Interactive Tensor Field Design Based on Line Singularities

Jiazhou chen*, Qi Lei†, Fan Zhong‡ and Qunsheng Peng†

*College of Computer Science, Zhejiang University of Technology, Hangzhou, P.R.China
†State Key Lab of CAD&CG, Zhejiang University, Hangzhou, P.R.China
‡School of Computer Science and Technology, Shandong University, Jinan, P.R.China

***Abstract*—Tensor field design plays an essential role in various computer graphics applications. One of the main challenges in field design is how to obtain a smooth field preserving meaningful singularities presented in the original scene. Compared with well-studied point singularities, line singularities, commonly occur on object boundaries and occluding contours, are not exploited enough for field design in previous work. In this paper, we discuss the definition of line singularities, and introduce a line-singularity-based interactive tensor field design method, allowing the user to design feature-preserving tensor fields with less effort and to preserve both input singularities and user-specified stroke directions. To avoid introducing extra interaction burdens to the user, our method automatically locates line singularities using a geodesic-based segmentation. We demonstrate the capabilities of our method on tensor field design with various nonphotorealistic rendering applications and the real-time performance accelerated on GPU.**

***Keywords*-tensor field design; line singularity; nonphotorealistic rendering;**

I. Introduction

A tensor field assigns a tensor to each point of a space. It has been used to guide the stroke placement in painterly rendering [1], [2], [3], to steer the orientation of texture coordinates in texture synthesis [4], [5], to align the smooth parameterization [6], [7] and to control the quadrangulation [8], [9] for triangulated surfaces, thus plays a dramatic role in Computer Graphics community. Figure 1 gives us such an example to illustrate what a tensor field is and how it reveals features. A closer look of these applications reveals that a high-quality tensor field should 1) preserve the meaningful singularities, 2) be smooth at other non-singular places, 3) preserve structural features of the original scenes. As already noticed by authors of these publications, a tensor field automatically derived from an image or a surface hardly meets all requirements. In this paper, we are concerned with providing an interactive method that allows the user to easily design such a high-quality field.

A number of methods have been proposed to design a tensor field interactively (see Section II). In particular, several recent methods offer topology control based on point singularities where the tensor field is not defined [10], [11], [12]. An energy function, constrained by these singularities and user-specified directions, is optimized in these methods to guarantee the smoothness at non-singular places. However, these methods all assume that singularities are isolated in their small neighborhoods. From our observation, singularities may connect with each other, and form a line of singularities on the object boundaries and occluding contours. In these situations, the isolation assumption prevents point-singularity-based methods from preserving structural features.

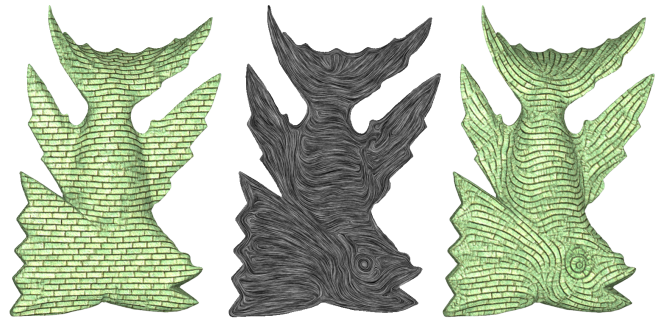


Figure 1. **Tensor field** has been used in texture synthesis to steer the orientation of texture coordinates.

To overcome the isolation assumption, recent region-based approaches first decomposed the field domain into smaller regions, and introduced discontinuity across the boundaries of different regions [13], [14], [15]. However, uniformly specifying the discontinuity for a sub-region can not always produce feature-preserving field. Take one petal in Figure 5 as an example, the nerve field on the original image shows that it is discontinuous across the occluding contours, while continuous around other part of its boundary. Therefore, the discontinuities should be specified non-uniformly according to the cause of the boundary and the user design intention.

To solve the aforementioned issues, we generalize the singularity definition from isolated points in previous work to line singularity that defines the discontinuities non-uniformly around the object boundaries in this paper. Based on this generalization, we introduce an interactive method on tensor field design. By employing a geodesic-based segmentation, our method uses cubic spline fitting of the segmented boundary to automatically locate line singularities. We then construct a tensor field for each region by combining constraints from these line singularities, user-specified strokes and image colors.

Compared with previous work, our method makes the

following contributions:

- We propose a generalization of singularity definition, and introduce an interactive tensor field design method that allows the user to design smooth yet feature-preserving tensor fields;
- A geodesic-based segmentation is employed to automatically locate line singularities, which relieves the interaction burdens for the user;
- With an implementation on modern GPU, our designing tool provides the user with an instant feedback on each interaction operation.

The rest of this paper is organized as following: after reviewing related work in Section II, we define line singularity mathematically in Section III. The algorithm will be presented in Section IV. Section V shows our results and comparisons with other state-of-the-art methods, and Section VI gives conclusion and future work.

II. Related work

There is a considerable literature on interactive methods to design various types of direction fields, such as vector fields, line fields, tensor fields and N-symmetry directional fields. Their definitions are close but differ from each other. Existing design methods for these fields can be classified into three categories: point-singularity-based, stroke-based and region-based. Here we give a brief review over each category for tensor fields and other related fields.

