Editors: Cláudio T. Silva, csilva@cs.utah.edu
Joel E. Tohline, tohline@lsu.edu

VISUALIZING A JOURNAL THAT SERVES THE COMPUTATIONAL SCIENCES COMMUNITY

By Joel E. Tohline and Emanuele Santos

An online article with a VisMashup application illustrates the advantages of an interactive journal over traditional printed formats.

Let's face it: The printed journal format that has served the scientific research community satisfactorily for more than 200 years doesn't serve the computational sciences community well at all. The community should, instead, communicate and archive the results of its research endeavors through a venue that lets students and colleagues fully examine published computational results and critique the numerical algorithms the authors used to generate them. A suitable computational sciences journal must principally reside in the digital rather than hardcopy world and, we propose, should consist of an interactive scientific visualization tool that is fully embedded within a versatile Wiki. The basic framework for such a journal already exists within the open source community. As we show here, for example, the SCI Institute's VisTrails Wiki (www.vistrails.org) provides many features that would be desirable in an interactive, visual, and archival computational sciences journal.

It's impossible to effectively illustrate the advantages of an interactive, visual, and archival journal (IVAJ) over a traditional journal through a *CiSE* Visualization Corner article, which itself is intended principally to appear in print. Hence, to fully engage in this discussion, we encourage readers to interact with an article that we've posted in a proposed IVAJ format. We provide instructions for that interaction here; you can also find an abbreviated set of instructions at www.vistrails.org/index.php/User:Tohline/IVAJ. Another useful reference is our May/June 2009 Visualization Corner article.[1] Here we demonstrate how an IVAJ can help that published article's printed pages come alive and can considerably expand on its basic scientific content.

Level 1 Enhancements

First, go to www.vistrails.org/index.php/User:Tohline/IVAJ/Level1. This will open a document within a Wiki on the VisTrails server at the University of Utah's SCI Institute that contains essentially the material that we published in the traditional manner in *CiSE*'s May/June 2009 issue. This Wiki document's main text is identical to the printed article, as are the three figures and the "Terminology" sidebar.

We've added several straightforward Level 1 enhancements to this online Wiki article. For example, we've linked each cited reference to an online version of the identified article and each author's name to his or her Web page. We've also added an active HTML anchor to each author's email address and, for each figure, we've added a link to a high-resolution version. These are the typical enhancements that scientific journals offer when posting a printed archival article online. They add relatively little extra value to the article—although, in this particular case, Figure 1's high-resolution version does make the text and inset images more readable.

One Level 1 online enhancement that provides substantive value over the printed page is the ability to cut and paste the Python source code provided in the article's "SwitchCoord Python Module" sidebar. The sidebar's printed version had drawbacks from the beginning because it didn't preserve all of the source code's line indentations. This isn't good, of course; Python uses indentations for block delimiters and the indentation level determines the statement groupings. Our Wiki document preserves the sidebar's Python source code structure and archives the source code.

Posting the article within a Wiki also lets readers comment on or even edit the article. (To use the Wiki's editing capabilities, of course, the reader must be granted user access to the VisTrails Wiki server. Try it out!) Comments or additions to a posted article by professional colleagues—or, at a later time, by any of the original authors—shouldn't detract from the article's original "archived" content because, by design, the Wiki preserves the articles' provenance. And, such comments or additions could significantly improve an article's long-term value.

Level 2 Enhancements

Next, go to www.vistrails.org/index.php/User:Tohline/IVAJ/Levels2and3. Here, you'll see a document from the VisTrails Wiki server that contains

