Björn Engquist
Encyclopedia of Applied and Computational Mathematics
10.1007/978-3-540-70529-1_368

Visualization

Christopher R. Johnson [1]

Scientific Computing and Imaging Institute, University of Utah, Warnock Engineering Building, Salt Lake City, UT, USA

Christopher R. Johnson
Email: crj@sci.utah.edu
URL: http://www.sci.utah.edu

Without Abstract

Synonyms

Scalar field visualization; Vector field visualization; Visualization software

---

Introduction

Computers are now extensively used throughout science, engineering, and medicine. Advances in computational geometric modeling, imaging, and simulation allow researchers to build and test models of increasing complexity and thus to generate unprecedented amounts of data. As noted in the NIH-NSF Visualization Research Challenges report, to effectively understand and make use of the vast amounts of information being produced is one of the greatest scientific challenges of the twenty-first century [1]. Visualization, namely, helping researchers explore measured or simulated data to gain insight into structures and relationships within the data, will be critical in achieving this goal and is fundamental to understanding models of complex phenomena. In this brief chapter, I will highlight visualization techniques for two common scientific data types, scalar fields, and vector fields with pointers to readily available visualization software.

Schroeder, Martin, and Lorensen have offered the following useful definition of visualization [2]:

> Scientific visualization is the formal name given to the field in computer science that encompasses user interface, data representation and processing algorithms, visual representations, and other sensory presentation such as sound or touch. The term data visualization is another phrase to describe visualization. Data visualization is generally interpreted to be more general than scientific visualization, since it implies treatment of data sources beyond the sciences and engineering. … Another recently emerging term is information visualization. This field endeavors to visualize abstract information such as hyper-text documents on the World Wide Web,

directory/file structures on a computer, or abstract data structures.

The field of visualization is focused on creating images that convey salient information about underlying data and processes. In the past three decades, there has been unprecedented growth in computational and acquisition technologies, a growth that has resulted in an increased ability both to sense the physical world in precise detail and to model and simulate complex physical phenomena. As such, visualization plays a crucial role in our ability to comprehend such large and complex data – data which, in two, three, or more dimensions, convey insight into such diverse applications as understanding the bioelectric currents within the heart, characterizing white matter tracts by diffusion tensor imaging, and understanding flow features within a fluid dynamic simulation, among many others.

Shown in Fig. 1, the “visualization pipeline” is one method of describing the process of visualization. The *filtering* step in the pipeline involves processing raw data and includes operations such as resampling, compression, and other image processing algorithms such as feature-preserving noise suppression. In what can be considered the core of the visualization process, the *mapping* stage transforms the preprocessed filtered data into geometric primitives along with additional visual attributes, such as color or opacity, determining the visual representation of the data. *Rendering* utilizes computer graphics techniques to generate the final image using the geometric primitives from the mapping process.

Visualization, Fig. 1

The visualization pipeline

While the range of different visualization applications is vast, the scientific visualization research community has found it useful to characterize scientific visualization techniques using a taxonomy associated with the dimensionality of the physical field to visualize:

- Scalar fields (temperature, voltage, density, magnitudes of vector fields, most image data)
- Vector fields (pressure, velocity, electric field, magnetic field)
- Tensor fields (diffusion, electrical and thermal conductivity, stress, strain, diffusion tensor image data)

I use this taxonomy to discuss visualization techniques in this entry.

