VisTrails

Erik Anderson, Jason Callahan, Steven Callahan, Juliana Freire*, David Koop, Emanuele Santos, Carlos Scheidegger, Claudio Silva*, and Huy Vo. *PI

University of Utah, SCI Institute

VisTrails provides a comprehensive provenance management infrastructure for computational tasks that are modeled as workflows. A key differentiator in VisTrails is its support for exploratory tasks such as visualization and data mining, where workflows are iteratively refined as users generate and test hypotheses. VisTrails introduced a new model that, in addition to data provenance, captures information about how workflows evolve over time. The system uses a change-based model that uniformly captures both changes to parameter values and to workflow definitions by unobtrusively tracking all changes that users make to workflows.

The stored provenance ensures result reproducibility and it also allows users to easily navigate through the space of workflows created for an exploration task. The VisTrails interface gives users the ability to query, interact with, and understand the history of the exploration process. In particular, they can return to previous versions of a workflow and change the specification or parameters to generate a new set of results without losing previous changes. Another important feature of the change-based provenance model is that it enables a series of operations that greatly simplify the exploration process and have the potential to reduce the time to insight. In particular, it allows the flexible re-use of workflows and it provides a scalable mechanism for creating and comparing a large number of data products as well as their corresponding workflows.

Important Features:

Flexible Provenance Architecture: VisTrails transparently tracks changes made to workflows as well as run-time information about their execution. It also provides a flexible annotation framework whereby users can specify application-specific provenance information.

Querying and Reusing History: The system provides intuitive query interfaces through which users can explore and re-use provenance information. Users can formulate simple keyword-based and selection queries as well as structured queries.

Support for Collaborative Exploration: VisTrails can be configured with a database backend that can be used as a shared repository. Users can also collaborate in a disconnected fashion by checking in their changes and synchronizing their vistrails against the repository. Users may also compare different workflows and their results through a combination of a visual difference interface and an interactive spreadsheet.

Extensibility: A plugin mechanism is provided that allows packages and libraries to be added VisTrails.

Scalable Derivation of Data Products: VisTrails supports the simultaneous generation of multiple data products through a parameter exploration interface.

Task Creation by Analogy: In VisTrails, new data products can be created semi-automatically using existing workflows, without requiring the user to directly manipulate or edit the actual workflow specification.

Funding provided by

www.vistrails.org

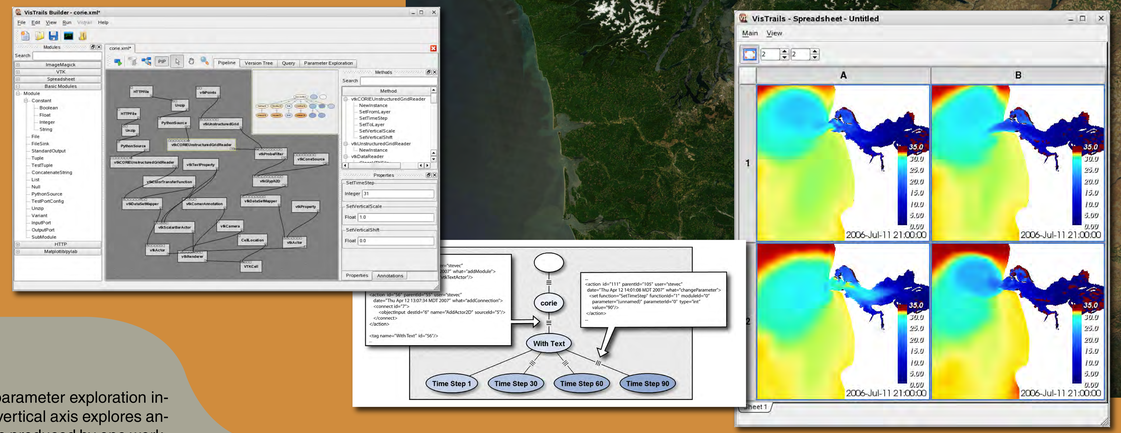
VisTrails Components

Each node in the Vistrails History Tree represents a workflow version. An edge between a parent and child nodes represents a set of actions applied to the parent to obtain the workflow for the child node.

The Builder allows users to create and modify workflows, while transparently tracking all the changes.

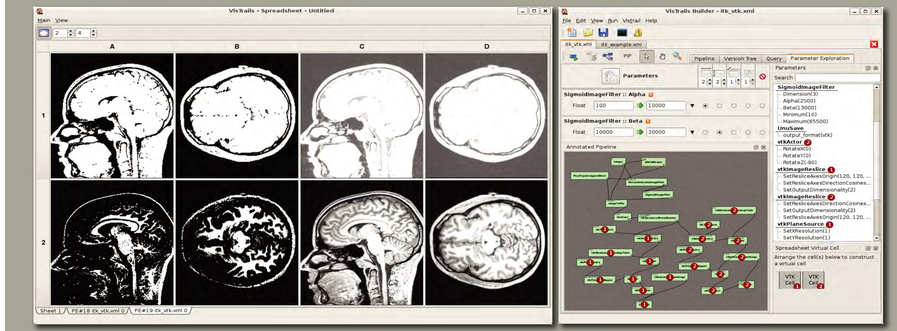
The Spreadsheet allows the results of multiple workflows or workflow instances to be compared side-by-side.

The spreadsheet on the right shows fours visualizations of fresh-water discharge from the Columbia River for different time steps. Each visualization is derived by a different workflow in the history tree displayed in the center.



