Animation of Orthogonal Texture-Based Vector Field Visualization

Sven Bachthaler and Daniel Weiskopf

Graphics, Usability, and Visualization (GrUVi) Lab, Simon Fraser University, Canada

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. In addition, a filtering process is described to achieve a consistent and temporally coherent animation of the orthogonal vector field visualization. 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.

Categories and Subject Descriptors (according to ACM CCS): I.3.3 [Computer Graphics]: Picture/Image Generation I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism

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 line-like 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 LIC (line integral convolution) 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 an animated visualization? In general, animation has been used successfully for vector field visualization because it can show the direction, orientation, and magnitude of the vec-

tor field. In addition, animation can alleviate the curve intersection issues that occur for long pathlines or streaklines of a time-dependent flow. However, we are not aware of any previous work that would have considered the perception of such animations. Based on results from vision research, we claim that existing approaches like animated streamlines are non-optimal 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 3.2).

Therefore, we propose to decouple the direction of the line-like 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 line-like 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). Furthermore, this visualization approach resembles a moving wave front of the vector field and therefore provides an intuitive analogy to the real world.

The main contributions of this paper are: (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 animated orthogonal vector field representation is introduced for the visualization of steady and unsteady flow. (3) A filtering process is proposed to obtain a consistent and temporally coherent animation of the orthogonal vector field visualization. (4) An efficient texture-based algorithm for 2D planar vector fields is described. (5) This algorithm is extended to tangential vector fields on curved surfaces. (6) Fast GPU implementations of both algorithms are presented.

2. Previous Work

This paper describes a technique for texture-based vector field visualization. We refer to the survey chapter [WE05] for an overview of vector field visualization in general and to the article [LHD*04] for a presentation of the state of the art in texture-based methods.

Our visualization approach relies on line integral convolution (LIC) [CL93] as a role model to extract and display line-like structures. For the visualization of 2D vector fields, we adopt a GPU version of LIC [WEE03]. For data sets given on curved surfaces embedded in 3D space, we extend image-space advection [LJH03, vW03] and combine it with a related hybrid image/object space method for LIC [WE04].

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 line-like 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 [CL93, LJL04]. Most methods for time-dependent data directly support an animated visualization, such as LIC methods [FC95], texture advection techniques [JEH02, vW02], or unsteady flow LIC (UFLIC) [SK97] and its recent variants [LM02,LTH06]. A decoupling of spatial structures and temporal behavior through animation is described in a generic texture-based framework [WEE03], which generalizes dynamic LIC [Sun03], designed for animated electric or magnetic fields. In this paper, the framework [WEE03] 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 [War04]. 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 [LSM03], preattentive processing [HBE96], filtering and brushing with motion [BW02], multivariate visualization [LWK89], perceptual limits on 2D motion-field visualization [LPR06], and the influence of color on motion perception [Wei04].

3. Orthogonal Vector Field Representation

We first introduce orthogonal vector field representation 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.

3.1. Spatial Patterns

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

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

The vector field maps a point $x \in M$ to a vector in the corresponding tangent space at that point, $T_x M$. 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 an angle $\pi/2$. The inverse operator Ω^{-1} yields a rotation by $-\pi/2$. The orthogonal vector field u is defined as

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

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 a texture-based visualization of streamlines by means of LIC (see Sections 4 and 5), but other methods such as geometrically constructed streamlines might also be employed. Figure 1 illustrates an example of u and v by 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
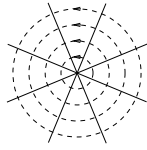
Figure 1: *Illustration of two perpendicular families of lines: one for a circular vector field v (dashed lines) and one for a radial vector field u (solid lines).*

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 a $\pi/2$ 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, Figure 1, 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. Adopting the notation of Polthier and Preuß [PP00], 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. 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 for curl-free vector fields.

