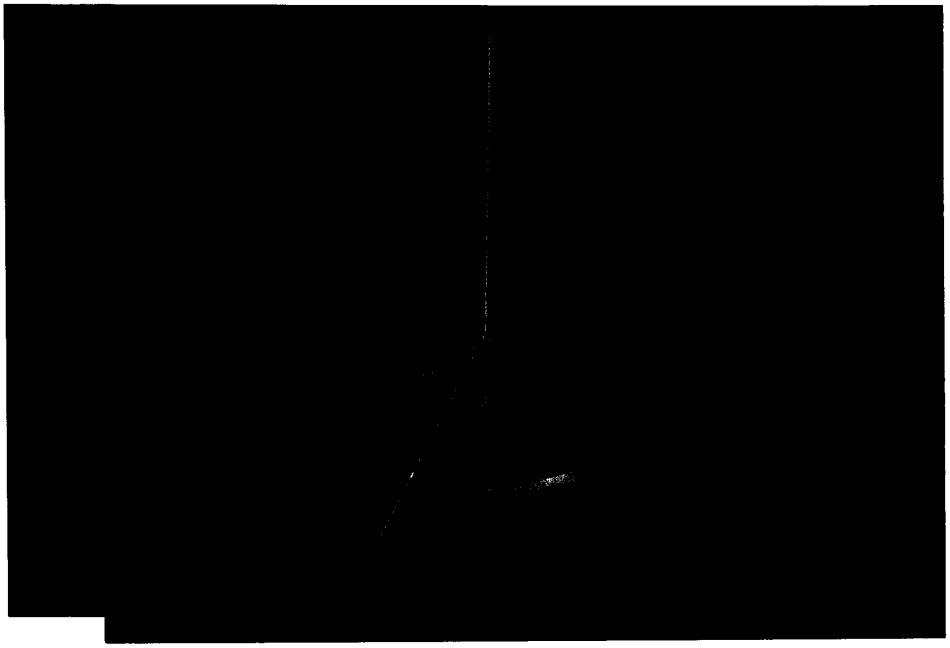


# Visualizing Vector Field Topology in Fluid Flows

James L. Helman and Lambertus Hesselink
Stanford University

*Using techniques to extract and visualize topological information, we can combine the simplicity of schematic depictions with the quantitative accuracy of curves and surfaces computed directly from the data.*

When scientists introduce computer graphics to a field of study, the first visualization techniques to emerge are the ones that most closely resemble the pictures already familiar to those in the field. To the researcher, who has seen thousands of them and has learned to interpret them, such images may be more useful than a new representation that actually contains more information. What x-ray images are to radiologists, and multiple needle strip charts are to seismologists, so oil streak patterns and smoke visualizations are to fluid dynamicists. The capabilities of the first generation of numeric flow visualization packages, such as Plot3D[1] reflect this approach. To visualize oil film patterns on the surface of a body in a flow, scientists integrated the tangential flow near the surface to generate curves on the body wall. To duplicate smoke visualizations (albeit without mass and diffusion), scientists integrated massless particles through the flow to generate streamlines.

We have developed methods to automate the analysis and display of vector field topology in general and flow topology in particular. The importance of topology in understanding fluid dynamics,[2-4] combined with the difficulty of extracting topological information with existing tools, motivated our efforts.



0272-17-16/91/0500-0036$01.00 ©1991 IEEE

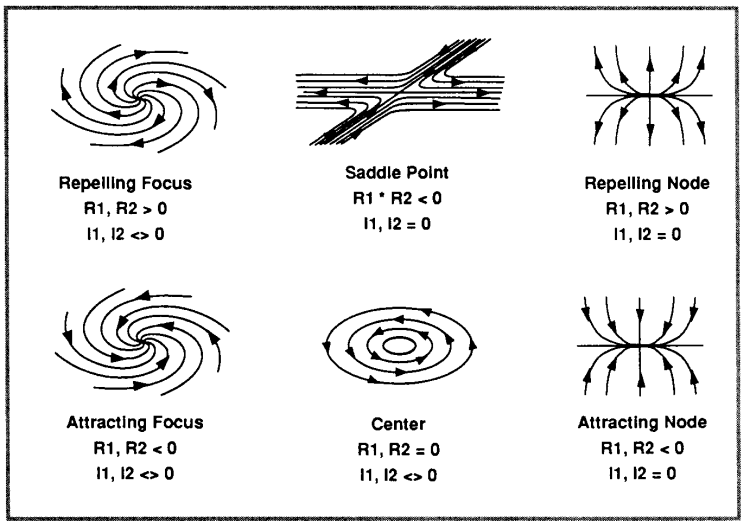



**Figure 1. Classification criteria for critical points. R1 and R2 denote the real parts of the eigenvalues of the Jacobian, I1 and I2 the imaginary parts.**

## 2D vector field topology

First, let's look at 2D vector field topology, the basis for the examination of topology in 3D separated flows.

Topological concepts are very powerful because, given the critical points in a vector field and the tangent curves or surfaces connecting them, you can infer the shape of other tangent curves and hence to some extent the structure of the entire vector field.

We can think of flow topology in terms of surfaces (in 3D domains) or curves (in 2D domains) that divide the flow into separate regions. Two sets of surfaces or lines are of particular interest[5]:

1. Tangent surfaces that actually intersect the wall of a body where the flow attaches to or separates from that wall. Tangent curves on either side are deflected, moving along the surface of the body.
2. Surfaces where tangent curves that start arbitrarily close to each other can end up in substantially different regions.

The first group of curves is related to the surface topology, and the second group is related to critical points both on walls and in the external flow. For example, a saddle point diverts tangent curves that pass directly into it to very different regions.

