Vector Field Design on Surfaces

Eugene Zhang, Konstantin Mischaikow and Greg Turk
Georgia Institute of Technology

Vector field design on surfaces is necessary for many graphics applications: example-based texture synthesis, non-photorealistic rendering, and fluid simulation. For these applications, singularities contained in the input vector field often cause visual artifacts. In this paper, we present a vector field design system that allows a user to create a wide variety of vector fields with control over vector field topology, such as the number and location of the singularities. Our system combines basis vector fields to make an initial vector field that meets the user specifications.

The initial vector field often contains unwanted singularities. Such singularities cannot always be eliminated, due to the Poincaré-Hopf index theorem. To reduce the visual artifacts caused by these singularities, our system allows a user to move a singularity to a more favorable location or to cancel a pair of singularities. These operations provide topological guarantees for the vector field in that they only affect the user-specified singularities. We provide efficient implementations of these operations based on *Conley index theory*. Our system also provides other editing operations so that the user may change the topological and geometric characteristics of the vector field.

To create continuous vector fields on curved surfaces represented as meshes, we make use of the ideas of *geodesic polar maps* and *parallel transport* to interpolate vector values defined at the vertices of the mesh. We also use geodesic polar maps and parallel transport to create basis vector fields on surfaces that meet the user specifications. These techniques allow our vector field design system to work for both planar domains and curved surfaces.

We demonstrate our vector field design system for several applications: example-based texture synthesis, painterly rendering of images, and pencil sketch illustrations of smooth surfaces.

Categories and Subject Descriptors: I.3.5 [**Computer Graphics**]: Computational Geometry and Object Modeling – Geometric algorithms, languages, and systems

General Terms: Algorithms

Additional Key Words and Phrases: Vector Field Design, Topology, Surfaces, Computational Geometry

1. INTRODUCTION

Many graphics applications require an input vector field. For instance, example-based texture synthesis makes use of a vector field to define local texture orientation and scale [Praun et al. 2000; Turk 2001; Wei and Levoy 2001]. In non-photorealistic rendering, vector fields are used to guide the orientation of brush strokes [Hertzmann 1998] and hatches [Hertzmann and Zorin 2000]. In fluid simulation, the external force is a vector field which need not correspond to any physical phenomenon and can exist on synthetic 3D surfaces [Stam 2003]. A vector field design system enables these applications to achieve many different visual effects by using different input vector fields. It can also be used to create vector fields for testing the efficiency and correctness of a particular vector field visualization technique [van

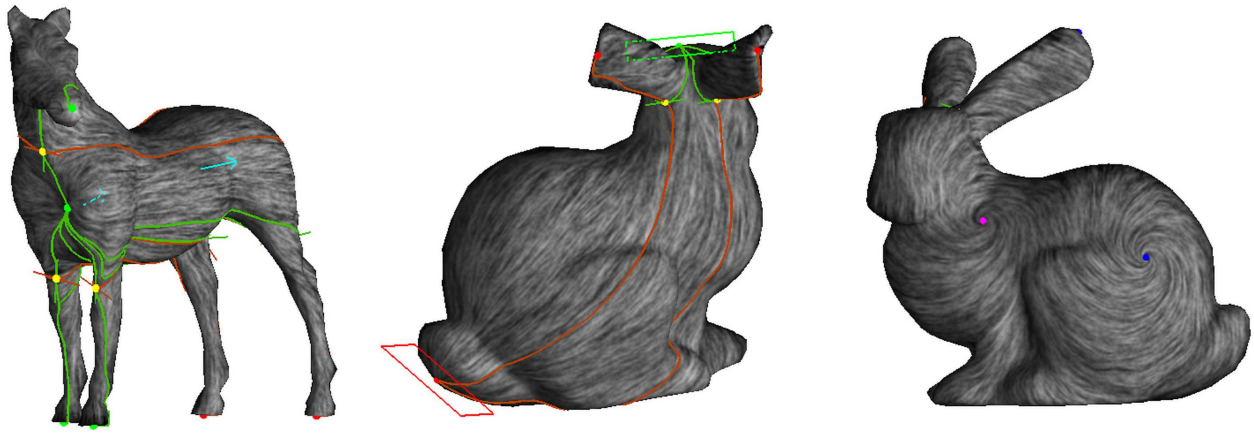

Fig. 1. This figure shows various vector fields created on surfaces using our vector field design system. The vector field shown in the right was used to guide texture synthesis shown in Figure 22 (upper-right).

Wijk 2002; 2003].

Vector field design refers to creating a continuous vector field on a 3D surface based on user specifications or application-dependent requirements. It is different from vector field simplification, which is used to reduce the complexity of large and/or noisy datasets while maintaining their major features. In vector field design, adding and removing features may be required.

There are several challenges to the problem of vector field design. First, the system should enable the user to create a wide variety of vector fields with relatively little effort. Most existing vector field design systems generate some sub-classes of vector fields, such as gradient and incompressible vector fields. This limits their potential applications. Second, the user needs to have control over vector field topology, such as the number and location of the singularities. As was pointed out in [Praun et al. 2000; Hertzmann and Zorin 2000], this is necessary for applications such as example-based texture synthesis and non-photorealistic rendering, in which unwanted singularities often cause visual artifacts. Figure 2 illustrates this with an example from texture synthesis, in which a sink (red dot) in the middle of the bunny's tail (left) causes the synthesis pattern to break up (right).

A vector field system should work for both planar domains and 3D surfaces. In Computer Graphics, 3D surfaces are often represented as meshes, with vertices, edges, and triangles. Surface normal and tangent planes are discontinuous at the vertices and across the edges, and the definition of vector field continuity from smooth manifolds does not apply. Yet, the need for control over vector field topology requires continuous vector fields, for which we can perform particle tracing following the vector field in a continuous and consistent manner. For this reason, we must come up a definition for vector field continuity on mesh surfaces. In addition, we need a vector field representation that guarantees vector field continuity and supports fast and efficient computation of vector field topology. Unfortunately, the popular piecewise linear representation [Tricoche et al. 2001] produces continuous vector fields *only* when the mesh represents a planar domain. For curved surfaces, Stam [2003] uses subdivision surfaces to ensure the vector field continuity. However, it is difficult to extract and control vector field topology with this representation because of its complexity. Also, subdivision surfaces often incur higher computational costs than polygonal meshes.

In this paper, we present a vector field design system for surfaces. This system employs a three-stage pipeline: initialization, analysis, and editing. In the initialization stage, the user can quickly create a vector field by using basis vector fields. Next, the system performs geometric and topological analysis on the vector field and provides visual feedback to the user. In the third stage, the user makes controlled editing operations to the current vector field, such as moving a singularity (*singularity movement*) or cancelling a pair of singularities (*singularity pair cancellation*). This

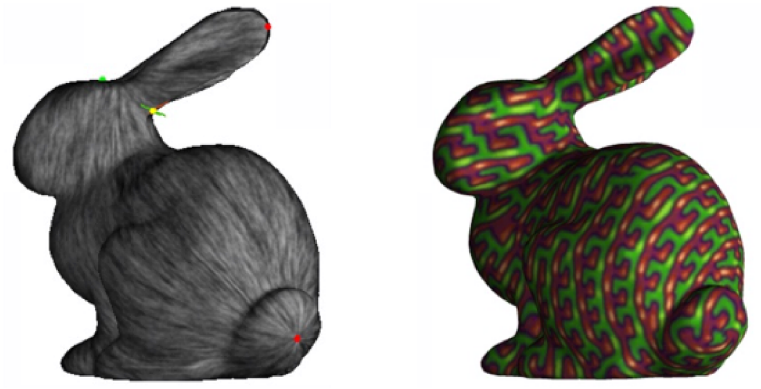

Fig. 2. This figure highlights the need for control over vector field topology in texture synthesis. The input vector field contains a singularity at the center of the bunny's tail (left), and it causes the synthesis patterns to break up.

process of iterative analysis and editing is repeated until the user is satisfied with the result.

Our system enables the user to create a wide variety of vector fields (curl-free, divergence-free, and generic) by using basis vector fields of different kinds. It also provides the user with control over vector field topology, such as the number and location of singularities. We provide efficient algorithms for both singularity pair cancellation and singularity movement based on ideas from Conley index theory, which is more general and powerful than the popular Poincaré index. To enable these operations to work for generic vector fields as opposed to only gradient vector fields, we use *flow rotations* and *reflections* to handle the numerical instabilities associated with regions of high curl and regions near a saddle.

To allow our system to work on 3D surfaces, we present a novel piecewise interpolation scheme for representing vector fields on meshes. This representation guarantees the creation of continuous vector fields based on vector values defined at the vertices, and it supports efficient analysis and editing of surface vector fields. Also, we will describe a new way of building basis vector fields on surfaces from user specifications in the initialization stage. The ideas for both the vector field representation and building surface basis vector fields are based on the concepts of *geodesic polar maps* and *parallel transport* from classical differential geometry.

Figure 1 shows some vector fields created using our vector field system. Our vector field interpolation scheme supports efficient vector field analysis, including topological skeleton extraction. The dots in this figure correspond to the singularities in the vector fields, and the colored curves indicate their connectivity.

The remainder of the paper is organized as follows. We first review the relevant background on vector fields in Section 2. Then, in Section 3 we review existing vector field design systems and vector field simplification techniques. We present our vector field design system for planar domains in Section 4 and 5, and our system for 3D mesh surfaces in Section 6. Section 7 provides some results of applying our vector field design system to various graphics applications, such as painterly rendering, pencil sketches of surfaces, and texture synthesis. Finally, we summarize our contributions and discuss some possible future work in Section 8.

2. BACKGROUND ON VECTOR FIELDS

In this section, we review some basic facts about vector fields. A *vector field* V for a manifold surface $\mathbf{S}$ is a smooth vector-valued function that associates to every point $\mathbf{p} \in \mathbf{S}$ a tangent vector $V(\mathbf{p})$. A vector field defines a system of differential equations:

$$\frac{d\mathbf{p}}{dt} = V(\mathbf{p}). \tag{1}$$

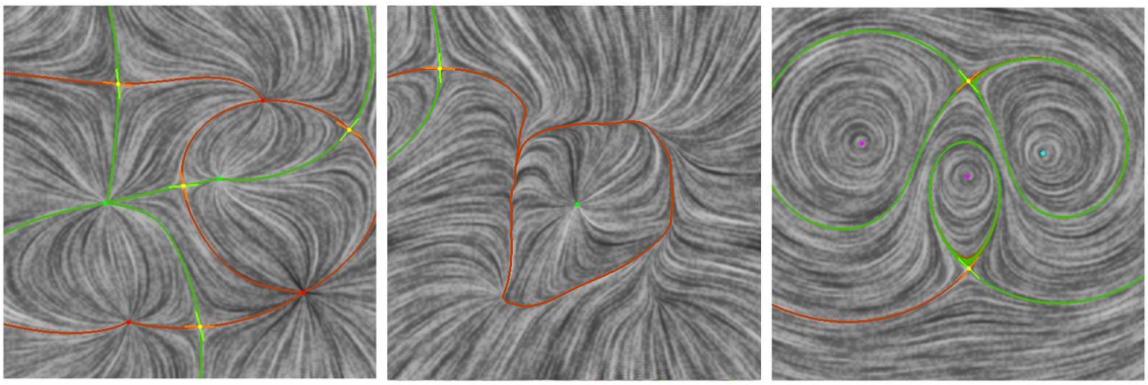

Fig. 3. This figure illustrates three vector fields that are curl-free (left), divergence-free (right), and neither (middle). Singularities are depicted as colored dots and principle directions for saddles are drawn as crosses. Furthermore, incoming separatrices for saddles are shown in green while outgoing separatrices are shown in red. The vector field in the middle contains a periodic orbit that separates the flow inside from the flow outside. The visualization technique is based on van Wijk [2002].

With appropriate restrictions on V, for each point $\mathbf{p}_0 \in \mathbf{S}$, there exists a solution $\mathbf{p} : \mathbb{R} \rightarrow \mathbf{S}$ with the property that $\mathbf{p}(0) = \mathbf{p}_0$ ([Hale and Kocak 1991; Hirsch and Smale 1974]). Because we will be interested in studying multiple solutions simultaneously, it is useful to introduce the notion of the *flow* induced by V that is a continuous function $\varphi : \mathbb{R} \times \mathbf{S} \rightarrow \mathbf{S}$ with the property that $\varphi(t, \mathbf{p}_0) = \mathbf{p}(t)$. The set $\{\mathbf{p}(t) \mid t \in \mathbb{R}\} = \varphi(\mathbb{R}, \mathbf{p}_0)$ is called the *trajectory* through $\mathbf{p}_0$. Uniqueness of solutions to ordinary differential equations guarantees that the set of trajectories forms an equivalence relationship on $\mathbf{S}$. In particular, if $\mathbf{q}_0$ belongs to the trajectory of $\mathbf{p}_0$, then $\mathbf{p}_0$ belongs to the trajectory of $\mathbf{q}_0$. This implies that $\mathbf{S}$ can be decomposed into the set of all trajectories. Some trajectories are of particular significance, such as *singularities*.

A singularity $\mathbf{p}_0 \in S$ is a point where $V = 0$. Observe that the trajectory through a singularity consists of a single point. For many of our calculations we will want to use a singularity classification based on the *local linearization* of the vector field. For simplicity, let V be a vector field defined for some planar domain $D \subset \mathbb{R}^2 = \{(x,y) \mid x,y \in \mathbb{R}\}$ such that $V(x,y) = \begin{pmatrix} F(x,y) & G(x,y) \end{pmatrix}$. The local linearization at a point $\mathbf{p}_0$ is: $V^*(\mathbf{p}) = V(\mathbf{p_0}) + DV(\mathbf{p}_0)(\mathbf{p} - \mathbf{p}_0)$, where $DV = \begin{pmatrix} \frac{\partial F}{\partial x} & \frac{\partial F}{\partial y} \\ \frac{\partial G}{\partial x} & \frac{\partial G}{\partial y} \end{pmatrix}$ is the *Jacobian* of V. A singularity $\mathbf{p}_0$ is *linear* if $DV(\mathbf{p}_0)$ has a full-rank. For the remainder of this discussion we will assume that $\mathbf{p}_0$ is a linear singularity. Results from linear algebra tell us that the two eigenvalues are either both real numbers or a pair of conjugate complex numbers. In the first case, a singularity is a *source* if both (real) eigenvalues are positive, a *sink* if both are negative, or a *saddle* if one is positive and the other is negative. On the other hand, a linear singularity with a pair of conjugate complex eigenvalues is either a *center* if the real part of both eigenvalues is zero, or a *focus* otherwise.

Other trajectories of particular importance are *separatrices* and *periodic orbits*. A separatrix is a trajectory for which the limit as $t \rightarrow \infty$ or $t \rightarrow -\infty$ of the solution function $\mathbf{p}(t)$ is a saddle. For planar vector fields, the *vector field topology* is determined by the set of singularities, separatrices, and periodic orbits. Figure 3 illustrates these special trajectories with three vector fields. Singularities are illustrated as colored dots: sources (green), sinks (red), centers (cyan or magenta depending on the orientations), and saddles (yellow). Furthermore, incoming and outgoing separatrices are colored in green and red, respectively. The vector field in the middle contains a periodic orbit.

Two useful *analytic* characterizations of a vector field are its curl and divergence. Divergence measures the difference between the amount of flow leaving and approaching the measurement point. For instance, a source has a positive divergence and a sink has a negative one. Curl measures the amount of flow that circles around the measurement point.

The distributions of curl and divergence in the domain can help us understand the geometric structure of the trajectories. The two extreme cases are *curl-free* vector fields in which case the curl is zero everywhere, and *divergence-free* vector fields in which case the divergence is zero everywhere. It should be noted that a typical vector field is neither curl-free nor divergence-free. Figure 3 shows three vector fields of different analytical behaviors. The vector field shown in the left is curl-free. In this case the typical singularities are sources, sinks, and saddles. Furthermore, the separatrices divide the domain into a number of combinatorial quadrilaterals called *basins*. The boundary of each basin consists of a source, a sink, and two saddles in between them. Inside each basin, all the trajectories leave the same source and approach the same sink. The vector field in the right is divergence-free, whose typical singularities are centers and saddles. The separatrices divide the domain into a number of bounded regions. Inside each region is a family of periodic orbits that circle around the same center. A generic vector field is shown in the middle, which is neither curl-free nor divergence-free and may contain periodic orbits.

2.1 *Topological Descriptions of Vector Fields*

The vector field design problem requires that the user be able to control the trajectories of a vector field both locally and globally. To do this requires the introduction of a topological characterization of vector fields. In this section, we review a well-known topological descriptor, the *Poincaré index*, and a more general characteristic, the *Conley index*.

