



SCIENTIFIC VISUALIZATION

A NECESSARY CHORE

By Joel E. Tohline

From the perspective of a computational scientist who models astrophysical fluid flows, the author describes the successes and frustrations he's had with the development and use of scientific visualization techniques over the past few decades. A biased view toward the future is also presented.

For the past 30 years, I've focused my research activities on gaining a better understanding of the nonlinear development of gravitationally driven instabilities in astrophysical fluid flows. The computational fluid dynamics (CFD) techniques that my group at Louisiana State University (LSU) has employed to model various astrophysical phenomena are, broadly speaking, similar to those the engineering community uses to model compressible, supersonic flows in various terrestrial environments. Currently, in collaboration with Juhan Frank and Patrick Motl at LSU, my students and I are using these techniques to simulate mass-transfer instabilities in binary star systems for which Newton's law provides an adequate description of the time-varying gravitational field;^{1,2} through a collaboration with Luis Lehner (also at LSU), my group is expanding the capabilities of our simulation tools to model compact binary star systems, which require a general relativistic treatment of gravity.³

Not surprisingly, over the years, my research has benefited from the rapidly advancing industry of high-performance computing (HPC) and communications. It has been exciting to be involved in large-scale numerical simulation development for the past three decades, as the raw capabilities of HPC resources have routinely doubled

according to Moore's law. However, continuously upgrading my group's software simulation algorithms to keep pace with ever-improving HPC hardware has been a tremendous challenge, to say the least.

The concurrent development of hardware and software tools to facilitate scientific visualization has been extraordinary as well. With ever increasing degrees of sophistication, these tools have provided absolutely critical support for our efforts to diagnose the results of complex CFD simulations. But this, too, has come with a price. As the students who have honed their research skills in my group will attest, roughly one-third of the person-hours invested in research every year have been directed at improving our abilities to effectively visualize the results of our 3D, time-dependent CFD simulations.

Most recently, my group has begun collaborating with Claudio Silva and his colleagues from the Scientific Computing Institute (SCI) at the University of Utah to explore how our scientific visualization efforts can benefit from VisTrails' provenance-tracking capabilities. Over the next few years, Claudio and I plan to use the *CiSE* Visualization Corner and attending Web technologies as a venue for sharing highlights of our collaboration in hopes of promoting the idea

of *reproducible visualizations* (see the "Reproducibility and Sharing Data and Processes for the Visualization Corner" sidebar in last issue's column⁴). Before embarking on this new adventure, however, it's useful—and certainly therapeutic—for me to recount key events from the past couple of decades that have strongly influenced my group's past and present scientific visualization activities. Here, I illustrate how one astrophysicist with an interest in scientific visualization but, until recently, little access to in-house visualization expertise, "made do" with the tools available at the time. I suspect that many computational scientists will identify with the struggles as well as the successes associated with these recounted events, at least in spirit if not in detail.

The Eighties

In the early 1980s, as a junior faculty member at LSU, I struggled to find an affordable way to digitally image 3D isodensity surfaces from my simulations of astrophysical fluid flows. I had access to a VAX 11/750 with an International Imaging System (I²S) image-processing accelerator and display that my astronomy colleagues used to manipulate moderately high-resolution (512 × 512 pixel) 2D images of the sky. At the time, however, no software tools were available to even identify,

SCIENTIFIC VISUALIZATION AS A VEHICLE FOR OUTREACH

Figure A served for more than a year as the banner on the US National Center for Supercomputing Application's (NCSA's) News Web page (www.ncsa.uiuc.edu/News/index.html) and was one of the images featured in a recent US National Science Foundation report¹ that details a "Cyberinfrastructure Vision for 21st Century Discovery." This single image was drawn from an approximately 1,500-frame (that is, 50-second) movie that illustrates how a close binary star system that encounters an unstable phase of mass transfer can violently merge to form a single star. *The Astrophysical Journal* has archived the movie in its entirety as a digital mpeg animation to extend the content of a figure in one of my research group's 2006 refereed publications;² it's also been featured in one of the NCSA's *Access Online* interactive

and animated magazine articles (http://access.ncsa.uiuc.edu/Stories/binary_stars).

