**Interactive Numerical Flow Visualization
Using Stream Surfaces**

J.P.M. Hultquist
`hultquist@nas.nasa.gov`

Report RNR-90-009
April 1990

Numerical Aerodynamic Simulation Systems Division
NASA Ames Research Center, Mail Stop T045-1
Moffett Field, CA 94035-1000

```
@article{xxx,
 author    = "J.P.M. Hultquist",
 title     = "Interactive Numerical Flow Visualization
              Using Stream Surfaces",
 journal   = "Computing Systems in Engineering",
 volume    = 1,
 number    = "2-4",
 year      = 1990,
 pages     = "349-353",
 note      = "(also RNR Tech Report 90-009)"}
```

Abstract

Particle traces and ribbons are often used to depict the structure of three-dimensional flow fields, but images produced using these models can be ambiguous. Stream surfaces offer a more visually-intuitive method for the depiction of flow fields, but interactive response is needed to allow the user to place surfaces which reveal the essential features of a given flow field. This report describes FLORA, a software package which supports the interactive calculation and display of stream surfaces on Silicon Graphics workstations.

Alternative methods for the integration of particle traces were examined, and calculation through computational space was found to provide rapid results with accuracy adequate for most purposes. Rapid calculation of traces is teamed with progressive refinement of approximated surfaces. An initial approximation provides immediate user feedback, and subsequent improvement of the surface ensures that the final image is an accurate representation of the flow field. FLORA normally depicts surfaces with shaded raster images, but can also produce line-art for publication.

1 Nomenclature

$\vec{X}[i,j,k]$	physical position of grid sample point
(x, y, z)	point in physical (Cartesian) space
(ξ, η, ζ)	point in computational (grid) space
(α, β, γ)	fractional parts of (ξ, η, ζ)

2 Traces, Ribbons, and Surfaces

The most common method of depicting a velocity field in three-dimensional flow data is the "particle trace." This is the curve which begins at a given seed point and is everywhere tangent to the interpolated vector samples. A collection of traces can show the shape of the numerical field just as smoke or dye reveals the structure of an empirical flow. Unfortunately, images of complex flows often resemble a plate of spaghetti. As simulation technology advances and simulated flow fields become more complex, this bad situation will get worse.

Somewhat more intelligible images can be made by representing each trace, not as a curved line, but as a space-filling and shaded cylindrical tube. These "3-d space tubes" [4] carry visual cues which help a viewer interpret the three-dimensional structure represented in the two dimensional image.

Alternatively, the shape of each curve can be conveyed to the viewer by depicting it as a flat ribbon which is everywhere aligned with the tangent and binormal of the local Frenet frame [12]. Instead of fixing the orientation of the ribbon with respect to purely geometric measures of the curve, Arrott [1] related it to the local value of the helicity. This produced "twisties" which form corkscrews in highly rotational flow regions.

Belie [2] depicted this rotational information by computing pairs of particle traces from closely spaced starting positions, and constructing the ribbons which span the narrow gap between them. In divergent flow that gap can grow quite large, and so he constrained the traces during calculation to a constant separation distance.

Whatever method is used to construct ribbons, the viewer is given little aid in the task of developing an understanding of the overall structure of the flow. Individual traces and ribbons may be understandable in themselves, but the collection of shapes conveys no global meaning.

Volpe [14] has demonstrated that careful manual placement of closely-spaced sets of streamlines can depict stream surfaces which appear to a viewer as a single cohesive structure. He shows that these objects are "an excellent way of depicting streamline roll-up and vortices." The shading and self-obscuration of these surfaces provide visual cues which help one to interpret the image. Surfaces can curve both across and down their length, and can widen in divergent flow. Surfaces can be overlaid with texture to represent additional data. Finally, surfaces rendered with variable transparency can mimic the appearance of empirical flow visualizations.

3 Interactivity

The usefulness of an image is critically dependent on the placement of each surface. The key to finding good placements is interactivity. The user must be able to move a seed through the flow field and be provided with almost immediate display of the resulting trace. This greatly speeds the task of constructing a useful image, since it allows the user to quickly evaluate a large number of alternative sites. Because surfaces are composed of a number of particle traces, they are that much more time-consuming to construct and that much more difficult to place.

I present new methods which support the interactive calculation and display of stream surfaces. These ideas have been implemented in FLORA, a software package which runs on Silicon Graphics workstations. Each surface is constructed over a set of traces which are computed from positions along a seed line or "rake." The system employs progressive refinement of the displayed surface, guided by differential measures of the evolving surface and physical constraints on its construction. These guidelines also automate some of the concerns of creating a useful image, such as the "ripping" of highly divergent surfaces and the terminating of traces which exceed a given error threshhold.

