USING INTEGRAL SURFACES TO VISUALIZE CFD SIMULATION DATA

Tony McLoughlin[1], Matthew Edmunds[1], Robert S. Laramee[1], Mark W. Jones[1], Guoning Chen[3], Eugene Zhang[2]

[1]Visual and Interactive Computing Group, Department of Computer Science, Swansea University, Swansea, UK
Email: {cstony,csmatti,r.s.laramee, m.w.jones}@swansea.ac.uk

[2]School of Electrical Engineering and Computer Science, Oregon State University, Corvallis, OR 97331
Email: zhange@eecs.oregonstate.edu

[3]Scientific Computing and Imaging Institute (SCI), University of Utah, Salt Lake City, UT 84112
Email: chengu@sci.utah.edu

THEME
Visualization

KEYWORDS
stream surfaces, path surfaces, streak surfaces, flow visualization, vector field visualization, CFD simulation data

SUMMARY

Visualization of 3D, unsteady (4D) flow is very difficult due to both perceptual challenges and the large size of 4D vector field data. Here, we describe the use of integral surfaces for visualization of CFD simulation data. By "integral" surfaces we mean surfaces based on massless particles that are integrated according to the underlying flow. Traditionally, integral curves, e.g., streamlines, pathlines, and streaklines are used to visualize 3D and 4D flow. However, integral surfaces offer clear benefits over integral curves when visualizing flow. Despite the clear benefits that stream, path, and streak surfaces bring when visualizing 4D vector fields, their use in both industry and for research has not proliferated. This is due, in part, to the complexity of integral surface construction algorithms. We introduce algorithms for the construction of stream, path, and streak surfaces that are fast and do not rely on any complicated data structures or surface parametrization. Our surface construction algorithms generate the surfaces using a quadrangular mesh. The algorithms can be applied to large data sets because they are based on local operations performed on quad primitives. The algorithms offer a combination of speed for exploration of 3D, unsteady flow and high precision. Thus they are suitable for inclusion into any visualization application. We demonstrate the techniques on a series of simulation data sets and show a number of benefits that stem naturally from them. We also introduce interaction techniques in order to address the perceptual challenges associated with visualizing 3D, time-dependent CFD simulation data.