### Critical points

*Critical points* are points at which the magnitude of the vector vanishes. These points can be characterized according to the behavior of nearby tangent curves. We can use a particular set of these curves to define a skeleton that characterizes the global behavior of all other tangent curves in the vector field. If we consider the Taylor series expansion of the field in the neighborhood of such a point, then the first order partial derivatives of the field (with respect to position) determine the vector field's behavior. Thus, for a nondegenerate critical point $(x0, y0)$, we can use the matrix of these derivatives—that is, the Jacobian matrix—to characterize the vector field and the behavior of nearby tangent curves:

$$\left[\frac{\partial(u, v)}{\partial(x, y)}\right]_{x0, y0} = \begin{bmatrix} \frac{\partial u}{\partial x} & \frac{\partial u}{\partial y} \\ \frac{\partial v}{\partial x} & \frac{\partial v}{\partial y} \end{bmatrix}_{x0, y0} \quad \textbf{(1)}$$

The eigenvalues and eigenvectors of this matrix are of particular interest. A real eigenvector of the matrix defines a direction such that if we move slightly off the critical point in that direction, the field is parallel to the direction we moved. Thus, at the critical point, the real eigenvectors are tangent to the trajectories that end on the point. The sign of the corresponding eigenvalue determines whether the trajectory is incoming (attracting) or outgoing (repelling) at the point. The imaginary part of an eigenvalue denotes circulation about the point.

Figure 1 shows how the eigenvalues classify a critical point as an *attracting node*, a *repelling node*, an *attracting focus*, a *repelling focus*, a *center*, or a *saddle*. Among these points, the saddle points are distinct in that there are only four tangent curves (two for each real eigenvector) that actually end at the point itself. At the saddle point, these curves are tangent to the two eigenvectors of the Jacobian matrix, which are the separatrices of the saddle point. The outgoing and incoming separatrices arearallel to the eigenvectors with positive and negative eigenvalues, respectively.

In addition to these 2D critical points, certain points on the walls of objects or bodies in a fluid flow can be important. On walls where the velocity is constrained to be zero (a no-slip boundary in fluid dynamics), certain points might occur. We refer to these points as *attachment nodes* or *detachment nodes*, at which a tangent curve impinging on the surface terminates on the surface, rather than being deflected by the tangential velocity.

The saddle points and attachment and detachment nodes differ from the other points in two regards. First, only a finite number of tangent curves (two for saddle points and one for the attachment/detachment nodes) end on the point itself. Second, the curves adjacent to these particular curves diverge at the critical point, which makes these curves significant to understanding the global behavior of other tangent curves. These curves connect the various critical points into a skeleton that represents the global topology of the 2D vector field, thus providing a very effective simplification for complex fields. (We

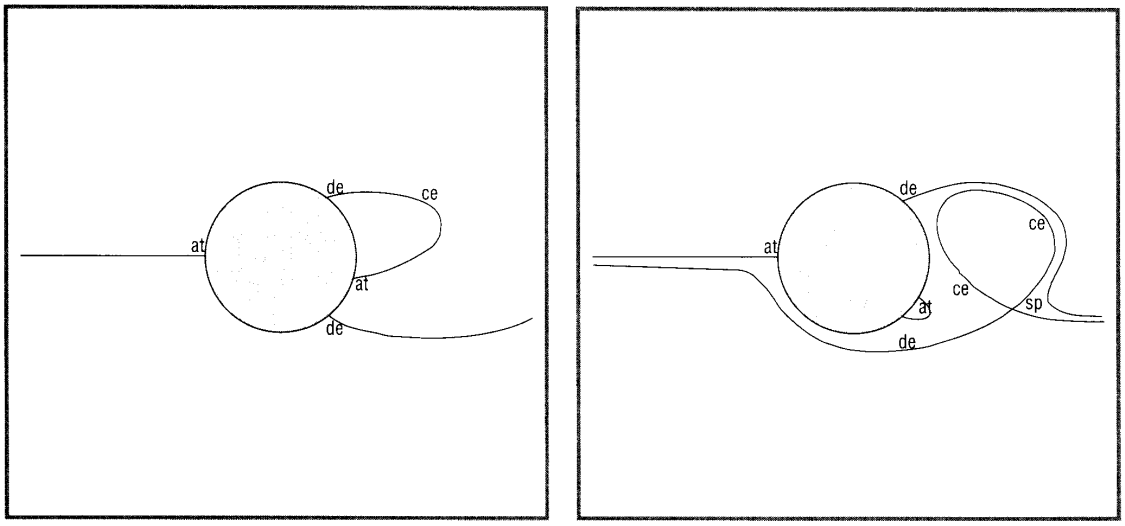



**Figure 2. Topology schematics for two time steps in the computed flow around a circular cylinder.**

discussed the details of the analysis and generation of the representations elsewhere.[6])

Figure 2 shows the topology skeletons generated for two time steps in a computed 2D flow around a circular cylinder.[7] The flow is incident from the left, with one instantaneous streamline ending directly on the front of the cylinder. Points *de*, *at*, *ce*, and *sp* denote detachment, attachment, center, and saddle points, respectively. All instantaneous streamlines starting above that curve are deflected over the top of the cylinder, and those starting below are deflected beneath it. Vortex shedding occurs behind the cylinder, as indicated by the detachment-attachment "bubble" in the first skeleton, which develops into a paired saddle and center in the second skeleton.

**Figure 3. Topological surfaces depicting the time evolution of the computed 2D flow past a circular cylinder.**

## 2D time-dependent flows

When a 2D vector field depends on time or another parameter, we can link the instantaneous topology skeletons together to denote the time evolution of the flow. We join the adjacent skeletons by linking their corresponding points and tangent curves. This provides a representation of the time development of the topologies that we can use to examine the formation of structures and locate topological transitions.

