Animation of Orthogonal Texture Patterns for Vector Field Visualization

Sven Bachthaler and Daniel Weiskopf, *Member*, *IEEE Computer Society*

Abstract—This paper introduces orthogonal vector field visualization on 2D manifolds: a representation by lines that are perpendicular to the input vector field. Line patterns are generated by line integral convolution (LIC). This visualization is combined with animation based on motion along the vector field. This decoupling of the line direction from the direction of animation allows us to choose the spatial frequencies along the direction of motion independently from the length scales along the LIC line patterns. Vision research indicates that local motion detectors are tuned to certain spatial frequencies of textures, and the above decoupling enables us to generate spatial frequencies optimized for motion perception. Furthermore, we introduce a combined visualization that employs orthogonal LIC patterns together with conventional, tangential streamline LIC patterns in order to benefit from the advantages of these two visualization approaches; the combination of orthogonal and tangential LIC is achieved by two novel image-space compositing schemes. In addition, a filtering process is described to achieve a consistent and temporally coherent animation of orthogonal vector field visualization. Different filter kernels and filter methods are compared and discussed in terms of visualization quality and speed. We present respective visualization algorithms for 2D planar vector fields and tangential vector fields on curved surfaces and demonstrate that those algorithms lend themselves to efficient and interactive GPU implementations.

Index Terms—Scientific visualization, time-dependent vector fields, flow visualization, texture advection, line integral convolution, texture synthesis, GPU programming.

1 Introduction

Vector field visualization—a classic topic within scientific visualization—addresses the display of the direction and magnitude of vectors. Direction can be visually encoded in linelike patterns that follow the vector field; a typical example is a collection of streamlines, which are the integral curves of a time-independent vector field. Texture-based methods such as line integral convolution (LIC) or texture advection achieve a dense coverage by those line patterns: integral curves are essentially drawn "everywhere" on the domain, avoiding the issue of identifying appropriate seed points for particle tracing. Dense texture-based representations are well understood for 2D vector fields and for tangential vector fields on curved surfaces.

This paper builds upon previous texture-based methods for 2D manifolds and addresses a specific issue that has been neglected so far: how well can a human observer perceive animated visualization? In general, animation has been used successfully for vector field visualization because it can show the direction, orientation, and magnitude of the vector field. In addition, animation can mitigate the curve intersection issues that occur for long pathlines or streaklines of a time-dependent flow. However, we are not aware of any prior work by others that would have considered the perception of such animations. Based on results from vision research, we would like to make the case that previous approaches like animated streamlines are nonoptimal for local motion perception. Texture-based methods significantly reduce spatial frequency along integral curves to display those curves. However, there is substantial evidence for a spatial frequency tuning of the motion detectors in our human visual system (HVS), and optimal spatial frequencies are typically much higher than the spatial frequencies produced by texture-based methods (see Section 4.2).

Therefore, we propose to decouple the direction of the linelike patterns from the direction of animation. More specifically, we propose to use an orthogonal vector field (i.e., the original vector field rotated by $\pi/2$) to construct linelike patterns and use the original vector field to drive the animation. In this way, the spatial frequencies along the direction of motion are determined by the spatial frequencies of the input noise (for LIC or texture advection), which are independent of the length scales along the line patterns (controlled by the filter length of LIC or texture advection). This visualization approach resembles a moving wave front of the vector field and therefore provides an intuitive analogy to the real world.

We denote the method of the orthogonal vector field and its corresponding visualization as *ortho-vis*. In contrast, the traditional texture-based visualization of streamlines or pathlines is called *tangent-vis* to indicate that it shows the tangential direction of the vector field. Ortho-vis and tangent-vis can be combined to a single visualization image that retains the perceptual benefits of ortho-vis while providing the traditional, familiar visualization via tangent-vis.

• *The authors are with VISUS—(Visualization Research Center), Universität Stuttgart, Nobelstrasse 15, 70569 Stuttgart, Germany. E-mail: {bachthaler, weiskopf}@visus.uni-stuttgart.de.*

Manuscript received 29 Aug. 2007; revised 9 Nov. 2007; accepted 15 Jan. 2008; published online 12 Feb. 2008.
Recommended for acceptance by T. Moeller.
For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org, and reference IEEECS Log Number TVCGSI-2007-08-0119.
Digital Object Identifier no. 10.1109/TVCG.2008.36.

The main contributions of this paper are summarized as follows:

1. The perceptual issues of animated flow visualization, which have not been addressed before, are pointed out and substantiated by references from the vision research literature.
2. The concept of orthogonal vector field representation is introduced for visualization purposes.
3. A filtering process is proposed to obtain a consistent and temporally coherent animation of the orthogonal vector field visualization. We discuss the mathematical reason for possible incoherence and describe and compare two variants of the filter process to overcome this incoherence: a direct computation of convolution filtering that allows for flexible filter design and an incremental and fast filter process by recurring alpha blending.
4. We propose two image-space compositing models that allow us to combine ortho-vis with traditional tangent-vis.
5. An efficient texture-based algorithm for 2D planar vector fields is described.
6. This algorithm is extended to tangential vector fields on curved surfaces.
7. Fast GPU implementations of both algorithms are presented.

This paper is an extended version of our previous work [2], and it provides the following extensions and modifications. First, the orthogonal lines are combined with the additional visualization of streamlines along the vector field. Second, we introduce and discuss image-space compositing methods that allow for the smooth combination of ortho-vis and tangent-vis. In particular, their appropriateness for embossed illuminated representations on surfaces is demonstrated. Third, the issue of possibly incoherent transport of the orthogonal vector field representation is detailed in a mathematical discussion that leads to a quantitative description of the level of incoherence that needs to be removed by temporal filtering. Fourth, this temporal filtering process of the ortho-vis patterns is extended to generic filter kernels, evaluating and comparing the filter quality for several relevant filter examples. Fifth, sparse noise and anisotropic filtering according to oriented LIC (OLIC) [44] are employed to show the downstream direction of the tangential vector field.

2 Previous Work

This paper describes a technique for texture-based vector field visualization. We refer to the survey chapter [48] and the book [46] for an overview of vector field visualization in general and to the article [23] for a presentation of the state of the art in texture-based methods.

Our visualization approach relies on LIC [5] as a role model to extract and display linelike structures. For the visualization of 2D vector fields, we adopt a GPU version of LIC [49]. For data sets given on curved surfaces embedded in 3D space, we extend image-space advection [24], [41] and combine it with a related hybrid image/object space method for LIC [50]. Since we generate texture patterns that are orthogonal (for the original vector field direction and the perpendicular direction), the visual signature of our final visualization images resembles the texture-based visualization of symmetric real 2D tensor fields by Hotz et al. [18], who overlay two LIC-type images, showing the two orthogonal eigendirections of the tensor field. In contrast to the work by Hotz et al., our problem setting with temporal filtering gives rise to potentially nonideal perpendicular line structures. Furthermore, we propose extended compositing models, whereas Hotz et al. do not focus on image-compositing algorithms. An alternative texture-based approach to tensor field visualization by Zhen and Pang [51] uses multipass LIC in the different eigendirections, leading to a single family of smeared-out lines, which has a different visual signature than the overlay of two independent families of lines obtained by our technique.

Animation plays an important role in our visualization approach, in addition to the spatial structures generated by LIC. Previous work on animated texture-based vector field visualization focuses on motion along linelike structures to show the direction, orientation, and velocity of the vector field, i.e., visual patterns and animation reveal essentially the same information. For time-independent vector fields, patterns typically move along streamlines [5], [25]. Most methods for time-dependent data directly support animated visualization such as LIC methods [10], texture advection techniques [20], [40], or unsteady flow LIC (UFLIC) [34] and its recent variants [26], [28]. A decoupling of spatial structures and temporal behavior through animation is described in a generic texture-based framework [49], which generalizes dynamic LIC [35], designed for animated electric or magnetic fields. In this paper, the framework [49] is adopted for specifically designed choices for temporal coherence and spatial patterns. In addition, we extend it to the visualization on curved surfaces.

Our visualization method targets an easy-to-perceive animation. In general, visual perception is of high interest for scientific visualization and information visualization alike [42]. In previous work, however, motion perception plays a less important role than the perception of static patterns. Only few prior papers specifically address motion perception for visualization purposes; examples include the kinetic visualization of shape [29], preattentive processing [16], filtering and brushing with motion [3], multivariate visualization [27], perceptual limits on 2D motion-field visualization [22], and the influence of color on motion perception [45]. In the context of general computer graphics, spatiotemporal contrast sensitivity functions of the HVS can be utilized for motion-aware metrics that are the basis for accelerated rendering [30], [31].

The technique of this paper utilizes the overlay of two different LIC textures to combine the visualization of the tangential and orthogonal vector fields. We avoid a direct color blending for compositing; instead, we apply a weaving of high-frequency spatial textures of different colors. Texture weaving has been shown to be more effective than color blending [15]—especially in the context of the multivariate visualization of a single vector field and an additional scalar field [39]. The approach [39] is also used in the paint-inspired color mixing in RYB color space for multivariate visualization [14]. In contrast to these previous papers, we visualize two families of texture line directions. Therefore, our compositing models are designed for the overlay of visual threads, which is inspired by perception-guided 3D vector field visualization according

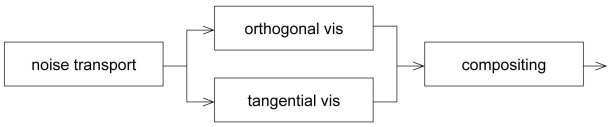


Fig. 1. High-level illustration of the structure of the complete visualization process.

to Interrante and Grosch [19]. Similar to Interrante and Grosch, we utilize the visual continuity of lines by appropriately chosen colors and by partial occlusion. Our approach, however, builds a 3D visual effect only from 2D LIC images by using specifically designed compositing schemes. Finally, our implementation additionally allows for emphasizing the pseudo-3D visual impression by bump mapping, which mimics the embossing of flow structures [38].

3 Algorithm Overview

Our method targets vector field visualization by combining texture-based representations of the tangential and orthogonal vector field directions. Our algorithm computes intermediate results of ortho-vis and tangent-vis elements in an essentially independent way, combining those results in a final image-compositing step. Fig. 1 provides a high-level view on the data flow and computational components of the complete algorithm. Temporally coherent animation starts with generating coherently moving noise images in the first part of the algorithm. Then, the process is split in the independent computation of ortho-vis and tangent-vis. Finally, image compositing results in a combined visualization of ortho-vis and tangent-vis.

On this abstract level, the above algorithm works for 2D data sets (i.e., on planar 2D manifolds) and 2.5D data sets (i.e., on curved surfaces embedded in 3D space) alike. Since we use an image-space approach for 2.5D visualization, image-space-oriented operations, as in Fig. 1, are sufficient to handle 2.5D vector fields.

The mathematical, algorithmic, and technical description in the following sections is structured along the data flow in Fig. 1. First, we focus on the ortho-vis part of the pipeline, starting with a formal and generic presentation in Section 4, followed by specific algorithms for 2D and 2.5D vector fields in Sections 5 and 6, respectively. The tangent-vis part is described in Section 7, which also discusses the compositing schemes for combining ortho-vis and tangent-vis.

