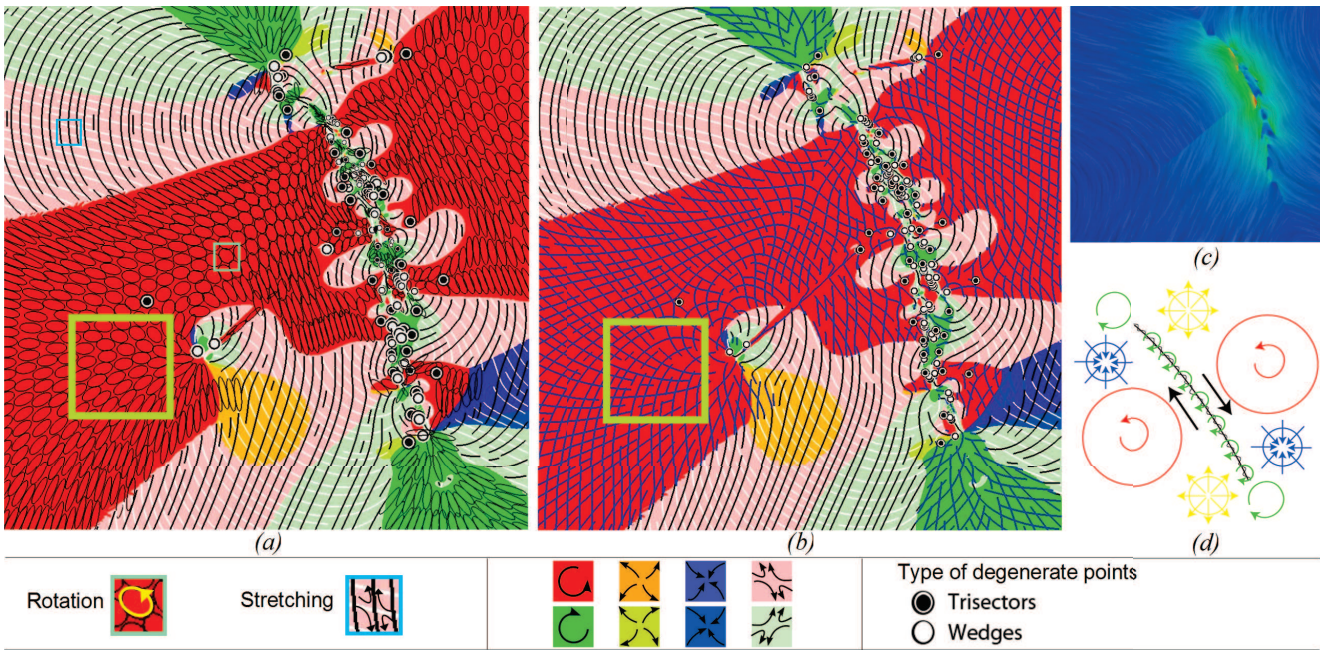


Fig. 1. Visualization of the ground deformation associated with a simulation of the June, 1992 Mw=7.3 Landers, CA earthquake. (a) new hybrid visualization of the displacement-gradient tensor field ($\alpha = 0.022$, $\beta = 0.8$, $iter = 800$, $m_r = 10$), (b) previous visualization using hyperstreamlines only, (c) a common visualization method used in earthquake deformation studies showing the displacement vector field only, (d) the expected deformation modes for a right-lateral fault. Note how the glyph packing in the complex domains (see the highlighted region) better conveys the elliptical deformation pattern than the previous methods (e.g. (b) and (c)).

Abstract—Asymmetric tensor field visualization can provide important insight into fluid flows and solid deformations. Existing techniques for asymmetric tensor fields focus on the analysis, and simply use evenly-spaced hyperstreamlines on surfaces following eigenvectors and dual-eigenvectors in the tensor field. In this paper, we describe a hybrid visualization technique in which hyperstreamlines and elliptical glyphs are used in real and complex domains, respectively. This enables a more faithful representation of flow behaviors inside complex domains. In addition, we encode tensor magnitude, an important quantity in tensor field analysis, using the density of hyperstreamlines and sizes of glyphs. This allows colors to be used to encode other important tensor quantities. To facilitate quick visual exploration of the data from different viewpoints and at different resolutions, we employ an efficient image-space approach in which hyperstreamlines and glyphs are generated quickly in the image plane. The combination of these techniques leads to an efficient tensor field visualization system for domain scientists. We demonstrate the effectiveness of our visualization technique through applications to complex simulated engine fluid flow and earthquake deformation data. Feedback from domain expert scientists, who are also co-authors, is provided.

Index Terms—Aasymmetric tensor fields, vector fields, glyph packing, hyperstreamline placement, view-dependent.

1 Introduction

Asymmetric tensor fields appear in a wide range of applications such as solid and fluid mechanics, structural engineering, and medical imaging. In these applications, the asymmetric tensor fields often appear in the form of the spatial gradient of a vector field, such as the velocity vector field in fluid dynamics or the deformation vector field in solid mechanics. The velocity gradient tensor field describes all of the non-translational kinematics in fluid parcels such as rotations, stretchings, and volume changes which cannot be easily inferred from direct visualization of the velocity vector field [43]. Consequently, effective visualization techniques for asymmetric tensor fields can potentially benefit many applications. In this work, we focus on the visualization of asymmetric tensor fields defined on a two-dimensional manifold (plane, surfaces). 2D asymmetric tensor field visualization is useful for cases where only two-dimensional data is available, such as data taken by the Particle Imagery Velocimetry (PIV), satellite remote sensing data on the water surface, and earthquake data. Moreover, as in the case of vector field and symmetric tensor field research, two-dimensional visualization often provides key insight into the more

- *Guoning Chen is with SCI, University of Utah, E-mail: chengu@sci.utah.edu.*
- *Darrel Palke, Zhongzang Lin, Harry Yeh, Paul Vincent, and Eugene Zhang are with Oregon State University. E-mail: {darrel.palke, linzhongzang}@gmail.com, harry@engr.orst.edu, pvincent@coas.oregonstate.edu, zhange@eecs.oregonstate.edu.*
- *Robert S. Laramee is with Swansea University, E-mail: R.S.Laramee@swansea.ac.uk.*

Manuscript received 31 March 2011; accepted 1 August 2011; posted online 23 October 2011; mailed on 14 October 2011.
For information on obtaining reprints of this article, please send email to: tvcg@computer.org.

challenging 3D analysis and visualization.

Despite the potentials of asymmetric tensor field visualization, there has been relatively little work in this area. Most existing tensor field visualization techniques focus on *symmetric* tensors and use either glyphs or hyperstreamlines following the major or minor eigenvectors of the tensor field. Due to fundamental differences between symmetric and *asymmetric* tensors, these techniques cannot be easily adapted to the visualization of the latter. For example, symmetric tensors always have real eigenvalues while asymmetric tensors can have *complex* eigenvalues. Furthermore, the major and minor eigenvectors of an asymmetric tensor with real eigenvalues need not be perpendicular.

To address this challenge, Zheng and Pang [46] introduce the concept of *dual-eigenvectors* which provide directional information of a 2D asymmetric tensor field inside complex domains (i.e., complex eigenvalues). Hyperstreamlines are used to follow the major dual-eigenvectors in the complex domains and either the major or minor eigenvectors in the real domains. Zhang et al. [43] extend this analysis by introducing the concepts of *eigenvalue manifold* and *eigenvector manifold*. With these manifolds, they develop analysis of 2D asymmetric tensor fields providing physical interpretation of the velocity gradient tensor field. Based on this analysis, the flow motion is dominated by anisotropic stretching and rotations in the real and complex domains, respectively. Zhang et al. also point out that the *elliptical* tensor patterns in the complex domains cannot be easily inferred from hyperstreamlines following the dual-eigenvectors since the eccentricity information is missing. To overcome this difficulty, they propose the concept of *pseudo-eigenvectors*. By intersecting evenly-spaced hyperstreamlines that follow either the major or minor pseudo-eigenvectors, one obtains diamond-shaped regions whose smallest enclosing ellipses reflect the tensor patterns (Figure 1(b)). Finally, Zhang et al. use colors to encode the tensor magnitude.

While the work by Zheng and Pang [46] and Zhang et al. [43] has advanced our understanding of asymmetric tensors, we have found that the provided visualization techniques are not adequate for domain experts from fluid mechanics and geophysics as pointed out by domain experts Harry Yeh and Paul Vincent (co-authors of this paper). The existing visualization of the asymmetric tensor field in complex domains using hyperstreamlines requires the user to *infer* the elliptical shapes using the intersection of the hyperstreamlines. This intersection often does not form the shape of a diamond due to the difficulty in achieving perfectly evenly-spaced hyperstreamlines (see the region highlighted by the yellow square in Figure 1(b)). This can cause difficulty in the physical interpretation. For instance, the domain expert may not be able to recognize the deformation pattern is rotation dominant because their appearance is similar to those in the real domain. Furthermore, the tensor magnitude, a quantity of great importance to the physical interpretation has not been included in the previous visualization techniques for asymmetric tensor fields. This means that the tensor magnitude has to be visualized using additional images, a situation domain scientists wish to eliminate. Finally, efficient visual exploration of the data sets requires the ability to inspect the data from any viewpoint and at any level of detail, with quick feedback. Tracing hyperstreamlines with controlled, possibly non-uniform spacing on surfaces is a computationally expensive task as it requires extensive geodesic distance computation. Compounding this with the fact that hyperstreamlines need to be regenerated every time the user changes the level of detail makes it computationally prohibitive to perform visual exploration of tensor fields on surfaces.

