# Geometric Flow Visualization Techniques for CFD Simulation Data

Robert S. Laramee and Helwig Hauser*
VRVis Research Center
Vienna, Austria

## Abstract

Visualization of CFD simulation data on unstructured, three-dimensional grids poses several challenges. The wide range of real-world data set sizes and the geometric versatility within individual, CFD simulation models present challenges to the engineers analyzing simulation results. Users also face perceptual problems such as occlusion, visual complexity, lack of directional cues, and lack of depth cues. We present a collection of geometric flow visualization techniques that address these challenges including oriented streamlines, streamlets, and a streamrunner tool. Two novel approaches are included: a real-time animated streamline technique and streamcomets. We place special emphasis on necessary measures required in order for geometric techniques to be applicable to real-world data sets.

**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 I.6.6 [Simulation and Modeling]: Simulation Output Analysis

**Keywords:** flow visualization, vector field visualization, streamlines, interaction, perception, CFD simulation data

## 1 Introduction

Demand for visualization solutions for CFD simulation data has grown rapidly in the last decade. This is due, in part, by the interest of manufactures in minimizing the time taken for their production cycle. This objective is realized with the use of software simulation tools to analyze design decisions before constructing real, heavy-weight objects.

At the VRVis research center we collaborate with AVL (`www.avl.com`) in order to provide flow visualization solutions for analysis of their CFD simulation result data. AVL's own engineers as well as engineers at industry affiliates use flow visualization software to analyze and evaluate the results of their automotive design and simulation on a daily basis. The analysis of an engineer includes tasks such as searching for areas of extreme pressure, looking for symmetries in the flow, searching for critical points, and comparing simulation results with measured, experimental results. One pervading message we hear consistently is: users are interested in more interactive control of the flow visualization results–a classic theme in the realm of scientific visualization [Hibbard and Santek 1989]. Engineers as well as users from other disciplines are interested in having a collection of user-options and parameters that allow them to fulfill their individual goals, whether their goals are exploration, analysis, or presentation. Interactive tools facilitate an iterative visual analysis and exploration process i.e., an environment in which the user is able to make rapid decisions and refinement based on visualization results.

*e-mail: {Laramee,Hauser}@VRVis.at

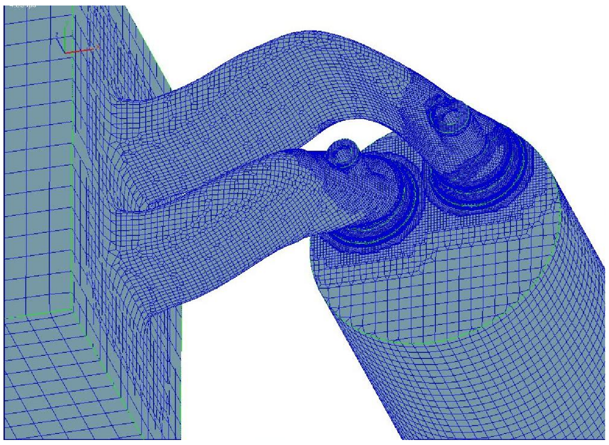
Figure 1: The CFD simulation grid of an intake port. This image illustrates the versatility of a typical, unstructured, CFD simulation grid.

AVL analyzes a large, varied collection of data sets ranging from small geometries such as small fluid conduits to mid-range size geometries such as cooling jackets, to large geometries such as automotive exteriors. The geometric sizes of these grids differ by *six* or more orders of magnitude as well as the sizes of the underlying polygons. Hence, the tools used to visualize the simulation results also need to span this range of sizes. We speculate that this difference will increase in the future.

### The Versatility of CFD Grids

Another reason the users request more interaction control over the visualization results is due to the fact that CFD meshes embrace a wide variety of components, features, and levels of resolution. To illustrate, we look at Figure 1 showing two intake ports– small valves in a car engine that allow air into the engine's cylinders. By looking at an overview, we observe what appear to be four adaptive levels of resolution: (1) for the flow source on the left and the combustion chamber on the lower, right, (2) another level of resolution for the connecting pipes in the middle, and two levels of resolution for the intake port components themselves.

When we zoom in (Figure 2) we find five adaptive levels of resolution used to evaluate the intake ports themselves: (a) two levels for the top of the ports, (b) approximately the same two levels of detail plus an added layer of finer resolution grid cells for the rings around the base of the ports. The facets in the flow source (on the left in Figure 1) are approximately 1000–2000 times larger than the finest resolution facets at the base of the intake ports. These grids are a daily experience in the industrial CFD community. Our goal is

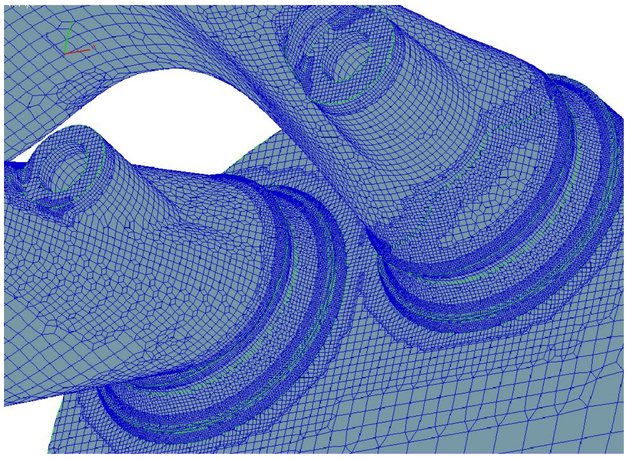

Figure 2: A close-up view of the intake ports in the same CFD simulation grid as shown in Figure 1. The mesh contains multiple, adaptive resolution levels of unstructured grid cells.

to provide flow visualization solutions that are equally as versatile and adaptive as the grids themselves.

