Visual Analysis
Adding breadth to a computer graphics course

Andrew Sears
Rosalee Wolfe
DePaul University
Chicago, IL 60604
sears@cs.depaul.edu

Abstract

Visual Analysis adds breadth to a computer graphics course by teaching students to compare and contrast the visual effects of rendering algorithms. Using slides to teach visual analysis minimizes the amount of required lecture time, and the interactive tool TERA facilitates further study outside of class. Visual analysis enhances depth as well as breadth of knowledge, because students become familiar with an algorithm's visual effect before they implement it.

Breadth in Computer Graphics

Breadth of coverage in computer graphics includes comparison of visual effects in addition to cost analysis of rendering algorithms. Breadth better prepares students to tackle such real-life questions as "What is the cheapest algorithm that can produce this effect?", or "Algorithm X is prohibitively expensive – Can we get the same effect by tweaking our implementation of algorithm Y?" This is analogous to the treatment of sorting algorithms [Tra93].

The biggest problem in adding breadth is finding the time in a class schedule that is already crowded due the displine's enormous expansion. Students still need a solid understanding of such underlying principles as coordinate transformation and clipping, and they do benefit from the experience of implementing a few carefully-selected rendering algorithms [Cle91], [Owe91], [Sch91]. Adding breadth cannot detract from the time students spend on acquiring programming experience as this skill is valued by potential employers [Bro89], [Sch93].

SIGCSE '95 3/95 Nashville, TN USA

Visual Analysis

Visual analysis adds breadth to an introductory computer graphics course. Visual analysis teaches students to identify and compare the effects of a variety of rendering algorithms. The technique stems from critical analysis, which establishes a structure for examining works of art [Arn74]. Students in the visual arts learn to describe and compare works in terms of design, concept, and media.

Instead of design, concept and media, computer graphics students learn to recognize:

- visibility of polygon faces or polygon edges
- transition from light to dark on diffuse-reflecting objects
- color and shape of specular highlights
- presence of transparency, reflection, refraction, patterns, and textures
- sharpness of shadow
- interactions between adjacent diffuse reflectors (color bleeding).

These features are usually sufficient to identify a rendering algorithm. See table one for a list of identifying features, and the corresponding algorithms. By learning to observe and describe these features, students are able to identify the interactions of rendering algorithms and objects. As their skills grow, students learn to suggest algorithms appropriate for a given object.

In the first lecture, the teacher discusses three or four algorithms that produce starkly different effects. Each week, the teacher adds more algorithms, and by midterm, the class has examined all commonly-used rendering algorithms.

Until the midterm examination, the teacher shows images that demonstrate an algorithm's characteristic

effects. After the midterm, the teacher shows how some effects can be achieved by multiple algorithms.

Classroom Presentation

During the last 15 minutes of each 3-hour class, the teacher shows two series of slides that demonstrate the effects of rendering algorithms. While showing the first series, the teacher points out the salient features of each visual effect, and names the algorithm that created the effect. While showing the second series, the teacher invites students to describe the appearance of an object in the image. After a student describes the visual effect, the teacher solicits suggestions for a probable algorithm. It is important that description proceed identification.

We use slides to demonstrate and discuss visual effects of rendering algorithms because no other technology currently provides the same high quality image that is visible to an entire class. A class of 25 cannot all see a workstation monitor at the same time. The color fidelity and resolution of projection televisions and four-color printed images is not as good as that of slides. [Fol90].

Study Materials

To encourage students to practice visual analysis outside the class, the teacher originally provided slides on reserve in the library and on-line images. Students said they benefited more from the question-and-answer sessions held at the end of each lecture. Images alone did not help them to practice.

Having students learn rendering packages helps somewhat, but it is time consuming for a student to learn the packages. There are a number of excellent pedagogical packages [Owe93], [Sch92] that facilitate study of individual graphics algorithms, but they are not geared for comparative study.