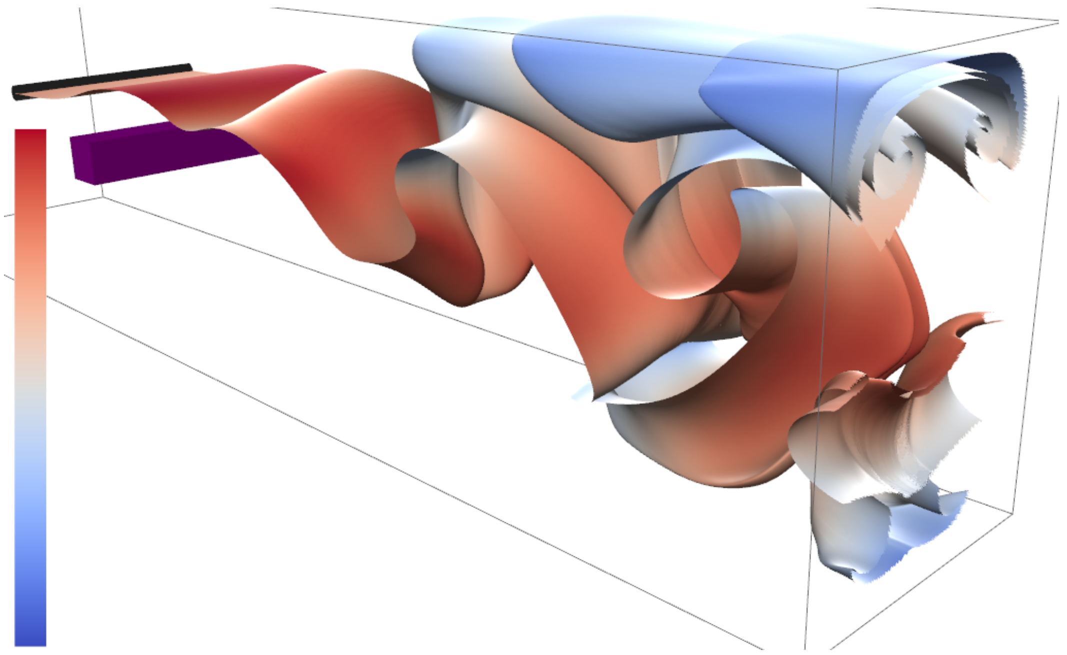

Figure 1: A complete streak surface depicting flow past a cuboid computed and rendered at approximately 3 frames-per-second. This image shows complex regions illustrating interesting flow structures. Color is mapped to velocity magnitude.

1 INTRODUCTION AND MOTIVATION

Streamlines, pathlines, and streaklines (also called integral lines) are visualization techniques commonly used to depict complex, 3D and 4D vector fields. Tracing and rendering many field lines can cause perceptual problems. Too many integral lines result in visual complexity, occlusion, and visual clutter. This makes using surface primitives an attractive alternative for the visualization of 3D and 4D flow. Shading can easily be applied to enhance depth perception and surfaces are, in general, characterized by much more visual coherency than lines.

Streak surfaces are the union of all streaklines continuously seeded from a common curve (the seeding curve) over time. However, despite the advantages and insight that stream, path, and streak surfaces (also called integral surfaces) enable when investigating flow fields, they are not generally included in visualization applications. This is due, in part to the computational complexity involved in generating their meshes.

The computational cost of constructing integral surfaces stems from factors including the large number of integrations required. For streak surfaces, every vertex in the mesh must be integrated at each time step. For large meshes this results in a large number of computations. Another major source of computational expense arises from computing the connectivity of the mesh vertices. Divergence, convergence and shear can occur anywhere in the surface. These factors can make the inclusion of streak surfaces into visualization packages prohibitively expensive.

We present CPU-based stream, path, and streak surfaces algorithms (Figure 1) that can run at interactive frame rates and places less restriction on mesh size and precision than a GPU-based implementation.

The main foci of this paper are:

- The description of stream, path, and streak surface algorithms that can offer interaction and high precision.
- Algorithms suitable for use on large out-of-core data sets and relatively simple to implement as a result of the local operations performed on quads. Thus, it is suitable for inclusion in any visualization system.

We provide user-controlled parameters that allow the user to trade off between performance and accuracy. This enables the user to switch between modes for quick investigation of the flow domain and high-accuracy representation for presentation and analysis. The use of quad meshes has increased in recent years, with many algorithms aimed at quad-based re-meshing or simplification of quad-meshes. Benefits of quad-based meshes are demonstrated by Alliez et al. [1] and Tong et al. [17].

However in order to develop such algorithms several challenges must be overcome such as maintaining a continuous, accurate, quad mesh under flow convergence, divergence and shear.

The rest of the paper is organised as follows: Section 2 provides a discussion of previous work related to flow surface construction algorithms. Section 3 describes the computational model of our algorithm. A detailed discussion of the implementation is provided in Section 4. Section 5 presents an evaluation of the algorithm showing it applied to various simulation data sets. Finally, Section 6 concludes the paper and identifies areas of future research.

2 RELATED WORK

For a complete overview of geometric visualization techniques. see McLoughlin et al.[12]

Hultquist introduces a method based on an advancing front [7]. The sampling rate is adjusted by the insertion or removal of streamlines based on distance between neighboring point on the front. In contrast to this local approach Van Wijk presents a global approach for stream surface generation [18]. A continuous function $f(x, y, z)$ is placed on the boundaries of the data set and values at all other grid points are solved numerically ([18] uses the convection equation). An iso-surface extraction technique can then be used to construct the stream surface. Scheuermann et al. assume a piecewise linear interpolation approximation of flow and a tetrahedral grid so that an analytical solution may be found for the construction of the stream surface
[15]. Garth et al. [5] present a method for the construction of stream surfaces in areas of complex flow. This is based upon the advancing front method introduced by Hultquist [7] but improves on it by using more complex parameterization for streamline integration and density control. As an extension to isosurfaces [9], Laramee et al. [10] combined texture advection with stream surfaces to provide a more detailed visualization by showing the inner flow structure of the surface. These previous methods are restricted to steady-state flow.