The attention drawn to this single "pink and blue" image has been quite surprising, but it illustrates very well how we can use scientific visualization to effectively communicate the results of complex numerical simulations to a very broad audience of scientists and engineers. By posting movies from a range of our simulations on YouTube (search for jet53man at www.youtube.com), we're exploring whether our scientific visualization efforts can also serve as an effective vehicle for outreach to the public at large.

References

1. "Cyberinfrastructure Vision for 21st Century Discovery," US Nat'l Science Foundation report, Mar. 2007, p. 48; www.nsf.gov/od/oci/CI_Vision_March07.pdf.
2. M. D'Souza et al., "Numerical Simulations of the Onset and Stability of Dynamical Mass Transfer in Binaries," *The Astrophysical J.*, vol. 643, 2006, pp. 381–401.

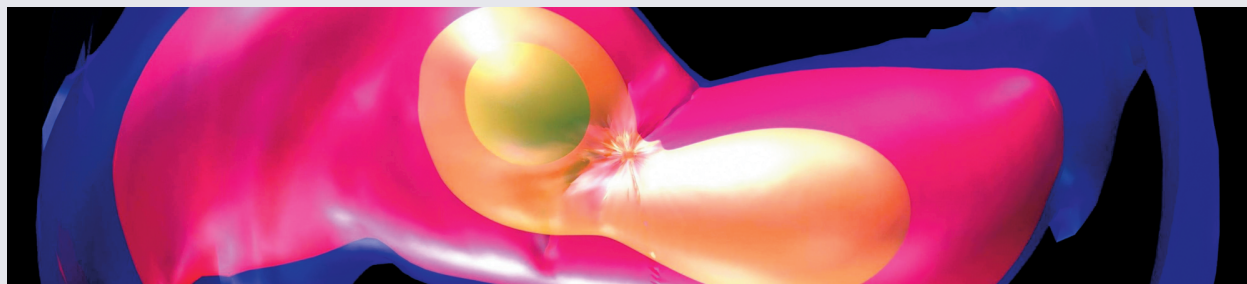


Figure A. Four nested isodensity surfaces. These images illustrate the spatial distribution of material at one instant in time during a 3D, time-dependent CFD simulation of a merging binary star system. Lighting, texture mapping, and ray-tracing were performed using Maya; an opaque, green surface identifies the region of highest density, whereas translucent yellow, red, and blue surfaces locate successively lower density regions of the flow.

let alone render, isosurfaces from our CFD flows. In 1985, on a gamble, I attended a "3D Visualization" workshop in Monterey, California, to see how researchers in other fields were tackling similar problems. This sparsely attended workshop was dominated by individuals who were exploring medical imaging techniques. Among them was Gabor T. Herman, who, at the time, was a professor in the University of Pennsylvania's radiology department. It was immediately clear to me that the techniques that he was employing to render 3D images of the human skull would suit my group's needs as well. The question was, how

much algorithm and code development would be required to transport Herman's ideas to my world?

To my delight, Herman explained that his rendering program was written in Fortran—my preferred language at the time—with relatively few details tuned specifically for his group's Data General computer. Upon his return to Pennsylvania, he very generously shipped the entire source code to me at LSU, remarking that he was pleased to be able to offer such help to an astrophysicist because his own group's research work had benefited from interactions that he'd had with radio astronomers who

were also developing 3D visualization capabilities. With the technical assistance of our VAX systems manager, Monika Lee, we tuned Herman's code to communicate with the I²S, and we were on our way. Granted, the process was slow because two to three hours of computing time were required to render an individual isosurface from a single point in time in a CFD simulation, but this was clearly a milestone in our scientific visualization efforts.

Next, my group investigated how we could affordably string together a sequence of rendered images to produce a movie of our time-dependent CFD flows. At the time, it wasn't practical to

pipe digital color images of even moderate resolution to an RGB display at the desired frame rate (for instance, 24 or 30 frames per second), so we pursued analog video-recording strategies. At no small expense, we acquired a 3/4-inch broadcast-quality Sony U-matic video recorder that could write one frame at a time onto a videotape; we also got a Lyon Lamb Mini-VAS animation controller, which we could program to control editing tasks on the video recorder, and which could hold a single video frame in its buffer for insertion onto the videotape. Converting each of our RGB-formatted digital images to a composite NTSC video signal proved to be surprisingly challenging. For several years, my group relied on a Lenco Color Encoder with sync generation capabilities to perform this translation and provide the necessary interface between our digital computer and the analog video recorder.