### Perceptual Challenges

A large amount of flow visualization research literature addresses two-dimensional visualization techniques. This is partly because flow visualization on boundary surfaces and in 3D presents additional perceptual challenges such as occlusion, lack of directional cues, lack of depth cues, and visual complexity. Most of the CFD simulation grids at AVL are unstructured and three dimensional. Although engineers often use 2D slices through the 3D meshes during analysis, there is a strong interest in 3D and boundary surface visualization techniques that address the perceptual problems mentioned above. We also know that there is strong evidence to support the notion that users acquire a better understanding of 3D data sets using 3D visualization techniques as opposed to 2D visualization techniques [Ware and Franck 1996].

The rest of this paper is organized as follows: In Section 2 we discuss related research in flow visualization with an emphasis on geometric approaches. Section 3 describes our approach of using oriented streamlines and streamlets. Section 4 introduces a novel animated streamline technique. Section 5 outlines the streamrunner and streamcomet concepts and resulting implementations. Each tool is applied to real-world data sets from CFD. Finally, we conclude with some initial results and ideas for future work.

## 2 Related Work

Four different approaches are widely used in flow visualization [Post et al. 2002]:

*Direct flow visualization:* This category of techniques uses a translation that is as straightforward as possible for representing flow data in the resulting visualization. The result is an overall picture of the flow. Common approaches are drawing arrows or color coding velocity. Intuitive pictures can be provided, especially in the case of two dimensions.

*Geometric flow visualization:* These approaches often first integrate the flow data and use geometric objects in the resulting visualization. The objects have a geometry that reflects the properties of the flow. Examples include streamlines, streaklines, streamsurfaces, and timelines. Not all geometric objects are based on integration. Another useful geometric approach is generating isosurfaces, e.g., with respect to an isovalue of pressure or magnitude of velocity. A more thorough description of geometric techniques is presented by Post et al. [Post et al. 2002]

*Dense, texture-based flow visualization:* A texture is computed that is used to generate a dense representation of the flow. A notion of where the flow moves is incorporated through co-related texture values along the vector field. In most cases this effect is achieved through filtering of texels according to the local flow vector. Texture-based methods offer a dense representation of the flow with complete coverage of the vector field. Recent examples include Image Based Flow Visualization (IBFV) [van Wijk 2002] and Image Space Advection (ISA) [Laramee et al. 2004b], which can generate both Spot Noise [van Wijk 1991] and LIC-like [Cabral and Leedom 1993] imagery. We note that a full comparison of texture-based flow visualization techniques is beyond the scope of this paper [Laramee et al. 2004a].

*Feature-based flow visualization:* Another approach makes use of an abstraction and/or extraction step which is performed before visualization. Special features are extracted from the original dataset, such as important phenomena or topological information of the flow. Visualization is then based on these flow features (instead of the entire dataset), allowing for compact and efficient flow visualization, even of very large and/or time-dependent datasets. This can also be thought of as visualization of *derived* data. Post et al. [Post et al. 2003] cover feature-based flow visualization in detail. See Doleisch et al. [Doleisch et al. 2004a; Doleisch et al. 2004b] for more recent developments in feature-based flow visualization.

We focus on interactive geometric visualization techniques because they are very suitable for 3D vector fields. Direct flow visualization approaches such as color mapping and using glyphs applied to 3D data result in images with a high amount of occlusion and less spatial coherency than geometric approaches. The same is true for texture-based flow visualization in 3D. Texture-based flow visualization in 3D is also usually very computationally expensive. Feature-based methods are also computationally expensive, generally more so than geometric methods.

There has been a lot of work done in this area. And while some of the geometric techniques here have been presented in previous literature, they are often not illustrated in the context of real-world data sets. Here we only highlight some of the related literature.

Jobard and Lefer present a 2D technique that preserves the density of evenly spaced streamlines [Jobard and Lefer 1997a]. Zöckler et al. [Zöckler et al. 1996] present interactive 3D flow visualization with real-time illuminated streamlines. The user-interaction component of their research consists of the use of “draggers” provided by the Open Inventor graphics toolkit. However, this method still suffers from occlusion and visual complexity.

Löffelmann and Gröller use streamlets in order to highlight characteristic structures of dynamical systems [Löffelmann and Gröller 1998]. The most interesting behavior of the dynamical systems is highlighted using an automatic seeding strategy.

