# Interactive Visualization of Rotational Symmetry Fields on Surfaces

Jonathan Palacios and Eugene Zhang, *Member*, *IEEE Computer Society*

**Abstract**—Rotational symmetries (RoSys) have found uses in several computer graphics applications, such as global surface parameterization, geometry remeshing, texture and geometry synthesis, and nonphotorealistic visualization of surfaces. The visualization of $N$-way rotational symmetry ($N$-RoSy) fields is a challenging problem due to the ambiguities in the $N$ directions represented by an $N$-way symmetry. We provide an algorithm that allows faithful and interactive representation of $N$-RoSy fields in the plane and on surfaces, by adapting the well-known line integral convolution (LIC) technique from vector and second-order tensor fields. Our algorithm captures $N$ directions associated with each point in a given field by decomposing the field into multiple different vector fields, generating LIC images of these fields, and then blending the results. To address the loss of contrast caused by the blending of images, we observe that the pixel values in LIC images closely approximate normally distributed random variables. This allows us to use concepts from probability theory to correct the loss of contrast without the need to perform any image analysis at each frame.

**Index Terms**—Rotational symmetry, RoSy, visualization, tensor field visualization, image blending, contrast adjustment.

## 1 Introduction

The visualization of rotational symmetry (RoSy) fields has many applications in computer graphics, such as surface parameterization [2], [7], [14]; quadrangular and triangular geometry remeshing [1], [8]; texture and geometry synthesis [23], [8]; architectural modeling [17]; and nonphotorealistic rendering [4], [24] (Fig. 1). Intuitively, an *$N$-way rotational symmetry* ($N$-RoSy) represents phenomena that are invariant under rotations of integer multiples of $\frac{2\pi}{N}$ [13], [15]. Examples of $N$-RoSys include a vector ($N = 1$), an eigenvector of a symmetric matrix ($N = 2$), and a cross ($N = 4$). An $N$-RoSy field is simply a directional field with a set of $N$ different directions associated with every point in the domain, each being a $\frac{2k\pi}{N}$ rotation of the others, where $k$ is an integer.

Despite the number of applications, there has been little work on the visualization of $N$-RoSy fields for $N > 2$. In this paper, we present a texture-based visualization technique that adapts the line integral convolution (LIC) algorithm [3] to handle the $N$-RoSy case. Texture-based flow visualization techniques are known to be space filling, easily adapted to render surface fields, and usually interactive when implemented on modern graphics hardware. The images generated by our method are similar to those seen in the works by Palacios and Zhang [13], and Ray et al. [15], the techniques for both of which remain unpublished.

The chief challenge presented by $N$-RoSy visualization when $N > 2$ lies in the fact that there are more than one trajectory that pass through each point in the domain. As shown in Fig. 2, it is difficult to understand the structures in an $N$-RoSy field when only showing one trajectory per point, i.e., treating the $N$-RoSy field as a vector field. In addition, in the presence of singularities, it is impossible to convert the $N$-RoSy field into a vector field without causing any visual discontinuity (Section 3).

Given a vector field, the LIC algorithm traces a streamline in both directions at each pixel, and uses these streamlines as convolution kernels to anisotropically blur a noise image. A brute-force way to extend the LIC algorithm to handle the $N$-RoSy case involves two changes. First, at each pixel, one needs to trace multiple trajectories. For each of these trajectories, the tracing method needs to be altered so that, at each integration step, all $N$ directions at the corresponding point are examined, and the one that best matches the previous direction is chosen. However, choosing the best of $N$ directions at every integration step introduces a large amount of branching in the tracing algorithm, which increases linearly with $N$. This branching can severely affect performance on modern parallel architectures, such as graphis processing units (GPUs).

To overcome the difficulty associated with branching, our algorithm decomposes a given $N$-RoSy field into multiple vector fields which, together, capture all $N$ directions at each point. We then generate LIC images for each vector field and blend these images together, which results in an image that visually represents the entire $N$-RoSy field. During numerical integration, our algorithm is significantly faster than the aforementioned brute-force adaptation of LIC (Table 1).

Further, difficulty arises from a generic problem in blending images from texture-based flow images; as we blend more and more images together, the resulting image begins to converge to the color gray. We thus make a further contribution in observing that the pixel values of LIC images generated from a binary noise texture are normally distributed random variables. This allows us to correct the loss of contrast resulting from the blending of LIC images without the need for image analysis. While this technique is

• *The authors are with the School of Electrical Engineering and Computer Science, Oregon State University, Corvallis OR 97331. E-mail: {palacijo, zhange}@eecs.oregonstate.edu.*

*Manuscript received 19 July 2009; revised 26 Jan. 2010; accepted 8 Feb. 2010; published online 10 Sept. 2010.*
*Recommended for acceptance by J. van Wijk.*
*For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org, and reference IEEECS Log Number TVCG-2009-07-0151.*
*Digital Object Identifier no. 10.1109/TVCG.2010.121.*

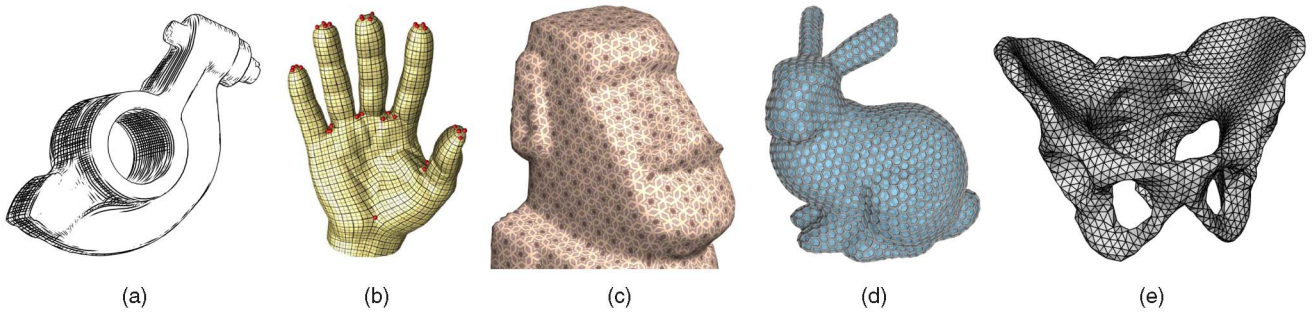



Fig. 1. (a) Pen-and-ink surface visualization algorithms, such as the one described in [4], require 4-RoSy fields as an input, as do many global parameterization algorithms, such as (b) QuadCover [7]. Applications of 6-RoSy fields include (c) regular hexagonal texture tiling, (d) geometry synthesis as well as (e) regular triangular remeshing. The highly regular triangular meshes generated using field-guided techniques can also be used as a starting point for circle packing algorithms, which have applications in free-form architecture [17].

part of our pipeline for $N$-RoSy field visualization, we believe that the fundamental idea can find use in any application where a number of texture-based vector and tensor field visualization need to be composited.