4 Orthogonal Vector Field Representation

We first introduce the orthogonal vector field representation (ortho-vis) in a formal, mathematical way and then motivate our new visualization method by showing similarities to existing visualization approaches and by providing a perceptual rationale. The subsequent parts of this section discuss the animation and temporal coherence of moving visualization patterns.

4.1 Spatial Patterns

We assume a tangential vector field v defined on a smooth and orientable 2D manifold M (with or without boundary):

$$v : M \to TM \quad \text{with } v(x) \in T_xM.$$

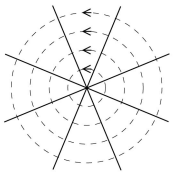


Fig. 2. Illustration of two perpendicular families of lines: for a circular vector field v (dashed lines) and a radial vector field u (solid lines). © Eurographics Association 2007; reproduced by kind permission of the Eurographics Association.

The vector field maps points $x \in M$ to a vector in the corresponding tangent space at that point, T_xM. Typically, M is either a flat 2D manifold or a surface embedded in 3D euclidean space. We refer to the first alternative as a 2D vector field and to the latter alternative as a 2.5D vector field. Since M is orientable, we can define an operator Ω that rotates a vector within the tangent plane by $\pi/2$. The inverse operator Ω^{-1} yields a rotation by $-\pi/2$. The orthogonal vector field u is defined as

$$u : M \to TM, \quad x \longmapsto \Omega v(x).$$

Our idea is to display the orthogonal vector field u instead of the original vector field v. The actual visualization relies on integral curves (streamlines in the context of flow visualization, field lines in the context of electric, magnetic, or related fields) to show the direction of u. In this paper, we focus on the texture-based visualization of streamlines by means of LIC (see Sections 5 and 6), but other methods such as geometrically constructed streamlines might also be employed. Fig. 2 illustrates an example of u and v by a means of a few geometric lines that represent integral curves. So far, a time-independent vector field has been assumed. For a time-dependent vector field, we apply the above approach to an instantaneous vector field for a given time in order to produce a single visualization for that time.

Before we consider animation—the main aspect of our visualization approach—we would like to motivate the use of the rotated vector field for a single frame of the visualization. First, the mapping by the rotation operator Ω is one to one, i.e., the original vector field v can be recovered by applying the uniquely defined inverse operator Ω^{-1}. While this argument shows that the same information content is displayed by u and v, the question remains how effective the rotated vector field is for visualization purposes. Here, the special choice of the $\pi/2$ rotation angle becomes important because analogous uses of perpendicular line structures are well known and accepted in visualization. One analogy comes from the representation of 2D scalar fields by either contour lines (isolines) or gradient directions: gradients and contours are perpendicular by construction. For example, Fig. 2, which was used to illustrate an orthogonal vector field, can also be interpreted as a visualization of a scalar field with maximum value in its center, concentric circles as contour lines (dashed), and radial gradient lines (solid).

A related analogy is based on the Helmholtz decomposition of vector fields [17]. Adopting the notation of Polthier and Preuß [32], a vector field v on a 2D manifold can be written as

$$v = \nabla\phi + \Omega\nabla\omega + \eta,$$

with the curl-free part $\nabla\phi$, the divergence-free part $\Omega\nabla\omega$, and the remaining harmonic part η. The scalar functions ϕ and ω serve as potentials for the gradient field $\nabla\phi$ and the cogradient field $\Omega\nabla\omega$ (defined as the gradient rotated by $\pi/2$). Recent publications on the Hodge-Helmholtz decomposition have focused on variational discretized computations for triangulated grids [32], [33], [37]. Assuming a divergence-free vector field represented by ω, our orthogonal vector field approach shows the field lines of the gradient $\nabla\omega$. Similarly, a curl-free vector field based on ϕ would show the gradient $\nabla\phi$ by traditional visualization methods. Therefore, the orthogonal vector field visualization could be regarded as the "dual" of the traditional flow visualization.

Another analogy can be found in the propagation of wave fronts. The Eikonal equation [12], which for example applies to the propagation of action in the Hamilton-Jacobi model of classical mechanics or to light propagation, describes this propagation in terms of fronts that are orthogonal to the propagation direction, i.e., those fronts are analogs of ortho-vis, whereas the propagation vectors are analogs of tangent-vis. A related physical analogy is the propagation of water waves, where the wave fronts are perpendicular to wave propagation.

4.2 Animation and Motion Perception

So far, we have only discussed static visualization by a single image. Although the spatial patterns in one frame are important, animation plays an even more crucial role in our approach. The basic idea is to drive the animation by the original vector field v, i.e., the direction of motion (given by v), and the integral curves in an image (determined by u) are perpendicular. Fig. 2 illustrates this approach: the solid radial lines show the curves of u, which are transported along the circular flow v, leading to a counterclockwise rotation.

Why is the decoupling of temporal evolution and spatial patterns useful? The main motivation comes from research on human visual perception. There is indication for a spatial frequency tuning of the HVS: how well we perceive motion depends on the spatial frequency of moving patterns. Low-level motion perception is based on small receptive fields that serve as local motion detectors (see, for example, [1]). Vision research and physiological investigations have addressed various aspects of motion perception, including the detection and discrimination of moving patterns, the influence of contrast and color, and the breakdown of the perception of coherent motion under certain conditions. Although this topic is still an area of active research, a general observation of a frequency tuning of motion detection can be found in the literature. In the following, we focus on a few, recent papers and refer the reader to references therein for further reading.

One interesting aspect of recent studies is that the characteristics of receptive fields may be adaptive—dependent on the stimulus. For example, Cavanaugh et al. [6] describe that at low contrast, a wider spatial region (with less surround suppression) is used as input to increase sensitivity, whereas a high-contrast stimulus leads to higher spatial resolution using increased surround suppression. This principal observation can also be found in the context of motion perception [36], where high contrast favors the detection of high-frequency stimuli, and low contrast favors large stimuli. Tadin and Lappin [36] report an optimal size of

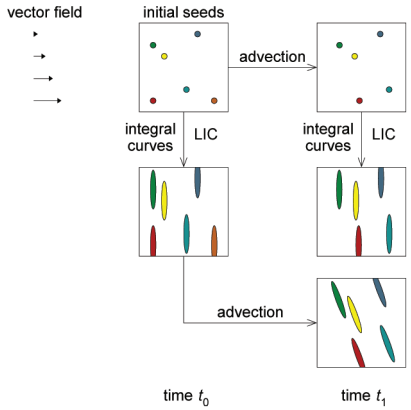


Fig. 3. Illustration of two transport and visualization approaches, applied to a shear flow. The two result images (middle and bottom images in the right column) differ because LIC and advection are not commutative. Seeds and streamlines are colored to allow for an easier recognition of correspondence between images. © Eurographics Association 2007; reproduced by kind permission of the Eurographics Association.

0.5 deg (degrees with respect to the subtended angle as seen by the viewer) for a high contrast of 92 percent. Typically, texture-based vector field visualization uses patterns of high luminance contrast, and therefore, high spatial frequency patterns are appropriate. Another observation is that local and global motion detectors can be distinguished (see, e.g., [4]). In this paper, we focus on local motion detection, which is optimal for certain spatial frequencies. Bex and Dakin [4] report a maximum sensitivity for local motion detection for spatial frequencies around 2 cycles/deg. A similar number of 3 cycles/deg is given by Watson and Turano [43] as optimal motion stimulus.

The actual value for the optimal spatial frequency of patterns depends on several outside parameters, but many studies agree upon frequencies somewhere around or above 2 cycles/deg. For typical visualization applications with extended streamlines of 100-200 pixels length, the spatial frequency along those lines is approximately 5-10 times greater than this optimum (depending on viewing distance and screen resolution).

4.3 Temporal Coherence

We intend to transport integral curves of the rotated vector field u along the original vector field v in order to control the spatial frequency of the transported patterns along the transport direction. Please note that the vector fields may be time dependent. This transport could be realized by first constructing integral curves of u for an initial time t_0 and then advecting those curves along v to a later time t_1. An alternative way is to first advect the seed points (i.e., initial noise for LIC) along v from time t_0 to t_1 and then construct the integral curves of u for time t_1. Fig. 3 illustrates both approaches for the example of a shear flow. Unfortunately, these two transport approaches do not necessarily lead to the same result, as illustrated in Fig. 3. The first approach guarantees temporal coherence of the transported integral curves because the curves themselves are advected. The second approach makes sure that the integral curves are always perpendicular to v. Since the two approaches may lead to different results, we are unable to have a mechanism

that maintains orthogonal vector field lines and achieves temporal coherence at the same time.

This conflict between orthogonality and temporal coherence is detailed for steady flow in the mathematical discussion of Appendix A. Equation (15) from Appendix A shows that the measure for the amount of inconsistency between orthogonality and temporal coherence is

$$\delta_{\mathrm{LIC}} = \frac{\partial v_x}{\partial y} + \frac{\partial v_y}{\partial x}. \tag{1}$$

The term $\frac{\partial v_y}{\partial x}$ describes the curvature of the vector field, the term $\frac{\partial v_x}{\partial y}$ describes the shear. Please note that a frame of reference is used in which the x-axis points along the vector field and the y-axis along the perpendicular vector field direction (see Appendix A for more details). Therefore, the measure of inconsistency is directly related to the curvature and shear of the vector field. In the example of pure shear, as illustrated in Fig. 3, δ_{LIC} is given by the nonvanishing shear value only. In contrast, the example of the circular flow (Fig. 2) does not exhibit any temporal incoherence because effects of curvature and shear cancel exactly. For unsteady flow, additional incoherence effects might be introduced.

To overcome the problem of temporal incoherence, we propose the following two-part process. The first part is a combination of advection and integral-curve construction: initial seed points are advected along v from the initial time t_0 to an intermediate time t_i $(t_0 \leq t_i \leq t_1)$; then integral curves of u are constructed at time t_i; finally, those integral curves are advected along v from t_i to t_1. We denote this overall operation as T_{t_i}. For a texture-based representation, T_{t_i} takes an initial noise image N as input and yields an image of transported integral curves. The second part applies a temporal filtering process in order to balance the conflicting goals of temporal coherence and orthogonal vector field lines. The actual visualization image at time t_1 is

$$I = \int_{t_0}^{t_1} k(t_i) T_{t_i}(N)\,\mathrm{d}t_i, \tag{2}$$

with a filter kernel $k(t)$ normalized according to $\int k(t)\,\mathrm{d}t = 1$. This filtering process allows us to trade clearly defined linelike patterns for a consistent and temporally coherent animation: the width of the filter interval $[t_0, t_1]$ determines the amount of "smearing out" and can be gradually adjusted. In fact, there are many important flow fields that exhibit no or only little inconsistencies and, thus, would not need any filtering. The circular flow in Fig. 2 is an example of a completely consistent advection and integral-curve construction.

