10-12-2009

FAnToM - Lessons Learned from Design, Implementation, Administration, and Use of a Visualization System for Over 10 Years

Alexander Wiebel

Christoph Garth

Mario Hlawitschka

Thomas Wischgoll

Wright State University - Main Campus, thomas.wischgoll@wright.edu

Gerik Scheuermann

Follow this and additional works at: http://corescholar.libraries.wright.edu/cse

Repository Citation

1 INTRODUCTION

Scientific visualization has become a central tool in many research areas since it has been established as a research discipline in 1987 [2]. Naturally, this development resulted in software tools specifically tailored for the visualization task at hand. While many such tools exist, the design choices underlying them vary greatly.

This abstract describes some aspects of the FAnToM visualization system that is being developed since 1999. Initially created to support research in topological methods for vector and tensor fields, the system quickly grew into a visualization platform for general flow visualization specialized to data represented on unstructured grids. From this origin, FAnToM derives advanced data structures for point location and interpolation over unstructured meshes, as well as fast integral curve capabilities. More recently, FAnToM has gradually been extended to serve a wider area of visualization applications, including medical and graph visualization. Throughout the development of FAnToM, close collaboration with application domain scientists has been a strong priority to facilitate the system’s usefulness on state-of-the-art problems.

The continuous development of this system over a period of ten years revealed a number of important aspects that are crucial for the usefulness of a visualization system. Furthermore, some design choices underlying FAnToM are uncommon among visualization systems in general. Here, it is our aim to discuss some aspects and design choices underlying the FAnToM system to illustrate some of its properties and differences from other visualization systems. During the discussion, we will point out some experiences and lessons learned in working with the system on modern visualization applications.

2 DESIGN CHOICES AND LESSONS LEARNED

The question of distinguishing the FAnToM visualization system from other, similar systems and tool-kits is one of the starting points of this report. In working with collaborators in the area of fluid dynamics research, we found that their data sets (obtained using CFD simulation) are usually given on large unstructured grids. Using FAnToM, they were able to quickly visualize data sets that were too large and too complicated for their commercial tools on existing hardware. The ability to handle large unstructured meshes with millions of cells was essential to them, and enabled qualitatively superior visualization and analysis. We will describe next FAnToM’s data structures and capabilities that facilitate the efficient handling of large unstructured meshes on commodity hardware.

1FAnToM – Field Analysis using Topology Methods
1% of the mesh vertices. The tree is used in a first step to identify a cell close to the interpolation point. From this cell, a ray is cast to the sought position using the cell adjacency for traversal. Special precautions are taken to reliably handle mesh holes and boundaries. The kd-tree and cell adjacency information can be built quickly when required and have a comparatively small memory footprint [1].

Overall, the up-front investment into the development of a custom, optimized data structure for unstructured meshes has had significant benefits. Besides enabling efficient application of existing visualization techniques, it has also facilitated the quick prototyping and development of novel techniques (such as e.g. [5]) that would otherwise have been prohibitively expensive to compute and develop.

2.3 Explicit Algorithm Execution

Many visualization systems are built on the premise of so-called data flow networks. These networks describe the flow of data from inputs such as data files through a sequence of filters to the output, which is typically a graphical representation of the original data. Data flow networks are very versatile, allow for parallel computation, and enable the construction of complex visualization algorithms from relatively simple component filters. Furthermore, such networks allow for caching of intermediate results in the case where only part of the filter sequence must be recomputed, e.g. in response to user feedback.

FAnToM uses a different approach that is aimed at explicit execution control by a visualization user. Using a custom plug-in architecture to support a large set of elementary algorithms, FAnToM allows for two different kinds of such algorithms: Visualization algorithms directly produce graphical representations from existing data; while data algorithms transform data sets. In this sense, visualization algorithms always present the last stage of a visualization technique, whereas data algorithms are intermediate pipeline stages. Execution or re-execution of these algorithms is explicitly controlled by the user, or implicitly performed through a scripting engine. Intermediate data sets are explicitly managed and can be written out to storage and re-loaded arbitrarily.

We have found this general philosophy of interacting with the visualization process beneficial in a number of specific settings. First, since FAnToM is aimed at treating comparatively large data sets on commodity hardware, the ability to explicitly save intermediate results has proven very valuable in performing visualization algorithms on such data. By effectively allowing the splitting of the pipeline at user-defined points, an expert user can thus carry out visualization tasks that would by far exceed the available memory in a data flow network setting. Secondly, we have found the additional flexibility of invoking arbitrary algorithms without the need to change the underlying data flow network valuable in increasing the interactivity of the visualization process. Lastly, during the development of new algorithms, this facility allows for quick sanity checks, thus reducing turn-around times and avoiding repeated re-computation of the same algorithms as required in a data flow setting.

2.4 Close Collaboration

We believe that close collaboration with application scientists has an especially important role in visualization research and for the design of visualization software. Among others, close collaboration has two implications.

Some visualization researchers first design their visualization and then search for an application that makes their method useful. This is at least not a very advisable way. Most of the time, discussions with domain scientist will allow executing research the other way around, i.e., developing new visualization techniques for existing problems. Domain scientists know best about the important problems in their domain; however, they not always know the best visualization techniques to solve their problem.

This leads us to the second implication: it is important to have new visualization techniques together with well-established ones in a visualization system. On the one hand, application domain users trust the methods they are familiar with, either because they know how the methods work (e.g. mathematically), or because they have seen the methods produce valid results often enough. On the other hand, many users distrust new methods; although, they might have many advantages, e.g. being more efficient and easier to use.

By presenting newly developed and well-established methods within the same environment, users are enabled to combine and thus compare the new and the well-established methods. As a result, it makes application domain scientists gain confidence in the correctness of the new methods. Using new and old methods together is also very sensible from a didactic point of view because the users will learn and remember more easily how to use the new methods if they applied them in a well-known context [3].

3 Results and Conclusion

Experiences from developing our visualization system over several years tell us, that if a system is aimed at more than a prototype one should invest the effort in designing a customized fast and appropriate data handling that is tailored to the special properties of the data of the system’s intended audience. Additionally, integrating common techniques into the software will greatly increase its usefulness.

Finally, it should be mentioned that FAnToM adheres to the aspects discussed in this abstract with its memory efficient data structures and algorithms and execution of the visualization pipeline. But it has some drawbacks as well, which are mainly regarding interactivity: FAnToM is quite flexible by its plug-in mechanism, but it is focused on autonomous processing algorithms that produce a visualization as output. These autonomous algorithms mostly only allow the specification of parameters in advance and do not provide means for changes after the execution of the algorithm. Elimination of this limitation will be the central part of the next large refactoring session for the FAnToM project in the future.

Acknowledgements

The authors wish to thank the many developers for their valuable work on FAnToM over the years. Special thanks go to Markus Rütten, from DLR in Göttingen for years of fruitful discussions on fluid dynamics and CFD which ultimately also helped to improve FAnToM.

References