The remainder of this paper is organized as follows: we first review related work in Section 2, and then some theoretical background on $N$-RoSy fields in Section 3. In Section 4, we give the details of our planar algorithm. Our proposed method of contrast correction is discussed in Section 5, and Section 6 covers the extension of the planar algorithm to surfaces. We give our results in Section 7, where we compare our algorithm to a brute-force implementation of $N$-RoSy LIC. Finally, we close with our conclusions in Section 8.

## 2 Related Work

While $N$-RoSy fields have been applied to many problems in geometric modeling and computer graphics, there has been little published work on their visualization when $N > 2$.

### 2.1 $N$-RoSy Fields

To the best of our knowledge, Hertzmann and Zorin [4] were the first to use 4-RoSy fields in their work on the illustration of smooth surfaces. In this work, they use a "cross" field derived from the principal curvature directions to guide the orientations of the hatches in their images.

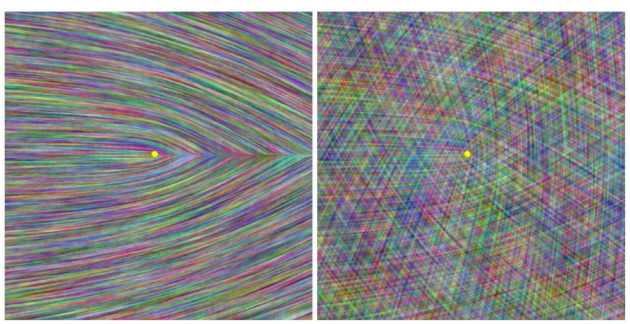



Fig. 2. Here, we demonstrate that visualizing only one of the directions of a 6-RoSy field not only results in a visual discontinuity ((a), the horizontal line to the right of the colored dot), but also does not allow a user to see the distinct patterns of the field around features like singularities. In order to achieve this goal, one must render all six directions at each point (b).

Ray et al. [14] use 4-RoSy fields to specify local $uv$-directions in their work on periodic global parameterization, which facilitates quadrangular remeshing. They later develop a design system [15] that allows the user to control the number and locations of singularities of $N$-RoSy fields on surfaces for arbitrary values of $N$. Their paper features LIC style images of 4-RoSy fields similar to ours, but the method used to generate them was never published. Palacios and Zhang [13] proposed an $N$-RoSy field design and analysis system that extracts singularities and separatrices for fields of arbitrary $N$. They also point out that $N$-RoSy fields can be represented by a special subspace of $N$th-order symmetric tensor fields, allowing mathematically sound algebraic operations, such as interpolation and change of basis. Again, that work features visualizations similar to ours, although no description of the visualization technique was provided in the paper.

### 2.2 Vector and Tensor Field Visualization

To review all previous work in vector and tensor field visualization is beyond the scope of this paper. We will cover only the most relevant here (namely, texture-based methods), and refer the readers to the works by Laramee et al. [9], Zhang et al. [24], and the references therein for further information. Texture-based vector field visualizations, first introduced by Van Wijk [20], are space filling and interactive when accelerated by modern graphics hardware. Many variations, accelerations and extensions have been proposed, such as image based flow visualization (IBFV) [21], IBFV for surfaces [22], [10]; LIC [3]; and others [19], [18], [5]. IBFV and LIC have both been adapted to also handle symmetric second-order tensors fields (2-RoSy), in [6], [25], and [24]. In particular, Hsu [6] was the first to apply LIC to diffusion tensor visualization,

TABLE 1
A Rendering Time Comparison between Our Algorithm and a Brute-Force Implementation of an $N$-RoSy Version of LIC

| | Our algorithm | | Brute-force algorithm | |
|---|---|---|---|---|
| $N$ | 12K $\Delta$ | 200K $\Delta$ | 12K $\Delta$ | 200K $\Delta$ |
| 2 | 17 | 28 | 211 | 237 |
| 3 | 46 | 57 | 1059 | 1182 |
| 4 | 32 | 43 | 564 | 627 |
| 6 | 46 | 57 | 1059 | 1182 |

*Rendering times are in milliseconds and mesh sizes in triangles appear in the second row.*

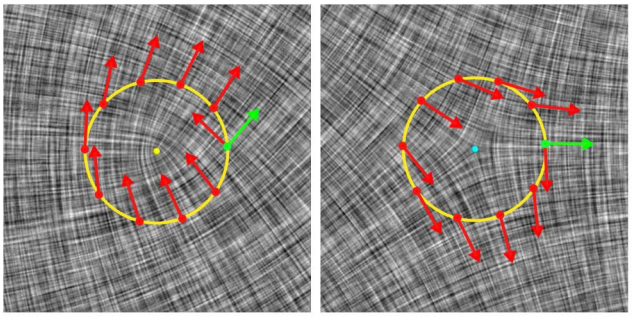



Fig. 3. Here we see two 4-RoSy singularities; starting with the green member vectors, we travel around the yellow circles in the counterclockwise direction, and track the changes in the angular component of these member vectors. When we get back to the same point we started at, the member vector has made a $\frac{\pi}{2}$ counterclockwise rotation (a) and a $\frac{\pi}{2}$ clockwise rotation (b) and that the starting and ending vectors are discontinuous. Notice that it is impossible to convert an $N$-RoSy field into a continuous vector field when such singularities are present.

and originated a two-pass method that appropriately blurs across the LIC result from the principal eigenvector when the two largest eigenvalues are nearly equal (in which case the corresponding eigenvectors are not well defined). Sanderson et al. [16] proposed a vector field visualization based on reaction-diffusion that combines concepts from glyph, streamline, and image-based techniques.

In our pipeline, we use a LIC implementation very similar to the GPU image-space technique presented by Li et al. [11] to generate each of the $N$ LIC images which are later composed to produce our final visualization. However, any other texture-based vector field visualization technique could also be easily incorporated into our general framework (e.g., IBFV).

## 3 Background: $N$-RoSy Fields

In this section, we briefly review the definition and properties of $N$-RoSy fields; for more in-depth treatments of this topic, we refer the reader to the works of Palacios and Zhang [13], and Ray et al. [15]. As mentioned previously, an $N$-RoSy represents phenomena that are invariant under integer multiples of rotations of $\frac{2k\pi}{N}$. More formally, an $N$-RoSy $s$ is a set of $N$ different 2D directions (or *member vectors*) of equal magnitude, where each is a $\frac{2k\pi}{N}$ rotation of the others, and $k \in \mathbb{Z}$:

$$s = \left\{ \rho \left( \cos\left(\theta + \frac{2k\pi}{N}\right) \sin\left(\theta + \frac{2k\pi}{N}\right) \right)^T \;\middle|\; 0 \le k \le N-1 \right\}. \tag{1}$$

Here, $\rho$ is the magnitude of $s$, and $\theta$ is the angle of one of the member vectors; only one member vector must be stored in order to retrieve the others. $N$-RoSys can be represented by a subspace of 2D, symmetric, $N$th-order tensors [13].