Fuhrmann and Gröller [Fuhrmann and Gröller 1998] use dash tubes with reduced occlusion, animation for clear direction, and fast rendering. There are a few ways in which their techniques are related to ours since they address the same perceptual problems associated with 3D visualization that we do. They also add strong user in-

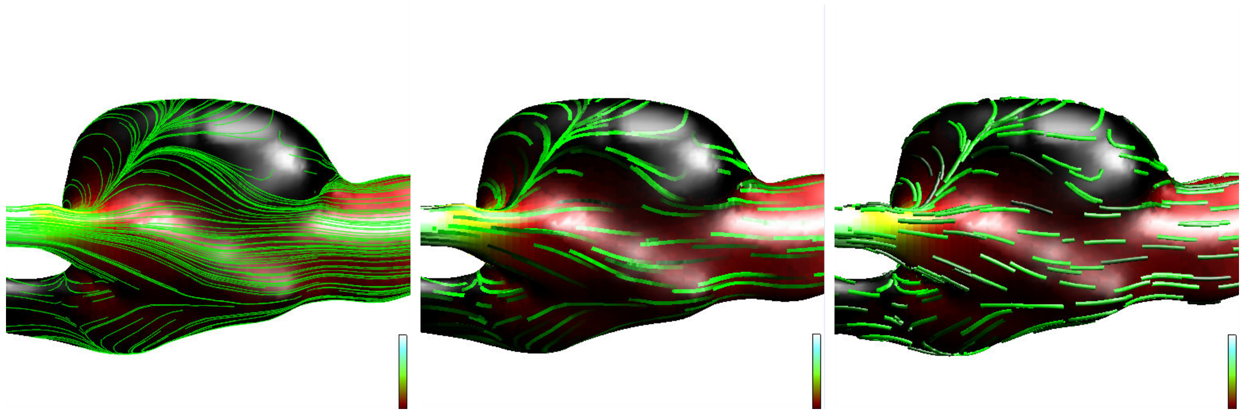

Figure 3: The visualization of blood flow at the surface of an aneurysm: (left) geometric flow visualization using streamlines (middle) oriented streamlines and (right) streamlets.

teraction techniques via the use of magic lenses and magic boxes. However, their presentation lacks application to practical data sets. In fact we see no clear illustration of their technique applied to a real data set. We also supply more interactive degrees of freedom to the visualization via new integration-based glyph representations.

## 3 Oriented Streamlines and Streamlets

One of the drawbacks of conventional streamlines is the lack of flow orientation (upstream vs. downstream direction) depicted in a still image. Our system incorporates an oriented streamline implementation. Oriented streamlines convey the downstream direction of the flow by varying the opacity as a function of particle trace evolution. In other words, the further downstream an integration path is traced, the higher the opacity of the streamline. This can be implemented by giving the streamlines a finite width, either automatically or through user-defined parameters, and using semi-transparent polygons in order to depict an oriented streamline (Figure 3, middle). Arrow heads could also be used to achieve the same effect. However, arrow head glyphs can lead to visual clutter without careful treatment.

Attention must be paid when rendering oriented streamlines on boundary surfaces in order to prevent artifacts resulting from overlapping streamline and boundary surface polygons. These artifacts can be avoided through the use of OpenGL's polygon offset functionality. The result is similar to that of OLIC (Oriented Line Integral Convolution) [Wegenkittl and Gröller 1997; Wegenkittl et al. 1997]. One important difference is that OLIC is based on a traditionally slower approach derived from LIC. Also OLIC is more suitable for the visualization of 2D vector fields.

For the case of unsteady flow, drawing a continuous particle path using a single time-step of the data set can be considered misleading. This is because no particle actually traces such a path. For the case of slices and surfaces, the visualization becomes even more problematic because a component of the vector field is taken away, namely that component orthogonal to the slice or surface, absent after a projection onto the slice or surface. One approach to handling this is through the use of streamlets (short streamlines). Figure 3, left-to-right, shows the use of streamlines, oriented streamlines, and streamlets all applied to the same data set. The data set in this case is simulation data coming from blood flow through an aneurysm.

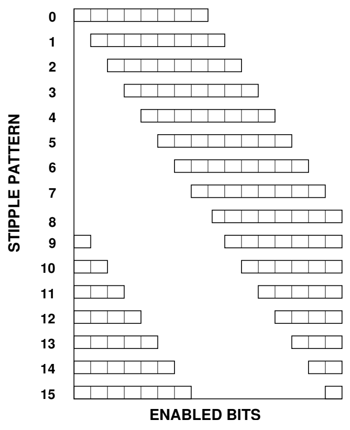

Figure 4: The 16-bit stipple pattern series used for animating streamlines in real-time, based on OpenGL 1.1.

## 4 Animated Streamlines

Here, we use a stippling approach to animate streamlines such that the downstream direction of the flow is depicted. The advantage here is that the stippling approach is supported by OpenGL 1.1. and commodity graphics hardware. Thus real-time frame rates can be achieved even for large numbers of streamlines as well as platform independence. Anti-aliasing, also supported by the graphics hardware, can be added to visually enhance the results at very little overhead.

We apply a line stipple pattern to streamline paths. Each streamline is rendered using one of 16 stipple patterns shown in Figure 4. In order to add animation, we simply shift the stipple pattern applied to the integral paths at rendering time [1]. This approach is reminiscent of that used by Jobard and Lefer [Jobard and Lefer 1997b] or Berger and Gröller [Berger and Gröller 2000] where a color-table look-up approach is used to animate the streamlines. One important difference is that the technique here applies well to 3D flow.