Animated visualization produces images for increasing end times t_1. For a constant filter width $(t_1 - t_0)$, the start time progresses accordingly. To achieve temporal coherence of the final images, the noise images N need to be temporally coherent for different times t_0, which can be ensured by advecting initial noise images along the vector field v.

An alternative approach allows us to avoid the two-part process: by reducing the length of the orthogonal LIC lines in regions of strong inconsistencies, these inconsistencies are reduced. However, this approach has the disadvantage that static images do not show the "wave fronts" of the vector field to the same degree as in the filtered version.

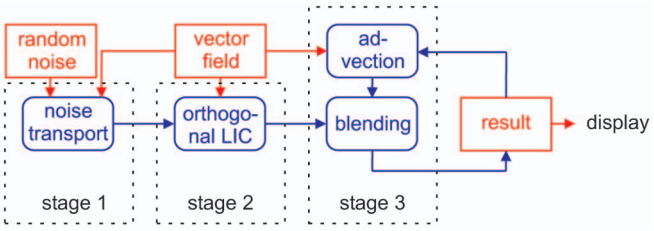


Fig. 4. Processing stages and data flow for the 2D algorithm. © Eurographics Association 2007; reproduced by kind permission of the Eurographics Association.

5 2D Algorithm

This section describes an algorithm that realizes the approach from the previous section for vector fields given on planar 2D domains. All relevant information is 2D (vector field, input noise images, and intermediate and final visualization images) and can be represented as 2D images, 2D textures, or 2D uniform grids. In the following, we refer to them as images or textures. Vector data on unstructured, triangulated grids would also work because a triangle mesh can be easily rendered (i.e., rasterized) into a 2D image.

The algorithm that produces the final output image can be seen as a pipeline consisting of three major stages (Fig. 4). Each of these stages creates an intermediate result that is used as input for the next stage. The first stage (noise transport) implements two aspects of the abstract approach from Section 4.3: 1) temporally coherent input noise for different starting times t_0 and 2) the advection of noise from t_0 to the intermediate time t_i. The second stage (orthogonal LIC) constructs an LIC image of the rotated vector field u at time t_i. The third stage (advection and blending) implements the transport of LIC patterns from time t_i to t_1 and computes the filter operation from (2). In the following, the three stages are explained in more detail.

The first stage is responsible for creating a temporally coherent noise that should move according to the possibly time-dependent vector field v. Similar to [49, Section 4], pathlines are traversed from the current time step backward in time in order to accumulate noise injection input from previous times in a Lagrangian manner. This accumulation yields a convolution in time along pathlines. The time span of backward particle tracing determines the scale of temporal correlation: typically, some 15-50 integration steps are appropriate for an adequate compromise between computation time and quality of temporal coherence.

The different noise injection images that serve as input for the temporal convolution need to be uncorrelated. To save memory for a large number of noise injection images, we construct them on the fly by reusing a single template image. We assume that the template noise image is periodic (i.e., a seamless texture), which, for example, is automatically achieved by generating a low-pass filtered noise via filtering in Fourier space using FFT. A new uncorrelated noise image is produced from the template image by Cranley-Patterson rotation [7], which adds the same random shift to each point of the template image. The random shifts are applied on the fly while the noise injection texture is accessed. The density of the visual

representation is controlled by the characteristics of noise injection. A dense representation is achieved by low-pass filtered white noise, a sparser representation is achieved by Gaussian-filtered sparse input constructed from randomly positioned dots.

Inflow and outflow at boundaries of the domain often cause problems for texture-based methods. This issue is addressed in two ways. First, the injection noise is periodic and thus virtually infinite in size. Second, the vector field is clamped at the boundary, making it virtually infinite as well. Therefore, particles can be traced beyond domain boundaries. Another issue is divergence or convergence of the flow, which could change the spatial frequency of injected noise by stretching or compression. Due to the limited integration length in time (some 15-50 integration steps), this problem does not lead to serious artifacts except for extremely large absolute values of divergence. Finally, the temporal convolution of uncorrelated noise images leads to reduced contrast. The convolution corresponds to a summation of (approximately) independent random variables, resulting in a normal distribution of values according to the central limit theorem. Contrast is restored by histogram equalization.

The result of the first stage is a noise texture that moves along the vector field v and serves as input to the second stage. The second stage creates LIC lines that visualize the orthogonal vector field u at a fixed time that corresponds to the current visualization time, which is similar to the spatial filtering process in [49]. The rotation of the original vector field $v = (v_x, v_y)$ is computed by a mapping to $u = (-v_y, v_x)$. Usual particle tracing and LIC integration are performed with the orthogonal vector field. Vectors are normalized to unit length to obtain LIC lines of equal length. Boundaries of the domain are taken into account by stopping the LIC integration once a particle trace crosses a boundary. Similar to stage one, contrast is enhanced by histogram equalization.

The third stage of the pipeline transports LIC patterns from the second stage and evaluates the filter operation from (2). The goal of the third stage is to produce a temporally coherent and consistent visualization with line patterns that are (approximately) perpendicular to the vector field. A generic implementation of the filtering (2) requires us to compute and store several intermediate images T_{t_i}. We have implemented this flexible and accurate approach mainly for evaluation and comparison purposes. The additional work and memory consumption for generic filtering can be avoided by restricting ourselves to an exponential filter kernel, which can be discretized in the form of a recurring application of the over operator (i.e., alpha blending with weights α and $(1-\alpha)$) [9]. The alpha value determines the falloff of the exponential filter. One image used for blending is the result of the second stage; the other image is the visualization result of the previous time step, transported to the current time step by semi-Lagrangian advection. This incremental computation of exponential filtering is mainly recommended for real-time rendering for interactive visualization. If memory consumption and additional computation time are not limiting factors, then the generic filtering process can be used in order to obtain images of higher quality. We compare the inexpensive incremental exponential filtering process with generic filter kernels in a parameter study in Section 9.3.

A different approach avoids the third stage completely. This approach adapts the length of orthogonal LIC lines in those regions where inconsistencies are present. To make this possible, the inconsistency is precomputed and used as a scaling factor at every point of the vector field to shorten the LIC lines proportional to the amount of inconsistency. The results of this approach are shown in Fig. 13c.

6 2.5D Algorithm

This section describes the visualization of tangential vector fields on curved surfaces embedded in 3D space. We adopt the same basic pipeline as for 2D vector fields (see Fig. 4) but need to include some modifications and extensions that are specific to 2.5D data. The following discussion is restricted to those modifications.

The input vector field may either be given as a 3D texture that is intersected by the surface, or it is attached to the vertices of the surface. Since our algorithm is designed for tangential vector fields, a possibly nontangential vector field is made tangential by subtracting the normal component of a vector.

The first stage of the pipeline (noise transport) adopts the hybrid object/image space LIC method on surfaces [50]. This LIC technique is turned into temporal convolution by the following modifications. First, particle paths are traced along pathlines backward in time only. Here, the vector field is not normalized to unit length. Second, a single input noise is replaced by uncorrelated noise inputs for different times, according to a Cranley-Patterson rotation. Noise is modeled as a 3D solid texture in order to achieve temporal coherence even under camera rotations, i.e., noise is attached to the surface geometry in object space. In addition, a MIPmapping approach is employed for antialiasing [50]. The result of the first stage is a temporally coherent noise image that moves along the surface of the scene geometry.

From now on, we work in image space only, reminiscent of image-space advection techniques [24], [41]. The second stage (orthogonal LIC) takes the original vector field given in 3D space, rotates it by $\pi/2$ around the local normal vector of the surface, and projects the orthogonal vector field onto image space. The rotation is determined in object space by computing the cross product of the surface normal and the tangential vector. The subsequent projection to image space yields a 2D vector field with respect to image-space coordinates. Finally, LIC is performed in image space, based on the noise image from stage one and the image-space vector field. Particle tracing for LIC is stopped at the boundaries (silhouette lines) of the object; the leaving of the object is determined according to a mask that contains the classification of pixels as foreground or background pixels. Because a complete LIC is evaluated, we are free to choose any filter kernel. In contrast, image-space advection techniques [24], [41] are restricted to an exponential kernel, which yields a lower image quality than the Gaussian kernel typically used in our implementation (see the discussion of filter quality in [47]).

The third stage (advection and blending) consists of the following components: projection of the original nonrotated vector field onto image space, semi-Lagrangian image-space advection of the visualization result from the previous time step, and blending of the advected image with the image

from stage two. The projection of the vector field is similar to the projection in stage two. Blending is a simple 2D image operation. Instead of the inexpensive blending operation that results in an exponential filtering, a generic filter kernel can be used here as well. To do this, we have to perform the same steps as for the temporal filtering in 2D. In particular, we have to store intermediate images of the second stage and perform a weighting and accumulation process using an arbitrary filter kernel. Similar to the 2D algorithm, the alternative approach of reducing LIC length in regions of inconsistency can be implemented by modifying stage two; in this case, stage three is not needed.

Semi-Lagrangian image-space advection can cause problems due to inflow at silhouette lines. To avoid inflow of background color, we employ a modified bilinear interpolation within the previous visualization image. This special filter works basically the same way as the standard bilinear filter—except for background texels, which are weighted zero. To decide whether a texel lies in the background or on the surface geometry, the same mask as in stage two is used. If all four texels lie on the background, a gray-scale value of 0.5 is assumed. Currently, internal edges are neglected, i.e., image information could be transported across such edges. The approach [24] could be included to overcome this issue.

For the final display, the texture from stage three is modulated by a rendered image of the surface geometry to simultaneously show the vector field texture and the surface shape. Our implementation supports the Blinn-Phong model and cool-warm shading [13] for surface illumination. Bump mapping is also available as an option to emphasize the structure of the vector field texture, mimicking the embossing of flow structures [38]. Here, the resulting texture from stage three is interpreted as a height field that perturbs the normal vectors.

7 Combined LIC and Image Compositing

Conventional LIC images that show lines along streamlines have the advantage that the direction of the flow is directly perceived in such an image. On the other hand, our proposed orthogonal LIC approach has the advantage that the spatial frequency of the moving pattern can be tuned to allow the optimal perception of the animated flow. In this section, we first describe the computation of conventional tangent-vis within our framework and then propose image-compositing methods to combine tangent-vis with ortho-vis. The motivation is to create an approach that benefits from the two different LIC methods and overcomes their respective drawbacks.

Let us begin with the modifications required to compute intermediate tangent-vis images. Since tangent-vis and ortho-vis mainly differ in the direction of their field lines, we restrict ourselves to the changes of the ortho-vis pipelines for 2D (Section 5) and 2.5D (Section 6). The most important difference is that the original vector field, not the rotated vector field, is used: the orthogonal LIC stage two in Fig. 4 is replaced by an LIC computation that works with the original vector field. Similarly, the 2.5D algorithm avoids the rotation of the vectors before projection to the image plane. Otherwise, the principal processing pipelines are identical for ortho-vis and tangent-vis, including stage one (transport of noise) and stage three (temporal filtering).

