LAWRENCE
LIVERMORE
NATIONAL
LABORATORY

UCRL-JRNL-208330

Progress in Scientific Visualization

N. Max

December 2, 2004

The Visual Computer

Progress in Scientific Visualization

NELSON MAX

Lawrence Livermore National Laboratory

Post Office Box 808, Mail Stop L-560

Livermore, CA 94550, USA

e-mail: max2@llnl.gov
phone: +1 925 422 4074
fax: +1 925 422 6287

abstract: This paper surveys recent work in the visualization of molecules, scalar fields, and vector fields.

Keywords: molecular visualization, volume visualization, flow visualization

Introduction

Visualization of observed data or simulation output is important to science and engineering. I have been particularly interested in visualizing 3-D structures, and report here my personal impressions on progress in the last 20 years in visualizing molecules, scalar fields, and vector fields and their associated flows. I have tried to keep the survey and list of references manageable, so apologize to those authors whose techniques I have not mentioned, or have described without a reference citation.

Molecular visualization.

In 1983 I wrote a survey article on molecular graphics [1], and the basic visualization paradigms discussed there are still in use now. These include stick representations of the covalent bonds (figure 1), small atom spheres connected by such sticks, "space filling" models with larger overlapping atom spheres and no sticks (figure 2), and smooth molecular surfaces. In the space filling models, the spheres have the atom's Van der

Waals radius, so that the contact forces from these "hard" spheres approximates the contact forces between the electron clouds around two non-bonded atoms.

Even in very early implementations [2] the models were interactively moveable, at least by rigid body translations and rotations, which are useful in motion parallax for 3-D understanding, and to move or "dock" one molecule relative to another. Molecules also have some internal flexibility. Usually the distance between two covalently bonded atoms cannot change by much, nor can the angle between two covalent bonds from the same atom. However, in situations where a covalent bond is not part of a cycle, so that removing this bond would break the molecule into left hand and right hand pieces, the right hand piece can often rotate with respect to the left hand piece, about an axis through the bond. Interactive molecular modeling packages also allow such bond rotations.

The smooth "solvent accessible" surface represents the inner surface of the volume filled by possible positions of a solid sphere approximating the shape of a solvent molecule like water, as it moves without penetrating the space filling model of the solute. This volume is bounded by convex pieces of the space filling surface where the solvent sphere touches one solute atom sphere, convex pieces of tori formed when the solvent sphere rolls touching two solute atom spheres, and convex pieces of the solute sphere, where it is in a fixed position touching three solute atom spheres. (See [3].)

Another smooth surface is formed by approximating the electron density by a sum of gaussian densities centered at the atomic nuclei, and taking a contour surface. An even smoother spherical harmonic surface represents the overall molecular shape without showing the individual atoms, as in figure 3. Such surfaces are parameterized by a sum of

orthogonal spherical harmonic functions on the unit sphere, or equivalently, by limited total degree polynomials of the x, y, and z coordinates of the point on the unit sphere.

Recently, Jane Richardson has suggested a ribbon representation for protein structures, showing a strip representing the "backbone" of the protein chain, spiraling around a cylinder for alpha helixes, or with a directional arrow for strands in beta sheets, and showing a narrower tube in other regions (figure 4). This representation shows the basic connectivity and secondary structure of the protein backbone, without obscuring it with the clutter of atoms in the side chains.

A recent development in computer modeling is rapid prototyping systems, which can construct a physical model from a computer one, a thin layer at a time. Arthur Olson and his colleagues at Scripps Research Institute have made colored physical molecular models in this way (figure 3), including flexible ones that can be rotated about covalent bonds. They have also instrumented the models for computer vision by adding readily recognizable high contrast patterns, so that the manipulations can be automatically recorded in electronic form, and computer data can be superimposed on the video image (see [4]). Thus, they have replaced the artificial reality interfaces for molecular modeling with true physical reality.

Volume Rendering

Rendering of a semi-transparent colored volume density representing a 3-D scalar field was pioneered in a series of publications in 1988 [5, 6, 7, 8]. See figure 5. This visualization method simulates the emission, absorption, and scattering of light from small dispersed particles in the volume, as described in [9]. So-called "transfer functions"

specify how the densities and colors of the particles depend on the scalar field. In order to emphasize the boundaries between different materials, the color and opacity can be modified to emphasize particular contour values or high gradients, and emulate shading and highlights on semitransparent or opaque surfaces.

Volume rendering was initially done in software, but it was soon implemented in hardware, either as a 3-D extension to 2-D texture mapping on high-end general purpose graphics machines, or on special purpose hardware. Now 3-D texturing is a standard feature of the relatively inexpensive graphics cards used in computer games. The 3-D texture method slices a number of closely spaced planes through the volume, parallel to the image plane. At each pixel in the projection of the polygon where the slice intersects the data volume, the 3-D scalar data is interpolated from the eight surrounding texture samples, and the RGBA values are determined from the transfer functions. Hardware back-to-front compositing of these textured polygons then approximates the integral for the light reaching the eye through each pixel.