After we have linked the instantaneous slices together, we can display the set of stacked topological representations as a set of surfaces, with the third dimension corresponding to time. We create the surfaces by tesselating strips between corresponding tangent curves in adjacent slices of the representation.

Figure 3 shows the surfaces in the periodic flow around a 2D circular cylinder. Time increases from back to front along the cylinder. The display uses several cues to aid visualization. We light and shade the surfaces, then color them according to their type. Surfaces corresponding to the incoming separatrix of a saddle point are colored yellow. Those surfaces corresponding to the outgoing direction are blue. Orange indicates surfaces from attachment points, while purple indicates surfaces from detachment points. You can see the periodic vortex shedding in the repeated development and movement downstream of saddle-center pairs.

# Topology in 3D separated flows

A primary purpose of our 2D work was to develop techniques that we could extend to the study of 3D separated flows. In these flows some stream surfaces near the surface of a body can abruptly move away and "separate" from the wall. The lines at which this occurs are known as *lines of separation*. These lines

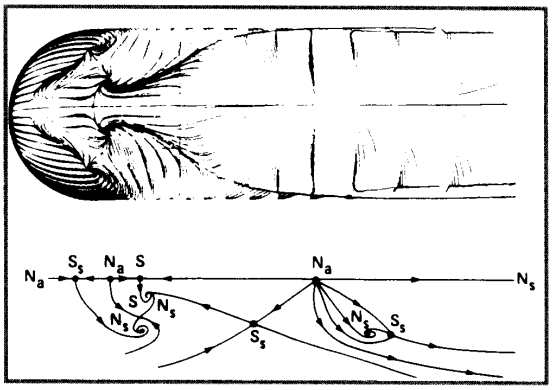



**Figure 4. (a) Surface particle traces in the computed flow past a hemisphere cylinder. (b) Corresponding manually generated schematic interpretation of surface topology. (Both from Ying, Schiff, and Steger.[8])**

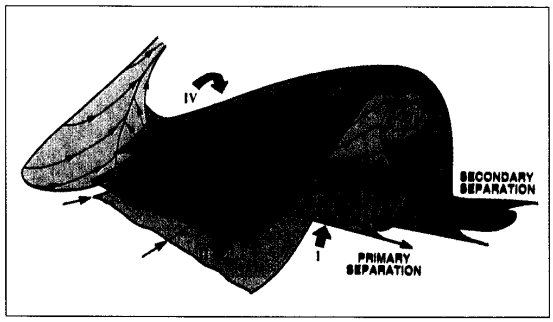



**Figure 5. Hand-drawn surfaces depicting separation topology for the computed flow past a hemisphere cylinder. (From Ying, Schiff, and Steger.[8])**

are the 3D extensions of the attachment and detachment nodes used in the 2D analysis. Namely, these are curves on a body wall that are the terminuses for tangent surfaces. Tangent surfaces on either side of the separation or attachment surface are deflected along the surface of the body.

Because we often associate separation surfaces with vortices and recirculation zones, determining separation topologies is important both for understanding fundamental fluid dynamics and practical applications in aircraft and jet nozzle design. But extracting topological information from numerical data sets using existing visualization tools is both difficult and time consuming. Typically, a researcher would try to determine the topology and the positions and shapes of the structures by looking at numerically integrated streamlines constrained to the surface (as in Figure 4a) and in the volume. The latter often become visual spaghetti, difficult to interpret in the neighborhood of vortices where the tangent curves swirl about each other. By manually selecting and refining integration starting points, the researcher can discern structures and connections. However, since topological structures are often complex and best portrayed graphically, the researcher must then draw those structures by hand (as in Figures 4b and 5) to capture their form, if not their exact shape, size, and position. Automatic methods for producing these schematic surfaces would simplify the work, eliminate manual errors, and, most importantly, accurately preserve and convey the quantitative aspects of the structures. We've developed one such automatic approach.

## Surface topology

In general, when examining the topology of a flow, we examine the surfaces of bodies in the flow first. In experimental work, researchers do this by examining the streaks that form in an oil film on the surface of a body in a wind tunnel. In computer simulated flows, we can derive similar information by examining curves integrated along the surface[8] (as in Figure 4a) to produce topology skeletons like the one shown in Figure 4b.

We can automatically generate the surface topology skeleton by applying the 2D algorithm outlined above to the tangential velocity field near the body. Grids in these data sets conform to the shape of the body, with one of the grid planes lying on the surface of the body. The velocity on this plane is zero. To analyze the surface topology, we create the 2D vector field that is the projection onto the surface of the velocities in the grid plane that are one point away from the surface. If the body is defined by the $k = 0$ grid plane, we compute the new 2D field $(u'(i, j), v'(i, j))$ as the projection of the 3D velocity $(u(i, j, 1), v(i, j, 1), w(i, j, 1))$ onto the plane tangent to the body at $(i, j, 0)$.

Applying the same algorithm that generated the skeletons in Figure 2 to this field produces the surface topology skeleton in Figure 6. Here we have labeled the critical points according to the sign of the normal component. *Ss* denotes a saddle of separation (normal velocity positive), and *Sa* denotes a saddle of attachment (normal velocity negative).

The positions and connections Ying, Schiff, and Steger[8] hypothesized (Figure 4b) correspond well to those in this skeleton. The scale distortions result from the use of grid index coordinates in Figure 6 rather than unrolled physical coordinates. Figure 7 shows these curves projected onto the surface of the hemisphere cylinder body. As indicated by the arrows, curves moving toward saddle points are yellow; curves moving away from those points are blue.

