Issues of Dataflow and View Presentation in Multiple View Visualization

Jonathan C. Roberts
University of Kent at Canterbury,
Computing Laboratory,
Canterbury, England, UK.

Abstract *Multiple views and Multiform visualization are hot topics in the field of visualization. Multiple views represent a fan-out technology, where (for example) the same information is displayed in different forms and often in different windows. Indeed, the different views may be generated by using different parameters or various visualization algorithms. Such methods can cause a display explosion that may confuse rather than aid the user. Thus, there are important research issues in effectively controlling the multiple view generation and in exploring the information (especially linked exploration between different representations). In this paper we discuss various issues regarding multiple view visualization, indeed, we present some solutions from current research and discuss other issues yet to be resolved.*

Keywords: Visualization, Exploration, Multiple views, Dataflow, Multiform, Fan-out, Fan-in

1 Introduction and Motivation

Much of the motivation, for generating *multiple views* or multiple presentations, comes from the argument that 'different representations give the user a better understanding of the underlying information' [1]. Consider, for example, a 3D medical data set, from a CT scan. When the data is depicted by several 2D slices the user may gain an understanding of adjacencies and positions of objects within certain slices through the data, however, it may not be until a 3D surface representation is presented that the user fully understands the three-dimensional nature of the information.

Logically, the different *forms* of realization are particularly good at presenting a specific aspect of the information, these different representations shown together provide the user with a richer understanding of the underlying information. Indeed, a surface representation presents distinct boundaries in the data that makes it easy for the user to make measurements on the data; moreover, a direct volume-rendering, depicting a gel-like image, would provide an overall understanding of the whole dataset; etc. These different depictions are said to have different forms or appearances and are thus known as *multiform* representations [1].

Further, it may be easier to to *select* and *manipulate* specific objects in one representation than in another. Indeed, certain representations may make it easier to perform specific manipulation tasks. For example, a 3D display of many isosurfaces will probably include internal surfaces that are impossible to select; however, these may be more easily selected in a 2D cross-sectioned representation. Thus, it is prudent to couple these views together and allow the user to achieve the operation in the easiest possible way; creating linked multiple views.

In this paper we wish to highlight and discuss important issues, opportunities and solutions from research surrounding Multiple View visualization. There are many fundamental issues. For example, from questions of how and where the different forms are generated; to issues regarding the presentation of the information in (say) different separate windows or presented as a spreadsheet of views; to how exploration is coordinated between the views and does the user of the system determine the linkage. We classify the issues by the following fundamental aspects:

- Processing and dataflow in multiple view environments;
- Management and method of containing and presenting the forms;
- Content and Form of the multiple representations;
- Methods, models and datastructures underpinning the views;
- Coordination, Control and exploration of the views.

In this paper we focus on the first and second aspects: Processing and dataflow, and management/ presentation methods. Further information of the content and form of the multiple presentations may be found in [1, 2, 3]. Additional material about coordination and coupling views may be found in [4, 5, 6].

2 Processes and dataflow

In this section we discuss dataflow and process issues when using multiple views. First, let us look at dataflow and fan-out techniques in visualization.

2.1 Issue: fan-out or fan-in

The issue here is 'does the information, from (say) a new parameterization, get presented in a new window or get overlayed to an existing one'?

Traditionally visualizations are generated using a visualization-system that follows the *dataflow* paradigm. This model is based on the visualization process of Haber and McNabb [7](Figure 1). Here the user 'plugs' a series of modules together to process the data into an appropriate visualization. The data is filtered, to select interesting 'features' of information, then this information is mapped into a geometric or abstract form, known as an Abstract Visualization Object (AVO), which is rendered into an image.

The user explores the information by changing various parameters at different stages of the dataflow. Hereby updating the information in the down-stream modules. Often, a single visualization window is used and so a change to any parameter will cause the downstream modules to appropriately update and a new representation replaces the old. This is known as the *Replacement* strategy [1].

This is good way of working as the user can *try out* different scenarios and see the results in the window. The user instantly realizes where the new visualization will appear. However, it is hard for the user to *roll back* to a previous scenario as previous parameterizations are not stored. Some visualization systems do overcome the ephemeral nature of the parameterization by storing previous parameter values; such as used in GRASPARC [8](also HyperScribe [9]) and Tioga [10], for example.

The *fan-out* method allows the output from any module of the dataflow pipeline to be split into two (or more outputs). For example, Figure 2 shows a schematic of several fan-out instances. Figure 2A is our reference result, B is different from A by a change in the filtering parameters; part C and D share the same Abstract Visualization Object (AVO) but are rendered using a different renderer or projection technique.

The *fan-in* method allows the information to be merged back into one representation. For example, a view that depicts both 2D cutting planes through 3D information and an isosurface. Here the information from different processes are merged or *Overlayed* in the same display.