A point-based method for stream surface and path surface construction was introduced by Schafhitzel et el. [14]. Insertion and removal of points are handled similar to Hultquist's method [7] to maintain sufficient density of points.

Garth et al. present an improved stream and path surface construction algorithm focusing on high accuracy [6]. This method decouples the surface integration and the surface rendering process. The surface construction comprises of advecting a set of timelines through the flow field. Connecting curves representing timelines are then computed. These curves are subject to given predicates in order to refine them where necessary.

McLoughlin et al. [11] demonstrate a simplified stream surface construction method using quad primitives. The refinement of the surface front is performed on a quad-by-quad basis. Schneider et al. [16] present a method of stream surface construction using a higher-order interpolation scheme. This method provides very smooth surfaces of fourth-order accuracy.

Von Funck et al. [19] describe the construction of smoke surfaces. Smoke surface generation involves coupling the opacity of the triangles that comprise the mesh to their size and shape. Krishnan et al. [8] present a novel streak and time surface algorithm. This technique guarantees a C^1 continuous curve for the integral curves. Three basic operations are defined for the surface adaptation process, these are edge split, edge flip and edge collapse. Bürger et al. [2] present two streak surface techniques implemented on the GPU. The first technique is based on quads. Both the quad and triangle-based versions of the algorithm are very fast, e.g., several frames per second. However their GPU-implementation places limits on both the size of the streak surface and the data set. The method presented here removes these restrictions whilst still retaining an interactive speed on a CPU. It is also applied to 4D simulation data.

3 METHOD

The algorithms for construction of integral surfaces consists of a series of operations as illustrated in Figure 2:

1. Seed a curve using an interactive rake.
2. Update the streak surface by integrating every mesh vertex.
3. Refine the surface according to local deformation, by inserting/removing vertices/quads and updating the mesh connectivity.
4. Update the sampling rate of the vector field, by inserting and/or removing mesh vertices and joining them together using a quad-based topology.
5. Test boundary conditions such as object boundary intersection and zero velocity.
6. Render the surface.
7. This process iterates until the surface exits the space-time domain. At which point integration is terminated and the surface is rendered.

Local operations are performed on quads and their neighbors which maintain sufficient sampling of the vector field. Vertex insertion is introduced when the Euclidean distance of a quad's edge is too long. Conversely a vertex is removed if neighboring points are too close.

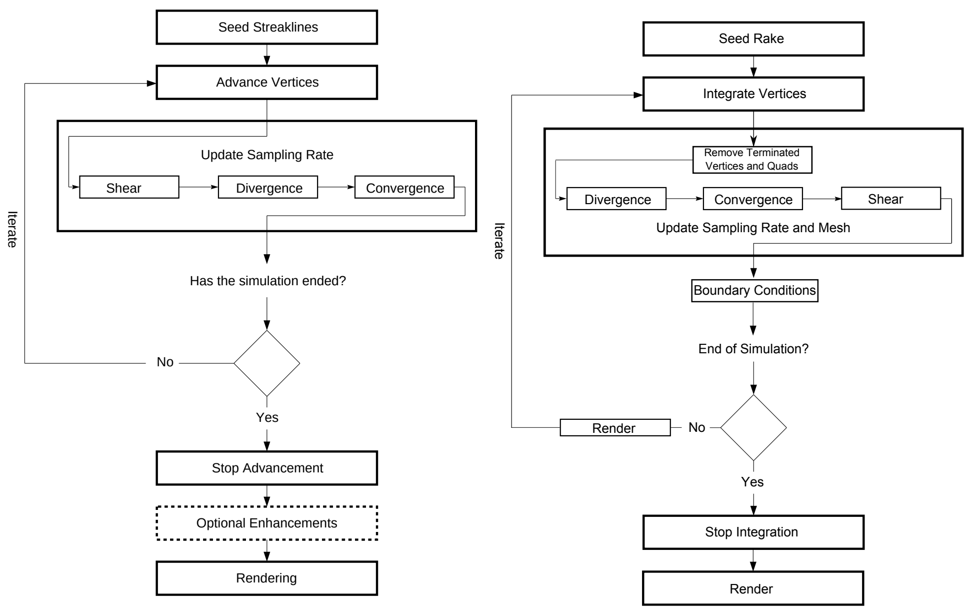

Figure 2: (left) The stream and path surface computation pipeline [11]. (right) An overview of the streak surface construction algorithm [13].

