Applications of Texture-Based Flow Visualization

Robert S. Laramee*
Department of Computer Science
Swansea University
Wales, UK

Gordon Erlebacher†
School of Computational Science & Information Technology
Florida State University
Tallahassee, FL

Christoph Garth‡
The Institute for Data Analysis and Visualization
University of California, Davis

Tobias Schafhitzel§
Visualization and Interactive Systems Group
University of Stuttgart
Stuttgart, Germany

Holger Theisel¶
Department of Simulation and Graphics
University of Magdeburg, Germany

Xavier Tricoche‖
Scientific Computing and Imaging Institute (SCI)
University of Utah
Salt Lake City, UT

Tino Weinkauf**
Konrad-Zuse-Zentrum für Informationstechnik Berlin
Division of Scientific Computing
Department of Visualization
Berlin-Dahlem, Germany

Daniel Weiskopf††
Visualization Research Center, Universität Stuttgart
Visualization and Interactive Systems Group
University of Stuttgart, Germany

ABSTRACT

Flow visualization is a classic sub-field of scientific visualization. The task of flow visualization algorithms is to depict vector data, i.e., data with magnitude *and* direction. An important category of flow visualization techniques makes use of textures in order to convey the properties of a vector field. Recently, several research advances have been made in this special category, of dense, texture-based techniques. We present the application of the most recent texture-based techniques to real world data from oceanography and meteorology, computational fluid dynamics (CFD) in the automotive industry, and medicine. We describe the motivations for using texture-based algorithms as well as the important recent advancements required for their successful incorporation into industry grade software. Our goal is to appeal to practitioners in the field interested in learning how these recent techniques can help them gain insight into problems that engineers and other professionals may be faced with on a daily basis. Many of these applications have only recently become possible.

CR Categories: I.3.3 [Computer Graphics]: Picture/Image Generation; I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Color, shading, shadowing, and texture

Keywords: flow visualization, vector field visualization, texture, computational fluid dynamics (CFD), meteorology, oceanography, in-cylinder flow, visualization systems, engine simulation, medical application

*e-mail: r.s.laramee@swansea.ac.uk
†e-mail: erlebach@csit.fsu.edu
‡e-mail: cgarth@ucdavis.edu
§schafhitzel@vis.uni-stuttgart.de
¶e-mail: theisel@isg.cs.uni-magdeburg.de
‖e-mail: tricoche@sci.utah.edu
**e-mail: weinkauf@zib.de
††e-mail: weiskopf@visus.uni-stuttgart.de

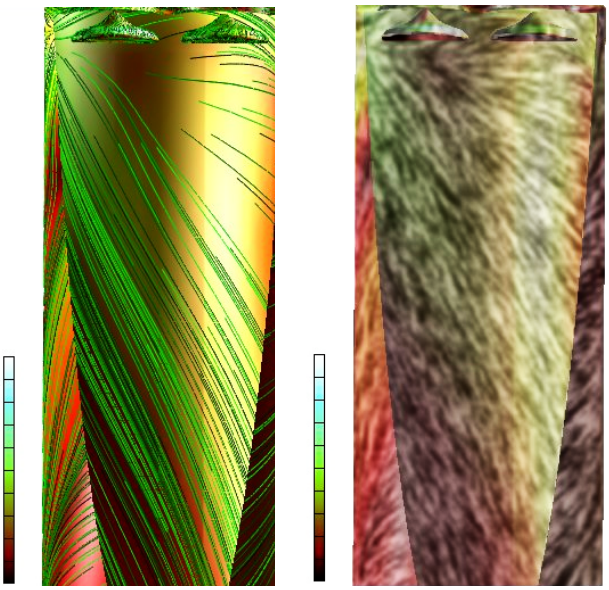

Figure 1: The visualization of swirl motion at the inner boundary surface of a combustion chamber from an automotive engine cylinder: (left) geometric flow visualization using streamlines, and (right) texture-based flow visualization, both being used in combination with velocity magnitude color-map.

1 INTRODUCTION

Flow visualization, a well established topic in scientific visualization is concerned with the task of showing vector quantities, i.e., magnitude *and* direction, with a special emphasis on direction. This can be contrasted with a wider range of visualization techniques which focus on conveying scalar quantities, e.g., isosurface extraction. An important subset of flow visualization techniques uses texture to convey the characteristics of a vector field. The result is a dense visualization with the special property of providing complete

coverage of the vector field. See the right image in Figure 1 for an example. This is in contrast to another more traditional class of approaches, namely geometric flow visualization techniques, that generate objects whose geometry reflects the characteristics of the vector field. A common example of a geometric technique is the use of streamlines, curves that are everywhere tangent to the vector field. Figure 1 illustrates geometric and texture-based approaches side-by-side, applied to the same in-cylinder flow of an engine simulation which we describe in more detail in Section 5.

The first texture-based flow visualization techniques were introduced in the early 1990s. They are Spot Noise [2, 19] and Line Integral Convolution (LIC) [1, 17]. Texture-based visualizations avoid the seeding problem associated with geometric techniques. The seeding problem is, namely, where to place streamlines such that important features of the vector field are not overlooked [16]. Although the early texture-based visualization approaches do not suffer from the seeding problem, their introduction brought new challenges with them.

The main problem that the texture-based flow visualization techniques brought with them was the expensive computational costs. Essentially, computing the resulting imagery took long enough that it prevented them from being attractive to a wide range of applications. Long computation time presented problems applying texture-based techniques to vector fields of higher dimensionality, e.g., unsteady flow, vector fields on surfaces, and in 3D. However, in the last few years we have witnessed the resolution of several of these problems. With the resolution of these problems has come their employment to new applications, several of which were not previously possible. The algorithms in sections that follow were chosen because they overcome these aforementioned problems.