3.2. Animation and Motion Perception

So far, we have only discussed the 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. Figure 1 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 an 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 exam-

ple, [AB87]). 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. [CBM02] 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 [TL05], where high contrast favors the detection of high-frequency stimuli and low contrast favors large stimuli. Tadin and Lappin [TL05] report an optimal size of 0.5 deg (degrees with respect to the subtended angle as seen by the viewer) for a high contrast of 92%. 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., [BD02]). In this paper, we focus on local motion detection, which is optimal for certain spatial frequencies. Bex and Dakin [BD02] 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 [WT95] 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 a typical setup with a 600^2 window viewed under a 30 deg field of view, 2 cycles/deg correspond to 2×60 dark or bright LIC lines (with one pair of dark and bright lines per cycle), or approximately 10 pixels wide line patterns. This is slightly wider than typical LIC lines of 2–3 pixels width, but still in a similar range. In contrast, animation along extended streamlines of 100–150 pixels length is much further away from the perceptual optimum.

3.3. 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
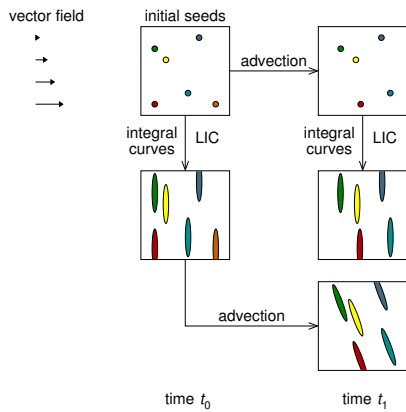
Figure 2: *Illustration of two transport and visualization approaches, applied to a shear flow. The two resulting images (middle and bottom images in the right column) differ because LIC and advection are not commutative. Seeds and streamlines are colored to allow for an easier recognition of correspondence between images.*

LIC) along v from time t_0 to t_1 and then construct the integral curves of u for time t_1. Figure 2 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 Figure 2. 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 construct a mechanism that maintains orthogonal vector field lines and achieves temporal coherence at the same time.

To overcome this problem, 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 ($t_0 \le t_i \le t_1$); 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 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

$$I = \int_{t_0}^{t_1} k(t_i) T_{t_i}(N)\, dt_i \,, \qquad (1)$$

with a filter kernel $k(t)$ normalized according to $\int k(t)\, dt = 1$. This filtering process allows us to trade clearly defined line-like 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 fil-

tering. The circular flow of Figure 1 is an example of a completely consistent advection and integral-curve construction.

An 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.

4. 2D Algorithm

For a flow on a planar 2D domain, all relevant information is 2D (vector field, input noise images, 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 (Figure 3). 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 3.3: (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 a 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 Eq. (1). 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. Similarly to [WEE03, 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
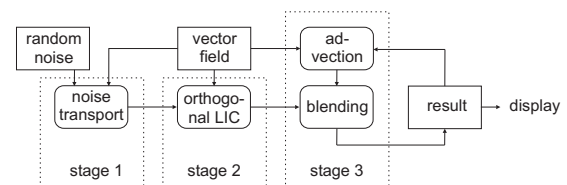

Figure 3: *Processing stages and data flow for the 2D algorithm.*

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 [CP76], which adds the same random shift to each point of the template image. The random shifts are applied on-the-fly while a noise injection texture is accessed.

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 [WEE03]. 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. Similarly 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 Eq. (1). 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 equation (1) would require to compute and store several intermediate images T_{t_i}. We avoid this additional work and memory consumption 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 $(1 - \alpha)$) [EJW05]. 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.

5. 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 Figure 3), 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 intersected by the surface or attached to the vertices of the surface. Since our algorithm is designed for tangential vector fields, a possibly non-tangential 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 [WE04]. 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 anti-aliasing [WE04]. The result of the first stage is a temporally coherent noise image that moves along the surface. This noise texture is given in image space.

From now on, we work in image space only, reminiscent of image-space advection techniques [LJH03, vW03]. 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 boundary is identified with a mask that contains the classification of pixels as foreground or background pixels. Since a complete LIC is evaluated, we are free to choose any filter kernel. In contrast, image-space advection techniques [LJH03, vW03] are restricted to an exponential kernel, which yields lower image quality than the Gaussian kernel used in our implementation (see the discussion of filter quality in [Wei07]).