A singularity $\mathbf{p}_0$ is *isolated* if there exists an open neighborhood U of $\mathbf{p}_0$ with the property that $\mathbf{p}_0$ is the unique singularity in the interior of U. An isolated singularity $\mathbf{p}_0$ can be characterized by its *Poincaré index*, which is defined in terms of the *winding number* for the *Gauss map*.

DEFINITION 2.1. Let V be a vector field defined on some planar domain D. Let $D_0 \subset D$ be the zero set for V. The *Gauss map* $\alpha : D \setminus D_0 \to S^1$ is defined as $\alpha(x) = \frac{V(x)}{|V(x)|}$.

For a simple closed curve $\Gamma \subset D \setminus D_0$, the Gauss map α induces a continuous map $\alpha|_\Gamma$. If one travels along Γ in the positive direction once, the image under $\alpha|_\Gamma$ necessarily covers the unit circle an integer number of times counting orientation. This integer is the *winding number* of V along Γ. The Poincaré index of an isolated singularity $\mathbf{p}_0$ is the winding number of any simply connected curve that encloses $\mathbf{p}_0$ and contains no other singularities either in its interior or on the boundary. Denote this number as $\kappa(V;\mathbf{p}_0)$. The Poincaré index is $+1$ for sources, sinks, centers, and foci. It is -1 for saddles, and 0 for regular points. The *Poincaré-Hopf theorem* links the vector field topology to the topology of the underlying domain in the following way. Let S be a closed orientable manifold with an Euler characteristic E. Furthermore, let V be a continuous vector field defined on S with only isolated singularities $\mathbf{p}_1, ..., \mathbf{p}_n$. Then $\sum_{i=1}^{n} \kappa(V;\mathbf{p}_i) = E$.

An immediate corollary of the Poincaré-Hopf theorem is that given a particular vector field V, if one wants to remove a singularity of a positive or negative Poincaré index, then one must simultaneously remove a singularity of the opposite sign. In fact, for a 2-manifold, a zero total Poincaré index for a region R guarantees that it is possible to replace the vector field inside R with a singularity-free vector field.

The Poincaré index is a powerful tool for describing singularities. However, it does not distinguish between sources and sinks, nor does it provide information about periodic orbits and separatrices. Figure 4 illustrates this with a number of examples. First, the Poincaré indices for the disk in case (b) and (d) are both one. When simplifying the vector field inside the disk such that only one singularity remains, the Poincaré index alone cannot predict whether the singularity is a source or a sink. Second, the Poincaré indices for the ring-shaped region in (e)-(g) are zero. However, the three vector fields have very different characteristics. For example, when the vector fields inside the region are singularity-free, the vector fields in (f) and (g) necessarily contain a periodic orbit, while the vector field in (e) does not. The design of vector fields requires the imposition of additional quantitative information including the location of the singularities, periodic orbits and separatrices, and the control of the smoothness and/or curvature of the vector field. In this work, we have chosen to control singularities for vector fields defined on 2-manifolds, for which the Poincaré index is insufficient since it does not distinguish between sources and sinks. In addition, we wish to set up a

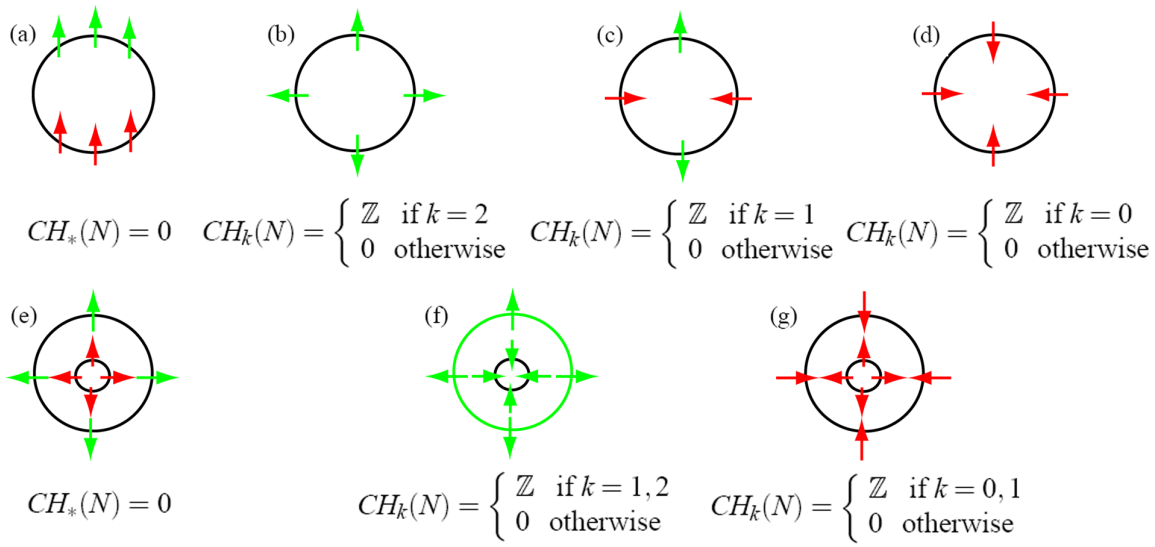

Fig. 4. Seven basic scenarios of isolating blocks and their associated Conley indices. The Conley indices can be used to distinguish between sources and sinks (b and d), and it provides information on periodic orbits (e, f, g). By comparison, the Poincaré indices are the same for cases (b) and (d), and for (e)-(g). Case (a), (b) and (e) are of particular interest since they are used in topological editing operations (Section 5.3).

framework that has the potential of being extended to the control of separatrices and periodic orbits. For these reasons, we borrow basic ideas from Conley index theory and provide implementations for our topological editing operations (Section 5.3) according to this theory.

The Conley index is defined in the context of arbitrary vector fields that produce continuous flows. It possesses the continuation properties of the Poincaré index while being able to distinguish between sinks and sources. It provides sufficient conditions on whether two singularities can cancel. More importantly, it can be used to identify periodic orbits and separatrices, and to indicate whether two periodic orbits can be cancelled.

The following concept is the starting point for Conley index theory. Given a region $N \subset \mathbf{S}$, let ∂N denote the boundary of N. A compact set N is an *isolating neighborhood* if for every $\mathbf{p} \in \partial N$, $\varphi(\mathbb{R}, \mathbf{p}) \not\subset N$, i.e., the trajectory of any point on ∂N leaves eventually either in forward or backward time. The set of boundary points which leave or enter N immediately can be characterized, respectively, by

$$N^- := \{\mathbf{p} \in \partial N \mid \varphi([0,t),\mathbf{p}) \not\subset N,\ \forall t > 0\}\ , \quad N^+ := \{\mathbf{p} \in \partial N \mid \varphi((t,0],\mathbf{p}) \not\subset N,\ \forall t < 0\}. \tag{2}$$

A compact set N is an *isolating block* if for each boundary point, there is either a forward or backward trajectory that immediately leaves the region: that is, $\partial N = N^- \cup N^+$. Observe that an isolating block is a special case of an isolating neighborhood.

Given an isolating block N for a vector field V, its Conley index is defined to be the relative homology [Kaczynski et al. 2004] of N with respect to N^-, i.e. $CH_*(N) := H_*(N, N^-)$. $CH_*(N) = \{CH_k(N) | k = 0,1,2,...\}$ is a collection of *groups*. When the domain of the vector field is a surface, $CH_k(N) = 0$ for $k \geq 3$ (see [Conley 1978; Mischaikow 2002; Mischaikow and Mrozek 2002] for further details and references). For the purposes of this paper the computation of this index is fairly simple since our isolating block N will always take the form of a polygonal region and N^- will be a finite number of disjoint sets consisting of boundary edges of N. Idealized isolating blocks and their associated Conley indices are indicated in Figure 4. Case (a) and (e) have the trivial Conley index, and (b), (c), and (d) have the Conley index of a source, a saddle, and a sink, respectively. Of particular interest are case (a), (b), and (e). We construct regions of these types for topological editing operations (Section 5.3).

3. PREVIOUS WORK

Vector field *analysis* and *visualization* have been well studied, and a good survey is available in [Hauser et al. 2002]. On the other hand, vector field *design* is far less explored. We will review existing vector field design systems, both for planar domains and 3D surfaces. In addition, since our system allows the user to perform vector field simplification, both geometrically and topologically, we also review existing vector field simplification techniques.

3.1 Vector Field Design Systems for Surfaces

There has been some prior work in creating vector fields on surfaces. In all the instances that we know, such systems have been created in a quick manner to generate vector fields for a particular application, such as texture synthesis [Praun et al. 2000; Turk 2001; Wei and Levoy 2001], fluid simulation [Stam 2003], or for testing a vector field visualization technique [van Wijk 2003]. Furthermore, the details of these design systems have not been published.

There are several approaches for creating a surface vector field using these systems. In the first approach, a 3D vector field is specified and projected onto the surface to obtain a tangential vector field [van Wijk 2003]. This is similar to performing texture synthesis on surfaces through solid textures. While it is simple and fast, achieving control is hard. In the second approach, the user specifies desired vector values at a few locations on the surface, and the system performs relaxation to obtain a global surface vector field [Turk 2001; Wei and Levoy 2001]. This can be seen as a diffusion process in which the desired vector values are smoothly propagated from the seed points to the rest of the surface. In the third approach, the user again specifies the vector values at a few places on the surface. Then a global vector field is constructed by interpolation these locations using Gaussian radial basis functions over the surface [Praun et al. 2000]. Another way of creating surface vector fields is to parameterize the surface and define vector fields in the parametric domain [Stam 2003].

These vector field design systems do not provide control over vector field topology, such as the number and location of the singularities in the vector field. However, we borrow some of these ideas to create an initial vector field in the first of a three-stage design pipeline.

3.2 Vector Field Design Systems for Planar Domains

For planar domains, vector field design systems based on topological information have been developed. Van Wijk created a vector field design system to demonstrate his image-based flow visualization technique [2002]. In this design system, the user specifies desired singularity locations and types. The system converts each specification into a simple vector field and combines them into a global vector field using radial basis functions. The idea of using basis vector fields is inspired by the work of Wejchert and Haumann [1991]. However, vector fields created in this manner often have more singularities than what the user has intended. The system does not provide a way of removing undesired singularities, and therefore it lacks control over vector field topology. Rockwood and Bunderwala [2001] propose a technique that uses geometric algebra to create a vector field based on user-specified singularity locations and types (source, saddle, etc). The user can interactively create a vector field by adding, removing and editing the singularities. This system also lacks control over vector field topology since the vector field created this way may have unspecified singularities. Theisel [2002] proposes a 2D vector field design system in which the user has the complete control over vector field topology. To do so, the user specifies the *topological skeleton* of the desired vector field and the system creates a piecewise-linear vector field to match it. This system requires the user to specify the desired vector field skeleton, which can be cumbersome for complicated vector fields. Both of the above topology-based design systems [Rockwood and Bunderwala 2001; Theisel 2002] require a planar parameterization, and it is not obvious how these systems should be generalized to curved surfaces.

All of above systems have certain traits that we wish to incorporate into the vector field design system. In fact, we will borrow techniques from existing systems to serve our purpose at various stages. This will become clear in Sections 4, 5, and 6.

3.3 Vector Field Topology

In their pioneering work, Helman and Hesselink [1991] visualize a vector field by extracting and visualizing its topological skeleton, which consists of the singularities and their connectivity. They propose an efficient method for extracting the topological skeleton for continuous vector fields defined in either a plane or a volume. This work has inspired a great deal of interests in understanding and visualizing vector fields through topological analysis. There has been considerable work in the Visualization community on vector field topology, and we only mention a few relevant publications here. Scheuermann et al. [1998] use Clifford algebra to study the non-linear singularities in a vector field and propose an efficient algorithm for merging nearby linear singularities into a higher-order singularity. Later, Polthier and Preuß use Hodge-Decomposition to locate singularities of different types in a vector field [2003]. Wischgoll and Scheuermann [2001] propose an efficient algorithm for computing the periodic orbits in a planar flow.

3.4 Vector Field Simplification

Vector field simplification has been well-researched by the Scientific Visualization community. Most of the datasets that come from scientific simulation are difficult to analyze due to noise in the data. Vector field simplification refers to reducing the complexity of a vector field while maintaining its major features. A vector field simplification technique can be either topology-based (TO) or non-topology-based (NTO).

NTO methods perform smoothing to a vector field, either globally or locally. Existing NTO techniques, such as [Polthier and Preuß 2003; Westermann et al. 2000; Tong et al. 2003], are often based on performing Laplacian smoothing on the potential of a vector field, which is a scalar field. For example, Tong et al. [2003] decompose a vector field into three components: curl-free, divergence-free, and harmonic. Each component is individually smoothed and the results are summed. Vector-based smoothing is performed *only* on the harmonic part, while potential-based smoothing applies to the divergence-free and curl-free components, respectively. Smoothing operations reduce the vector field complexity and most likely remove a large percentage of the singularities in the original vector field.

TO methods simplify the topology of a vector field explicitly. According to the Poincaré-Hopf theorem, it is possible to eliminate a pair of singularities with opposite Poincaré indices at the same time. This idea has been formulated into an operation called singularity pair cancellation, which forms the foundation for many existing TO methods. A class of TO methods performs pair cancellation on scalar fields defined on surfaces [Edelsbrunner et al. 2002; Edelsbrunner et al. 2003] by changing the values of the scalar function near the singularity pair. This is equivalent to simplifying the gradient vector field of the scalar function. Ni et al. [2004] allow the user to design fair Morse functions over a mesh surface, which is equivalent to designing gradient vector fields. In their work, the user specifies the desired number and configuration of the critical points of the function, and the system performs multi-grid relaxation to determine a Morse function that meets the requirements.

Another class of TO methods perform cancellation on a vector field directly, such as the technique by Tricoche et al. [2001]. This technique first locates a region surrounding the singularity pair, and then performs a non-linear optimization on the vector values at the interior vertices such that the Poincaré indices are zero for every triangle inside the region.

All the TO methods mentioned above are based on Morse theory, e.g., gradient vector fields.

Our system provides both a NTO method (Section 5.2) and a TO method (Section 5.3.1), and the implementations of our methods are rather different from existing techniques. For instance, our singularity pair cancellation algorithm is based on Conley index theory, which allows us to work with arbitrary vector fields. Furthermore, existing topological analysis and simplification techniques are limited to planar and volume domains because it is not clear how to represent a continuous vector field on a mesh surface. We present a piecewise interpolation scheme in Section 6 that overcomes this problem, therefore allowing vector field analysis and editing to be adapted to meshes.

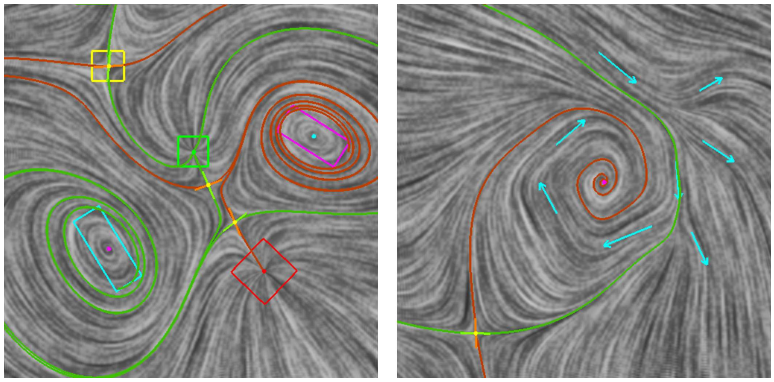

Fig. 5. An initial vector field can be created using *singular elements* (left, highlighted by colored boxes), and *regular elements* (right, highlighted by colored arrows). The centers of the colored boxes are the locations of the desired singularities. Notice in both cases, there are singularities not specified by the user.

4. DESIGN FOR PLANAR DOMAINS

Our planar vector field design system consists of three stages: *initialization*, *analysis*, and *editing*. During the initialization stage, the user quickly creates a vector field with a set of specifications. Vector field topology is not a concern at this stage. Next, the system performs both geometric and topological analysis of the current vector field and provides visual feedback to the user. In the editing stage, the user modifies the vector field through a set of pre-defined editing operations. The user may perform many editing operations before accepting the result. The initialization and analysis stages are relatively straightforward, and we describe them in Section 4.1 and 4.2, respectively. The editing stage is at the core of our vector field design system, and we will describe this in Section 5.

4.1 Creating the Initial Vector Field

The first stage allows the user to easily create an initial vector field without being concerned about vector field topology. There have been two ways of creating such a field: relaxation [Turk 2001; Wei and Levoy 2001], and using basis vector fields [Praun et al. 2000; van Wijk 2002]. We adopt van Wijk's basis vector approach [2002] because we are impressed by its intuitive nature and its simplicity. In this approach, every user-specified constraint is used to create a basis vector field defined in the plane. An initial vector field is then constructed as a weighted sum of these basis vector fields. We will refer to each user-specified constraint as a *design element*, which can be either *singular* or *regular*. A design element has a center location and a set of control parameters, which will be described next.