The one minor difference between these figures is the position of the point $rn - 1$ (and it's symmetry reflection $rn - 4$). We can attribute the difference to the coarse resolution of the grid that the flow solver used in this downstream region. The data set contains the topology shown in Figure 6, but the physically correct solution has the postulated topology shown in Figure 4. The ability of this type of visualization to reveal anomalies

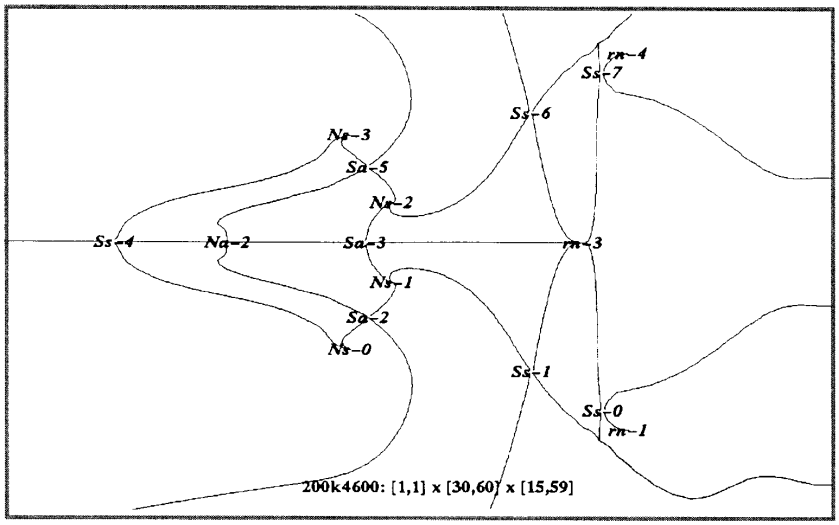



**Figure 6. Computer-generated skeleton of surface topology corresponding to Figure 4.**

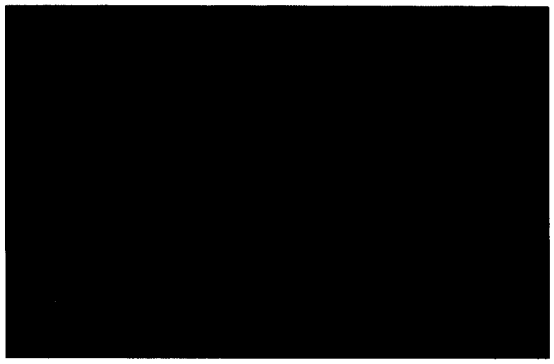


**Figure 7. Surface topology skeleton shown on the body surface with a parameter space clipped stream surface.**

**Figure 8. Stream surfaces depicting separation topology.**

otherwise difficult to detect illustrates their potential value in tuning grids and solvers.

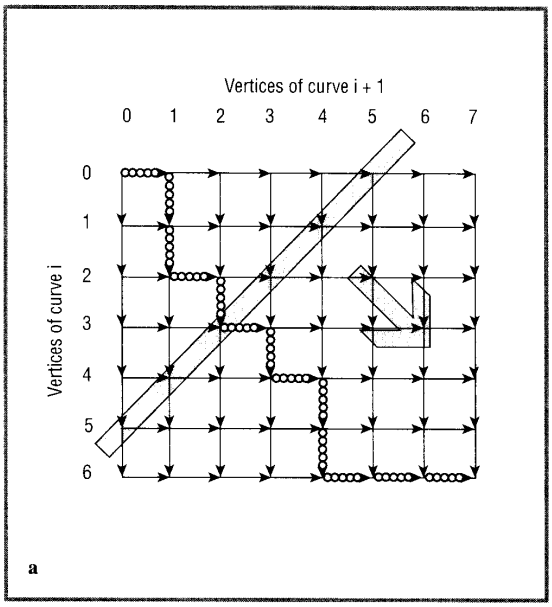
### 3D separation structures

As shown in Figure 5, the saddles of separation on the surface generate complex *surfaces of separation* in the external flow. These points, which are saddles in the tangent plane, are repelling nodes in the plane normal to the surface and parallel to the outgoing separatrix (the blue line) of the saddle point. The stream surface generated by this repelling node is the surface of separation. We can see this in the surface generated by the saddles of separation labeled $S_s$ in Figure 5. In principal, we can generate this entire stream surface by starting curve integrations in this plane in the neighborhood of the critical point. But note the manner in which the curves move away from the surface along separation curves labeled I and III downstream from the saddles. The normal velocity along these lines of separation on the body is sufficiently large that, given reasonable grid spacings, it is impossible to start integrations close enough to the surface so that they will remain near the surface downstream. Thus, in practice, we cannot generate the surfaces of separation solely from integrations in the neighborhood of the saddles of separation.

However, from our integrations of the 2D tangential field, for each saddle we know the corresponding separation curves on the surface (the blue curves in Figure 7). Since analytically, as the starting point gets arbitrarily close to the body, the integrated curve should approach the separation curve, the separation curve can provide starting points for integrating curves into the external flow. In Figure 8, we show the results of integrating stream lines along two separation curves and tesselating them with a surface.

## Tangent surfaces

In both two- and three-dimensional vector fields, this work requires us to construct representations of tangent surfaces that are accurate as well as efficient to compute and display. We must balance the accuracy of the surface against the time required to compute and display a higher resolution depiction. To do this effectively, we must make the best use of a limited number of tangent curves to define the surface. Hence, determining what surface to draw (given a number of known tangent curves) and choosing the number and location of tangent curves to be integrated are important.