Encouraged by a couple of my graduate students who were NeXT Computer fanatics, a NeXTcube joined our equipment ranks in early 1989 and replaced the Lenco Color Encoder. (The NeXTcube had a built-in RGB-to-NTSC signal converter; in this regard, as well as others, the NeXT computer was revolutionary at the time.) In addition, the NeXTcube provided a friendly programming environment through which we could completely automate the sequence of steps required to generate a video from a stack of digital images. Even this automated process was excruciatingly slow, however. The bottleneck for recording each frame in the sequence was the time required for the video recorder to rewind past the desired insertion point on the tape, then get a running start so that the tape would be feeding through at its normal play rate when the anima-

tion controller sent the instruction to insert and record the new image at its assigned frame position. Under the NeXTcube's automated control, we could typically record 600 individual frames overnight, and thereby produce a 20-second movie segment. We'd reached another milestone in our scientific visualization efforts—for the first time, we could routinely view movies that illustrated the results of our complex, time-dependent 3D CFD simulations.

The Nineties

During the 1990s, the time required to render individual images from discrete points in time during each CFD simulation steadily decreased as workstations' generic processing speed increased. In conjunction with the Web's emergence, standardized digital compression algorithms came of age during the '90s. This fortunate advancement greatly simplified the steps required to turn our digital images into movies. (For quite some time, our preferred concatenation tool was "dmconvert," which was available on most Silicon Graphics [SGI] workstations.) I'm not ashamed to confess that our video recorder with single-frame editing capabilities has been gathering dust for approximately a dozen years.

In 1989, I became convinced that it was time to move beyond the introductory visualization tools Herman had given my group. In that year, a colleague of mine at Indiana University, Richard Durisen, hooked up with a scientific visualization specialist, J.B. Yost, at the University of Illinois's National Center for Supercomputing Applications (NCSA) to produce a high-quality movie that used multiple, translucent isodensity surfaces to beautifully illustrate results from a CFD simulation on which Durisen

and I had collaborated several years earlier.^{5,6} This was the first time I fully appreciated that images created using ray-tracing techniques could have significantly enhanced aesthetic appeal and reveal more detail about a 3D fluid flow's underlying structure than images constructed using volume-rendering techniques alone.

My group's search to find an accessible rendering tool to generate images that were as visually appealing as those NCSA's visualization specialists created was successful in 1997, when two graduate students and I spent a week in the relatively new Advanced Scientific Visualization Laboratory (Vislab) at the San Diego Supercomputing Center (SDSC). Although several rather sophisticated (and expensive) commercial rendering packages had been installed on the SDSC's multiprocessor SGI Onyx (such as, the Application Visualization System [AVS] originally developed by Stardent, and IBM's Data Explorer [DX]), we found that we could execute only the Alias|Wavefront package in a noninteractive batch mode via a Unix script. This requirement was critical because we needed to be able to routinely execute visualization tasks from our remote site at LSU without asking anyone to initiate an interactive session at the SDSC Vislab. Given that the ray-tracing algorithms in Alias|Wavefront also produced beautiful surface images that had very complex geometries, we quickly adopted it as our rendering package.

Alias|Wavefront did initially exhibit one weakness for our purposes: it didn't contain an algorithm for extracting isodensity surfaces from the output of our CFD simulations. Fortunately, via the Web's burgeoning capabilities, we discovered that J.J. Jensen, at the Electronics Institute

of the Technical University of Denmark (now part of the TUD's Department of Mathematical Modeling), had written a code named *polyr* (<http://hendrix.imm.dtu.dk/software/software.html>), which implemented the Lorensen and Cline⁷ "marching cubes" algorithm to perform the desired iso-surface extractions. When used in tandem with *polyr*, Alias|Wavefront's versatile rendering capabilities provided the much more sophisticated scientific visualization tool we'd been seeking. To our delight, we discovered that the wireframe models *polyr* generated could also readily be formatted in Virtual Reality Markup Language (VRML) and fed into Web-browser-based VRML viewers for a taste of virtual reality (see more on this topic in the next section).