A singular element corresponds to a vector field that has a singularity of certain type at a desired location. For instance, if the user desires an isotropic source at location $\mathbf{p}_0 = (x_0, y_0)$ with strength $k > 0$, the system will create the following vector field for any point $\mathbf{p} = (x, y)$ in the plane:.

$$V(\mathbf{p}) = e^{-d\|\mathbf{p}-\mathbf{p}_0\|^2} \begin{pmatrix} k & 0 \\ 0 & k \end{pmatrix} \begin{pmatrix} x - x_0 \\ y - y_0 \end{pmatrix} \tag{3}$$

Here, d is a decay constant that is used to control the amount of influence of the basis vector field. Other isotropic singular elements include a sink, a saddle, a counter-clockwise center, and a clockwise center, whose matrices are the following: $\begin{pmatrix} -k & 0 \\ 0 & -k \end{pmatrix}$, $\begin{pmatrix} k & 0 \\ 0 & -k \end{pmatrix}$, $\begin{pmatrix} 0 & -k \\ k & 0 \end{pmatrix}$, $\begin{pmatrix} 0 & k \\ -k & 0 \end{pmatrix}$. The system allows the user to modify the scale, orientation and center location of an existing singular element as well as to remove one. Modifications to singular elements will result in more complicated matrices (details can be found in [van Wijk 2002]).

A regular element assigns a particular nonzero vector value V_0 at a desired location $\mathbf{p}_0$. Again, the system creates a

basis vector field as follows:

$$V(\mathbf{p}) = e^{-d\|\mathbf{p}-\mathbf{p}_0\|^2} V_0 \tag{4}$$

The resulting vector field is interactively updated and displayed as the user continues to make adjustment to the set of regular and singular elements. Figure 5 shows two vector fields that were generated using singular elements (left) and regular elements (right). In practice, both types of specifications can be combined to create an initial vector field. Notice that summing the basis vector fields may cause additional (perhaps unwanted) singularities to appear, i.e., the ones that are not at the centers of any colored box in this figure. The unwanted singularities will be handled through the topological editing operations that we will describe in Section 5.

4.2 Vector Field Representation and Analysis

Our system performs the following analysis on a given vector field: computing curl and divergence, locating singularities and determining their types, and tracing separatrices.

The initial vector field created in the first stage is difficult to analyze because of its complicated formula (Equation 3 and 4). Furthermore, analytical formulas are not available for 3D mesh surfaces that lack a global parameterization. To perform analysis in a fast and efficient manner and to be able to generalize the method to surfaces, we follow the approach by Helman and Hesselink [1991] and use a piecewise approximation in which the underlying domain is tessellated by a triangular mesh. The vector values are sampled at the vertices according to the analytic formula and are linearly interpolated on the edges and across the interiors of the triangles. To be more specific, for a given planar triangular mesh, our system represents a vector field V by assigning vector values $\{W_1, W_2, ..., W_n\}$ at the mesh vertices $\{v_1, v_2, ..., v_n\}$. For a point $\mathbf{p} = (x, y)$ inside a triangle $T = \{v_{T_1}, v_{T_2}, v_{T_3}\}$ whose barycentric coordinates are $(\alpha_1, \alpha_2, \alpha_3)$, we have

$$V(\mathbf{p}) = \sum_{j=1}^{3} \alpha_j W_{T_j} \tag{5}$$

or, under some local coordinate system of T, $V(\mathbf{p}) = \mathbf{M_T} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} e \\ f \end{pmatrix}$, where $\mathbf{M_T} = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$. This representation does not require an analytical formula and is compatible with many graphics applications that use vector fields. Furthermore, Equation 5 can be adapted to represent a continuous surface vector field (Section 6.2.1).

For each triangle, our system computes the following information: the divergence and curl, the Poincaré index, the location of the singularity inside if any, and the incoming and outgoing directions if the triangle contains a saddle. Details of computing these quantities using the piecewise linear representation can be found in [Tricoche 2002]. We also compute the topological skeleton of the vector field, which is done by following the approach of Helman and Hesselink [1991]. Starting from every saddle point, we follow the flow forward in its outgoing directions until the flow is stopped at a singularity or hits the boundary. To trace the trajectories away from a saddle we use a Runge-Kutta algorithm with adaptive stepsize control [Cash and Karp 1990]. This gives us the two outgoing separatrices. Similarly, we obtain the two incoming separatrices by following the flow backward along the incoming directions of the saddle. Figure 3 and 5 show the topological skeletons of the corresponding vector fields.

5. EDITING

The vector field editing stage is at the heart of our design system. The set of useful editing operations are application-dependent. For instance, in texture synthesis and non-photorealistic rendering, the user often needs to remove unwanted singularities or to move them to less visible regions. Fluid simulation may require adjusting the amount of curl and divergence of an external force. Furthermore, noisy datasets often contain a large number of singularities and rather complex behaviors. Simplifying the flow while maintaining its major features is a necessary task for any vector field design system. We provide the following operations:

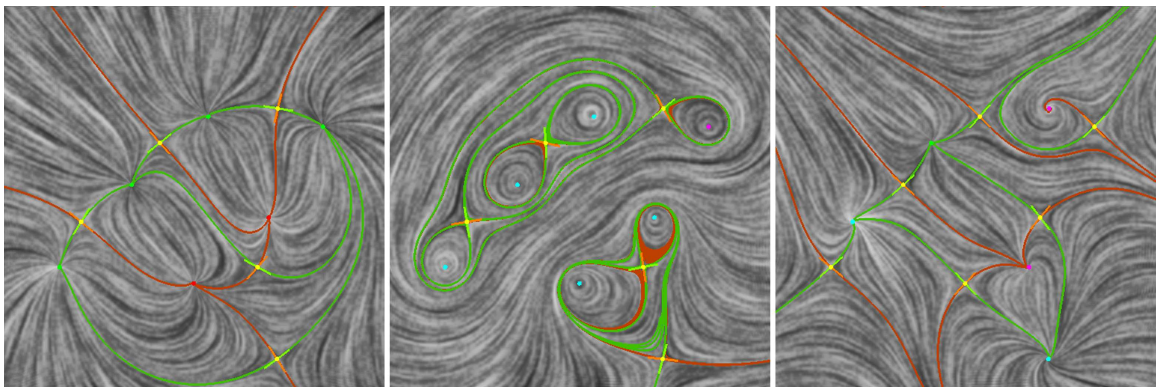

Fig. 6. In this figure, a vector field (left) is first rotated by $\frac{\pi}{2}$ (middle), then reflected with respect to the X-axis (right).

(1) Matrix actions on flows: *flow rotations* and *flow reflections*.
(2) *Flow smoothing* within a user-defined region.
(3) Topological editing operations: *singularity pair cancellation* and *singularity movement*.

Matrix actions can be used to adjust flow characteristics such as curl and divergence. Flow smoothing is an efficient vector field simplification operation that can also simplify vector field topology.

Topological editing operations are used to provide explicit control over the number and location of the singularities in the vector field. Most existing singularity pair cancellation algorithms assume that there is a connecting orbit between the singularity pair, as in the case of a source/saddle or sink/saddle cancellation. When the pair involves a center or a focus of a high curl, however, the connecting orbit either does not exist or cannot be computed in a numerically stable fashion. Consequently, these techniques do not address such cases. Similar issue comes up in singularity movement, where it is necessary to compute the trajectory that connects the singularity to its new desired location under the current flow. Such a trajectory does not always exist when the singularity is a saddle or a center. As we will describe later Sections 5.3.1 and 5.3.2, matrix actions can also be used to modify the types of singularities such that the aforementioned connecting orbit exists and can be computed easily in the modified field. This is essential to overcome numerical instabilities associated with regions of high curl and regions near saddles.

Since we use a piecewise linear approximation, all the editing operations affect the vector values at the vertices only. These values are then extended to a continuous vector field defined on the whole mesh surface through piecewise linear interpolation.

5.1 *Matrix Actions on Flows*

Any 2×2 matrix $M=\begin{pmatrix} a & b \\ c & d \end{pmatrix}$ induces a vector field operator as follows: $(M(V))(\mathbf{p})=\begin{pmatrix} a & b \\ c & d \end{pmatrix} V(\mathbf{p})$. When M has a full rank, it does not change the number or location of the singularities in the vector field. Furthermore, M maintains the Poincaré index if $\det(M)>0$, and negates it if $det(M)<0$. For any $\theta\in\mathbb{R}$, $R_\theta=\begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix}$ is a *flow rotation operator* and $F_\theta=\begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ -\sin(\theta) & -\cos(\theta) \end{pmatrix}$ is *flow reflector operator*. Actions of R_θ's and F_θ's are of particular interests to us, and we will describe them in detail next.

For any θ and any vector field V, it is straightforward to verify that

$$(curl(R_\theta(V)))^2+(div(R_\theta(V)))^2=(curl(V))^2+(div(V))^2 \tag{6}$$

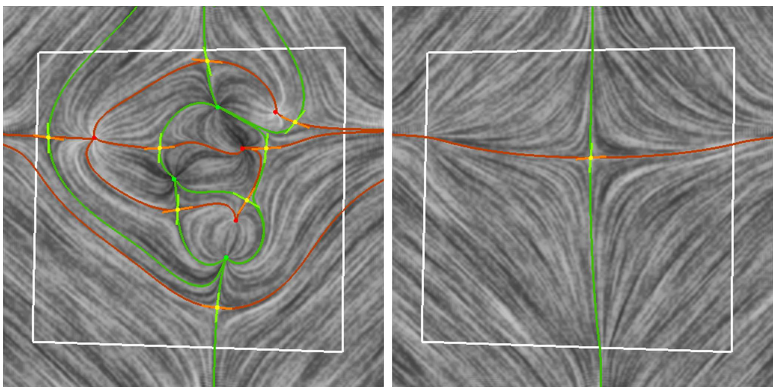

Fig. 7. This figure shows the results of applying flow smoothing to a user specified region (inside the white boundary). Notice the vector field defined outside the region is not changed.

This implies that for any point $\mathbf{p}$ in the domain, there are appropriate rotations of V such that $curl(R_\theta(V))(\mathbf{p}) = 0$ or $div(R_\theta(V))(\mathbf{p}) = 0$. Furthermore, a curl-free vector field can be rotated into a divergence-free vector field and vice versa with a $\frac{\pi}{2}$ rotation. Topologically speaking, flow rotations do not alter the number, the location, or the Poincaré index of the singularities ($\det(R_\theta) = 1 > 0$). Any singularity with a Poincaré index of $+1$ can be converted into a source with an appropriate rotation. A saddle remains a saddle under flow rotations; however, its incoming and outgoing directions are rotated, possibly by different amounts. These topological properties make flow rotations essential for the topological editing operations such as singularity pair cancellation (Section 5.3.1) and singularity movement (Section 5.3.2), especially in regions of high curl.

F_θ induces a reflection on the vector values of a vector field V with respect to axis $\cos(\frac{\theta}{2})X + \sin(\frac{\theta}{2})Y = 0$. It is straightforward to verify that $F_\theta^2 = Id$. For planar domains, flow reflections do not alter the number or location of the singularities in V. Since $\det(F_\theta) = -1 < 0$, they negate the sign of the Poincaré indices. Just as flow rotations can convert a singularity with a Poincaré index of $+1$ into a source, flow reflections can turn any saddle into a source with an appropriate choice of the reflection axis. This makes flow reflections crucial for our singularity movement operations on saddles (Section 5.3.2).

Figure 6 shows the effect of applying flow rotations and reflections to a planar vector field. The actions are $R_0 = Id$ (left), $R_{\frac{\pi}{2}}$ (middle), and $F_{\frac{\pi}{2}}$ (right). Notice that in all instances, the number and location of the singularities do not change. Flow rotations maintain the Poincaré indices while flow reflections negate their signs.

The concepts of flow rotations and reflections are not new. Theisel and Weinkauf [2002] define four types of operations for feature-matching between vector fields. These operations include rotation and negative scaling (including reflection). However, we believe that it is a novel idea to use flow rotations and reflections to overcome the numerical difficulties associated with regions of high curl and regions near saddles.

5.2 Flow Smoothing

A vector field often contains noise, and vector field simplification can be used to reduce the flow complexity while maintaining the major features. Here, we describe a non-topology-based simplification method called flow smoothing, which is carried out in two stages. First, the user specifies a simply-connected region by drawing a closed loop in the domain. Second, the vector field inside the region is replaced with a "simpler" vector field. The key for this operation is to let the user decide the region for smoothing. Once the region is determined, a number of known smoothing techniques, such as [Westermann et al. 2000; Tong et al. 2003] can be used to replace the flow inside. In particular, Tong et al. [2003] compute a Hodge-Helmholtz decomposition of the original vector field. Smoothing is performed on the potentials of the curl-free and divergence-free parts. However, smoothing the harmonic component

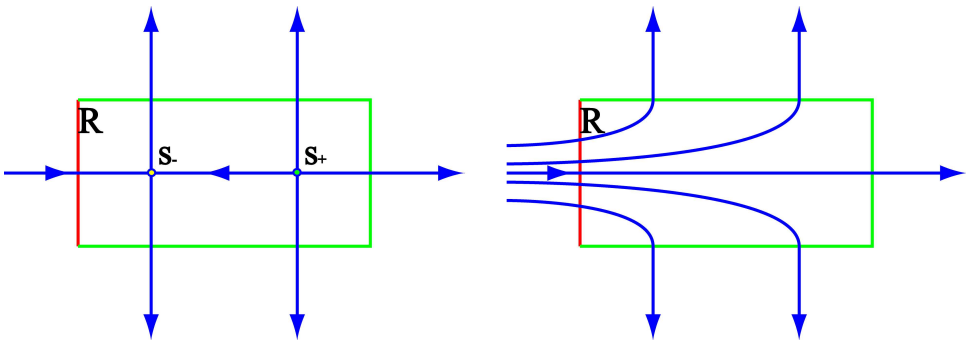

Fig. 8. This figure illustrates our two-step algorithm for singularity pair cancellation between a source $\mathbf{s}_+$ and a saddle $\mathbf{s}_-$. In the left, an isolating block R is found to enclose both singularities and its boundary consists of two segments: inflow (red) and outflow (green). The vector field inside R is replaced with a flow that has no singularities (right).

still requires vector-valued smoothing. Here, we use a smoothing approach on the vector values directly without performing the Hodge-Helmholtz decomposition. This approach avoids the costs of performing the decomposition and the two additional potential-based smoothing operations, and therefore is faster. Furthermore, vector-valued smoothing tends to remove high-frequency noise from the data as well as reduce the number of singularities in the vector field. This is supported by our numerical tests. For remeshing purposes, Alliez et al. [2003] employ a similar component-based approach to smooth curvature tensor fields.

Given a vector field V and a user-specified region R, we replace V with another vector field $\overline{V}$ inside R. This is achieved by solving the vector-valued Laplace equation inside R, with V being fixed on ∂R. Let $\overline{V}(\mathbf{p}) = \begin{pmatrix}\overline{F}(p_1,p_2) & \overline{G}(p_1,p_2)\end{pmatrix}$. Then the new vector field $\overline{V}$ inside R is given by:

$$\begin{pmatrix}\nabla^2\overline{F} = 0\\ \nabla^2\overline{G} = 0\end{pmatrix} \tag{7}$$

In practice, the user-specified region R is part of the underlying mesh that is used to represent the planar domain. To solve Equation 7 on this discrete mesh, the vector values are fixed at the boundary vertices of R, i.e., $\overline{F} = F, \quad \overline{G} = G$. The vector values of $\overline{F}$ and $\overline{G}$ for an interior vertex v_i is determined by:

$$\begin{pmatrix}\overline{F}(v_i)\\ \overline{G}(v_i)\end{pmatrix} = \sum_{j\in J}\omega_{ij}\begin{pmatrix}\overline{F}(v_j)\\ \overline{G}(v_j)\end{pmatrix} \tag{8}$$

Here, J is the set of index j's such that (v_i, v_j) is an edge in the mesh. The weights ω_{ij} are defined according to the mean-value coordinates of Floater [2003] since this method guarantees ω_{ij} to be non-negative. This leads to a pair of sparse linear systems, which we solve through an implicit bi-conjugate solver.

In Figure 7, a complicated vector field with many singularities (left) is converted into a vector field with only one singularity (right) through smoothing. The boundary of the user-specified region is highlighted with a white loop. Notice that the vector field is not altered outside the user-specified region. Later, we make use of flow smoothing to perform singularity pair cancellation (Section 5.3.1) and singularity movement (Section 5.3.2).