Further research is required to appropriately manage fan-in and fan-out strategies, and

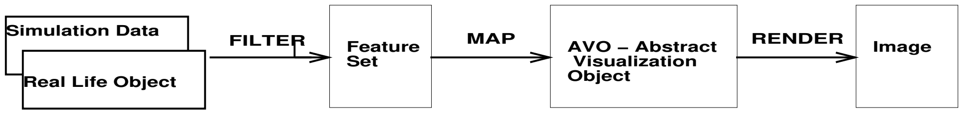

Figure 1: Dataflow model. The data is *filtered*, to create a subset, which is then *mapped* into a representation which can be *displayed*

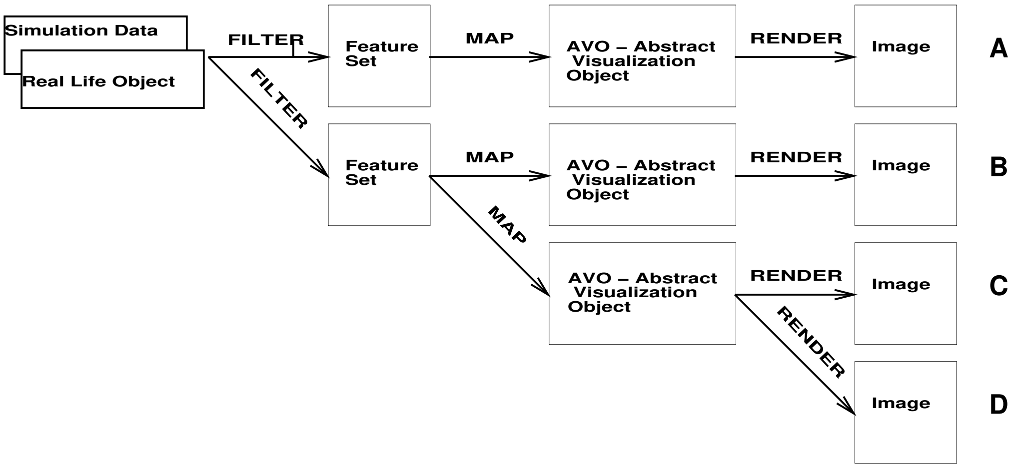

Figure 2: Dataflow model with several fan-out instances.

questions remain of when is it effective to operate a replacement, fan-out (replication) or fan-in (merging) strategy?

2.2 Issue: fan-out replication, when to replicate?

The issue here is when to allow fan-out replication of modules and views. When the user changes a parameter value does a new representation automatically appear? Or is it when they select 'ok' or are many representations initiated? Or do many (possibly infinite) representations appear as the user changes a continues parameter, from (say) a slider?

In practice it is probably best if the user requests the new view to happen at their command, otherwise they may get annoyed of the situation that when they 'touch' a parameter a new view pops-up. Moreover, the user may request a replacement strategy to present the updated information in an already present view; but, as we shall see there are issues here of what to then update in following modules.

2.3 Issue: controlling the view explosion

The system may make it easy to generate multiple views, thus, there may be a view explosion. This can cause an 'information overload', where there is so much information displayed in so many windows that the user cannot understand how each relates and what is really being displayed. Indeed, in any computer based window system there is this problem of controlling and managing the use of multiple windows, this is discussed further in section 3.1.

Indeed, if a new window appears when the user changes (or even touches a parameter) especially if the parameter is a continues quantity then the amount of views displayed could be infinite. A question remains, is this view explosion useful? Consider the direct manipulation situation; the user may wish to directly change a value that automatically updates (by replication) a single view with the new information; e.g. when the user changes a colour bar the result automatically gets updated. Now, it may be useful to display the result of the parameter change in 'many' windows, where 'many' is a finite number, such as by depicting this change by some key 'frames' of information [11].

2.4 Issue: hierarchical exploration and information refinement

Hierarchical exploration seems a good way of exploring information. The idea is to first generate an abstract representation (or a depiction of the whole dataset) and then refine it into a less abstract (or more specialised) representation. Such a method is used in the Waltz visualization system [12]. The specialization may occur by *selecting* a subset of information in one view. This information is then depicted in another window. It may be useful to depict this subsequent subset of information at a higher resolution, if available. This method implicitly uses multiple views.

The issues here, include, how does the user understand the relationship between the subset, the original in the exploration; and how is the information updated. For example, if the user chooses to replace the data in an upstream module, does the information in subsequent, and dependent modules, get automatically updated? In Waltz, the user controls if the information in subsequent (dependent) views is updated. Moreover, there may be problems of the dependent view not having any data to depict; for example, when an upstream module filters out the information that was previously visualized in a down stream module.