Shortly after returning to LSU from the SDSC, a colleague informed me that a visualization facility patterned after the SDSC Vislab had just been constructed to support the US Department of Defense's (DoD's) Naval Oceanographic Office (NAVOCEANO) at NASA's John C. Stennis Space Center on the Mississippi Gulf Coast. The NAVOCEANO Vislab had also installed Alias|Wavefront on an SGI Onyx and, for a few years, we were able to take advantage of the center's geographic proximity to LSU while exploring better ways to visualize data from our simulations. In particular, guided by the NAVOCEANO Vislab staff's expertise, we tuned the lighting and texture-mapping choices within Alias|Wavefront to significantly improve our rendered images' quality. Unfortunately, in 2001, our informal access to this facility disappeared as much tighter security restrictions went into effect at all DoD laboratories.

Throughout the '90s, my group

also explored ways we could generate the time sequence of rendered images necessary for creating a movie "on the fly" during each CFD simulation so that we could significantly reduce the requirements for data storage and the person-hours devoted to postprocessing tasks. Initially, we developed a heterogeneous computing environment that accomplished this objective using existing computing platforms within LSU's Department of Physics and Astronomy.⁸ Specifically, we performed our CFD simulations on an 8,192-node MasPar MP-1 and the primary volume-rendering tasks on a Sun Microsystems Sparcstation using volume visualization routines within the Interactive Data Language (IDL). At predetermined intervals during a simulation, our CFD code would recognize that we needed to construct a volume-rendered image of the flow for inclusion in an animation sequence. The code would then write one 3D data array to a disk that was mounted on both the MasPar and the Sparcstation as part of a Unix Network File System (NFS). The MasPar then passed process control for the visualization task to the Sparcstation via Unix sockets. After spawning the visualization task, the MasPar would continue following the fluid flow's evolution, running the CFD simulation in parallel with the visualization task on the Sparcstation. As the volume-rendering algorithm finished generating each image, it would immediately delete the 3D data set, thereby conserving substantial disk space.

In 1998, we constructed an analogous heterogeneous computing environment at the SDSC.⁸ We conducted our CFD simulations on the Cray T3E and performed primary visualization tasks on Vislab's SGI Onyx2 using Alias|Wavefront software, as we described earlier. In this case, the two

selected hardware platforms didn't share cross-mounted disks, so we had to rely on FTP commands to transfer data from the T3E to a disk mounted on the Onyx2, and pass process control for the visualization task via a remote shell script. Typically, during an eight-hour overnight CFD simulation, the Cray T3E would send 60 3D data sets and accompanying image requests to the Onyx2. Then, at the end of a typical overnight simulation, 60 images—sufficient for producing an animation sequence only two seconds long—would be automatically transferred to LSU for us to concatenate with images generated on earlier nights. Through this heterogeneous computing environment, we routinely created 15- to 30-second movies to illustrate CFD simulation results that required 60 to 120 wall-clock hours to complete on the Cray T3E.

The New Millennium

Thankfully, technology industries and open source user groups are continuing to lighten some of the load associated with scientific visualization through advancements in, for example, workstation and PC capabilities, compression algorithms and video-editing tools, and rendering software. The Alias|Wavefront product Maya, for example, now offers a very affordable means for performing rendering and ray-tracing tasks on Apple G5 or Linux-PC platforms. Additionally, Final Cut Pro is an affordable PC-based tool for editing, concatenating, and annotating movies. Through collaboration with Claudio Silva, my group is adding VisTrails to our arsenal and, as a result, we're becoming familiar with the diverse and rapidly expanding offerings of the open source Visualization Toolkit (VTK; www.vtk.org).