This work addresses these challenges and presents an intuitive visualization in the context of solid and fluid mechanics as well as other applications. Specifically, we make the following contributions: First, we adapt the technique of glyph packing (represented as ellipses) to the visualization of elliptical patterns in the complex domains. An elliptical shaped glyph can provide more explicit visualization for rotation dominant motion with the direction of stretching, and highlight the distinct dynamic behaviors within real and complex domains better than previous method using different colors [43] (see Figure 1 (a) and (b)). Second, we encode the tensor magnitude into the size of the glyphs and the density of the hyperstreamlines. Specifically, we use denser hyperstreamlines and glyphs with smaller sizes (i.e. denser packing since the density of glyphs is inversely proportional to glyph size) to represent larger tensor magnitude. Even though tensor glyphs typically indicate higher tensor magnitude with larger glyphs [17], we feel an inverse magnitude representation is more consistent with that used in flow vector fields, whereby the denser packing of streamlines and the vortex lines represents larger discharge rates and vorticity, respectively [35, 27]. Encoding the tensor magnitude in our hybrid visualization allows for the tensor patterns, magnitude, and eigenvalue analysis to be presented in the same image, which facilitates the visual exploration and physical interpretation of the data. While this approach has the advantage of including all deformation modes visualized in one plot, we recognize that users may prefer a different, perhaps separate image to display the tensor magnitude. As such, we are exploring different ways to display the tensor magnitude both in the same plot as well as an option to toggle on and off a separate co-registered image of tensor magnitude. Third, we present an efficient image-space approach for our hybrid glyph and hyperstreamline technique. By carefully projecting the tensor field onto the image plane, we reduce the run time for glyph packing and hyperstreamline placement on surfaces with complex geometry, such as the cooling jacket, from hours using an object-space approach to seconds; a two orders of magnitude speed increase. The combination of these techniques provides an interactive visualization tool for domain experts to examine their asymmetric tensor fields. The system provides users the ability to quickly explore their data using hyperstreamlines, which is fast and can provide the global configuration information of the data, followed by the hybrid visualization when the user is satisfied with the current view point. Finally, we apply our hybrid asymmetric tensor field visualization to both flow visualization and earthquake simulations. The latter represents a new application of asymmetric tensor field visualization, from earthquake fault mechanics. Interpretations from domain experts Yeh and Vincent are provided.

The rest of the paper is organized as follows: Section 2 reviews previous work related to this paper. Section 3 provides a brief description of basic concepts associated with asymmetric tensor field analysis. The pipeline of our hybrid visualization framework is presented in Section 4. Section 5 provides details of our modified glyph packing in complex domains. An image-space implementation is described in Section 6. In Section 7 we present visualization results of simulated fluid flow and deformation data sets as well as physical interpretations and analysis by domain experts. Section 8 summarizes this work.

2 Related Work

There has been much work in the area of vector field visualization though covering this work is beyond the scope of this paper. We refer interested readers to the following surveys for comprehensive reviews of these techniques [21, 22, 27]. Some of these techniques have been adapted to second-order symmetric tensor fields. In contrast, there has been relatively little work in the visualization of asymmetric tensor fields.

Symmetric Tensor Field Visualization Symmetric tensor field analysis and visualization have been well researched for both two and three dimensions. For the purpose of this paper, we will only refer to the most relevant work. Delmarcelle and Hesselink [7] provide a comprehensive study on the topology of 2D symmetric tensor fields and define hyperstreamlines, which they use to visualize tensor fields. This research is later extended to analysis in three dimensions [10, 45, 47] and topological tracking in time-varying symmetric tensor fields [36]. Zheng and Pang provide a high-quality texture-based tensor field visualization technique, HyperLIC [44], which adapts the idea of Line Integral Convolution (LIC) [4] to symmetric tensor fields. Hotz et al. [13] present a texture-based method for visualizing 2D symmetric tensor fields.

Evenly-spaced streamlines have been used to visualize vector fields [5, 15, 24, 25, 28, 37, 39, 41]. Spencer et al. [35] improve the efficiency of the streamline placement on surfaces via an image-space approach. Rosanwo et al. [31] propose the dual streamline seeding to avoid expensive geodesic computation on surfaces. The idea of using evenly-spaced streamlines has been extended to tensor fields [1, 9, 26, 42] and more generally N-way rotational symmetry

(*N-RoSy*) fields [29, 30]. McLoughlin et al. [27] provide an overview of seeding strategies.

Laidlaw et al. [19] stochastically place glyphs to minimize overlap when generating multi-layered diffusion tensor visualization. A similar glyph placement technique is introduced in the work of Kirby et al. [18] in which glyphs represent certain vector and tensor attributes of complex flow fields. The tensor splat method is proposed to convert tensor values into tuned Gabor functions which are encoded into 2D and 3D textures [2, 3]. Reaction-diffusion equations have been adapted by Kindlmann for tensor visualization [16] which are extended to the work on glyph packing [17]. In this study, a tensor-based potential energy is defined to derive the placement of a system of particles whose final positions will be used to place glyphs. Hlawitschka et al. [12] present an alternative glyph packing using Delaunay triangulation which successfully reduces the computation cost. A similar technique that uses the dual of Delaunay triangulation, i.e. Voronoi tessellation is also proposed by Feng et al. [8] to achieve better convergence of glyph packing. Recently, Schultz and Kindlmann [33] introduce the superquadric glyphs that can be used to visualize the general symmetric second order tensors that could be non-positive-definite. Ellipsoid packing has also been applied to generate anisotropic meshes [34].

In the flow visualization community, De Leeuw and Van Wijk [6] visualize the local properties of a flow field by visualizing its gradient (Jacobian). They decompose the Jacobian matrix into symmetrical and anti-symmetrical parts which are transformed to local frame and further decomposed to different components representing acceleration, shear, curvature, torsion, and convergence, respectively. These components can then be mapped to different geometric primitives for visualization. This is a relevant work. However, we apply a different tensor decomposition in the present work.

Some past work has placed glyphs along hyperstreamlines such as the work by Hlawitschka and Scheuermann on higher-order tensor field analysis [11]. While such work also uses both primitives, it is fundamentally different from our work since they are not placed in complementary regions as in our case. It is not clear how to extend their work to visualize asymmetric tensor fields in a straightforward fashion. To our knowledge, our algorithm is the first to apply glyph packing to visualize asymmetric tensor fields.

Asymmetric Tensor Field Visualization Hyperstreamline-based techniques are employed for asymmetric tensor fields by Zheng and Pang [46]. They present the concept of dual-eigenvectors for the complex domains where eigenvalues and eigenvectors are complex. Their visualization consists of hyperstreamlines following major and minor eigenvectors in real domains and major dual-eigenvectors in complex domains. Zhang et al. [43] extend this visualization and provide physical interpretation in the context of flow visualization, when the asymmetric tensor is the velocity gradient tensor. They introduce the idea of pseudo-eigenvectors, which are used to better illustrate the flow patterns in the complex domains. They also define the concept of eigenvalue and eigenvector manifolds used in tensor field analysis. However, the visualization of the asymmetric tensor fields using hyperstreamlines computed in object space is prohibitively expensive for the users. This work builds upon the analysis of Zhang et al. [43] and focuses on the efficient visualization of the analysis result. This has led to a thorough and interactive visualization system for asymmetric tensor fields which can now be applied to the applications of fluid dynamics and earthquake engineering. We refer interested readers to the accompanying video for a demonstration of our visualization system.

3 Background

In this section we review the relevant background on asymmetric tensor fields, based on [43, 46]. A *second-order* tensor $\mathbf{T}$ can be represented by an $N \times N$ matrix T_{ij} where N is the dimension of the tensor. $\mathbf{T}$ is *symmetric* when $T_{ij} = T_{ji}$ or *anti-symmetric* when $T_{ij} = -T_{ji}$. The *trace* of $\mathbf{T}$ is defined as $\sum_{1 \le i \le N} T_{ii}$. $\mathbf{T}$ is *traceless* when the trace of $\mathbf{T}$ is zero. Note that for any anti-symmetric tensor $\mathbf{T}$ $T_{ii} = 0$ for $1 \le i \le N$.

Any second-order tensor $\mathbf{T}$ can be *uniquely* decomposed as follows:

$$\mathbf{T} = \mathbf{D} + \mathbf{S} + \mathbf{R} \tag{1}$$

where $\mathbf{D}$ is a multiple of the identity matrix, $\mathbf{S}$ is a symmetric and traceless matrix, and $\mathbf{R}$ is an anti-symmetric matrix. When $\mathbf{T}$ is the velocity gradient tensor, $\mathbf{D}$, $\mathbf{S}$, and $\mathbf{R}$ represent the time rate of volume change, angular deformation, and rotation, respectively. Similarly for the deformation gradient tensor in solid mechanics, $\mathbf{D}$, $\mathbf{S}$, and $\mathbf{R}$ represent dilation or contraction, angular shear, and rotation, respectively.

In this paper, we will focus on two-dimensional asymmetric tensors, i.e., $N = 2$. For this case, Equation 1 can be rewritten as

$$\mathbf{T} = \gamma_d I + \gamma_s \begin{pmatrix} \cos\theta & \sin\theta \\ \sin\theta & -\cos\theta \end{pmatrix} + \gamma_r \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix} \tag{2}$$

where $\gamma_d = \frac{T_{11}+T_{22}}{2}$, $\gamma_s = \frac{\sqrt{(T_{11}-T_{22})^2+(T_{12}+T_{21})^2}}{2}$, and $\gamma_r = \frac{T_{21}-T_{12}}{2}$ are the strengths of $\mathbf{D}$, $\mathbf{S}$, and $\mathbf{R}$, respectively, while θ encodes the directions of angular deformation.