5.3 *Topological Editing Operations*

A vector field often contains unwanted singularities. To allow a user to control the number and location of the singularities in a vector field, our system provides two topological editing operations: *singularity pair cancellation*, and *singularity movement*. Singularity pair cancellation refers to removing a pair of (unwanted) singularities with opposite Poincaré indices, while singularity movement is used to move a singularity to a more desirable location. Both oper-

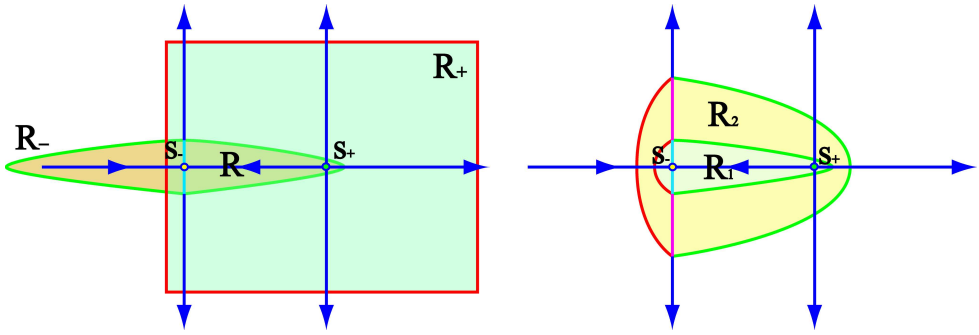

Fig. 9. This figure illustrates our construction of an isolating block R for singularity pair cancellation. In the left, a region R_+ is generated by following the flow forward from a neighborhood of $\mathbf{s}_+$. Similarly, a region R_- is obtained by following the reverse flow from a neighborhood of $\mathbf{s}_-$. When there is a unique connecting orbit between $\mathbf{s}_+$ and $\mathbf{s}_-$, $R = R_+ \bigcap R_-$ is an isolating block with the trivial Conley index. In the right, two valid regions R_1 and R_2 are obtained by using different sizes of the neighborhoods of $\mathbf{s}_-$. R_2 is preferred since it is larger and tends to result in smoother flows after the cancellation.

ations provide topological guarantees in that they only affect the intended singularities, and our implementation are based on Conley index theory.

5.3.1 *Singularity Pair Cancellation.* As discussed in Section 2.1, singularity elimination must be performed for a pair of singularities with opposite Poincaré indices. This operation is therefore called *singularity pair cancellation*. There have been several pair cancellation methods for simplifying scalar fields on surfaces [Edelsbrunner et al. 2002; Edelsbrunner et al. 2003]. These techniques achieve singularity pair cancellation for the gradient field by modifying the scalar values in a region near the singularity pair. It is not clear how these techniques can be used for generic vector fields, which need not correspond to any scalar functions.

Tricoche et al. [2001] propose a pair cancellation technique for vector fields by allowing a saddle to be cancelled with either a source or a sink. To achieve this, they first find a narrow neighborhood that encloses the singularity pair and their connecting orbit. Then an iterative non-linear optimization is performed on the vector values at the interior vertices of this region so that the Poincaré index for every triangle is zero. There are a number of issues with this approach. From a theoretical viewpoint, any simplification technique based on the Poincaré index cannot be applied to the cancellation of a repeller/attractor pair in which one of the entities is a periodic orbit (Section 2.1). From a numerical point of view, this technique is not robust in handling pair cancellations that involve a center or a focus with high curl. Furthermore, the non-linear optimization technique is computationally expensive and it does not guarantee that a solution can be found.

In this work, we propose a new pair cancellation technique based on Conley index theory, which provides theoretical guarantees for any attractor/repeller pair including objects other than singularities. We will only consider the case of a source/saddle pair cancellation. If the singularity with a positive Poincaré index is not a source, we can always find an appropriate rotation to turn it into a source while the saddle does not change its type. Our algorithm consists of two stages. First, the system determines an isolating block R with the trivial Conley index such that R encloses the singularity pair in its interior. Second, the flow inside R is replaced with a new, singularity-free vector field. Figure 8 provides an illustration of the idea. Later, we use a similar two-stage approach for moving a singularity (Section 5.3.2).

Let $\mathbf{s}_+$ and $\mathbf{s}_-$ be the source and saddle, respectively. When there is a unique connecting separatrix between them, we can construct an isolating block R containing $\mathbf{s}_+$ and $\mathbf{s}_-$ on which the cancellation can be performed [Mischaikow and Mrozek 2002]. We need the following definition:

DEFINITION 5.1. For a given vector field V, let φ denote the flow induced by V. For a region Q in the domain, we

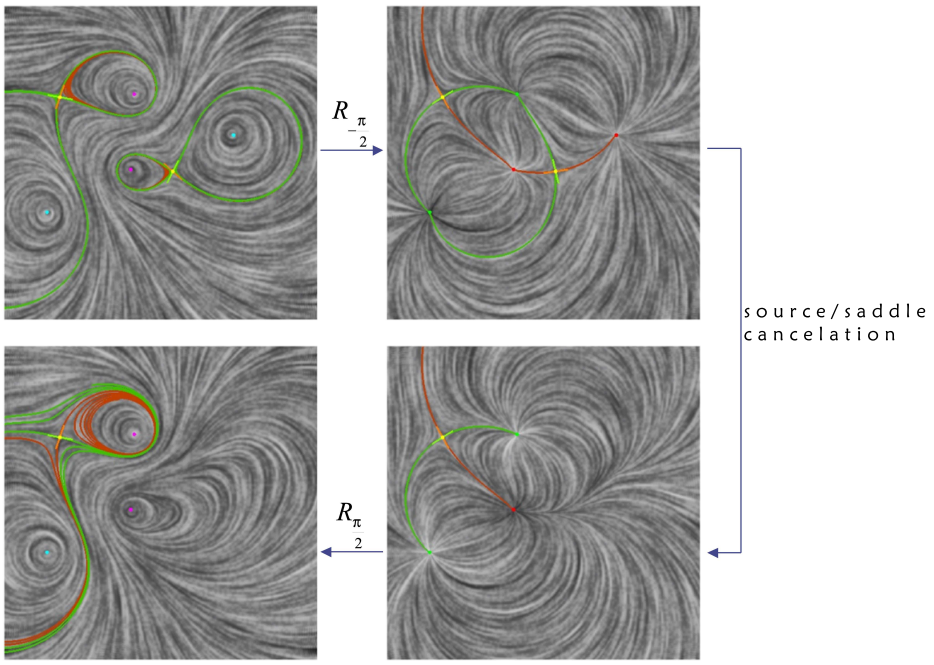

Fig. 10. This figure shows how flow rotations help overcome the numerical difficulties associated with high curl in a vector field. A center/saddle cancellation is performed on a vector field (upper-left) to obtain a new vector field (lower-left). The vector field is first rotated by $\frac{\pi}{2}$ (upper-right), followed by a pair cancellation (lower-right) before a compensating rotation is performed (lower-left).

define its images under the forward and reverse flow as:

$$\Omega(Q) = \varphi(Q,[0,\infty)) \quad \Omega^{-1}(Q) = \varphi(Q,(-\infty,0]). \tag{9}$$

To find the isolating block R, we begin with isolating neighborhoods M and N of $\mathbf{s}_+$ and $\mathbf{s}_-$, respectively. In general, $R = \Omega(M) \bigcap \Omega^{-1}(N)$ is an isolating neighborhood. If there exists a unique separatrix going from $\mathbf{s}_+$ to $\mathbf{s}_-$, then the Conley index of R is trivial and it is possible to replace the vector field inside R with one that is singularity free [Mischaikow and Mrozek 2002] (Figure 9, left).

Next, we describe a practical algorithm for computing R over a domain represented by a triangular mesh. Let M and N be sets of triangles that enclose $\mathbf{s}_+$ and $\mathbf{s}_-$, respectively. $\Omega(M)$ is obtained by performing region growing from M and following the flow forward. Similarly, $\Omega^{-1}(N)$ is obtained by performing regions growing from N and following the flow backward. We need the following definition:

DEFINITION 5.2. Given a vector field V and a polygonal region R, a boundary edge e is an *exit* for the *forward* flow with respect to V if $\max_{\mathbf{p}\in e}(N_\mathbf{p} \cdot V_\mathbf{p}) > 0$. Similarly, e is an *exit* for the *backward* flow with respect to V if $\min_{\mathbf{p}\in e}(N_\mathbf{p} \cdot V_\mathbf{p}) < 0$. $N_\mathbf{p}$ is the outward normal to the region at point $\mathbf{p}$.

If V is a piecewise linear vector field on a boundary edge e, then e is an exit edge for the *forward* flow with respect to V if $\max(V(v_1)\cdot N_e, V(v_2)\cdot N_e) > 0$. Similarly, e is an exit edge for *backward* flow with respect to V if $\min(V(v_1)\cdot N_e, V(v_2)\cdot N_e) < 0$. Here, N_e is the outward normal to the region along edge e.

Let M be the triangle that contains $\mathbf{s}_+$. Starting from M, we construct $\Omega(M)$ by adding one triangle at a time and keeping track of the behavior of the flow on the boundary edges of $\Omega(M)$. A new triangle can be added only by crossing an exit edge. The region growing process continues until there are no more exit edges, i.e., the flow enters

$\Omega(M)$ everywhere on its boundary.

$\Omega^{-1}(N)$ is constructed in a similar fashion by starting from N and following the flow backward. However, the choice of N is a delicate issue, and its choice affects the shape of $\Omega^{-1}(N)$ and subsequently R. Due to the limited resolution of the underlying mesh, R needs to be as large as possible, so long as its Conley index remains trivial. We perform a linear search on the length of the outgoing separatrices of $\mathbf{s}_-$ such that the covering triangles form N. Figure 9 shows the effect of following these separatrices to varying lengths.

To replace the flow inside R, we use flow smoothing. As described earlier, this operation tends to simplify the vector field topology, and our numerical results indicate that flow smoothing is efficient for singularity pair cancellation as long as the region R has a reasonable shape. The following pseudo-code illustrates our algorithm for cancelling a source/saddle pair where the source $\mathbf{s}_+$ has a zero curl.

```
1  PairCancalation(V, s+, s-)
2  Let M be the triangle containing s+, and we use region growing to find Ω(M).
3  Let γ = 1, dγ = 0.5, and V_working = V_1.
4  Let S_1 and S_2 be the two outgoing separatrices at s-.
5  while γ > 0 and dγ > δ (delta is a user-specified constant)
6    Let S_{1,γ} ⊂ S_1 be the portion starting from s- such that length(S_{1,γ}) = γ length(S_1). Define S_{2,γ} similarly.
7    Let N be the minimal set of triangles that contains S_{1,γ} and S_{2,γ}, and compute Ω^{-1}(N) using region growing.
8    R = Ω(M) ⋂ Ω^{-1}(N).
9    if R does not satisfy the necessary Conley condition,
10     dγ = dγ/2, γ = γ − dγ.
11   else
12     perform smoothing in R.
13     if the resulting flow contains any singularity
14       undo smoothing.
15       dγ = dγ/2, γ = γ − dγ.
16     else
17       update V_working.
18       dγ = dγ/2, γ = γ + dγ.
19     end if
20   end if
21 end while
```

In Line 9, in order to meet Conley conditions, R must be simply-connected, contain no other singularities except $\mathbf{s}_+$ and $\mathbf{s}_-$, and have a trivial Conley index. The purpose of the binary search on γ is to determine an optimal length along the separatrices such that the region R has a reasonable shape. When $\gamma = 0$, R is a narrow region that covers the singularity pair and their connecting orbits with a trivial Conley index. When $\gamma \to 1$, R tends to have a nice shape. However, it may cover other singularities, such as the sinks that are linked to $\mathbf{s}_-$ through the outgoing separatrices, and the Conley index of R is no longer trivial. The binary search process balances between the two factors, and it tends to converge very quickly since region growing and smoothing is very fast. In addition, when $d\gamma$ is small enough, γ and $\gamma + d\gamma$ correspond to the same set of triangles when computing N, and the computation can simply be avoided.

Flow rotation is crucial for the success of pair cancellation operations. If the original vector field has high curl around $\mathbf{s}_+$ as in the case of a divergence-free flow, the connecting orbit between the singularities may not even exist. Figure 10 demonstrates how our system cancels a center and saddle pair (upper-left). The flow is first rotated by $\frac{\pi}{2}$

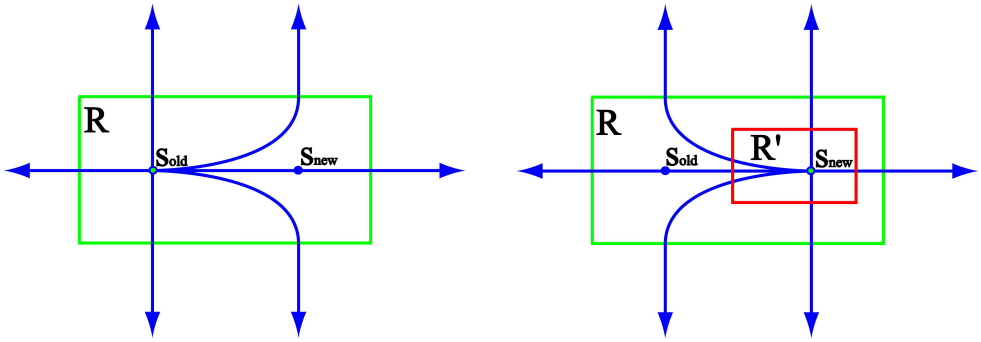

Fig. 11. This figure illustrates the concept of moving a source from $\mathbf{s_{old}}$ to $\mathbf{s_{new}}$. An isolating block R is found to enclose both $\mathbf{s_{old}}$ and $\mathbf{s_{new}}$ such that R has the Conley index of a source. Then a small region R' is found to enclose $\mathbf{s_{new}}$, and appropriate vector values are assigned to $\partial R'$ such that it forces a source at $\mathbf{s_{new}}$. For region $R \setminus R'$, flow smoothing operation produces a new vector field without any singularity.

into a curl-free vector field (upper-right) in which the center becomes a source and there is now a connecting orbit between the source and the saddle. Next, the source and the saddle are cancelled. Finally, a compensating rotation of $-\frac{\pi}{2}$ is performed (lower-left).

5.3.2 *Singularity Movement.* Moving a singularity to a new location provides the user with control over the position of the singularities in a vector field. To our knowledge, this is the first time such an operation is proposed and an algorithm is presented. Through flow reflection and flow rotation, the problem of moving a singularity is reduced to moving a source. Similar to singularity pair cancellation, our algorithm for moving a source is based on Conley index theory and is carried out in two stages. First, we compute an isolating block R such that it encloses the connecting orbit for the current location $\mathbf{s_{old}}$ and the desired new location $\mathbf{s_{new}}$ under the current vector field V. By construction, R has the Conley index of a source and does not contain any other singularities either in its interior or on its boundary (case (b) in Figure 4). Second, the vector field inside R is modified to contain only one singularity at $\mathbf{s_{new}}$ (Figure 11).

Let $R = \Omega(M) \bigcap \Omega^{-1}(N)$, where M is a small neighborhood of $\mathbf{s_{old}}$ and N is a neighborhood of $\mathbf{s_{new}}$. To ensure $\mathbf{s_{new}}$ is in the interior of R, another point $\mathbf{s}'$ is located such that it is on the forward trajectory from $\mathbf{s_{new}}$ under V. Let us consider the trajectory J of $\mathbf{s}'$ under the flow $R_{\frac{\pi}{2}}(V)$. J serves the same purpose as the outgoing separatrices of the saddle in pair cancellation. Let N be the largest segment on J that makes R an isolating block with the Conley index of a source. This ensures that R is a wide region.

Let T be the triangle that contains $\mathbf{s_{new}}$. Our system assigns vector values at the three vertices of T to force a source at $\mathbf{s_{new}}$. Let $R' = \{T\}$. Then region $L = R \setminus R'$ has two boundaries. The flow enters L from the inner boundary and leaves at the outer boundary. L therefore has the trivial Conley index (see Figure 4(e)), and flow smoothing inside L usually produces a vector field without singularities.

Moving a center or a saddle is considerably more difficult than moving a source. First, finding a connecting orbit between the saddle and a regular point is numerical unstable. Moreover, finding a connecting orbit between a center and a regular point is almost impossible. To solve these numerical issues, we make use of flow rotations and reflections to make singularity movement applicable to any linear singularity. If $\mathbf{s_{old}}$ is a saddle, we use flow reflection to turn it into a positive index singularity. If the vector field has high curl around $\mathbf{s_{old}}$, then we rotate the vector field so that the flow is converted to a vector field that has little curl at $\mathbf{s_{old}}$. This simplifies the process of locating the connecting orbit between $\mathbf{s_{old}}$ and $\mathbf{s_{new}}$. Figure 12 provides an example of moving a saddle in a vector field (upper-left). First, flow reflection is applied (upper-right) to turn the saddle into a source. Next, the source is moved (lower-right) before a compensating reflection is applied (lower-left).