Point singularity is the most noticeable characteristic of a tensor field, also called a degenerate point [16]. To make use of this characteristic, Zhang et al. employed Conley index theory and introduced an effective pair cancellation method that can simplify a tensor field with simple interactions [16]. For vector fields, Chen et al. proposed a unified design framework that allows the user to cancel both singularity points and periodic orbits [17], [18]. To design direction fields with higher-order rotational symmetry, Ray and his colleges introduced a concise yet complete interpolation scheme of direction fields with higher-order point singularities on arbitrary triangulated surfaces based on Poincaré-Hopf theorem [19], [11], [12].

Though point-singularity-based method allows the user to design a field with arbitrary topology, it is neither intuitive nor easy to decide the location, quantity and indices of point singularities, as it asks for where and how the direction vanishes rather than what the direction is [15]. Therefore, directly specifying a direction stroke in the space has become a requisite in many recent work [10], [16], [20]. To interpolate stroke directions to the whole designing space, Zhang et al. generate an individual basis field for each user-specified direction separately [16], and combine their tensors using a weighted sum. For vector fields, Fisher et al. compute a harmonic vector field with a sparse set of user-specified directions [10], Hays and Essa interpolate the vector field using image gradient directions with strongest magnitudes [2], and Xu et al. diffuse the vectors from the tangential directions of prominent feature lines [5].

With both point-singularity-based method and stroke-based method, each singularity or user-specified direction has an unlimited influence over the whole field domain. It reduces their capabilities of local field control, since fields on different objects may have completely different topologies. To solve this issue, Theisel introduced separatrices as another topological component of a vector field to separate the field behaviors spatially [21], which was also used in quad meshing [6] and rotational symmetry field design [22]. However, separatrix is mathematically defined as a tangent curve. That means the field around the separatrix is always continuous with its tangential directions. Unfortunately, the discontinuity is often desirable in many situations, as discussed in [13], [23], [14], [3]. For this sake, Chen et al. decompose the tensor field domain into smaller regions, and then design individual tensor fields within each sub-region independently [13]. The idea about introducing discontinuity across the boundaries of segmented sub-regions has also been adopted in many painterly rendering approaches [23], [14], [3]. Since a closed separatrix can be considered as the region boundary with continuous fields, we categorize separatrix-based methods into region-based ones.

As pointed in Section I, uniformly specifying the discontinuity for a sub-region can not always produce feature-preserving tensor field. The discontinuities should be specified non-uniformly according to the cause of the boundary and the user design intention. To solve this problem, we generalize the singularity definition from isolated points in previous work to line singularity, that allows the user to define the discontinuity non-uniformly around the object boundaries.

III. Definition of line singularity

A vector field over a closed region of an image $D \subset \mathbb{R}^2$ defines a vector $\mathbf{v}$ at each point along with geometric or rendering elements that can be placed. It is also called a direction field, better to be called a line field when considering $\mathbf{v}$ is un-oriented [15], and called a tensor field if $\mathbf{v}$ is represented as the major eigenvector of a 2×2 tensor matrix [16]. In this paper, we employ the tensor field, due to the weighted sum of multiple fields and the curvature computation using the tensor matrix.

A point $\mathbf{p}$ in the field domain is defined as a point singularity if $\mathcal{T}(\mathbf{p}) = 0$ and $\mathcal{T}(\mathbf{q}) \neq 0$ for any $\mathbf{q} \in \delta(\mathbf{p})$, where $\delta(\mathbf{p})$ represents the immediate neighborhood of point $\mathbf{p}$. The isolation of point singularities limits them to describe all tensor fields in natural scenes. In the places where singularities are connected with each other, such as on the object boundaries and occluding contours, the mathematical theories built on point singularity are not suitable any more.

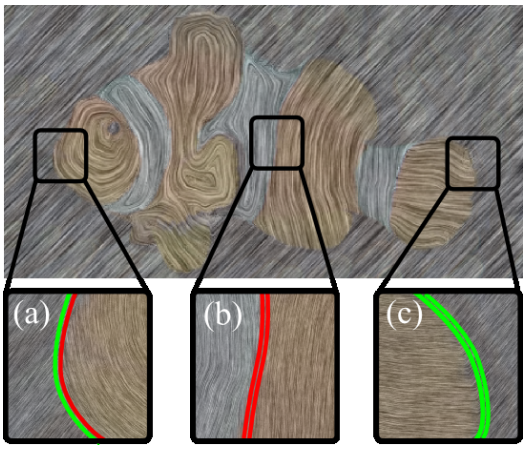


Figure 2. **Examples of boundaries with different continuity properties in clown fish image.** Tensor fields on only one side are discontinuous for single line singularity (LS) (a); tensor fields on both sides are discontinuous for dual LS (c); and neither of them is discontinuous for separatrix (b). Tensor fields are visualized using Line Integral Convolution (LIC) [24] of tangential directions, blended with the input image. And line singularities and separatrices are visualized as dual curves: red curves represent continuity while green ones represent discontinuity.

Rather than studying the index number and the rotational symmetry of point singularity as much previous work, we are more interested in the connected set of singularities. Since we focus on singularities on the object boundaries, such a singularity union will not be a 2D singular area, it can be always represented as a set of singularity points that form a line. We call it a line singularity (LS).