This hardware 3-D texture method only works if the scalar field is sampled on a regular cubic grid (or an affine or projective transformation of such a grid). In finite element simulation for solids, or Lagrangian simulations for fluids, the data samples are irregularly spaced. They can be resampled onto a regular grid, but this can create too many samples, introduce blurring, or miss small details, so it is sometimes better to directly render the irregular grid. Irregular tetrahedral grids were first ray traced in software by Garrity [10], and in hardware by Weiler *et al*. [11]. Alternatively, general polyhedral cells can be projected as a whole in back to front order. Currently, the polyhedron projection can be done in hardware, but the necessary back to front visibility

sort is usually done in software [12]. As graphics processing units (GPUs) become more powerful and flexible, more and more graphics and visualization algorithms will migrate from the CPU to the GPU, where they can take advantage of local data, parallel computation streams, and proximity to the frame buffer.

The visualization of continuous 3-D data on a 2-D image remains problematical, even in the presence of binocular stereo and/or motion parallax, since a 2-D image cannot possibly convey all the data in a dense volume distribution. So, as the size and complexity of volume data grows in the future, automatic methods of recognizing and highlighting the salient regions will become more important. Highlighting appropriate contour surfaces with enhanced shading is a first step in this direction.

Contour surfaces

Instead of enhancing contours with volume rendered shading, the contour surface can be triangulated, and rendered with traditional polygon-based graphics hardware. The scalar field is sampled at the vertices of a regular cubical grid. The topology of the surface on a grid square or cube depends in part on which of the vertices have field value greater than the contour value. Straight contour lines are drawn on each cube face, joining the points on the edges where the linearly interpolated scalar takes on the contour value. One must then choose a polygonal surface inside each cube whose boundary is these contour lines.

Wyvill *et al*. [13] considered each of the 16 possible cases for each cube face. One face case, where the scalar field is greater than the contour value at only two diagonally opposite vertices, is ambiguous because the four edge points that result can be

joined by a pairs of lines in two different ways. Wyvill *et al*. [13] chose one. Lorenson and Cline, in a later but better known paper [14], considered the 256 possible cases for a cube, and then reduced them by rotation and reflection symmetry to only 14 essentially different ones, for which they proposed triangulations. Unfortunately, ambiguous faces were not always treated the same on adjacent cubes, creating cracks in the surface. Hamann and Nielson [15] showed how to resolve this ambiguity by choosing the contour lines on a face so that their topology is consistent with that of the field bilinearly interpolated from the four vertices, and Nielson [16] showed how to triangulate the interior of a cube so that the topology is consistent with that of the trilinearly interpolated field.

These contour surfaces often have too many polygons to be viewed interactively, and are prime candidates for hierarchical surface simplification, by eliminating vertices, edges, or faces in areas of less detail, and retriangulating. An alternative is a hierarchical representation by wavelets defined on a simple base mesh, as in [17].

Flow Visualization

The velocity of a flow is a vector field, and any vector field can be integrated to produce a flow, so flows and vector fields are essentially equivalent for the purposes of visualization. The simple method of placing a vector arrow with its tail at each point in a regular grid sometimes works well in 2-D, but usually produces an unintelligible clutter in 3-D. More successful methods emulate the visualization techniques used in physical experiments: smearing colored oil on surfaces, releasing smoke into a gas or dye into a liquid, or releasing discrete particles whose motion can be tracked.

For an unsteady flow, with time-varying velocity, the track of dye or smoke continuously released from a point is called a streak line, and can move in time, while the track left by a single released particle in a time exposure is called a path line. For a steady flow, these two tracks are the same, and are called streamlines. Since the visualization techniques for these three types of lines are similar, I will refer only to streamlines below.

Streamlines are easy to draw in 3-D, but if they originate at each point in a regular grid, they can become cluttered, so care must be taken in placing them. Turk and Banks [18] used image processing and optimization to place streamlines evenly in 2-D, as in figure 6. In 3-D, streamlines are often placed interactively by the user, at specified points, or spaced evenly along specified line segments.

If streamlines are drawn through every point on a line segment, they sweep out a stream ribbon, which reveals rotation or shear in the flow. A stream tube, formed from all stream lines passing through a closed curve like a circle shows expansion and contraction perpendicular to the flow direction. Finally, the union of the stream lines passing through the interior of a polygon fills up a "flow volume", which can be rendered by polyhedron projection, as in figure 7. In all these techniques, care must be taken to add new elements to the surface or volume as the flow diverges, or break it when the flow passes around an obstacle.

As in volume rendering, it is difficult to perceive the 3-D structure in a collection of streamlines that overlap in a 2-D image, so shading and occlusion cues can help. One can simply render a stream line twice; once as a wide dark line, and again as a narrower brighter line in front of it. When used with depth buffer visibility testing, this produces an outlined or haloed line effect. One can also render the lines as fully modeled cylindrical

tubes, although this takes more time, or simulate the cylindrical shading with a texture map, as in figure 8. Finally, one can shade the stream line by the average color integrated across the width of the projected cylinder, which can be computed with an analytic formula from the viewing and lighting directions, and/or stored in a table or texture. All these techniques are discussed by Schussman[19].

Another way to avoid clutter is to automatically place only a few stream lines near significant features like object surfaces, regions of high velocity, or vortices. When studying drag or turbulence, the vortices may be the main objects of interest, and can be indicated by vortex tubes, as in Banks and Singer [20].

In aerodynamic simulations, the flow near the aircraft surface is important, and can be visualized in a wind tunnel by putting colored oil on the model surface, and watching how it is smeared by the wind. An equivalent effect can be achieved by the line integral convolution (LIC) technique of Cabral and Leedom[21] by integrating a noise filter along the streamline through every pixel, weighted by a filter to limit the integration to a small region (see figure 9). If the filter changes in time, an animation cycle can be created. This was originally done in software for 2D vector fields on a plane, but has now been extended to curved surfaces, and implemented on GPUs. Directly advecting a texture by the flow is a related technique which can also now be done on the GPU.