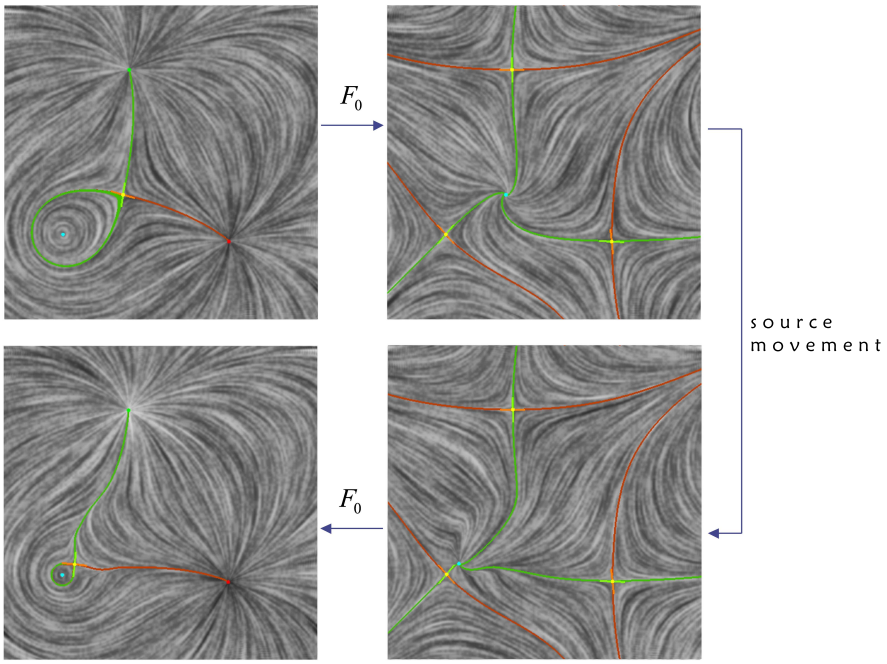

Fig. 12. This figure shows how flow reflections help overcome the numerical difficulties associated with saddles. A singularity movement operation is applied to a vector field (upper-left) to obtain a new vector field (lower-left). The vector field is first reflected so that the saddle becomes a source (upper-right), followed by a source movement (lower-right) before a compensating reflection is performed (lower-left).

6. DESIGN FOR 3D MESH SURFACES

In this section, we describe how we adapt our three-stage vector field system for planar domains to mesh surfaces. There are several challenges that we must overcome. First, a 3D surface often lacks a global parameterization. However, such a parameterization allows the correlation of tangent vectors defined at different locations, which is needed to build surface-based basis vector fields from design elements. Second, topological analysis of vector fields requires a scheme that interpolates vector values defined at the vertices and produces a continuous vector field everywhere inside triangles and along edges. However, tangent planes of a mesh surface are discontinuous at the vertices and edges, and the definition of vector field continuity from smooth manifolds does not apply. In addition, as we will demonstrate in Section 6.2.1, the piecewise linear interpolation scheme that works well for planar vector fields will cause inconsistent vector fields across edges. To address these issues, we borrow ideas of *geodesic polar maps* and *parallel transport* from classical differential geometry to set up correlations between tangent vectors defined at different parts of the surface. The correlations are used for two purposes. First, we extend the construction of basis vector fields (Section 4.1) to surfaces by parallel transporting tangent vectors from the location of the design element to anywhere on the surface. Second, we adapt the piecewise linear approximation from planar domains (Section 4.2) to mesh surfaces by parallel transporting vector values from a vertex to anywhere inside its 1-ring neighborhood. The piecewise interpolation scheme results in a continuous surface vector field, and it supports efficient vector field analysis and editing operations.

6.1 *Basis Vector Fields for Initialization*

In this section, we describe an approach in which surface basis vector fields are directly constructed from design elements. Recall that in the planar case, a design element O is converted into a global basis vector field according to Equations 3 and 4. To extend this to surfaces, we perform the following three-step process, as illustrated in Figure 13.

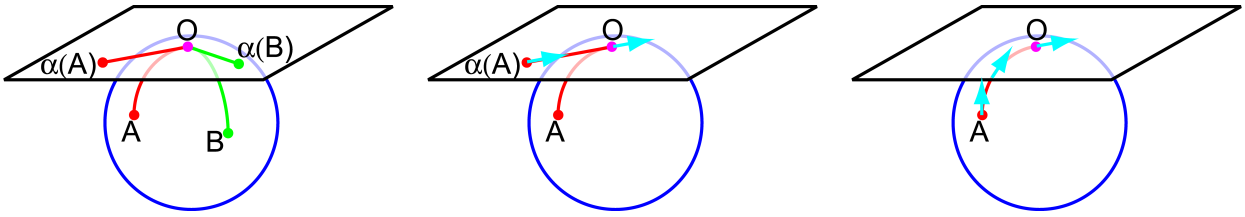

Fig. 13. Our three-step algorithm (from left to right) for creating a surface basis vector field from a user-specified constraint O. First, the surface is parameterized using a *geodesic polar map* with respect to O. This parameterization is denoted as α. Second, the basis vector field is computed inside the tangent plane at O with the polar coordinates from α. Finally, the vectors are *parallel transported* along shortest geodesics on the surface.

First, we compute a *geodesic polar map* with respect to the location of O (left), which assigns every point A on the surface with a pair of coordinates (x_A, y_A). This can seen as building a global parameterization for the surface using the tangent plane at O. Next, (x_A, y_A) are substituted into Equation 3 or 4 to obtain a tangent vector value W_A defined at O (middle). Finally, W_A is *parallel transported* from O to A along the shortest geodesic connecting them (right). The process is based on several ideas from classical differential geometry, namely, *geodesics*, *geodesic polar maps*, and *parallel transport*. We will review each of these in turn.

A geodesic on a curved surface is a locally shortest and straightest curve. It is a generalization of a straight line in the plane. Starting from a point $\mathbf{p}$ on the surface, there is a geodesic in every tangent direction $\overrightarrow{v}$. Denote this geodesic by $\gamma_{\mathbf{p},\overrightarrow{v}}$. A point $\mathbf{q}$ on $\gamma_{\mathbf{p},\overrightarrow{v}}$ with a distance ρ from $\mathbf{p}$ can be identified by the coordinates (ρ,θ). Here θ is the angular coordinate of $\overrightarrow{v}$ with respect some local frame at $\mathbf{p}$. In the plane, the coordinates reduce to the familiar polar coordinates. This map is the *geodesic polar map*. On a curved surface, a geodesic polar map is neither bijective nor continuous. For example, on the Earth, a geodesic polar map with respect to the North Pole will have discontinuity at the South Pole. However, since the focus of a design element is in a nearby region, the geodesic polar map with respect to the design element meets our needs.

In differential geometry, parallel transport is used to correlate tangent vectors that are defined at different locations using a geodesic that connects them. Formally,

DEFINITION 6.1. Let $\mathbf{p}$ and $\mathbf{q}$ be two points on a smooth manifold S, and let $\gamma : [0,1] \rightarrow S$ be a geodesic such that $\gamma(0) = \mathbf{p}$ and $\gamma(1) = \mathbf{q}$. Furthermore, let $V_\mathbf{p}$ and $V_\mathbf{q}$ be tangent vectors defined at $\mathbf{p}$ and $\mathbf{q}$, respectively. Then $V_\mathbf{p}$ and $V_\mathbf{q}$ are said to be *parallel* with respect to γ if the oriented angle between $\gamma'(0)$ and $V_\mathbf{p}$ equals that between $\gamma'(1)$ and $V_\mathbf{q}$. Furthermore, $V_\mathbf{q}$ is said to be the *parallel transport* of $V_\mathbf{p}$ along γ.

In the above definition, γ gives rise to an orthonormal and bijective linear map between $TM_\mathbf{p}$ and $TM_\mathbf{q}$, the tangent planes at $\mathbf{p}$ and $\mathbf{q}$. This map is a *transport* function and is denoted by $f_\mathbf{pq}$.

For a design element d, let $\alpha_d : S \rightarrow \mathbb{R}^2$ be a geodesic polar map with respect to d, and let $f_{d\mathbf{p}} : TM_d \rightarrow TM_\mathbf{p}$ be the transport function along a geodesic $\gamma_{d\mathbf{p}}$. Then the surface basis vector field $W(\mathbf{p})$ corresponding to a design element d is constructed as $W(\mathbf{p}) = f_{d\mathbf{p}} V(\alpha_d(\mathbf{p}))$. In this equation, V is evaluated according to Equation 3 or 4. For the purpose of building the geodesic polar map α_d and computing the transport function $f_{d\mathbf{p}}$, we need to compute a geodesic from any vertex of the surface to the design element d. In the following section, we will describe how to create a continuous vector field everywhere on the surface by interpolating the vector values defined at the vertices.

Assume that the design element d is situated inside a triangle T. We first compute the geodesic distance function g_d with respect to d for every vertex using the fast marching method [Kimmel and Sethian 1998]. The values of g_d at a vertex is the radial (ρ) component of the geodesic polar map. To construct the angular (θ) component in the ideal situation, one needs to perform particle tracing from a vertex in the opposite direction of $\bigtriangledown g_d$, the gradient vector field of g_d. However, performing particle tracing for every vertex is expensive. In addition, $-\bigtriangledown g_d$ often has local minima other than d. To overcome these problems, we propose a two-region approach in which the angular component θ is computed directly *only* within a surface disk surrounding d such that the disk contains a user-specified percentage of

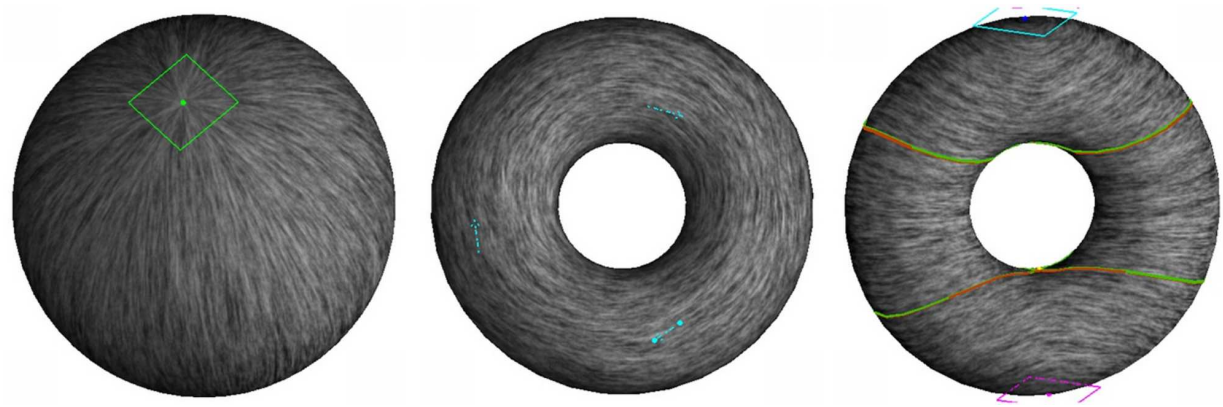

Fig. 14. Design elements for creating an initial vector field. From left to right: a dipole vector field on a sphere using a source element, a singularity-free vector field on a torus with three regular elements, and another vector field on the torus with a clockwise center element and a counter clockwise element. Notice the surface basis vector fields are very efficient for creating initial vector fields.

the total vertices in the mesh. We use 25% in practice. For a point inside the disk, we project it onto the tangent plane at d to obtain θ. For a vertex $\mathbf{p}$ outside the disk, we perform particle tracing from $\mathbf{p}$ in the direction of $-\bigtriangledown g_d$ until it hits an edge $e = \mathbf{rs}$ that is on boundary of the disk. If $\theta(\mathbf{r})$ and $\theta(\mathbf{s})$ are both known, then $\theta(\mathbf{p})$ is obtained by linearly interpolating between $\theta(\mathbf{r})$ and $\theta(\mathbf{s})$. If particle tracing from $\mathbf{p}$ fails to reach any boundary edge, then there is not a shortest geodesic between $\mathbf{p}$ and d. In this case, a random θ value is assigned to $\mathbf{p}$. Although this may seem to have created discontinuities in the vector field, let us recall that the vector values are only computed at the vertices at this stage. In the next section, we will describe a piecewise interpolation scheme in which a continuous vector field is created based on the values defined at the vertices.

The above method works well for nearly flat or spherical regions. However, nearly cylindrical features, the projection is likely to result in undesired vector values on the side of the cylinder opposite to the location of the design element. We are investigating other possible surface parameterization, such as cylindrical coordinates, to address such cases.

Given a geodesic polar map, a tangent vector can be parallel transported to a vertex along a geodesic. This completes the construction of a basis vector field. Figure 14 provides three example vector fields created using basis vector fields: a dipole vector field on a sphere with a single source element (left), a singularity-free vector field on a torus with three regular elements (middle), and another vector field on a torus with a clockwise center element and a counterclockwise center element (right).

Let us stress that this is not the only way to create basis vector fields. In van Wijk's visualization tool [2003], an element is translated into a 3D vector field before being projected onto the surface. While our approach appears to be more intuitive in this case given that a surface is locally homeomorphic to a plane, van Wijk's 3D projection method is faster since it does not require the construction of geodesic polar maps. Constrained optimization [Turk 2001; Wei and Levoy 2001] is another way to produce an initial vector field with desired behaviors. Praun et al. [2000] propose a vector field propagation approach in which a vector value is defined inside one face of the mesh surface. Through region growing, the vector value for a new triangle is obtained by computing the average tangent vectors at its neighboring triangles that are already part of region. This vector is then projected onto the face.

6.2 *Vector Field Continuity and Piecewise Approximation for Vector Fields on Meshes*

Once the vector values are obtained at the vertices, we use a piecewise interpolation scheme to construct a continuous vector field on the mesh surface. Unfortunately, the piecewise linear approximation that we have used in the planar

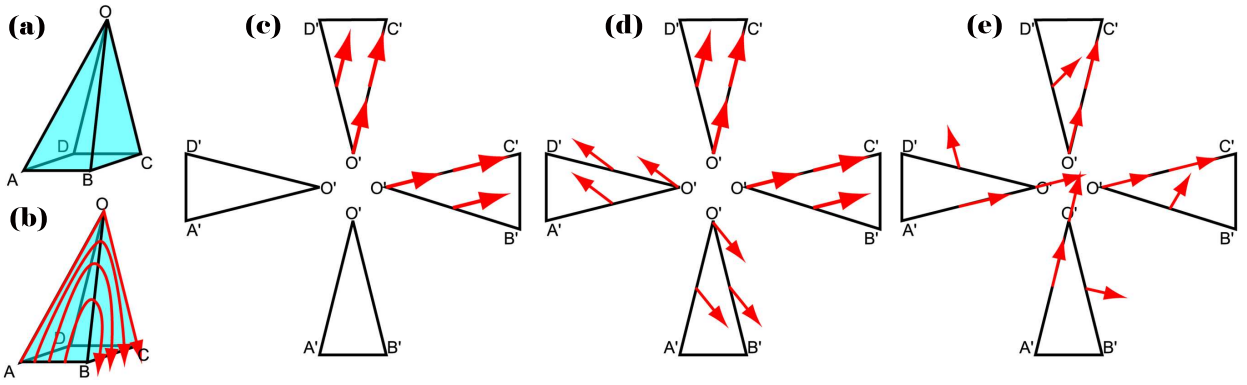

Fig. 15. This figure illustrates that the piecewise linear representation does not produce continuous vector fields on mesh surfaces. The vector values are zero at *A*, *B*, *C*, and *D*. The vector value at *O* is in the direction of $\overrightarrow{OC}$ (a). The piecewise linear representation and vector field consistency along edge *OD* and *OB* eventually lead to vector field discontinuity along edge *OA* (c and d). In contrast, our piecewise interpolation scheme (Section 6.2) produces a continuous vector field (e), which corresponds to a family of non-intersecting and spacing-filling trajectories in the 1-ring neighborhood of O (b).

case does not produce consistent vector fields on mesh surfaces. Figure 15 illustrates the problem for a vertex *O* and its 1-ring neighborhood (a). The vector values are zero at *A*, *B*, *C*, and *D*, and $V(O)$ is in the direction of $\overrightarrow{OC}$. With the piecewise linear representation, the vector values at the midpoints of edge *OD* and *OB* are fixed (c). Due to the continuity constraints across edges, the vector values at the middle points *OA* in triangle $\triangle ODA$ and $\triangle OAB$ lead to inconsistencies (d). The problem is due to the angle deficit caused by the discontinuity of tangent planes at the vertices and across the edges. However, we need consistent vector fields for compute and control over vector field topology.