Without special handling, geometric techniques can also suffer from some of the same perceptual problems that direct flow visualization can. One means by which to focus on a particular subset,

[1]For supplementary images and MPEG animations, please visit: `http://www.VRVis.at/scivis/geometricApproach/`

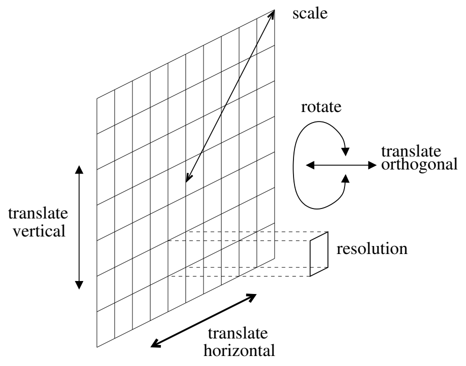

Figure 5: Our seeding plane implementation has several interactive DoFs including: three translational, scaling, rotation, resolution control.

area of interest, or feature of a flow field is via a streamline seeding strategy. In general, three popular streamline seeding strategies are often used: (1) *image-based* seeding strategies such as that described by Turk and Banks [Turk and Banks 1996] or the evenly spaced-streamline seeding strategy presented by Jobard and Lefer [Jobard and Lefer 1997a], (2) *topological* or feature-based, seeding strategies such as those presented by and L̈ffelmann and Gröller [Löffelmann and Gröller 1998], Sanna et al. [Sanna et al. 2000], or Verma et al. [Verma et al. 2000] and (3) *interactive* seeding strategies using a streamline seeding rake used by Bryson and Levit [Bryson and Levit 1992] or Schultz et al. [Schulz et al. 1999]. Our approach falls into the third category–an interactive streamline seeding strategy. Users would like full control over which subsets of the vector field to highlight in order to highlight both desirable *and* undesirable characteristics of the flow.

A schematic of our interactive streamline seeding tool is shown in Figure 5. This tool provides the user with several interactive degrees of freedom (DoF): three translational, scaling, rotational, and resolution control. These interactive DoFs are required to investigate the results of CFD simulations because the meshes from CFD embrace a wide variety of components, features, and levels of resolution. Ideally, the tools used to analyze and visualize these data sets should be flexible enough to adapt their size, orientation, and resolution to fit the features of interest either automatically or through user-specified parameters.

Figure 6 shows animated-dashed streamlines used to visualize tumble motion [Laramee et al. 2004c]. Tumble motion is the name given to an ideal pattern of flow within the combustion chamber of a gas engine. The sparser animated-dashed streamlines allow the user to see through the volume. Furthermore, the implementation is simpler than the dash tube technique of Fuhrmann and Gröller [Fuhrmann and Gröller 1998].

## 5 Streamcomets

Streamcomets are an extension of the streamrunner [Laramee 2002]. The streamrunner addresses the problems of occlusion and scene complexity by giving the user control over the evolution of streamlines from seeding time until they terminate. A streamline may terminate when it reaches a boundary in the geometry, reaches a region of zero velocity, or reaches a maximum length set by the user. The two interactive DoFs afforded by the streamrunner are: (1) the position of the stream's head along the integral path and (2) the diameter of the the integral object, in this case the tube diameter.

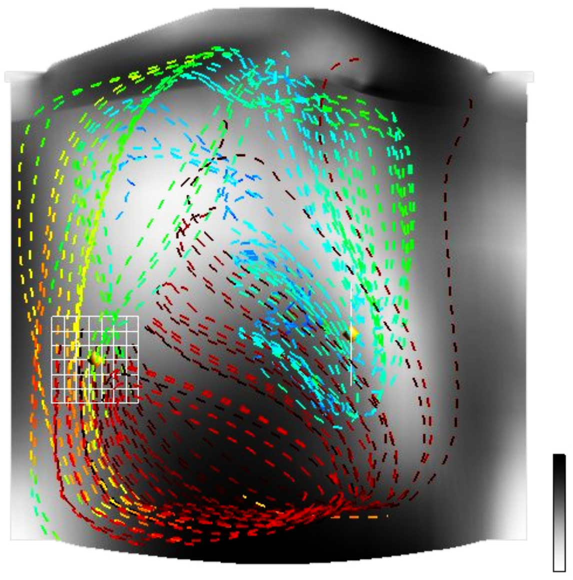

Figure 6: The visualization of tumble motion using animated, dashed streamlines. Two seeding planes are used: one seeding color mapped streamlines, the other emanating red streamlines. A gray-scale mapped slice serves as context information.

Using the streamrunner, the user is able to set the stream evolution to its origin as shown in Figure 7. In this figure, only the seeds are shown. Individual streamlines are easily distinguished and focused upon early in their evolution because occlusion has been almost eliminated while visual complexity is at a minimum. The streamrunner can then be used to change the current geometric length of the shaded tubes such that the user can watch the streamlines grow, or *run*, in the direction of the flow. This gives a clear, unequivocal indication of flow direction. The user is able to focus on an individual streamline, a group of streamlines, or a particular area of the flow as users adjust the current geometric length. Watching the streams flow in 3D combined with shading, gives added depth cues. The streamrunner also allows the user to trace the evolution of the streamlines *backwards* in order to see where a path originated.