In this article, we discuss recent developments and novel applications of texture-based flow visualization algorithms including:

- Necessary pre-requisites and user requirements in order for this class of techniques to be incorporated into a wider range of applications (Section 2)
- Some background and key developments that have taken place within the last few years (Section 3)
- Important characteristics of texture-based algorithms, namely, characteristics that both relate and distinguish these methods from others (Sections 3 and 4)
- The application of a selection of recent techniques to real world problems. This includes the texture-based techniques used to visualize data from multiple domains including oceanography, meteorology (Section 4), computation fluid dynamics simulations in the automotive industry (Section 5), and a novel medical application (Section 6).
- A description of the insight that the use of these techniques has brought with them outlining both the advantages and disadvantages of the approaches.

Our description of the applications goes into more detail than previous literature. We also offer our thoughts on future directions and unsolved problems related to this line of research in Section 7.

2 Requirements for a Wide Range of Applications

Here we describe some of the necessary requirements to make texture-based flow visualization techniques appealing to a wide range of applications.

The first aspect to consider for scientific visualization applications is the user. The users are often experts, such as mechanical engineers, who are already familiar to some extent with the data they are investigating. Typical users undergo training and may have many years of experience.

In order for flow visualization methods to be truly appealing to the engineers (or practitioners from other fields), a few goals have to be addressed:

- The software should support interactivity in order to explore the data and possibly introduce expert knowledge. Interactivity is crucial for virtual exploration by camera control, i.e., for understanding three-dimensional structure. Ideally, real-time rendering is supported. Closely related to this is for the application to support a fast user response time.
- Support for large data sets: CFD data generated from flow simulations results in very large data sets, often on the gigabyte scale.
- Related to the previous point is support for unstructured, adaptive resolution grids. This is especially important in CFD because simulation grids are usually unstructured and use higher resolution in areas with greater importance.

Support for interaction and real-time frame rates has come with the introduction of new algorithms that make intelligent use of graphics card hardware. This is especially true for the case of the graphics processing unit (GPU), which has evolved rapidly since around 1999–a topic we return to in Sections 3 and 4. Support for large data sets with unstructured, adaptive resolution grids is demonstrated in Section 5. Perceptual issues inherent to 3D visualization such as occlusion, visual complexity, and depth perception, become especially problematic when using texture-based flow visualization. These issues are discussed in Section 6.

3 Background and Key Developments

Before presenting the applications, we outline some basic principles common to these texture-based algorithms and terminology. Alongside, we mention some of the key developments in the field that have enabled recent progress. More on these and related topics can be found in related literature [3, 10].

The texture-based algorithms we describe take, as input, a vector field and a gray-scale noise texture. The gray-scale texture is then smeared in the direction of the given vector field. In general, a *convolution filter*.is used to correlate texels such that the texture reflects the properties of the vector field. To convolve texture values results in streamwise correlation between them. From a mathematical viewpoint, convolution can be expressed as the integral along a curve. In this case, the correlation is computed such that the vector field is represented. More specifically, given a streamline σ, convolution consists of calculating the intensity I for a pixel located at $\mathbf{x}_0 = \sigma(s_0)$ by:

$$I(\mathbf{x}_0) = \int_{s_0-L/2}^{s_0+L/2} k(s-s_0)T(\sigma(s))\,ds \qquad (1)$$

where T stands for an input noise texture, k denotes the filter kernel, s is an arc length used to parameterize the streamline curve, and L represents the filter kernel length [10, 17]. The result of this convolution is streaks in the texture in the direction of the local flow field. This computation is referred to as Line Integral Convolution (LIC) [1, 17]. It is this convolution (also known as filtering) process which consumed so much computation time with earlier texture-based algorithms and produced static imagery.

Recent research efforts have resulted in fast convolution processes which allow animation or *advection* of the texture properties in the direction of the flow at several frames per second [7, 8, 20, 24]. Advection was first introduced by Max et

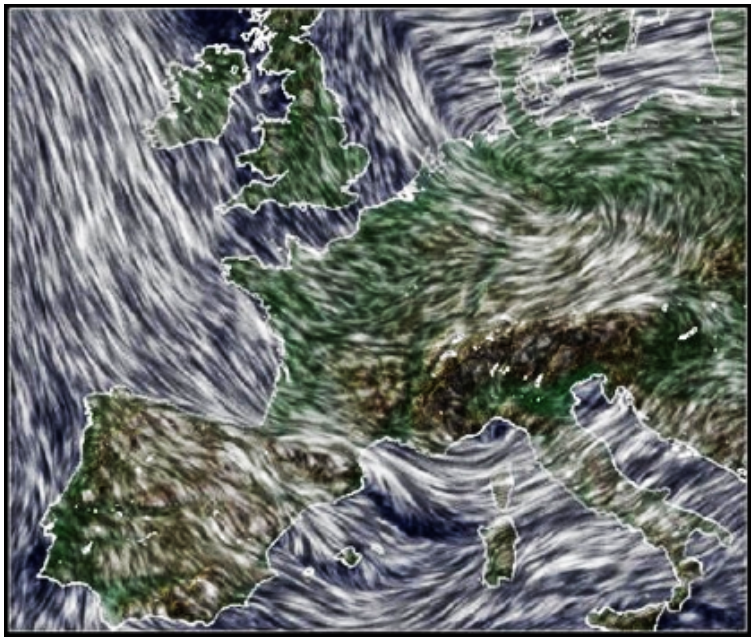

Figure 2: The time-dependent weather patterns over Europe. This visualization is the result of the Lagrangian-Eulerian Advection (LEA) algorithm [8]. (`http://srnwp.cscs.ch/Gallery/texture_loop.html`) Data courtesy of the Swiss Center for Scientific Computing.

al. [14, 15]. Conceptually LIC and texture-advection are similar in that they both convolve a noise texture to visualize the characteristics of a vector field. However, texture advection quickly brings the texture properties into motion, thus making it more suitable for the depiction of unsteady flow. The animation resulting from texture advection illustrates the downstream direction of the flow. Also, texture-advection algorithms may visualize vector field magnitude by the amount of smearing (in a still image) or the rate of motion for local texture properties. More smearing and faster motion reflect faster flow. For the majority of LIC algorithms, the amount of smearing is uniform throughout the domain.