For interactive use, some error in the computed surface is acceptable, as long as that error will be corrected eventually, and as long as the interim results are not radically different from the final version. When the rake is moved to a new position, the surface is initially displayed in a crude form based on a small number of traces. The approximate solution provides a guide which allows quick comparison of different seed placements. As long as the rake remains in the same place, additional traces are added to improve the accuracy of the surface.

4 Trace Calculation

PLOT3D [3] is the plotting package most widely used at NASA-Ames. It depicts flow fields using a number of different methods, including particle traces. Each trace is calculated with a second-order Runge-Kutta method through the physical coordinates (x, y, z) of the sample vector values. The interpolation scheme, however, is defined in the computational-space coordinates (ξ, η, ζ) of the given point. Letting (α, β, γ) represent the fractional parts of the computational space coordinates of a given point, the interpolant maps the vertices $\vec{X}[i, j, k]$ of the enclosing cell to the physical space coordinates of that point:

$$\begin{array}{rlllll} \vec{p} = & (1-\alpha) & (1-\beta) & (1-\gamma) & \vec{X}[0,0,0] \\ + & (1-\alpha) & (1-\beta) & (\gamma) & \vec{X}[0,0,1] \\ + & (1-\alpha) & (\beta) & (1-\gamma) & \vec{X}[0,1,0] \\ + & \ldots & & & \\ + & (\alpha) & (\beta) & (\gamma) & \vec{X}[1,1,1] \end{array}$$

In this "mixed-space" method, it is rather difficult to determine the computational-space offsets within a cell which correspond to a given point in physical coordinates. It is essentially a problem of inverting the trilinear vector equation to find (α, β, γ) for a given physical space position. Once these weights are known, they may be used to interpolate the velocity at that position.

Because this inversion of the interpolant can be so time-consuming, the integration code of PLOT3D has been installed on the Cray-2 at NASA-Ames. This numerical code can be teamed with a graphics front-end running on a Silicon Graphics workstation. This "Remote Interactive Particle-tracer" [10] combines the speed and memory capacity of the supercomputer with the graphics display hardware of the workstation to produce a highly-interactive visualization tool.

If the available memory is sufficient to store the grid and flow field data, adequate speed can be obtained on workstations by converting the vector field samples from physical space to computational coordinates. Each trace is then computed through the set of cubic cells, and the result is mapped into physical space for display. In the program STREAM3D [5], the conversion of vector samples is performed on an as-needed basis, while Volpe [14] chose to convert all the samples in a single preprocessing step. Since this "computational-space method" avoids repeated inversion of the interpolant between physical and computational space, it is much faster then the mixed-space method. A 1.5 megaFLOP workstation can compute and display a trace of several hundred points in under one second.

This speed comes at the cost of decreased accuracy. The grid transformation is rarely defined in closed-form and must be approximated by a local Jacobian matrix, which is itself approximated using stencils. These stencils tend to smooth the grid, so that when a calculated trace is mapped into physical space for display, grid discontinuities reappear as "kinks" in the traces. These errors are usually not large, and so this method can be used to support interactive manipulation of stream surfaces represented by a handful of traces.

5 High-Level Control

The task of controlling a surface to produce of useful image can be difficult; the variations are many and the results vastly different. FLORA provides mechanisms which enable the user to control the appearance of a surface by imposing limits on its construction. Since this is still very much an unexplored area, these constraints have been implemented in Scheme, a fully-general interpreted programming language. This environment provides the ability to define functions "on-the-fly" in the midst of a visualization session and supports the development of constraints of arbitrary complexity.

The construction and refinement of a surface must satisfy constraints which ensure accurate "physical" behavior. The satisfaction of physical constraints include restrictions that prevent a trace from entering a solid object. It also precludes the construction of a ribbon which has edges passing on opposite sides of an obstacle. Targets for accuracy limit the length of each trace to maintain an upper bound on the accumulated numerical error. Other constraints limit the maximum width of a ribbon; a pair of traces which diverge too far apart from one another should not be connected by large (and meaningless) triangles.

Aesthetic concerns are addressed using the same mechanism of constraints. The amount of stretch accommodated by a surface before it is torn can be specified as a limit on the rate of growth of the width of each gap. Truncation of a surface which encounters a region of high density, for example, is a potential useful visualization technique which could be specified in this environment.