An $N$-RoSy field $S$ is a continuous $N$-RoSy-valued function of some spatial domain; that is, for every point $\mathbf{p}$ in the domain, there is an associated $N$-RoSy:

$$S(\mathbf{p}) = \left\{ \rho(\mathbf{p}) \begin{pmatrix} \cos\left(\theta(\mathbf{p}) + \frac{2k\pi}{N}\right) \\ \sin\left(\theta(\mathbf{p}) + \frac{2k\pi}{N}\right) \end{pmatrix} \;\middle|\; 0 \le k \le N-1 \right\}. \tag{2}$$

Moreover, the continuity of an $N$-RoSy field is defined in terms of the continuity of corresponding higher order tensor field [13]. In this paper, the domain is either the 2D euclidean plane, or a 2D surface embedded in 3D euclidean space.

A *singularity* in an $N$-RoSy field $S$ is any point $\mathbf{p}_0$ in the domain where $\rho(\mathbf{p}_0) = 0$, i.e., $\theta(\mathbf{p}_0)$ is undefined. As Fig. 3 demonstrates, it is impossible to decompose the member vectors of the field in the regions surrounding some singularities into continuous vector fields. Since part of our algorithm (described in Section 4) decomposes a given $N$-RoSy field into $N$ vector fields, this fact represents an obstacle that we must overcome in order to produce quality visualizations.

## 4 Visualization of $N$-RoSy Fields

In this section, we describe our algorithm in further detail. As previously mentioned, our basic planar algorithm has the following steps:

1. Decompose the original $N$-RoSy field into a set of $N$ member vector fields.
2. Generate texture-based flow images for each of these vector fields.
3. Compose (blend) these images uniformly into a single image that captures the original field.

There are some complications that arise from the fact that, in the general case, a member vector field will be discontinuous when $N > 1$ (see Section 3), and thus further steps must be taken to reduce the visual artifacts that result from these discontinuities. We now describe each step in more detail; the entire pipeline is outlined in Fig. 5 for a 3-RoSy field.

Recall that every point $\mathbf{p}$ in the domain of an $N$-RoSy field $S$ (except singularities) has $N$ member vectors (2). We first discuss the case where $N$ is odd. Our goal for this stage of our pipeline is to decompose the original field $S$ into a set of $N$ vector fields $\{V_0, V_1, \ldots, V_{N-1}\}$, such that for every point $\mathbf{p}$ in the domain $V_i(\mathbf{p}) \in S(\mathbf{p})$ and $V_i(\mathbf{p}) \neq V_j(\mathbf{p})$, where $0 \le i < j \le N-1$. That is, the direction at every point in each vector field $V_i$ is one of the directions at the same point in $S$, and none of the vector fields have the same direction at the same point. Thus, every direction at every nonsingularity point $\mathbf{p}$ in $S$ is represented by the set $\{V_0(\mathbf{p}), \ldots V_{N-1}(\mathbf{p})\}$. As mentioned earlier, in the vicinity of a singularity visual discontinuity occurs when one attempts to construct a continuous vector field from S. Consequently, it is insufficient to use only N vector fields to represent the $N$-RoSy field. To address this, we will generate $2N$ vector fields which, when blended properly, will result in a high-quality visualization. To facilitate the discussion, we give the following definitions.

Given an $N$-RoSy $s = \{\rho(\cos(\theta + \frac{2k\pi}{N}) \sin(\theta + \frac{2k\pi}{N}))^T \mid 0 \le k \le N-1\}$ and a *guiding angle* $\phi \in [0, 2\pi)$, let

$$v_{s,\phi} = \rho \left( \cos\left(\theta + \frac{2l_{s,\phi}\pi}{N}\right) \sin\left(\theta + \frac{2l_{s,\phi}\pi}{N}\right) \right)^T, \tag{3}$$

where

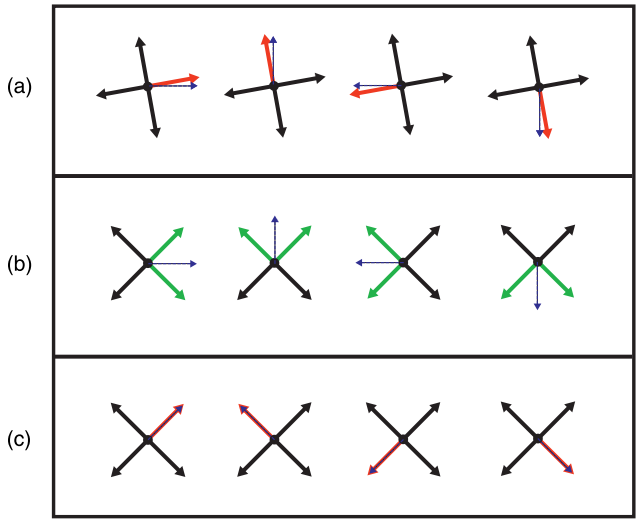



Fig. 4. In (a), we see how the guiding angle $\phi$ (indicated by the blue arrow) allows us to select one of the $N$ directions (in red) from an $N$-RoSy ($N = 4$ in this case). Note that if we use $N$ guiding angles with values of $\phi + \frac{2i\pi}{N}$ ($\phi = 0$ in this case), we will get one of the $N$ directions for each. In (b), we see that if there are two directions (in green) that are equally close to $\phi$ for a given $N$-RoSy $s$, then there will also be two such directions for each $\phi + \frac{2i\pi}{N}$. (c) illustrates that using $\phi + \frac{(2i+1)\pi}{N}$ as our set of guiding angles on the same $N$-RoSy from (b) will result in one direction. These facts together indicate that if we assign each $V_i$ by using guiding angles of $\phi + \frac{2i\pi}{N}$ ($i = 0, 1, \ldots N-1$), all directions at every point will be captured, the breaking points will be in the same locations in each $V_i$, and $V_i$ and $V_i'$ will have nonoverlapping sets of breaking points (except at singularities).

$$l_{s,\phi} = argmin_k \left| \theta + \frac{2k\pi}{N} - \phi \right|. \quad (4)$$

Intuitively, $v_{s,\phi}$ is the member vector of $s$ that has the smallest angular difference with $\phi$. $v_{S,\phi}$ is a vector field that is generated by selecting a member vector in this fashion everywhere in the domain of $S$. It is well defined at nonsingularity points and continuous where only one minimum exists (Fig. 4a). On the other hand, when there are two minima, we have two candidate member vectors which are $2\pi/N$ apart, an ambiguity which leads to visual discontinuity (Fig. 4b). In these cases, we choose the one such that $\theta + \frac{2k\pi}{N} - \phi > 0$. For a given $\phi$, the points where there are two minima for $S$ will be referred to as *breaking points* with respect to $\phi$.