Because the eigenvector and dual-eigenvector information is not dependent on γ_d, we can focus on traceless (deviatoric) tensors. Such tensors can be parameterized as follows:

$$T(\rho,\theta,\varphi) = \rho\cos\varphi \begin{pmatrix} \cos\theta & \sin\theta \\ \sin\theta & -\cos\theta \end{pmatrix} + \rho\sin\varphi \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix} \tag{3}$$

Notice that the above form is a special case of Equation 2 in which $\gamma_d = 0$, $\rho = \sqrt{\gamma_s^2 + \gamma_r^2}$ and $\varphi = tan^{-1}(\frac{\gamma_r}{\gamma_s}) \in [-\frac{\pi}{2}, \frac{\pi}{2}]$. The eigenvalues of $T(\rho,\theta,\varphi)$ are:

$$E_{1,2} = \begin{cases} \pm\rho\sqrt{\cos 2\varphi} & \text{if } 0 \le |\varphi| \le \frac{\pi}{4} \\ \pm\rho\sqrt{-\cos 2\varphi}\,\mathbf{i} & \text{if } \frac{\pi}{4} < |\varphi| \le \frac{\pi}{2} \end{cases} \tag{4}$$

where $\mathbf{i}$ satisfy $\mathbf{i}^2 = -1$. A *tensor field* $\mathbf{T}(\mathbf{p})$ is a tensor-valued function defined on an N-dimensional manifold D ($N = 2$ in our case). We consider the map $\tau : D \to S^2$ defined by $\tau : \mathbf{p} \mapsto (\theta_\mathbf{p}, \varphi_\mathbf{p})$ where $\theta_\mathbf{p}$ and $\varphi_\mathbf{p}$ are the parameters corresponding to $T(\mathbf{p})$. The S^2 is the so-called *eigenvector manifold* [43] for which the North and South Poles ($\varphi = \pi/2$ and $-\pi/2$, respectively) correspond to pure rotations. The latitude circles $\varphi = \pm\frac{\pi}{4}$ represent tensors with equal real eigenvalues, and they form the boundaries of tensors with real eigenvalues and with complex eigenvalues. The pre-image of τ of these tensors are referred to as the *degenerate curves* [46], which divide the domain into *real domains* ($\tau^{-1}(\{(\theta,\varphi) : 0 \le |\varphi| < \frac{\pi}{4}\})$) and *complex domains* ($\tau^{-1}(\{(\theta,\varphi) : \frac{\pi}{4} < |\varphi| \le \frac{\pi}{2}\})$).

In the context of fluids, i.e., when the tensor field is the gradient of the velocity vector field, the following interpretation is applicable [43]. In the real domains, a linearized local flow pattern at a point resembles a distorted hyperbola (see Figure 1 the stretching illustration in the bottom row). The (real) eigenvectors indicate the direction of stretching an compression of fluid parcels. In the complex domains, linearized local flow patterns are elliptical whose eccentricity is given by $e = \sqrt{\frac{2\sin(2|\varphi|)}{1+\sin(2|\varphi|)}}$ for $\frac{\pi}{4} < |\varphi| \le \frac{\pi}{2}$. The major and minor axes of the ellipses are given by the *dual-eigenvectors* of the tensor, which are the major and minor eigenvectors of the following symmetric matrix:

$$S(T) = \frac{\sin\varphi}{|\sin\varphi|}\cos\varphi \begin{pmatrix} \cos(\theta+\frac{\pi}{2}) & \sin(\theta+\frac{\pi}{2}) \\ \sin(\theta+\frac{\pi}{2}) & -\cos(\theta+\frac{\pi}{2}) \end{pmatrix} \tag{5}$$

Notice the dual-eigenvectors are not well-defined along the *Equator* ($\varphi = 0$), pure symmetric tensors, and at the *Poles* ($\varphi = \pm\pi/2$), *degenerate points* in the eigenvector manifold. The set of degenerate points in an asymmetric tensor field has a one-to-one correspondence with the set of degenerate point of the symmetric tensor of Equation 5 [43].

4 Hybrid Asymmetric Tensor Field Visualization

The input to our visualization method is a triangular mesh and an asymmetric tensor field defined on the vertices of the mesh. The spatial gradient of a vector field is an example of the asymmetric tensor fields.

The first step of our visualization pipeline is to perform tensor field analysis [43], which computes the eigenvectors and dual-eigenvectors,

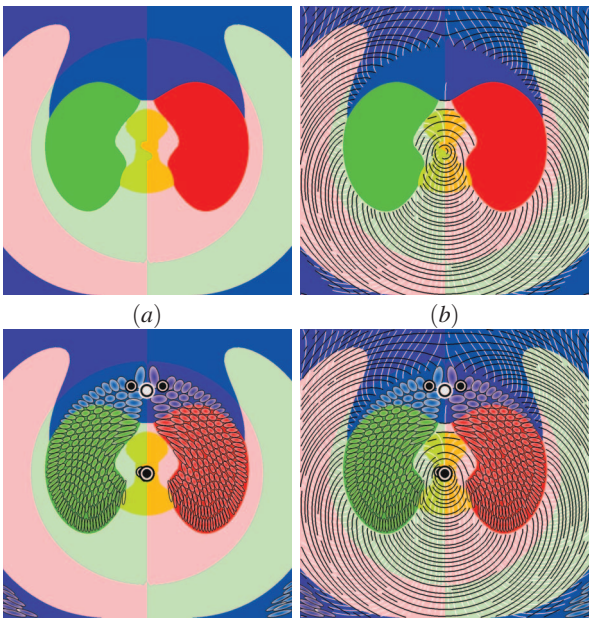


Fig. 2. This figure illustrates the constituents of our hybrid visualization using a synthetic vector field: (a) the color coding used for the background of the visualization which is based on the eigenvalue and eigenvector manifolds [43]. (b) the hyperstreamlines in the real domains, (c) the glyphs in the complex domains. (d) shows the final visualization.

tensor magnitude, and the strengths of rotation, volume change, and anisotropic stretching. Real and complex domain classification is performed, and degenerate points are also extracted at this stage. In the second step, we produce three intermediate constituents needed by the final visualization: (a) colors encoding the eigen-analysis [43], (b) hyperstreamlines following the major and minor eigenvectors in the real domains, and (c) glyphs showing the elliptical tensor patterns in the complex domains. All three constituents are then integrated in the same image (Figure 2(d)). The color coding for the eigenvalue analysis (Figure 2(a)) is adopted from [43], which we briefly review here. Regions dominated by expansion (positive volume change), contraction (negative volume change), anisotropic stretching, counterclockwise rotation, and clockwise rotation are colored with yellow, blue, white, red, and green, respectively. For expansion, contraction, and anisotropic stretching, we blend the colors with red or green to also indicate the orientation of rotations in those regions (red: counterclockwise; green: clockwise) even though rotation is not the dominant motion in these regions.

As the analysis of the input tensor field is handled using [43], we now turn to the discussion of hyperstreamline placement in the real domains and glyph packing in the complex domains, respectively. There are a number of parameters we need for these two processes: 1) α: a global scaling value to control the sizes of the glyphs and densities of the hyperstreamlines; 2) β: the ratio of the total glyph area to the area of a complex domain; 3) *iter*: the number of iterations for the glyph packing; 4) m_r: user specified maximum ratio of the maximum tensor magnitude to the minimum tensor magnitude.

Hyperstreamline placement Our hyperstreamline placement adapts the evenly-spaced streamline placement method of Jobard and Lefer [15] to asymmetric tensor fields. Similar to [15], our method traces hyperstreamlines from a set of seed points following the major and minor eigenvectors, respectively. For each seed, a hyperstreamline is generated and additional seeds along the hyperstreamline are added. The tracing of a hyperstreamline stops when it approaches a degenerate point, hits the boundary of the mesh or a region, gets too close to an existing hyperstreamline including itself, or has exceeded a maximal length. To reflect the tensor magnitude, we incorporate this information in the density of the hyperstreamlines. Specifically, during the placement of a hyperstreamline, we reject the current integration point p if its distance to a sample s on an existing hyperstreamline is smaller than $k\alpha$ where k is a scaling factor determined by the tensor magnitude which will be described in Section 5.4. Note that $k \equiv 1$ returns evenly-spaced hyperstreamlines. Similarly, we use k to determine the positions of the new seeds from the latest hyperstreamline. For a point p on the new hyperstreamline, we place two new seeds in directions perpendicular to the hyperstreamline direction with a distance of $k\alpha$ away from p. This placement ensures that potential seeds are not immediately rejected for being too close to an existing hyperstreamline.

We now turn to the description of our algorithm for glyph packing in the complex domains.

5 Glyph Packing In Complex Domains

Our glyph packing technique extends the work of Kindlmann and Westin [17] by incorporating degenerate points into the packing process as well as taking into account the boundaries of the disjoint complex regions (Figure 2(c)). We first briefly review their method. Kindlmann and Westin visualize a symmetric tensor field using elliptical glyphs such that the size, shape and orientation of the glyphs reflect the tensor field at the center location of the glyph. This is achieved by placing a set of initial glyphs in the domain and moving them through repulsion forces between nearby glyphs. The repulsion force is derived from the underlying tensor field. The process terminates when convergence is reached.