For a segmented region $\Omega \in \mathbb{R}^2$, a line singularity on its boundary $\partial\Omega$ can be defined as a connected path, all of whose points are singular, mathematically equals to $\mathbf{L} = \{\mathbf{p} \in \partial\Omega \mid \mathcal{T}(\mathbf{p}) = 0 \; and \; connect(\mathbf{p}, \mathbf{q}) \; for \; \forall \mathbf{q} \in \mathbf{L}\}$, where $connect(\mathbf{p}, \mathbf{q})$ means part of points in $\mathbf{L}$ can form a path starting from $\mathbf{p}$ and ending at $\mathbf{q}$. This path may be very long, even has branching lines. We thus segment it into short paths, and fit them using cubic splines. The subdivision and fitting allow the user to easily control a line singularity, including its location and continuity. The details about the generation of line singularity will be presented in Section IV-A.

On one hand, since points on line singularities have zero tensor and are not isolated as point singularities, they thus have no direction information; on the other hand, the path itself has tangential directions, which can impact the tensor field on non-singular points. Considering the continuity between the tangential directions of a line singularity and directions of its surrounding non-singular points, we categorize these paths on the region boundary $\partial\Omega$ into three types:

- **Dual LS:** Tensor fields on both sides of the boundary are discontinuous with its tangential directions;
- **Single LS:** Tensor fields on only one side of the boundary is discontinuous with its tangential directions;
- **Separatrix:** Tensor fields on both sides of the boundary are continuous with its tangential directions.

Figure 2 gives examples of these three types of region boundaries. Here we clarify the relationship between our line singularity and other boundary discontinuity definitions in previous work. Strictly speaking, separatrix can not be regarded as any line singularity, but separatrix and line singularity can easily switch to each other by adjusting the continuities on both sides. The introduction of line singularity also provides the possibility to completely categorize all types of region boundaries with different continuity properties. For concision purpose, we will use types of line singularity instead of types of boundary paths in the rest of this paper. Line singularity differs from the discontinuity in region-based design methods [13], [23], [14], [3], since their discontinuity can only be specified for a whole segmented region uniformly. Finally, the flow-out segments in [15] and primal sketches in [23] are in fact single line singularity.

IV. Algorithm

Line singularities produce tensor fields preserving better the structural features, but require more user interactions to create them, as pointed in region-based field design methods [15]. Instead of sketching out the region boundary by hand, we adopt a geodesic-based image segmentation to automatically locate line singularities. Line singularities are further fitted as cubic spline curves that support naturally various editing operations.

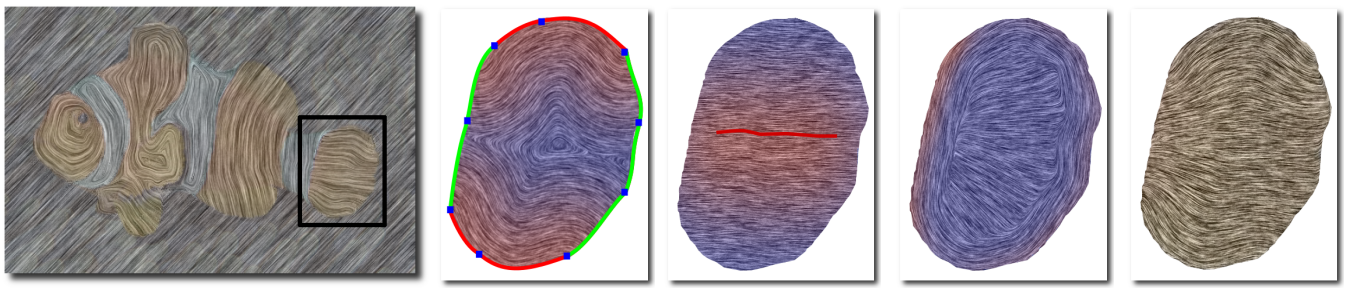
A. *Generation of line singularities*

To automatically locate line singularities, our method makes use of user-specified strokes for direction constraints. In other words, the user puts strokes over the reference image to control the tensor field, while our method automatically segments the influence region of these strokes, and takes the region boundary as the location of line singularities. Denoting the reference image as I, we create a binary mask image M, where $M(\mathbf{x}) = 1$ if point $\mathbf{x}$ is on user-specified strokes, and $M(\mathbf{x}) = 0$ if not. Then the unsigned geodesic distance of each point from the specified region $\mathcal{R} = \{\ \mathbf{x} \mid M(\mathbf{x}) = 1\}$ is defined as [25]:

$$D(\mathbf{x};\ M, \nabla I) \ = \min_{\{\mathbf{x}' | M(\mathbf{x}')=1\}} d(\mathbf{x}, \mathbf{x}'), \quad \text{with} \qquad (1)$$

$$d(\mathbf{a}, \mathbf{b}) = \min_{\mathbf{\Gamma} \in \mathcal{P}_{\mathbf{a},\mathbf{b}}} \int_0^1 \sqrt{||\mathbf{\Gamma}'(s)||^2 + \gamma^2 (\nabla I \cdot \mathbf{u})^2}\ ds \qquad (2)$$

with $\mathcal{P}_{\mathbf{a},\mathbf{b}}$ the set of all paths between points $\mathbf{a}$ and $\mathbf{b}$; and $\mathbf{\Gamma}(s) : \mathbb{R} \rightarrow \mathbb{R}^2$ indicating one such path, parameterized by $s \in [0, 1]$. The spatial derivative $\mathbf{\Gamma}'(s)$ is $\partial\mathbf{\Gamma}(s)/\partial s$. And the unit vector $\mathbf{u} = \mathbf{\Gamma}'(s)/||\mathbf{\Gamma}'(s)||$ is tangent to