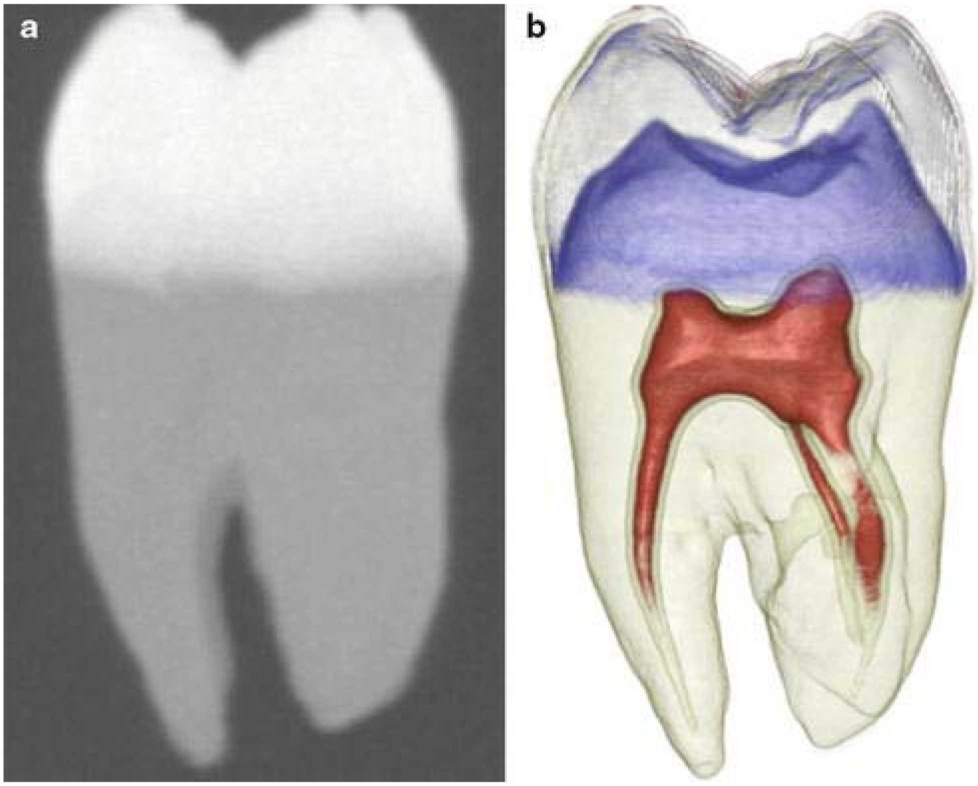
Scalar Field Visualization

Scalar data is prevalent throughout science, engineering, and medicine. In scientific computing, scalar fields represent a quantity associated with a single (scalar) number, such as voltage, temperature, and the magnitude of velocity. Scalar fields are among the most common datasets in scientific visualization, and thus they have received the most research attention (see [3] for an overview of scalar field visualization research). There are two main techniques for visualizing three-dimensional scalar data: volume rendering and isosurface extraction.

Volume Rendering

Volume rendering is a method of displaying three-dimensional volumetric scalar data as two-dimensional images and is one of the simplest ways to visualize volume data. The individual values in the dataset are made visible by the choice of a transfer function that maps the data to optical properties, like color and opacity, which are then projected and composited to form an image. As a tool for scientific visualization, the appeal of direct volume rendering is that no intermediate geometric information need be calculated, so the process maps from the dataset “directly” to an image. This is in contrast to other rendering techniques such as isosurfacing or segmentation, in which one must first extract elements from the data before rendering them. To create an effective visualization with direct volume rendering, the researcher must find the right transfer function to highlight regions and features of interest.

A common visualization goal in volume rendering is the depiction of the interface between two different materials in a volume dataset. The material surface can usually be seen with a simple transfer function that assigns opacity only to a narrow range of values between the data values associated with each of the two materials. In datasets characterized by noise or a more complicated relationship among multiple materials, statistical analysis of the dataset values can help to guide the transfer function design process. Moreover, in cases where datasets and associated volume rendering methods are more complex (such as volumetric fields of vector or tensor values), methods for guiding the user toward useful parameter settings, based on information about the goals of the visualization, become necessary to generate informative scientific visualizations. Figure 2a shows a maximum intensity projection (MIP) of a tooth from x-ray CT data. The maximum intensity projection volume rendering method is the most simple form of volume rendering.

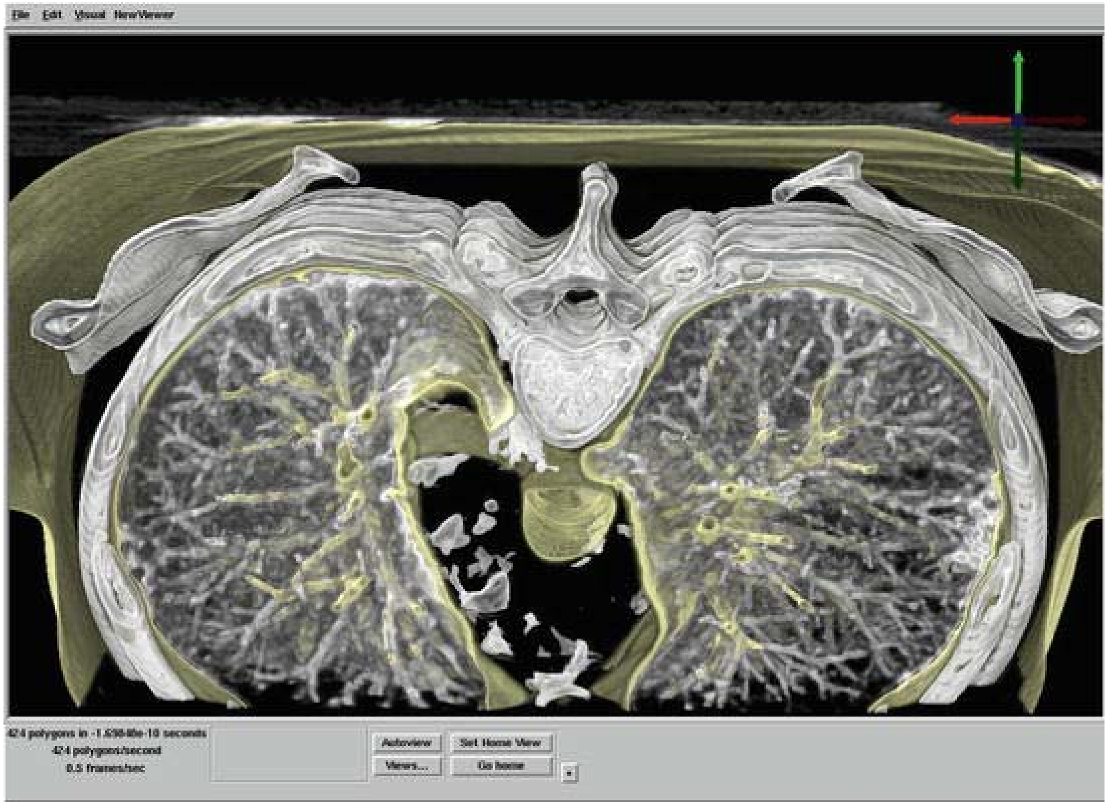


Visualization, Fig. 2

(**a**) Maximum intensity projection (MIP) volume rendering of a tooth from CT data and (**b**) a full volume rendering of the same data using multidimensional transfer functions with ImageVis3D