The next aspect concerns the actual combination of the orthogonal and tangential representations. These two representations are combined by compositing in image space using the LIC density (i.e., the gray-scale value that originates from the LIC computation). We denote the LIC density computed for the orthogonal and tangential LIC images as I_{ortho} and I_{tangent}, respectively. Often, those LIC images are immediately interpreted as gray-scale values for final display. However, we already consider the possibility of some additional color map applied to the LIC images before the final result is produced. The respective color-mapped images are denoted C_{ortho} and C_{tangent}.

Based on this terminology, we propose two different compositing models. The first one is called the *engraving* model and is formulated by the compositing equation:

$$C_{\text{comp}} = \begin{cases} C_{\text{ortho}} & \text{if } I_{\text{ortho}} > I_{\text{tangent}}, \\ C_{\text{tangent}} & \text{otherwise}, \end{cases} \tag{3}$$

which is applied to each pixel independently. The engraving approach essentially results in a maximum computation: the maximum of the two LIC densities fully determines the final color. For the special case of an identity color map, which implies a gray-scale image, (3) is reduced to the maximum operator $I_{\text{comp}} = \max(I_{\text{ortho}}, I_{\text{tangent}})$. This expression motivates the term "engraving": the LIC values can be interpreted as depth values of a height map; only the deepest engravings survive the compositing (i.e., the subsequent application) of intermediate engravings. Visually, the engraving effect is especially clear in combination with bump mapping, which is demonstrated in Section 9.2 and Fig. 9.

The engraving approach is easily computed, it has a simple interpretation, and it immediately works with height maps and bump mapping for embossing. However, one possible shortcoming is that engraving only keeps the LIC lines with large values, i.e., dark lines of one representation can never occlude bright lines of the other representation. To overcome this problem, we propose an alternative compositing scheme, called *interweaving*. The basic idea is that the color and brightness of LIC structures should be chosen independent of their relative depth, leading to interweaving patterns reminiscent of the interweaving of threads in fabrics. Independence is achieved by adding height-map textures H_{ortho} and H_{tangent}, respectively. The height maps are produced by the same pipeline as the original LIC density textures but are based on completely independent input noise. The compositing equation of interweaving is

$$C_{\text{comp}} = \begin{cases} C_{\text{ortho}} & \text{if } H_{\text{ortho}} > H_{\text{tangent}}, \\ C_{\text{tangent}} & \text{otherwise}. \end{cases} \tag{4}$$

According to the value stored in the height-map texture, either the tangential LIC image or the orthogonal LIC image is drawn. By doing this, the LIC images of both types are drawn on top of each other in an alternating fashion, thus creating the desired interweaving effect. The input noise for the height-map computation is always based on filtered

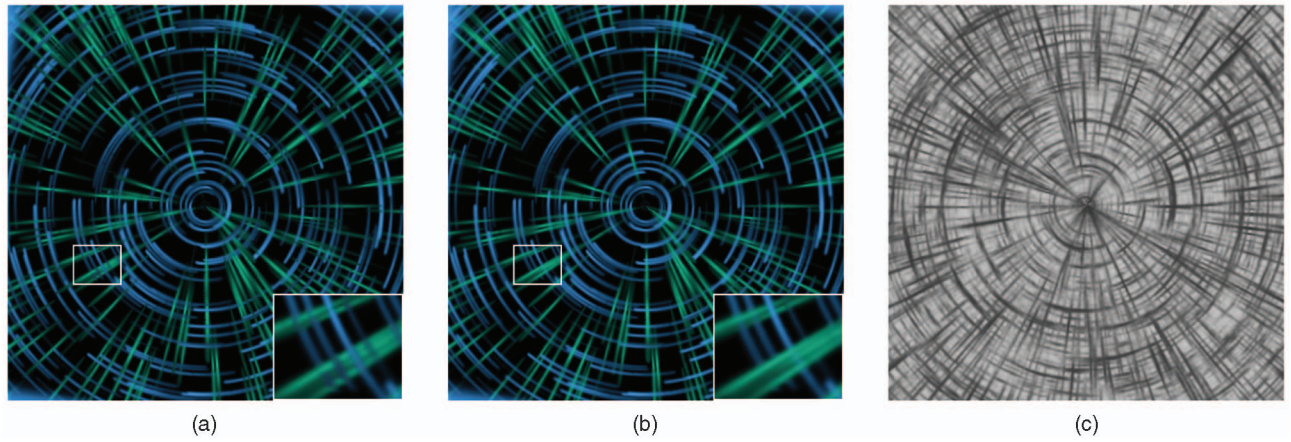


Fig. 5. A circular vector field visualized with combined LIC: (a) interweaved LIC, (b) engraved LIC, and (c) engraved LIC with dense noise.

white noise in order to densely cover the whole domain by height-field values, without any holes. The height-map texture provides a certain amount of control over the appearance of the final interweaving pattern: by modifying the scale of the white noise, we can control the width of the resulting (invisible) depth LIC lines and, therefore, how often one of the visible interweaving LIC lines "changes depth."

In contrast to the height-map textures H, the density LIC textures I may use different noise models. For example, we recommend moderately sparse noise together with asymmetric filtering in order to visualize downstream flow direction for tangent-vis according to the idea of OLIC [44]. For visual consistency, the ortho-vis image should also be based on a similar kind of sparse noise in this case. However, there is no natural downstream direction for the orthogonal vector field, and thus, a symmetric filter kernel like the Gaussian function should be applied for ortho-vis.

Different parameters and compositing models for combined LIC are compared in Fig. 5. An example of interweaving is depicted in Fig. 5a, where LIC lines of one type are running on top and below LIC lines of the other type in an alternating fashion. The data set is a circular flow. Fig. 5b applies the engraving model to the same visualization scenario. Zoomed-in views of these two figures demonstrate that engraving and interweaving lead to different occlusion behavior. Similar to color weaving [39], we also recommend appropriate color mapping for effective engraving and interweaving: the two combined LIC images use two different colors that are highly saturated and approximately isoluminant. Approximate isoluminance is instrumental in avoiding induced perception of shape and other features that could affect the visualization. Isoluminant colors can be conveniently generated by the method of Kindlmann et al. [21]. The colors are chosen to be highly discriminable in order to achieve an easy separation of the line structures and good visual continuity [19]. In addition, we choose colors that induce the same depth perception to prevent conflicting depth cues (e.g., red and blue would be bad choices because they induce different perceptual depth).

Fig. 5c shows another example of engraving. Here, sparse noise for the density LIC images is replaced by white noise to achieve a denser visual representation. In addition, the isoluminant color table is substituted by a gray-scale mapping. Although no bump mapping or embossing are applied, this image provides a subtle depth impression due to perceptually induced shape recognition connected to luminance changes. We recommend gray-scale mapping only in combination with the engraving model because this compositing scheme guarantees the consistency of gray-scale values and depth. Furthermore, Fig. 5c demonstrates that combined LIC also works with dense noise.

8 Implementation

Our GPU implementations of the 2D and 2.5D algorithms are based on C++, DirectX 9.0, and HLSL for shader programming. The above algorithms are mapped to vertex and pixel shaders, the data structures are realized by 2D or 3D textures. Shader model 3.0 is essential because we use loops in pixel shaders.

For the 2D implementation of ortho-vis with incremental temporal filtering, each operation in the pipeline in Fig. 4 is mapped to one pixel shader program that works on 2D images represented by 2D textures. Similarly, the analogous pipeline is implemented for tangent-vis, except for the missing rotation of the vector field. For steady vector fields, multiple render passes are not required for any of the stages. The template noise is precomputed on the CPU and low-pass filtered in Fourier space by using FFTW.[1] Data between different stages is transferred as 2D textures (16-bit floating-point format) filled by a means of the render-to-texture functionality. Semi-Lagrangian advection uses the built-in bilinear interpolation within 16-bit floating-point textures.

For the 2.5D implementation with incremental temporal filtering, each stage is essentially mapped to two shaders and two render passes: one shader implements the different variants of projecting the vector field onto the image plane; the subsequent shader is responsible for the actual particle tracing and/or integration. Data between stages is transferred as 2D textures with a 32-bit floating-point format. Semi-Lagrangian advection is based on a modified version of bilinear interpolation (see Section 6) that is explicitly implemented in a pixel shader. Once again, ortho-vis and

1. http://www.fftw.org.

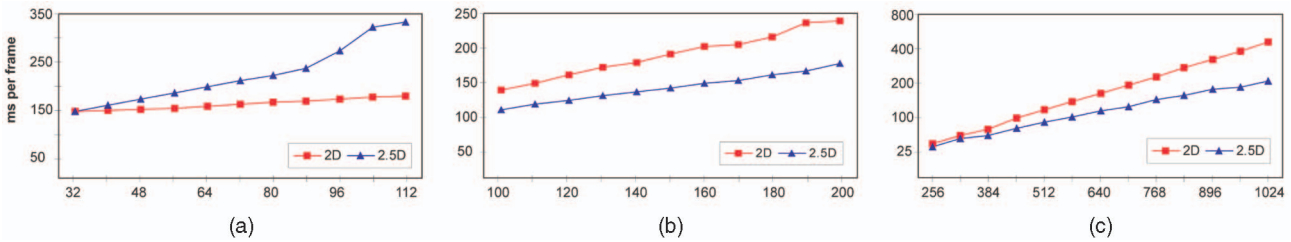


Fig. 6. Performance results for varying parameters and squared viewports: (a) number of virtual noise textures, (b) LIC integration length, (c) resolution. All vertical axes show rendering times in ms/frame. © Eurographics Association 2007; reproduced by kind permission of the Eurographics Association.

tangent-vis are based on essentially the same processing pipelines and implementations, except for the difference in the computation of the vector field.

Image compositing for combined LIC is implemented by one image-space operation that takes the intermediate results from ortho-vis and tangent-vis as input and that outputs the composited image. The engraving model works directly on the LIC density values of the intermediate images. For the interweaving model, additional height-map textures have to be generated during the computation of ortho-vis and tangent-vis. Here, it is not necessary to use multiple render passes to create the additional textures. Instead, the additional rendering is performed in parallel by writing to multiple render targets.

Time-dependent flow requires minor changes in the implementation—the first stage is modified, since the coherent noise is accumulated over past time steps. Here, for each of these time steps, the vector field is found by interpolating between the previous and next time step of the flow. The noise values are accumulated by using multiple render passes. The second and third stage remain unchanged.

The generic temporal filtering process requires a few changes to the aforementioned processing pipeline. First, intermediate, temporally unfiltered images of different times are stored in additional textures. Those images are the result of the second stage in the pipeline in Fig. 4. Stage three (semi-Lagrangian advection and alpha blending) is disabled. The textures with the unfiltered images are organized as a ring buffer that keeps track of the time tag of the buffer textures: the oldest texture is overwritten with the current result of the second stage. The number of elements in the ring buffer determines the filter size for temporal filtering. The actual filter process reads all unfiltered intermediate images, applies the convolution kernel, accumulates the filtered values, and writes the final result. This is implemented using multiple render passes: the images stored in the ring buffer are weighted and accumulated inside this loop with ping-pong rendering. Therefore, the complexity of the generic temporal filtering algorithm is linear with respect to the filter size, whereas the incremental filtering by recurring alpha blending is independent of the filter size (i.e., the alpha value).