Streamcomets follow a very intuitive metaphor. They offer four interactive DoFs as shown in Figure 8. The user is given interactive control over: (1) the position of the head along the integral path, (2) the diameter of the comet head and comet tail, (3) the length of the semi-transparent comet tail, and optionally (4) the animation speed of the comet along the path of integration. Coupled with more interactive degrees of freedom, streamcomets offer the advantage of showing local flow direction and curvature for static images. There is strong evidence to support the notion that flow visualization objects that show the direction of the local vector field improve the user's ability to identify critical points and understand particle advection paths [Laidlaw et al. 2001].

Figure 9 gives us an impression of what it is like to use the streamcomets for 3D flow visualization. We include the semi-transparent ring geometry as context information. We also apply a semi-transparent function to the comet tails and give them a glowing effect. The alpha value along each comet tail is a function of the distance to the comet head i.e., the further away from the head, the more transparent the tail.

Another useful feature is the option of animating the streamcomets. Conceptually, animating the streamcomets such that the comet head

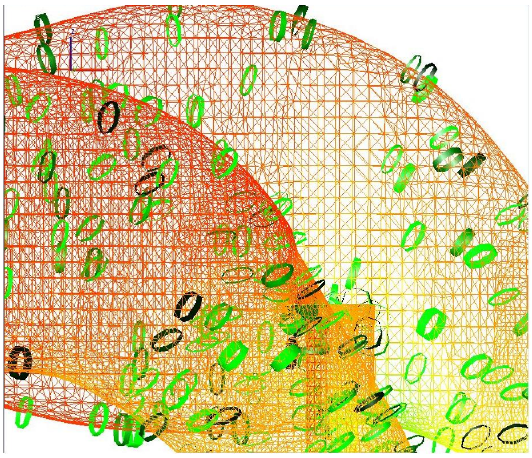

Figure 7: This image shows stream seeds as short pipe segments including a wire-frame context of the connecting pipes in the intake ports data set. In this way occlusion and image complexity are minimized.

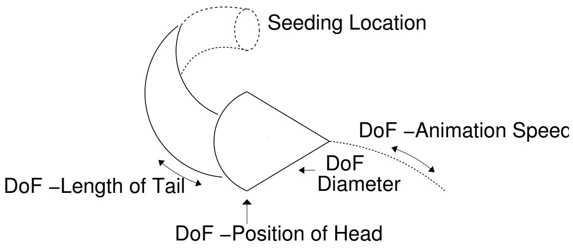

Figure 8: The streamcomet promotes four interactive degrees of freedom: (1) the position of the comet head along the path of integration, (2) the diameter of the comet head and tail, (3) the length of the comet tail, and optionally (4) the animation speed of the comets

position is automatically incremented along the path of integration, acts as a visual search function. The viewer is able to use the animation to search for optimal comet head positions. This is very useful when the user is not sure where to position the head, searching for interesting features in the flow field, or optimizing the other interactive degrees of freedom. We also give the user the option of interactively adjusting the animation speed.

We do not propose the streamrunner and streamcomet as standalone features. They are meant to be combined with other classic, 3D interaction techniques such as rotation, scaling, and translation. Additional important features we have included are: the option of choosing a non-uniform coloring scheme so colliding geometric objects can be more easily distinguished, turning on or off semi-transparent or wire-frame context information, and adjusting the streamline seeding density in the flow field.

The comet glyph can intuitively encode time attributes for unsteady flow visualization. At the top of Figure 10, we see a sample seed point whose geometric location is constant over time and from which streamcomets are injected into the flow, similar to a *streakline* – the line traced by a set of particles that have previously passed through a unique point in the domain [Schroeder et al. 2003]. As the comet ages (after being injected into the flow), the size of the head decreases as does a real comet when traveling through space. Also the length of the tail encodes the local instantaneous velocity at the comet's current position. The color of the comet head encodes the local temperature and the color of the comet tail reflects

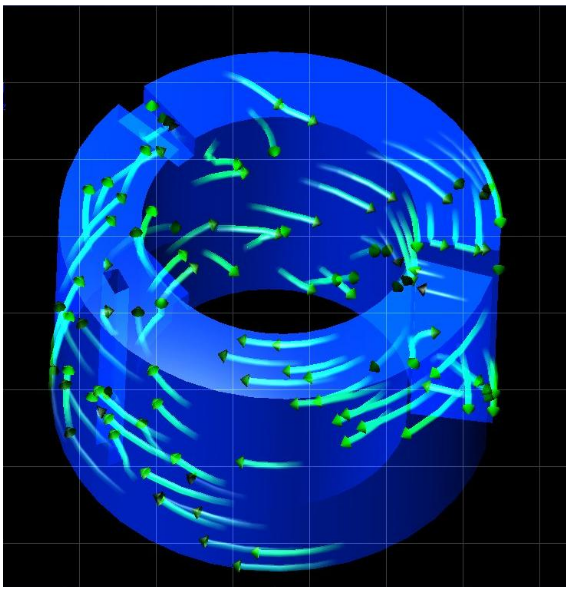

Figure 9: Here streamcomets are rendered in the context of a semi-transparent ring geometry. We add semi-transparency and a glowing impression to the streamcomet tails, whose transparency increases with the distance from the comet head.

another scalar attribute of the flow such as pressure. If we represent comet tails using streamtubes [Ueng et al. 1996], the local convergence and divergence of the flow may be encoded. If comet tails are represented using streamribbons [Ueng et al. 1996], local vorticity is encoded. Ideally, the user is able to toggle between the two representations. We claim that the use of streamcomet glyphs for encoding attributes of the flow is more intuitive than using other glyphs such as superquadric shapes. The interactive analogue of a streakline is a *streakrunner*. The streakrunner is an interactive control that defines the geometric length of the streakline. Such a line is shown in Figure 10 connecting the comet heads.