The MIP algorithm works by projecting parallel rays (ray casting) through the volume from the viewpoint of the user. For each ray, the algorithm selects the maximum scalar value and uses that value to determine the color of the corresponding pixel on the two-dimensional image plane. Volume rendering using MIP yields what looks like “three-dimensional x-rays” in gray scales of the scalar volume data. Full volume rendering, on the other hand, traverses the rays and accumulates (integrates) color and opacity contributions along the ray. Volume rendering using full volume rendering techniques yields an image that looks much more like what you might expect a three-dimensional volume projection to look like in color. The differences are evident as shown below in Fig. 2b.

Finding a good transfer function is critical to producing an informative rendering, but this can be a difficult task even if the only variable to set is opacity. Recently, multidimensional transfer functions were created to allow for more specificity in exploring and visualizing the data [*4*]. Multidimensional transfer functions are sensitive to more than one aspect of the volume data, for example, both the intensity and one or more spatial gradients or other derived parameters. Such transfer functions have wide applicability in volume rendering for biomedical imaging and scientific visualization of complex three-dimensional scalar fields (Fig. 3). For more on volume rendering, see [*4*– *9*].



Visualization, Fig. 3

A volume-rendered image using multidimensional transfer functions. This view highlights the detailed vasculature of the lungs (Data courtesy of George Chen, MGH)

Isosurface Extraction

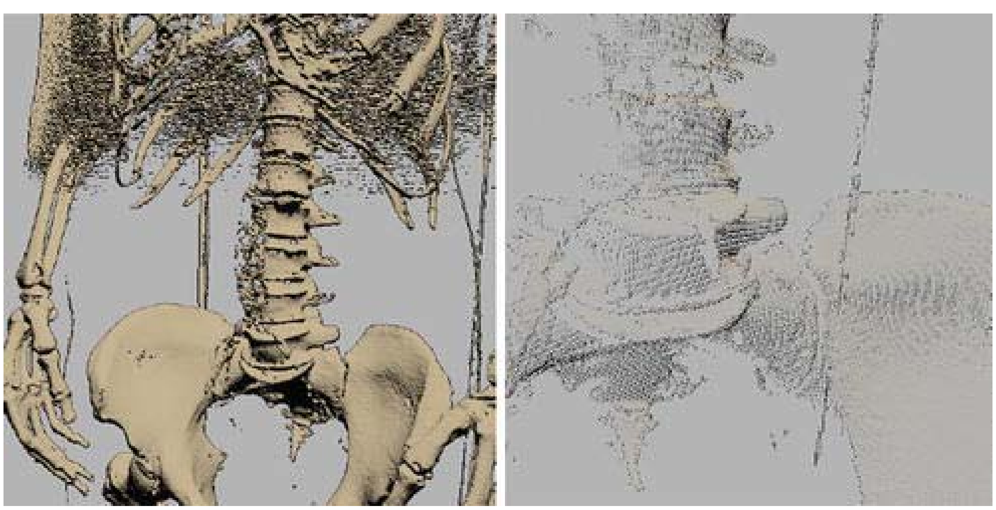
Isosurface extraction is a powerful tool for investigating volumetric scalar fields. An isosurface in a scalar volume is a surface on which the data value is constant, separating regions of higher and lower value. Given the physical or biological significance of the scalar data value, the position of an isosurface, as well as its relation to other neighboring isosurfaces, can provide clues to the underlying structure of the scalar field. In imaging applications, isosurfaces permit the extraction of particular anatomical structures and tissues; however, these isosurfaces are typically static in nature. A more dynamic use of isosurfaces can provide better visualization of complex space- or time-dependent behaviors in many scientific applications.

Within the last 15 years, isosurface extraction methods have advanced significantly from an off-line, single-surface extraction process into an interactive, exploratory visualization tool. Interactivity is especially important in exploratory visualization where the user has no a priori knowledge of any underlying structures in the data. A typical data exploration session therefore requires the researcher to make many isovalue changes in search of interesting features. In addition, it is helpful to provide global views (to place an isosurface in the context of the entire dataset) and detailed views of small sections of interest. Maintaining interactivity while meeting these visualization goals is especially challenging for large datasets and complex isosurface geometry.

The *marching cubes* [10, 11] method, introduced in 1986, was the first practical and most successful isosurface extraction algorithm. Its simplicity has made it the de facto standard extraction method even to this date. The marching cubes algorithm demonstrated that isosurface extraction can be reduced, using a divide and conquer approach, to solving a local triangulation problem. In addition, the marching cubes method proposed a simple and efficient local triangulation scheme that uses a lookup table. Subsequently, researchers created methods for accelerating the search phase for isosurface extraction [12, 13] all of which have a complexity of $O(n)$, where n is the number of voxels in the volume. We introduced the *span space* [14] as a means for mapping the search onto a two-dimensional space and then used it to create a *near optimal isosurface extraction* (NOISE) algorithm that has a time complexity of $O(\sqrt{n} + k)$, where k is the size of the isosurface. Cignoni et al. [15] employed another decomposition of the span space leading to a search method with optimal time complexity of $O(\log n + k)$, albeit with larger storage requirements. In addition, Bajaj et al. introduced the contour spectrum, which provides a fast isosurface algorithm and a user interface component that improves qualitative user interaction and provides real-time exact quantification in the visualization of isocontours [16].