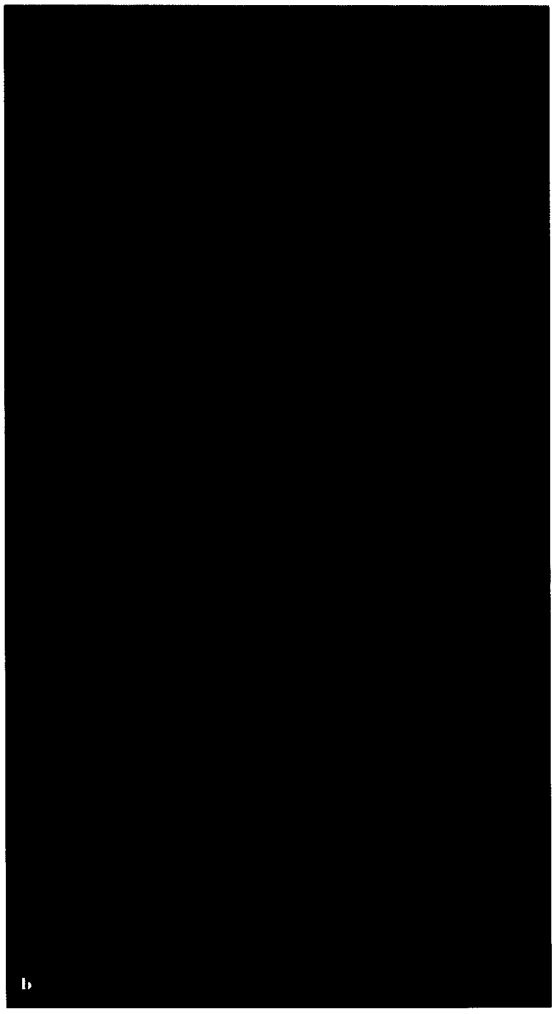




**Figure 9. Minimal surface generation. (a) Directed graph used in optimal surface generation. Diagonal box and arrow denote the search order. (b) Skeleton of triangular mesh strips in a vortex. Two strips (three curves) in a skewed mesh before minimization (left) and after minimization (right).**

## Tesselation

To generate surfaces from adjacent curves, we must define a polygonal tesselation. Since each of the surfaces including the body typically contains 7,000 to 30,000 triangles, drawing time is a concern. To maximize the drawing rates on our graphics platform (a Silicon Graphics 4D/220GTX), we tile all surfaces with triangular meshes. Because we defined our surfaces in terms of adjacent tangent curves, this tiling appears quite straightforward. If the number of vertices in each curve is the same (if not, it can be reinterpolated), then we define a mesh by going through the two curves, alternating their corresponding vertices. This is only one of many possible tesselations.

Given two curves $C_1$ and $C_2$, defined as a series of line segments connecting the vertices $C_1(i) : 0 < i < n - 1$ and $C_2(j) : 0 < j < m - 1$, there are $(m + n)! \setminus (m!n!)$ different triangular tesselations of the surface consistent with those line segments. The correct surface is the one that comes closest to the surface defined by all the other intermediate curves that we did not integrate. We can compute enough of these curves so that their density exceeds the display resolution, but that is impractical.

A natural surface to choose is one that minimizes the surface area (as a soap bubble does). To do this efficiently, we adapted Fuchs, Kedem, and Uselton's algorithm,[9] a method developed for tesselating closed contours in parallel cross sections. In their algorithm,

1. each segment between the curves defining the edge of a triangle corresponds to a vertex in a directed graph, and
2. each triangle corresponds to an edge in the graph (as in Figure 9a).

By assigning a cost to each edge in the graph, we reduce finding an optimal tesselation to the problem of finding a least-cost path through the graph from the vertex (0, 0) to the vertex $(n, m)$. We can find such a path quite simply by computing the

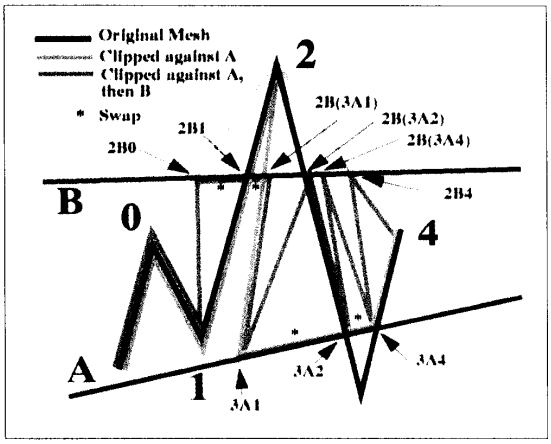



**Figure 10. Clipping new triangular mesh.**

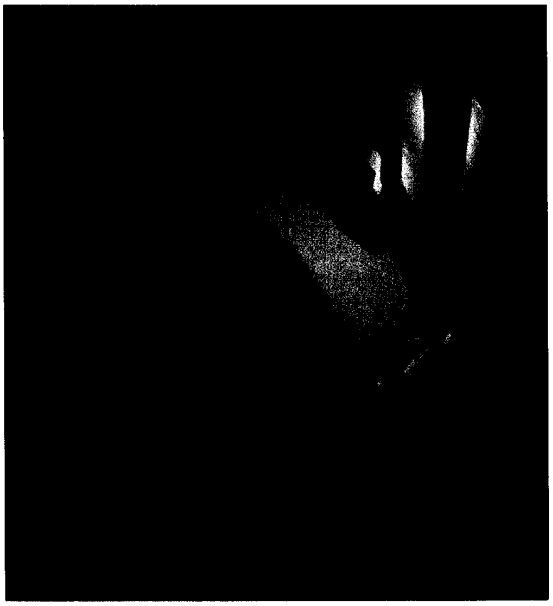


**Figure 11. Cross sectional view of a new triangular mesh showing interior structure of vortical stream surfaces produced using physical space clipping planes.**

lowest cost path to each vertex along the diagonal $i+j=C$ starting with $C=1$ and ending with $C=m+n-2$. At each of these vertices, we must compute the cost of the triangle created by moving from either of two vertices on the previous diagonal (or only one previous vertex when the current vertex is on a border of the graph) to the vertex under consideration. Finding a path in this way requires $O(mn)$ operations.