(a) Final tensor field (b) Line tensor field (c) User tensor field (d) Color tensor field (e) Combination

Figure 3. **Combination of different tensor fields** zoomed in the tail fin of the clown fish image (a). (b) Line tensor field is constructed according to line singularities that are continuous inward (red solid splines); (c) user tensor field is computed using user-specified strokes (the red curve), and (d) color tensor field is automatically derived from the reference image. Their combining weights are shown by a red-blue color mapping, red represents high weights. Notice that unexpected tensor fields with low magnitudes are faded out in the combined tensor field (e). Note that tensor fields are visualized using LIC of tangential directions, not the gradients.

the direction of the path. The geodesic factor γ weighs the contribution of the image gradient versus the spatial distances. The image gradient here is computed by applying a Sobel filter to the grayscale of the input image. In this paper, we use $\gamma = 1$ for all results. We then clamp the geodesic distance by a threshold τ to segment the image $\Psi_i = \{\mathbf{x} \mid D(\mathbf{x}; M, \nabla I) \leq \tau\}$. Notice that a bilateral filter is employed before the segmentation to suppress the noises in the input image. The boundary of this region gives us an estimate of the location of line singularities.

To get a smooth line singularity that supports further combination and user editing, we have to convert the region boundary to vectorization representation. After implementing spline fitting with different orders, we found cubic spline curve is the most suitable choice for this task. We first take a traversal of the boundary points $\mathbf{p}_i$, $(i \in N)$, where N is the number of boundary points, and $\mathbf{p}_0 = \mathbf{p}_N$ due to the boundary closeness. Secondly, we compute the sharpness of each point by $s_i = (\mathbf{p}_{i-10} - \mathbf{p}_i) \cdot (\mathbf{p}_{i+10} - \mathbf{p}_i)/||\mathbf{p}_{i-10} - \mathbf{p}_i|| \cdot ||\mathbf{p}_{i+10} - \mathbf{p}_i||$, and pick up boundary points with local maximal sharpness as key points. To avoid too long interval between adjacent key points, we add a key point every 30 points when no sharp key point is found. Thirdly, we fit a cubic spline for every path between adjacent key points. Denoting this spline function as $\mathbf{f}(t) = \sum_{i=0}^{3} \mathbf{a}_i \cdot t^i$, we solve a least square problem on all boundary points between a pair of adjacent key points $\mathbf{p}_{K_i}$ and $\mathbf{p}_{K_{i+1}}$. To obtain proper parameters $\mathbf{a}_i$, we minimize the fitting error $\sum_{j=0}^{\ell}[\mathbf{f}(j/\ell) - \mathbf{p}_{K_i+j}]^2$, where ℓ is the point number from $\mathbf{p}_{K_i}$ to $\mathbf{p}_{K_{i+1}}$, and $\mathbf{p}_{K_i+\ell} = \mathbf{p}_{K_{i+1}}$.

After the cubic splines are fitted, the user has to decide the singularity type for each spline. By default, our method sets inward side of a spline to be continuous with the tangential directions of this spline, and discontinuous for outward side. We visualize the singularity type as dual curves: red curves represent continuity on one side of the line singularity while green ones represent discontinuity. The user can simply adjust the singularity type by selecting and changing the continuity of each side. Notice that since the segmentation is done sequentially, new segmented region will cover previous ones if they have large overlapping, and it will be considered as a new tensor field region if it has no interaction with any previous region.

B. *Tensor field construction*

Comparing with point singularity, we do not have mathematical tools for line singularity. The tensor field construction method we apply should naturally support the use of line singularities, allow the user to fully control the field behaviors both around the line singularities and far away from them, and also be fast enough. For these purposes, we compute three tensor fields separately, and combine them using distance-related weights, rather than solving a large linear system like [10], [5], [15].

Thanks to the use of tensor field, we can directly summing them without considering the sign of directions. The final tensor field is combined as:

$$\mathcal{T} = \frac{w_L \cdot \mathcal{T}_L + w_U \cdot \mathcal{T}_U + w_C \cdot \mathcal{T}_C}{w_L + w_U + w_C} \tag{3}$$

where $\mathcal{T}_L$ ($\mathcal{T}_U$,$\mathcal{T}_C$ resp.) is line tensor field (user tensor field, color tensor field resp.), contributing to the final tensor field with corresponding distance-related weights w_L (w_U, w_C resp.). Given a direction vector $\mathbf{v} = [v_x; v_y]$, we can construct a structural tensor as a 2×2 matrix:

$$\mathcal{T} = \begin{pmatrix} v_x \cdot v_x & v_x \cdot v_y \\ v_y \cdot v_x & v_y \cdot v_y \end{pmatrix}. \tag{4}$$

And we can get the direction vector from the tensor matrix using eigenvector analysis [26].