A final way to analyze a vector field is by its topology, which can be studied by locating the critical points where the velocity is zero, and analyzing the derivatives of the velocity there. A general vector field can have critical points at sources and sinks near which the velocity is incoming or outgoing , respectively. This is impossible for an incompressible flow, but all flows can have saddle critical points where the flow is

incoming along some directions, and outgoing along others. These critical points can be classified by looking at the eigenvalues and eigenvectors of the matrix of partial derivatives of the velocity vector, as described in Hellman and Hesselink [22].

Acknowledgements

This work was performed under the auspices of the U.S. Department of Energy by University of California Lawrence Livermore National Laboratory under contract No. W-7405-Eng-48. Figures 1, 2, and 4 were produced by the ProteinShop system described in Kreylos et al. [23].

1. Max, N (1983) Computer representation of molecular surfaces. IEEE Computer Graphics and applications 3(5) 21-29.
2. Levinthal, C (1966) Molecular model building by computer. Scientific American 214(6) 42.
3. Connolly, M (1983) Analytical molecular surface calculation. J. Appl. Crystallogr. 16 538-558.
4. Gillet, M, Sanner, M, Stoffler D, Goodsell, D and Olson, A (2004) Augmented reality with tangible auto-fabricated models for molecular biology applications. IEEE Visualization 2004 proceedings, pp. 235 – 241.
5. Levoy, M (1988) Display of surfaces from volume data. IEEE Computer Graphics and Applications 8(3) 29-37.
6. Sabella P (1988) A rendering algorithm for visualizing scalar fields. Computer Graphics 22(4) 51-58.
7. Upson C and Keeler M (1988) V-BUFFER: visible volume rendering. Computer Graphics 22(4) 59-64.
8. Dreben R, Carpenter L and Hanrahan P (1988) Volume rendering. Computer Graphics 22(4) 65-74.
9. Max N (1995) Optical models for direct volume rendering. IEEE Transactions on Visualization and Computer Graphics 1(2) 99-108.
10. Garrity M (1990) Ray tracing irregular volume data. Computer Graphics 24(5) 35-40.
11. Weiler M, Kraus M, Merz M, and Ertl T (2003) Hardware-based ray casting for tetrahedral meshes. IEEE Visualization 2003 proceedings, pp. 333-340.
12. Max N, Williams P, Silva C, and Cook, R (2003) Volume rendering for curvilinear and unstructured grids. Computer Graphics International proceedings, IEEE Computer Society, pp. 210-215..
13. Wyvill, G, McPheeters, C, and Wyvill B (1986) Data structures for soft objects. The Visual Computer 2(4): 227-234.
14. Lorensen W and Cline H (1987) Marching cubes: a high resolution surface construction algorithm. Computer Graphics 21(4): 163-169.
15. Nielson G and Hamann B (1991) The asymptotic decider: resolving the ambiguity in marching cubes. IEEE Visualization 1991 proceedings, pp. 83-91.

16. Nielson G (2003) On Marching Cubes. IEEE Transactions on Visualization and Computer Graphics 9(3) 283-297.
17. Bertram M, Laney D, Duchaineau M, Hansen C and Hamann B (2001) Wavelet representation of contour sets. IEEE Visualization 2001 proceedings, pp. 303-310.
18. Turk G and Banks D (1996) Image-gided streamline placement. ACM Siggraph 96 Conference proceedings, pp. 453-460.
19. Schussman G (2003) Interactive and perceptually enhanced visualization of large, complex line-based data sets, PhD thesis, Computer Science, University of California, Davis
20. Banks D and Singer B (1995) A predictor-corrector technique for visualizing unsteady flow. Transactions on Visualization and Computer Graphics 1(2): 151-163.
21. Cabral B and Leedom L (1993) Imaging vector fields using line integral convolution. Siggraph 93 Conference proceedings, pp. 263-270.
22. Helman J and Hesselink L (1991) Visualizing vector field topology in fluid flows. IEEE Computer Graphics and Applications 11(3) 36-46.
23. Kreylos, O, Max, N, Hamann, B, Crivelli, S, and Bethel E W (2003) Interactive protein manipulation. IEEE Visualization 2003 proceedings, pp. 581 – 588.