We improved further on these isosurface extraction methods by using a different visibility testing approach and virtual buffer rendering to achieve a real-time, view-dependent isosurface extraction [17]. We also presented a progressive hardware-assisted isosurface extraction (PHASE) that is suitable for remote visualization, i.e., when the data and display device reside on separate computers. This approach works by reusing, when a view point is changed, the information and triangles that were extracted from the previous view point. Using this approach, we can extract only newly visible sections of the isosurface and thus improve visualization performance.

Following the same view-dependent approach, we have recently proposed a novel point-based approach to isosurface extraction [17]. The basic idea of our method is to address the challenge posed by the geometric complexity of very large isosurfaces by a point-based representation of sub-pixel triangles. Combined with a new fast visibility query and a robust normal estimation scheme, our method allows for the interactive interrogation of large datasets on a single desktop computer (Fig. 4).



Visualization, Fig. 4

Isosurface extraction of the full CT data (512 × 512 × 1, 734, 1 mm spacing) of the NIH NLM Visible Female. *Left*: a section of the skeleton extracted by the PISA algorithm [17]. *Right*: a close-up view of the extracted points. Point shading is determined by an image-based normal computation technique that ensures high-quality results

---

Vector Field Visualization

Vector fields are a fundamental quantity that describe the underlying continuous flow structures of physical processes. Examples of important vector fields include electric fields, magnetic fields, the velocities and pressures of fluids, and the forces associated with mechanics. Vector-valued quantities also appear in the form of derivatives of scalar fields.

Visualizing vector field data is challenging because no existing natural representation can visually convey large amounts of three-dimensional directional information. Visualization methods for three-dimensional vector fields must balance the conflicting goals of displaying large amounts of directional information while maintaining an informative and uncluttered display.

The methods used to visualize vector field datasets take their inspiration in real-world experiments where a wealth of physical flow visualization techniques have been designed to gain insight into complex natural flow phenomena. To this end external materials such as dye, hydrogen bubbles, or heat energy can be injected into the flow. As these external materials are carried through the flow, an observer can track them visually and thus infer the underlying flow structure.

Analogues to these experimental techniques have been adopted by scientific visualization researchers, particularly in the computational fluid dynamics (CFD) field. CFD practitioners have used numerical methods and three-dimensional computer graphics techniques to produce graphical icons such as arrows, motion particles, and other representations that highlight different aspects of the flow.

Among existing flow visualization methods, the techniques relevant to the visual analysis of vector fields can be categorized as follows:

1. The simplest techniques correspond to an intuitive, straightforward mapping of the discrete vector information to so-called glyphs. Glyphs are graphical primitives that range from mere arrows to fairly complex graphical icons that display directional information, magnitude, as well as additional derived quantities such as the curl and divergence altogether.

2. The second category corresponds to the set of techniques that are based on the integration of streamlines. Streamlines are fast to compute and offer an intuitive illustration of the local flow behavior.

3. Stream surfaces constitute a significant improvement over individual streamlines for the exploration of three-dimensional flows since they provide a better understanding of depth and spatial relationships. Conceptually they correspond to the surface spanned by an arbitrary starting curve advected along the flow.

4. Textures and other dense representations offer a complete picture of the flow, thus avoiding the shortcomings of discrete samples. Their major application is the visualization of flows defined over a plane or a curved surface.

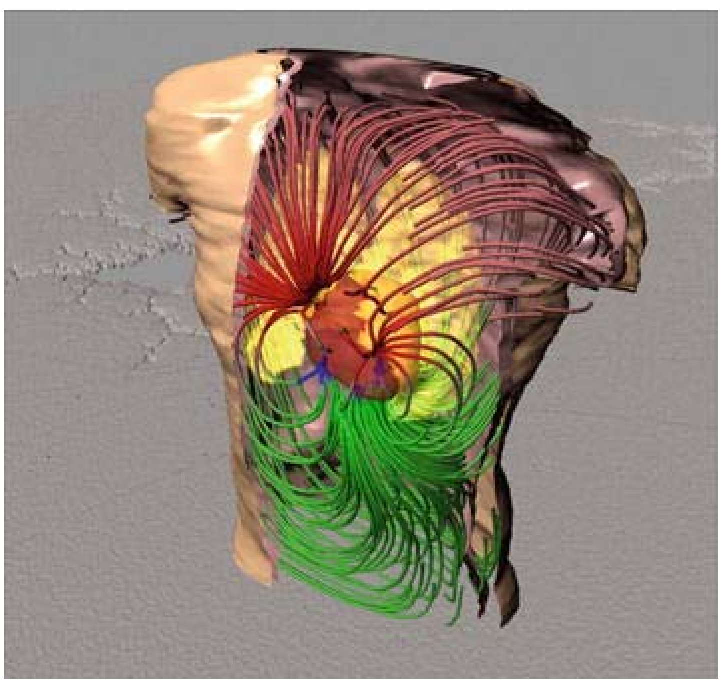
5. The last type of flow visualization techniques is based on the notion of flow topology. Topology offers an abstract representation of the flow and its global structure. Sinks and sources are the basic ingredients of a segmentation of the volume into regions connecting the same spots along the flow.