Figure 9b shows how the algorithm reduces the slenderness of triangles in a mesh generated using fractional arc length parameterization between the original curves. We show two strips, one purple and one orange, defined by three tangent curves. In the resulting tesselation, the algorithm substantially reduces the displacement between vertices on adjacent curves.

This global optimal path determination is computationally intensive, but we haven't found any local heuristic or faster global tesselation method that works as well for defining approximate tangent surfaces from a limited number of tangent curves.

## Clipping

The complexity of stream surfaces, especially in regions where the flow is "swirling" near foci or vortices, causes much internal structure to be hidden when we display the entire surface (as in Figure 8). We frequently use clipping to reveal interior structures. In our case, two types of clipping have been useful separately and in combination, namely physical space and parameter space clipping.

### Physical space clipping

Clipping planes in the physical coordinates of the data set lets us see the shape and internal structure of features close to the surface of the body. Figure 10, for instance, shows the cross section of the vortices.

To keep drawing times as low as possible, we draw the clipped surfaces as triangular meshes. Generating a new mesh, storing it in memory, and then drawing it can be inefficient when the clipping planes change with each frame or when memory is limited. This is an important consideration when forming animated sequences by moving the clipping planes themselves. For these reasons, we developed a method for clipping a triangular mesh against several clipping planes while drawing.

The algorithm for doing this clipping is similar to Maillot's,[10] except that

1. we perform the clipping while drawing, rather than producing a complete, new mesh in memory, and
2. we recursively clip each triangle against all clipping planes; we use the stack to provide the temporary storage for partially clipped results.

The algorithm is straightforward once we realize that we can clip each triangle independently. While still maintaining the proper mesh ordering, we can clip each triangle without knowing how the previous triangle was clipped.

In Figure 11 we show an example of the clipping of a triangular mesh against two clipping planes (A and B). We denote new points according to intersections with clipping planes. For example 3A1 denotes the intersection of segment 31 with clipping plane A. Pseudocode for the algorithm appears in Figure 12.

```
SegInter(v0), v1, plane):      Returns intersection of segment v0 v1 with plane

ClipType(v0, v1, v2, plane):   Returns 3 bits indicating the in/out status of each of
                               the three vertices with respect to the plane

BeginTriangles():              Starts new mesh strip, sets Drawing to TRUE

EndTriangles():                Ends current mesh strip, sets Drawing to FALSE

DrawVertex(v0):                Draws vertex in mesh

SwapMesh():                    Reverses mesh ordering of previous two vertices drawn

ClipTriangle(v0, v1, v2, plane)
{
local 11, 12;
plane ←plane + 1;
if ( plane > nplanes ) then begin
  if ( !Drawing ) then begin
     BeginTriangles()
     DrawVertex(v0)
     DrawVertex(v1)
  end;
  DrawVertex(v2)
else
  select ClipType(v0, v1, v2, plane) from
    0: begin v0 in, v1 in, v2 in
      ClipTriangle(v0, v1, v2, plane)
    end;
    1: begin v0 in, v1 in, v2 out
      ClipTriangle(v0, v1, l1 ←SegInter(v0, v2, plane), plane)
      SwapMesh()
      ClipTriangle(l1, v1, l2 ←SegInter(v1, v2, plane), plane)
    end;
    2: begin v0 in, v1 out, v2 in
      SwapMesh()
      ClipTriangle(l1 ←SegInter(v0, v1, plane), v0, l2 ←SegInter(v2, v1, plane), plane)
      ClipTriangle(v0, l2, v2, plane)
    end;
    3: begin v0 in, v1 out, v2 out
      ClipTriangle(v0, l1 ←SegInter(v0, v1, plane), l2 ←SegInter(v0, v2, plane), plane)
      EndTriangles()
    end;
    4: begin v0 out, v1 in, v2 in
      ClipTriangle(l1 ←SegInter(v1, v0, plane), v1, l2 ←SegInter(v2, v0, plane), plane)
      SwapMesh()
      ClipTriangle(l2, v1, v2, plane)
    end;
    5: begin v0 out, v1 in, v2 out
      ClipTriangle(l1 ←SegInter(v1, v0, plane), v1, l2 ←SegInter(v1, v2, plane), plane)
    end;
    6: begin v0 out, v1 out, v2 in
      ClipTriangle(l1 ←SegInter(v2, v0, plane), l2 ←SegInter(v2, v1, plane), v2, plane)
    end;
    7: /* v0 out, v1 out, v2 out */
    endcase;
 end;
}
```

**Figure 12. The clipping pseudocode.**

### Parameter space clipping

Clipping planes in a parameter space of the surface reveal a different aspect of hidden surfaces. In our approach, the fractional length down each curve and the number of curves down a surface provide a natural pair of parameters. Figure 7 demonstrates the application of this sort of clipping to reveal the stream surface near a vortex core.

## Refinement and stability

When determining how many tangent curve integrations to launch from a seed curve, we choose an increment and start integrations at intervals with that spacing along the seed curve. However, adjacent curves started in this manner may diverge downstream. Usually, this divergence is small. We can approximate the surface between the two curves by examining the distance between the integrated curves, then recursively refining the surface by introducing additional, intermediate starting points until the maximum distance between the curves falls below some specified tolerance.

As you can see in Figure 8, at several places, especially on the surface near the nose, the starting points have widely varying spacings. This is a consequence of the successive refinement outlined earlier. However, the curves integrated with starting points along the primary line of separation curl up into two distinct vortices. Near the place where the surface "splits," integrated curves exhibit extreme sensitivity to initial conditions (see Figure 13). In the absence of a singularity, a continuous surface should join the two parts of the split surface. However, the extreme sensitivity to initial conditions makes it impossible to generate the surface by merely introducing new integration starting points along the original curve. In this case, doing so repeatedly reduces the precision below an acceptable level and introduces unacceptably large errors into the integration. We need a method other than direct integration to resolve the tangent surface in this region.