Mesh connectivity is then resolved ensuring all quad primitives are generally regular and that the surface is smooth. The algorithm for generating streak surfaces is a generalization of that for stream and path surfaces. Full algorithm details are given by McLoughlin et al. [11], [13].

4 RESULTS

In Figure 1, a complex streak surface is constructed. This surface shows interesting flow structures that would be difficult to see if only streaklines were rendered. Surfaces also aid in identifying the transition of turbulence be showing stretching and folding. Our system constructs and renders this streak surface at approximately 3 frames per second (fps).

Figure 3 depicts a complete streaksurface exhibiting a multitude of flow characteristics. This surface contains regions of divergence, convergence and shear as well as the splitting behavior when an object is encountered. This surface is computed and rendered at roughly 2 fps. Color is mapped to velocity magnitude in all our examples unless stated otherwise. Figure 3 depicts another semi-transparent streak surface generated from the simulation of flow past a cuboid. The result demonstrates the tearing of the surface into two independent regions when an object boundary is encountered. Splitting is detected when the velocity magnitude of an internal streakline drops below a given threshold. When the surface tears, the separated wavefront are advanced independently. Figure 4 shows streak surfaces generated on a time-dependent tornado simulation. The tornado exhibits large regions of shear flow and is ideal to test the robustness of our shear-handling method. Our implementation renders this at 10 fps.

Our system supports out of core memory management in order to handle large data sets. We

adopt a method similar to Bürger et
al [2]. We store as many time-steps as possible in main memory. We then employ a sliding window. When performing the particle advection, we interpolate between a pair of timesteps.

Figure 4 depicts a streak sheet on the full resolution ($500^2 \times 100 \times 48$) Hurricane Isabel simulation. A streak sheet is created by simply terminating the insertion of new points at the seeding rake after a period of time. In this case the sheet was released so that it was captured by the eye of the hurricane. It then traces the hurricane's path.

Our results show that the integration phase comprises a large proportion of the computational effort – typically 40-80% of the computation time. The mesh vertex advection is a largely parallel component, however our implementation performs this operation within a single thread. A multi-threaded version of this stage would greatly reduce the cost of this stage and consequently speed up the algorithm. However even in its current single-threaded state the algorithm generally still performs at interactive rates. While not as fast as a GPU-based version [2] the mesh does not need to fit in graphics card memory, and thus handles larger streak surfaces.

5 CONCLUSION

We present stream, path and streak surface construction algorithms for the visualization of 3D and 4D flow. They combine the advantages of both speed and size with efficient CPU-based, platform-independent implementations that place less restriction on memory and precision than a GPU-version. It is this quad-based approach from which the speed of the algorithms stem. They avoid expensive mesh re-triangulation. The local nature of the operations applied to the quad primitives enables the algorithms to be applied to large data sets. The algorithms handle flow divergence, convergence, and shear. The algorithms also allow the surface to split when it meets object boundaries. They are demonstrated on a variety of data sets posing various challenges such as turbulence and large-scale simulation.

As future work we would like to treat the case of strongly deformed non-planar quads. For example, we may be able to parameterize deformation in terms of integration time in order to create a smoother surface.

6 ACKNOWLEDGEMENTS

This work was supported in part by the EPSRC Research Grant EP/F002335/1. The authors would like to thank Roger Crawfis for providing the procedural function used to generate the tornado data simulation [4]. The authors would also like to thank Tino Weinkauf, Simone Camarri, Maria-Vittoria Salvetti, Marcelo Buffoni and Angelo Iollo for providing the flow behind a square cylinder data simulation
[3]. Hurricane Isabel data produced by the Weather Research and Forecast (WRF) model, courtesy of NCAR and the U.S. National Science Foundation (NSF). The authors would also like to thank Harry Yeh of Oregon State University for valuable discussions.