## 6 Results

Performance times depend on the number of streamlines. Performance times for the animated streamlines are given in Table 1. Performance was evaluated on a machine running Red Hat Linux with a 3.2 GHz Intel Xeon dual processor, 2 GB of RAM, and an *NVIDIA Quadro FX 1300* graphics card. Note that the frame rate also varies as a result of caching. Anti-aliasing adds very little overhead since it is built into OpenGL 1.2 and hence is supported by most graphics cards. As we see, the stippling approach allows animation of thousands of streamlines in real-time. Furthermore, we have not employed display lists to increase the frame rates. Figure

| no. of streamlines | with anti-aliasing | without anti-aliasing |
|---|---|---|
| 10 | 101 | 101 |
| 100 | 101 | 101 |
| 1,000 | 64 | 66 |
| 2,500 | 35 | 40 |
| 5,000 | 20 | 24 |
| 10,000 | 11 | 14 |

Table 1: Sample frame rates for the animated streamlines in frames per second.

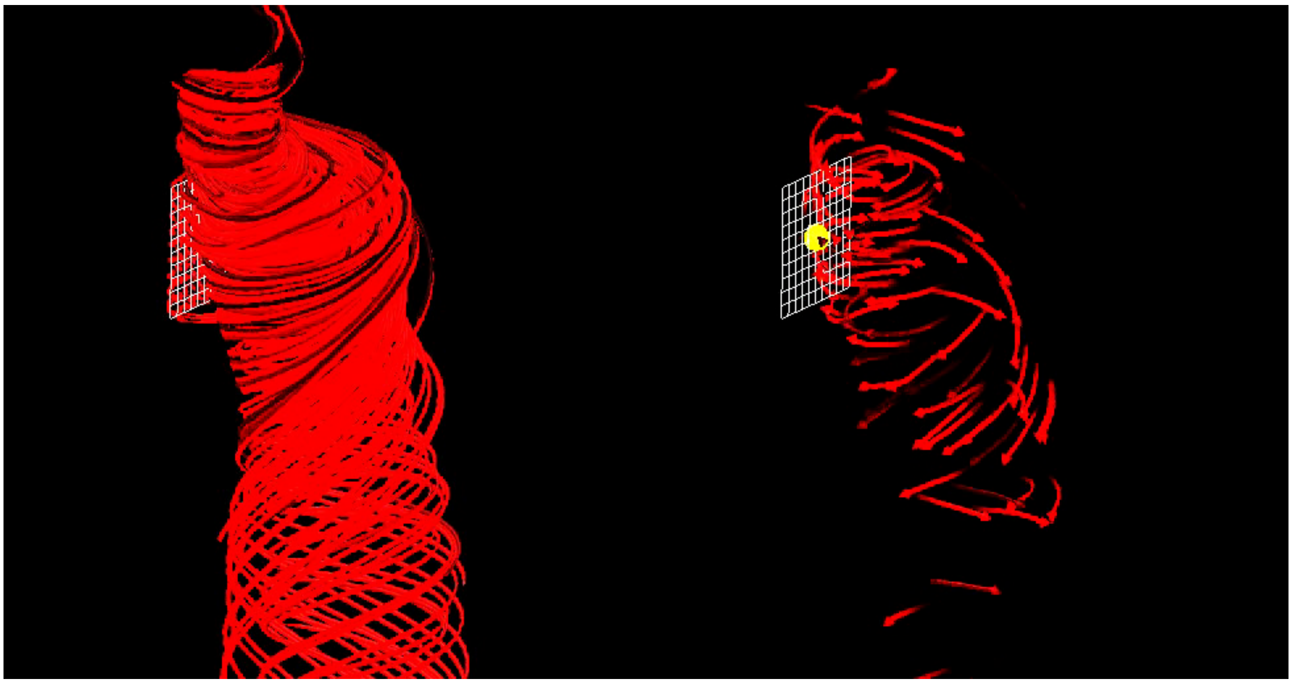

Figure 12: The visualization of a vortex using (top) streamlines and (bottom) animated streamcomets. The streamcomets reduce occlusion and provide the same coverage when animated.

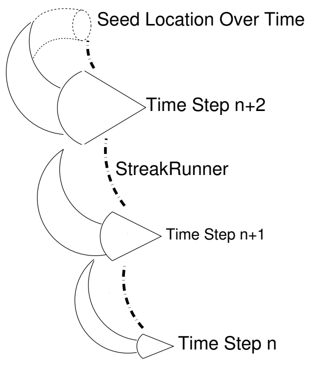

Figure 10: The use of the streamrunner and streamcomets for unsteady flow visualization. Comet heads shrink over time. i.e., the older the comet, the smaller the comet head. The interactive equivalent of a streakline is a streakrunner, which interactively controls the number of discrete time steps along the streakline defined by the series of comet heads.