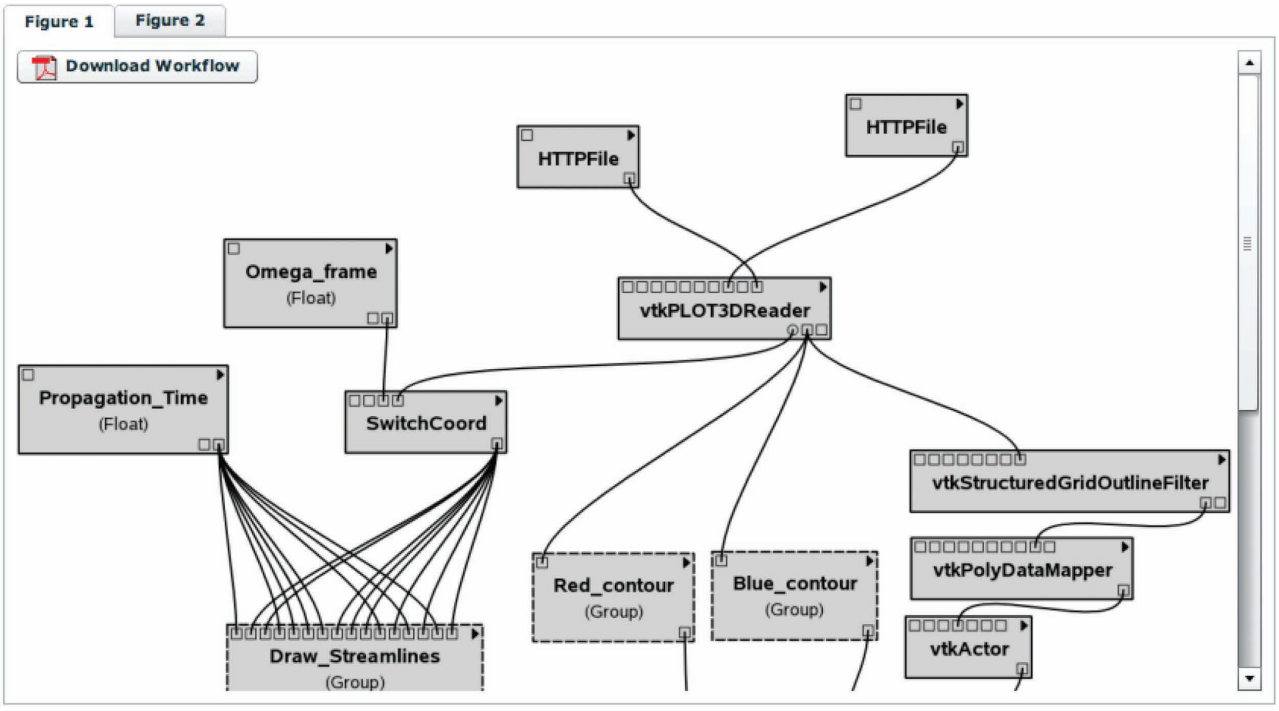

Figure 1. The VisMashup window that displays when users select the "Figure 1" tab (see www.vistrails.org/index.php/User:Tohline/IVAJ/Levels2and3). The image shows a layout of the VisTrails workflow that we used to generate Figure 2.

essentially the same text as the document discussed earlier, but we've replaced Figures 1 and 2 with an embedded VisMashup application[2] (hereafter, *App*). When you initially load the Wiki page containing these figures in your browser, it projects results that we've generated using a VisTrails workflow execution on the same server that hosts the Wiki. Unlike the Level 1 document or the article's original printed version, the Level 2 Wiki document's Figure 2 isn't a screenshot of the VisTrails spreadsheet window. Instead, Figure 2 contains an image directly generated by the VisTrails workflow that is sent to your browser window by the server that executed the workflow. And Figure 1 doesn't contain a screenshot of the VisTrails builder window; it contains a layout—and therefore also an archival record—of precisely the workflow that the server used to generate Figure 2.

Initially, the embedded App's results might take 10 to 15 seconds to download. If the embedded App images, buttons, and associated scrollbars aren't initially fully accessible inside the App window, you can use your browser's standard "zoom out" menu option to adjust the fit.

Initially, Figure 2 shows one result from a VisTrails parameter exploration in which we've set the value of the model parameter `omega_frame` to −0.041. As explained in the original article, this value best explains the simulation's underlying physics. Through the embedded VisMashup App, however, you can explore other parameter values and, in doing so, change the image displayed in Figure 2. For illustration purposes, we've configured the Figure 2 App so that you can select any of five discrete values for the `omega_frame` parameter: −0.06, −0.041, −0.02, 0.0, and +0.02. These correspond to the values we used to generate the original Figure 2's composite image. When you select a new value for the `omega_frame` parameter and click the green "Update" button within the Figure 2 window, the Wiki server will execute Figure 1's VisTrails workflow using the new parameter value and display the resulting image as a new Figure 2. (Clicking on the Figure 2 image opens a new browser window that stores the displayed image for comparison with subsequently generated images.)