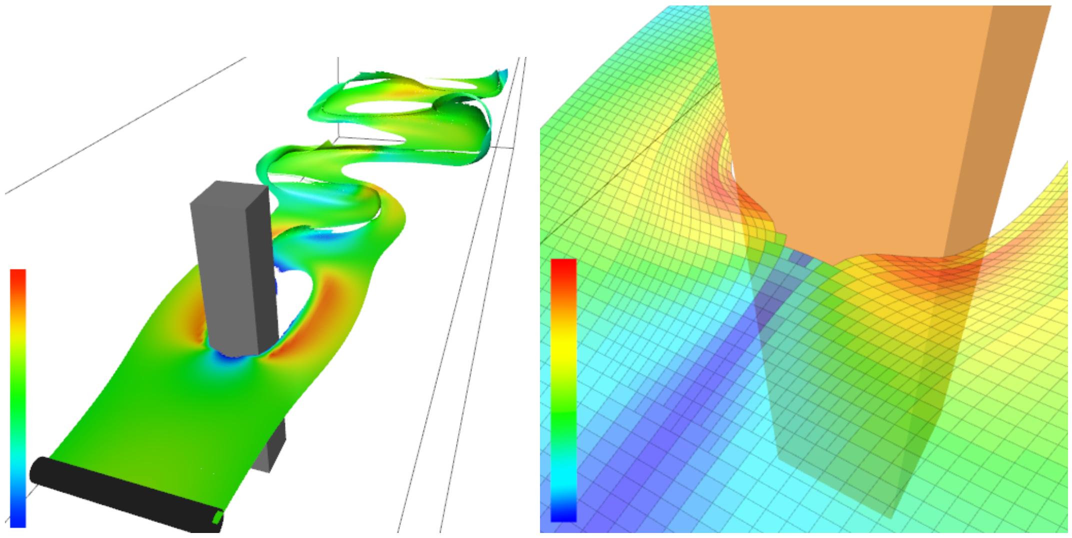

Figure 3: A streak surface of the simulation of flow past a square cylinder. The image shows a late stage of the simulation and shows the complete surface exhibiting divergence, convergence, shear and splitting behavior. When an object boundary is encountered the surface splits and moves around the boundary. This surface encounters a cuboid. Color is mapped to the local deformation of the quad primitives.

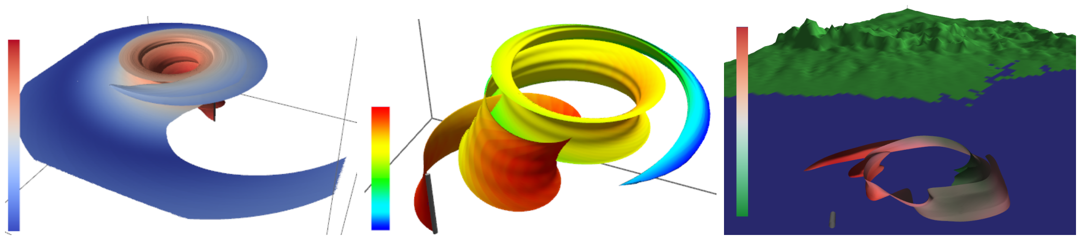

Figure 4: A streaksurface depicting a tornado simulation. The surface encounters large amounts of shear that accumulate as the simulation progresses. (middle) This case also demonstrates the property of self-intersection. (right) The surface generated from the full resolution ($500^2 \times 100 \times 48$) of the Hurricane Isabel simulation.

References

[1] P. Alliez, D. Cohen-Steiner, O. Devillers, B. Levy, and M. Desbrun. Anisotropic Polygonal Remeshing. *ACM Transactions on Graphics (SIGGRAPH)*, 22(3):485–493, 2003.