5.1 Glyph Tensor

In order to apply glyph packing which requires symmetric tensor fields as input, we compute the following symmetric tensor which is equivalent to that of Equation 5:

$$T = (u_1\ u_2)\begin{pmatrix} J_1 & 0 \\ 0 & J_2 \end{pmatrix}(u_1\ u_2)^T \tag{6}$$

where u_1 and u_2 are the two eigenvectors of the desired symmetric matrix given by Equation 5, which are the major and minor dual-eigenvectors of the original asymmetric tensor [43].

$$\begin{cases} J_1 = \max\{|\gamma_s + \gamma_r|,\ |\gamma_s - \gamma_r|\} \\ J_2 = \min\{|\gamma_s + \gamma_r|,\ |\gamma_s - \gamma_r|\} \end{cases} \tag{7}$$

Note J_1 and J_2 computed this way are the singular values for the asymmetric tensor obtained by subtracting the trace. This treatment takes on the following physical interpretation. For 2D incompressible fluids, the local linearization at any point inside the complex domain is an elliptical pattern whose eccentricity and semi-axes are defined by Equation 6. In this case, eccentricity ($\frac{J_1}{J_2}$) indicates the relative strength between stretching and rotation. The smaller the eccentricity, the stronger the rotation. When eccentricity is minimum ($\gamma_s = 0$), there is no stretching but only rotation. That is where degenerate points occur (Poles in the eigenvector manifold). The limit when eccentricity approaches infinity ($\gamma_s = |\gamma_r|$), indicates degenerate curves. In addition, J_1 and J_2 correspond to the lengths of the semi-axes of the glyph which determine the size of the ellipse and reflect the tensor magnitude at the center of the glyph, i.e.

$$\sqrt{\frac{J_1^2 + J_2^2}{2}} = \sqrt{\frac{|\gamma_s + \gamma_r|^2 + |\gamma_s - \gamma_r|^2}{2}} = \sqrt{\gamma_s^2 + \gamma_r^2}$$

5.2 Glyph Seeding Strategy with Degenerate Points

Once we have generated the aforementioned symmetric tensor field inside the complex domains, we start the glyph packing process for one region at a time. We have observed that the quality and computational cost of glyph packing is greatly impacted by the initial seeding strategy, i.e., how many seeds and where to place them. For aesthetically pleasing results, we note that the area of the region to be seeded should be slightly larger than the total area of the glyphs for the region. When these two values are equal, glyph overlap will occur. To control this, we use a density parameter β to help control the overall number of glyphs seeded in the region.

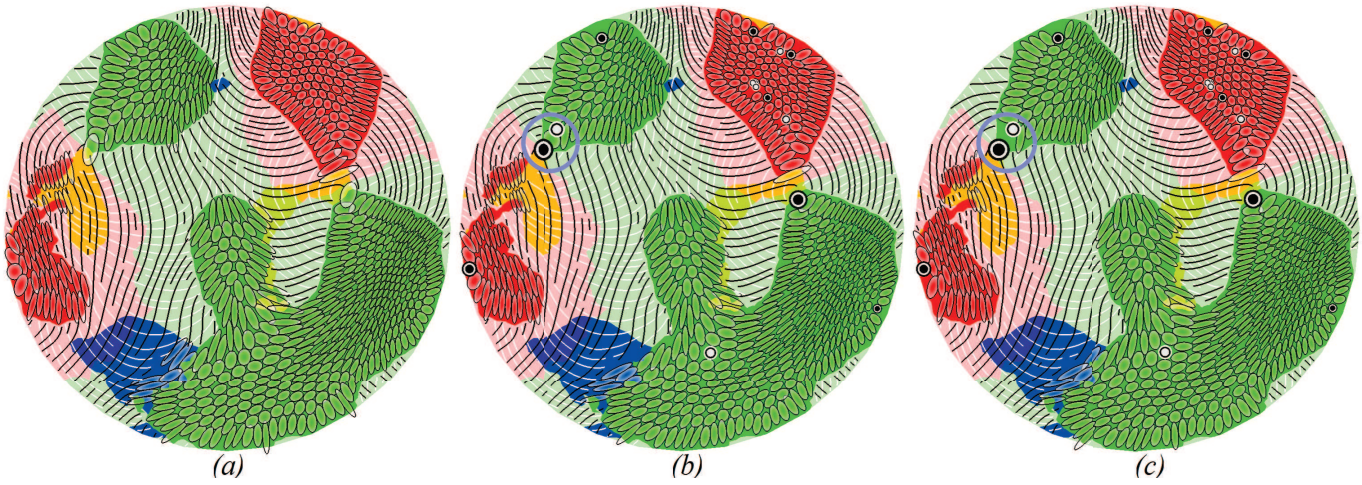


Fig. 3. Comparison of the asymmetric tensor field visualization results without seeding at degenerate points (a) and with fixed particles at degenerate points (b). Note that how the glyphs at degenerate points deliver the characteristics (isotropic) of tensor field near them (b) indicating a pure circular rotation of the flow at those locations. The image in (c) shows the result of seeding degenerate points with fixed sizes during the repulsion. This could prevent a proper placement of other seeds from being achieved (see the circled area).

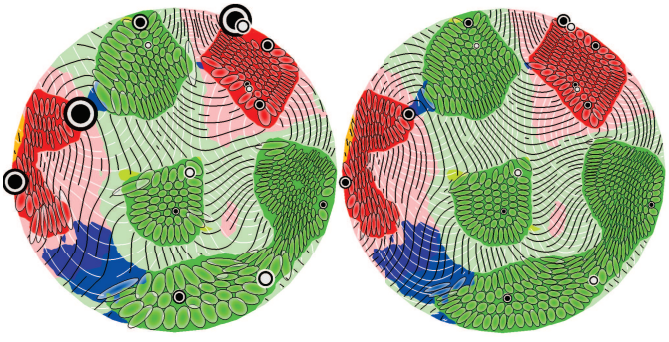


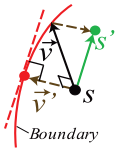
Fig. 4. This figure shows the results of glyph packing without capping the tensor magnitude (left) and with capping (right).

To seed a region A, we randomly place points in A until the total area of seeds within A is larger than βArea(A). In our implementation, β values range from 0.7 to 0.9 depending on the particular data set. To generate seeds, for a given candidate position s, we calculate $P = \left(1 - \frac{\det}{\det_{max}}\right)$ where det is the tensor determinant of the glyph tensor T (Equation 6) at s and $\det_{max}$ is the maximum determinant over A, which can be computed through a linear search over the tensors at all the samples in the domain. Since the determinant is proportional to the area of the glyph, P approaches zero as the glyphs get larger. With the probability P we accept s as a seed and update the total area of the seeds. The area of each seed point is evaluated as the area of the ellipse with the seed point as the center (i.e. $\alpha^2\pi J_1 J_2$ where J_1 and J_2 are the two values used in the creation of the symmetric glyph tensor from Equation 6 at the seed). Note that this algorithm is conducted inside each connected component of the complex domain individually.

The above seeding process is similar to [17]. However, Kindlmann and Westin do not consider degenerate points, important features in solid and fluid mechanics [43], in the seeding process. To better visualize the tensor behaviors near and at the degenerate points of an asymmetric tensor field, we assign a seed point at the center of each degenerate point. These glyphs will not move during glyph packing although they can generate repulsion forces on nearby regular glyphs. In our original implementation, seeds representing degenerate points are inserted at the beginning of the initialization. They are fixed during the repulsion stage. This poses some issues during the placement of other seeds (Figure 3, right). More specifically, these fixed seeds block the movement of other seeds from reaching optimal locations, which leads to a number of holes in the result. To handle this, we modify the seeding scheme at degenerate points as follows. Initially, we place seeds at degenerate points with zero area (i.e. having zero influence on the other seeds). After a certain number of iterations of repulsion, we start increasing the area of the glyphs linearly located at the degenerate points until they achieve the optimal sizes for final visualization.

5.3 Boundary Handling

To prevent the glyphs from entering or overlapping with specific areas, Kindlmann and Westin use a metric forcing points into the interior of the specific regions [17]. To obtain a continuous visualization between real and complex domains, our method requires more control over the boundary handling. Given a seed s, whose velocity is $\vec{v}$, the next location of s is the current position of s plus $\vec{v}$. We compute the point on the boundary of the region that has the shortest distance to s, denoted by $\vec{v'}$. If the length of $\vec{v'}$ is smaller than a specific value b, we flag s for boundary handling. We calculate b as $(J_1 u_1) \cdot \frac{\vec{v}}{|\vec{v}|}$ where J_1 is the major eigenvalue of the glyph tensor and u_1 is the major axis of the glyph. The value of b reflects the width of the glyph along $\vec{v'}$. If s is flagged for boundary handling, we modify the movement vector $\vec{v}$ to move laterally along the boundary by subtracting $\vec{v'}$ from $\vec{v}$ to obtain the new position s'. If this new position is still outside the boundary, s remains fixed in its original position. This allows the particles to move along boundaries and through narrow regions or enter large and open parts of the region without exiting it. The alignment of packed glyphs at domain boundaries can create regular patterns that may visually distract from the flow patterns. Addressing this is a direction of further research.



5.4 Clamping

In our experiments, the determinants in different complex domains of an asymmetric tensor field stemming from the flow field vary greatly. So do their eigenvalues J_1 and J_2. This typically leads to large variation of the sizes of the glyphs. Figure 4 (left) provides such an example. In addition, large eigenvalues require a large neighborhood computation when accumulating forces exerted on a seed. This increases the computation expense. To overcome this, we follow Kindlemann and Westin [17] by allowing the user to specify a threshold m_r for the ratio of the maximum (m_{max}) to the minimum (m_{min}) tensor magnitudes in the whole domain. Let $r = \frac{m_{max}}{m_{min}}$ and m be the tensor magnitude at a point p. We first normalize m as $k'' = \frac{m - m_{min}}{m_{max} - m_{min}}$. If $r > m_r$, set $r = m_r$. Then, a linear mapping is applied to k'' such that $k' = ak'' + b$, where $a = \frac{r-1}{\sqrt{r}}$ and $b = \frac{1}{\sqrt{r}}$. This maps the tensor magnitude to the range of $[\frac{1}{\sqrt{r}}, \sqrt{r}]$. This mapping preserves the ratio of J_1 and J_2. Finally, we set $k = \frac{1}{k'}$ such that the larger the tensor mag-

nitude is, the denser the hyperstreamlines and the smaller the glyphs. Figure 4 (right) shows the result after applying this mapping.