Figures

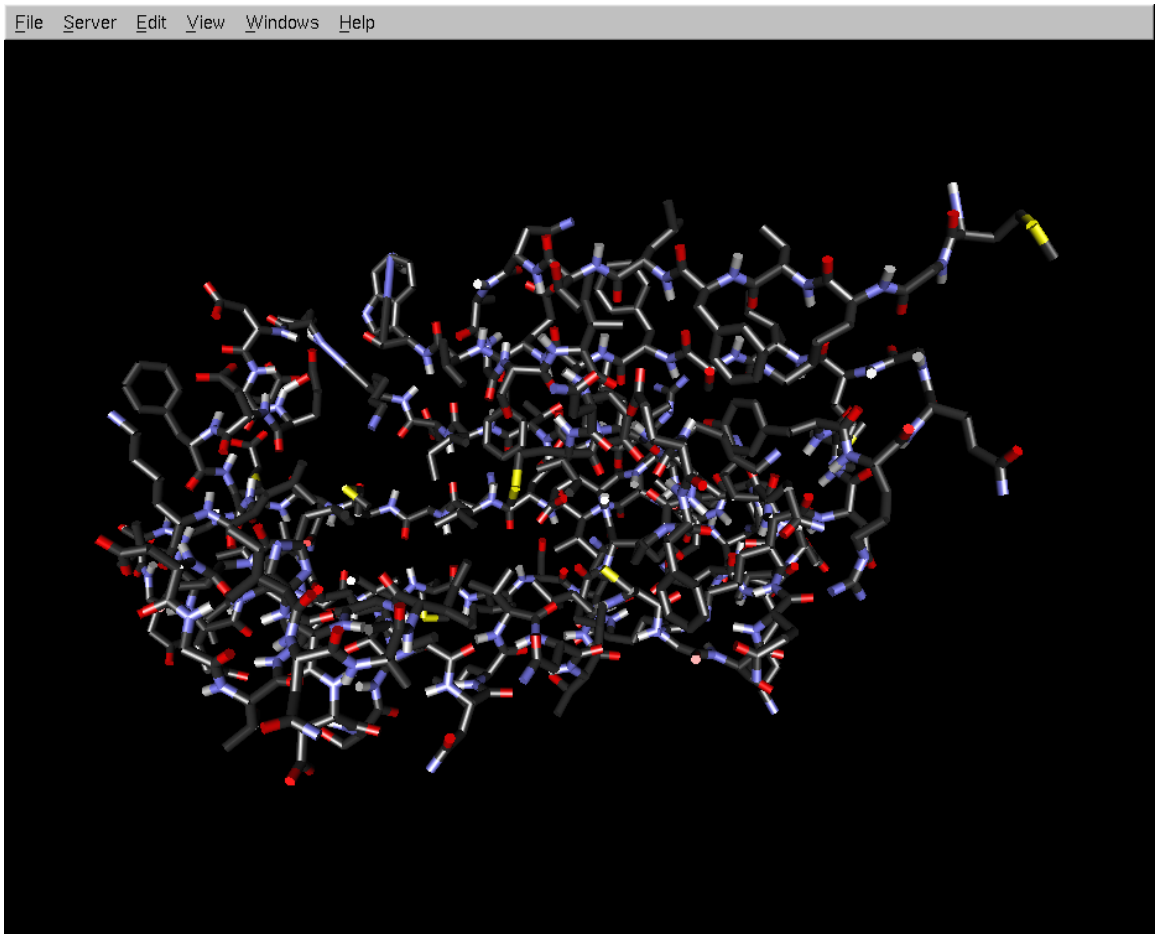

Figure 1. Ball and stick model of the immunoglobin-binding domain of domain of streptococcal protein G (1pgx from the Protein Data Bank www.rcsb.org/pdb/).