Next, we describe a few of these vector field visualization techniques.

Streamline-Based Techniques

Streamlines offer a natural way to interrogate a vector dataset. Given a starting position selected by the user, numerical integration over the continuous representation of the vector field yields a curve that can be readily visualized. The numerical schemes commonly used for the integration range from the first-order Euler scheme with fixed step size to Runge-Kutta methods with higher-order precision and adaptive step size. The choice of the appropriate method requires to take into account the complexity of the structures at play and the smoothness of the flow.

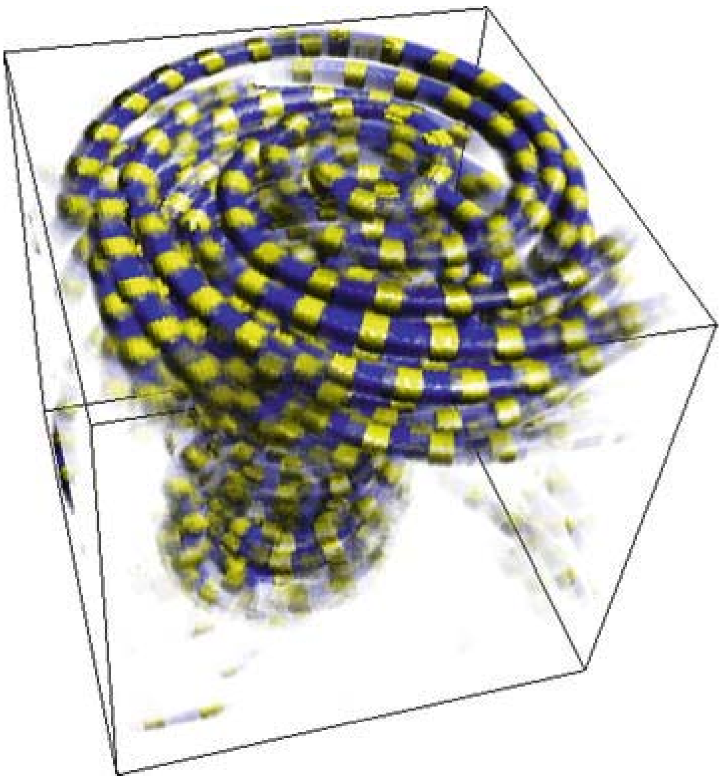
Since streamlines are unable to fill the space without visual clutter, the task of selecting an appropriate set of starting points (commonly called seed points) is critical to obtaining an effective visualization. A variety of solutions have been proposed over the years to address this problem. A simple interactive solution consists in letting the user place a probe in the data volume over which seed points are evenly distributed. The orientation and spatial extent of the rack, as well as the number of seed points, can be adjusted to allow for the selective exploration of a particular region of interest, as shown in Fig. 5.



Visualization, Fig. 5

Applications of streamlines to a finite element simulation of the bioelectric field in the torso visualized through streamlines seeded randomly around the epicardium

An additional limitation of flow visualizations based upon streamline techniques concerns the difficult interpretation of the depth and relative position of curves in a three-dimensional space. A solution consists in creating artificial lighting effects that emphasize curvature and assist the user in his/her perception of depth [18]. An alternative method that can be implemented on the graphics hardware assigns a nonzero volume to individual streamlines. These streamlines are then depicted as tubes and filled with 3D textures to create expressive images in which various visual cues are used to enhance perception [19]. Refer to Fig. 6.