The alternative approach of shortened LIC lines needs an additional texture precomputed to store the amount of inconsistency at all texel locations. During the second stage, this texture is accessed to retrieve and apply the scaling factor for the LIC length.

9 Results

9.1 Performance Results

The following tests were conducted on a Windows PC with Intel Dual Core CPU (1.86 GHz), 2 Gbytes of RAM, and NVIDIA GeForce 7900GS GPU with 256 Mbytes of texture memory. Fig. 6 documents timings (in milliseconds) for rendering a single image of orthogonal vector field visualization. Each plot shows measurements for a 2D data set (Benard convection, depicted in Fig. 7) and a 2.5D data set (a spherical object with a vector field given on a 3D texture).

Fig. 6a shows the behavior for varying integration length in the first stage in Fig. 4 (noise transport), i.e., different temporal convolution of virtual input noise images. The

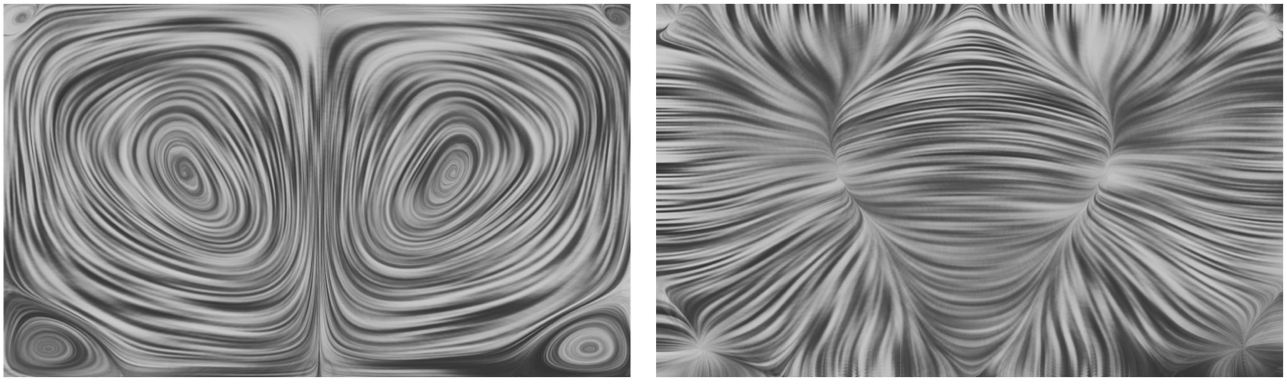


Fig. 7. 2D vector flow visualized in two ways: (a) conventional tangent LIC and (b) orthogonal vector field visualization.

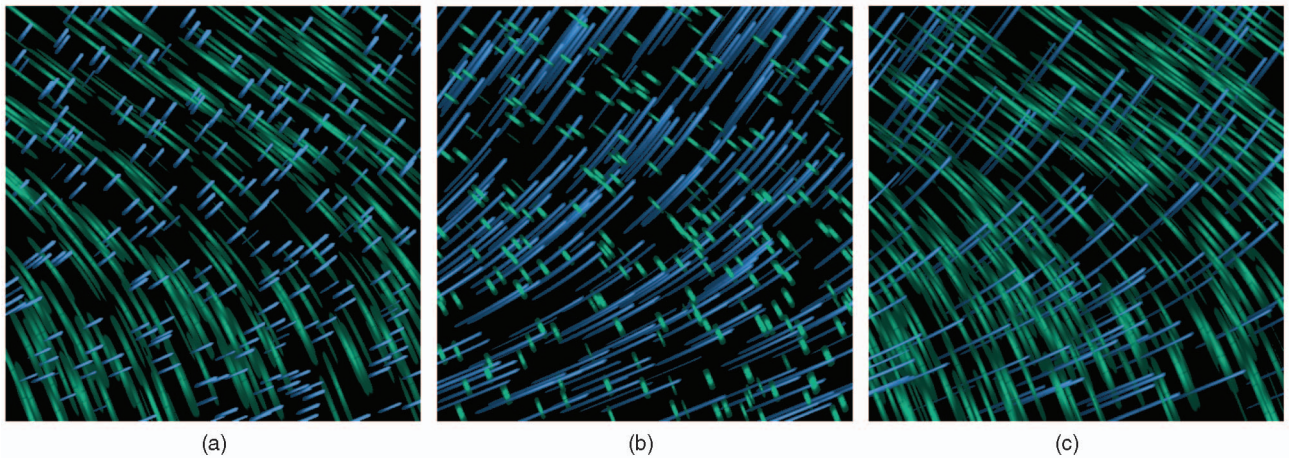


Fig. 8. Different parameters for interweaved LIC: (a) integration length for conventional LIC lines four times smaller than for orthogonal LIC lines, (b) integration length for orthogonal LIC lines four times smaller than for conventional LIC lines, and (c) density of orthogonal LIC lines is three times higher than for conventional LIC lines.

viewport size is 512^2, and the LIC convolution length is 2×100. The 2D case exhibits an almost linear behavior. The 2.5D algorithm shows an unexpected increase of rendering time for long convolution lengths, which might be explained by an influence of the texture cache. Fig. 6b reports timings for varying LIC integration length (for stage two in Fig. 4). The integration length is given as the length in one direction, i.e., the total number of integration steps is twice the displayed number. The viewport size is 512^2, and the temporal convolution length is 16. Finally, Fig. 6c illustrates the influence of the viewport size, for constant temporal convolution length (16 steps) and constant LIC convolution length (2×100 steps). Please note that the y-axis has a quadratic scale. As expected, all plots show almost linear behavior for the varying parameters. Therefore, quality (i.e., longer integration or more pixels) can be gradually balanced with rendering speed. In general, typical 2D and 2.5D visualizations render at some 10 frames per second, which facilitates interactive applications.

When interweaved LIC is enabled, the computation time for rendering one frame increases by approximately 60 percent, regardless of the other parameters. This is an expected result since additional render targets are used, and therefore, the execution time for the pixel shaders of stages one and two increases.

9.2 Qualitative Results

First, we compare our orthogonal vector field visualization with the existing method of tangential streamline LIC. In Fig. 7a, convection flow is used to present the two different LIC approaches for the 2D case. Fig. 7a shows the conventional way of visualizing the vector field using the standard LIC approach, whereas Fig. 7b shows the method of orthogonal vector field visualization.

Different compositing models for combined LIC are already discussed in Section 7 in Fig. 5. In addition, combined LIC allows us to change several parameters in order to modify the look of the resulting images, as shown in Fig. 8. Depending on how much detail in the visualization is desired, the density of the sparse noise for the conventional streamline LIC image, as well as for the orthogonal LIC image can be adjusted. In addition, the integration length for the two different LIC images can be modified.

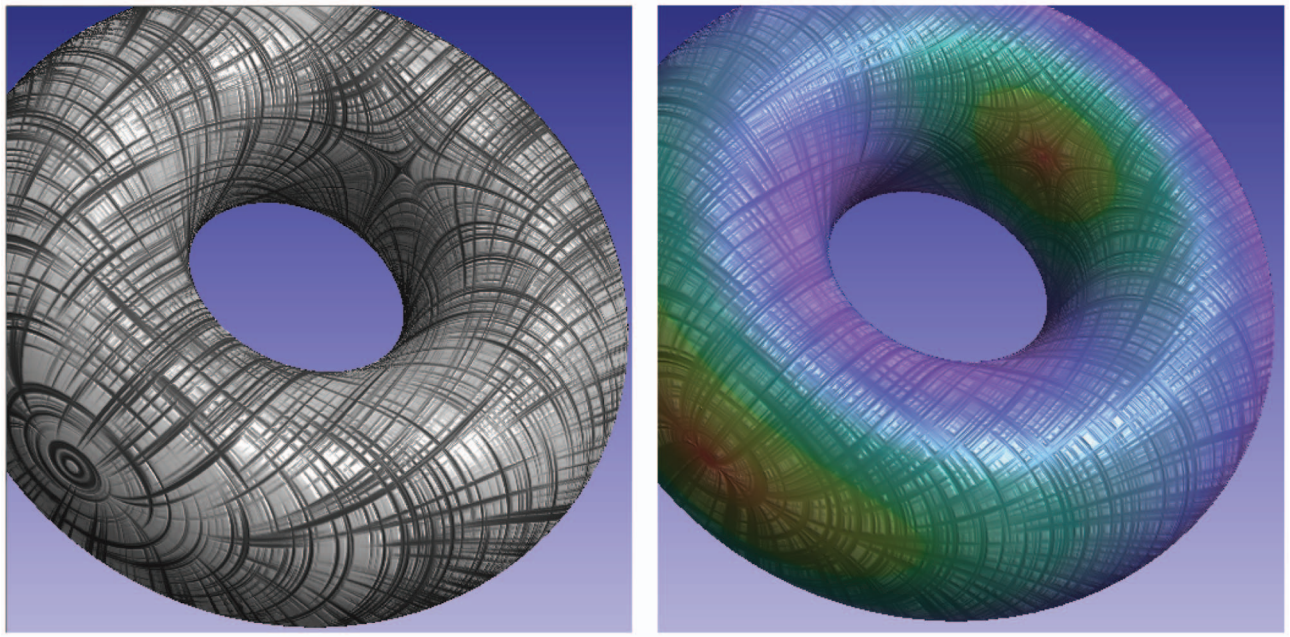
Fig. 10 illustrates tangent-vis and ortho-vis side-by-side for a 2.5D data set from an automotive CFD simulation. We can also combine our orthogonal vector field visualization with conventional LIC images in 2.5D: in Fig. 9a, a uniform vector field is projected onto the surface of a torus. Here, the engraving compositing scheme is applied, which immediately leads to a height-field interpretation. The height-field character is emphasized by bump mapping applied to the composited image (bump-mapping increases the contrast of the lines, making them even better perceivable). In Fig. 9b, the same technique is used, except for additional color coding of the magnitude of the vector field. The color image is blended with the LIC image, providing an additional cue for the velocity of the vector field.

Unfortunately, the static images in this paper are not sufficient to convey our visualization method, which makes heavy use of animation. Therefore, we strongly recommend watching the accompanying videos (available as supplemental material on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TVCG.2008.36 or on our Web page).[2] The videos, for example, compare traditional animated LIC and animated orthogonal LIC, demonstrating the differences in perceived speed of moving patterns (for same physical speed). Since the motion detectors are tuned for certain spatial frequencies, we suggest that the reader views the videos from varying distances in order to change the perception of motion.