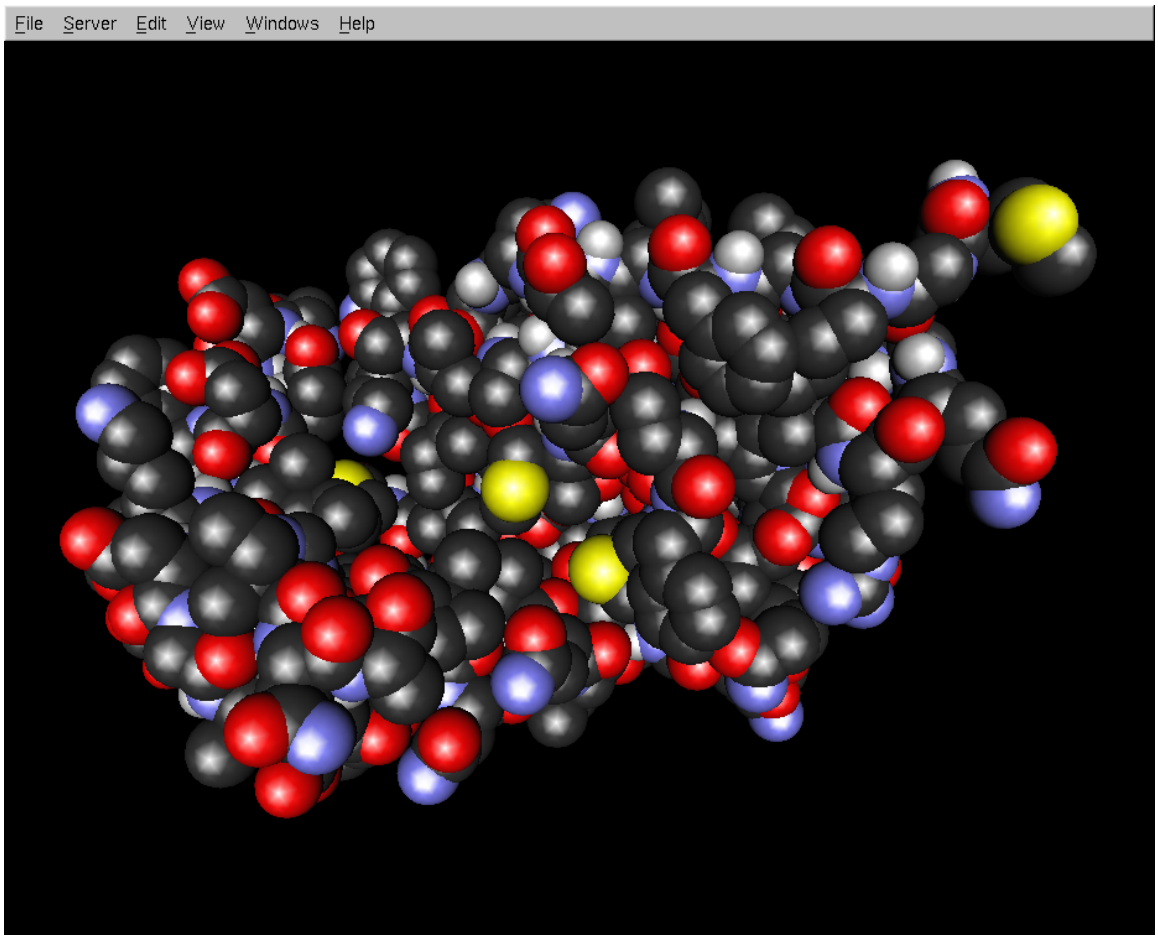

Figure 2. Space filling sphere model of 1pgx.

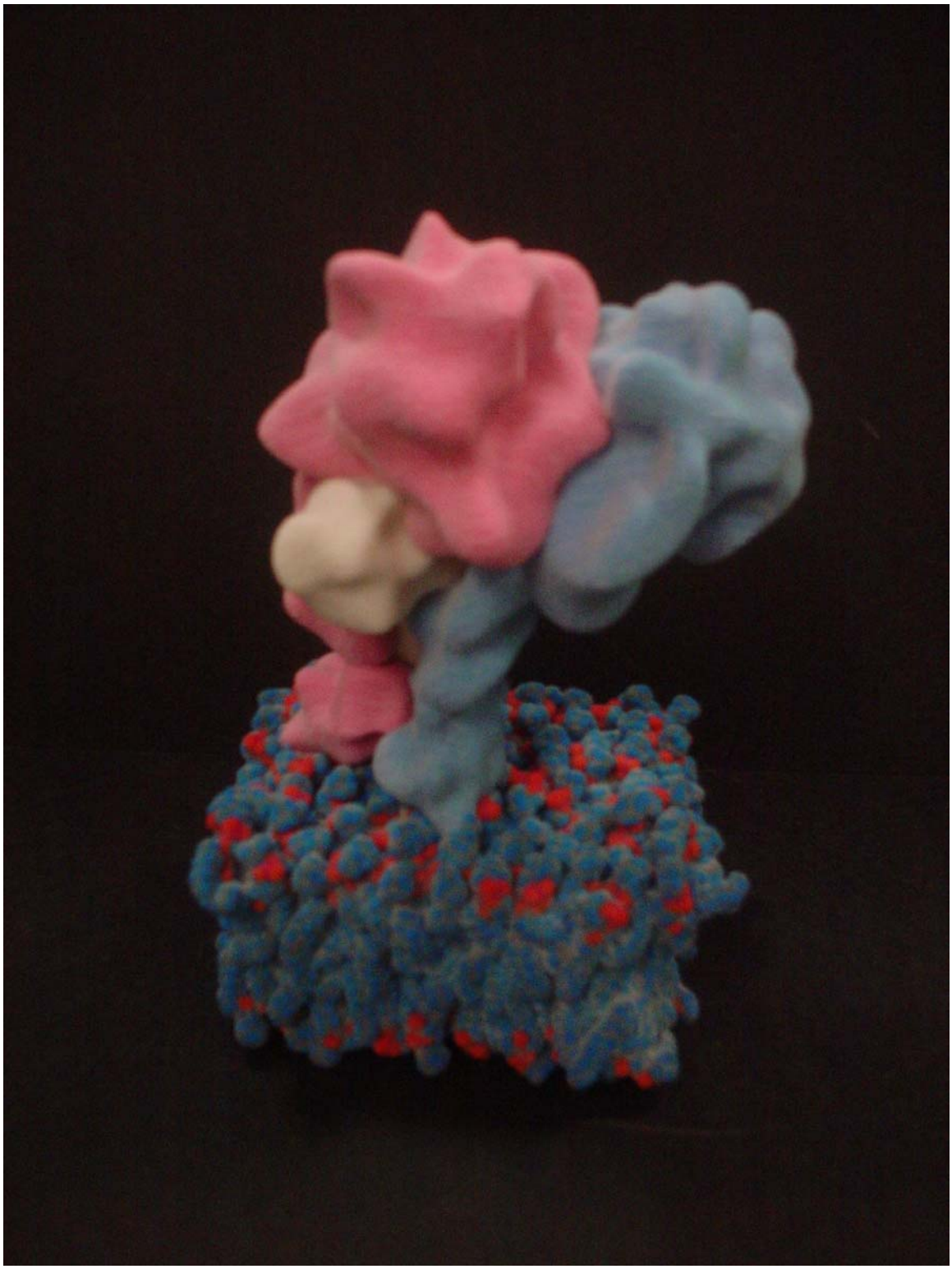

Figure 3. Photograph of a physical rapid prototyping spherical model of the complex between Tissue Factor (white), Factor VIIa (pink) and Factor X (blue) inserted into a lipid membrane. This is the initiation complex for blood coagulation.