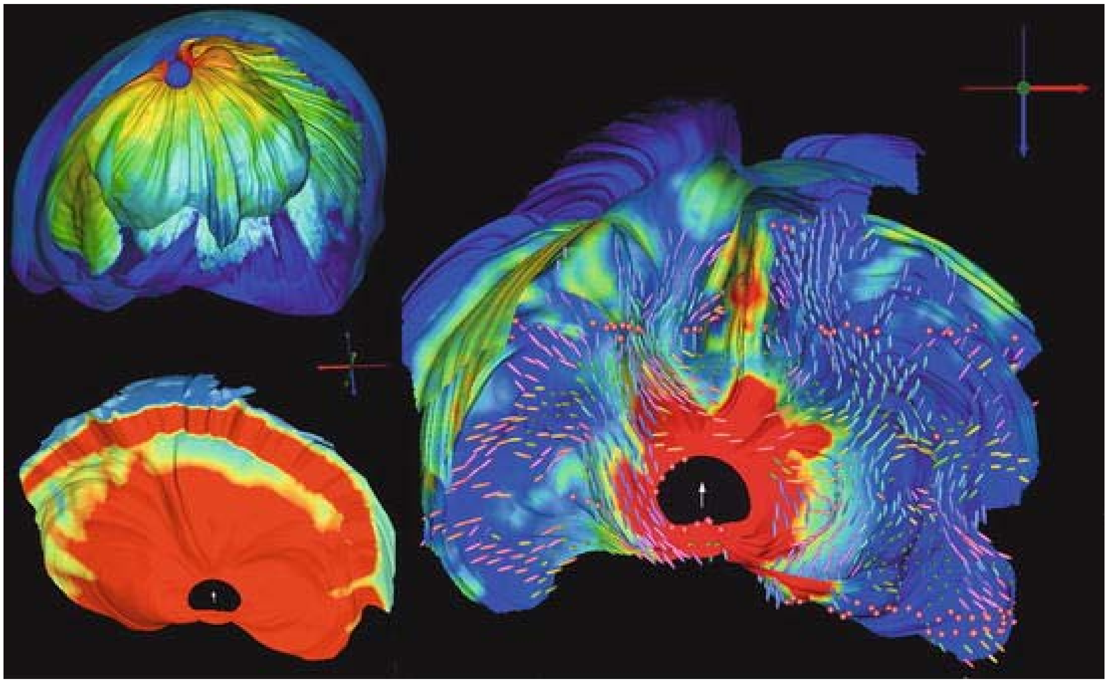


Visualization, Fig. 6

An extension of streamline-based flow visualization. The image shows a combination of streamlines and 3D textures in the visualization of a tornado dataset. Textures permit to embed additional information and ease the interpretation of the spatial context (From [19])

Stream Surfaces

The intuitive representations offered by stream surfaces make them a very valuable tool in the exploration of three-dimensional flows. The standard method for stream surface integration is Hultquist's advancing front algorithm [20]. The basic idea is to propagate a polygonal front along the flow, while accounting for possible divergence and convergence by adapting the front resolution. Yet, this method yields triangulated surfaces of poor quality when the flow exhibits complex structures. We recently proposed a modified stream surface algorithm that improves on Hultquist's original scheme by allowing for an accurate control of the front curvature [21]. This method creates smooth, high-quality surfaces, even for very intricate flow patterns. For example, as shown in Fig. 7, stream surfaces were used to visualize the electric current computed by a high-resolution finite element simulation using a realistic head model. In this case stream surfaces proved instrumental in assessing the impact of various models of the white matter anisotropy on the current pattern and its interconnection with anatomical structures.



Visualization, Fig. 7

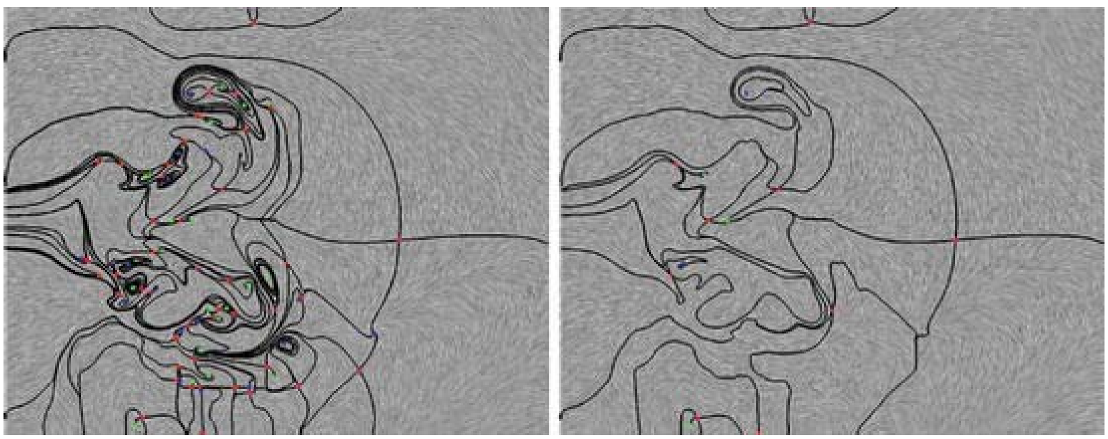
Stream surface visualization of bioelectric field induced by a dipolar source in left thalamus. *Left top*. Stream surfaces seeded along isocontour of electric flux on sphere enclosing the source. Culling is used to address occlusion. White matter has anisotropic conductivity. *Left bottom*. Stream surface started along circle contained in coronal slice and centered around source location. White matter is assumed isotropic. Color coding corresponds to magnitude of electric field. *Right*. Similar image obtained for anisotropic white matter. Glyphs visualize major eigenvector of conductivity tensor. *Color* coding shows magnitude of return current

Texture Representations

Texture-based flow visualization methods provide a unique means to address the limitations of depictions based on a limited set of streamlines. They yield an effective, dense representation which conveys essential patterns of the vector field and does not require the tedious seeding of individual streamlines to capture all the structure of interest [22]. Arguably the most prominent of those methods is Line Integral Convolution (*LIC*) proposed by Cabral and Leedom [23]. The basic idea is to apply a one-dimensional low-pass filter to a white noise texture covering the two-dimensional flow domain. The filter kernel at each pixel is aligned with streamlines of the underlying flow. Consequently the resulting image exhibits a high correlation of the color values along the flow and little or no correlation across the flow. Hence this method produces a dense set of streamline-type patterns that fill the domain and reveal all the flow structures that are large enough to be captured by the fixed resolution of the texture. This seminal work has inspired a number of other methods. In particular, improvements were proposed to permit the texture-based visualization of time-dependent flows [*24*], flows defined over arbitrary surfaces [*25*], and dye advection. Some attempts were made to extend this visual metaphor to three-dimensional flows [*26*].