A few years ago, Wes Even, a graduate student in my group, wrote his own

marching cubes algorithm to replace J.J. Jensen's polyr routine in our visualization toolkit. Even's new routine is more suitable to our needs in two respects: first, it can identify the vertices and polygons that define isosurfaces in our CFD flows directly from the curvilinear coordinate system we use in our CFD simulations, thus bypassing the mapping to Cartesian coordinates that was required in conjunction with polyr. Second, we can integrate it with and execute it in our CFD algorithm's parallel environment. When our CFD code is run on, say, 256 processors of a supercomputer, and it recognizes that a volume-rendered image of the flow needs to be included in an animation sequence, Even's routine executes on all 256 processors to identify—within the subset of data that resides in each processor's memory—the vertices and polygons that lie on various segments of each desired isosurface. Rather than transferring the entire 3D data set, the CFD code transfers these vertices and polygons via parallel I/O channels to an external disk for visualization purposes. The computer that handles visualization tasks concatenates these separate lists of vertices and polygons to sew the various isosurface segments together before feeding each assembled surface into the rendering program (Maya). The new capabilities provided by Even's parallel marching cubes algorithm have further reduced the time and overhead associated with our routine scientific visualization tasks.

It's embarrassing to admit that, in the past, we've invested relatively little effort in determining how to visualize the time-dependent vector (principally velocity) flow fields that naturally accompany all our CFD simulations. Of course, we periodically examine static plots of component velocity fields in 2D slices through

our 3D computational domains, but we haven't developed or identified tools that permit us to routinely study the behavior of time-dependent, 3D velocity fields. On this count, I plead exhaustion. As I detailed earlier, my group has invested a considerable amount of effort over the years just putting together the tools to routinely analyze time-dependent scalar (for example, matter density) fields. We simply haven't had the person-power available to adequately tackle this generally more difficult scientific visualization task. In the absence of such visualization tools, I'm certain that we've failed to fully appreciate the richness of the flows that have developed in our simulations. In the new millennium, we're particularly anxious to alleviate this shortcoming.

I'm also anxious for more realistic, less cumbersome, and less expensive virtual reality environments to become available for scientific visualization. Even unsophisticated Web-browser-based VRML viewers illustrate that it's useful to be able to "fly around" a rendered 3D structure in real time. In the late '90s, principally through interactions with the materials simulation group at LSU, my group gained access to an ImmersaDesk and could thus explore how wide field-of-view, stereoscopic visualizations can assist the human brain in interpreting the results of 3D fluid simulations. Several months ago in the newly constructed six-wall Cave Automatic Virtual Environment (CAVE) at the Louisiana Immersive Technologies Enterprise (www.lite3d.com), we were also able to literally walk inside a rendering of 3D isodensity surfaces from one of our simulations. I'm willing to believe that we can enhance our understanding of numerical simulations through the routine use of a CAVE, an ImmersaDesk, or other

similar facility. However, experience tells me that additional dedicated staffing is required if such environments are to effectively support, rather than frustrate, the typical researcher; furthermore, proximity to users is essential if such facilities are to provide routine scientific visualization support.

Personally, I'm looking forward to seeing accelerated development in digital holographic techniques over the coming decade. Digital monitors that can display holographic movies in which every frame possesses the detail and clarity already realized in the large-scale, static white-light holograms produced by Zebra Imaging (www.zebraimaging.com), for example, will truly revolutionize the art of scientific visualization, not to mention the entertainment industry. I'm sufficiently convinced of the positive impact that such tools and techniques will have on our field that I've recently collaborated with an LSU computer science graduate student to explore what developments are required to advance computer-generated holography toward mainstream use.⁹

Disappointment, Frustration, and Optimism

My biggest frustration has been my group's inability to adequately present and properly record the results of our numerical simulations in the professional literature. The custodians of archival publications in our science and engineering community—generally speaking, our various professional societies—have been slow to appreciate that other researchers often can't understand, properly review, or fully appreciate numerical simulation results without help from visualization tools. For the most part, manuscripts that report the results of complex simulations must conform to the printed

page—a “lowest common denominator” format that our scientific forefathers would have recognized more than a century ago. Although many professional journals will now accept, for example, mpeg-formatted movies along with an electronic manuscript, such movies are generally considered supplemental, rather than essential, material for readers and for the archival record. When publishing the results of a 3D, time-dependent CFD simulation, at the very least we should have at our disposal all of the dynamic presentation tools available through Web browsers. It would be much more satisfying, however, to be able to archive the entire data set from a simulation (in addition to a manuscript, sample illustrations, and an example movie) along with VTK, for example, thereby enabling interested readers to dissect and analyze results from the simulation in whatever way they see fit.