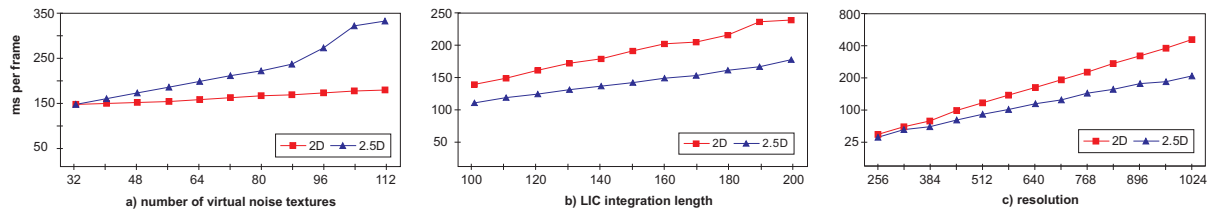The third stage (advection and blending) consists of

Figure 4: *Performance results for varying parameters and squared viewports. All vertical axes show ms/frame.*

the following components: projection of the original, non-rotated 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. However, 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 [LJH03] 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 [GGSC98] for surface illumination. Bump mapping is also available as an option to emphasize the structure of the vector field texture. Here, the texture from stage three is interpreted as a height field that perturbs the normal vectors.

6. 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, each operation in the pipeline of Figure 3 is mapped to one pixel shader program that works on 2D images represented by 2D textures. The template noise is precomputed on the CPU and low-pass filtered in Fourier space by using FFTW. Data between different stages is transferred as 2D textures (16-bit floating point format) filled by 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, 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 32-bit floating point format. Semi-Lagrangian advection is based on a modified version of bilinear interpolation (see Section 5) that is explicitly implemented in a pixel shader.

Unsteady flows require only minor changes in the implementation. The time-dependent vector field is represented as a temporal stack of textures. The first stage accesses different time steps of the vector field during particle tracing. In contrast to the time-independent case, multiple render passes are used to trace particles while the different time steps of the vector field are accessed. Linear interpolation between two time steps of the vector field allows us to decouple the animation rate from the temporal sampling of the data set. The second stage and third stage remain unchanged, provided that the current time step of the flow is used.

7. Results

The following tests were conducted on a Windows PC with Intel Dual Core CPU (1.86 GHz), 2 GB RAM, and NVIDIA GeForce 7900GS GPU with 256 MB of texture memory. Figure 4 documents timings (in milliseconds) for rendering a single image. Each plot shows measurements for a 2D data set (Benard convection) and a 2.5D data set (a spherical object with a vector field given on a 3D texture). Figure 4a) reports timings for varying LIC integration length (for stage two). 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. Figure 4b) shows the behavior for varying integration length in the first stage (noise transport), i.e., different temporal convolution of virtual input noise images. The 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 (for the dependent 3D texture lookups due to the on-the-fly Cranley-Patterson rotation). Finally, Figure 4c) 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
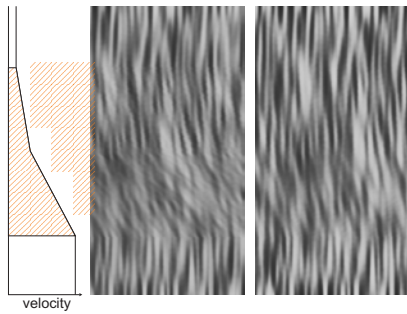
Figure 5: *Visualization of shear flow with different alpha blending weights: (a) $\alpha = 0.1$, (b) $\alpha = 0.3$.*

a quadratic scale. As expected, all the 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.

Figure 5 shows the 2D orthogonal vector field visualization of a shear flow. As discussed in Section 3.3, a shear flow is an extremely challenging example because of substantial inconsistencies in the transport of orthogonal LIC patterns. Figure 5 compares different alpha values for the blending in stage three of the algorithm. A smaller alpha value leads to a wider filtering and thus to a larger blurring of the image in regions of inconsistency. In addition, the accompanying videos[§] show that alpha blending results in a reduction of flickering and shower-door effects (overlaid patterns seem to move at different speed). From experience, useful alpha values are in the range of 0.05–0.3, depending on the animation speed and structure of the vector field.

Figure 6 illustrates standard LIC and orthogonal vector field visualization for a 2.5D data set from an automotive CFD simulation. Unfortunately, the static images in the paper are not sufficient to convey our visualization method, which makes heavy use of animation. Therefore, we strongly recommend watching the accompanying videos[§]. 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.