9.3 Temporal Filtering

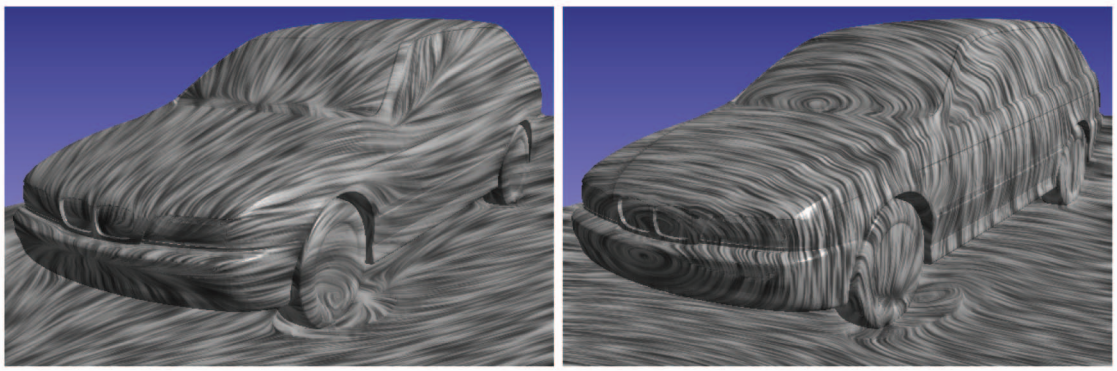
Fig. 11 shows the 2D orthogonal vector field visualization of a shear flow. As discussed in Section 4.3, a shear flow is a challenging example because of substantial inconsistencies in the transport of orthogonal LIC patterns. In Fig. 11a, the inexpensive exponential filter is used with an alpha value of 0.075. When alpha blending is being used to perform temporal filtering, a smaller alpha value leads to a wider filtering and, thus, to a larger blurring of the image in regions of inconsistency. This is helpful in perceiving the overall motion of the vector field. In addition, the accompanying videos (see footnote 2) show that alpha

2. http://www.vis.uni-stuttgart.de/texflowvis.



(a) (b)

Fig. 9. An example of combined LIC visualizing the tangential vector flow on the surface of a torus: (a) Bump mapping improves the perception of the LIC patterns. (b) Velocity magnitude is additionally encoded by using a color map.



(a) (b)

Fig. 10. Flow visualization on curved surfaces: (a) conventional, tangent LIC and (b) orthogonal vector field visualization. © Eurographics Association 2007; reproduced by kind permission of the Eurographics Association.

blending results in a reduction of shower-door effects (i.e., overlaid patterns seem to move at different speed). From experience, useful alpha values are in the range of 0.03-0.3, depending on the animation speed and structure of the vector field.

In Fig. 11b, again, an exponential filter is used; however, this time we use the version that results from the generic filtering approach with 25 intermediate images. This version shows slightly less pronounced artifacts because the generic filtering pipeline performs a completely Lagrangian particle transport. In contrast, the alpha blending of the incremental filtering process implies the use of semi-Lagrangian advection, which leads to numerical diffusion (see discussion in [49]). Fig. 11c is created with a Gaussian filter kernel using the generic filtering pipeline. This kind of filtering reduces irritating patterns in the region of the highest velocity because the Gaussian function exhibits a faster falloff in frequency space than the exponential function [47]. These irritating patterns can be observed in Figs. 11a and 11b as slightly undulating patterns. Fig. 11d shows the result of filtering with the sinc() kernel. In theory, the sinc() filter provides perfect temporal low-pass filtering; however, the filter has to be of infinite length in order to reach this quality. We use a relatively short filter length of 15 and cut the rest off of the infinite filter function. Nevertheless, the results show better visual quality than the exponential filter of length 25. Finally, Fig. 11e illustrates box filtering, used as a negative example: the region of high velocity is washed out and additional aliasing patterns appear. The reason for the bad quality of the box filter is its high contents of spurious high-frequency contributions in frequency space. In summary, Fig. 11 demonstrates that the generic filtering pipeline along with appropriate low-pass filters (like Gaussian or sinc()) provides best quality. Nevertheless, exponential filtering—even in the form of incremental alpha blending—leads to acceptable results. Therefore, our incremental filtering process is an acceptable option for fast interactive visualization.

Fig. 12 demonstrates the effects of increasing filter length. In this case, we choose the Gaussian filter and vary its kernel length from 10 up to 50. As the filter length

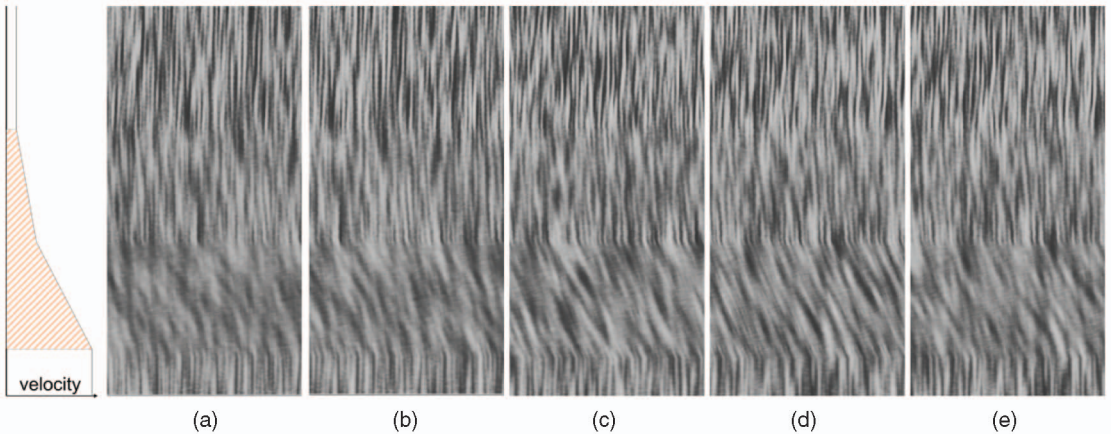


Fig. 11. Visualization of shear flow with different filters: (a) recurrent alpha blending with $\alpha = 0.075$, (b) exponential filter with the same falloff rate as in (a), (c) Gaussian filter, (d) sinc() filter, and (e) box filter.

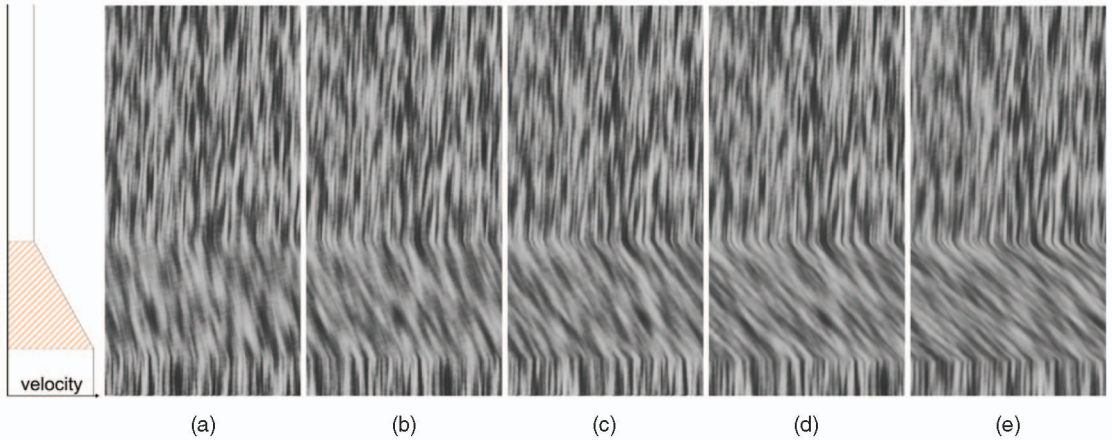


Fig. 12. Comparison of Gaussian filter kernels of varying length: (a) length 10, (b) length 20, (c) length 30, (d) length 40, and (e) length 50.

increases, more images of previous time steps are included in the filtering process. For each rendered frame, the filtering process advects the images of the filter support along the original vector field (see Section 4.3 for more details). Therefore, we can see an increase in the slope of the LIC patterns in regions of increasing velocity. This behavior is a by-product of our temporal filtering technique and can be observed regardless of the choice of filter kernel. This effect might be misleading in static images, since the direction of the flow in the shear area seems to change for increasing filter lengths. However, according to our observations, this misinterpretation is less likely in the animated visualization. The advantage of an increased filter length is improved temporal coherence. Based on our experience, insufficient temporal coherence results in clearly visible shower-door effects, whereas stronger temporal coherence with a filter length of approximately 30 or greater reduces the shower-door effects almost completely. Since temporal coherence cannot be shown in still images, we refer to the supplemental video material. Additionally, we hypothesize that strong temporal coherence and reduced shower-door effects facilitate the perception of consistent flow; however, this hypothesis requires further investigation by user studies.

Another way of overcoming the problem with shearing flow is shown in Fig. 13. For comparison, Figs. 13a and 13b show the results of recurrent alpha blending and a Gaussian filter, respectively, whereas Fig. 13c was generated without any temporal filtering. Here, the length of orthogonal LIC lines is modified proportionally to the strength of the shear in the visualized vector field. This way, the lines become very short in regions of strong shear to avoid that the LIC lines are "torn apart." The disadvantage of this approach is that it works only with animated visualization—in static images, the underlying vector field is harder to recognize because the line patterns become degenerate.

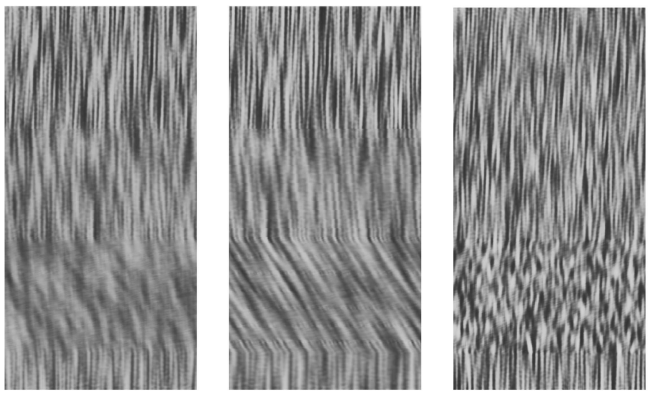


Fig. 13. Different ways to overcome the problem of shearing flow: (a) recurrent alpha-blending approach, (b) Gaussian filter, and (c) orthogonal LIC lines are shortened in regions of shearing flow.

10 Conclusion and Future Work

We have presented orthogonal vector field representation as a new means of displaying flow on 2D manifolds. The main motivation for choosing a perpendicular vector field is to decouple the spatial resolution of patterns along their motion direction from the length scale of those patterns. In this way, we can tune the spatial frequency to the local motion detectors of the HVS. Inconsistencies between orthogonal field lines and a time-coherent transport of those lines are resolved either by a generic temporal filtering process or alternatively by an inexpensive exponential filter implemented via recurring, incremental alpha blending. For typical data sets, however, these inconsistencies are not very prominent. For example, incompressible fluid flows are dominated by the divergence-free parts of the Hodge-Helmholtz decomposition and therefore often resemble the circular flow in Fig. 2 on a local scale. These kinds of data sets are ideally represented by animated orthogonal vector field visualization. In contrast, filtering is only needed for extreme flows with strong shear or curvature. Furthermore, we have demonstrated that the incremental computation of the exponential filter lends itself to a fast GPU implementation that is suitable for interactive visualization of 2D and 2.5D flow. In addition to the orthogonal vector field representation, the conventional visualization of tangential streamlines is supported. We have proposed novel compositing schemes to combine both representations in a single image: one scheme imitates engravings on a surface, whereas the other scheme leads to interweaved LIC lines.