The development of efficient texture advection was central in expanding the number of applications to which these algorithms could be applied. A typical advection computation is computed as follows: Given a position, $\mathbf{x}_0(i,j) = (i,j)$ of each particle in a 2D flow, (backward) integration over a time interval h determines its position at a previous time step

$$\mathbf{x}_{-h}(i,j) = \mathbf{x}_0(i,j) + \int_{\tau=0}^{h} \mathbf{v}_{-\tau}(\mathbf{x}_{-\tau}(i,j))d\tau \qquad (2)$$

where h is the integration step, $\mathbf{x}_{-\tau}(i,j)$ represents intermediary positions along the pathline passing through $\mathbf{x}_0(i,j)$, and $\mathbf{v}_\tau$ is the vector field at time τ. For more detail on this topic, please see previous literature [10, 16, 24]. A sample advection result is the right image in Figure 1.

As soon as the textures are animated to reflect the motion of the flow, another problem is introduced, namely, the texture properties may simply flow, or be transported out of the domain, leaving some or all of the domain uncovered. Thus we must introduce some sort of refresh process, usually called *noise injection and blending* in the case that noise textures are used. The noise injection and blending process introduces new noise or texture properties into an animation such that complete coverage of the domain is maintained over time.

Another key development has been the rapid increase in hardware's processing speed. Growth in graphics card processing power continues to excel at a rate faster than that of CPUs. This, coupled with the new features of programmable graphics hardware has also been a key contribution to some of the recent development in this field.

4 Time-Dependent Planar Flow–Oceanography and Meteorology

This section discusses applications of texture-based flow visualization to planar domains. First, we start off with a description of the spatio-temporal characteristics of the ideal dense, texture-based algorithm, followed by a discussion of dye injection. Dye injection may be used to represent a vector field with increased spatial and temporal coherence but with a trade-off for complete coverage.

4.1 Spatial and Temporal Coherence

Two key notions to a successful visualization of time-dependent flow fields on two-dimensional surfaces are temporal and spatial correlation. Temporal correlation ensures that a particle (often represented by one or more texture elements, i.e., texels) maintains its identity (often represented by color or color distribution) over time. Temporal correlation may enable the user to track individual flow elements (modeled by particles) in time. An individual particle is difficult to track, although patterns formed by a collection of particles can be identified and followed in time.

The spatial patterns identified in each individual image frame are the result of spatial correlation between adjacent particles. Thus the particles that form a gray-scale streak on a black background form a pattern.

Most texture-advection algorithms are constructed from three basic steps:

1. Initialize one or more textures with particles and their properties. Sometimes a single texture element represents a single particle.

2. Update the position of these particles in space and time (called particle advection) according to Equation 2, and

3. Post-process the resulting image to enhance the visual representation.

While these three steps are similar to modern methods, how they are actually implemented in practice can greatly affect the quality of the final visualization.

The quality of the time-dependent images depends strongly on the characteristics of temporal and spatial correlation built into the algorithm. Many time-dependent algorithms were constructed based on intuition, with attention dedicated to correlation issues *a posteriori*. Erlebacher et al. [3] have recently proposed a rather general framework that treats the temporal and spatial correlations explicitly instead of having them be an intrinsic part of some algorithm. In this approach, correlation is to be interpreted as smoothing along a curve, the curve being defined in a two or three dimensional vector field domain.

Figure 2 shows a dense, time-dependent visualization from the field of meteorology. The result was generated using the Lagrangian-Eulerian Advection (LEA) algorithm [8]. The other algorithms employed in this article could also have been used. Depicted are the weather patterns over Europe, i.e., the winds over Europe. Regions of higher velocity magnitude are characterized by brighter streaks whereas the textures become more transparent in areas of slower wind. The application of LEA to oceanography is also demonstrated [8].

For readers that are interested in a hands-on experience with a live, industry-grade texture-based flow visualization application, we recommend looking at the web site for the Swiss National Supercomputing Center (CSCS, `http://www.cscs.ch/`). The link, "`http://srnwp.cscs.ch/Gallery/texture_loop.html`", directs the viewer to the online application. The viewer is then shown a Short Range Numerical Weather Prediction (SRNWP). The visualization provides insight with respect to patterns of wind that are predicted to cover Europe within the next few days. The data, updated daily, is collected from the last 72 hours of wind forecast. The LEA algorithm then represents the time-dependent weather data as a collection of dense, animated textures. Note that the texture-based visualization has the advantage of covering the whole spatial domain. A velocity mask has been used here that makes areas of low velocity magnitude more transparent. The user can control the animation just as one does with a traditional media player. With time-dependent data, it is interesting to note the temporal sampling frequency used because it may vary by several orders of magnitude. For example, the data in this application is sampled at a rate of approximately once per hour. The temporal sampling frequency of the application we describe in Section 5 is multiple orders of magnitude higher.

4.2 Dye Advection

Dye advection is complementary to the dense visualization approaches discussed above. Dense techniques seek to inject the flow with a uniform coverage of particles and at the same time convey the direction and speed each particle is moving. On the other hand, dye techniques are the numerical implementation of experimental techniques used since the 1960s to help elucidate the flow patterns in the vicinity of bodies in fast moving air or immersed in liquids. A colored liquid can be released into the flow. This liquid is then transported by the underlying flow; the surface covered by the dye provides valuable qualitative and quantitative information to experimental researchers. There is great flexibility in the use of dye as a diagnostic tool. At its most basic, dye is continuously released from a single point. The path followed by the dye is called a streakline.