We then assign $V_i(\mathbf{p}) = v_{S(\mathbf{p}),\frac{2i\pi}{N}}$. $\{V_0, \ldots, V_{N-1}\}$ assigned in this fashion are mutually different and can collectively capture all member directions in the $N$-RoSy field everywhere in the domain. While there will be discontinuities in each $V_i$, they will be in the exact same locations.

Further, note that for every two fields $V_i$ and $V_j$, $V_i$ is a $\frac{2\pi(i-j)}{N}$ rotation of $V_j$. Thus, in practice, we only have to assign $V_0$ in the way described above, and $V_1, \ldots, V_{N-1}$ can be assigned as rotations of $V_0$.

We now generate $N$ LIC images $I_0, \ldots, I_{N-1}$, one for each of the respective vector fields $V_0, \ldots, V_{N-1}$ (Figs. 5a, 5b, and 5c). Once we have the $N$ images, we compose them by blending them uniformly into a new image $I = \frac{1}{N}\sum_{j=0}^{N-1} I_j$, which captures all the $N$ directions everywhere in the field (Fig. 5d). Note, however, that due to the discontinuities in our vector fields, we have artifacts at the corresponding locations in $I$ (Fig. 5d).

We thus generate another set of vector fields $\{V_0', V_1', \ldots, V_{N-1}'\}$, where $V_i' = v_{S,\frac{(2i+1)\pi}{N}}$. While these vector fields also contain visual discontinuities, the discontinuities (set of breaking points) will also be in the same places (regardless of $i$) that are different from the discontinuities in $V_0, \ldots, V_{N-1}$ (Figs. 4b and 4c).

Now, when we generate LIC images $I_0', \ldots, I_{N-1}'$ (Figs. 5f, 5g, and 5h) from this second set of vector fields, and the image $I' = \frac{1}{N}\sum_{j=0}^{N-1} I_j'$, all of the artifacts appear in complementary locations to those from $I$ (Fig. 5i). The intersection of the two sets of artifacts (breaking points) consists of only the singularities in the field. We can then blend $I$ and $I'$ so that in places where visual artifacts occur in one image we will use corresponding regions in the other image. There are different ways in which this can be achieved. We choose to adapt the weighting scheme of Zhang et al. [24], which corresponds to the case N = 2. In this scheme, the weighting functions are $\cos^2 N\theta$ and $\sin^2 N\theta$ where $\theta$ is angular component of one of the member vectors at the corresponding pixel location (Fig. 5e). We chose these blend weights because they will always sum to 1, because the regions where there are artifacts in $I$ correspond to the places where $\cos^2 \frac{N\theta}{2} = 0$, and because the regions where there are artifacts in $I'$ correspond to the places where $\sin^2 \frac{N\theta}{2} = 0$. One can also easily modify this weight so that it is binary valued (either 0 or 1) and the color of a pixel in the composited image will come from either I or I', though the difference between the results these two weights produce is virtually imperceptible.

When $N$ is even, we can save the computational cost by observing that each of the $N$ directions at every point in $S$ is the opposite of another direction at the same point, i.e., $V_i = -V_{i+N/2}$. Because of how the LIC algorithm works, we need only generate $\frac{N}{2}$ fields $(V_0, \ldots, V_{\frac{N}{2}-1})$ and images $(I_0, \ldots, I_{\frac{N}{2}-1})$ to visually capture all directions.

The result, shown in Fig. 5j, is an image where the artifacts due to vector field discontinuities are no longer visible (except very close to singularities), and where all directions at every point in the field are captured. Our technique works for all values of $N$, including when $N = 1$ and $N = 2$ (though when $N = 1$, there will not be any breaking points, and thus generating the image $I'$ is unnecessary). Also, note that the image in Fig. 5j is actually contrast corrected by the process to be described in Section 5.

Color can be integrated into LIC with relative ease, as has been done in several other works [12], [14], [15]. We find that with the larger amount of directional information present in $N$-RoSy fields, color often helps the LIC streaks in the final image stand out better against each other, making the directions at each point more apparent (Fig. 11).

To perform a colored version of our algorithm, we simply generate three different black and white noise textures, map each to a color channel, and use this "colored" noise image as the initial LIC image in the steps above. Note that this is the same as performing LIC three times in parallel.

## 5 Contrast Correction

Uniformly blending the images $I_0, \ldots, I_{M-1}$ (where $M$ is equal to $N$ or $N/2$, depending on whether $N$ is odd or even) can result in $I$ having lower contrast than each $I_j$, making the image appear faded and the directions at each difficult