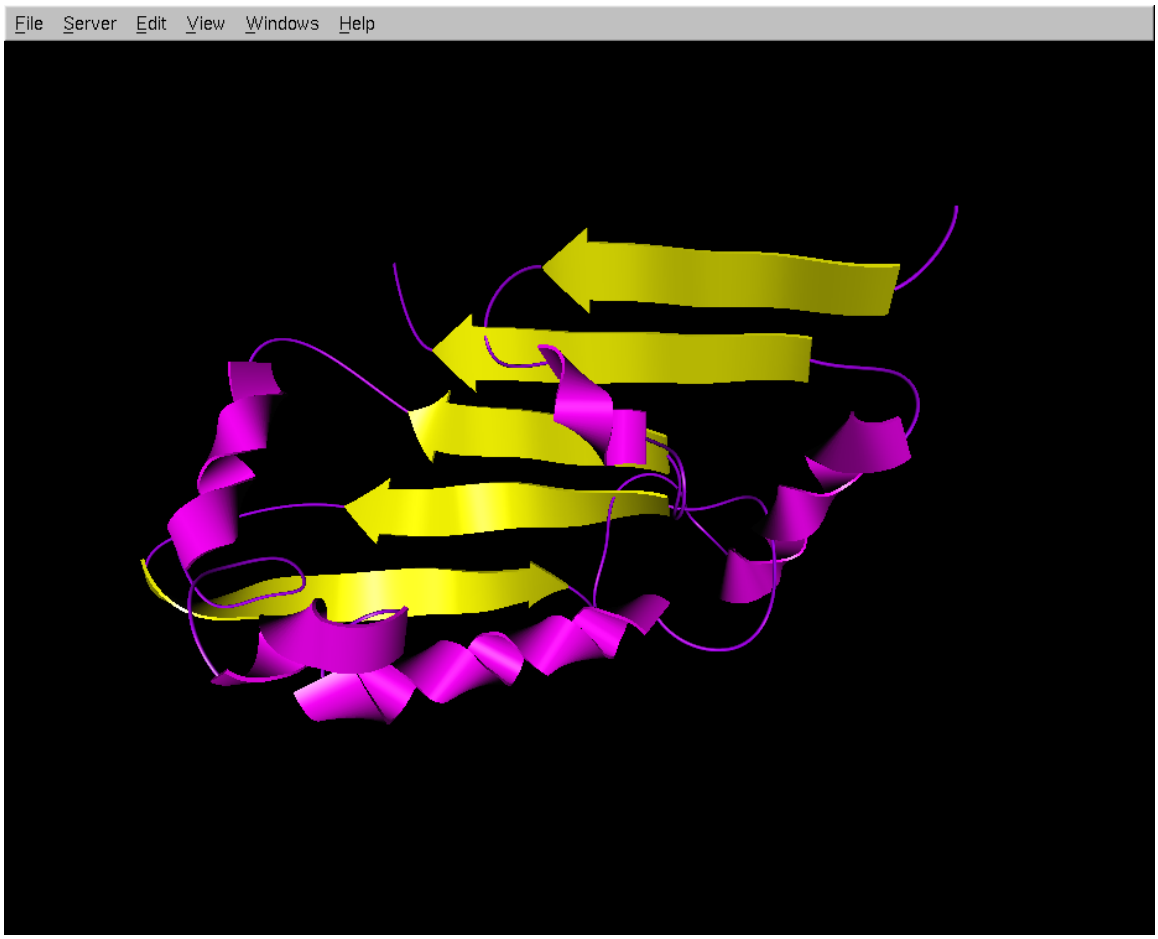

Figure 4. A ribbon representation of 1pgx.