Be patient: for each update, the server must execute a nontrivial visualization workflow on a moderately large dataset, so the new image typically won't appear in the Figure 2 window until 30 to 90 seconds after you click the "Update" button.

We could've configured the VisMashup App to let users type in `omega_frame` values other than our selected set of five discrete values. Instead, we chose to let users explore two model parameters—`propagation_time` and `rho_min`—whose variations weren't discussed in the original published article. (We recommend using parameter values in the range $0.1 <$ `propagation_time` < 3.5 and $0.0005 <$ `rho_min` < 0.05.) The `propagation_time` appears explicitly as a VisTrails workflow parameter (see Figure 1) that we use to examine how far test particles residing in different flow field regions will travel in a given amount of time; in general, the collection

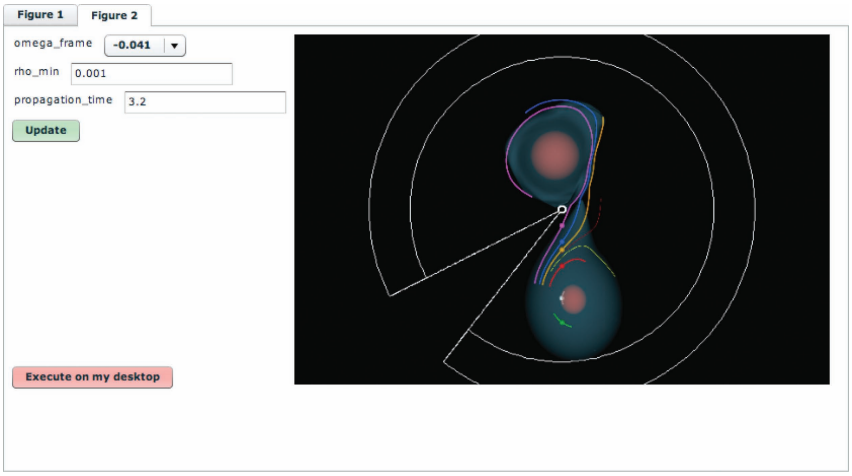

Figure 2. The VisMashup window that displays when users select the "Figure 2" tab (see www.vistrails.org/index.php/User:Tohline/IVAJ/Levels2and3). The window displays an image generated by a customized VisTrails workflow using the indicated values of the three variable parameters, `Omega_frame` (= $\Delta\Omega$), `rho_min`, and `Propagation_time`. The VisMashup App generates a new image in the online article (in accordance with the workflow shown in Figure 1) if the reader selects a different set of parameters and clicks the green "Update" button. Clicking on the red "Execute on my desktop" button downloads the Figure 1 workflow to the reader's computer system for local execution.

of streamlines will shorten if we specify a smaller `propagation_time` value. As the article's "SwitchCoord Python Module" sidebar describes, `rho_min` is an additional parameter that the customized Python module uses; individual streamlines are truncated once the test particle traveling along that streamline enters a region where the gas density is less than `rho_min`. (Densities have been normalized such that the model's maximum density is unity.)

This Level 2 enhancement lets users examine more thoroughly the astrophysical model that we focused on in the original printed article. By actively adjusting one or more of the key model parameter values and using the embedded VisMashup App to generate a new figure based on those values, users likely will gain a better appreciation of our original article's conclusions. Further, using the Wiki's standard editing features, users can comment on the insights they've gained from examining a range of model parameters outside those originally discussed.

We invested considerable time in our original article, piecing together a visualization workflow that let us satisfactorily analyze the underlying properties of the flow that resulted from our astrophysical fluid simulation. It's not unusual for computational sciences researchers to invest such time on postprocessing analysis (especially on visualization tasks). In the original article, we captured the scientific fruits of this labor in two static images (Figures 2 and 3). Our embedded VisMashup App executes exactly the same visualization workflow as the original article. Hence, with the investment of relatively little additional time, we can bring the original figures to life and reap additional benefits from our original code-development efforts.

