13 Ways to Say Nothing with Scientific Visualization

Al Globus, Computer Sciences Corporation[1]
E. Raible, NASA Ames Research Center

Report RNR-92-006
25 Feb. 92

Abstract

Scientific visualization can be used to produce very beautiful images. Frequently users and others not properly initiated into the mysteries of visualization research fail to appreciate the artistic qualities of these images. Scientists will frequently use our work to needlessly understand the data from which it is derived. This paper describes a number of effective techniques to confound such pernicious activity.

Copies of this report are available from:

NAS Applied Research Branch
Mail Stop T045-1
NASA Ames Research Center
Moffett Field, CA 94035
(415) 604-4332

1. This work is supported through NASA contract NAS2-12961.

13 Ways to Say Nothing with Scientific Visualization

A. Globus, Computer Sciences Corporation[1]
E. Raible, NASA Ames Research Center

Scientific visualization can be used to produce beautiful images. Users and others not properly initiated into the mysteries of visualization research fail to appreciate the artistic qualities of these images. Scientists will frequently use our work to needlessly understand the data from which it is derived. This paper describes a number of effective techniques to confound such pernicious activity.

Introduction

Upon reading D. Bailey's seminal work, "Twelve Ways to Fool the Masses When Giving Performance Results on Parallel Computers" [DHB91], the authors were struck by the brilliant simplicity of the concept. Bailey ends with the admonition "... conclude your technical presentation and roll the videotape. Audiences love razzle-dazzle color graphics, and this material often helps deflect attention from the substantive technical issues." Unfortunately, Bailey gives no guidance in the means and methods to produce the intended result. The present article humbly seeks to fill this void.

There are a number of time-tested scientific visualization techniques for producing pretty pictures while avoiding unnecessary illumination of the data. Our collection has been culled from the scientific visualization literature and numerous presentations the authors have given and attended.

1. Never Include a Color Legend

Many visualization techniques involve assigning colors to scalar data values. In lesser sciences, a legend relating colors to values is provided. In our exalted art form not only does a legend mar the beauty of an image, but the viewer may be diverted into idle contemplation of reality.

Note: images can be particularly enthralling if the sequence of colors is chosen solely on aesthetic grounds. For optimal results, quietly use separate color mappings for different parts of the image.

2. Avoid Annotation

In dreary old fashioned sciences like physics and biology, investigators have been known to annotate their images with arrows pointing out features of supposed interest along with explanatory text. This promotes clarity of understanding, undermining the sense of awe and confusion the best scientific visualization engenders.

3. Never Mention Error Characteristics

If scientists using visualization software were aware that visualization software might introduce error, they might not be properly impressed by our masterworks. Therefore, never imply by word or deed that your algorithm introduces any numerical or algorithmic error whatsoever. After all, if the picture looks good it must be correct.

1. This work is supported through NASA contract NAS2-12961.

4. When in Doubt, Smooth

Always strive for the smoothest possible surfaces since they look so much better than numerous ugly facets. For example, choose lighting normals to hide sharp edges in the data. Smoothing can also obscure errors and allow users to publish their results earlier. Both beauty and speed of publication are more important than mere accuracy.

5. Avoid Providing Performance Data

When you are presenting a pretty picture, some stick-in-the-mud may ask how long it took to calculate. The fact that your ray-cast isosurface took hours of massively parallel supercomputer time to generate when nearly identical results could be achieved using marching cubes [LC87] in seconds on a workstation is irrelevant. In addition to being smoother (see rule 4), a ray cast image can include some wispy stuff scattered around to give the image an ethereal quality.

6. Quietly Use Stop-Frame Video Techniques

Each frame of a scientific video usually takes seconds, minutes, or even hours to produce. To achieve smooth animation it is usually necessary to generate video frames one at a time and transfer each separately to tape. They can then be played back at 30 or even 60 frames a second. Stop-frame techniques can dramatically improve perceived software performance. The magic is lost, however, if you are so foolish as to tell anyone what you're doing.

Faithful adherence to the rest of the rules will help avoid tedious debugging of software that already produces pretty pictures.

7. Never Learn Anything About the Data or Scientific Discipline

Debugging scientific visualization software is much more difficult if you are worried about producing correct results. Irritating details like accurate interpolation techniques get in the way; in many cases ad-hoc interpolation techniques can produce much prettier pictures with significantly less work. Better yet, programming bugs can sometimes produce stunning images. If you don't know what to expect, you won't have to find and fix such bugs. Besides, beauty is the higher truth.

8. Never Compare Your Results with Other Visualization Techniques

Comparison of results with other visualization techniques is fraught with danger. You may detect bugs in your code that will need to be fixed, a tedious chore. Much worse, other techniques may produce prettier pictures.

9. Avoid Visualization Systems

Visualization systems, such as FAST [BAN90] and AVS [UPS89], provide mechanisms to add modules implementing new visualization techniques. There are two problem with these systems. First, users may violate rule 8 to your discomfort. Second, visualization systems are usually Not Invented Here.

10. Never Cite References for the Data

If you cite a reference describing the data used to generate images, someone may read the paper and discover that your visualization bears no relationship to the key elements the original experiment was meant to elucidate. This will detract from your picture's appeal and should be avoided.

11. Claim Generality but Show Results from a Single Data Set

It can be difficult to write visualization algorithms that function properly on a variety of data. Much effort may be saved by running your software on one (small) data set and using viewing angle and color map manipulations to make the images look different. Follow rule 10 so that no one will know what you're doing.

12. Use Viewing Angle to Hide Blemishes

Many otherwise excellent algorithms produce 3D objects containing unsightly blemishes. Avoid carelessly choosing viewing angles that expose such flaws. If a suitable angle cannot be found, try another data set.

13. 'This is easily extended to 3-D'

3-D algorithms are almost always much more difficult than 2-D. Again, the effort of generalizing a promising 2-D algorithm to 3-D can detract from producing pretty pictures. To both impress your colleagues and avoid much tedious work, simply claim that your algorithm 'is easily extended to three or more dimensions.' Only the real pros will know you are lying, but they won't challenge you since we all make identical claims.

Conclusion

As Dr. Bailey pointed out in [DHB91], "... it is often necessary for us to adopt some advanced techniques in order to deflect attention from possibly unfavorable facts." This paper details a set of techniques to divert attention away from data and towards beauty. Follow these rules faithfully and you never need to sully your pretty pictures with the grubby realities of science. May your images be accepted by SIGGRAPH.

Acknowledgments

The authors wish to acknowledge helpful contributions and comments by D. Bailey, D. Asimov, M. Gerald-Yamasaki, C. Levit, and S. Uselton.

References

[BAN90] G. Bancroft, F. Merritt, T. Plessel, P. Kelaita, R. McCabe, A. Globus, "FAST: A Multi-Processing Environment for Visualization of CFD," *Proc.Visualization '90*, IEEE Computer Society, San Francisco (1990).

[DHB91] D. Bailey, "Twelve Ways to Fool the Masses When Giving Performance Results on Parallel Computers," Supercomputer Review, Aug. 1991, pp. 54-55.

[LC87] W. E. Lorenson and H. E. Cline, "Marching Cubes: a High Resolution 3d Surface Construction Algorithm," *Computer Graphics*, Vol. 21, No 4, July 1987, pp 163-169.

[UPS89] C. Upson, T. Faulhaber, D. Kamins, D. Laidlaw, D. Schlegel, J. Vroom, R. Gurwitz, A. van Dam, "The Application Visualization System: A Computational Environment for Scientific Visualization," *IEEE Computer Graphics and Applications*, July 1989, pp. 30-41.