Line tensor field We construct a line tensor field constrained by line singularities, including their locations and types. We first employ a distance transforming algorithm proposed by Felzenszwalb and Huttenlocher [27] to compute a nearest distance field $d_L(\mathbf{x})$ from each point $\mathbf{x}$ to line singularities. To guarantee the discontinuity/continuity, here

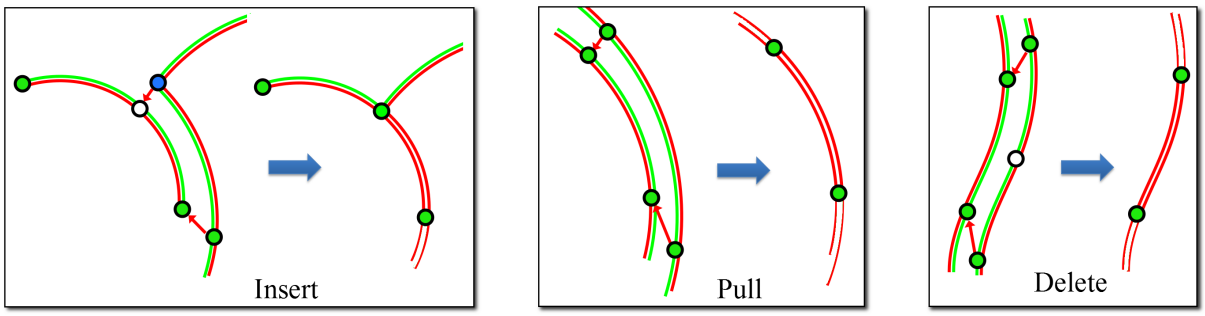


Figure 4. **Automatic operators to unify line singularities** on the shared boundary between adjacent regions.

we only use dual LS and single LS whose inward side is continuous to compute this distance transformation, shown as red curves in Figure 3(b). We then apply a Sobel filter to this distance field to compute a gradient field $\mathbf{v}_L(\mathbf{x}) = [\partial d_L(\mathbf{x})/\partial \mathbf{x}]$. Line tensor field $\mathcal{T}_L$ can be computed by putting this gradient field into formula (4). And we use the normalized distance to compute the combining weight of line tensor field: $w_L(\mathbf{x}) = 1 - d_L(\mathbf{x})/d_L^{max}$. A tangent LIC visualization of line tensor field, along with its combining weights, can be found in Figure 3(b).

User tensor field To provide the user with more control of the tensor field far away from line singularities, our method allows the user to directly add direction constraints by putting strokes. A user-specified stroke (denoted $\mathbf{S}$, we removed the superscript i of region index for the sake of concision) is composed by a point sequence $\mathbf{p}_j$ ($j \in [0, \ell]$), where ℓ is the point number of stroke $\mathbf{S}$. We compute the tangential directions for each stroke point as $\mathbf{v}_j = (\mathbf{p}_{j+\zeta} - \mathbf{p}_{j-\zeta})/2$. ζ controls the smoothness of tangential directions, we take $\zeta = 3$ in this paper. To construct a tensor field in the whole segmented region, we employ a radial basis function averaging as suggested by [3]:

$$\mathcal{T}_U(\mathbf{x}) = \frac{\sum_j \mathcal{T}_j \cdot b(\mathbf{x}, \mathbf{p}_j)}{\sum_j b(\mathbf{x}, \mathbf{p}_j)} \tag{5}$$

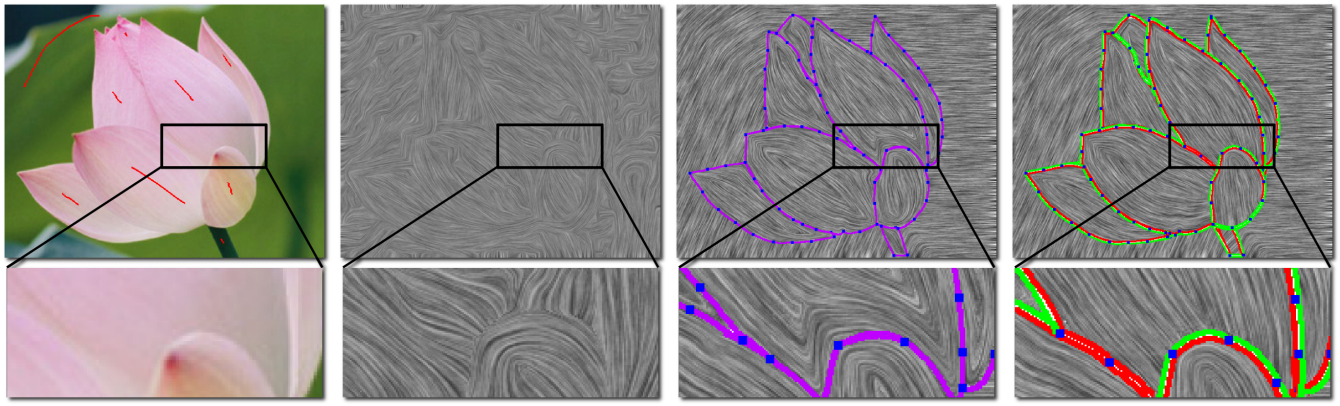
where the basis functions $b(\mathbf{x}, \mathbf{y})$ equals to $exp(-||\mathbf{x} - \mathbf{y}||/5)$ and $\mathcal{T}_j$ is a tensor matrix that puts the orthogonal vector of $\mathbf{v}_j$ into formula(4). Ideally, the sum should be applied to all stroke points. To reduce the computation burden, we take only five stroke points nearest to $\mathbf{x}$ to approximate this sum. Similar to line tensor field, we use the normalized distance to compute the combining weight of the user tensor field: $w_U(\mathbf{x}) = 1 - d_U(\mathbf{x})/d_U^{max}$. A tangent LIC visualization of user tensor field, along with its combining weights, can be found in Figure 3(c).