[2] K. Buerger, F. Ferstl, H. Theisel, and R. Westermann. Interactive Streak Surface Visualization on the GPU. *IEEE Transactions on Visualization and Computer Graphics (IEEE TVCG)*, 15(6):1259–1266, November/December 2009.

[3] S. Camarri, M.V. Salvetti, B. Koobus, and A. Dervieux. Large-eddy simulation of a bluff-body flow on unstructured Grids. *International Journal for Numerical Methods in Fluids*, 40(11):1431–1460, 2002.

[4] R. A. Crawfis and N. Max. Texture Splats for 3D Scalar and Vector Field Visualization. In *Proceedings IEEE Visualization '93*, pages 261–267. IEEE Computer Society, October 1993.

[6] C. Garth, H. Krishnan, X. Tricoche, T. Tricoche, and K.I. Joy. Generation of Accurate Integral Surfaces in Time-Dependent Vector Fields. *IEEE Transactions on Visualization and Computer Graphics (IEEE TVCG)*, 14(6):1404–1411, 2008.

[5] C. Garth, X. Tricoche, T. Salzbrunn, T. Bobach, and G. Scheuermann. Surface Techniques for Vortex Visualization. In *Data Visualization, Proceedings of the 6th Joint IEEE TCVG–EUROGRAPHICS Symposium on Visualization (VisSym 2004)*, pages 155–164, May 2004.

[7] J. P. M. Hultquist. Constructing Stream Surfaces in Steady 3D Vector Fields. In *Proceedings IEEE Visualization '92*, pages 171–178, 1992.

[8] H. Krishnan, C. Garth, and K.I. Joy. Time and Streak Surfaces for Flow Visualization in Large Time-Varying Data Sets. *IEEE Transactions on Visualization and Computer Graphics (IEEE TVCG)*, 15(6):1267–1274, 2009.

[10] 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.

[9] 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] T. McLoughlin, R.S. Laramee, R. Peikert, F. H. Post, and M. Chen. Over Two Decades of Integration-Based, Geometric Flow Visualization. *Computer Graphics Forum (CGF)*, 29(6):1807–1829, 2010. (available online).

[11] T. McLoughlin, R.S. Laramee, and E. Zhang. Easy Integral Surfaces: A Fast, Quad-based Stream and Path Surface Algorithm. In *Proceedings of Computer Graphics International (CGI '09)*, pages 67–76. Computer Graphics Society, Springer, May 2009. (available online).

[13] T. McLoughlin, R.S. Laramee, and E. Zhang. Constructing Streak Surfaces in 3D Unsteady Vector Fields. In *Proceedings of the Spring Conference on Computer Graphics (SCCG)*, pages 25–32, May 2010. (available online).

[14] T. Schafhitzel, E. Tejada, D. Weiskopf, and T. Ertl. Point-based Stream Surfaces and Path Surfaces. In *Graphics Interface*, pages 289–296. ACM Press, 2007.

[15] G. Scheuermann, T. Bobach, H. Hagen, K. Mahrous, B. Hamann, K. I. Joy, and W. Kollmann. A Tetrahedral-based Stream Surface Algorithm. In *Proceedings IEEE Visualization 2001*, pages 151–157, October 2001.

[16] D. Schneider, A. Wiebel, and G. Scheuermann. Smooth Stream Surfaces of Fourth Order Precision. *Computer Graphics Forum (CGF)*, 28(3):871–878, 2009.

[17] Y. Tong, P. Alliez, D. Cohen-Steiner, and M. Desbrun1. Designing Quadrangulations with

Discrete Harmonic Forms. In *Eurographics Symposium on Geometry Processing (2006)*, pages 201–210. Eurographics, 2006.

[18] J.J. van Wijk. Implicit Stream Surfaces. In *Proceedings of the Visualization ’93 Conference*, pages 245–252. IEEE Computer Society, October 1993.

[19] w. Von Funck, T. Weinkauf, H. Theisel, and H.-P. Seidel. Smoke Surfaces: An Interactive Flow Visualization Technique Inspired by Real-World Flow Experiments. *IEEE Transactions on Visualization and Computer Graphics (IEEE TVCG)*, 14(6):1396–1403, 2008.