Announcing VisTrails 1.0

by the VisTrails Group

VisTrails is a new system that provides data and process management support for exploratory computational tasks. It combines features of both workflow and visualization systems. Like many workflow systems, it enables seamless integration of loosely-coupled resources such as specialized libraries, grid, and web services. Likewise, it parallels some visualization systems by providing a mechanism to perform parameter explorations and result comparisons. But unlike these systems, VisTrails was designed to manage exploratory activities, where computational tasks are iteratively refined as users formulate and test hypotheses. A key distinguishing feature of VisTrails is a *comprehensive provenance infrastructure* that maintains detailed history information about the steps followed and data derived in the course of an exploratory task. VisTrails leverages this information to provide novel operations and user interfaces that streamline this process.

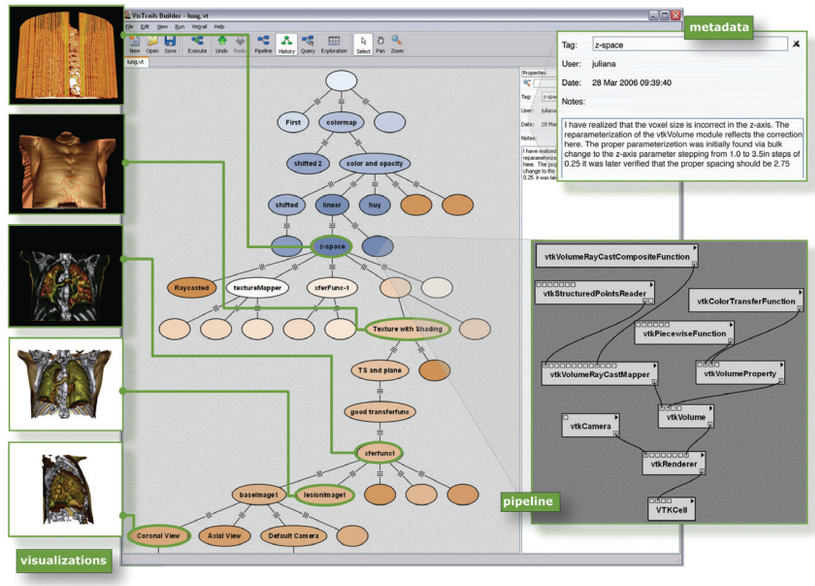

Fig 1: The VisTrails history tree contains a node for each version of a workflow (or pipeline) as it evolves over time. This results in a complete audit trail of the steps that were taken in a computational task.

Project History. Professors Juliana Freire and Claudio Silva started this project in 2005 and they have been funded by grants and contracts from NSF, DOE, and IBM. The VisTrails team includes six Ph.D. students as well as M.S. and undergraduate students. Since its early development stages, the VisTrails system has been available to a number of external collaborators who have provided invaluable feedback. These include researchers at Cornell University, California Institute of Technology, The Oregon & Health Science University, and the University of Utah. A beta version of VisTrails was made available (as open source) in late January 2007. Since then, the system has been downloaded over 2000 times. VisTrails 1.0 will be released on October 31st.

On the Need of Provenance for Computational Tasks. Computing has been an enormous accelerator for science, leading to an information explosion in many different fields. Future scientific advances depend on our ability to comprehend the vast amounts of data currently being produced and acquired. However, to analyze and understand this data we must assemble complex computational processes and generate insightful visualizations, which often require combining loosely coupled resources, specialized libraries, and grid and Web-services. Such processes generate yet more data, adding to the information overflow scientists currently deal with. Today, the scientific community uses ad hoc approaches for data exploration, but such approaches have serious limitations. In particular, scientists and engineers must expend substantial effort to manage these data (such as scripts that encode computational tasks, raw data, data products, images, and notes) and record provenance so that they can answer basic questions: Who created a data product and when? When was it modified, and who modified it? What process was used to create the data product? Were two data products derived from the same raw data? This process is not only time-consuming, but also error-prone. Without provenance, it's difficult (and sometimes impossible) to reproduce and share results, solve problems collaboratively, validate results with different input data, and understand the process used to solve a particular problem. In addition, the longevity of data products becomes limited—without precise and sufficient information about how the data product was generated, its value diminishes significantly.

The lack of adequate provenance support in visualization and workflow systems motivated us to build VisTrails, an open source provenance management system that provides infrastructure for data exploration and visualization through workflows. VisTrails transparently records detailed provenance of exploratory computational tasks (see Fig. 1). This information not only enables the reproducibility of results, but it also allows scientists to easily navigate through the space of workflows and parameter settings used in a given exploration task. Powerful operations are also possible through direct manipulation of the provenance information. These include the ability to re-use workflows and workflow fragments through a mechanism for refining workflows by analogies; to explore a multi-dimensional slice of the

parameter space of a workflow and generate a large number of data products through bulk-updates; to analyze (and visualize) the differences between two workflows (see Fig. 2); and to support collaborative data exploration in a distributed and disconnected fashion. These operations, combined with an intuitive interface for comparing the results of different workflows, simplify, to a great extent, the scientific discovery process.