Color tensor field By using line tensor field and user tensor field, the user can nearly design arbitrary tensor field. But, the user always fails to design fine fields for texture details. our method automatically estimates a color tensor field from the reference image to relieve this issue. We employ a simplified version of the structure tensor calculation method in [26]. To preserve the color information into the tensor field, we apply a Gaussian 1st-order derivative filter along x-coordinate and y-coordinate to compute the spatial derivatives $v_x = [R_x; G_x; B_x]$ and $v_y = [R_y; G_y; B_y]$ separately (R, G, B are the color channels of the reference image), and then construct the structure tensor $\mathcal{T}_C$ using formula(4). The directional field is finally computed by analyzing the eigenvectors of this structure tensor. Since we are computing the color gradient of the reference image, we are more confident about the color tensor field in the places with high gradient magnitude. We compute this confidence as the normalized eigenvalue $e_1(\mathbf{x})/e_1^{max}$ corresponding to the major eigenvector of $\mathcal{T}_C$. Both line tensor field and color tensor field may have high weights around the region boundaries, we thus compute the combining weight of color tensor field as $w_C = max(e_1(\mathbf{x})/e_1^{max} - w_L,\ 0)$ to respect the tensor field constructed using line singularities.

C. *Implementation details*

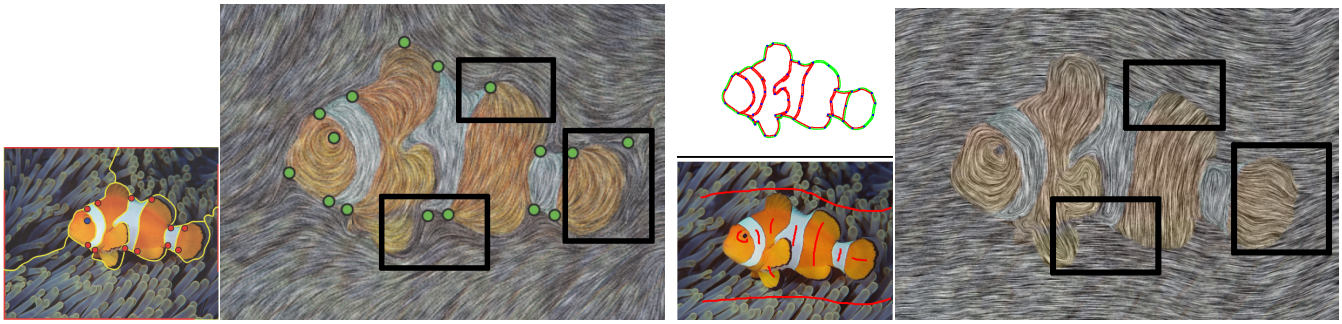
When the user puts strokes to design the tensor field, our method automatically segments the image and locates line-singularities as the boundary of this segmented region. Since the image space is segmented progressively, existing line singularities have to unify the new generated one automatically to avoid gaps and interference between each other. For this purpose, we first unify key points on the shared boundaries by introducing three key point operators, as illustrated in Figure 4:

- **Insert:** Sharp key points should be fixed to preserve shape features, they have the priority to insert a common key point on the adjacent line singularities;
- **Pull:** Sharp key points also have priority to pull an existing common key point on the adjacent line singularities if it is close enough to them; pulling can also occur between two close common key points of adjacent line singularities;
- **Delete:** A common key point will be deleted if it is too close to other key points forward and backward of the same line singularity.



(a) Input image (b) Automatic estimation [26] (c) Separatrices (d) Line singularities

Figure 5. **Comparison with automatic method and separatrix-based method.** (a) Input image and user-specified strokes; (b) automatic estimation method fails to balance between accuracy and removing noises; (c) separatrix-based method introduces unexpected continuity around occluding contours; while our method allows the user to control both field directions and continuities by adjusting line singularities (d).



(a) User interaction (b) Region-based line field [15] (c) User interaction (d) Our method

Figure 6. **Comparison with region-based method [15].** Region-based method asks the user to sketch out image segments by hands, while our design method automatically generates these boundaries from user-specified direction strokes, thus reduces the user interaction burdens. Our method also improves the designed tensor field that produces all types of line singularities.