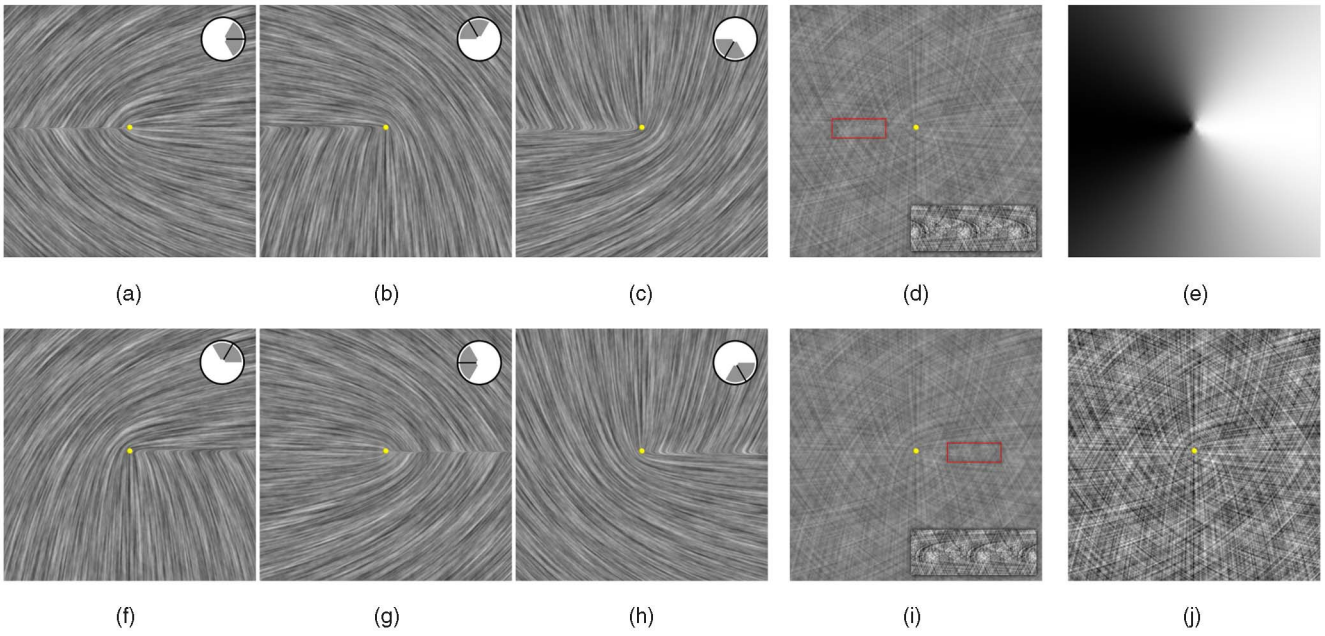



Fig. 5. Our visualization algorithm is demonstrated with an example 3-RoSy field $S$. In (a), (b), and (c), we applied the LIC algorithm to $V_0$, $V_1$, and $V_2$ (the guiding angle for each is shown in the upper-right corner) to obtain $I_0$, $I_1$, and $I_2$, respectively. Notice that while (a), (b), and (c) provide a complete coverage of the streamlines passing through any regular point in the domain, they have the same regions of breaking points (left $X$-axis). By blending them uniformly, we obtain $I$ (d), a visualization of $S$ with visual artifacts in the same place (a closeup of the artifact, highlighted in red, is seen as an inset with the contrast enhanced; note the curving patterns in a region that should be regular). To remedy the problem, we also apply the LIC algorithm to $V_0'$, $V_1'$, and $V_2'$, generating the images $I_0'$ (f), $I_1'$ (g), and $I_2'$ (h), and blend them uniformly to obtain $I'$ (i). The visual artifacts in $I'$ appear on the right side (again, a closeup is inset) of the $X$-axis. By blending $I$ and $I'$ using the weight map $w$ (e), we obtain the final image in (j) in which the artifacts due to field discontinuities are no longer visible. Note that the image in (j) has had its contrast corrected via the transformation described in Section 5.

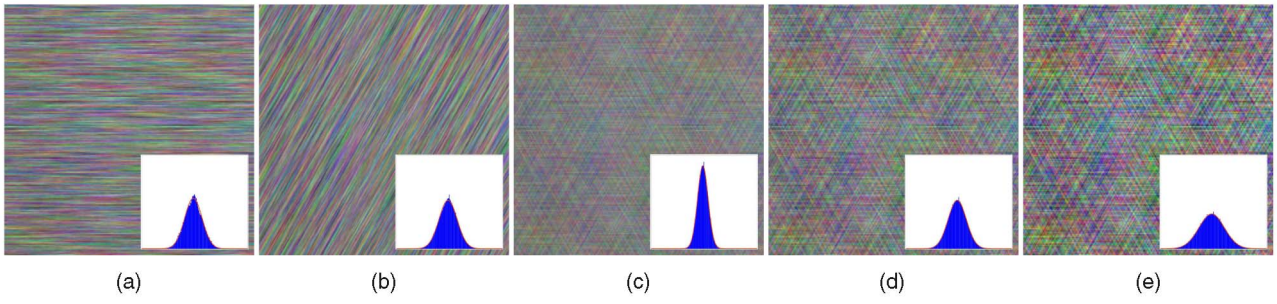



Fig. 6. Here we demonstrate our contrast correction and enhancement for a constant 6-RoSy field. Note that the inset histograms only show the red channel of the color images, but the distributions for the other two channels are similar. The three member vector field images $I_0$ (a), $I_1$ (b), and $I_2$ (not shown) are blended uniformly to get $I$ (c). However, the pixel signals in $I$ have reduced variance, which makes the image appear washed out and gray, and so we apply the transformation in (6) to get $I_{corrected}$ (d), whose pixel signals have the same variance as those in $I_0$ and $I_1$. We can then (optionally) expand the variance further, by using the same transformation with a larger scaling factor to get the image in (e).

to discern (Fig. 5d). Here, we explore this issue further, and provide a systematic solution which will eliminate this contrast loss without performing any image analysis.

Recall that, given a pixel $P$ in the image $I$ (defined in Section 4) and $M$ signals at that pixel $P_0, \ldots, P_{M-1}$ (one from each respective $I_j$), we have $P = \frac{1}{M}\sum_{j=0}^{M-1} P_j$. Given that each $I_j$ is generated from the same initial noise texture with the same LIC parameter L, $P_0, \ldots, P_{M-1}$ are normally distributed random variables with the same mean $\mu$ and and variance $\sigma^2$ (that is, $P_j \sim \mathbf{N}(\mu, \sigma^2)$, where $0 \leq j \leq M-1$). Consequently, $P$ is also normally distributed as follows:

$$P \sim \mathbf{N}\left(\mu, \frac{\sigma^2}{M}\right). \tag{5}$$

Note that the mean has remained the same (in our experiments the mean is always almost exactly 0.5), but the variance has shrunk by a factor of $\frac{1}{M}$. This explains why the images $I$ and $I'$ appear more gray than $I_0, \ldots, I_{M-1}$ and $I_0', \ldots, I_{M-1}'$ (Figs. 6a, 6b, and 6c).

Fortunately, in order to "correct" $P$ so that its distribution is the same as that of $P_0, \ldots, P_{M-1}$, we need only apply the following transformation:

$$P_{corrected} = \sqrt{M}(P - \mu) + \mu. \tag{6}$$

Note that $P_{corrected} \sim \mathbf{N}(\mu, \sigma^2)$, and our empirical data support this hypothesis (Fig. 6d). Further, note that if we apply this transformation to every pixel in $I$ (and $I'$) using our empirical value of $\mu = 0.5$, this amounts to contrast scaling. Note that the scaling can be applied either before or

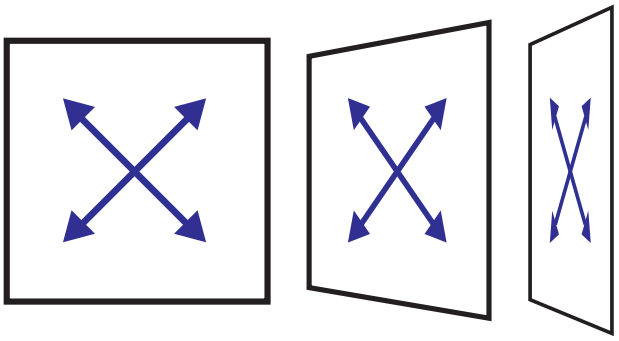
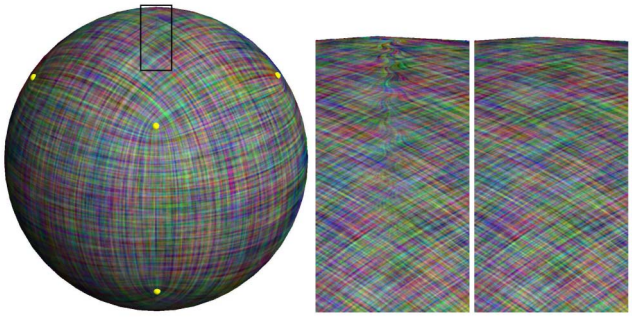
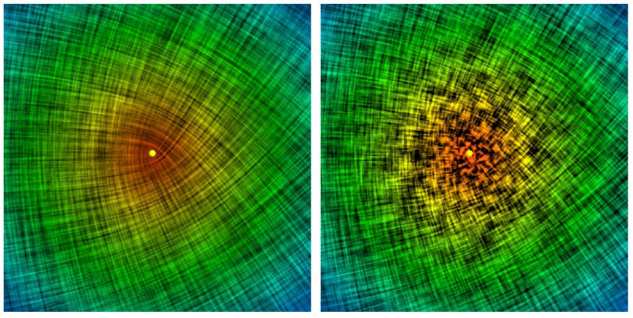