Topology

The topological approach provides a powerful framework for flow visualization in a broad range of applications [*27*]. For planar vector fields, as well as vector fields defined over curved surfaces, it has established itself as a method of reference to characterize and visualize flow structures. The excessive complexity of the topology of intricate flows can be addressed by simplifying the resulting graphs while preserving essential properties in order to facilitate the analysis of large-scale flow patterns. Refer to Fig. 8.



Visualization, Fig. 8

Topology simplification. The *left image* shows the original topology obtained for a CFD simulation of a streaming jet with inflow into a steady medium. Numerous small-scale structures lead to a cluttered depiction. The *right image* shows the same dataset after topology simplification

---

Visualization Software

There are a variety of commercially available and research-based general visualization systems that may be useful for scientific visualization (see [*3*] for an overview of visualization systems). While certainly not an exhaustive list, examples of these systems are:

Amira:

Amira is a professional image segmentation, reconstruction, and 3D model generation application produced by Mercury Computer Systems GmbH (www.amiravis.com). It is used by research and development groups in chemistry, biology, medicine, material science, etc. Amira is designed to handle confocal microscopy, MRI, or CT data. It uses the Tcl language as a command interface for scripting and is built on top of the OpenGL and Open Inventor toolkits. Modules can be developed to extend the Amira system and can use parallelization techniques if the developer so desires.

ImageVis3D:

ImageVis3D (www.imagevis3D.org) is an open-source, cross-platform volume visualization program that scales to very large data on modest hardware. The main design goals of ImageVis3D are simplicity, scalability, and interactivity. Simplicity is achieved with a new user interface that gives an increased level of flexibility. Scalability and interactivity for ImageVis3D mean that the user can interactively explore very large (gigabyte and terabyte) sized datasets on either a notebook computer or a high-end graphics workstation. The open-source nature of ImageVis3D, as well as the strict component-by-component design, allows developers not only to extend ImageVis3D itself but also to reuse parts of it, such as the volume rendering core for other visualization applications.

ParaView:

ParaView (www.paraview.org) is an open-source, multi-platform data analysis and visualization application. ParaView users can quickly build visualizations to analyze their data using qualitative and quantitative techniques. The data exploration can be done interactively in 3D or programmatically using ParaView's batch processing capabilities. ParaView was developed to analyze extremely large datasets using distributed memory computing resources. It can be run on supercomputers to analyze datasets of terascale as well as on laptops for smaller data.

SCIRun:

SCIRun is an open-source scientific computing problem-solving environment created by the Scientific Computing and Imaging (SCI) Institute (www.sci.utah.edu) [*28*]. SCIRun provides software modules for scalar, vector, and some tensor field visualization. In addition, SCIRun has modules for geometric modeling and simulation.

VisIt:

VisIt (wci.llnl.gov/codes/visit) is a free interactive parallel visualization and graphical analysis tool for viewing scientific data on Unix and PC platforms. Users can quickly generate visualizations from their data, animate them through time, manipulate them, and save the resulting images for presentations. VisIt contains a rich set of visualization features so that you can view your data in a variety of ways. It can be used to visualize scalar and vector fields defined on two- and three-dimensional (2D and 3D) structured and unstructured meshes. VisIt was designed to handle very large dataset sizes in the terascale range and yet can also handle small datasets in the kilobyte range.

VTK:

VTK, the Visualization Toolkit (www.kitware.com), is an open-source visualization package that is widely used in both classroom settings and research labs. It provides general visualization capabilities for scalars, vectors, tensors, textures, and volumetric data. Written in $C++$, VTK includes Tcl, Python, and Java bindings for application development and prototyping. VTK contains some built-in parallelization pieces for both threading and MPI. Both ParaView and VisIt are built upon the VTK libraries.

Acknowledgements

The author would like to thank the many people who contributed to this article including Charles Hansen, Gordon Kindlmann, Joe Kniss, Rob MacLeod, Steve Parker, Yarden Livnat, and Xavier Tricoche. This work was funded by grants from the NSF,DOE SciDAC, and NETL, the NIH NIGMS 8 P41 GM103545-14, and the King Abdullah University for Science and Technology.

References

1. Johnson, C.R., Moorhead, R., Munzner, T., Pfister, H., Rheingans, P., Yoo, T.S.: NIH-NSF Visualization Research Challenges Report. http://tab.computer.org/vgtc/vrc/index.html (2006)

2. Schroeder, W., Martin, K., Lorensen, B.: The Visualization Toolkit: An Object Oriented Approach to Graphics, chap. 6. Kitware, Clifton Park/New York (2003)

3. Hansen, C.D., Johnson, C.R.: The Visualization Handbook. Elsevier, Amsterdam (2005)

4. Kniss, J.M., Kindlmann, G., Hansen, C.D.: Multidimensional transfer functions for interactive volume rendering. IEEE Trans. Vis. Comput. Graph. **8**(3), 270–285 (2002)
CrossRef

5. ImageVis3D: An interactive visualization software system for large-scale volume data. Scientific Computing and Imaging (SCI) Institute. Download from: http://www.imagevis3d.org (2013)