6 The Construction of Ribbons

A surface is composed of a number of adjacent ribbons; each depicted on the workstation display as a sequence of triangles. A ribbon is constructed starting at the initial points of two traces. This edge is combined with the second point on one of the two traces to form the first triangle. This point and the vertex on the opposite trace form a new base edge, which is shared with the next triangle. The entire ribbon is constructed as the result of a sequence of choices which advance along the two traces, producing a new triangle at each step. The repeated choice of whether to advance along the left or the right trace can be based on any one of four different criteria. In the limiting case, with tightly-spaced traces and tightly-spaced points along each trace, the choice of construction method will make little difference in the appearance of a surface. But when a surface is approximated by a very few traces, perhaps computed with very large stepsizes, the method of construction can greatly alter the result.

The simplest method alternates sides, thereby using up point values on the two traces at an equal rate. Since the points which share the same index on two traces do not share the same time value, it would be difficult to overlay the surface with lines of constant elapsed time. Renormalizing the points along a trace so that successive points on a trace differ by a constant time increment is one solution to this problem, provided that adequately close physical spacing is maintained.

In highly sheared flow, points of equal elapsed time on two closely spaced traces may themselves be quite widely separated. Lockstep connection of these points will produce highly skewed triangles and will exhaust one trace before reaching the end of the other. Siclari [11] proposed normalizing the traces by arc length to overcome this difficulty.

In helical flow, maintaining connection on the basis of equal arc length will produce highly suspect results. The trace which lies closest to the center will be straighter than its partner, and triangles will once again span an increasingly greater distance. In this case, construction of a ribbon which maintains a minimal width may be preferable, at least until additional traces can be computed to resolve more detail in this portion of the flow. A related method [7] could be applied to derive the set of polygons which comprise the minimal surface between the two traces.

Even as simple a matter as the construction of triangles between two particle traces has several "correct" solutions. Direct control over this construction, and over other aspects of the visualization process, allows the user to produce images which are accurate and intuitively understandable representations of the flow field data.

7 Image Rendering

For interactive use, the system displays stream surfaces as shaded raster images using the display hardware of the Silicon Graphics workstation. Hard copy is available using color printers manufactured by Seiko and Tektronix. Alternatively, the surfaces can be depicted as line art described in the Postscript page description language.

Line art offers several advantages over more realistic images. It is accepted by all major journals; shaded color images are not. Information presented as line art survives photocopying and preservation as microfiche. Finally, line art can present the essential features of a structure without distraction. It is for this reason that automobile repair manuals are filled with illustrations, rather than photographs.

Line art is not supported by graphics workstations; custom software is used to produce these images. The primary task is the identification of "silhouette edges" which depend upon the shape of the surface and upon the angle of view. These edges are explicitly added to the scene to be displayed.

The image is rendered using the "painter's" algorithm [6,9]. The polygons which comprise the scene are sorted by distance from the viewer. These polygons are then displayed, starting with the furthermost and ending with those at the front of the scene, with each new polygon obscuring those which lie behind it. In some cases, a partial ordering of the polygons cannot be found and additional subdivision of some polygons must be performed.

8 System Implementation

FLORA was prototyped in the C language, and is roughly 10,000 lines of source code. It makes extensive use of the "Panel Library," [13] a software package which implements sliders, buttons, and other components of a mouse-based user interface.

To obtain maximal benefit from the interactive speed provided by this software, the user interface has been designed to be as intuitive as possible. "Point-and-click" mouse actions are used for the most common operations. To move a rake into position, the user selects it using the mouse and then drags the rake across the screen. The rake's position shifts through a plane orthogonal to the current viewing direction. While the rake is in motion, traces attached to that rake are computed and displayed at interactive frame rates.

The constructed surfaces can be displayed in a number of separate windows; each with its own viewing angle and magnification. Users often move a rake in one window at high magnification, while viewing the resultant surface from another angle in a second window. This approach allows the precision placement of several rakes in a few minutes; a vast improvement over what has previously been possible.

Current efforts are aimed at combining the central numerical kernel of the prototype with a reconfigurable user interface implemented in Scheme, a dialect of LISP. The intent is to combine the best features of each language into a single program. The compiled C modules provide the speed required for calculation and display of surfaces. The interpreted Scheme environment simplifies the implementation of those features which demand flexibility, such as the definition of surface constraints and the specification of surface colors and light sources.

9 Results

The accompanying figures depict flow field data computed by Hung and Buning [8]. This is a symmetric flow about a vertical blunt fin. The grid and the surfaces have been reflected about the midline so that two views of the data may be presented in a single image. The first figure shows the primary horseshoe vortex using traditional particle traces. In the second figure, the vortex is depicted far more clearly when ribbons are constructed between each adjacent pair of traces. A wider surface presents a somewhat different view of flow field in figure three, and user controlled trimming of that surface produces the result shown in figure four.