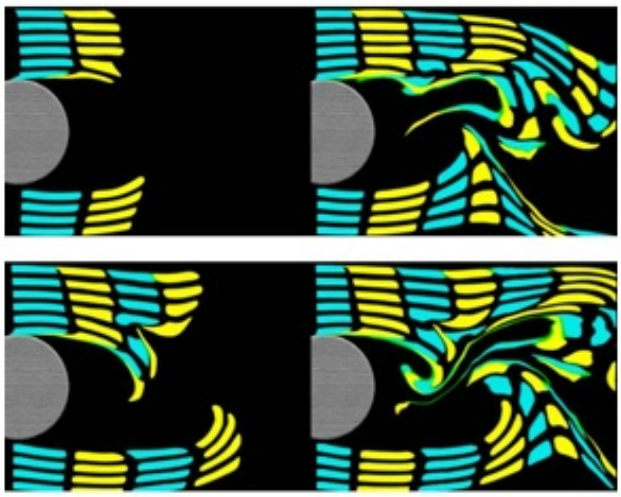

Figure 3: Time-varying, planar flow past a cylinder visualized with dye advection. In this case, the dye sources result in streaklines. The color of the streaklines is toggled over regular, periodic intervals. Data courtesy of X. Ding from Florida State University.

If multiple fixed points release dye, the patterns is a rake of streaklines (see Figure 3 for an example). By definition, a streakline is a curve formed by joining the particles released from a fixed spatial location. If dye is released from all points on a line segment, dye will be transported along all streamlines (or pathlines in the time-dependent case) passing through the line segment. If the injection is toggled periodically, timelines result. Such periodic toggling of dye, or pulsing is also depicted in Figure 3 as the dye source colors are toggled between yellow and light blue. The underlying fluid particles are transported as a line that distorts in space. The motion of a timeline is representative of the motion of a collection of particles. It is possible to blend two techniques: (1) continuous dye injection from a collection of points and (2) a periodic injection from a continuous line, by thickening the spatial extent of the dye injection points, and lengthening the duration of the pulse. An example of this technique is illustrated in Figure 3, which shows flow around a cylinder. The data was obtained from a laboratory experiment by X. Ding of Florida State University.

5 Flow Visualization on Surfaces–Computational Fluid Dynamics for the Automotive Industry

Automobile manufacturers use CFD simulation software not only to evaluate and optimize the design of engine components but also to investigate causes of engine failure. The overall evaluation process can be broken down into three principle stages: (1) *modelling*: starting with a model generated by computer aided design (CAD) software, a 3D unstructured mesh is generated consisting of volumetric cells, (2) *simulation*: given the 3D mesh and a set of initial conditions, a simulation of flow through the model is computed, and (3) *visualization*: the results of the simulation are explored, analyzed, and presented with a variety of visualization tools. Mechanical engineers investigate the results looking for things like areas of extreme pressure or temperature, symmetries in the flow, stagnant regions, and turbulent flow. With respect to turbulent flow engineers look for anomalies in the turbulent flow patters such as backscatter and vortex shedding. They also compare the simulation results with real measured results. Figure 4 shows modern measuring equipment used to monitor the behavior of a real engine.

Figure 4: Monitoring equipment is hooked up to a running engine and catalytic converters: (1) the running engine, (2) connected measurement equipment, (3) the exhaust system, and (4) equipment measuring and monitoring the exhaust system. This measured data is compared to CFD simulation data. Photo courtesy of AVL.

The process is iterative. After visualizing the simulation results, engineers then tune design parameters with the goal of improving the movement of flow through the geometry. This iterative approach should be faster with software tools as opposed to constructing and tuning real heavy-weight objects. Here, we illustrate this idea with two concrete examples from engine design and simulation.

For many of the automotive components that undergo evaluation, there is an ideal pattern of flow the engineers attempt to create. Figure 5 shows ideal patterns of fluid motion in diesel and gas engine cylinders. For the case of in-cylinder flow, we can distinguish between two types of motion: swirl flow commonly found in diesel engines and tumble flow commonly found in gas engines. In both cases, rotational motion occurs about a central axis, though the position of the respective axis is different. The swirl motion axis is coincident with the cylinder axis and is static. In the case of tumble flow, the rotation axis is perpendicular to the cylinder axis and is time dependent, thus making the ideal tumble motion more complex and difficult to achieve.

We mentioned in Section 4 that the temporal sampling frequency can vary by several orders of magnitude depending on the application. In the case of tumble motion (Figure 5, right) the volume inside the cylinder is time-dependent. The rate at which the cylinder head rotates, the lower boundary of the cylinder (not shown), is described by an equation well known to the mechanical engineers using this application: $\theta = 2\pi\omega t$, where θ is the crank angle of the engine and ω is the engine revolution rate given in rotations per

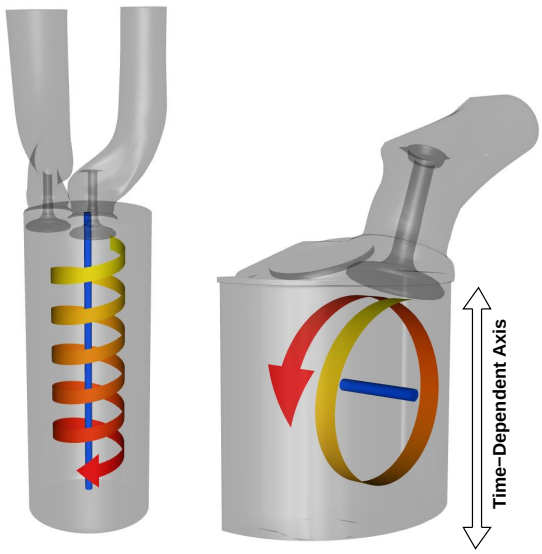

Figure 5: (left) The swirling motion of flow in the combustion chamber of a diesel engine. *Swirl* is used to describe circulation about the cylinder axis. The intake ports at the top provide the tangential component of the flow necessary for swirl. The simulation model consists of 776,000 unstructured, adaptive resolution grid cells. (right) Some gas engine components require a *tumble* motion flow pattern in order to mix fluid with oxygen. Tumble flow circulates around a time-dependent axis perpendicular to the (stationary) cylinder axis, orthogonal to swirl flow. The gas engine model has a time-dependent volume.