While our approach is motivated by the characteristics of local motion perception, more refined investigations of those perceptual aspects are needed to quantify the effectiveness of animated orthogonal vector field visualization. Future perception studies could measure the discrimination and perceived speed of moving patterns under realistic settings; previous vision research uses a restricted class of stimuli that is not identical to LIC patterns. Furthermore, we have focused on low-level local motion perception. Therefore, the relationship between global motion perception and the effectiveness of conveying flow structures remains an open question. Further investigations are also needed to make sure that the visual signatures introduced by the temporal filtering process give the correct impression of the visualized vector field. Similarly, the effect of temporal filtering on the perception of temporal coherence could be analyzed by user studies. Finally, we would like to point out that our approach is restricted to 2D manifolds and cannot be directly extended to higher dimensions because there is no unique orthogonal vector field in 3D or higher dimensional space.

Appendix A

Mathematical Derivation of Inconsistent Transport

This appendix derives mathematical expressions that quantitatively describe the inconsistency between the advection of orthogonal LIC lines and the from-scratch construction of those lines. Those expressions are used in Section 4.3, providing the motivation of temporal filtering in our visualization approach.

We first derive the distance between a particle transported by the original steady vector field $\mathbf{v}$ and a particle transported by the orthogonal vector field $\mathbf{u} = \Omega\mathbf{v}$. Later, we apply this result to the specific case of advected

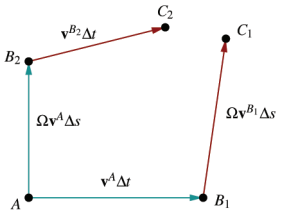


Fig. 14. Points and transport vectors used for the derivation of inconsistent transport of orthogonal lines.

orthogonal LIC lines. The derivation targets a description by means of differentials; therefore, the derivation utilizes first-order Taylor expansions to derive mathematical expressions, which is completely consistent with final results based on differential quantities.

We use the following notation for the mathematical expressions. Uppercase letters label points in 2D space, and those labels are used as superscripts attached to respective variables. One variable is the position of a point, denoted by $\mathbf{p}$; i.e., $\mathbf{p}^A$ is the position of point A. 2D vector field values are denoted $\mathbf{v}$; i.e., $\mathbf{v}^A$ is the vector field at point A. Components of points and vectors are indicated by subscripts $_x$ and $_y$, respectively. To denote the approximation that is introduced by using Taylor expansion, we use the approximation sign "$\approx$".

As illustrated in Fig. 14, we consider the following points related by first-order Euler integration for particle transport:

$$\begin{aligned} &\mathbf{p}^A \\ &\mathbf{p}^{B_1} \approx \mathbf{p}^A + \mathbf{v}^A\Delta t, \end{aligned} \tag{5}$$

$$\mathbf{p}^{B_2} \approx \mathbf{p}^A + \Omega\mathbf{v}^A\Delta s, \tag{6}$$

$$\mathbf{p}^{C_1} \approx \mathbf{p}^{B_1} + \Omega\mathbf{v}^{B_1}\Delta s, \tag{7}$$

$$\mathbf{p}^{C_2} \approx \mathbf{p}^{B_2} + \mathbf{v}^{B_2}\Delta t. \tag{8}$$

Point A is the starting point for our discussion. The step sizes Δt and Δs correspond to transport along the vector field $\mathbf{v}$ and the rotated vector field $\Omega\mathbf{v}$, respectively. The rotation operator is defined in a componentwise notation as

$$\Omega\begin{pmatrix} v_x \\ v_y \end{pmatrix} = \begin{pmatrix} -v_y \\ v_x \end{pmatrix}.$$

First-order Taylor expansion of the vector field around the point A leads to

$$\mathbf{v}^{B_1} \approx \mathbf{v}^A + (J\mathbf{v}^A)\,(\mathbf{p}^{B_1} - \mathbf{p}^A) \approx \mathbf{v}^A + (J\mathbf{v}^A)\,(\mathbf{v}^A\Delta t), \tag{9}$$

and

$$\mathbf{v}^{B_2} \approx \mathbf{v}^A + (J\mathbf{v}^A)\,(\mathbf{p}^{B_2} - \mathbf{p}^A) \approx \mathbf{v}^A + (J\mathbf{v}^A)\,(\Omega\mathbf{v}^A\Delta s), \tag{10}$$

with the Jacobi matrix

$$J\mathbf{v}^A = \begin{pmatrix} \frac{\partial v_x^A}{\partial x} & \frac{\partial v_x^A}{\partial y} \\ \frac{\partial v_y^A}{\partial x} & \frac{\partial v_y^A}{\partial y} \end{pmatrix}.$$

Without loss of generality, we use a Frenet frame [11] along the streamline of $\mathbf{v}$ as coordinate system. One axis of the Frenet frame points along the tangent direction $\mathbf{v}$, the other

axis points along the perpendicular direction $\Omega\mathbf{v}$. Any orthogonal coordinate system can be transformed to the Frenet frame by rotation. After this rotation, the x-axis is aligned with $\mathbf{v}$, and the y-axis is aligned with $\Omega\mathbf{v}$. In this coordinate system

$$\mathbf{v}^A = \begin{pmatrix} v_x^A \\ 0 \end{pmatrix}.$$

Using this coordinate system, (9) and (10) lead to the componentwise expressions:

$$\Omega\mathbf{v}^{B_1} \approx \begin{pmatrix} 0 \\ v_x^A \end{pmatrix} + \begin{pmatrix} -\frac{\partial v_y^A}{\partial x} \\ \frac{\partial v_x^A}{\partial x} \end{pmatrix} v_x^A \Delta t, \tag{11}$$

and

$$\mathbf{v}^{B_2} \approx \begin{pmatrix} v_x^A \\ 0 \end{pmatrix} + \begin{pmatrix} \frac{\partial v_x^A}{\partial y} \\ \frac{\partial v_y^A}{\partial y} \end{pmatrix} v_x^A \Delta s. \tag{12}$$

Using the Taylor expansions for the vector field from (11) and (12), the position expressions from (5)-(8) can be combined to yield the difference vector:

$$\begin{aligned} \mathbf{p}^{C_2} - \mathbf{p}^{C_1} \approx{}& \mathbf{p}^A + \begin{pmatrix} 0 \\ v_x^A \end{pmatrix} \Delta s + \left[\begin{pmatrix} v_x^A \\ 0 \end{pmatrix} + \begin{pmatrix} \frac{\partial v_x^A}{\partial y} \\ \frac{\partial v_y^A}{\partial y} \end{pmatrix} v_x^A \Delta s \right] \Delta t \\ & - \left\{ \mathbf{p}^A + \begin{pmatrix} v_x^A \\ 0 \end{pmatrix} \Delta t + \left[\begin{pmatrix} 0 \\ v_x^A \end{pmatrix} + \begin{pmatrix} -\frac{\partial v_y^A}{\partial x} \\ \frac{\partial v_x^A}{\partial x} \end{pmatrix} v_x^A \Delta t \right] \Delta s \right\} \\ ={}& \begin{pmatrix} \frac{\partial v_x^A}{\partial y} + \frac{\partial v_y^A}{\partial x} \\ \frac{\partial v_y^A}{\partial y} - \frac{\partial v_x^A}{\partial x} \end{pmatrix} v_x^A \Delta s \Delta t. \end{aligned} \tag{13}$$

If we were to construct orthogonal vector field lines that would exhibit velocity-dependent markers along those lines, then (13) would appropriately describe the inconsistency between advection of orthogonal lines from the previous time step and from-scratch construction of orthogonal lines for the current time step, i.e., (13) describes the difference in the transport of point particles along $\mathbf{v}$ and $\Omega\mathbf{v}$, respectively. The scaling factors Δs and Δt represent the step sizes for the discretization of transport, and v_x^A can be considered as an overall scaling factor for flow velocity. Therefore,

$$\boldsymbol{\delta}_{\text{full}} = \begin{pmatrix} \frac{\partial v_x^A}{\partial y} + \frac{\partial v_y^A}{\partial x} \\ \frac{\partial v_y^A}{\partial y} - \frac{\partial v_x^A}{\partial x} \end{pmatrix} \tag{14}$$

is the interesting vector-valued measure for the inconsistencies of particle transport.

In fact, the orthogonal vector field is visualized by LIC lines based on the normalized vector field. Put differently, we are not interested in how far $\boldsymbol{\delta}_{\text{full}}$ differs in the y-direction because a difference in distance along the y-direction is compensated by the normalization during the LIC computation. Therefore, only the x component of $\boldsymbol{\delta}_{\text{full}}$ is relevant for our visualization, leading to a single-component measure for inconsistency in orthogonal LIC transport:

$$\boldsymbol{\delta}_{LIC} = \frac{\partial v_x^A}{\partial y} + \frac{\partial v_y^A}{\partial x}. \tag{15}$$

The term $\frac{\partial v_y^A}{\partial x}$ describes the curvature of the vector field, whereas the term $\frac{\partial v_x^A}{\partial y}$ describes the shear. We refer to De Leeuw and Van Wijk [8] for a detailed description and interpretation of those differential quantities of a vector field. Their discussion targets 3D flow, which can be immediately applied to our scenario of 2D flow by restriction to two coordinates x and y.

ACKNOWLEDGMENTS

This paper is an extended version of a paper that appeared in the Proceedings of the Eurographics/IEEE Visualization and Graphics Technical Committee Symposium on Visualization (Eurovis '07) [2] (Used by the kind permission of the Eurographics Association). The data set used in Fig. 10 was kindly provided by the BMW Group. Special thanks to the anonymous reviewers for the very constructive comments.