A system has been built which can compute and display stream surfaces in response to a user's interactive command. These surfaces can be used to depict a flow field more accurately and intuitively than has been possible with particle traces and flow ribbons. Line art depiction of the surfaces for publication is supported.

This work has demonstrated that with cautious allocation of system resources modern graphics workstations can support highly complex tasks. Response time for each user command is kept minimal, even if this requires the presentation of highly simplified results. These crude initial results are sufficient to allow the user to identify those regions of the flow field which merit attention. Eventual refinement of the surface ensures that accurate results are obtained without distracting the user with unpredictable delays.

10 Acknowledgements

I wish to thank Pieter Buning for the use of his fluid flow data. I also am indebted to Creon Levit, Eric Raible, Sam Uselton and Mike Yamasaki for their technical advice. Tom Lasinski and Fred Brooks lent their valuable guidance, and Mary Hultquist has always provided constant encouragement and support.

11 References

1. M. Arrott. Private communication.

2. R.G. Belie. "Flow Visualization in the space shuttle's main engine," *Mechanical Engineering* **9**, 27–33, (1985).

3. P.G. Buning and J.L. Steger. "Graphics and flow visualization in computational fluid dynamics," *AIAA Paper 85-1507*, AIAA CFD Conference, Cincinnati OH, July 1985.

4. R.R. Dickinson. "A unified approach to the design of visualization software for the analysis of field problems," Three-dimensional Visualization and Display Technologies, *SPIE* **1083**, 173–180, (1989).

5. P. Eliasson, J. Oppelstrup, and A. Rizzi. "STREAM3D: computer graphics program for streamline visualization," *Advanced Engineering Software* **11**, 162–168, (1989).

6. J.D. Foley and A.Van Dam. *Fundamentals of Interactive Computer Graphics*, Addison-Wesley, Reading, MA, 1984.

7. H. Fuchs, Z. Kedem, and S. Uselton. "Optimal surface reconstruction from planar contours," *Communications of the ACM* **20**, 693–702, (1977).

8. C.-M. Hung and P.G. Buning. "Simulation of blunt-fin induced shock wave and turbulent boundary layer separation," *AIAA Paper 84-0457*, AIAA Aerospace Sciences Meeting, Reno NV, January 1984.

9. M.E. Newell, R.G Newell, and T.L. Sancha. "A new approach to the shaded picture problem," Proceedings of the ACM National Conference, 443–?, (1972).

10. S.E. Rogers, P.G. Buning, and F.J. Merritt. "Distributed interactive graphics applications in computational fluid dynamics," *International Journal of Supercomputing Applications* **1**, 96–105, (1987).

11. M. Siclari. Private communication, regarding the image on the cover of *Science* **245**, 333-440, (28 July 1989).

12. M.F. Smith. "On application of computer graphics and animation to the analysis of steady and unsteady CFD data sets," unpublished Master's thesis, University of Colorado at Boulder, 1989.

13. D.A. Tristram and P.P. Walatka. "Panel library programmers' user manual," NASA-Ames, NAS division preprint, April 1989.

14. G. Volpe. "Streamlines and streamribbons in aerodynamics," *AIAA Paper 89-0140*, AIAA Aerospace Sciences Meeting, Reno NV, January 1989.

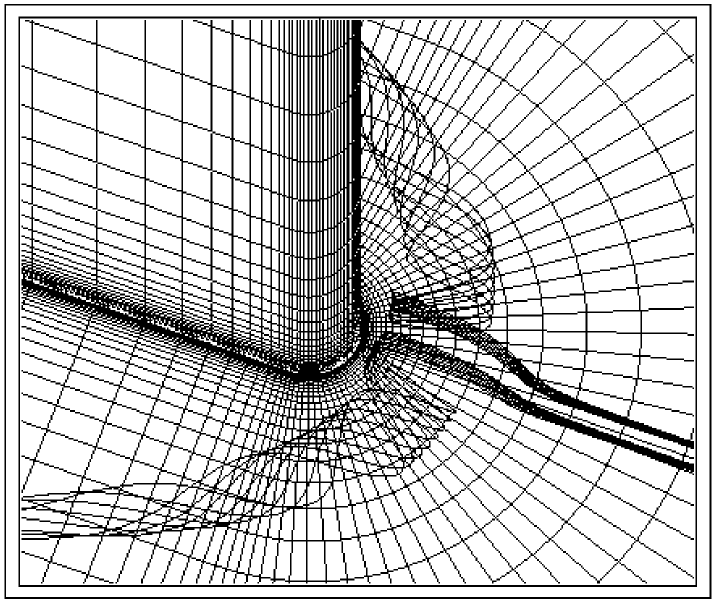

Figure 1: Particle traces about a vertical blunt fin.