minute (RPM). An $\omega = 2{,}000$ RPM engine, like the one described here, rotates at 33.33 Hz. Substituting our values and solving for t:

$$t = \frac{2\pi \; radians/rotation}{2\pi \; (33.33 \; rotations/second)} \tag{3}$$

we learn that the engine completes a revolution in only 0.03 seconds. If the simulation data is sampled once at each crank angle, then the temporal sampling frequency is roughly 12,000 times per second–several orders of magnitude higher than that of the meteorology application described in Section 4.

In order to generate swirl or tumble motion, fluid enters the combustion chamber from the intake ports (top of Figure 5). The kinetic energy associated with this fluid motion is used to generate turbulence for mixing of fresh oxygen with evaporated fuel. Engineers seek an optimal level of turbulence to accommodate the best possible mixing of fuel and air. By stable we mean achieving the same conditions for each engine cycle. Ideally, enough turbulent mixing is generated such that 100% of the fuel is burned. From the point of view of the mechanical engineers, the ideal flow pattern leads to beneficial conditions including: improved mixing of fuel and oxygen, a decrease in fuel consumption, and lower emissions. However, too much swirl (or tumble) can displace the flame used to ignite the fuel, cause irregular flame propagation, or result in less fuel combustion.

We used recently developed texture-based flow visualization algorithms, namely Image Space Advection (ISA) and Image Based Flow Visualization for Curved Surfaces (IBFVS) [12] in order to investigate the simulation results from these two in-cylinder motion cases. ISA and IBFVS have several properties that make them suitable for the type of application we present here including the ability to:

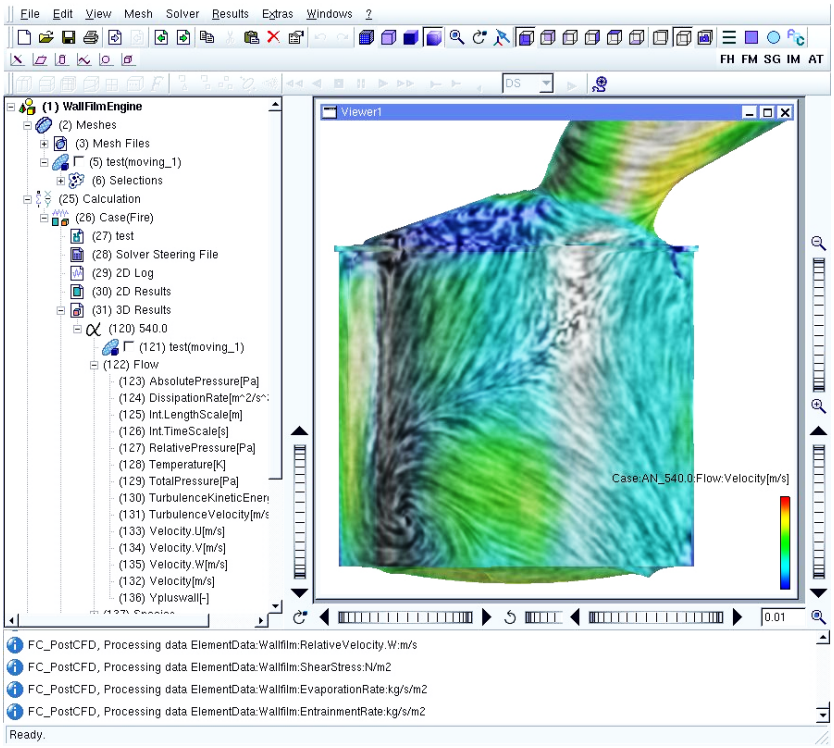

Figure 6: A recent texture-based flow visualization algorithm (ISA) used in a commercial application to visualize the boundary flow at the surface of a piston cylinder. In this case, engineers are trying to re-create a tumble motion pattern of flow. Unlike other visualization techniques, like hedgehog plots or streamlines, texture-based methods provide complete coverage of the vector field at the boundary surface. Hence, no important features of the flow field are left out of the resulting visualization.

- generate a dense representation of unsteady flow on surfaces,
- visualize flow on complex surfaces composed of polygons whose number is on the order of 250,000 or more,
- handle arbitrary, complex meshes without relying on a parametrization,
- support user-interaction such as rotation, translation, and zooming while maintaining a constant, high spatial resolution,
- deliver high performance, i.e., several frames per second, and
- visualize flow on dynamic meshes with time-dependent geometry and topology.

These characteristics are essential to make texture-based methods applicable to real-world CFD data sets from the automotive industry. We mention this because these properties are directly related to the requirements outlined in Section 2. A comparison of the ISA and IBFVS algorithms is given by Laramee et al [12]. For an overview and comparison of texture-based algorithms in general, see Laramee et al [10].

They key behind the performance of these algorithms lies in simplifying the texture-mapping process. Traditional visualization of boundary flow using textures first maps one or more 2D textures to a surface geometry defined in 3D space. The textured geometry is then rendered to image space. Here, we alter this classic order of operations. First, we project the surface geometry and its associated vector field to image space and then apply texturing.