For planar domains, the concept of vector field continuity is well-defined because any two vectors can be compared regardless their locations. This is no longer true for a general surface since the tangent planes at different locations are distinct and there is not an obvious and consistent way to correlate them without a global parameterization. Furthermore, the tangent planes of mesh surfaces are often discontinuous across the vertices and the edges. Stam [2003] addresses the problem by using a subdivision surface, whose tangent planes are continuous everywhere. However, for most geometric processing operations, subdivision surfaces incur higher computational costs than polygonal meshes.

In this section, we describe an interpolation scheme that is guaranteed to produce a continuous vector field directly on mesh surfaces. This scheme is a generalization of the piecewise linear representation from the planar case, and it allows fast and efficient analysis and editing of vector fields on meshes. Before describing the scheme, however, we first need a definition of vector field continuity for mesh surfaces.

Recall that for a smooth vector field, a point is either a singularity or a regular point. We can always define singularities for surface vector fields because zero vectors can be identified regardless of locations. In addition, the *flow-box theorem* for Ordinary Differential Equations gives us a picture of what happens near a regular point [Hale and Kocak 1991].

THEOREM 6.2. *Let V be a smooth vector field defined in $D \subset \mathbb{R}^n$. If $\mathbf{p}_0 \in D$ is a regular point of V, then there exists a neighborhood U of $\mathbf{p}_0$ and a homeomorphism $h : U \to \mathbb{R}^n$ which carries each piece of a trajectory lying on U onto a straight line of $\mathbb{R}^n$ parallel to the X-axis.*

In other words, near a regular point, it is possible to warp the space such that the nearby trajectories are parallel and space-filling. We propose to use this property as the definition for *vector field consistency* (continuity) for a regular point on mesh surfaces. Notice that the flow-box theorem is true even when the vector field V is only *Lipschitz-continuous* [Calcaterra and Boldt 2003], which is a stronger condition than continuity, but weaker than smoothness.

Basically, A vector field V over a domain is Lipschitz-continuous if there exists a constant K such that for any $\mathbf{x}, \mathbf{y}$ in the domain, $|V(\mathbf{x}) - V(\mathbf{y})| < K|\mathbf{x} - \mathbf{y}|$. We propose the following definition:

DEFINITION 6.3. Let V be a vector field defined on a mesh surface $\mathbf{M}$. V is *consistent* at a point $\mathbf{p}_0 \in \mathbf{M}$ if one of the following situations is true:

(a) For any *path* $\gamma : [0,1) \rightarrow \mathbf{M}$ such that $V(\gamma(t))$ is well defined for any $t \in [0,1)$ and $\lim_{t\rightarrow 1} \gamma(t) = \mathbf{p}_0$, we have $\lim_{t\rightarrow 1} V(\gamma(t)) = 0$. In this case, $\mathbf{p}_0$ is a *singularity*.

(b) There exists a neighborhood U of $\mathbf{p}_0$ and a homeomorphism $h : U \rightarrow \mathbb{R}^2$ which carries each piece of a trajectory lying in U onto a straight line in $\mathbb{R}^2$ parallel to the x-axis. In this case, $\mathbf{p}_0$ is *regular*.

In other words, a consistent vector field on a mesh surface should exhibit the same local behaviors as those defined in a plane. Notice in this definition, we require continuity at singularities, and unique solvability at regular points. Unique solvability refers to the fact that for any point $\mathbf{p}_0$, there exists a unique solution to the differential equation induced by the vector field. Observe that as curves on a continuous surface, it makes sense to discuss the continuity of the trajectories of a vector field. Figure 15 (e) illustrates the result of our interpolation scheme to be described next (compare this to d). Notice that this scheme leads to a family of non-intersecting and space-filling trajectories in the 1-ring neighborhood of O (b).

6.2.1 *Piecewise Approximation.* For every vertex in the mesh, we record its surface normal and the coordinate system for the tangent plane. This allows us to easily transform a tangent vector from its local coordinates to global coordinates.

A vector field V on a mesh surface is represented by assigning tangent vectors $\{W_1, W_2, ..., W_n\}$ at the mesh vertices $\{v_1, v_2, ..., v_n\}$. For each W_i, we maintain its local coordinates for vector field design and 3D coordinates for display. We cannot simply perform interpolation of W_i's since they are in general not co-planar. Furthermore, without a surface parameterization, tangent vectors that are defined at different vertices are not correlated. To overcome these problems, we first define a local parameterization for the 1-ring neighborhood of a vertex v_i. This parameterization allows the parallel transport of W_i to any point $\mathbf{p}$ inside v_i's 1-ring neighborhood. Let μ_i be such transport function (which we will soon describe). Then, for a point $\mathbf{p}$ inside a triangle $T = \{v_{T_1}, v_{T_2}, v_{T_3}\}$ whose barycentric coordinates are $(\alpha_1, \alpha_2, \alpha_3)$, Equation 5 can now be rewritten as the weighted sum of the tangent vectors that are parallel transported from the three vertices:

$$V(\mathbf{p}) = \sum_{j=1}^{3} \alpha_j \mu_{T_j}(W_{T_j}, \mathbf{p}) \tag{10}$$

Let us consider V_i, the vector field that is constructed according to Equation 10 under the assumption that $W_j = 0$ for every $j \neq i$. We have $V = \sum_{i=1}^{n} V_i$. Notice V_i is zero outside the 1-ring neighborhood of vertex v_i. As we will see soon, V_i is a consistent vector field over the mesh surface for every i, and so is V. Before we describe the parameterization and the transport function in detail, we need the following definitions [Polthier and Schmies 1998].

DEFINITION 6.4. Let M be a polyhedral mesh representing a closed curved surface. Let v be a vertex with incident triangles T_j ($j = 1, ..n$), and θ_j be interior angle of T_j at v. Then

(1) the *total vertex angle* at $\theta(v)$ is given by $\theta(v) = \sum_{j=1}^{n} \theta_j$.
(2) $2\pi - \theta(v)$ is the *Gaussian curvature* at v. A vertex v is called *Euclidean* if has a zero Gaussian curvature; otherwise, it is *non-Euclidean*.

Let $r = \frac{\theta(v)}{2\pi}$. Notice that $r = 1$ for vertices with a zero Gaussian curvature. There are two ways of measuring angles between two rays emanating from a vertex v. The first is the *Euclidean angle*, which is the angle measured on the mesh. The second angle is the *normalized angle* as measured in the tangent space (each ray corresponds to a tangent vector). The normalized angle α is related to the Euclidean angle β by $\beta = r\alpha$ [Polthier and Schmies 1998]. Figure 16 provides

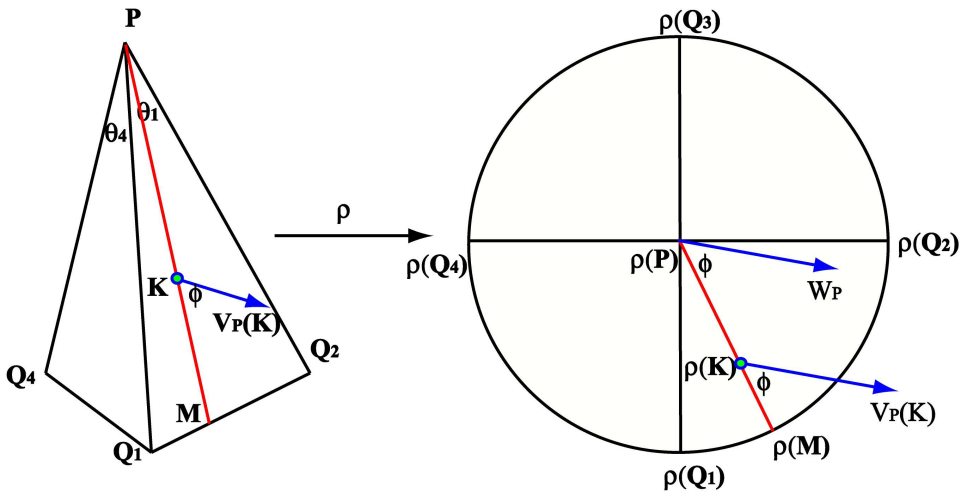

Fig. 16. This figure illustrates the idea of parallel transporting a tangent vector W_i from a vertex $v_i = \mathbf{P}$ to a point $\mathbf{K}$ inside $\mathbf{P}$'s 1-ring neighborhood, R. First, we build a local parameterization ρ for R. Then, $W_\mathbf{P}$ is parallel transported to $\mathbf{K}$ along the ray $\overrightarrow{\rho(\mathbf{P})\rho(\mathbf{K})}$. This construction guarantees the vector field consistency on mesh surfaces.

an illustration. In the left portion, $\mathbf{P} = v_i$ is a vertex with the tangent plane $TM_\mathbf{P}$ (right). Its 1-ring neighborhood R consists of the triangles $\triangle\mathbf{PQ}_1\mathbf{Q}_2$, $\triangle\mathbf{PQ}_2\mathbf{Q}_3$, ..., and $\triangle\mathbf{PQ}_n\mathbf{Q}_1$ ($n = 4$). Let $\theta_j = \angle\mathbf{Q}_j\mathbf{PQ}_{j+1}$. Then $\theta(\mathbf{P}) = \sum_{j=1}^{n}\theta_j$ and $r = \frac{\theta}{2\pi}$. In the right portion, let D be the unit disc in $TM_\mathbf{P}$ and let ρ be the following homeomorphism from R to D.

(1) ρ induces a bijective mapping between the boundary of R and the boundary of the unit circle. For any point $\mathbf{M} \in \partial R$, ρ is a linear map from $\overrightarrow{\mathbf{PM}}$ to $\overrightarrow{\rho(\mathbf{P})\rho(\mathbf{M})}$.

(2) ρ is linear scaling on the angles between rays. If two rays emanating from $\mathbf{P}$ have an Euclidean angle of θ, then the angle between their images (normalized angle) is $r\theta$.

Note that this construction is similar to the "geodesic polar maps" used by Welch and Witkin [1994] for free-form shape design, with a minor difference: in their setting the parameterization domain is a polygon, not the unit disc as in our case. Polthier and Schmies [1998] have used similar maps to perform parallel translation on vectors over a polygonal surface.

To transfer W_i to a point $\mathbf{K}$ inside triangle $\mathbf{Q}_j\mathbf{PQ}_{j+1}$, we first locate the ray $\overrightarrow{\mathbf{PM}}$ that contains $\mathbf{K}$. Let ϕ be the counter-clockwise angle between W_i and the ray $\overrightarrow{\rho(\mathbf{P})\rho(\mathbf{M})}$, then we define $\mu_i(W_i, \mathbf{p})$ as the vector at $\mathbf{K}$ such that the angle between $\mu_i(W_i, \mathbf{p})$ and $\overrightarrow{\mathbf{PM}}$ equals ϕ. Furthermore, $|\mu_i(W_i, \mathbf{p})| = |W_i|$.

Basically, we have created a constant vector field ($= W_i$) inside the unit disk (Figure 16, right), which is then scaled such that the magnitude is one at the origin and it linearly decreases to zero along each line segment connecting the origin and a point on the boundary of the disk. Notice the resulting vector field is Lipschitz-continuous. Finally, the scaled vector field is mapped to the 1-ring neighborhood to obtain V_i through parallel transport as described above. Note that the distortion contained in the parameterization ρ is caused by the Gaussian curvature of v_i, and therefore has an upper bound. This implies that V_i is also Lipschitz-continuous. As a result, vector field consistency (continuity and unique solvability) is ensured. Intuitively, as a parameterization, ρ does not distinguish between points inside a triangle or on an edge. Consequently, vector field continuity is automatically guaranteed there. Furthermore, the continuity of ρ ensures the continuity of the resulting vector field at the vertices. For planar domains, $r = 1$ everywhere and this

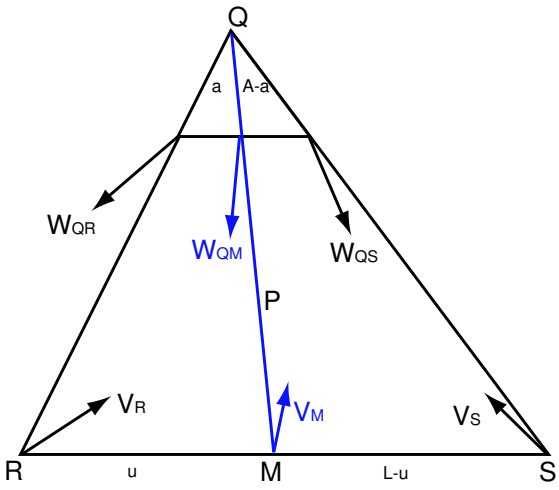

Fig. 17. This figure illustrates how to determine the location of a singularity inside a triangle under the piecewise interpolation scheme. Here, Q is the only non-Euclidean vertex for the triangle.

approximation reduces to the piecewise linear representation (Section 4.2).

Our piecewise interpolation scheme induces a vector field W that is a continuous and non-linear inside each triangle T minus three vertices. We can think of such a region as a triangle minus three arbitrarily small corners, each around a vertex. Therefore, W can be seen as being defined over a hexagon that is arbitrarily close to T. Along each edge of the triangle, W is linear in terms of length. Along the sides where the corners are cut, W is linear in terms of vertex angle. Locating singularities of W is rather difficult under this setting. Furthermore, the Poincaré index for a hexagon can be ± 2, which implies possibly two linear singularities or one second-order singularity inside T. This makes topological control more difficult. To overcome these difficulties, we perform a four-fold triangle subdivision for the input mesh. Basically, the mid-point of every edge in the original mesh becomes a new Euclidean vertex since it total vertex angle is 2π. This means every triangle in the subdivided mesh can have at most one non-Euclidean vertex, and analysis becomes more tractable. From now on, we will assume the input mesh already satisfies this requirement.

6.2.2 *Analysis.* Our piecewise interpolation scheme results in a non-linear vector field inside a triangle, which requires new ways of computing the Jacobian, curl and divergence, and singularities and separatrices. In this section, we provide solutions to these issues. Let $T = \triangle QRS$ be a triangle with exactly one non-Euclidean vertex Q. Then the vector field V as constructed in Equation 10 is linear on RS and along any ray emanating from Q. Furthermore, W is a continuous vector field defined on T minus an arbitrarily small corner near Q, i.e., a quadrilateral as illustrated in Figure 17. Let $V_R = V(R)$ and $V_S = V(S)$ be the vector values at R and S, respectively. At Q, we need a direction to determine the vector value. Let W denote the vector field of V along an arbitrarily small line segment near Q such that $W_{QR} = W(QR)$ and $W_{QS} = W(QS)$ are vector values in the direction $\overrightarrow{QR}$ and $\overrightarrow{QS}$.

Since the Jacobian is no longer constant inside T, we compute pointwise Jacobian through local approximation. First, two points M_1 and M_2 are selected inside T such that they are sufficiently close to M_0, and $\overrightarrow{M_0M_1}$ and $\overrightarrow{M_0M_2}$ are not co-linear. Next, we build a linear vector field V' such that $V'(M_i) = V(M_i)$ for $i = 0,1,2$. Finally, the Jacobian of V at M_0 is approximated by the Jacobian of V'.