2.5 Issue: push or pull dataflow

The issue here is how does the information in the views get updated through a push or pull model of operation. Particularly, how does this update occur in a hierarchical exploration model?

In a dataflow model there are two well known methods of controlling the information flow; First, is the eager update model that updates the downstream modules whenever a parameter is changed upstream. Second, is a lazy style of operation that updates the information in upstream modules when the downstream modules (including the view module) require updated information.

In a multiple view system, the push method may update views that the user did not wish to have changed, conversely, the pull method may not update the modules when the user expects an update.

2.6 Issue: replication means more work for the user

The module building environments such as AVS [13], IRIS Explorer [14], and IBM Data Explorer [15] are extremely expandable and diverse. However they require a lot of user control to replicate parts of the module flow. Many modules need to be copied, module connections made and parameters need to be changed.

Effective ways of replication modules need to be developed. The render Group method is one such method. For example, Yagel et al [3] group four volume renderers to generate low quality to high quality images of the same data [3]. Roberts [16] says "The render-groups provide a convenient container for the multiple views. Here, consistency between views and the close coupling of views may be easily maintained, additional views of the same information may be easily requested and added to the render-group and automatically coupled to existing displays".

However, further methods of batching together of filter, map and display parameters should be investigated.

3 The structure of the presentation

In any window based systems there is always a problem of displaying too many windows onto one screen; the 'real-estate' of the screen is not large enough!

3.1 Issue: effective screen management

There are many effective ways of controlling this management [17]. Some well used techniques include:

Iconification where the windows are temporarily closed and their existence is depicted by a place-holder that takes the form of a name or picture (icon) [17].

Cascading where the windows are laid on top of each other with a some small portion of the underlying window showing through, [17].

Tiling where the windows are placed adjacently without any overlap, [17]. This is similar to the tabular, elastic views [18] of Kandogan and Shneiderman, and spreadsheet [19] style of presentation.

Scaling where some windows are scaled smaller. Usually, the scaled windows still depict the structure of the contained information but at a lower resolution; such methods are used in Pad++ [20].

There remains many questions. One important question to research is: are the requirements of visualization exploration systems that use multiple views significantly different from other multiple view systems, such as window managers?

3.2 Issue: understanding window, parameter and session relationship

The multiple windows act as an exploration history. The user can see the different experiments and explorations they have made. Some systems, as discussed in section 2.1 allow the user to roll-back, examine and change previous experimentations. But, with all the windows on the screen how does the user know the relationship between the different windows and the content contained within.

Some sort of labelling is one solution. For example, each window may be labelled with the corresponding name being shown in a diagram of the session. For example, Waltz [12] uses such a scheme. Here the naming scheme is similar to section numbering; so each refinement is labelled with a new number, such as "1" with subsequent sub-refinements named as "1.1", a new refinement would be "1,2" etc. As shown in Figure 3. But, further techniques to manage this window relationship, especially within multiform visualization, should be investigated and evaluated.

4 Conclusion

We have discussed many of the issues surrounding dataflow, process and presentation for multiple view visualization systems. However, there are still a lot more issues to discuss. There is much research still to be achieved in this area of multiple and multiform visualization. Indeed, Baldonado et al [21] state "multiple view systems are highly challenging to design. They often use sophisticated coordination mechanisms and layout". There is much work to be done!

References

[1] Jonathan C. Roberts. Multiple-view and multiform visualization. In Robert F. Erbacher, Philip Chen, Jonathan C. Roberts, and Craig Wittenbrink, editors, *Visual Data Exploration and Analysis VII, Proceedings of SPIE*, volume 3960. IS&T and SPIE, January 2000.

[2] Eduard Grller Andreas H. Knig, Helmut Doleisch. Multiple views and magic mirrors – fmri visualization of the human brain. http://www.cg.tuwien.ac.at/ research/vis/vismed/MM/.

[3] Roni Yagel, David S. Ebert, James N. Scott, and Yair Kurzion. Grouping volume renderers for enhanced visualization in computational fluid dynamics. *IEEE Transactions on Visualization and Computer Graphics*, 1(2):117–132, June 1995.

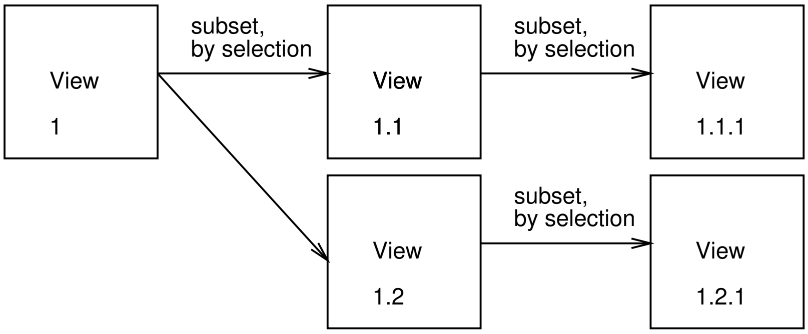

Figure 3: Refinement numbering scheme, after Waltz. The naming scheme is similar to section numbering, with subsequent sub-refinements named as integers following a full-stop.