While, in general, we have not seen the proliferation of texture-based methods into commercial applications, recent advances in this line of research have lead some of them to be incorporated into real-world industry-grade software. Figure 6 illustrates ISA being used in a commercial application used daily by mechanical engineers to evaluate CFD simulation results [1]. This image visualizes the behavior of tumble motion at the boundary surface of in-cylinder flow. The simulation results here adhere to the 'no-slip' boundary condition, i.e., the velocity is defined as zero at the boundary surface. What we mean by "at the boundary surface" is actually just inside the boundary surface. We have extrapolated the vector field to the surface for visualization purposes. For a more comprehensive discussion on this topic, including how to include the surface's normal component of the vector field into the visualization, see Laramee et al [9, 11].

The visualization is fast, generating real-time animation. The viewer's eyes are immediately drawn to the vortex in the lower left hand corner. This vortex turns out to be the axis of tumble motion which is off-center. This can be verified by looking inside the volume. Similarly, the right image of Figure 1 visualizes the behavior of swirl motion characteristic of in-cylinder flow in a diesel engine. Engineers often start their analysis with an overview of the simulation results by looking at boundary behavior. In this image, the rear boundary wall is revealed through the use of a clipping plane to cut away the front boundary wall of the cylinder. The same goal could be reached by rotating the geometry 180 degrees. It is evident that the flow exhibits general swirl-rotational flow behavior at the inner boundary. Various methods are then used to examine the flow characteristics inside the volume. We have also implemented ISA to be

[1] For supplementary material including high resolution images and animations, please visit:
`http://cs.swan.ac.uk/~csbob/research/application/`

Figure 7: Texture-based flow visualization on a curved surface for an automotive CFD simulation.

used on slices, isosurfaces, and analytic surfaces inside the volume in this commercial application.

Another typical and related application of CFD simulation in the automotive industry is to evaluate the air flow around a moving car. Such simulations are used to avoid traditional wind tunnel experiments in order to save costs and achieve a faster development cycle. For this application, surface-based visualization [21] is useful because the air flow in the vicinity of the car surface plays an important role in understanding the structure of vortices and other important flow features. Figure 7 shows an example of a dense representation of air flow on the curved surface of a car. Here, the domain of the data set is reduced from the full 3D unstructured grid to a triangular mesh representation in a preprocessing step. A tangential vector field is constructed from the original 3D simulation data by projecting the vector field onto the surface. After such a preprocessing, an interactive visualization has become possible even for data sets in the range of more than a million triangles.

6 3D Flow Visualization of Experimental and Medical Data

The previous examples show that a surface-based visualization can be successful in visualizing 3D flow data. For other applications, though, a complete representation of the 3D simulation data may become necessary. This is especially true for data sets that show a complex behavior without any restriction to spatial symmetries. Figure 8 shows an example of flow with several 3D swirling features.

Unfortunately, a texture-based visualization of 3D flow is quite challenging: First, computational costs show a cubic increase with increasing grid resolution—as opposed to only a quadratic behavior for surface-oriented visualization. Second, the goal of a dense representation cannot be directly adopted in a volumetric data set because of severe occlusion problems and perceptual issues.

The first issue–high computational costs–can be addressed by using several strategies. One approach is to employ efficient data structures. Therefore, most of the interactive visualization systems rely on data given on a uniform grid, which provides most efficient interpolation and cell location schemes. Since data is often computed on curvilinear or even unstructured grids, resampling methods are employed in a preprocessing step to generate a uniform visualization grid. Another strategy is to find a good compromise between cost-intensive flexibility and efficiency through extensive preprocessing. For example, the Chameleon system [13] finds such a compromise by pre-computing particle traces and storing them in rasterized form on a uniform grid. For interactive visualization, volume rendering is applied to display these particle traces. A third

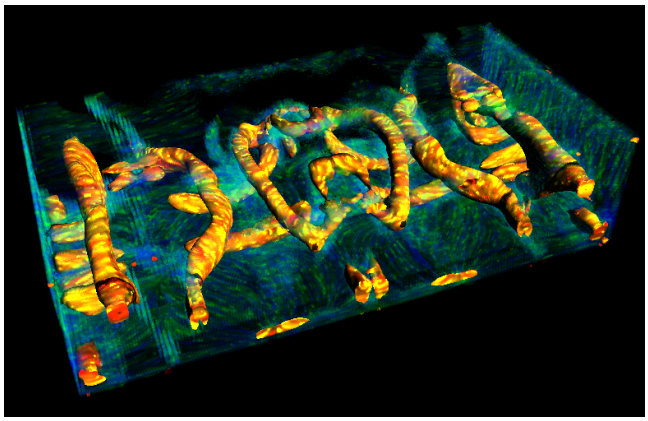

Figure 8: 3D visualization of Benard convection. Benard convection leads to an exchange of heat between a hot bottom surface and a colder top surface.

strategy makes use of graphics hardware to accelerate flow visualization based on 3D texture advection [18, 22] and texture-based volume rendering. In this way, an on-the-fly computation and visualization of particle traces is possible. These challenges are also addressed by Helgeland and Elboth [5].

The other main issue is to find a visual representation that avoids occlusion problems and supports good spatial perception. An important strategy is to emphasize features of the 3D flow and suppress the visualization of uninteresting flow regions. For example, the visualization in Figure 8 highlights vortex behavior by opaque shades of yellow and red color, while less interesting parts are rendered almost transparent in shades of blue and green. This approach is an example for a focus-and-context visualization that allows the user to focus on a critical aspect of the dataset in the spatial context of the surrounding flow. Three-dimensional texture advection has the advantage of avoiding any significant pre-computation; therefore, focus areas can be selected interactively and rendered on-the-fly. In general, the focus-and-context approach can be controlled by a 3D importance function that describes the space-variant opacities for volume rendering.