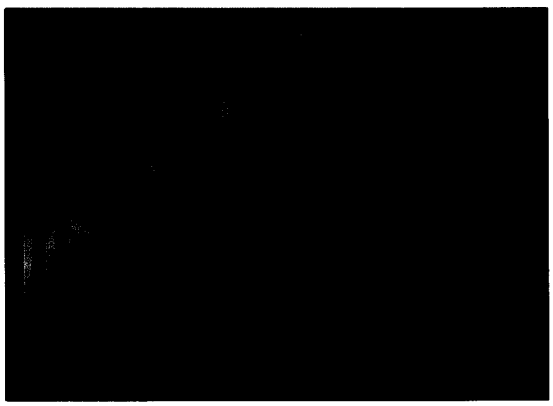


**Figure 13. Closeup of separation surfaces showing divergence of integrated curves.**

## 3D critical points

The zeros of a 2D vector field give rise to the basic set of points shown in Figure 1. In 3D, the zeros of a vector field are

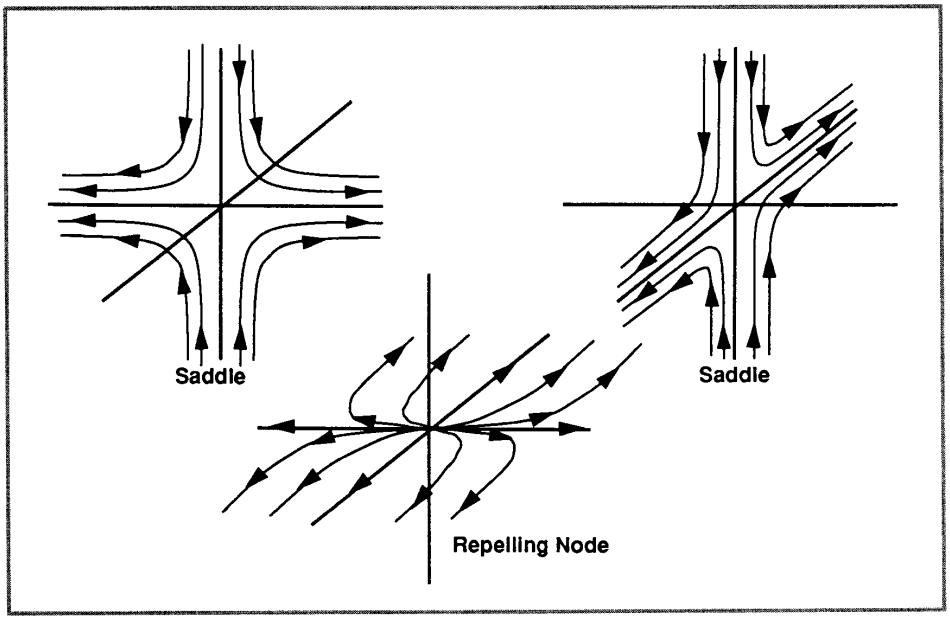



**Figure 14. A 3D saddle/saddle/repelling node depicted by tangent curves in its principal planes.**

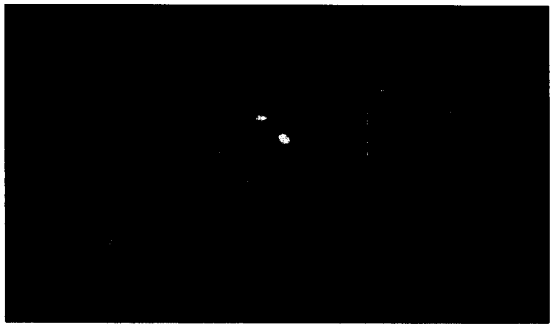


**Figure 15. Several critical points with their eigenvectors and eigenvalues displayed as arrows and disks.**

critical points that we can classify by simple generalization[11] of the 2D classification shown in Figure 1. Alternately, we can classify the critical points by examining the invariants of the matrix.[12]

In the case of a 3D vector $(u, v, w)$ field over a 3D domain $(x, y, z)$, the Jacobian matrix is $3 \times 3$:

$$\left[\frac{\partial(u, v, w)}{\partial(x, y, z)}\right]_{x0, y0, z0} = \begin{bmatrix} \frac{\partial u}{\partial x} & \frac{\partial u}{\partial y} & \frac{\partial u}{\partial z} \\ \frac{\partial v}{\partial x} & \frac{\partial v}{\partial y} & \frac{\partial v}{\partial z} \\ \frac{\partial w}{\partial x} & \frac{\partial w}{\partial y} & \frac{\partial w}{\partial z} \end{bmatrix}_{x0, y0, z0} \quad (2)$$

Thus, three eigenvalues and three eigenvectors are possible. The meanings of the eigenvalues are the same: A positive real part signifies a repelling direction, a negative real part signifies an attracting direction, and an imaginary part denotes circulation.

Complex eigenvalues always occur in conjugate pairs. Two complex eigenvalues are always paired with a real one. Therefore, we can have foci or centers that are attracting or repelling in the third dimension.

For critical points with all real eigenvalues, all three can have the same sign, producing a purely repelling or purely attracting node that appears as a 2D node in each of the three planes spanned by pairs of eigenvectors. Alternately, two eigenvalues can have one sign while the third has an opposite sign; this produces a *saddle/saddle/node*, a point that appears in two planes as a saddle and as a node in the third. We show one type of saddle/saddle/node in Figure 14.

## Locating and characterizing

In a numerical data set, we can locate 3D critical points by searching for zeros of the field. We compute the matrix of the first derivatives of the field with respect to physical space for the point, then solve for the eigenvalues and eigenvectors.