It's important to note that each time a user changes a parameter value and executes the VisMashup App, it performs the requested analysis *on the original model data*. That is, we've archived the original astrophysical fluid simulation's model data to support our effort to enhance the article's content. This is a step in the right direction, as efforts to demonstrate the reproducibility of large-scale numerical simulations aren't likely to succeed until the computational sciences community makes a commitment to archive simulation results. Our IVAJ-formatted article with Level 2 enhancements illustrates how such archival data can naturally enrich the content of published journal articles.

Level 3 Enhancements

As the example at www.vistrails.org/index.php/User:Tohline/IVAJ/Levels2and3 shows, our IVAJ article offers yet another enhancement level over traditional journal articles. By clicking the red "Execute on my desktop" button displayed within the Figure 2 window of the VisMashup App, users can execute Figure 1's VisTrails workflow on their own computers. Of course, they can realize this Level 3 enhancement only if they've previously installed VisTrails (version 1.4.2 or later) as a functioning application on their local system. (VisTrails is an open source application designed to run under a wide range of operating systems, so we hope this local installation requirement won't discourage readers from exploring and considering the added value that such applications can bring to a modern IVAJ.)

Following the local execution of Figure 1's workflow using the model parameters initially displayed in Figure 2, the App displays the rendered configuration outside the browser, in one cell of a VisTrails spreadsheet. (The initial download and execution can take 10 minutes or more, depending on network transfer speeds.) Using the spreadsheet's interactive tools—such as rotate, zoom, and pan, which are now standard within virtually all scientific visualization applications—users can immediately examine the model's full 3D structure. This brings to life the static, 2D image that appeared in the original published article. Numerous features of the complex 3D mass distribution and flow field

that were indiscernible in the published image become clear when the IVAJ-formatted article taps into and activates the local computer's built-in hardware rendering capabilities.

Clicking Figure 2's red button does much more than just breathe life into the original static 2D image. It loads the workflow pipeline into the VisTrails application. It also transfers the archived model data to the user's local system. Anyone accustomed to using VisTrails immediately knows how to change any number of workflow parameters—beyond the three illustrated above by our embedded VisMashup App—or, indeed, to modify workflow modules and thus examine the archived model's structure in depth. For example, we illustrated only two isodensity surfaces and seven streamlines of the flow in the published and App-created images displayed above as Figure 2. With access to the entire archived model data and to our analysis tool, a colleague or student could examine other features, properties, and/or regions of the flow either from general curiosity or with an eye toward reproducing, verifying, or validating the published results.

We urge members of the computational sciences community—and the funding agencies that support this rapidly growing, multidisciplinary field—to vigorously promote development and adoption of an IVAJ of the type illustrated here. We also encourage readers of this *CiSE* Visualization Corner article to request access to the VisTrails Wiki if they're interested in discussing the need for a modern, interactive computational sciences journal. We'll report on this expanded discussion in a later Visualization Corner article if the level of expressed interest is sufficiently high.

Acknowledgments

Interactions with Claudio Silva, Juliana Freire, Will Hires, and William Armstrong have significantly benefited our project's progress. The US National Science Foundation (AST-0708551, DGE-0504507), the US Department of Energy, NASA (NNX07AG84G and NNX10AC72G), and IBM have supported this work, with mass-transfer simulations made possible through computing time allocations on TeraGrid resources (TG-MCA98N043) at the National Center for Supercomputer Applications (NCSA) and through the Louisiana Optical Network Initiative (LONI).

References

1. J.E. Tohline et al., "A Customized Python Module for CFD Flow Analysis within VisTrails," *Computing in Science & Eng.*, vol. 11, no. 3, 2009, pp. 68–73.
2. E. Santos et al., "VisMashup: Streamlining the Creation of Custom Visualization Applications," *IEEE Trans. Visualization and Computer Graphics*, vol. 15, no. 6, 2009, pp. 1539–1546.

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's a fellow of the American Association for the Advancement of Science and a member of the International Astronomical Union, the American Astronomical Society, and the American Physical Society. Contact him at tohline@lsu.edu.

Emanuele Santos is a research assistant and PhD candidate at the University of Utah. Her research interests include scientific data management, visualization, and comparative visualization. Santos has an MS in computer science from the Federal University of Ceara in Brazil. Contact her at emanuele@sci.utah.edu.