6 An Image-Space Implementation

Note that the placement of hyperstreamlines in real domains and the glyph packing in complex domains can be accomplished in the object space to preserve data authenticity. However, working in the object space presents difficulties when placing hyperstreamlines or packing glyphs on surface geometry. This is because points on hyperstreamlines and glyphs are frequently compared with other nearby points in the domain [15, 17]. For surfaces, this requires the geodesic distances between points to be computed to determine neighbors within a certain distance which is a computationally expensive task. This extra computational cost is possibly offset by the fact that the visualization needs to be generated just once after which the user can find a particular suitable viewpoint. However, under levels of magnification, the original visualization can become sparse prompting the re-computation of the visualization. If no method of determining visible regions is employed, the data generation time can increase dramatically as glyphs and hyperstreamlines are more tightly packed in the object space.

Using an image-space approach can alleviate these issues. The tensor field is projected to the image plane and the visualization is performed in this fixed frame. The costly geodesic distance computation can be omitted as the visualization no longer requires expensive distance computation on surfaces. Also, exploration of the data is possible as the visualization is only generated for a specific view. Image-space methods have been used to visualize vector fields [23, 35, 38], tensor fields [42] as well as generate pen-and-ink sketching [9] on surfaces with much success. The key is to be able to project the tensor field from the surface onto the image plane with minimal error, account for the distortion introduced by projection, and handle occlusion.

Tensor Field Projection There are two approaches to projecting a tensor field. In the first approach, the tensor values at the vertices of each triangle are projected onto the image plane. The tensor value for a point inside the triangle can be obtained through barycentric interpolation of the projected tensor values at the vertices of the triangle. This approach is fast and lends itself naturally to GPU processing. However, blending projected tensor values can often lead to relatively large errors. We instead use the second approach. In this case, for each pixel in the image plane, we identify the nearest point s on the surface whose projection covers the pixel. We then compute the tensor value at s, which is then projected onto the image plane. While this approach is slower than the first approach, it is still relatively fast and provides *pixel-level accuracy* while the first approach only provides vertex-level accuracy. To quickly identify the nearest point on the surface that corresponds to a given pixel in the image plane, we render the surface from the same viewpoint in two passes. In the first pass, we assign a unique color to each triangle. In the second pass, we color each vertex in the mesh with its barycentric coordinates. This results in two buffers: triangle ID buffer, and barycentric coordinate buffer, which will be stored as textures. With these two buffers, we can easily identify the nearest point on the surface that corresponds to a pixel through texture accessing and obtain the tensor value at that point.

The original surface tensor is then used to decide whether it is in the real or complex domain, compute the tensor magnitude, γ_d, γ_r, and γ_s, as well as derive the shape, size, and orientation of glyphs (complex domain) and the directions of hyperstreamlines (real domain). The tensor quantities γ_d, γ_r, and γ_s are then used to decide the color of the pixel based on the color scheme in [43]. If a pixel's corresponding surface point is in the real domain, we will compute the 3D coordinates of the major and minor eigenvectors of the tangential tensor and project them onto the image plane. This ensures that the projection onto the image plane of a hyperstreamline on the surface passing through a point s will coincide with the hyperstreamline in the image plane that passes through the pixel corresponding to s. Similarly, we wish to ensure that the projection of a glyph in the complex domain of the surface matches the glyph generated from the projected tensor. This is discussed later as the solution to this problem can be combined to also deal with distortions introduced by projecting distance or vectors onto the image plane.

Distortion in Projection Projections onto the image plane from a point on the surface introduces *anisotropic* distortion. This means that distance computations, used in controlling the spacing between hyperstreamlines and glyphs, need to account for this distortion or the result will look different from the projection of the hyperstreamlines and glyphs from the object space. We observe that under orthographic projection, the length of a vector is maintained in a certain direction $\vec{v}$ but distorted the most in the perpendicular direction of $\vec{v}$. This direction is defined using the object normal at the point p on the surface. Let $\vec{v} = \vec{n} \times \vec{z}$ where $\vec{n}$ is the object normal at p and $\vec{z}$ is the image normal. We can see that the length of $\vec{v}$ is the same in both the local frame of the point and the image plane. Along $\vec{u} = \vec{v} \times \vec{n}$, distances are shortened by a factor of $d = \vec{n} \cdot \vec{z}$ which is simply the z-component of $\vec{n}$. In fact, this anisotropic distortion can be described by a symmetric tensor, much like the stretching tensor used in surface parametrization [32].

We make use of this anisotropic scaling information along with the tensor magnitude to vary the density of the hyperstreamlines. To calculate the magnitude at a given point p under projection, we use the following matrix:

$$M = (\vec{u}_i\ \vec{v}_i)^T \begin{pmatrix} md & 0 \\ 0 & m \end{pmatrix} (\vec{u}_i\ \vec{v}_i) \tag{8}$$

Here m is the magnitude of the tensor at point p, d is the shortening factor associated with the normal at p, $\vec{v}_i$ is the projection of $\vec{v}$ to the image plane and $\vec{u}_i \perp \vec{v}_i$. $\vec{x}^T M \vec{x}$ is defined as the magnitude of the tensor where $\vec{x}$ is a direction in the image plane. This can be used to define the tensor magnitude at p in the direction of a seed s to be tested. By setting $\vec{x} = \frac{p-s}{|p-s|}$, we obtain the tensor magnitude. Instead of simply storing the tensor magnitude per pixel, we store M.

For glyph packing, the anisotropic scaling can be encoded into the glyph tensor instead of treating it separately as for hyperstreamline tracing. We note that the magnitude is already inherently encoded into the unprojected glyph tensor as well. The glyph tensor is defined in the local 2D frame of a specific triangle and our target frame is the image plane, which is also 2D. Therefore we can perform a 2D to 2D projection of the symmetric glyph tensor to obtain this projected glyph tensor. This becomes the following:

$$G' = (\vec{u}_i\ \vec{v}_i)^T \begin{pmatrix} \sqrt{d} & 0 \\ 0 & 1 \end{pmatrix} (\vec{u}_l\ \vec{v}_l)\, G\, (\vec{u}_l\ \vec{v}_l)^T \begin{pmatrix} \sqrt{d} & 0 \\ 0 & 1 \end{pmatrix} (\vec{u}_i\ \vec{v}_i) \tag{9}$$

Here G is the glyph tensor also defined in the local frame at p, $\vec{u}_i$, $\vec{v}_i$ and d are as in Equation 8, $\vec{v}_l$ is the projection of $\vec{v}$ into the local frame defined for the triangle in which p is located and $\vec{u}_l \perp \vec{v}_l$. We first perform a change of basis from the local frame at p to the basis $\vec{u}_l, \vec{v}_l$. We then scale the glyph by the z component of $\vec{n}$ in the direction of $\vec{u}_l$ and 1 in the direction of $\vec{v}_l$. We then perform another change of basis back to the image plane. The resulting matrix G' is defined in the image space and can be used in the glyph packing.

A side effect of this projection is that degenerate points are not maintained as their original circular shapes. Hence, degenerate points cannot be classified after projection in the image plane. To account for this, we extract and classify the degenerate points in the object space, then project the positions of the visible degenerate points for a particular viewpoint as well as their glyph tensors at those positions. Notice that the extraction of degenerate points is local (per triangle in the mesh) and there is at most one degenerate point per triangle. Consequently, degenerate point extraction is faster when compared with glyph packing. Moreover, it only needs to be performed once.

After projections, all tensors for points in the domain are computed through a bilinear interpolation between pixels in the image space.

Occlusion Handling There are two additional considerations when projecting the tensor field. First, occlusions introduced by the projection often lead to discontinuities in the image space. Such an issue has been raised in [23, 35]. To address this, we borrow the technique from [35] and make use of the depth buffer to detect pixels where depths change sharply. We mark these pixels as *cliff pixels*. If the depth value of two adjacent pixels is higher than a threshold value, we denote a depth discontinuity. Note that these pixels may occur in

the interior of a real or complex domain, and they do not always form closed loops. Hyperstreamline tracing and glyph movement cannot cross cliff pixels. In our implementation of glyph packing, the cliff pixels are treated the same way as pixels on the boundaries between the real and complex domains.

Binning Strategy As shown above, a neighborhood computation is needed for both hyperstreamline placement and glyph packing. This requires to identify the neighboring seeds within a disk centered at a given seed. To do so, all seeds in the domain need to be checked (i.e., computing distance to the center seed). To speed up this neighborhood computation, we adopt a spatial binning strategy in the image space for each region. The size of each bin in a region is $2\alpha\lambda_{max}$ due to its anisotropic property, where λ_{max} is the maximum eigenvalue within this region if it is real or the maximum J_1 if it is complex. When searching the neighborhood of a point p which is located at bin $B(i,j)$, all the its neighboring bins, $\bigcup_{r=i-1}^{i+1}\bigcup_{c=j-1}^{j+1}B(r,c)$ are considered as well as $B(i,j)$. Only seeds in these bins will be considered.

7 Applications

We have applied our hybrid visualization technique to simulated flows inside a diesel engine and cooling jacket as well as a simulated earthquake deformation. For each of the images shown in the paper, hyperstreamline tracing typically took less than one second, and glyph packing took $5-20$ seconds, depending on the numbers of glyphs and hyperstreamlines generated. This represents two orders of magnitude speedup over the object-space approach. All the examples were generated on a workstation having Intel Xeon(R) CPU with 2 processors (each of them is 2.33GHz), 8GB RAM, and an NVIDIA GeForce GTX 285 graphics card. Multi-thread programming is applied.