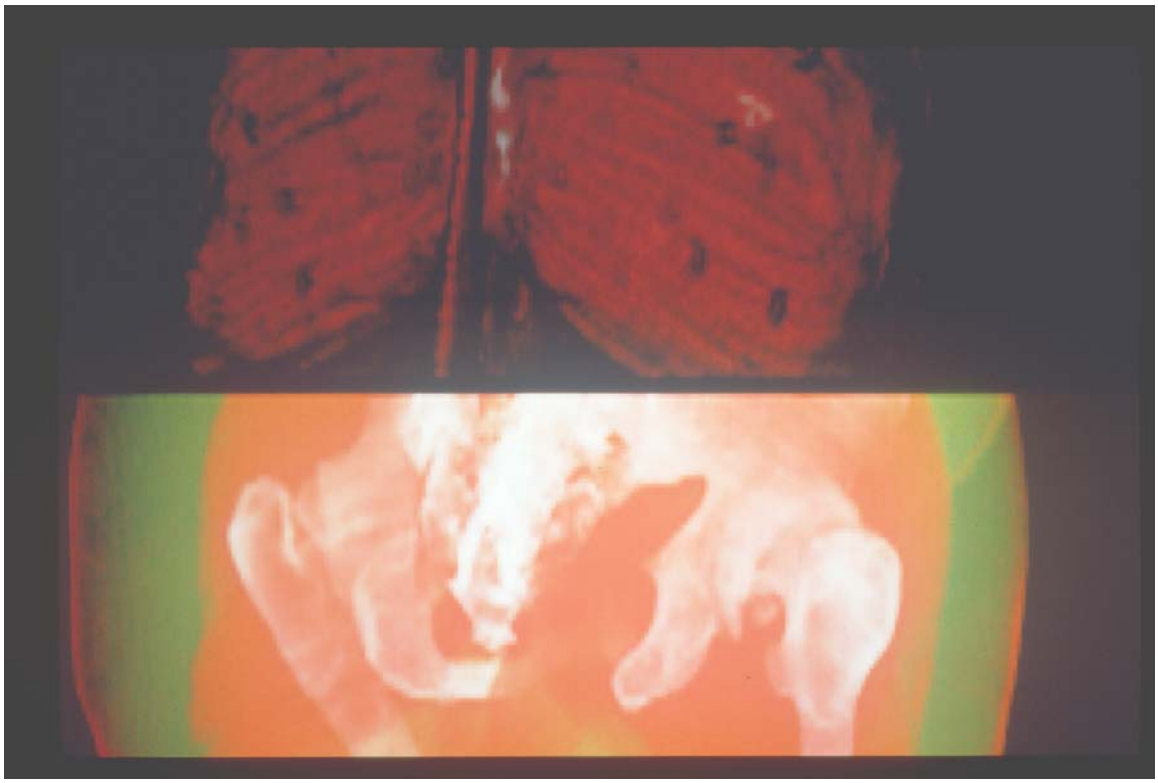

Figure 5. An early volume rendering from Pixar, showing fat in green, muscle in red, and bone in white, using the methods of [8].

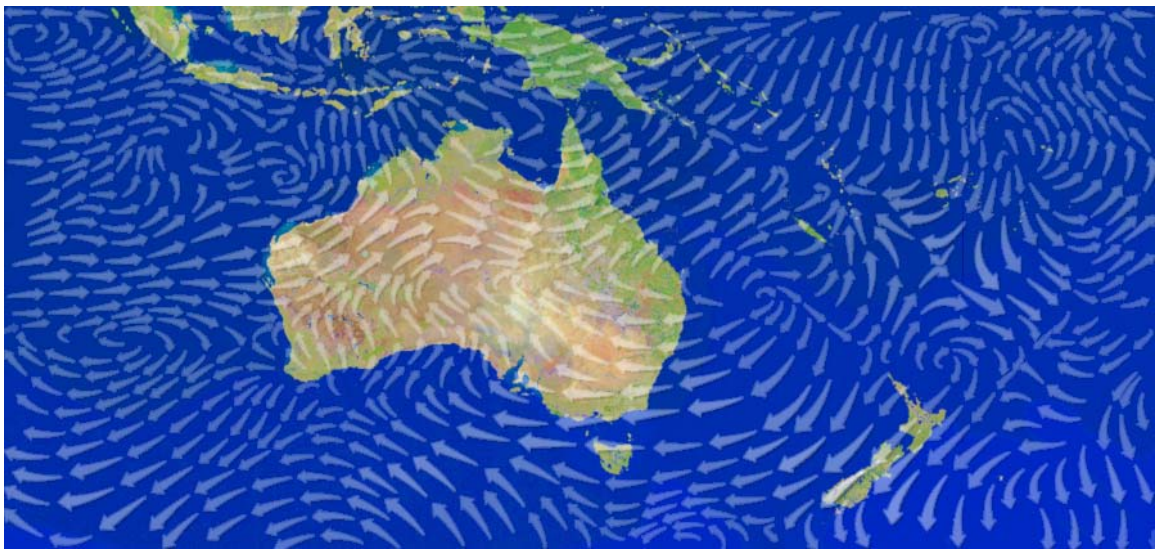

Figure 6. Optimized placement of stream line arrows, from Turk and Banks [18]. Reprint permission applied for from IEEE.

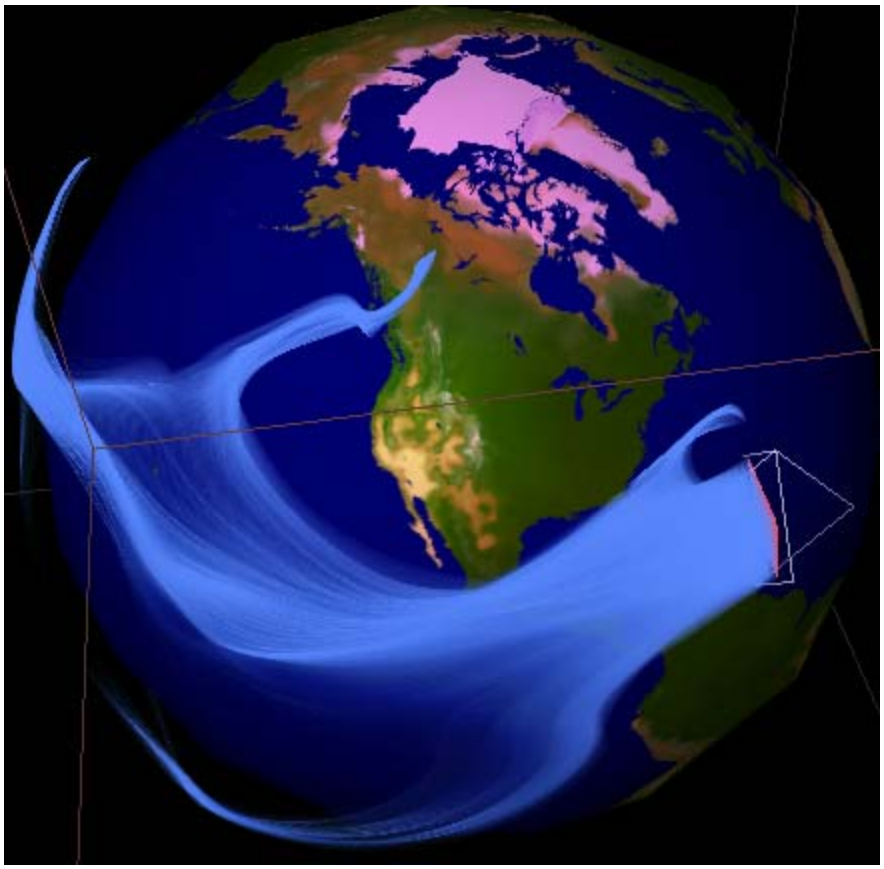

Figure 7. A flow volume for global winds.