11 shows two seeding planes inside the combustion chamber of a piston valve. The seeding plane in the top (foreground) has streamcomets emanating from it. The seeding plane in the middle (background) seeds shaded streamlines. We emphasize the importance of the user's ability to resize the streamcomets along arbitrary dimensions when zooming in and out of the data sets. It is important to note that changes to the diameter of the comet heads apply to the entire collection of streamcomets, and are not applied on a per-comet basis. Applying size changes to individual comets would lead to misleading visualization results, e.g., the user may interpret different comet head sizes to be a reflection of scalar properties inherent in the flow field.

Figure 12 gives us another impression of what it is like to use the streamcomets for flow visualization in 3D. Here, both streamlines and streamcomets are used to visualize a vortex. Giving the user interactive control over the placement of the comet heads, the diameter of the comet heads and tails, the seeding density, and the length of semi-transparent comet tails, affords the user a very good opportunity to see the characteristics of the flow field.

## 7 Conclusions and Future Work

The added interaction provided by our geometric flow visualization techniques is very useful for flow visualization in 3D and within the domain of versatile grids associated with CFD simulations. This is because they are based on geometric primitives that are more suitable for the visualization of 3D flow than approaches based on color-mapping, glyphs, or textures only. The user control afforded by the streamcomets as well as the intuitive metaphor on which they are based makes them more versatile for 3D flow visualization than previous techniques. Furthermore, the simplicity of our approaches makes them strong candidates for inclusion in other flow visualization software packages. The approaches described here have been included in a cross-platform, industry-level visualization application for the analysis of CFD simulation data. These geometric objects give a new level of control over to users investigating a vector field. We encourage the reader to view the animations at the given URL.

Future work could go in several directions including: (1) an implementation prototype of the streamrunner and the streamcomet for unsteady flow visualization including the introduction of a *pathrunner* – the unsteady equivalent of a streamrunner, a streakrunner – the interactive equivalent of a streakline or (2) a formal HCI evaluation of the perceptual effectiveness of the streamrunner and streamcomets for 3D flow visualization.

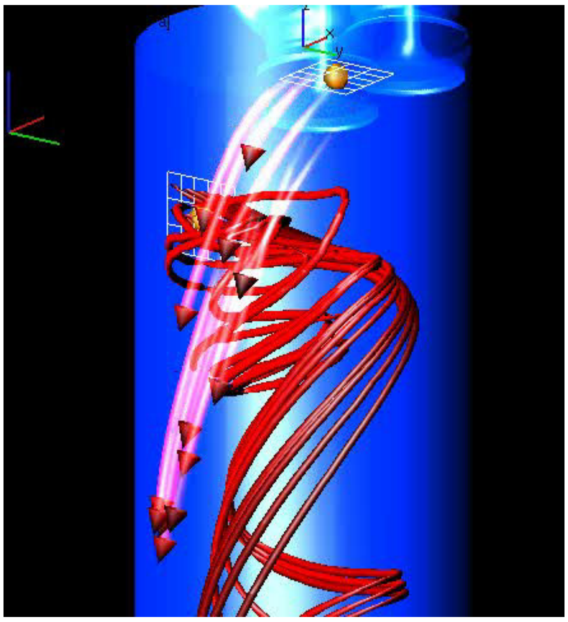

Figure 11: Two seeding planes in the combustion chamber of a piston valve: one seeding streamcomets, the other seeding shaded streamlines.

## 8 Acknowledgments

We would like to thank all those who have contributed to financing this research, including AVL (`www.avl.com`) and the Austrian national research program Kplus (`www.kplus.at`). A special thanks goes to Helmut Doleisch of VRVis for his valuable feedback. All CFD simulation data is courtesy of AVL.

## References

BERGER, S., AND GRÖLLER, M. E. 2000. Color-Table Animation of Fast Oriented Line Intgral Convolution for Vector Field Visualization. In *WSCG 2000 Conference Proceedings*, 4–11.

BRYSON, S., AND LEVIT, C. 1992. The Virtual Wind Tunnel. *IEEE Computer Graphics and Applications 12*, 4 (July), 25–34.

CABRAL, B., AND LEEDOM, L. C. 1993. Imaging Vector Fields Using Line Integral Convolution. In *Poceedings of ACM SIGGRAPH 1993*, ACM Press / ACM SIGGRAPH, Annual Conference Series, 263–272.

DOLEISCH, H., MAYER, M., GASSER, M., PRIESCHING, P., AND HAUSER, H. 2004. Interactive Feature Specification for the Visual Analysis of a Diesel Engine. Technical Report TR-VRVis-2004-011, VRVis Research Center.

DOLEISCH, H., MAYER, M., GASSER, M., WANKER, R., AND HAUSER, H. 2004. Case Study: Visual Analysis of Complex, Time-Dependent Simulation Results of a Diesel Exhaust System. In *Data Visualization, Proceedings of the 6th Joint IEEE TCVG–EUROGRAPHICS Symposium on Visualization (VisSym 2004)*, 91–96.

FUHRMANN, A. L., AND GRÖLLER, M. E. 1998. Real-Time Techniques for 3D Flow Visualization. In *Proceedings IEEE Visualization '98*, IEEE, 305–312.

HIBBARD, W., AND SANTEK, D. 1989. Interactivity is the Key. In *Proceedings of the Chapel Hill Workshop on Volume Visualization*, 39–43.