After each key point operation, we update its forward and backward splines using parameters of corresponding adjacent splines to eliminate gaps between adjacent regions. We also unify the types of line singularities between adjacent regions: line singularities on the shared boundary preserves the inward continuity, and take the inward continuity of adjacent line singularity as their outward continuity.

To provide the user an instant feedback of each interaction, we have implemented several acceleration techniques. In the field computation step, we implemented the computation of color tensor field, the weighted sum of tensor field and nonphotorealistic rendering on modern GPU, and employed K-Nearest Neighbor algorithm for user tensor field computation. In the interaction step, we record region indices in a index image to allow the user to quickly select a region, and generate a spline footprint image to accelerate the searching of line singularities or key points.

V. Results and Applications

A. *Comparison*

A tensor field automatically derived from the input image tends to quickly lose accuracy in Figure 5(b). Separatrix-based method implemented in our framework can not preserve well the structural features on one side of the occluding contour in Figure 5(c), due to the incorrect continuous assumption. Using our method, both the tensor field and discontinuities can be designed easily to follow the nerve field of the petals, as shown in Figure 5(d).

Figure 6 compares our approach with region-based method [15]. Rather than asking the user to sketch out image segments manually, we automatically generate them using the user-specified strokes for direction controls. The boundaries of segmented regions in region-based methods play the same role as separatrices, thus introduce unexpected impacts on their sides, such as in the background around the clown fish and the structural textures on the fins. Our method allows the user to control these continuities by simply specifying the type of each line singularity.

Figure 7. **Three more results of painterly rendering stylization.**

B. *Painterly rendering applications*

We have implemented several nonphotorealistic rendering applications to show how our tensor field improves their quality. *Painterly rendering* puts color strokes randomly on the input image, whose orientations are guided by the tensor field [1]. We vary the stroke sizes with three layers: large for background, small for details and middle for other foreground. To simulate the physical appearance of paint strokes under lighting [28], we employed Hertzmann's fast paint texture approach, that computes a height field to emboss the painting. And to avoid the computation-consuming streamline tracking, we compute the curvature from the tensor matrix to bend the strokes, as shown in Figure 7.

C. *Performance and limitations*

Our experiments are performed on a 2.33 GHz Inter Core 2 Duo PC with 3GB of RAM. We use an approximated implementation of geodesic distance computation, which takes $0.1 - 0.15$ seconds, while the total cost for the generation of line singularity is $0.07 - 0.14$ seconds. The tensor field construction costs only $0.02 - 0.04$ seconds per frames, as the implementation on modern GPU with NVidia GeForce GTX 260. Nonphotorealistic rendering is also accelerated on GPU, most of which costs little time, except that painterly rendering takes $0.4 - 0.9$ seconds per frame. The user interaction time depends on the number of strokes and editing operations for line singularities.

Our method has one major parameter: geodesic threshold for region segmentation τ. Increasing this threshold tends to have larger segmented region. Since the generation of line singularities heavily relies on the quality of segmentation, the user has to carefully adjust this threshold. Although putting multiple strokes using a small threshold for the same region may relieve this issue, we still regard it as one main drawback of our method. Extracting the image structure from textures may help to solve this issue, such as via relative total variation [29].

Though our method provides user-friendly tensor field design tool, it may produce unexpected line or point singularities. To analyze our resulting tensor field, we implemented a simplified version of the automatic singularity identification approach proposed by [30]. From our observation, we believe a very promising direction to avoid unwanted singularities in the future is to study deeper the mathematical properties of line singularities, and the link between line singularity and point singularity.

VI. Conclusion and Future Work

In this paper, we generalize the singularity definition from isolated points to line singularity, that specifies the discontinuities non-uniformly around the object boundary. Based on this generalization, we present a novel tensor field design approach, that allows the user to design smooth yet feature-preserving tensor fields with less efforts. We employ a geodesic-based segmentation to automatically locate line singularities, and a magnitude-weighted tensor field combination that respects line singularities, user-specified stroke directions and also image colors. We demonstrated the capabilities of our method on various nonphotorealistic rendering applications, and the real time performance with an implementation on modern GPU.

Acknowledgment

We would like to thank the reviewers for their thoughtful comments. This research was supported by National Natural Science Foundation of China (No. 61303138).

References

[1] A. Hertzmann, "Painterly rendering with curved brush strokes of multiple sizes," in *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*. ACM, 1998, pp. 453–460.

[2] J. Hays and I. Essa, "Image and video based painterly animation," in *Proceedings of international symposium on Non-photorealistic animation and rendering*. ACM, 2004, pp. 113–120.

[3] P. O'Donovan and A. Hertzmann, "Anipaint: interactive painterly animation from video," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 18, no. 3, pp. 475–487, 2012.

[4] G. Turk, "Texture synthesis on surfaces," in *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*. ACM, 2001, pp. 347–354.

[5] K. Xu, D. Cohen-Or, T. Ju, L. Liu, H. Zhang, S. Zhou, and Y. Xiong, "Feature-aligned shape texturing," in *ACM Transactions on Graphics (TOG)*, vol. 28, no. 5. ACM, 2009, p. 108.