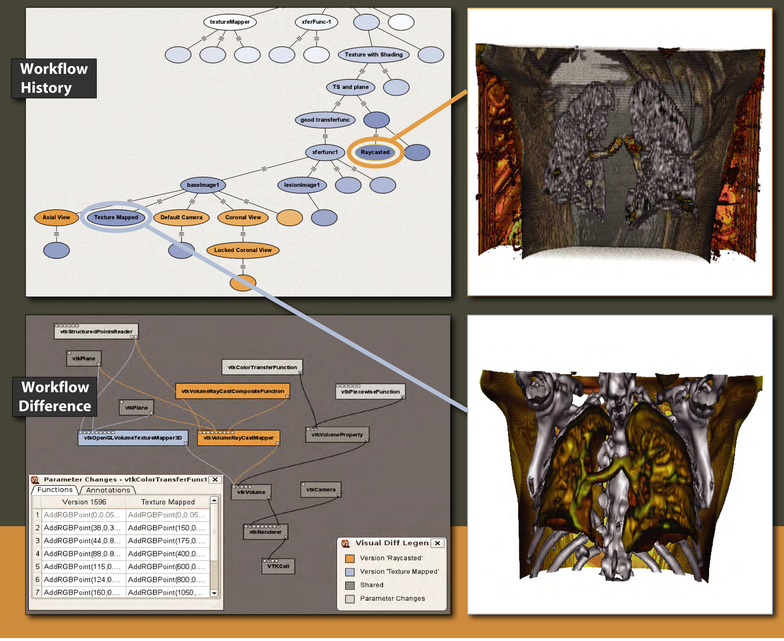
Parameter Exploration:

This spreadsheet contains visualizations of MRI data of a head and was generated using the parameter exploration interface shown on the right. The horizontal axis varies one parameter of an image filter and the vertical axis explores another. The interface allows the user to manage the layout of cells when multiple visualizations are produced by one workflow as shown with the two views (labeled 1 and 2), as well as to synchronize cells with respect to different parameters.



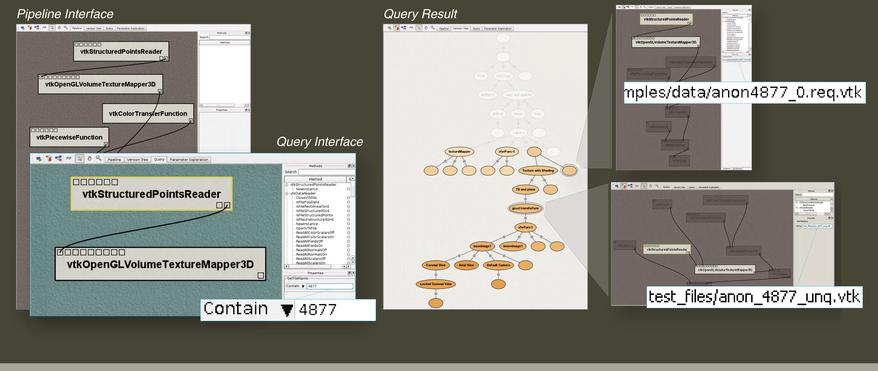
Understanding the Exploration History:

VisTrails allows users to visually compare different workflows created in an exploratory task, possibly by different users. Here, the difference is computed between two workflows that were used to derive visualizations of pathological issue in CT data of a lung. In VisTrails, the user can select nodes (workflows) in the history tree to be compared. In this case the nodes labeled "Raycasted" and "Texture Mapped" are selected, their corresponding visualizations are shown on the right, and the difference between the two workflows is shown on the bottom. Modules shown in blue are present only in "Texture Mapped", orange modules are only present in "Raycasted", dark gray modules are present in both workflows, and light-gray modules have different parameter values in the two versions, as shown on the bottom left for the "vtkColorTransferFunction" module.



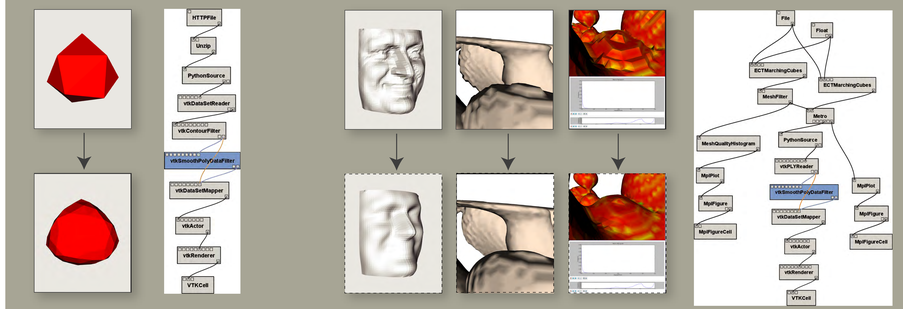
Querying Workflows and Provenance:

In addition to simple keyword-based queries, VisTrails provides the ability to query workflows by example. The interface for building a query over an ensemble of workflows is the same as the one for constructing and updating the workflows. In fact, they work together: portions of a pipeline can become query templates by directly pasting them onto the query canvas. In this example, a portion of one workflow is used to find workflows that use similar input and rendering techniques.



Workflow Creation by Analogy:

VisTrails allows the user to create new workflows from existing workflows by analogy. The user first chooses a pair of workflows to create an analogy. In this case, the pair represents a change where a file downloaded from the web is smoothed. Then, the user chooses the workflow that she desires to change in a similar fashion and applies the analogy. A new workflow is derived automatically. The workflow on the left reflects the original changes, and the one on the right reflects the changes when translated to the last workflow on the right. When data types do not match exactly, the system infers the most likely match.



THE UNIVERSITY OF UTAH

SCI www.sci.utah.edu