Fig. 7. On the left, we see a 4-RoSy on a plane that is parallel to the view plane; all of its directions are $\frac{2k\pi}{4}$ rotations of the others, where $k$ is an integer. As the plane is rotated so that it is nearly orthogonal to the view plane (middle and left), these relationships no longer hold.



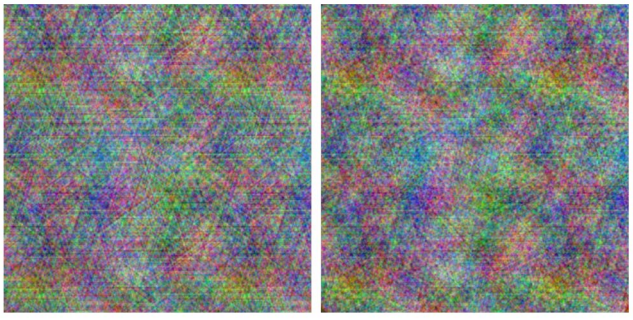
(a) (b)

Fig. 8. (a) A 4-RoSy field on a sphere that forms a cube pattern. In the left side of (b), a closeup is shown of an artifact in the visualization on the left (the region indicated by the black box in (a)). We propose a method to fix the problem (b) in Section 6.



(a) (b)

Fig. 9. Our method can represent continuous $N$-RoSy fields by mapping the magnitude to color (a), varying the length of the streamline traced per pixel, or both (b).



(a) (b)

Fig. 10. We find that due to the abundance of directional information at each point, the effectiveness of our algorithm is decreased as $N$ reaches higher numbers, such as 5 (a) and 7 (b).

after image blending; this is because that we do not have to perform any image analysis on the blended image.

On the other hand, we do not perform contrast correction when blending the images $I$ and $I'$. Because of how these images are generated (using the same noise image, and the same field), $I$ and $I'$ end up being almost exactly identical, except in the regions near their artifacts caused by breaking points (Figs. 2b and 2c and Figs. 5d and 5i). Thus, there is very little interference between pixel signals when blending these two images, leading to little loss of contrast in the resulting image.

Finally, when performing the colored version of our algorithm, the above operations can be performed in exactly the same manner, but per channel; in this case, we are correcting a loss of variance in image hue (Fig. 6c).

## 6 Extension to Surfaces

The algorithm described in Section 4 can be adapted to visualize $N$-RoSy fields on surfaces with relative ease, by performing almost all of the previously described operations in image space and adding diffuse lighting; this is akin to techniques proposed by Van Wijk [22] and Laramee et al. [10]. Fig. 11 shows our results on several surfaces. The implementation is almost identical to that of the planar case, except that we must take care to use the correct set of member vectors; this is because the member vectors at a point on the surface are rotations of integer multiples of $\frac{2\pi}{N}$ of each other in tangent space, but their projections onto the view plane do not necessarily share this property (Fig. 7).

This distinction, however, actually leads to an issue which lessens the effectiveness of our artifact reduction step (see Section 4). Because the projections of the member vectors in the view plane do not preserve the rotational relationships between them, the weights we use to to blend $I$ and $I'$ are less effective in places where the surface normal is not parallel to the view direction. This can lead to the reappearance of the artifacts described in Section 4 in the corresponding regions of the image (Fig. 8 left and middle). While these artifacts always appear in regions that are not directly facing the view plane, they can nonetheless be distracting, and so we propose a method to eliminate them here.

At each point $\mathbf{p}$ on the surface whose normal is not parallel to the normal of the view plane, there is a unique minimal rotation $R$ that takes a vector in the tangent plane at $\mathbf{p}$ to be parallel to the view plane. This 3D rotation will introduce a rigid, orientation-preserving transformation between tangent vectors from the two planes: the tangent plane at $\mathbf{p}$ and the view plane. Consequently, an $N$-RoSy can be maintained during this transformation. Using this transformation, we simply rotate each $N$-RoSy into the view plane, and then perform the field decomposition and weight computation on these rotated RoSys. The resulting vector fields can then be translated back into tangent space by the inverse rotation $R^{-1}$. As Fig. 8b shows, this completely eliminates the artifacts.