7.1 Domain Expert Review of Engine Simulation Flows (Domain Expert Yeh)

Cooling Jacket Simulation Figure 5 shows the flow in the outer surface of a cooling jacket data set. The upper portion is the jacket for the cylinder head and the lower portion is for the cylinder block. It is important to design a cooling jacket so that the flow within the jacket be adequately mixed for efficient heat transfer. Note that the combination of stagnation and rotation could be a sign of inefficient heat transfer that must be avoided. Our visualization of the velocity gradient tensor field allows to effectively explore flow kinematics on this surface (Figure 5 (c) and (d)). For example, it is straightforward to identify rotation dominant regions and how fluid elements deform quantitatively within the regions by the glyphs. This is not the case for the previous visualization as shown in Figure 5(b): the rotational pattern must be inferred from the area enclosed by the hyperstreamlines. With the proposed hybrid visualization, the density of the hyperstreamlines and sizes of the glyphs represent the tensor magnitude: the denser the hyperstreamlines and the smaller the glyphs, the larger the tensor magnitudes, consistent with the standard fluid mechanics representations such as streamlines and vortex lines (see any fundamental fluid mechanics textbooks). Figure 5(c) is the visualization of the same data set from the inlet, which is the front portion of the jacket. The coolant enters through the round triangular-shaped inlet that is shown near the bottom. The flow pattern adjacent to the inlet can be interpreted by observing the orientation of major and minor eigenvectors. The major eigenvectors are oriented parallel to the port edge, while the minor eigenvectors are perpendicular to the edge. This indicates that the flow decelerates by stretching the fluid parcels in the transverse direction to the flow. Also note the formation of alternating clockwise-counterclockwise rotations after the coolant enters the cylinder block although they are still in the real domain, i.e. stretching dominant flows. Note that this type of kinematic information cannot be extracted explicitly from the velocity vector visualization [20].

Near the top of the inlet, we see a pair of rotation dominant regions (i.e. complex domain): the counterclockwise rotation (red) on the right and the clockwise (green) on the left, connected with the region of flow contraction (blue). Note that the coolant fluids are incompressible; hence the blue regions in the figure are interpreted as contraction caused by the flow to the interior away from the visualized surface in the figure. (Likewise, the yellow regions are interpreted as expansion caused by the flow from the third dimension normal to the surface of visualization.) The vicinity of the blue (flow contraction) region between this pair of counter-rotating complex domains is a spot for further investigation. The flow there has a small rate of deformation (the glyph size and the space of hyperstreamlines there are large) and is rotation dominant which means small fluid mixing. Therefore, the heat transfer in this vicinity might be inefficient. On the other hand, this area is very close to the coolant intake; hence, we do not anticipate that the inefficient mixing behavior there would cause a serious problem - the coolant temperature should still remain cool enough and would unlikely cause overheating to induce undesirable air bubbles. The glyph presentation shows how fluid parcels deform with rotation. The elongation changes from the horizontal direction near the inlet to the vertical direction towards the cylinder head. Note that the cylinder block and head are connected through the gasket, and some of the coolant enters the cylinder head through the gasket right above the inlet. The tensor magnitude adjacent to the connection is relatively large (denser hyperstreamlines) and the major eigenvectors are oriented towards the gasket, i.e. stretching, together with the flow contraction (blue).

Once the coolant enters the cylinder head, a pair of vortices appears: the clockwise rotation (green) on the right and the counterclockwise (red) on the left. The glyph presentation shows the fluid parcel deformation pattern associated with the pair of vortices created by a jet-like flow through gasket. The horizontal orientation of major hyperstreamlines (i.e. stretching) near the top of the cylinder head separates the two counter-rotating vortices. The velocity-gradient tensor magnitude is relatively large in this area, indicating significant deformation of the fluid. The similar flow pattern can be observed in Figure 5(d) at every gasket connecting the cylinder block and head.

Diesel Engine Simulation In Figure 6(c), gas enters through the left intake port and exits through the right. We can see that flow contraction (blue) at the foot of the exhaust port (the right pipe) and a pair of regions with opposing rotations in the pipe which indicate that the gas is just commencing out from the cylinder to the exhaust. Unlike the previous example of the cooling jacket, the flow in the diesel engine cylinder is compressible; hence the yellow and blue regions represent the combination of the actual volumetric changes and the effect of the flow normal to the surface of visualization. The increase in magnitude of rotation (green) in the top surface of the cylinder and the elongation toward the exhaust are clearly shown by the glyphs (with smaller sizes). On the other hand, such a detailed fluid deformation patterns are difficult to extract from the previous visualization without the use of glyph presentation (Figure 6(a)). In addition, no information of tensor magnitude can be included in the previous visualization of [13]. Note that without showing the tensor magnitude through the varying sizes or densities of the elements, such information is difficult to identify (see Figure 6(b) where hyperstreamlines are evenly-spaced and the glyphs have equal area).

Figure 6(d) shows the flow on the sidewall of the cylinder. We can see that the tensor magnitude continues to decrease as the flow migrates towards the bottom of the cylinder. The contrast of the tensor magnitude near the bottom with that on the top surface of the cylinder is evident. This fluid deformation pattern indeed represents the combustion stage at the commencement of the exhaust process. The glyphs in the complex domain clearly exhibit the elongated counterclockwise rotation patterns. The sizes of glyphs are fairly uniform, except near the bottom where the glyphs are slightly larger (i.e. weak tensor magnitude) noting the motion being constrained by the bottom face (the top of the piston). There appear four degenerating points: two trisectors and two wedges. Considering that they are located near the boundary of the complex domain, within the dilation dominant region (yellow) and near the irrotational flow (at the interface of light red and light green), those weak degenerating points must represent flow stagnation. It is emphasized that the visualization of tensor magnitude with the density of hyperstreamlines and the glyph sizes enables us to better understand the flow fields. The foregoing intriguing flow behaviors are a few examples. Many other features can be detected

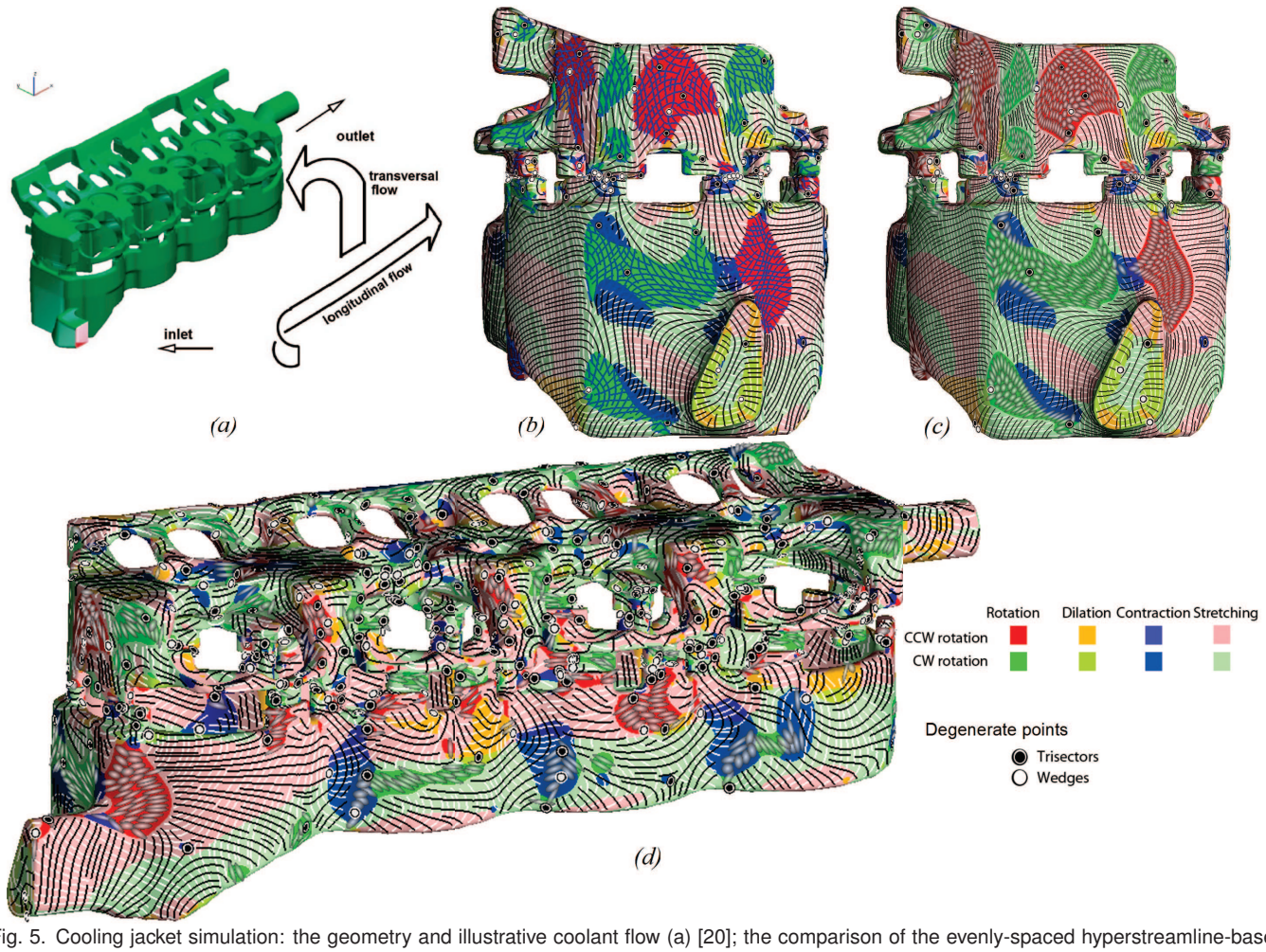


Fig. 5. Cooling jacket simulation: the geometry and illustrative coolant flow (a) [20]; the comparison of the evenly-spaced hyperstreamline-based visualization (b) and our hybrid visualization (c) ($\alpha = 0.0029$, $\beta = 0.8$, $iter = 330$, and $m_r = 50$) from the side view, and the hybrid visualization from the front view (d) ($\alpha = 0.0029$, $\beta = 0.8$, $iter = 300$, and $m_r = 20$).