REFERENCES

[1] S.J. Anderson and D.C. Burr, "Receptive Field Size of Human Motion Detection Units," *Vision Research*, vol. 27, pp. 621-635, 1987.
[2] S. Bachthaler and D. Weiskopf, "Animation of Orthogonal Texture-Based Vector Field Visualization," *Proc. EG/IEEE VGTC Symp. Visualization (Eurovis '07)*, pp. 219-226, 2007.
[3] L. Bartram and C. Ware, "Filtering and Brushing with Motion," *Information Visualization*, vol. 1, no. 1, pp. 66-79, 2002.
[4] P.J. Bex and S.C. Dakin, "Comparison of the Spatial-Frequency Selectivity of Local and Global Motion Detectors," *J. Optical Soc. of America A*, vol. 19, pp. 670-677, 2002.
[5] B. Cabral and L.C. Leedom, "Imaging Vector Fields Using Line Integral Convolution," *Proc. ACM SIGGRAPH '93*, pp. 263-270, 1993.
[6] J.R. Cavanaugh, W. Bair, and J.A. Movshon, "Nature and Interaction of Signals from the Receptive Field Center and Surround in Macaque V1 Neurons," *J. Neurophysiology*, vol. 88, pp. 2530-2546, 2002.
[7] R. Cranley and T. Patterson, "Randomization of Number Theoretic Methods for Multiple Integration," *SIAM J. Numerical Analysis*, vol. 13, pp. 904-914, 1976.
[8] W.C. de Leeuw and J.J. van Wijk, "A Probe for Local Flow Field Visualization," *Proc. IEEE Visualization*, pp. 39-45, 1993.
[9] G. Erlebacher, B. Jobard, and D. Weiskopf, "Flow Textures," *The Visualization Handbook*, C.D. Hansen and C.R. Johnson, eds., pp. 279-293, Elsevier, 2005.
[10] L.K. Forssell and S.D. Cohen, "Using Line Integral Convolution for Flow Visualization: Curvilinear Grids, Variable-Speed Animation, and Unsteady Flows," *IEEE Trans. Visualization and Computer Graphics*, vol. 1, no. 2, pp. 133-141, June 1995.
[11] J. Gallier, *Geometric Methods and Applications: For Computer Science and Engineering*. Springer, 2001.
[12] H. Goldstein, *Classical Mechanics*, second ed. Addison-Wesley, 1980.
[13] A. Gooch, B. Gooch, P. Shirley, and E. Cohen, "A Non-Photorealistic Lighting Model for Automatic Technical Illustration," *Proc. ACM SIGGRAPH '98*, pp. 447-452, 1998.
[14] N. Gossett and B. Chen, "Paint Inspired Color Mixing and Compositing for Visualization," *Proc. IEEE Symp. Information Visualization*, pp. 113-118, 2004.
[15] H. Hagh-Shenas, V. Interrante, C. Healey, and S. Kim, "Weaving versus Blending: A Quantitative Assessment of the Information Carrying Capacities of Two Alternative Methods for Conveying Multivariate Data with Color," *Proc. IEEE Symp. Information Visualization*, pp. 1270-1277, 2007.
[16] C.G. Healey, K.S. Booth, and J.T. Enns, "High-Speed Visual Estimation Using Preattentive Processing," *ACM Trans. Computer-Human Interaction*, vol. 3, no. 2, pp. 107-135, 1996.

[17] H. Helmholtz, "Über Integrale der Hydrodynamischen Gleichungen, welche den Wirbelbewegungen entsprechen," *J. für die reine und angewandte Math.*, vol. 55, pp. 25-55, 1858.
[18] I. Hotz, L. Feng, H. Hagen, B. Hamann, K. Joy, and B. Jeremic, "Physically Based Methods for Tensor Field Visualization," *Proc. IEEE Visualization*, pp. 123-130, 2004.
[19] V. Interrante and C. Grosch, "Strategies for Effectively Visualizing 3D Flow with Volume LIC," *Proc. IEEE Visualization*, pp. 421-424, 1997.
[20] B. Jobard, G. Erlebacher, and M.Y. Hussaini, "Lagrangian-Eulerian Advection of Noise and Dye Textures for Unsteady Flow Visualization," *IEEE Trans. Visualization and Computer Graphics*, vol. 8, no. 3, pp. 211-222, July-Sept. 2002.
[21] G. Kindlmann, E. Reinhard, and S. Creem, "Face-Based Luminance Matching for Perceptual Colormap Generation," *Proc. IEEE Visualization Conf.*, pp. 299-306, 2002.
[22] M.S. Langer, J. Pereira, and D. Rekhi, "Perceptual Limits on 2D Motion-Field Visualization," *ACM Trans. Applied Perception*, vol. 3, no. 3, pp. 179-193, 2006.
[23] R.S. Laramee, H. Hauser, H. Doleisch, B. Vrolijk, F.H. Post, and D. Weiskopf, "The State of the Art in Flow Visualization: Dense and Texture-Based Techniques," *Computer Graphics Forum*, vol. 23, no. 2, pp. 143-161, 2004.
[24] R.S. Laramee, B. Jobard, and H. Hauser, "Image Space Based Visualization of Unsteady Flow on Surfaces," *Proc. IEEE Visualization*, pp. 131-138, 2003.
[25] W. Lefer, B. Jobard, and C. Leduc, "High-Quality Animation of 2D Steady Vector Fields," *IEEE Trans. Visualization and Computer Graphics*, vol. 10, no. 1, pp. 2-14, Jan./Feb. 2004.
[26] G.-S. Li, X. Tricoche, and C. Hansen, "GPUFLIC: Interactive and Accurate Dense Visualization of Unsteady Flows," *Proc. EG/IEEE VGTC Symp. Visualization (Eurovis '06)*, pp. 29-34, 2006.
[27] S. Limoges, C. Ware, and W. Knight, "Displaying Correlations Using Position, Motion, Point Size or Point Color," *Proc. Graphics Interface*, pp. 262-265, 1989.
[28] Z.P. Liu and R.J. Moorhead, "AUFLIC: An Accelerated Algorithm for Unsteady Flow Line Integral Convolution," *Proc. EG/IEEE TCVG Symp. Visualization*, pp. 43-52, 2002.
[29] E.B. Lum, A. Stompel, and K.-L. Ma, "Using Motion to Illustrate Static 3D Shape—Kinetic Visualization," *IEEE Trans. Visualization and Computer Graphics*, vol. 9, no. 2, pp. 115-126, Apr.-June 2003.
[30] K. Myszkowski, P. Rokita, and T. Tawara, "Perception-Based Fast Rendering and Antialiasing of Walkthrough Sequences," *IEEE Trans. Visualization and Computer Graphics*, vol. 6, no. 4, pp. 360-379, Oct.-Dec. 2000.
[31] K. Myszkowski, T. Tawara, H. Akamine, and H.-P. Seidel, "Perception-Guided Global Illumination Solution for Animation Rendering," *Proc. ACM SIGGRAPH '01*, pp. 221-230, 2001.
[32] K. Polthier and E. Preuß, "Variational Approach to Vector Field Decomposition," *Proc. EG/IEEE TCVG Symp. Visualization*, pp. 147-155, 2000.
[33] K. Polthier and E. Preuß, "Identifying Vector Field Singularities Using a Discrete Hodge Decomposition," *Visualization and Math. III*, H.-C. Hege and K. Polthier, eds., pp. 113-134, Springer, 2003.
[34] H.-W. Shen and D.L. Kao, "UFLIC: A Line Integral Convolution Algorithm for Visualizing Unsteady Flows," *Proc. IEEE Visualization Conf.*, pp. 317-323, 1997.
[35] A. Sundquist, "Dynamic Line Integral Convolution for Visualizing Streamline Evolution," *IEEE Trans. Visualization and Computer Graphics*, vol. 9, no. 2, pp. 273-282, Apr.-June 2003.
[36] D. Tadin and J.S. Lappin, "Optimal Size for Perceiving Motion Decreases with Contrast," *Vision Research*, vol. 45, pp. 2059-2064, 2005.
[37] Y. Tong, S. Lombeyda, A.N. Hirani, and M. Desbrun, "Discrete Multiscale Vector Field Decomposition," *ACM Trans. Graphics, Proc. ACM SIGGRAPH '03*, vol. 22, no. 3, pp. 445-452, 2003.
[38] T. Urness, V. Interrante, E. Longmire, I. Marusic, and B. Ganapathisubramani, "Techniques for Visualizing Multi-Valued Flow Data," *Proc. Eurographics/IEEE TCVG Symp. Visualization*, pp. 165-172, 2004.
[39] T. Urness, V. Interrante, I. Marusic, E. Longmire, and B. Ganapathisubramani, "Effectively Visualizing Multi-Valued Flow Data Using Color and Texture," *Proc. IEEE Visualization*, pp. 115-121, 2003.
[40] J.J. van Wijk, "Image Based Flow Visualization," *ACM Trans. Graphics (Proc. ACM SIGGRAPH '02)*, vol. 21, no. 3, pp. 745-754, 2002.
[41] J.J. van Wijk, "Image Based Flow Visualization for Curved Surfaces," *Proc. IEEE Visualization Conf.*, pp. 123-130, 2003.
[42] C. Ware, *Information Visualization: Perception for Design*, second ed. Morgan Kaufmann, 2004.
[43] A.B. Watson and K. Turano, "The Optimal Motion Stimulus," *Vision Research*, vol. 35, pp. 325-336, 1995.
[44] R. Wegenkittl, E. Gröller, and W. Purgathofer, "Animating Flow Fields: Rendering of Oriented Line Integral Convolution," *Proc. Computer Animation*, pp. 15-21, 1997.
[45] D. Weiskopf, "On the Role of Color in the Perception of Motion in Animated Visualizations," *Proc. IEEE Visualization*, pp. 305-312, 2004.
[46] D. Weiskopf, *GPU-Based Interactive Visualization Techniques*. Springer, 2006.
[47] D. Weiskopf, "Iterative Twofold Line Integral Convolution for Texture-Based Vector Field Visualization," *Math. Foundations of Scientific Visualization, Computer Graphics, and Massive Data Exploration*, T. Möller, B. Hamann, and R. Russell, eds., Springer, http://www.visus.uni-stuttgart.de/~weiskopf/publications, 2007.
[48] D. Weiskopf and G. Erlebacher, "Overview of Flow Visualization," *The Visualization Handbook*, C.D. Hansen and C.R. Johnson, eds., pp. 261-278, Elsevier, 2005.
[49] D. Weiskopf, G. Erlebacher, and T. Ertl, "A Texture-Based Framework for Spacetime-Coherent Visualization of Time-Dependent Vector Fields," *Proc. IEEE Visualization*, pp. 107-114, 2003.
[50] D. Weiskopf and T. Ertl, "A Hybrid Physical/Device-Space Approach for Spatio-Temporally Coherent Interactive Texture Advection on Curved Surfaces," *Graphics Interface*, pp. 263-270, 2004.
[51] X. Zheng and A. Pang, "HyperLIC," *Proc. IEEE Visualization*, pp. 249-256, 2003.

Sven Bachthaler received the Dipl. Inf. (MSc) degree from the Universität Stuttgart, Germany, in 2005. He started with the graduate program of computing science at Simon Fraser University, Canada, in 2006 and is currently working toward the PhD degree at Universität Stuttgart, Germany. His research interests include scientific visualization and computer graphics.

Daniel Weiskopf received the MSc (Diplom) degree in physics and the PhD degree in physics from Eberhard-Karls-Universität Tübingen, Germany, and the Habilitation degree in computer science at Universität Stuttgart, Germany. From 2005 to 2007, he was an assistant professor of computing science at Simon Fraser University, Canada. Since 2007, he has been a professor of computer science at the Visualization Research Center, Universität Stuttgart (VISUS) and at the Visualization and Interactive Systems Institute (VIS), Universität Stuttgart. His research interests include scientific visualization, GPU methods, real-time computer graphics, mixed realities, ubiquitous visualization, perception-oriented computer graphics, and special and general relativity. He is a member of the ACM SIGGRAPH, the Gesellschaft für Informatik, and the IEEE Computer Society.