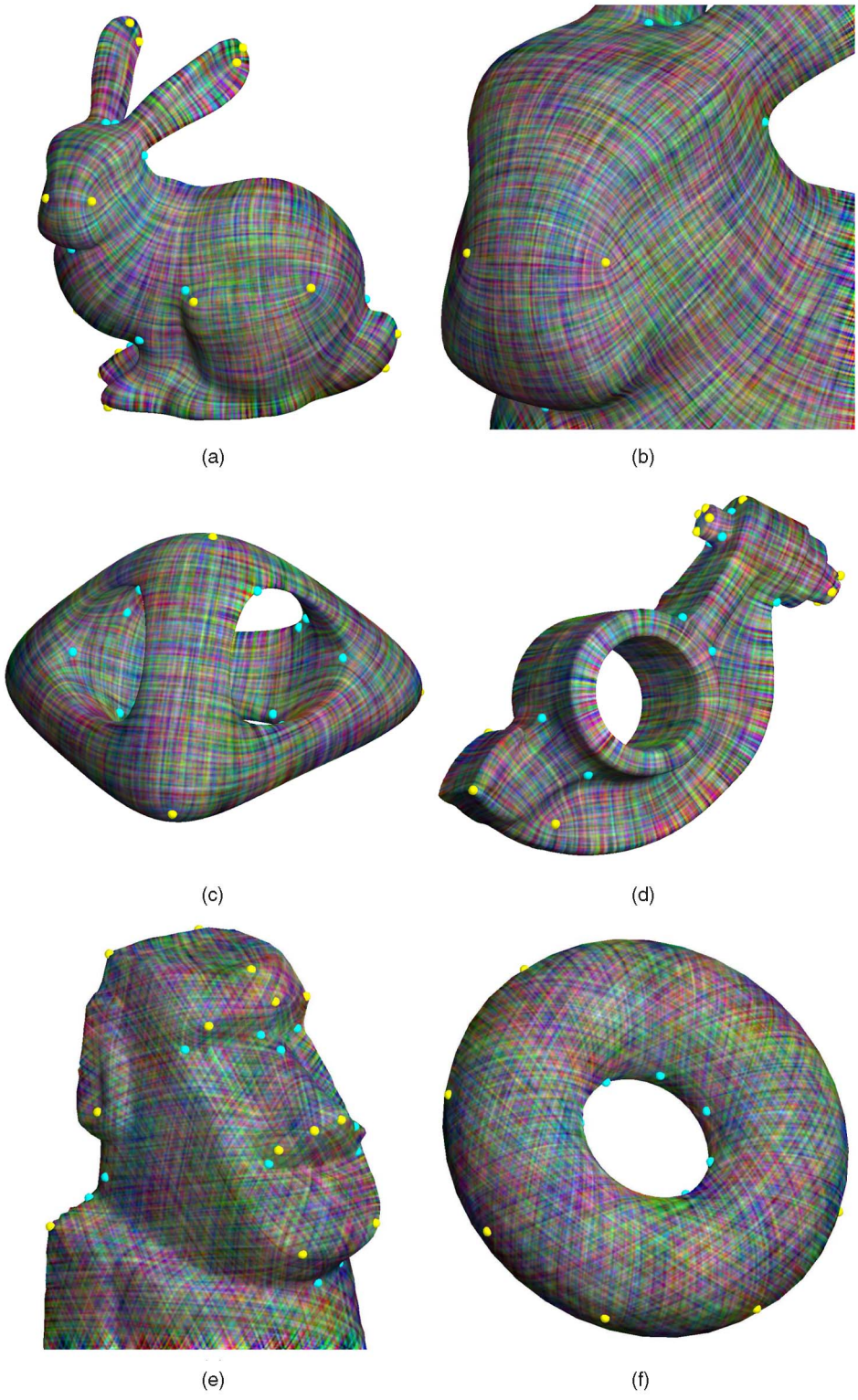



Fig. 11. Here, we see 4-RoSy fields on the Bunny (a)-(b), triple-torus (c), and Rocker Arm models visualized using our system. Color can be added by performing our algorithm three times in parallel and mapping each output to a color channel, resulting in a plaid-like pattern; alternatively color can be reserved for for visualizing other values, such as field magnitude. In (e), we see a 6-RoSy field field on the Moai statue and another on the torus in (f).

## 7 Results

Our algorithm produces accurate visual representations of $N$-RoSy fields at interactive frame rates. Table 1 shows rendering times for our algorithm compared to those for a brute-force implementation of $N$-RoSy LIC. Recall that a brute-force implementation simply alters the tracing method of the LIC algorithm so that all the $N$ directions are examined at each integration step, and the one that most closely matches the direction at the previous step is chosen. Further, $N$ ($N/2$ when $N$ is even) streamlines are traced for

each pixel (one for each direction at that pixel), used as convolution kernels to anisotropically blur the noise texture, and the result from these kernels is then uniformly blended to get the final pixel color. In this method, there is no need to decompose the original field into vector fields. However, choosing the best direction at every tracing step requires a great deal of branching, making this algorithm suboptimal for parallel architectures, such as graphics cards. In contrast, our algorithm benefits greatly from parallelization.

All of our experiments were performed on a desktop computer with a dual-core 3.73 GHz Intel Pentium D processor and an NVIDIA GeForce 8600 GTS video card. Both algorithms were accelerated using the programmable graphics hardware, and all renderings were done at a resolution of $512 \times 512$ using the surface implementations of these algorithms. The two methods produce virtually identical images. Our algorithm outperforms the brute-force implementation in all of our experiments, and maintains interactive frame rates for fields with values of $N = 2, 3, 4$, and 6, even for large meshes.

Not only is our algorithm faster, it also scales much better with $N$; as $N$ and the number of directions at each point increase, the brute-force algorithm suffers greatly due to increased branching. Our algorithm incurs only a moderate additional cost associated with generating additional LIC images. Note the cases where $N = 4$ outperform the ones where $N = 3$; this is because when $N$ is even, we only need generate $N/2$ images to represent all $N$ directions everywhere in the field. The brute-force implementation takes advantage of this as well, but is still outperformed by our method in every experiment.

Also note that, at a fixed resolution, a large increase in mesh size (from 12K to 200K triangles) results in only a relatively small offset in rendering time for both algorithms. This indicates that the rendering time is pixel bound, which is to be expected for image space methods.

In all of the applications that we know of involving $N$-RoSy fields (where $N > 2$), it is the directions of the RoSy that are of ultimate importance. On the other hand, the magnitude is equally important in vector and tensor field visualization ($N = 1, 2$). Our framework inherits the flexibility of texture-based methods in which the magnitude can be conveyed by color, varying length of streamlines, or both (Fig. 9).

The effectiveness of this visualization technique begins to lessen as $N$ increases, and computational cost grows; it becomes difficult to discern meaningful information in cases where $N = 5, 7$, and beyond (Fig. 10). Fortunately, for the known applications involving $N$-RoSy fields, such as surface parameterization, remeshing, and nonphotorealistic rendering, only fields where $N = 2$, 4, and 6 have found uses thus far.

## 8 Conclusions

We have presented an interactive texture-based visualization algorithm for $N$-RoSy fields on surfaces. Our algorithm can show all the $N$ directions at every point and captures the features of $N$-RoSy fields, such as singularities. In our technique, an $N$-RoSy field is decomposed into a number of vector fields, which are visualized using the LIC techniques and then combined. We note that the number of vector fields used is only twice that of the minimal number of images needed in order to show all the N directions.

Further, we have observed that the pixel values in LIC images are normally distributed random variables, allowing us to use concepts from probability theory to correct the loss of contrast usually associated with the blending of such images. Contrast can also be further enhanced if desired.

In the future, we wish to investigate efficient contrast adjustment when the input images are not gray scale and have different hues. One example of this is to visualize both the major and minor eigenvector fields of a second-order tensor. In addition, we will consider how to best adjust the contrast given lighting conditions. Also of interest are new decomposition strategies that will lead to fewer images to blend, thus increasing the interactivity. Finally, $N$-RoSy fields are special types of $N$th-order symmetric tensors. We wish to extend our method to handle symmetric $N$th-order tensor field visualization, in which angles between neighboring member vectors are no longer constant.

## Acknowledgments

The authors would like to thank the following groups for the 3D models which they have made available: Marc Levoy and the Stanford Graphics Group, Cyberware, and the AIM@Shape Repository. They also thank Konrad Polthier, Felix Kälberer, and Matthias Nieser for the use of their images, and Randall Rauwendaal and Jonathan Dodge for their help in proofreading the paper. The discussions with Greg Turk have been most valuable, and the incredibly astute and constructive criticism and comments from the editor, Dr. Jarke van Wijk, and the anonymous reviewers have definitely resulted in a better paper for which the authors are most grateful. This work is partially supported by the US National Science Foundation (NSF) CCF-0546881 and CCF-0830808. Jonathan Palacios is also supported by an NSF IGERT Fellowship DGE-0333257.