[4] Jonathan C. Roberts. On Encouraging Coupled Views for Visualization Exploration. In Robert F. Erbacher, Philip Chen, and Craig Wittenbrink, editors, *Visual Data Exploration and Analysis VI, Proceedings of SPIE*, volume 3643, pages 14–24. IS&T and SPIE, January 1999.

[5] Chris North. Robust, end-user programmable, multiple-window coordination. In *Proceedings of ACM CHI 98 Conference on Human Factors in Computing Systems (Summary)*, volume 2 of *Doctoral Consortium*, pages 60–61, 1998. Student Posters: Interaction Techniques.

[6] Jurriaan D. Mulder and Jarke J. van Wijk. Parametrizable cameras for 3D computational steering. In W. Lefer and M. Grave, editors, *Visualization in Scientific Computing '97*, Eurographics, pages 165–176. Springer-Verlag Wien New York, 1997.

[7] R. B. Haber and D. A. McNabb. Visualization idioms: A conceptual model for scientific visualization systems. In B. Shriver, G. M. Nielson, and L. J. Rosenblum, editors, *Visualization in Scientific Computing*, pages 74–93. IEEE Computer Society Press, 1990.

[8] Ken Brodlie, Andrew Poon, Helen Wright, Lesley Brankin, Greg Banecki, and Alan Gay. GRASPARC – a problem solving environment integrating computation and visualization. In *Proceedings Visualization '93*, pages 102–109. IEEE Computer Society Press, 1993.

[9] Helen Wright. The HyperScribe Data Management Facility. Technical Report TR1 96, NAG, 1996. www.nag.co.uk /doc/techrep/ps/tr1_96.ps.

[10] Michael Stonebraker, Jolly Chen, Nobuko Nathan, Caroline Paxon, Alan Su, and Jiang Wu. Tioga: A database-oriented visualization tool. In *Proceedings Visualization '93*, pages 86–93. IEEE Computer Society Press, 1993.

[11] Jonathan C. Roberts and Andy King. A strategy for controlling view explosion in multiple views. Technical report, University of Kent at Canterbury, Computing Laboratory, March 2001. (Private Communication).

[12] Jonathan C. Roberts. Waltz: an exploratory visualization tool for volume data using multiform abstract displays. *Visual Data Exploration and Analysis V, Proceedings of SPIE Vol. 3298*, pages 112–122, January 1998.

[13] C. Upson, T. Faulhaber, D. Kamins, D. Schlegel, D. Laidlaw, F. Vroom, R. Gurwitz, and A. van Dam. The application visualization system: A computational environment for scientific visualization. *IEEE Computer Graphics and Applications*, 9(4):30–42, 1989.

[14] NAG, Numerical Algorithms Group. *IRIS Explorer Users' Guide*, October 1999. www.nag.co.uk/visual/ie/iecbb/doc/.

[15] Bruce Lucas, Gregory D. Abram, Nancy S. Collins, David A. Epstein, Donna L. Gresh, and Kevin P. McAuliffe. An architecture for a scientific visualization system. In *Proceedings Visualization '92*, pages 107–114. IEEE Computer Society Press, 1992.

[16] Jonathan C. Roberts. On Encouraging Multiple Views for Visualization. In Ebad Banissi, Farzad Khosrowshahi, and Muhammad Sarfraz, editors, *Information Visualization IV'98*, pages 8–14. IEEE Computer Society, 29-31 July 1998.

[17] Jenny Preece, Yvonne Rogers, Helen Sharp, David Nebyon, Simon Holland, and Tom Carey. *Human-Computer Interaction*. Addison-Wesley, 1996.

[18] Eser Kandogan and Ben Shneiderman. Elastic windows: Evaluation of multi-window operations. In *CHI*, pages 250–257, 1997.

[19] Ed Huai-hsin Chi, John Riedl, Phillip Barry, and Joseph Konstan. Principles for information visualization spreadsheets. *IEEE Computer Graphics & Applications*, 18(4):30–38, 1998.

[20] Ken Perlin and David Fox. Pad: An alternative approach to the computer interface. In James T. Kajiya, editor, *Computer Graphics (SIGGRAPH '93 Proceedings)*, volume 27, pages 57–64, August 1993.

[21] Michelle Q. Wang Baldonado, Allison Woodruff, and Allan Kuchinsky. Guidelines for using multiple views in information visualization. In *Advanced Visual Interfaces*, pages 110–119, 2000.