In addition to occlusion, other perceptual issues have to be considered for a 3D visualization. For example, it becomes much more difficult to recognize the orientation of a 3D streamline than that of 2D a streamline. By including further perceptual cues, the visualization of 3D structures can be enhanced. Figure 9 shows an example of 3D texture advection combined with Phong illumination to improve the perception of 3D orientation. Such a shading approach can be combined with other methods such as halo lines or a selection of colors that highlight the continuity of partly occluded lines. Perceptual issues when visualizing 3D vector fields have been discussed in previous literature [4, 5, 6, 23]

We have applied 3D texture advection to a novel medical application. The goal of this medical application is to investigate how to include information about organ motion in radiotherapy treatment. Radiotherapy is a very widely used cancer therapy. One approach is to radiate a tumor with high energy particles while preserving surrounding healthy tissues. By gaining a better understanding of lung motion, doctors can increase their chances of radiating only the cancerous lung tissue. Visualization can help provide insight into the characteristics of lung motion. Figure 9 shows a 3D texture-based depiction of lung motion during a respiratory cycle. In this case, texture properties are advected in the direction of lung motion. A vector field is created by capturing snapshots (or samples) of the lungs at different times between a deep inhale and a full exhale. These snapshots can then be compared to a registration image, i.e.,

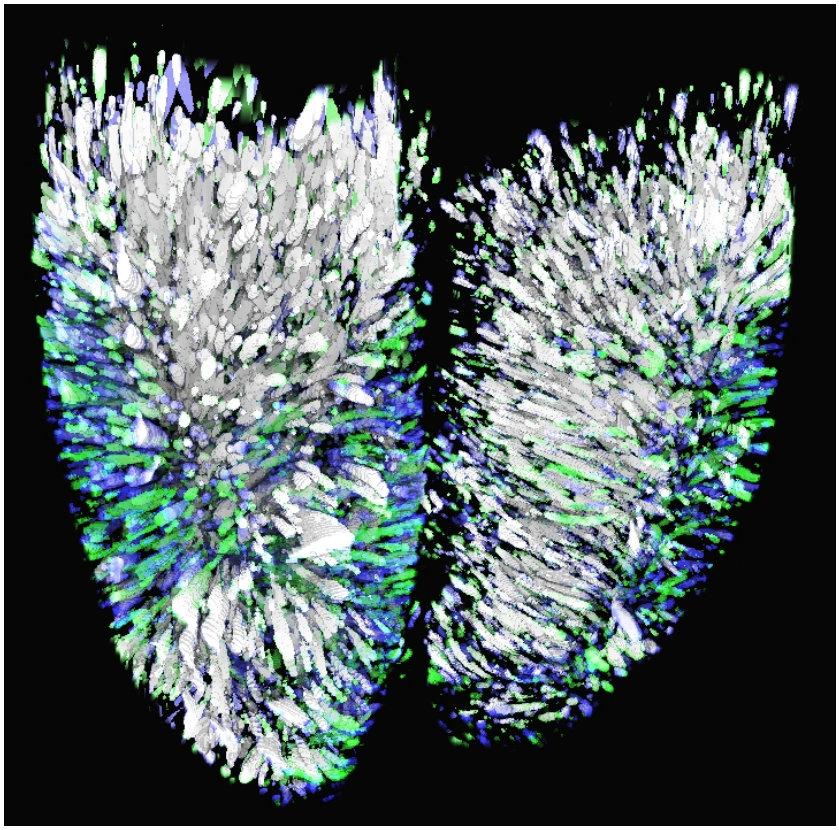

Figure 9: Phong illumination and 3D texture advection are used to visualize the characteristics of lung motion during an inhale-exhale cycle.

a snapshot of the lungs in a relaxed, neutral state. The difference between the images can be used to create a time-dependent vector field that describes the motion of the lungs. We used a mask as a transfer function for volume rendering. The CT values corresponding to known lung tissue are used to identify those voxels with non-zero opacity. The resulting volume rendering then provides insight into both the lung structure and its dynamic nature during a respiratory cycle. This type of analysis is used in order to perform treatment planning. From a sampling point of view, the rate of temporal sampling is on the order of one second–between the temporal sampling rates of our meteorological and automotive applications.

7 Thoughts on the Future

The last five years have witnessed a surge in the field of research in dense texture-based flow visualization techniques. The advances in research have given rise to several applications ranging from meteorology, oceanography, engine simulation, and visualization of experimental data, applications that have only recently become feasible. We have even seen the proliferation of these algorithms into industry-grade software, two of which are described in this article (Sections 4 and 5). In the future, we expect this trend to continue, i.e., the incorporation of dense texture-based flow visualization algorithms into even more commercial and non-commercial software. This is especially true for applications with planar flow and flow on surfaces.

The future of texture-based flow visualization as it applies to true 3D flow is still uncertain. Problems still remain due to the expensive computation time, especially in the case of time-dependent 3D data, as well as perceptual issues associated with visualization in 3D. This is because one of the very characteristic benefits that texture-based methods provide, namely complete coverage of the vector field, may become a disadvantage in three dimensions. Some researchers believe we may never solve these perceptual problems. These problems provide strong motivation for feature-based flow visualization techniques–algorithms that extract a use-defined subset of the vector field before rendering the result.

8 Acknowledgements

The authors thank all those who have contributed to this research including AVL (www.avl.com) and the Austrian research program Kplus (`www.kplus.at`). The CFD simulation data sets are courtesy of AVL, Florida State University, and the Swiss Center for Scientific Computing, amongst others. The online application was implemented by Bruno Jobard during his stay at the Swiss Center for Scientific Computing. He is now an assistant professor at the University of Pau in France.

References

[1] B. Cabral and L. C. Leedom. Imaging Vector Fields Using Line Integral Convolution. In *Poceedings of ACM SIGGRAPH 1993*, Annual Conference Series, pages 263–272, 1993.