## References

[1] P. Alliez, D. Cohen-Steiner, O. Devillers, B. Lévy, and M. Desbrun, "Anisotropic Polygonal Remeshing," *ACM Trans. Graphics*, vol. 22, no. 3, pp. 485-493, July 2003.

[2] D. Bommes, H. Zimmer, and L. Kobbelt, "Mixed-Integer Quadrangulation," *ACM Trans. Graphics*, vol. 28, no. 3, pp. 1-10, 2009.

[3] B. Cabral and L.C. Leedom, "Imaging Vector Fields Using Line Integral Convolution," *Proc. ACM SIGGRAPH*, pp. 263-270, 1993.

[4] A. Hertzmann and D. Zorin, "Illustrating Smooth Surfaces," *Proc. ACM SIGGRAPH*, pp. 517-526, Aug. 2000.

[5] I. Hotz, L. Feng, H. Hagen, B. Hamann, B. Jeremic, and K. Joy, "Physically Based Methods for Tensor Field Visualization," *Proc. IEEE Visualization Conf.*, pp. 123-130, 2004.

[6] E. Hsu, "Generalized Line Integral Convolution Rendering of Diffusion Tensor Fields," *Proc. Int'l Soc. for Magnetic Resonance in Medicine (ISMRM)*, vol. 790, 2001.

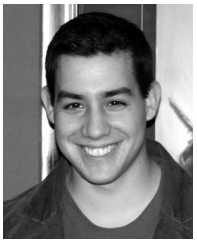
[7] F. Kälberer, M. Nieser, and K. Polthier, "Quadcover—Surface Parameterization Using Branched Coverings," *Proc. Eurographics*, pp. 375-384, 2007.

[8] Y.K. Lai, M. Jin, X. Xie, Y. He, J. Palacios, E. Zhang, S.M. Hu, and X. Gu, "Metric-Driven Rosy Field Design and Remeshing," *IEEE Trans. Visualization and Computer Graphics*, vol. 16, no. 1, pp. 95-108, Jan./Feb. 2010.

[9] R.S. Laramee, H. Hauser, H. Doleisch, B. Vrolijk, F.H. Post, and D. Weiskopf, "The State of the Art in Flow Visualization: Dense and Texture-Based Techniques," *Computer Graphics Forum*, vol. 23, pp. 203-221, 2004.

[10] R.S. Laramee, B. Jobard, and H. Hauser, "Image Space Based Visualization of Unsteady Flow on Surfaces," *Proc. 14th IEEE Visualization (VIS) Conf.*, p. 18, 2003.

[11] W.C. Li, B. Vallet, N. Ray, and B. Levy, "Representing Higher-Order Singularities in Vector Fields on Piecewise Linear Surfaces," *IEEE Trans. Visualization and Computer Graphics,* vol. 12, no. 5, pp. 1315-1322, Sept./Oct. 2006.
[12] Z. Liu, R. Moorhead, II, and J. Groner, "An Advanced Evenly-Spaced Streamline Placement Algorithm," *IEEE Trans. Visualization and Computer Graphics,* vol. 12, no. 5, pp. 965-972, Sept./Oct. 2006.
[13] J. Palacios and E. Zhang, "Rotational Symmetry Field Design on Surfaces," *Proc. ACM SIGGRAPH Papers,* p. 55, 2007.
[14] N. Ray, W.C. Li, B. Lévy, A. Sheffer, and P. Alliez, "Periodic Global Parameterization," *ACM Trans. Graphics,* vol. 25, no. 4, pp. 1460-1485, 2006.
[15] N. Ray, B. Vallet, W.C. Li, and B. Levy, "N-Symmetry Direction Field Design," *ACM Trans. Graphics,* vol. 27, no. 2, pp. 1-13, May 2008.
[16] A.R. Sanderson, C.R. Johnson, and R.M. Kirby, "Display of Vector Fields Using a Reaction-Diffusion Model," *Proc. IEEE Visualization Conf.,* pp. 115-122, 2004.
[17] A. Schiftner, M. Hobinger, J. Wallner, and H. Pottmann, "Packing Circles and Spheres on Surfaces," *ACM Trans. Graphics,* vol. 28, no. 5, pp. 1-8, 2010.
[18] H.W. Shen and D.L. Kao, "A New Line Integral Convolution Algorithm for Visualizing Time-Varying Flow Fields," *IEEE Trans. Visualization and Computer Graphics,* vol. 4, no. 2, pp. 98-108, Apr./June 1998.
[19] D. Stalling and H.C. Hege, "Fast and Resolution Independent Line Integral Convolution," *Proc. ACM SIGGRAPH,* pp. 249-256, 1995.
[20] J.J. van Wijk, "Spot Noise Texture Synthesis for Data Visualization," *Proc. ACM SIGGRAPH,* pp. 309-318, 1991.
[21] J.J. van Wijk, "Image Based Flow Visualization," *ACM Trans. Graphics,* vol. 21, no. 3, pp. 745-754, July 2002.
[22] J.J. van Wijk, "Image Based Flow Visualization for Curved Surfaces," *Proc. IEEE Visualization,* G. Turk, J. van Wijk, and R. Moorhead, eds., pp. 123-130, Oct. 2003.
[23] L.Y. Wei and M. Levoy, "Texture Synthesis over Arbitrary Manifold Surfaces," *Proc. ACM SIGGRAPH,* pp. 355-360, 2001.
[24] E. Zhang, J. Hays, and G. Turk, "Interactive Tensor Field Design and Visualization on Surfaces," *IEEE Trans. Visualization and Computer Graphics,* vol. 13, no. 1, pp. 94-107, Jan. 2007.
[25] X. Zheng and A. Pang, "Hyperlic," *Proc. 14th IEEE Visualization (VIS) Conf.,* p. 33, 2003.



**Jonathan Palacios** is currently working toward the PhD degree in the Department of Electrical Engineering and Computer Science, Oregon State University, under the supervision of Dr. Eugene Zhang. His current research interests include computer graphics, geometric modeling, symmetry, and higher order tensor field visualization and analysis. He is a US National Science Foundation (NSF) IGERT fellow, and a member of the ACM.

**Eugene Zhang** received the PhD degree in computer science from the Georgia Institute of Technology in 2004. He is currently an associate professor at Oregon State University, where he is a member of the School of Electrical Engineering and Computer Science. His research interests include computer graphics, scientific visualization, and geometric modeling. He received a US National Science Foundation (NSF) CAREER Award in 2006. He is a member of the IEEE Computer Society and a senior member of the ACM.