VisTrails provides a comprehensive provenance management infrastructure that can be combined with and extend existing workflow and visualization systems. Some distinguishing features of the system include:

- *Flexible Provenance Architecture.* VisTrails transparently tracks changes made to workflows by recording all the steps followed during the exploration. The system can optionally track run-time information about the execution of workflows (e.g., who executed a module, on which machine, elapsed time etc.). VisTrails also provides a flexible annotation framework whereby users can specify application-specific provenance information.
- *Querying and Re-using History.* The provenance information is stored in a structured way. Users have a choice of using a relational database (e.g., MySQL and IBM DB2) or XML files in the file system. The system provides flexible and intuitive query interfaces through which users can explore and re-use provenance information. Users can formulate simple keyword-based and selection queries (e.g., visualizations created by a given user) as well as structured queries (e.g., visualizations that apply simplification before an isosurface computation for irregular grid data sets). VisTrails also provides a query-by-example interface in which users create queries using the same interface they use to construct workflows.
- *Support for collaborative exploration.* The system can be configured with a database backend that is used as a shared repository for generated workflows and provenance information. It also provides a synchronization facility that allows users to collaborate asynchronously and in a disconnected fashion: users can check in and check out changes, akin to a version control system (e.g., svn).
- *Extensibility.* VisTrails provides a very simple plug-in functionality that can be used to dynamically add packages and libraries. Neither changes to the user interface nor recompilation of the system are necessary. Because VisTrails is written in Python, the integration of Python-wrapped libraries is straightforward.
- *Scalable Derivation of Data Products and Parameter Exploration.* VisTrails supports a series of operations for the simultaneous generation of multiple data products including an interface that allows users to specify sets of values for different parameters in a workflow. The results of parameter explorations are displayed side by side in the VisTrails Spreadsheet for easy comparison.
- *Task Creation by Analogy.* Analogies are supported as first-class operations to guide semi-automated changes to multiple workflows without requiring users to directly manipulate or edit the workflow specifications.

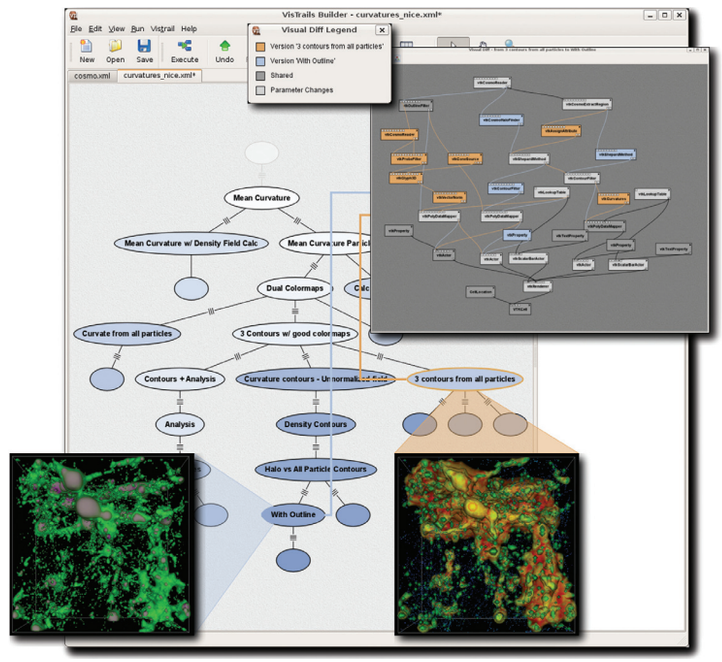

Fig 2: The VisTrails Visual Difference interface shows the difference between two nodes in the history tree as an annotated workflow. Modules that are unique are shown in orange and blue, modules that contain a parameter change between versions are shown in light gray, and unchanged modules are shown in dark gray. This feature is important for understanding a computational task by allowing the user to interact with their history.

Obtaining the Software. Visit http://www.vistrails.org to access the VisTrails community Web site. There, you will find information including instructions for obtaining the software, online documentation, video tutorials, and pointers to papers and presentations. VisTrails is written in Python and it uses the multi-platform Qt library for its user interface. The system is available as open source; it is released under the GPL 2.0 license. The pre-compiled versions for Windows, Mac OS X, and Linux, come with an installer and include a number of packages, including VTK, matplotlib, and ImageMagick. Additional packages, including packages written by users, are also available (e.g., ITK, Matlab, Metro).