[2] W.C. de Leeuw and J.J. van Wijk. Enhanced Spot Noise for Vector Field Visualization. In *Proceedings IEEE Visualization '95*, pages 233–239. IEEE Computer Society, October 1995.

[3] G. Erlebacher, B. Jobard, and D. Weiskopf. Flow textures: High-resolution flow visualization. In C. R. Johnson and C. D. Hansen, editors, *The Visualization Handbook*, pages 279–293. Elsevier, 2005.

[4] A. Helgeland and O. Andreassen. Visualization of Vector Fields Using Seed LIC and Volume Rendering. *IEEE Transactions on Visualization and Computer Graphics*, 10(6):673–682, 2004.

[5] A. Helgeland and T. Elboth. High-Quality and Interactive Animations of 3D Time-Varying Vector Fields. *IEEE Transactions on Visualization and Computer Graphics*, 12(6):1535–1546, December 2006.

[6] V. Interrante and C. Grosch. Strategies for Effectively Visualizing 3D Flow with Volume LIC. In *Proceedings IEEE Visualization '97*, pages 421–424, 1997.

[7] B. Jobard, G. Erlebacher, and M. Y. Hussaini. Lagrangian-Eulerian Advection for Unsteady Flow Visualization. In *Proceedings IEEE Visualization '01*, pages 53–60. IEEE Computer Society, October 2001.

[8] B. Jobard, G. Erlebacher, and Y. Hussaini. Lagrangian-Eulerian Advection of Noise and Dye Textures for Unsteady Flow Visualization. *IEEE Transactions on Visualization and Computer Graphics*, 8(3):211–222, 2002.

[9] R. S. Laramee, C. Garth, J. Schneider, and H. Hauser. Texture-Advection on Stream Surfaces: A Novel Hybrid Visualization Applied to CFD Results. In *Data Visualization, The Joint Eurographics-IEEE VGTC Symposium on Visualization (EuroVis 2006)*, pages 155–162,368. Eurographics Association, 2006.

[10] R. S. Laramee, H. Hauser, H. Doleisch, F. H. Post, B. Vrolijk, and D. Weiskopf. The State of the Art in Flow Visualization: Dense and Texture-Based Techniques. *Computer Graphics Forum*, 23(2):203–221, June 2004.

[11] R. S. Laramee, J. Schneider, and H. Hauser. Texture-Based Flow Visualization on Isosurfaces from Computational Fluid Dynamics. In *Data Visualization, The Joint Eurographics-IEEE TVCG Symposium on Visualization (VisSym '04)*, pages 85–90,342. Eurographics Association, 2004.

[12] R. S. Laramee, J. J. van Wijk, B. Jobard, and H. Hauser. ISA and IBFVS: Image Space Based Visualization of Flow on Surfaces. *IEEE Transactions on Visualization and Computer Graphics*, 10(6):637–648, November 2004.

[13] G. S. Li, U. Bordoloi, and H. W. Shen. Chameleon: An Interactive Texture-based Framework for Visualizing Three-dimensional Vector Fields. In *Proceedings IEEE Visualization '03*, pages 241–248. IEEE Computer Society, 2003.

[14] N. Max and B. Becker. Flow Visualization Using Moving Textures. In *Proceedings of the ICASW/LaRC Symposium on Visualizing Time-Varying Data*, pages 77–87, September 1995.

[15] N. Max, R. Crawfis, and D. Williams. Visualizing Wind Velocities by Advecting Cloud Textures. In *Proceedings IEEE Visualization '92*, pages 179–185, 1992.

[16] F. H. Post, B. Vrolijk, H. Hauser, R. S. Laramee, and H. Doleisch. Feature Extraction and Visualization of Flow Fields. In *Eurographics 2002 State-of-the-Art Reports*, pages 69–100, 2–6 September 2002.

[17] D. Stalling and H. Hege. Fast and Resolution Independent Line Integral Convolution. In *Proceedings of ACM SIGGRAPH 95*, Annual Conference Series, pages 249–256. ACM SIGGRAPH, ACM Press / ACM SIGGRAPH, 1995.

[18] A. Telea and J. J. van Wijk. 3D IBFV: Hardware-Accelerated 3D Flow Visualization. In *Proceedings IEEE Visualization '03*, pages 233–240. IEEE Computer Society, 2003.

[19] J. J. van Wijk. Spot noise-Texture Synthesis for Data Visualization. In Thomas W. Sederberg, editor, *Computer Graphics (Proceedings of ACM SIGGRAPH 91)*, volume 25, pages 309–318, 1991.

[20] J. J. van Wijk. Image Based Flow Visualization. *ACM Transactions on Graphics*, 21(3):745–754, 2002.

[21] D. Weiskopf and T. Ertl. A Hybrid Physical/Device-Space Approach for Spatio-Temporally Coherent Interactive Texture Advection on Curved Surfaces. In *Proceedings of Graphics Interface*, pages 263–270, 2004.

[22] D. Weiskopf, T. Schafhitzel, and T. Ertl. Real-Time Advection and Volumetric Illumination for the Visualization of 3D Unsteady Flow. In *Data Visualization, Proceedings of the 7th Joint EUROGRAPHICS–IEEE VGTG Symposium on Visualization (EuroVis 2005)*, pages 13–20, May 2005.

[23] D. Weiskopf, T. Schafhitzel, and T T. Ertl. Texture-based visualization of unsteady 3d flow by real-time advection and volumetric illumination. In *IEEE Transactions on Visualization and Computer Graphics*, volume 13, pages 569–582, 2007.

[24] D. Weiskopf, F. Schramm, G. Erlebacher, and T. Ertl. Particle and Texture Based Spatiotemporal Visualization of Time-Depedent Vector Fields. In *Proceedings IEEE Visualization '05*, pages 639–646, 2005.