6. Bernardon, F.F., Callahan, S.P., Comba, J.L.D., Silva, C.T.: Interactive volume rendering of unstructured grids with time-varying scalar fields. In: Proceedings of the Eurographics Symposium on Parallel Graphics and Visualization, Braga, pp. 51–58 (2006)

7. Kniss, J.M., Premoze, S., Ikits, M., Lefohn, A.E., Hansen, C.D., Praun, E.: Gaussian transfer functions for multi-field volume visualization. In: Proceedings of the IEEE Visualization, Seattle, pp. 497–504 (2003)

8. Kindlmann, G.L., Weinstein, D.M., Hart, D.: Strategies for direct volume rendering of diffusion tensor fields. IEEE Trans. Vis. Comput. Graph. **6**(2), 124–138 (2000)
CrossRef

9. Kindlmann, G., Durkin, J.: Semi-automatic generation of transfer functions for direct volume rendering. In: IEEE Symposium on Volume Visualization, Durham, pp. 79–86. IEEE, Los Alamitos (1998)

10. Lorensen, W.E., Cline, H.E.: Marching cubes: a high resolution 3D surface construction algorithm. Comput. Graph. **21**(4), 163–169 (1987)
CrossRef

11. Wyvill, G., McPheeters, C., Wyvill, B.: Data structure for soft objects. Vis. Comput. **2**, 227–234 (1986)
CrossRef

12. Shen, H., Johnson, C.R.: Sweeping simplicies: a fast iso-surface extraction algorithm for unstructured grids. In: Proceedings of the IEEE Visualization, Atlanta, pp. 143–150 (1995)

13. Shen, H.W., Hansen, C.D., Livnat, Y., Johnson, C.R.: Isosurfacing in span space with utmost efficiency

(ISSUE). In: Proceedings of the IEEE Visualization, San Francisco, pp. 287–294. IEEE, Los Alamitos (1996)

14. Livnat, Y., Shen, H., Johnson, C.R.: A near optimal isosurface extraction algorithm using the span space. IEEE Trans. Vis. Comp. Graph. **2**(1), 73–84 (1996)
CrossRef

15. Cignoni, P., Montani, C., Puppo, E., Scopigno, R.: Optimal isosurface extraction from irregular volume data. In: Proceedings of the IEEE 1996 Symposium on Volume Visualization, San Francisco. ACM, New York (1996)

16. Bajaj, C.L., Pascucci, V., Schikore, D.R.: The contour spectrum. In: Proceedings of the IEEE Visualization, Phoenix, pp. 167–173 (1997)

17. Livnat, Y., Tricoche, X.: Interactive point based isosurface extraction. In: Proceedings of IEEE Visualization 2004 pp. 457–464 (2004)

18. Mallo, O., Peikert, R., Sigg, C., Saldo, F.: Illuminated streamlines revisited. In: Proceedings of the IEEE Visualization, Minneapolis, pp. 19–26 (2005)

19. Li, G., Bordoloi, U.D., Shen, H.: Chameleon: An interactive texture-based rendering framework for visualizing three-dimensional vector fields. In: Proceedings of the IEEE Visualization, Seattle, p. 32, (2003)

20. Hultquist, J.P.M.: Constructing stream surfaces in steady 3D vector fields. In: Proceedings of the IEEE Visualization, Boston, pp. 171–178 (1992)

21. Garth, C., Tricoche, X., Salzbrunn, T., Bobach, T., Scheuermann, G.: Surface techniques for vortex visualization. In: Proceedings of the Joint Eurographics – IEEE TCVG Symposium on Visualization, Konstanz, pp. 155–164 (2004)

22. Erlebacher, G., Weiskopf, D.: Flow textures: high-resolution flow visualization. In: Hansen, C.D., Johnson, C.R. (eds.) Visualization Handbook, pp. 279–294. Elsevier/Academic, Amsterdam (2005)
CrossRef

23. Cabral, B., Leedom, C.: Imaging vector fields using line integral convolution. In: Proceedings of the SIGGRAPH, San Diego, pp. 263–270. ACM, New York (1993)

24. Li, G.-S., Tricoche, X., Hansen, C.D.: Gpuflic: Interactive and dense visualization of unsteady flows. In: Data Analysis 2006: Proceedings of the Joint IEEE VGTC and EG Symposium on Visualization (EuroVis), Lisbon, pp. 29–34 (2006)

25. Weiskopf, D., Ertl, T.: A hybrid physical/device space approach for spatio-temporally coherent interactive texture advection on curved surfaces. In: Proceedings of the Graphics Interface, London,

pp. 263–270 (2004)

26. Resz-Salama, C., Hastreiter, P., Teitzel, C., Ertl, T.: Interactive exploration of volume line integral convolution based on 3d-texture mapping. In: Proceedings of the IEEE Visualization, San Francisco, pp. 233–240 (1999)

27. Scheuermann, G., Tricoche, X.: Topological methods for flow visualization. In: Hansen, C.D., Johnson, C.R. (eds.) The Visualization Handbook, pp. 341–356. Elsevier, Amsterdam (2005)
CrossRef

28. SCIRun: A Scientific Computing Problem Solving Environment, Scientific Computing and Imaging Institute (SCI). Download from: http://www.scirun.org (2012)