As computational scientists, our contributions to the fields of science and engineering would be even more valuable if the published results from each of our numerical simulations also included an archival record of the computer programs that we used to generate the results, along with the complete set of parameters and data that we used to specify, for example, initial conditions for each simulation. Without this level of detail and completeness, simulation results are virtually impossible to reproduce, and until reproducibility is attached to simulation-guided research, it will be difficult to consider the results of such research on equal footing with experimental results. Building on the experience we’ve gained over the years in identifying or developing tools to effectively visualize results from our CFD simulations, my

group at LSU is looking forward to exploring VisTrails’ capabilities to track provenance and thereby better ensure the reproducibility of our work. You can follow our progress over the next few years at www.vistrails.org/index.php/CiSE.

Acknowledgments

Beyond those whose names I mentioned in the article, I’d like to acknowledge the important contributions that various LSU graduate students have made toward the development and clever application of my group’s scientific visualization tools. John Woodward and Paul Fisher were the NeXTcube fanatics who guided development work in the late ’80s; John Cazes and Howard Cohl were the architects of our heterogeneous computing environment in the late ’90s; Eric Barnes wrote the Maya script; and Richard Muffoletto is the computer science student who has shared my interest in and fascination with computer-generated holography over the past few years. Several LSU undergraduates have also made important contributions: Jeffrey Anderson, Donovan Domingue, Jack McGee, Peter Nelson, and David Sherfese. My students and I also benefited from our interactions with LSU’s materials simulation group, led by Priya Vashishta, Rajiv Kalia, and Aiichiro Nakano. Our research has been generously supported through US National Science Foundation and NASA grants, most recently NSF/AST-0708551, NSF/AST-0407070, and NASA/ATP-NNX07AG84G, and through the allocation of computing resources at various national supercomputing centers, including NCSA and SDSC through allocation MCA98N043 and AST070025. LSU and the state of Louisiana have been generous in acquiring robust hardware and skilled personnel

to support a vibrant cyberinfrastructure, especially through the establishment of the Center for Computation & Technology (CCT) and the Louisiana Optical Network Initiative (LONI).

References

1. M. D’Souza et al., “Numerical Simulations of the Onset and Stability of Dynamical Mass Transfer in Binaries,” *The Astrophysical J.*, vol. 643, 2006, pp. 381–401.
2. P.M. Motl et al., “The Stability of Double White Dwarf Binaries Undergoing Direct Impact Accretion,” to be published in *The Astrophysical J.*, 2007; <http://arxiv.org/abs/astro-ph/0702388>.
3. M. Anderson et al., “Simulating Binary Neutron Stars: Dynamics and Gravitational Waves,” submitted to *Physical Review D*, 2007; <http://arxiv.org/abs/0708.2720>.
4. C.T. Silva, J. Freire, and S.P. Callahan, “Provenance for Visualizations,” *Computing in Science & Eng.*, vol. 9, no. 5, 2007, p. 83.
5. R.H. Durisen et al., “Two-Armed Instability of a Rotating Polytropic Star: A 3D Animation of Supercomputer Results,” *Bulletin Am. Astronomical Soc.*, vol. 21, 1989, p. 794.
6. R.H. Durisen et al., “Birth of a Spinning Star,” *Sky & Telescope*, Sept. 1989, p. 239.
7. W.E. Lorensen and H.E. Cline, “Marching Cubes: A High Resolution 3D Surface Construction Algorithm,” *Proc. Computer Graphics (SIGGRAPH 87)*, vol. 21, no. 4, 1987, pp. 163–170.
8. J.E. Cazes et al., “A Heterogeneous Computing Environment to Simulate Astrophysical Fluid Flows,” *Department of Defense Users Group Conf. (UGC 99)*, 1999; www.phys.lsu.edu/astro/cazes/dodconf/hce.html.
9. R.P. Muffoletto, J.M. Tyler, and J.E. Tohline, “Shifted Fresnel Diffraction for Computational Holography,” *Optics Express*, vol. 15, no. 9, 2007, pp. 5631–5640.

Joel E. Tohline is a professor at Louisiana State University. His research interests include astrophysics, computational fluid dynamics, and high-performance computing. Tohline has a PhD in astronomy from the University of California, Santa Cruz. He is a member of the AAAS, the International Astronomical Union, the American Astronomical Society, and the American Physical Society. Contact him at tohline@lsu.edu.