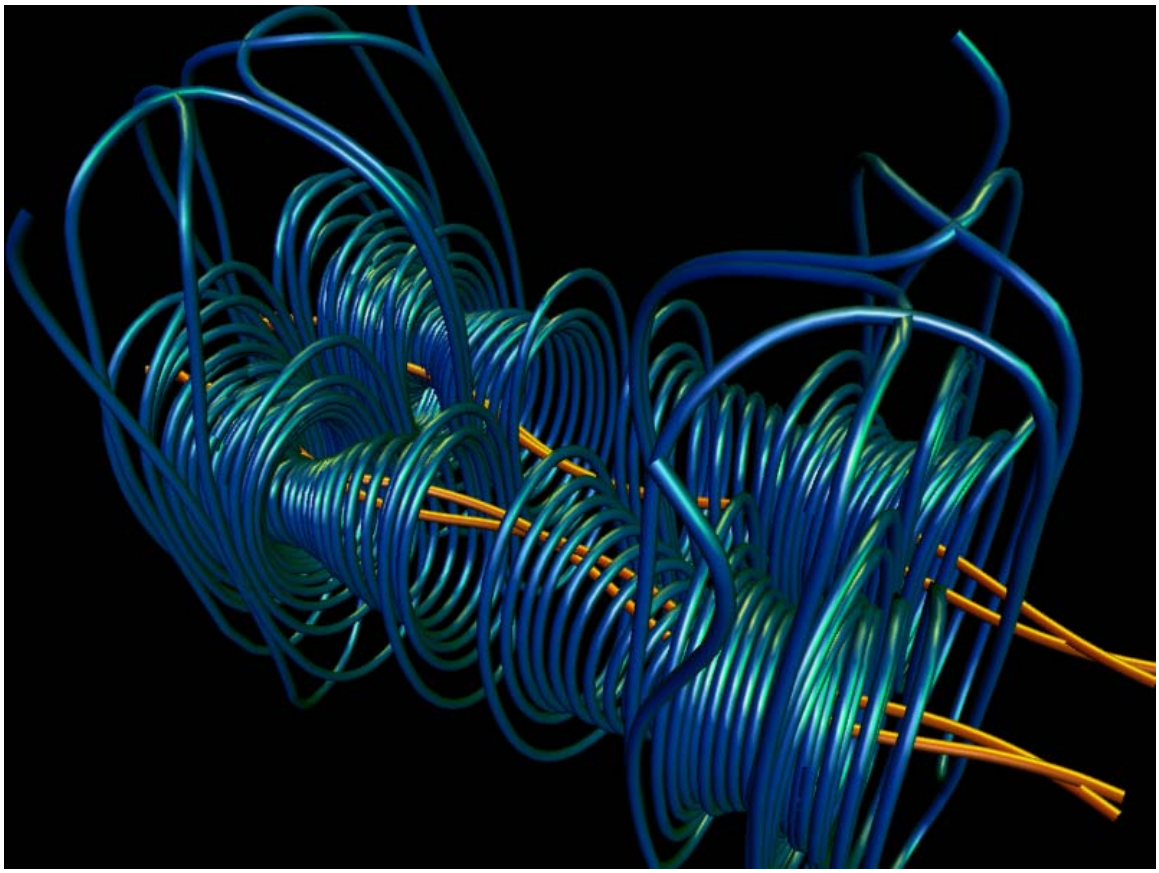

Figure 8. Magnetic field lines with cylindrical shading from a texture map, rendered by the techniques of Schussman [19].

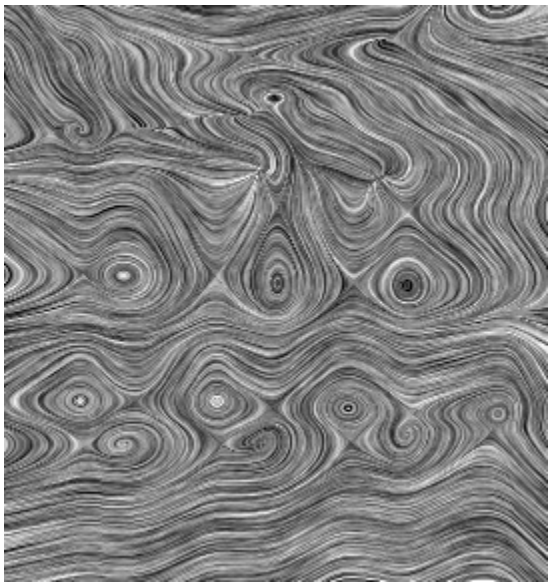

Figure 9. A line integral convolution image for a velocity field from computational fluid dynamics, from Cabral and Leedom [21]. Reprint permission requested from the ACM.