Pointwise curl and divergence are computed from the Jacobian. On the other hand, the curl and divergence for a triangle can be obtained accurately by calculating the divergence and curl along the three edges of the triangle. Notice, the piecewise interpolation scheme that we described in this section is linear along these edges. Let N_e and D_e be the

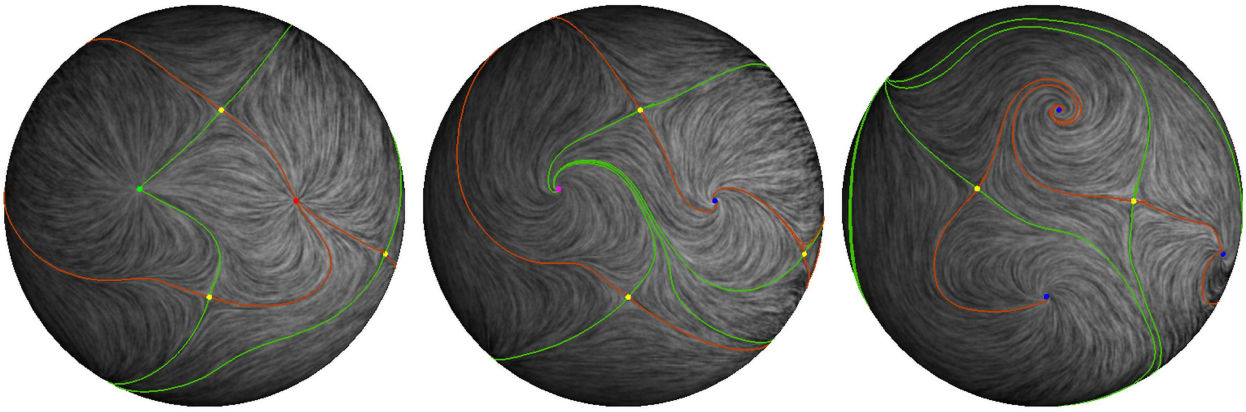

Fig. 18. Flow rotations are applied to a vector field on a sphere. Top row, from left to right are rotations of 0, $\frac{\pi}{4}$, $\frac{\pi}{2}$. Flow reflection is applied to these vector fields to produce corresponding images in the bottom row. Observe that flow rotations maintain the number, the location, and the Poincaré index of the singularities. On the other hand, flow reflection maintain the location of singularities while negating the sign of their Poincaré indices.

outward normal and directional vectors on an edge e, then we have the following results:

$$curl(T) = \frac{(W_{QR}+V_R)\cdot N_{\overrightarrow{QR}}}{2}|\overrightarrow{QR}| + \frac{(V_R+V_S)\cdot N_{\overrightarrow{RS}}}{2}|\overrightarrow{RS}| + \frac{(V_S+W_{SQ})\cdot N_{\overrightarrow{SQ}}}{2}|\overrightarrow{SQ}| \tag{11}$$

$$div(T) = \frac{(W_{QR}+V_R)\cdot D_{\overrightarrow{QR}}}{2}|\overrightarrow{QR}| + \frac{(V_R+V_S)\cdot D_{\overrightarrow{RS}}}{2}|\overrightarrow{RS}| + \frac{(V_S+W_{SQ})\cdot D_{\overrightarrow{SQ}}}{2}|\overrightarrow{SQ}| \tag{12}$$

Note that the total curl and divergence is zero for any closed 2-manifold. By construction, our piecewise interpolation scheme maintains this property for mesh surfaces by construction.

The Poincaré index of a triangle is computed for a quadrilateral as illustrated in Figure 17. Along each side, the vector field continuously and monotonically interpolates the vector values at the end points. Let us consider W_{QR} and W_{QS} as vector values defined at the ends of an arbitrarily small edge. The total index angle is in $(-4\pi, 4\pi)$, which implies that T can have at most one linear singularity. The piecewise interpolation scheme maintains the Poincaré-Hopf theorem, and our numerical results support this. If the Poincaré index of T is not zero, there must be a singularity inside. To locate the singularity P, we perform a binary search for a point $M \in \overrightarrow{RS}$ such that $V_M = V(M)$ and $W_{QM} = W(QM)$ are co-linear and they point in opposite directions. Let $W_{QM} = -\alpha V_M$, then $P = (1-m)Q + mM$ $(m = \alpha/(\alpha+1))$ is the singularity. The Jacobian at P is used to determine its type, and in the case of a saddle, the incoming and outgoing directions.

The Runge-Kutta method that we used for computing separatrices for planar vector fields (Section 4.2) can be adapted to mesh surfaces. Polthier and Schmies have also suggested a fourth-order Runge-Kutta method for computing trajectories for a continuous vector field on mesh surfaces [1998]. Figure 18 show some examples vector fields on a sphere along with their topological skeletons. The analysis is performed using the piecewise interpolation scheme that we have described in this section.

6.3 *Editing Operations*

While the main concepts for editing operations on a surface remain the same as those for planar domains, some changes need to be made to reflect the differences in vector field representation and the complexity of the surface geometry such as curvature and higher genus.

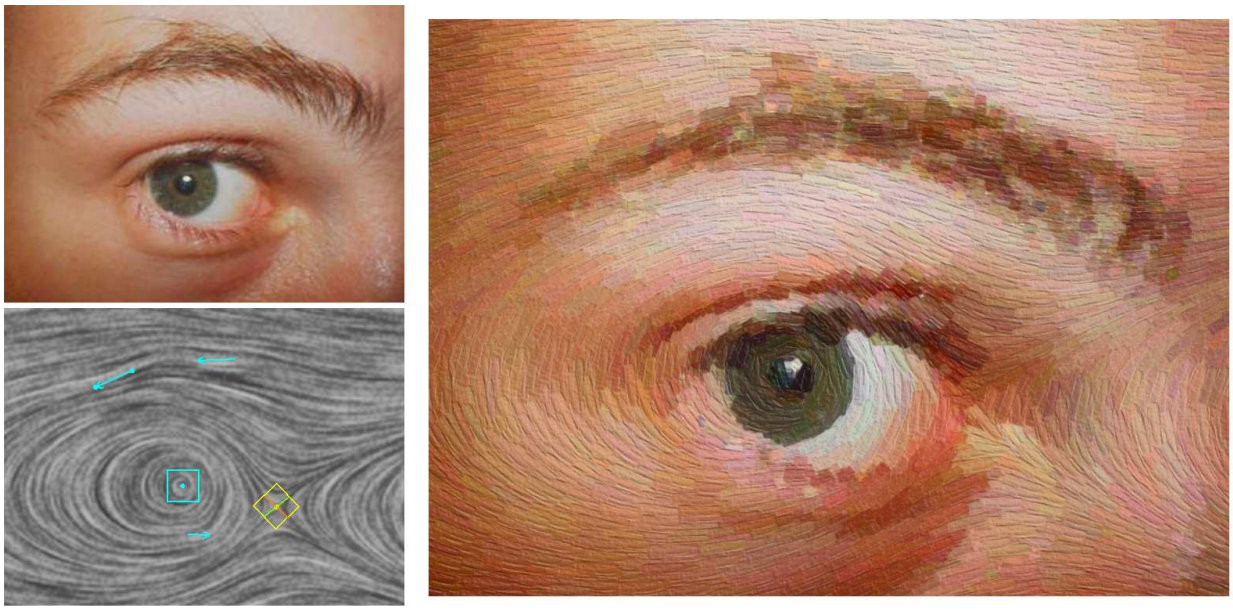

Fig. 19. Painterly rendering of a human eye image through vector field design. The input vector field was created using our system (lower-left). The high-quality van Gogh-style rendering (right) was produced off-line with the approach of Hays and Essa [2004].

To perform flow rotation on a mesh surface, we simply rotate the vector values W_i's in the tangent planes at each vertex. Since the transport functions are orthonormal transformations between the tangent planes (Section 6.2.1), rotating W_i's by an angle of θ results in a rotation of the same angle inside every triangle and edge. Therefore, flow rotations maintain the number, the location, and the Poincaré index of the singularities. Furthermore, for any point inside a triangle, Equation 6 remains valid. In contrast, flow reflection requires that local frames and reflection axes at every vertex be correlated. We make use of a global polar map to parallel transport this information. Global reflection negates the sign of the Poincaré indices. In addition, it may create some additional singularities due to the singularities in the fields of local frames and the field of reflection axes. Regardless of these issues, the resulting vector field still satisfies the Poincaré-Hopf theorem. Figure 18 illustrates the effects of flow rotations and reflections on a vector field defined over a sphere. The original field (left) is first rotated by $\frac{\pi}{2}$ (middle), then reflected (right). Compare this figure with Figure 6.

Similar to the planar case, flow smoothing on a surface vector field is carried out by performing vector-valued smoothing inside a user-specified region. We have implemented two variations of flow smoothing for meshes. In the first variation, we perform vector-valued smoothing to the original vector field as a 3D vector field and project the resulting vector field onto the surface. The second approach parameterizes R based on some planar domains and perform smoothing in this domain. Both smoothing techniques provide similar results. However, theoretically speaking, the second approach seems more natural for vector-valued smoothing on mesh surfaces.

7. APPLICATIONS

All the vector fields shown in this paper are created with our system. In addition, we have applied our vector field design system to several graphics applications: painterly rendering of images, pencil sketch illustration of smooth surfaces, and example-based texture synthesis.

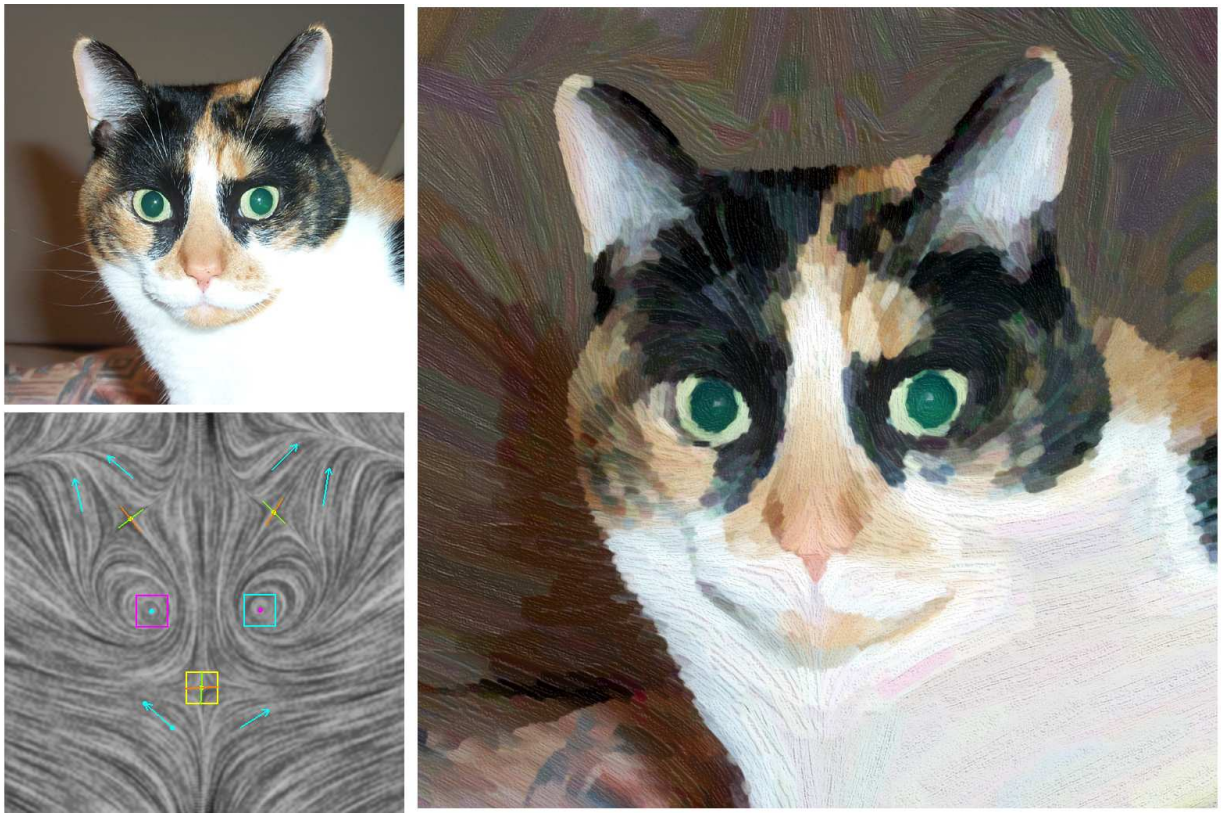

Fig. 20. Another example of painterly rendering based on vector field design: a cat's face. The high-quality impressionistic image shown in the right was also produced off-line with the approach of Hays and Essa [2004].

7.1 *Painterly Rendering*

Painterly rendering refers to creating digital images that have the appearance of being painted. There are numerous published approaches to painterly rendering, and to review them all is beyond the scope of the paper. These techniques have focused on providing the user with control over certain aspects of brush strokes (textures, styles) while automatically determining other aspects (base colors and orientations). In particular, image-based gradient fields have often been used to guide the orientation of brush strokes. While this may be appropriate for some parts of the image (near the feature lines), it often produces brush strokes with noisy orientations in areas with nearly uniform colors. Furthermore, it creates unnecessary constraints on the way that artists may express themselves. Our goal for this application is to let the user control the brush stroke orientation through vector field design.

We use a level-of-detail approach by Hertzmann [1998]. In this approach, a painting is created in a series of layers, starting with a rough sketch drawn with brush strokes of a large size. Then the sketch is painted over with brush strokes of gradually decreasing sizes at places where signals of higher-frequencies are present. This approach is very fast and has a high quality. However, we make the following modification: instead of using the image gradient field to guide the brush stroke orientations, let the user create a synthetic vector field with our vector field design system. To make this task fast and effective, we incorporate the painterly rendering algorithm into our vector field design system. In addition to viewing the vector field, the user can also switch to the painterly rendering that uses the current vector field. The results are interactively displayed as the user makes changes to the vector field. Figure 19 and 20 show the

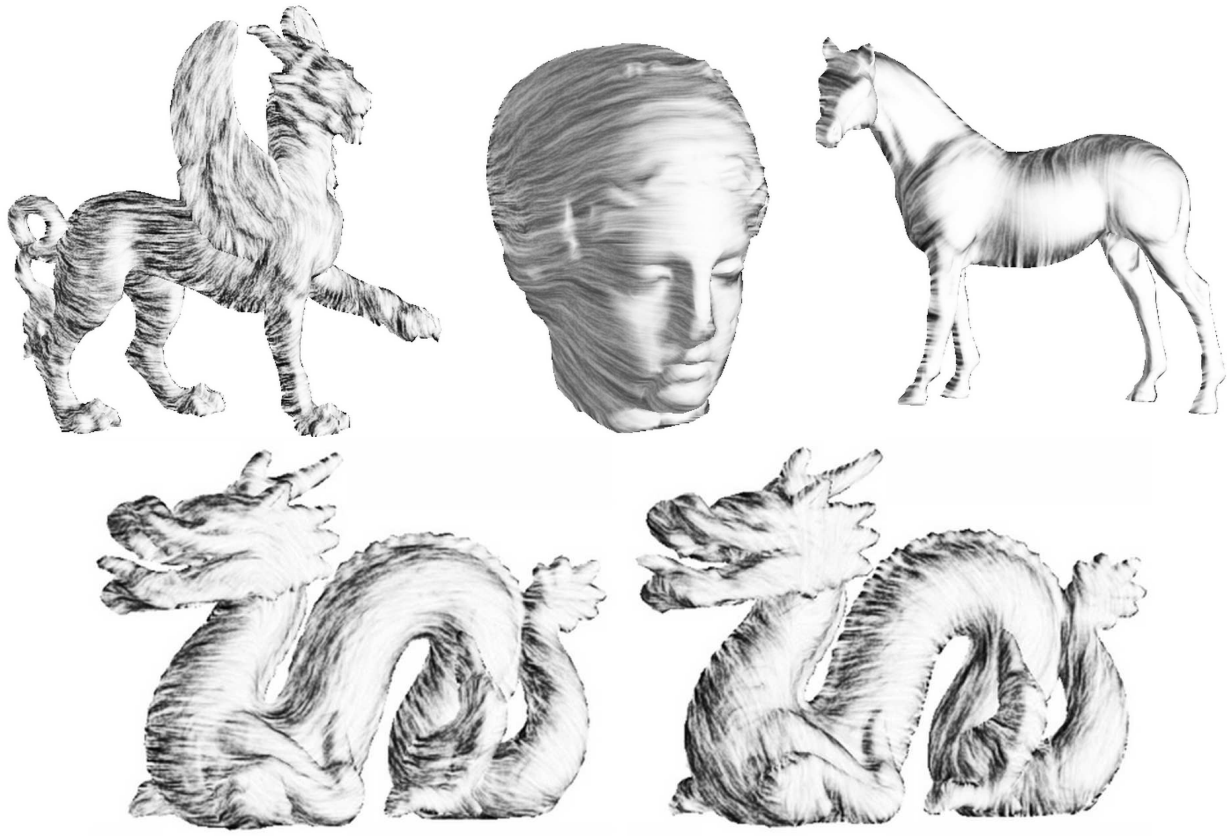

Fig. 21. We have applied vector field design to non-photorealistic illustration of 3D surfaces. The pencil style illustration is based on van Wijk's image-based flow visualization technique [2003]. The vector field used for the dragon image in the lower-right was obtained by rotating the vector field from the lower-left by $\frac{\pi}{3}$.

painterly rendering results for two source images: a human's eye (Van Gogh style) and a cat face (impressionism). For the human's eye, a center element was placed at the middle of the pupil and a saddle element was placed at the corner of the eye. Two regular elements were placed along the eyebrow to ensure that the brush strokes do not cross the feature. With five elements, a vector field (lower-left) was produced that matches the main features of the image (the eye and the eyebrow). For the cat's face, two center elements of opposite orientations were placed at the middle of each eye. A saddle element was placed underneath the nose and six regular elements were placed along the ears and the chin. The final high-quality painterly images in both figures were created off-line using the algorithm of Hays and Essa [2004].