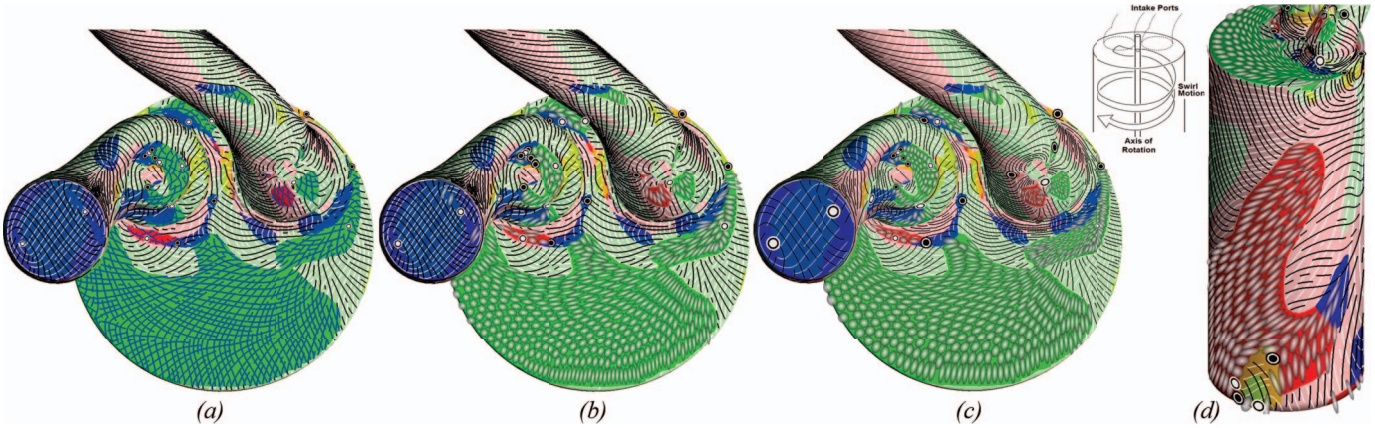


Fig. 6. The comparison of the hyperstreamline-based visualization (a), hybrid visualization without encoding tensor magnitude (b) ($\alpha = 0.023$, $\beta = 0.75$, $iter = 1200$, m_r is ignored here), and with tensor magnitude encoded (c), ($\alpha = 0.012$, $\beta = 0.76$, $m_r = 10$, $iter = 1200$) of the diesel engine simulation. (d) provides a side view of the hybrid visualization.

effectively with the presented hybrid tensor field visualization, much better than the previous visualization with the use of hyperstreamlines only and no representation of tensor magnitudes.

7.2 Domain Expert Review of Simulated Earthquake Deformation Data (Domain Expert Vincent)

We have also applied our hybrid visualization technique to a simulation of coseismic displacements from the June, 1992 Mw = 7.3 Landers, CA earthquake sequence. The simulation data is from [40] and based on the geodetic inversion source slip model of [14]. The model has 29 different fault segments (including two conjugate faults at high angle relative to the main rupture zone), each with its own strike (azimuth), length, dip, and coseismic slip vector. The fault segments with their associated slip vectors were combined to compute the surface displacements assuming an elastic half-space medium with a Poison's ratio of 0.25 [40].

Figure 1(c) shows displacement vector field using a combination of LIC-style streamlines (direction) and colors (magnitude) for the horizontal (north and east) components of surface displacements from the Landers earthquake simulation data set. The horizontal scale is 90 km $\times$ 90 km and the relative sense of motion across the fault zone is right-lateral–standing on one side of the fault looking toward the opposite side one sees points move to the right. While the general pattern of coseismic displacements is what would be expected for a right-lateral strike-slip fault, there is additional information not contained in this streamline/vector magnitude plot that becomes apparent when we apply our hybrid visualization technique to the simulation data set.

Figure 1(a) shows our hybrid visualization technique applied to the data. Several additional features associated with the displacement field become apparent. To help elucidate the additional information content available by applying our visualization technique, and to put this information in the proper context of fault mechanics, Figure 1(d) is a schematic drawing to illustrate the expected deformation modes for a right-lateral fault that now becomes visible using our visualization technique. For right-lateral motion across a fault (or fault zone) there is a dominant counterclockwise rotation on both sides of the fault. This is because right-lateral slip occurs on the fault but not away from the fault setting up a counterclockwise rotation pattern on both sides of the fault. Counterclockwise rotation can be seen in Figure 1(a) as red coloring in the real and complex domains, and the glyphs represent the complex eigenvalue regions where the rotational component dominates the shear component of deformation. The green regions at the fault tips and close to the fault represent clockwise rotations and similarly where the glyphs are located are regions dominated by rotation compared to shear. Boundaries between green and red (clockwise and counterclockwise rotation respectively) represent regions of compression (northwest and southeast quadrants) and dilatation (southwest and northeast quadrants) and are consistent with the seismic focal mechanism used to predict the seismic radiation pattern associated with right-lateral slip across a vertical fault. This radiation pattern is not visible using standard methods of displaying geodetic (surface deformation) data of earthquakes. It is inferred from first arrivals on distant seismometers (up or down for compression or dilatation respectively) distributed in directions at each of the four quadrant azimuths. Using our visualization method they can be seen easily. The two conjugate faults (a large one to the west and a small one to the northeast of the main rupture zone) are left-lateral and act to increase the rotational component of deformation locally which can be seen as elongation of the glyphs in the case of the large conjugate fault on the west side of the rupture zone (see Figure 7). Again, this deformation pattern cannot be seen using previous methods to display surface deformation data. The combined deformation modes are now apparent using our hybrid visualization method. By extracting more information, we are able to show a multitude of relevant deformation modes that encompass both geodetic displacement and seismic radiation pattern information in one plot. The additional information content now available will help geodesists and seismologists jointly interpret what has historically been disparate data sets. For example, insights into earthquake rupture dynamics processes can now potentially be made by viewing areas of localized stretching or dilation, or abrupt transitions between stretching and rotation, that are not visible using standard geodetic data visualizations of displacement or displacement gradients alone (e.g., Figure 1(c)). This new information could lead to new insights into how faults rupture, why rupture begins and ends where it does, and where the next rupture might originate. The visualization of asymmetric tensor fields associated with surface deformation, using this approach, provides a critical connection between seismic and geodetic data and interpretation that opens up many new avenues for research.

8 Conclusion

Asymmetric tensor field visualization is becoming an important topic within the visualization community where more work is needed. In this paper, we highlight the challenges faced by existing techniques for asymmetric tensor field visualization, including the loss of magnitude information and the lack of effectiveness for conveying the elliptical tensor patterns in complex domains. In order to address these challenges, we introduce a hybrid visualization technique for asymmetric tensor fields in which hyperstreamlines and glyphs are used to represent tensor patterns in real and complex domains, respectively. The sizes of the glyphs and densities of the hyperstreamlines are used to convey tensor magnitude, and degenerate points are retained in the visualization. This is the first time glyph packing is used in conjunction with asymmetric tensor fields. The combination of these techniques generates new hybrid visualization results that are capable of revealing the underlying physical characteristics of the data more effectively and efficiently. We also present an efficient image-space approach for visualizing asymmetric tensor data on surfaces. This reduces the computation time for the visualization by two orders of magnitude and allows the user to have more control over which portions of the view are magnified and highlighted. This is also the first time an image-space method is used for glyph packing for the depiction of asymmetric tensor fields on surfaces. The result is an interactive system with which the user can explore their asymmetric tensor fields using the proposed hybrid visualization and other options. Finallly, we introduce an important new application of asymmetric tensor field data, namely earthquake deformation.

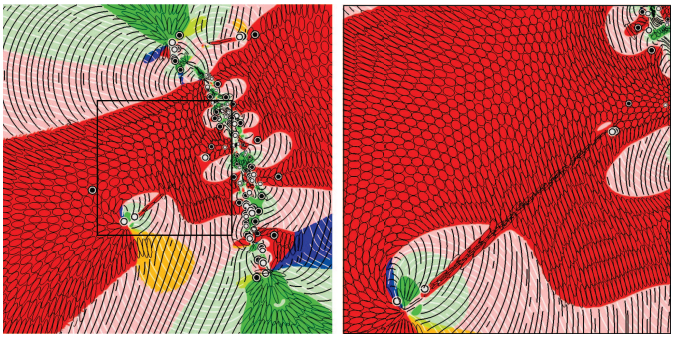


Fig. 7. Magnified view of the deformation associated with the conjugate fault to the west of the main rupture zone.

While the domain experts are more favarable toward the presented new hybrid visualization, we recognize that this is not the only solution. To encode tensor magnitude, other strategies, such as varying contrast and thickness of lines, can be employed. We also provide an option in our system to allow the tensor magnitude to be displayed in a separate plot. In addition, although the visual contrast between hyperstreamlines and glyphs at the boundaries of real and complex domains effectively highlights the distinct behaviors of these two domains, it may cause visual distraction. As such, our system retains the option of visualizing both domains using hyperstreamlines for more coherent transition through the degenerate curves. Exploring other unified visual primitive for both real and complex domains is possible. Superquadric glyphs [33] is a good candidate. In the future, we wish to explore additional applications for asymmetric tensor visualization. As our understanding of 2D asymmetric tensor fields matures, we wish to expand the analysis and visualization to 3D asymmetric tensor fields, which are significantly more challenging due to that fact there are nine numbers in a 3×3 tensor versus four numbers in a 2×2 tensor. In addition, we plan to investigate faster glyph packing techniques based on more recent development in Centroidal Voronoi Tessellations (CVT) [8].

Acknowledgments

The research is partially supported by NSF awards IIS-0546881 and CCF-0830808. Guoning Chen was partially supported by DOE SciDAC VACET and Fusion SAP. Harry Yeh acknowledges the support of the Edwards endowment.