8. Conclusion and Future Work

We have presented an orthogonal vector field representation as a new means of displaying flow on 2D manifolds. The main motivation for choosing a perpendicular vector field is
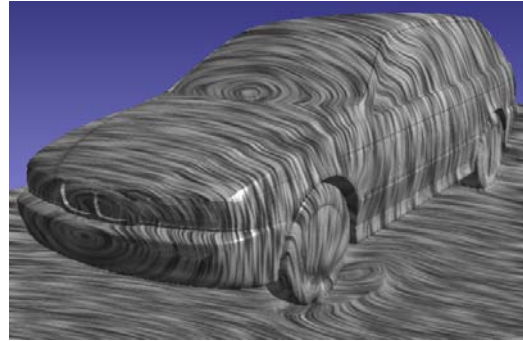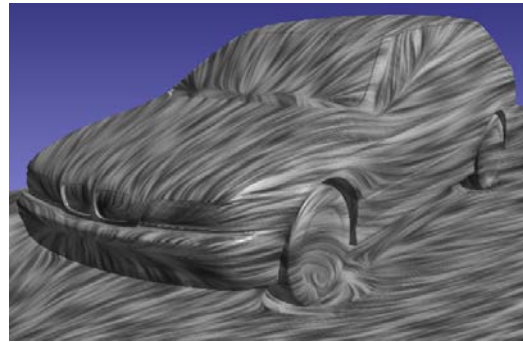

Figure 6: *Flow visualization on curved surfaces: (top) standard LIC, (bottom) orthogonal vector field visualization.*

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 by an exponential filter implemented via recurring 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 from Figure 1 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 cases like shear flows. In future work, the quality of the filter could be improved by including kernels with a faster falloff in frequency space than the exponential function (see the related discussion [Wei07]).

While our approach is motivated by the characteristics of 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

---

[§] http://www.cs.sfu.ca/~sbachtha/personal/eurovis07

relationship between global motion perception and the effectiveness of conveying flow structures remains an open question. 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.

Acknowledgements

References

[AB87] ANDERSON S. J., BURR D. C.: Receptive field size of human motion detection units. *Vision Research 27* (1987), 621–635. 3

[BD02] BEX P. J., DAKIN S. C.: Comparison of the spatial-frequency selectivity of local and global motion detectors. *Journal of the Optical Society of America A 19* (2002), 670–677. 3

[BW02] BARTRAM L., WARE C.: Filtering and brushing with motion. *Information Visualization 1*, 1 (2002), 66–79. 2

[CBM02] CAVANAUGH J. R., BAIR W., MOVSHON J. A.: Nature and interaction of signals from the receptive field center and surround in macaque V1 neurons. *Journal of Neurophysiology 88* (2002), 2530–2546. 3

[CL93] CABRAL B., LEEDOM L. C.: Imaging vector fields using line integral convolution. In *ACM SIGGRAPH 1993 Conference* (1993), pp. 263–270. 2

[CP76] CRANLEY R., PATTERSON T.: Randomization of number theoretic methods for multiple integration. *SIAM Journal on Numerical Analysis 13* (1976), 904–914. 5

[EJW05] ERLEBACHER G., JOBARD B., WEISKOPF D.: Flow textures. In *The Visualization Handbook*, Hansen C. D., Johnson C. R., (Eds.). Elsevier, Amsterdam, 2005, pp. 279–293. 5

[FC95] FORSSELL L. K., COHEN S. D.: Using line integral convolution for flow visualization: Curvilinear grids, variable-speed animation, and unsteady flows. *IEEE Transactions on Visualization and Computer Graphics 1*, 2 (1995), 133–141. 2

[GGSC98] GOOCH A., GOOCH B., SHIRLEY P., COHEN E.: A non-photorealistic lighting model for automatic technical illustration. In *ACM SIGGRAPH 1998 Conference* (1998), pp. 447–452. 6

[HBE96] HEALEY C. G., BOOTH K. S., ENNS J. T.: High-speed visual estimation using preattentive processing. *ACM Transactions on Computer-Human Interaction 3*, 2 (1996), 107–135. 2

[JEH02] JOBARD B., ERLEBACHER G., HUSSAINI M. Y.: Lagrangian-Eulerian advection of noise and dye textures for unsteady flow visualization. *IEEE Transactions on Visualization and Computer Graphics 8*, 3 (2002), 211–222. 2