JOBARD, B., AND LEFER, W. 1997. Creating Evenly–Spaced Streamlines of Arbitrary Density. In *Proceedings of the Eurographics Workshop on Visualization in Scientific Computing '97*, Springer-Verlag, vol. 7, Eurographics.

JOBARD, B., AND LEFER, W. 1997. The Motion Map: Efficient Computation of Steady Flow Animations. In *Proceedings IEEE Visualization '97*, IEEE Computer Society, 323–328.

LAIDLAW, D. H., KIRBY, R. M., DAVIDSON, J. S., MILLER, T. S., DA SILVA, M., WARREN, W. H., AND TARR, M. 2001. Quantitative Comparative Evaluation of 2D Vector Field Visualization Methods. In *Proceedings IEEE Visualization 2001*, IEEE Computer Society, 143–150.

LARAMEE, R. S., HAUSER, H., DOLEISCH, H., POST, F. H., VROLIJK, B., AND WEISKOPF, D. 2004. The State of the Art in Flow Visualization: Dense and Texture-Based Techniques. *Computer Graphics Forum 23*, 2 (June), 203–221.

LARAMEE, R. S., VAN WIJK, J. J., JOBARD, B., AND HAUSER, H. 2004. ISA and IBFVS: Image Space Based Visualization of Flow on Surfaces. *IEEE Transactions on Visualization and Computer Graphics 10*, 6 (Nov.), 637–648.

LARAMEE, R. S., WEISKOPF, D., SCHNEIDER, J., AND HAUSER, H. 2004. Investigating Swirl and Tumble Flow with a Comparison of Visuaization Techniques. In *Proceedings IEEE Visualization '04*, IEEE Computer Society, 51–58.

LARAMEE, R. S. 2002. Interactive 3D Flow Visualization Using a Streamrunner. In *CHI 2002, Conference on Human Factors in Computing Systems, Extended Abstracts*, ACM Press, ACM SIGCHI, 804–805.

LÖFFELMANN, H., AND GRÖLLER, M. E. 1998. Enhancing the Visualization of Characteristic Structures in Dynamical Systems. In *Proceedings of the 9th Eurographics Workshop on Visualization in Scientific Computing*, Springer-Verlag, Eurographics, 35–46.

POST, F. H., VROLIJK, B., HAUSER, H., LARAMEE, R. S., AND DOLEISCH, H. 2002. Feature Extraction and Visualization of Flow Fields. In *Eurographics 2002 State-of-the-Art Reports*, The Eurographics Association, 69–100.

POST, F. H., VROLIJK, B., HAUSER, H., LARAMEE, R. S., AND DOLEISCH, H. 2003. The State of the Art in Flow Visualization: Feature Extraction and Tracking. *Computer Graphics Forum 22*, 4 (Dec.), 775–792.

SANNA, A., MONTRUCCHIO, B., AND ARINAZ, R. 2000. Visualizing Unsteady Flows by Adaptive Streaklines. In *WSCG 2000 Conference Proceedings*, 84–91.

SCHROEDER, W. J., MARTIN, K. M., AND LORENSEN, W. E. 2003. *The Visualization Toolkit, An Object-Oriented Approach to 3D Graphics*, 3rd ed. Kitware, Inc.

SCHULZ, M., RECK, F., BARTELHEIMER, W., AND ERTL, T. 1999. Interactive Visualization of Fluid Dynamics Simulations in Locally Refined Cartesian Grids. In *Proceedings IEEE Visualization '99*, 413–416.

TURK, G., AND BANKS, D. 1996. Image-Guided Streamline Placement. In *SIGGRAPH 96 Conference Proceedings*, Addison Wesley, Annual Conference Series, ACM SIGGRAPH, 453–460.

UENG, S. K., SIKORSKI, C., AND MA, K. L. 1996. Efficient Streamline, Streamribbon, and Streamtube Constructions on Unstructured Grids. *IEEE Transactions on Visualization and Computer Graphics 2*, 2 (June), 100–110.

VAN WIJK, J. J. 1991. Spot noise-Texture Synthesis for Data Visualization. In *Computer Graphics (Proceedings of ACM SIGGRAPH 91)*, ACM, T. W. Sederberg, Ed., vol. 25, 309–318.

VAN WIJK, J. J. 2002. Image Based Flow Visualization. *ACM Transactions on Graphics 21*, 3, 745–754.

VERMA, V., KAO, D., AND PANG, A. 2000. A Flow-guided Streamline Seeding Strategy. In *Proceedings IEEE Visualization 2000*, 163–170.

WARE, C., AND FRANCK, G. 1996. Evaluating Stereo and Motion Cues for Visualizing Information Nets in Three Dimensions. *ACM Transactions on Graphics 15*, 2 (Apr.), 121–140.

WEGENKITTL, R., AND GRÖLLER, M. E. 1997. Fast Oriented Line Integral Convolution for Vector Field Visualization via the Internet. In *Proceedings IEEE Visualization '97*, 309–316.

WEGENKITTL, R., GRÖLLER, M. E., AND PURGATHOFER, W. 1997. Animating Flow Fields: Rendering of Oriented Line Integral Convolution. In *Computer Animation '97 Proceedings*, 15–21.

ZÖCKLER, M., STALLING, D., AND HEGE, H. 1996. Interactive Visualization of 3D-Vector Fields Using Illuminated Streamlines. In *Proceedings IEEE Visualization '96*, 107–113.