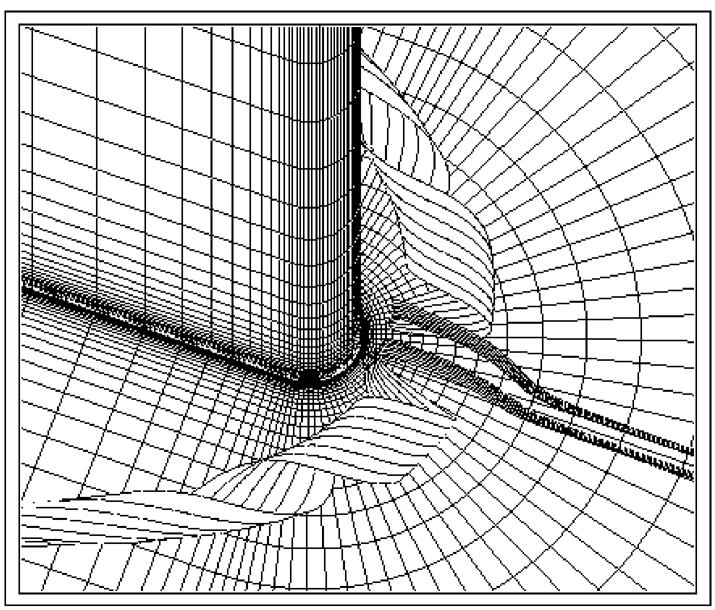

Figure 2: Narrow surface constructed on the same traces.

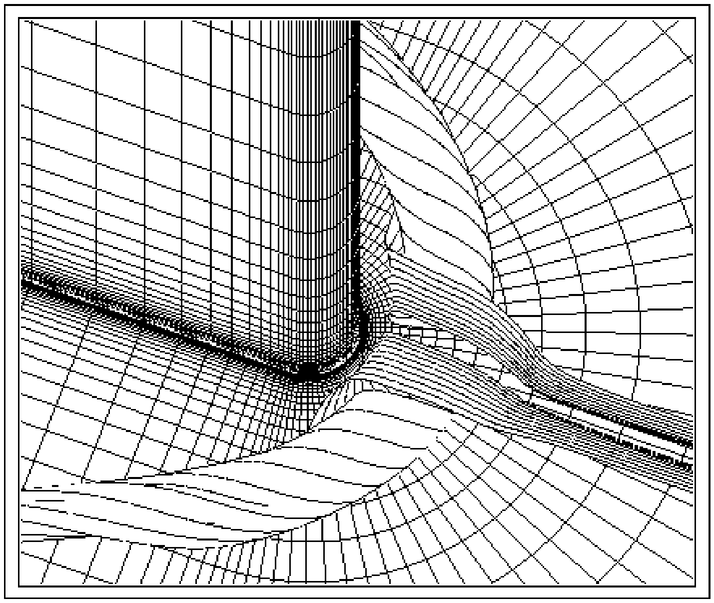

Figure 3: Wide surface enclosing the primary vortex.

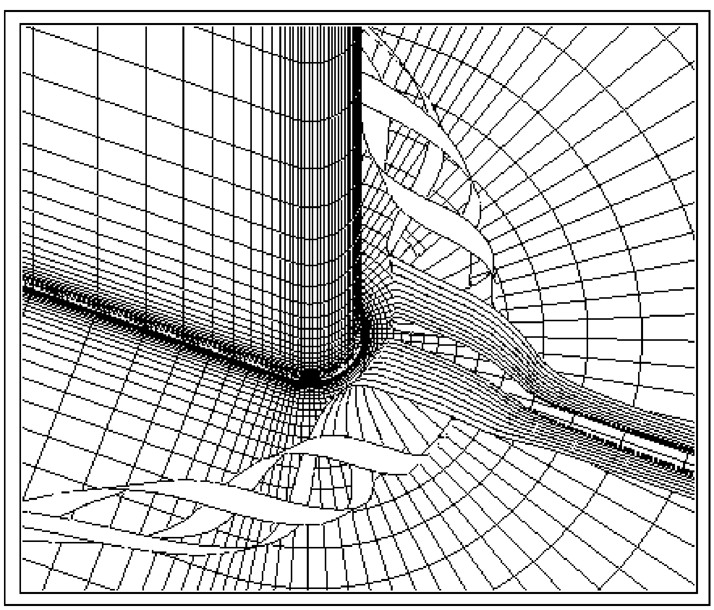

Figure 4: Surface limited by user-defined constraints.