[6] Y. Tong, P. Alliez, D. Cohen-Steiner, and M. Desbrun, "Designing quadrangulations with discrete harmonic forms," in *Proceedings of the fourth Eurographics symposium on Geometry processing*. Eurographics Association, 2006, pp. 201–210.

[7] N. Ray, W. C. Li, B. Lévy, A. Sheffer, and P. Alliez, "Periodic global parameterization," *ACM Transactions on Graphics (TOG)*, vol. 25, no. 4, pp. 1460–1485, 2006.

[8] P. Alliez, D. Cohen-Steiner, O. Devillers, B. Lévy, and M. Desbrun, "Anisotropic polygonal remeshing," in *ACM Transactions on Graphics (TOG)*, vol. 22, no. 3. ACM, 2003, pp. 485–493.

[9] J. Huang, M. Zhang, J. Ma, X. Liu, L. Kobbelt, and H. Bao, "Spectral quadrangulation with orientation and alignment control," in *ACM Transactions on Graphics (TOG)*, vol. 27, no. 5. ACM, 2008, p. 147.

[10] M. Fisher, P. Schröder, M. Desbrun, and H. Hoppe, "Design of tangent vector fields," in *ACM Transactions on Graphics (TOG)*, vol. 26, no. 3. ACM, 2007, p. 56.

[11] N. Ray, B. Vallet, W. C. Li, and B. Lévy, "N-symmetry direction field design," *ACM Transactions on Graphics (TOG)*, vol. 27, no. 2, p. 10, 2008.

[12] N. Ray, B. Vallet, L. Alonso, and B. Levy, "Geometry-aware direction field processing," *ACM Transactions on Graphics (TOG)*, vol. 29, no. 1, p. 1, 2009.

[13] G. Chen, G. Esch, P. Wonka, P. Müller, and E. Zhang, "Interactive procedural street modeling," in *ACM Transactions on Graphics (TOG)*, vol. 27, no. 3. ACM, 2008, p. 103.

[14] M. Kagaya, W. Brendel, Q. Deng, T. Kesterson, S. Todorovic, P. J. Neill, and E. Zhang, "Video painting with space-time-varying style parameters," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 17, no. 1, pp. 74–87, 2011.

[15] C.-Y. Yao, M.-T. Chi, T.-Y. Lee, and T. Ju, "Region-based line field design using harmonic functions," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 18, no. 6, pp. 902–913, 2012.

[16] E. Zhang, J. Hays, and G. Turk, "Interactive tensor field design and visualization on surfaces," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 13, no. 1, pp. 94–107, 2007.

[17] G. Chen, K. Mischaikow, R. S. Laramee, P. Pilarczyk, and E. Zhang, "Vector field editing and periodic orbit extraction using morse decomposition," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 13, no. 4, pp. 769–785, 2007.

[18] G. Chen, K. Mischaikow, R. S. Laramee, and E. Zhang, "Efficient morse decompositions of vector fields," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 14, no. 4, pp. 848–862, 2008.

[19] W.-C. Li, B. Vallet, N. Ray, and B. Lévy, "Representing higher-order singularities in vector fields on piecewise linear surfaces," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 12, no. 5, pp. 1315–1322, 2006.

[20] K. Xu, H. Zhang, D. Cohen-Or, and Y. Xiong, "Dynamic harmonic fields for surface processing," *Computers & Graphics*, vol. 33, no. 3, pp. 391–398, 2009.

[21] H. Theisel, "Designing 2d vector fields of arbitrary topology," in *Computer Graphics Forum*, vol. 21, no. 3. Wiley Online Library, 2002, pp. 595–604.

[22] J. Palacios and E. Zhang, "Rotational symmetry field design on surfaces," in *ACM Transactions on Graphics (TOG)*, vol. 26, no. 3. ACM, 2007, p. 55.

[23] K. Zeng, M. Zhao, C. Xiong, and S.-C. Zhu, "From image parsing to painterly rendering," *ACM Trans. Graph*, vol. 29, no. 1, p. 2, 2009.

[24] B. Cabral and L. C. Leedom, "Imaging vector fields using line integral convolution," in *Proceedings of conference on Computer graphics and interactive techniques*. ACM, 1993, pp. 263–270.

[25] A. Criminisi, T. Sharp, and A. Blake, "Geos: Geodesic image segmentation," in *ECCV 2008*. Springer, 2008, pp. 99–112.

[26] J. Kyprianidis and H. Kang, "Image and video abstraction by coherence-enhancing filtering," *Computer Graphics Forum*, vol. 30, no. 2, pp. 593–602, 2011.

[27] P. Felzenszwalb and D. Huttenlocher, "Distance transforms of sampled functions," Cornell University, Tech. Rep., 2004.

[28] A. Hertzmann, "Fast paint texture," in *Proceedings of the 2nd international symposium on Non-photorealistic animation and rendering*. ACM, 2002, pp. 91–ff.

[29] L. Xu, Q. Yan, Y. Xia, and J. Jia, "Structure extraction from texture via relative total variation," *ACM Transactions on Graphics (TOG)*, vol. 31, no. 6, p. 139, 2012.

[30] K. Polthier and E. Preuss, "Identifying vector field singularities using a discrete hodge decomposition," *Visualization and Mathematics*, vol. 3, pp. 113–134, 2003.