## Display

As shown in Figure 15, sets of arrows and disks can display the critical points. The arrows point in the direction of the real eigenvectors, with length proportional to the scaled eigenvalue and color denoting the sign of the eigenvalue (yellow is negative, blue is positive). The disks are in the plane spanned by the complex eigenvectors. The diameters of different colored disks are proportional to the real and imaginary parts of the scaled eigenvalue. Here, the dark blue or yellow disks represent the real part; the light blue or red ones represent the imaginary part.

Because the eigenvalues can range over several orders of magnitude, we need a scaling function to display them in this manner. We use the function $f(x) = x^a$ where $a$, $0 \le a \le 1$, is specified through the graphical user interface at display time. In Figure 14, $a = \frac{1}{2}$.

# Conclusions

When we process and extract the desired structural information before visualizing it, we reduce a large volume of data to a manageable amount containing the most relevant aspects. (In our case, the surface representation requires about one tenth the storage of the full data set.) By extracting before rendering, our approach also eliminates tedious manual interpretation and produces images that retain more quantitative information than traditional hand-drawn ones. For vector fields in particular, graphical depictions consisting of critical points and the curves and surfaces that connect them are an effective means of communicating information about the topology. ❑


## Acknowledgments

We thank Brian Cantwell of Stanford, Stuart Rogers and Lewis Schiff of NASA Ames Research Center for providing us with interesting data sets, and John Recker of Sun Microsystems for the timely insight that independently clipping meshed triangles doesn't disrupt the mesh.

NASA supports this work under contract NAG-2-489-S3, including support from the NASA Ames Numerical Aerodynamics Simulation Program and the NASA Ames Fluid Dynamics Division. The computing and graphics facilities are partially supported by the National Science Foundation under grant ECS 8815815.


## References


1. P. Buning and J. Steger, "Graphics and Flow Visualization in Computational Fluid Dynamics," *Proc. AIAA 7th Computational Fluid Dynamics Conf.*, American Institute of Aeronautics and Astronautics, New York, 1985, pp.162-170.
2. M. Tobak and D.J. Peake, "Topology of 3D Separated Flows," *Ann. Rev. Fluid Mechanics*, Vol. 14, 1982, pp. 61-85.
3. U. Dallmann, "Topological Structures of 3D Flow Separation," Tech. Report DFVLR-IB 221-82 A 07, German Research Institute for Aerospace, Göttingen, Germany, Apr. 1983.
4. A.E. Perry and M.S. Chong, "A Description of Eddying Motions and Flow Patterns Using Critical Point Concepts," *Ann. Rev. Fluid Mechanics*, 1987, pp. 125-156.
5. J.L. Helman and L. Hesselink, "Automated Analysis of Fluid Flow Topology," *3D Visualization and Display Technologies* (Proc. SPIE), Vol. 1083, SPIE, Bellingham, Wash., Jan. 1989, pp. 825-835.
6. J.L. Helman and L. Hesselink, "Representation and Display of Vector Field Topology in Fluid Flow Data Sets," *Computer*, Vol. 22, No. 8, Aug. 1989, pp. 27-36.

7. S. Rogers and D. Kwak, "An Upwind Differencing Scheme for Time Accurate Incompressible Navier-Stokes Equations," *Proc. AIAA 6th Applied Aerodynamics Conference*, American Institute of Aeronautics and Astronautics, New York, June 1988, pp. 492-502.
8. S.X. Ying, L.B. Schiff, and J.L. Steger, "A Numerical Study of 3D Separated Flow Past a Hemisphere Cylinder," *Proc. AIAA 19th Fluid Dynamics, Plasma Dynamics and Lasers Conf.*, American Institute of Aeronautics and Astronautics, June 1987, paper 87-1207.
9. H. Fuchs, Z.M. Kedem, and S.P. Uselton, "Optimal Surface Reconstructions from Planar Contours," *Comm. ACM*, Vol. 20, No. 10, Oct. 1977, pp. 693-702.
10. P. Maillot, "Three Dimensional Homogeneous Clipping of Triangular Strips," Tech. Report, Sun Microsystems, Mountain View, Calif., Oct. 1990.
11. A. Blaquiere, *Nonlinear System Analysis*, Academic, New York, 1966.
12. M.S. Chong, A.E. Perry, and B.J. Cantwell, "A General Classification of 3D Flow Fields," *Physics of Fluids Ann.*, Vol. 2, No. 5, May 1990, pp. 765-777.

**James L. Helman** is a research assistant in the Fourier Optics and Optical Diagnostics Laboratory at Stanford, where he is completing work on a PhD in applied physics. His dissertation is on the interpretation and visualization of fluid flow topology, and his other research interests include visual programming, virtual environments, and 3D input devices.

Helman received a BA in physics and math from Washington University (St. Louis) and an MS in applied physics from Stanford in 1984. He is a member of the American Physical Society, ACM, and IEEE.

**Lambertus Hesselink** holds a joint appointment as professor in the electrical engineering and aeronautics and astronautics departments at Stanford University. He has served on several scientific advisory committees for industry and the U.S. Government, most recently on the advisory committee for the Hubble Space Telescope. His research interests include nonlinear optics, optical phase conjugation, optical signal processing, optical computing, optical diagnostics, and 3D digital image processing.

Hesselink received a BS in applied mechanics and a separate BS in applied physics from the Twente Institute of Technology in the Netherlands. He received his MS and PhD from the California Institute of Technology. He is a member of the American Institute of Aeronautics and Astronautics, American Physical Society, and the Society of Photo-optical Instrumentation Engineers.

Helman and Hesselink can be reached at the Department of Applied Physics, Durand 358, Stanford University, Stanford, CA, 94305-4035.