References

[1] P. Alliez, D. Cohen-Steiner, O. Devillers, B. Lévy, and M. Desbrun. Anisotropic polygonal remeshing. *ACM Transactions on Graphics (SIGGRAPH 2003)*, 22(3):485–493, July 2003.

[2] W. Benger and H.-C. Hege. Tensor splats. In *Visualization and Data Analysis 2004, Proc. of SPIE*, volume 5295, pages 151–162, June 2004.

[3] A. Bhalerao and C.-F. Westin. Tensor splats: visualising tensor fields by texture mapped volume rendering. In *Sixth International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI'03)*, pages 294–901, Montreal, Canada, November 2003.

[4] B. Cabral and L. C. Leedom. imaging vector fields using line integral convolution. In *Poceedings of ACM SIGGRAPH 1993*, Annual Conference Series, pages 263–272, 1993.

[5] G. Chen, K. Mischaikow, R. S. Laramee, P. Pilarczyk, and E. Zhang. Vector field editing and periodic orbit extraction using Morse decomposition. *IEEE Transactions on Visualization and Computer Graphics*, 13(4):769–785, 2007.

[6] W. C. de Leeuw and J. J. van Wijk. A probe for local flow field visualization. In *Proceedings of the 4th conference on Visualization '93*, VIS '93, pages 39–45, Washington, DC, USA, 1993. IEEE Computer Society.

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

[8] L. Feng, I. Hotz, B. Hamann, and K. Joy. Anisotropic noise samples. *IEEE Transactions on Visualization and Computer Graphics*, 14:342–354, 2008.

[9] A. Hertzmann and D. Zorin. Illustrating smooth surfaces. *Computer Graphics Proceedings, Annual Conference Series (SIGGRAPH 2000)*, pages 517–526, Aug. 2000.

[10] L. Hesselink, Y. Levy, and Y. Lavin. The topology of symmetric, second-order 3D tensor fields. *IEEE Transactions on Visualization and Computer Graphics*, 3(1):1–11, Mar. 1997.

[11] M. Hlawitschka and G. Scheuermann. HOT lines: tracking lines in higher order tensor fields. In *Proceedings IEEE Visualization 2005*, pages 27–34, 2005.

[12] M. Hlawitschka, G. Scheuermann, and B. Hamann. Interactive glyph placement for tensor fields. In *Proceedings of the 3rd international conference on Advances in visual computing - Volume Part I*, ISVC'07, pages 331–340, Berlin, Heidelberg, 2007. Springer-Verlag.

[13] H. Hotz, L. Feng, H. Hagen, B. Hamann, K. Joy, and B. Jeremic. Physically based methods for tensor field visualization. In *Proceedings IEEE Visualization 2004*, pages 123–130, 2004.

[14] K. W. Hudnut. Coseismic displacements of the 1992 Landers earthquake sequence. *Bull. Seism. Soc. Am.*, 84:625–645, 1994.

[15] B. Jobard and W. Lefer. Creating evenly–spaced streamlines of arbitrary density. In *Proceedings of the Eurographics Workshop on Visualization in Scientific Computing '97*, volume 7, pages 45–55, 1997.

[16] G. Kindlmann. Superquadric tensor glyphs. In *Proceedings of IEEE TVCG/EG Symposium on Visualization 2004*, pages 147–154, May 2004.

[17] G. Kindlmann and C.-F. Westin. Diffusion tensor visualization with glyph packing. *IEEE Transactions on Visualization and Computer Graphics (Proceedings Visualization / Information Visualization 2006)*, 12(5):1329–1335, September-October 2006.

[18] R. M. Kirby, H. Marmanis, and D. H. Laidlaw. Visualizing multivalued data from 2D incompressible flows using concepts from painting. In *Proceedings IEEE Visualization '99*, pages 333–340. ACM Press, Oct. 25–29 1999.

[19] D. H. Laidlaw, E. T. Ahrens, D. Kremers, M. J. Avalos, R. E. Jacobs, and C. Readhead. Visualizing diffusion tensor images of the mouse spinal cord. *Visualization Conference, IEEE*, pages 127–134, 1998.

[20] R. S. Laramee, C. Garth, H. Doleisch, J. Schneider, H. Hauser, and H. Hagen. Visual analysis and exploration of fluid flow in a cooling jacket. In *Proceedings IEEE Visualization 2005*, pages 623–630, 2005.

[21] R. S. Laramee, H. Hauser, H. Doleisch, F. H. Post, B. Vrolijk, and D. Weiskopf. The state of the art in flow visualization: dense and texture-based techniques. *Computer Graphics Forum*, 23(2):203–221, June 2004.

[22] R. S. Laramee, H. Hauser, L. Zhao, and F. H. Post. Topology-based flow visualization: the state of the art. In *The Topology-Based Methods in Visualization Workshop (TopoInVis 2005), Visualization and Mathematics*, pages 1–19, 2007.

[23] R. S. Laramee, J. J. van Wijk, B. Jobard, and H. Hauser. ISA and IBFVS: image space based visualization of flow on surfaces. *IEEE Transactions on Visualization and Computer Graphics*, 10(6):637–648, Nov. 2004.

[24] Z. P. Liu and R. J. Moorhead, II. An advanced evenly-spaced streamline placement algorithm. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):965–972, 2006.

[25] X. Mao, Y. Hatanaka, H. Higashida, and A. Imamiya. Image-guided streamline placement on curvilinear grid surfaces. In *Proceedings IEEE Visualization '98*, pages 135–142, 1998.

[26] M. Marinov and L. Kobbelt. Direct anisotropic quad-dominant remeshing. *Computer Graphics and Applications, 12th Pacific Conference on (PG'04)*, pages 207–216, 2004.

[27] T. McLoughlin, R. S. Laramee, R. Peikert, F. H. Post, and M. Chen. Over two decades of integration-based, geometric flow visualization. *Computer Graphics Forum*, 29(6):1807–1829, 2010.

[28] A. Mebarki, P. Alliez, and O. Devillers. Farthest point seeding for efficient placement of streamlines. In *Proceedings IEEE Visualization 2005*, pages 479–486. IEEE Computer Society, 2005.

[29] J. Palacios and E. Zhang. Rotational symmetry field design on surfaces. *ACM Trans. Graph.*, 26(3):55, 2007.

[30] N. Ray, B. Vallet, W. C. Li, and B. Lévy. N-symmetry direction field design. *ACM Trans. Graph.*, 27(2):1–13, 2008.

[31] O. Rosanwo, C. Petz, S. Prohaska, H.-C. Hege, and I. Hotz. Dual streamline seeding. In *Proceedings of the 2009 IEEE Pacific Visualization Symposium*, PACIFICVIS '09, pages 9–16, Washington, DC, USA, 2009. IEEE Computer Society.

[32] P. V. Sander, J. Snyder, S. J. Gortler, and H. Hoppe. Texture mapping progressive meshes. In *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 409–416, New York, NY, USA, 2001. ACM.

[33] T. Schultz and G. L. Kindlmann. Superquadric glyphs for symmetric second-order tensors. *IEEE Transactions on Visualization and Computer Graphics*, 16:1595–1604, 2010.

[34] K. Shimada, A. Yamada, and T. Itoh. Anisotropic triangulation of parametric surfaces via close packing of ellipsoids. *International Journal of Computational Geometry and Applications*, 10:417–440, 2000.

[35] B. Spencer, R. S. Laramee, G. Chen, and E. Zhang. Evenly-spaced streamlines for surfaces: an image-based approach. *Computer Graphics Forum*, 28(6):1618–1631, 2009.

[36] X. Tricoche, G. Scheuermann, and H. Hagen. Tensor topology tracking: a visualization method for time-dependent 2D symmetric tensor fields. In *Computer Graphics Forum 20(3) (Eurographics 2001)*, pages 461–470, Sept. 2001.

[37] G. Turk and D. Banks. Image-guided streamline placement. In *ACM SIGGRAPH 96 Conference Proceedings*, pages 453–460, Aug. 1996.

[38] J. J. van Wijk. Image based flow visualization for curved surfaces. In *Proceedings IEEE Visualization '03*, pages 123–130. IEEE Computer Society, 2003.

[39] V. Verma, D. Kao, and A. Pang. A flow-guided streamline seeding strategy. In *Proceedings IEEE Visualization 2000*, pages 163–170, 2000.

[40] P. Vincent. *Application of SAR Interferometry to Low-rate Crustal Deformation Fields*. PhD thesis, University of Colorado, 1998.

[41] K. Wu, Z. Liu, S. Zhang, and R. Moorhead. Topology-aware evenly spaced streamline placement. *Visualization and Computer Graphics, IEEE Transactions on*, 16(5):791 –801, sept.-oct. 2010.

[42] E. Zhang, J. Hays, and G. Turk. Interactive tensor field design and visualization on surfaces. *IEEE Transactions on Visualization and Computer Graphics*, 13(1):94–107, 2007.

[43] E. Zhang, H. Yeh, Z. Lin, and R. S. Laramee. Asymmetric tensor analysis for flow visualization. *IEEE Transactions on Visualization and Computer Graphics*, 15(1):106–122, 2009.

[44] X. Zheng and A. Pang. HyperLIC. In *Proceedings IEEE Visualization 2003*. IEEE Computer Society, 2003.

[45] X. Zheng and A. Pang. Topological lines in 3D tensor fields. In *Proceedings IEEE Visualization '04*, pages 313–320, 2004.

[46] X. Zheng and A. Pang. 2D asymmetric tensor analysis. *IEEE Proceedings on Visualization*, pages 3–10, Oct 2005.

[47] X. Zheng, B. Parlett, and A. Pang. Topological structures of 3D tensor fields. In *Proceedings IEEE Visualization 2005*, pages 551–558, 2005.