7.2 *Non-Photorealistic Illustration of Surfaces*

There has been much work in creating hatch-based illustrations of surfaces, and to review all of them is beyond the scope of this paper. Girshick et al. [2000] have shown that the principle curvature fields are good at conveying shapes. Traditional techniques often make use of principle curvature directions to guide the hatch field. Hertzmann and Zorin [2000] present an efficient algorithm for approximating the principle curvature fields over the mesh by local surface fitting, which leads to a high-quality pen-and-ink style of rendering of 3D shapes. Praun et al. [2001] treat the

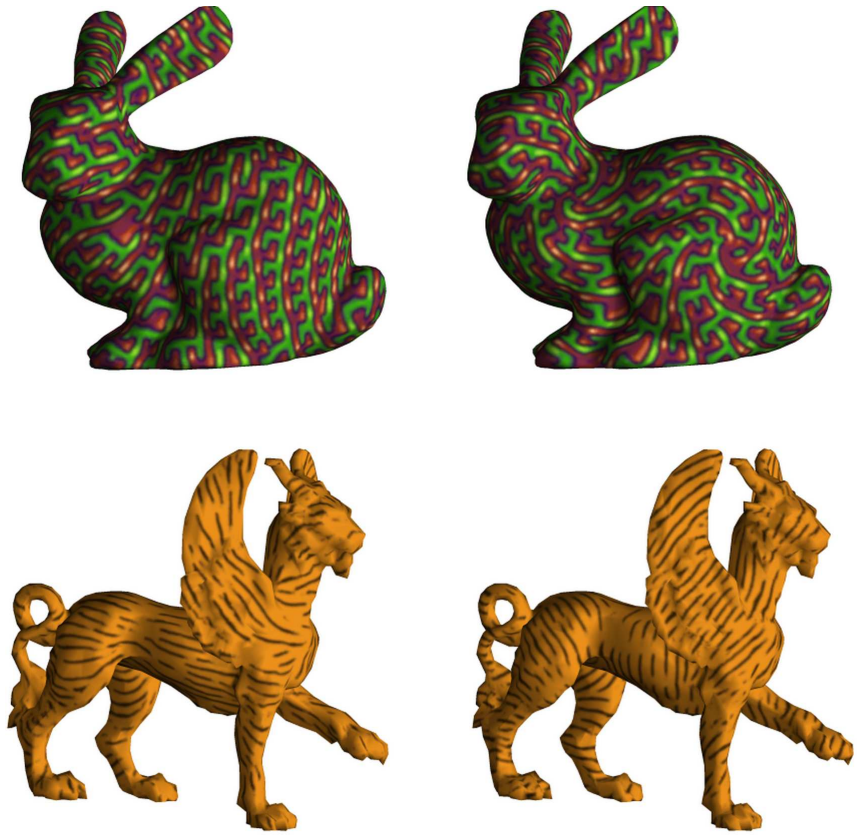

Fig. 22. This figure shows the results of applying our surface vector field design system to texture synthesis. Two vector fields are used for each model: the bunny, and the feline. Notice that singularities lead to the breakup of the synthesis patterns (upper-right). Also, the spirals around these singularities are obvious in the synthesis result. For anisotropic textures, different vector fields lead to different visual appearances (compare the bunny images and the feline images, respectively).

problem of hatch-based illustration as performing texture synthesis on surfaces, which leads to a real-time hatching system in which the user has the option to guide the orientation of hatches with a vector field on a 3D model. Van Wijk [2003] applies his image based flow visualization technique to curvature fields to produce non-photorealistic illustrations of 3D surfaces. Similar to the image gradient field, the principle curvature fields are rather noisy for regions where the principle directions are not prominent.

In this work, we allow the user to guide the hatch field through vector field design. Figure 21 shows the results of applying this technique to various 3D models. The vector field used for the dragon model in the lower-right image was obtained by rotating the vector field from the lower-left image by $\frac{\pi}{3}$.

7.3 *Example-Based Texture Synthesis*

Example-based texture synthesis refers to creating patterns on surfaces based on a given input image of a texture. Praun et al. [2000] propose “lapped textures” in which the surface is partitioned into overlapping regions and each region receives a portion of the input image. This method is fast, but causes seams due to surface partition. For textures that contain only high frequencies, the seams are relative unnoticeable. Another class of methods [Turk 2001; Wei and Levoy 2001] perform synthesis on surfaces directly without creating seams. For any point on the surface,

its color is copied from the pixel in the same image that provides the best neighborhood match based on a distance criterion. For both types of synthesis methods, a vector field is used to provide local orientation and scale as well as to determine the synthesis order.

Figure 22 shows the results of applying our vector field design system to texture synthesis on the bunny and the feline. The texture synthesis method is based on [Turk 2001; Wei and Levoy 2001]. The two vector fields used for the bunny are: a sink element at the tail and a source element on the forehead (upper-left), and a $\frac{\pi}{3}$ rotation of a dipole (a source element and a sink element) on the visible side of bunny (upper-right). Notice that the spiraling in the second vector field near the singularities on the side of the bunny is evident in the texture in the upper-right image. Figure 1 shows the visualization of these vector fields (middle and right). The bottom row of Figure 22 shows the feline with a tiger stripe pattern that is guided by two different vector fields. Both vector fields lead to reasonable results.

8. CONCLUSION AND FUTURE WORK

Vector field design on surfaces is an important problem that has received relatively little attention. We have identified a number of graphics applications, such as non-photorealistic rendering and texture synthesis, for which a vector field design system is needed. We also propose a set of requirements for a vector field design system. Namely, the user can create a wide variety of vector fields (not some sub-class) with relative little effort. Also, the user has control over the number and location of the singularities.

We present a vector field design system for both planar domain and 3D mesh surfaces. To our knowledge, this is the first system that produces continuous vector fields on mesh surface and provides control over vector field topology. The system has a three-stage pipeline: creating an initial vector field, analysis, and editing. The editing operations are at the core of our system. To make the system fully functional, we have introduced algorithms to resolve several problems. Many of these problems are challenging by themselves.

(1) The piecewise interpolation scheme for vector fields on mesh surfaces is novel, and it enables efficient vector field analysis and editing on meshes. To our knowledge, existing singularity pair cancellation techniques work only for planar domains.

(2) We describe a new technique to construct surface basis vector fields based on the concept of *geodesic polar maps* and *parallel transport*.

(3) We allow the user to control the location of singularities by a novel singularity movement operation. Furthermore, we provide a unified framework for implementing both singularity pair cancellation and singularity movement based on Conley index theory, which is more general and powerful than the well-known Poincaré index. To our knowledge, this is the first time Conley index theory is applied to Computer Graphics. Furthermore, the region optimization technique that we describe helps to produce smooth vector fields after editing operations.

(4) We use flow rotations and reflections to overcome numerical instabilities associated with regions of high curl and regions near saddles, which allows control over any linear singularities.

There are a number of issues that we wish to improve upon our current system. First, our system uses the same decay constant d for all design elements for creating basis vector field (Equation 3 and 4). It may be desirable to let the user control this. Second, our algorithm for building isolating blocks sometimes produces regions that are larger than necessary. This means that the behavior of the flow may be changed at places that are far away from the user-specified singularities. We plan to investigate ways of restricting the size of such regions. Third, our system requires the user to specify the pair of singularities for cancellation. It would be nice to provide the functionality “automatic singularity pairing for elimination”, in which the user specifies one singularity to be removed and allow the system to determine another singularity for pair cancellation. Finally, our surface vector field design system currently only handles closed surfaces. Surfaces with holes may cause incorrect results when computing geodesic polar maps, and we use hole-filling techniques to remove these holes. It is desirable to consider other approaches in which surfaces with holes can be handled directly without the filling.

There are several possible areas of future research. So far we have focused on controlling singularities in a vector field. It is natural to ask for controls over separatrices and periodic orbits, which are also part of vector field topology. Can we extend the concept of singular elements and allow the user to create canonical separatrices and periodic orbits? What editing operations are necessary to edit them? Finally, and maybe more fundamentally, what graphics applications will benefit from these operations?

Singularity pair cancellation can be seen as performing a particular type of *bifurcation* if one tracks the continuous change that is involved. Many other types of bifurcations exist. They are interesting mathematically, and they also have applications for scientific visualization.

Vector field design for surface might be extended to handle vector fields defined for volumes or other higher-dimensional datasets. Also, we are interested in identifying other applications for vector field design, such as fluid simulation and hairstyle design. Fluid simulation will require a vector field design to be able to create wavy functions as well as time-dependent flows.

Another important application is for educational purpose in which students learn important concepts of vector fields through creating and manipulating vector fields and observing the changes.

ACKNOWLEDGMENTS

We would like to thank the following people and groups for the 3D models they provided: Mark Levoy and the Stanford Graphics Group, Andrzej Szymczak, and Cyberware. We also appreciate the discussions with Jarek Rossignac, Andrzej Szymczak, John C. Hart, and Todd Moeller. Thanks to James Hays and Irfan Essa for their high-quality painterly rendering program that produces the painterly images shown in this paper. Furthermore, we are grateful to the help by Spencer Reynolds in preparing the video. Finally, we wish to thank our anonymous reviewers for their valuable comments and suggestions.

This work is funded by NSF grants DMS-0138420 and DMS-0107396.

REFERENCES

ALLIEZ, P., COHEN-STEINER, D., DEVILLERS, O., LÉVY, B., AND DESBRUN, M. 2003. Anisotropic polygonal remeshing. *ACM Transactions on Graphics (SIGGRAPH 2003) 22*, 3 (July), 485–493.

CALCATERRA, C. AND BOLDT, A. 2003. Flow-box theorem for lipschitz continuous vector fields. http://www.citebase.org/cgi-bin/citations?id=oai:arXiv.org:math/0305207.

CASH, J. R. AND KARP, A. H. 1990. A variable order Runge-Kutta method for initial value problems with rapidly varying right-hand sides. *ACM Transactions on Mathematical Software 16*, 201–222.

CONLEY, C. 1978. *Isolated Invariant Sets and the Morse Index*. AMS, Providence, RI. CBMS **38**.

EDELSBRUNNER, H., HARER, J., AND ZOMORODIAN, A. 2003. Hierarchical Morse-Smale complexes for piecewise linear 2-manifolds. *Discrete Comput. Geom. 30*, 87–107.

EDELSBRUNNER, H., LETSCHER, D., AND ZOMORODIAN, A. 2002. Topological persistence and simplification. *Discrete Comput. Geom. 28*, 511–533.

FLOATER, M. S. 2003. Mean value coordinates. *CAGD 20*, 19–27.

GIRSHICK, A., INTERRANTE, V., HAKER, S., AND LEMOINE, T. 2000. Line direction matters: an argument for the use of principal directions in 3d line drawings. *Proceedings of NPAR*, 43–52.

HALE, J. AND KOCAK, H. 1991. *Dynamics and Bifurcations*. Springer-Verlag, New York. Texts in Applied Mathematics **3**.

HAUSER, H., LARAMEE, R. S., AND DOLEISCH, H. 2002. State-of-the-art report 2002 in flow visualization. Tech. Rep. TR-VRVis-2002-003, VRVis Research Center, Vienna, Austria. Jan.

HAYS, J. H. AND ESSA, I. 2004. Image and video based painterly animation. *NPAR 2004: Third International Symposium on Non-Photorealistic Animation and Rendering*, 113–120.

HELMAN, J. L. AND HESSELINK, L. 1991. Visualizing vector field topology in fluid flows. *IEEE Computer Graphics and Applications 11*, 3 (May), 36–46.

HERTZMANN, A. 1998. Painterly rendering with curved brush strokes of multiple sizes. *Computer Graphics Proceedings, Annual Conference Series (SIGGRAPH 1998)*, 453–460.

HERTZMANN, A. AND ZORIN, D. 2000. Illustrating smooth surfaces. *Computer Graphics Proceedings, Annual Conference Series (SIGGRAPH 2000)*, 517–526.

HIRSCH, M. AND SMALE, S. 1974. *Differential Equations, Dynamical Systems, and Linear Algebra*. Academic Press.

KACZYNSKI, T., MISCHAIKOW, K., AND MROZEK, M. 2004. *Computational Homology*. Springer. Applied Mathematical Sciences **157**.

KIMMEL, R. AND SETHIAN, J. A. 1998. Computing geodesic paths on manifolds. *Proceedings of National Academy of Sciences 95*, 15 (July), 8431–8435.

MISCHAIKOW, K. 2002. Topological techniques for efficient rigorous computation in dynamics. *Acta Numerica 2002*, 435–478. Cambridge University Press.

MISCHAIKOW, K. AND MROZEK, M. 2002. Conley index. *Handbook of Dynamic Systems, North-Holland 2*, 393–460.

NI, X., GARLAND, M., AND HART, J. C. 2004. Fair morse functions for extracting the topological structure of a surface mesh. *ACM Transactions on Graphics (SIGGRAPH 2004) 23*, 3 (Aug.), 613–622.

POLTHIER, K. AND PREUSS, E. 2003. Identifying vector fields singularities using a discrete hodge decomposition. *Mathematical Visualization III, Ed: H.C. Hege, K. Polthier*, 112–134.

POLTHIER, K. AND SCHMIES, M. 1998. Straightest geodesics on polyhedral surfaces. *Mathematical Visualization, Ed: H.C. Hege, K. Polthier*, 135–150.

PRAUN, E., FINKELSTEIN, A., AND HOPPE, H. 2000. Lapped textures. *Computer Graphics Proceedings, Annual Conference Series (SIGGRAPH 2000)*, 465–470.

PRAUN, E., HOPPE, H., WEBB, M., AND FINKELSTEIN, A. 2001. Real-time hatching. *Computer Graphics Proceedings, Annual Conference Series (SIGGRAPH 2001)*, 581–586.

ROCKWOOD, A. AND BUNDERWALA, S. 2001. A toy vector field based on geometric algebra. *Proceeding Application of Geometric Algebra in Computer Science and Engineering, (AGACSE2001)*, 179–185.

SCHEUERMANN, G., KRGER, H., MENZEL, M., AND ROCKWOOD, A. P. 1998. Visualizing nonlinear vector field topology. *IEEE Transactions on Visualization and ComputerGrapics 4*, 2, 109–116.

STAM, J. 2003. Flows on surfaces of arbitrary topology. *ACM Transactions on Graphics (SIGGRAPH 2003) 22*, 3 (July), 724–731.

THEISEL, H. 2002. Designing 2d vector fields of arbitrary topology. *Computer Graphics Forum (Proceedings Eurographics 2002) 21*, 3 (July), 595–604.

THEISEL, H. AND WEINKAUF, T. 2002. Vector field metrics based on distance measures of first order critical points. *V. Skala (editor): Journal of WSCG 10*, 121–128.

TONG, Y., LOMBEYDA, S., HIRANI, A., AND DESBRUN, M. 2003. Discrete multiscale vector field decomposition. *ACM Transactions on Graphics (SIGGRAPH 2003) 22*, 3 (July), 445–452.

TRICOCHE, X. 2002. Vector and tensor field topology simplification, tracking, and visualization. Ph.D. thesis, Universität Kaiserslautern.

TRICOCHE, X., SCHEUERMANN, G., AND HAGEN, H. 2001. Continous topology simplification of planar vector fields. *Proceeding IEEE Visualization, IEEE Computer Society Press*, 159–166.

TURK, G. 2001. Texture synthesis on surfaces. *Computer Graphics Proceedings, Annual Conference Series (SIGGRAPH 2001)*, 347–354.

VAN WIJK, J. J. 2002. Image based flow visualization. *ACM Transactions on Graphics (SIGGRAPH 2002) 21*, 3 (July), 745–754.

VAN WIJK, J. J. 2003. Image based flow visualization for curved surfaces. *In: G. Turk, J. van Wijk, R. Moorhead (eds.), Proceedings IEEE Visualization*, 123–130.

WEI, L. Y. AND LEVOY, M. 2001. Texture synthesis over arbitrary manifold surfaces. *Computer Graphics Proceedings, Annual Conference Series (SIGGRAPH 2001)*, 355–360.

WEJCHERT, J. AND HAUMANN, D. 1991. Animation aerodynamics. *Computer Graphics Proceedings, Annual Conference Series (SIGGRAPH 1991)*, 19–22.

WELCH, W. AND WITKIN, A. 1994. Free-form shape design using triangulated surfaces. *Computer Graphics Proceedings, Annual Conference Series (SIGGRAPH 1994)*, 247–256.

WESTERMANN, R., JOHNSON, C., AND ERTL, T. 2000. A level-set method for flow visualization. *Proceeding IEEE Visualization*, 147–154.

WISCHGOLL, T. AND SCHEUERMANN, G. 2001. Detection and visualization of planar closed streamline. *IEEE Transactions on Visualization and ComputerGrapics 7*, 2, 165–172.