[LHD*04] LARAMEE R. S., HAUSER H., DOLEISCH H., VROLIJK B., POST F. H., WEISKOPF D.: The state of the art in flow visualization: Dense and texture-based techniques. *Computer Graphics Forum 23*, 2 (2004), 143–161. 2

[LJH03] LARAMEE R. S., JOBARD B., HAUSER H.: Image space based visualization of unsteady flow on surfaces. In *IEEE Visualization* (2003), pp. 131–138. 2, 5, 6

[LJL04] LEFER W., JOBARD B., LEDUC C.: High-quality animation of 2D steady vector fields. *IEEE Transactions on Visualization and Computer Graphics 10*, 1 (2004), 2–14. 2

[LM02] LIU Z. P., MOORHEAD R. J.: AUFLIC: An accelerated algorithm for unsteady flow line integral convolution. In *EG / IEEE TCVG Symposium on Visualization* (2002), pp. 43–52. 2

[LPR06] LANGER M. S., PEREIRA J., REKHI D.: Perceptual limits on 2D motion-field visualization. *ACM Transactions on Applied Perception 3*, 3 (2006), 179–193. 2

[LSM03] LUM E. B., STOMPEL A., MA K.-L.: Using motion to illustrate static 3D shape—kinetic visualization. *IEEE Transactions on Visualization and Computer Graphics 9*, 2 (2003), 115–126. 2

[LTH06] LI G.-S., TRICOCHE X., HANSEN C.: GPUFLIC: interactive and accurate dense visualization of unsteady flows. In *Eurovis (EG / IEEE VGTC Symposium on Visualization)* (2006), pp. 29–34. 2

[LWK89] LIMOGES S., WARE C., KNIGHT W.: Displaying correlations using position, motion, point size or point color. In *Graphics Interface* (1989), pp. 262–265. 2

[PP00] POLTHIER K., PREUSS E.: Variational approach to vector field decomposition. In *EG / IEEE TCVG Symposium on Visualization* (2000), pp. 147–155. 3

[SK97] SHEN H.-W., KAO D. L.: UFLIC: A line integral convolution algorithm for visualizing unsteady flows. In *IEEE Visualization* (1997), pp. 317–323. 2

[Sun03] SUNDQUIST A.: Dynamic line integral convolution for visualizing streamline evolution. *IEEE Transactions on Visualization and Computer Graphics 9*, 2 (2003), 273–282. 2

[TL05] TADIN D., LAPPIN J. S.: Optimal size for perceiving motion decreases with contrast. *Vision Research 45* (2005), 2059–2064. 3

[vW02] VAN WIJK J. J.: Image based flow visualization. *ACM Transactions on Graphics (ACM SIGGRAPH 2002) 21*, 3 (2002), 745–754. 2

[vW03] VAN WIJK J. J.: Image based flow visualization for curved surfaces. In *IEEE Visualization* (2003), pp. 123–130. 2, 5

[War04] WARE C.: *Information Visualization: Perception for Design*, 2nd ed. Morgan Kaufmann, San Francisco, 2004. 2

[WE04] WEISKOPF D., ERTL T.: A hybrid physical/device-space approach for spatio-temporally coherent interactive texture advection on curved surfaces. In *Graphics Interface* (2004), pp. 263–270. 2, 5

[WE05] WEISKOPF D., ERLEBACHER G.: Overview of flow visualization. In *The Visualization Handbook*, Hansen C. D., Johnson C. R., (Eds.). Elsevier, Amsterdam, 2005, pp. 261–278. 2

[WEE03] WEISKOPF D., ERLEBACHER G., ERTL T.: A texture-based framework for spacetime-coherent visualization of time-dependent vector fields. In *IEEE Visualization* (2003), pp. 107–114. 2, 4, 5

[Wei04] WEISKOPF D.: On the role of color in the perception of motion in animated visualizations. In *IEEE Visualization* (2004), pp. 305–312. 2

[Wei07] WEISKOPF D.: Iterative twofold line integral convolution for texture-based vector field visualization. In *Mathematical Foundations of Scientific Visualization, Computer Graphics, and Massive Data Exploration*, Möller T., Hamann B., Russell R., (Eds.). Springer, 2007. Preprint available on http://www.cs.sfu.ca/~weiskopf/publications. 5, 7

[WT95] WATSON A. B., TURANO K.: The optimal motion stimulus. *Vision Research 35* (1995), 325–336. 3