Recently we developed TERA, a Tool for Exploring Rendering Algorithms [Wol95]. TERA is an interactive program that facilitates visual analysis. A student can choose a scene, and specify the algorithm for any object in the scene, which allows them to compare the effects of different algorithms. Students can practic visual analysis using TERA. Students select a scene and TERA presents it with each object rendered by a random algorithm. Students then guess the rendering algorithm for each object in the scene. TERA responds with "Correct", "Try Again" or "Close Enough." The "Close Enough" response is for those cases when multiple algorithms produce similar visual effects.

Whether exploring the effects of algorithms, or quizzing themselves, students can always select the "Tell Me More" button. The "Tell Me More" button provides detailed feedback about their last selection.

Figure one shows a screen snapshot during a session with TERA. The current version of TERA was evaluated by nine students who had just finished our introductory graphics course. Each student used TERA for five minutes with no verbal instruction. On average, students took 32 seconds to begin useful interactions with TERA. Students liked using TERA. A typical response was "Why didn't we have this before the midterm?"

Results

Students demonstrated that they could successfully identify a wide variety of algorithms by performing well on the visual-analysis portion of the midterm. The first class scored on average of 87%, and the second class scored approximately 85%.

In addition to adding breadth, visual analysis has enhanced depth of understanding. The last homework is to implement a Gouraud shader. After introducing visual analysis, the question, "Is it right?" has disappeared entirely, and has been replaced by questions similar to "My highlights aren't right – what's wrong with my specular light?" By the time they implement the algorithm, they have expectations of how a Gouraud-shaded object should appear.

Conclusions and Future Plans

Visual analysis adds breadth to an introductory computer graphics course by teaching students to compare and contrast the visual effects of a wide variety of rendering algorithms. Visual analysis also enhances depth of knowledge of those rendering algorithms the student does implement, because the student is familiar with the visual effects of the algorithms. Use of slides minimizes the amount of lecture time required to teach visual analysis, and we have developed an interactive tool TERA that facilitates comparative study among algorithms.

Future plans are to port TERA from its current Unix/X environment to Microsoft Windows. Although students have used TERA in a previous course offering, it was not available for the entire length of the course. The next time we offer the course, each student will receive a copy of TERA. We intend to a conduct additional experiments to test TERA's effectiveness in developing a student's visual analysis skills.

References

[Arn74] R. Arnheim. *Art and Visual Perception: A Psychology of the Creative Eye.* 2nd ed. Berkeley: University of California Press, 1974.

[Bro89] J. Brown, S. Cunningham, L. Hala, C. Keith, and R. Keith. "Computer Graphics Career Information Handbook" *Computer Graphics* **23**(1) February, 1989.

[Cle91] J. Clevenger, R. Chaddock, and R. Bending. "TUGS - A Tool for Teaching Computer Graphics." *Computer Graphics* **25**(3) July 1991, 158–164.

[Fol90] J. Foley, A. van Dam, S. Feiner, and J. Hughes. *Computer Graphics; Principles and Practice.* 2nd. ed. Reading, Massachusetts: Addison-Wesley, 1990. p. 180, 599–600.

[Owe93] G. S. Owen, "Using Hypermedia to Teach Computer Graphics." Presented at the Graphics and Visualization Education Workshop at *Eurographics 93.* September 1993.

[Owe91] G. S. Owen, "Considerations in Teaching a Two Quarter Computer Graphics Sequence." *Computer Graphics,* **25**(3) July 1991. 147–150.

[Sch93] N. Schaller. "Graphics Education for Computer Science - Panel Report." *Computer Graphics* **27**(3) January 1993, 6–10.

[Sch92] D. Schweitzer. "Designing Interactive Visualization Tools for the Graphics Classroom." *SIGCSE Bulletin* **24**(1) March 1992, 299–303.

[Sch91] D. Schweitzer and L. Northrop. "Getting to the "Graphics" in a Graphics Exercise." *Computer Graphics* **25**(3) July 1991, 151–157.

[Tra93] R. Trahan and S. Rodger, "Simulation and Visualization Tools for Teaching Parallel Merge Sort." *SIGCSE Bulletin* **25**(1) March 1993, 237–241.

[Wol95] R. Wolfe and A. Sears, "TERA: an interactive Tool for Exploring Rendering Algorithms." *Journal of Computing in Small Colleges* **10**(4) February 1995, 41–46.

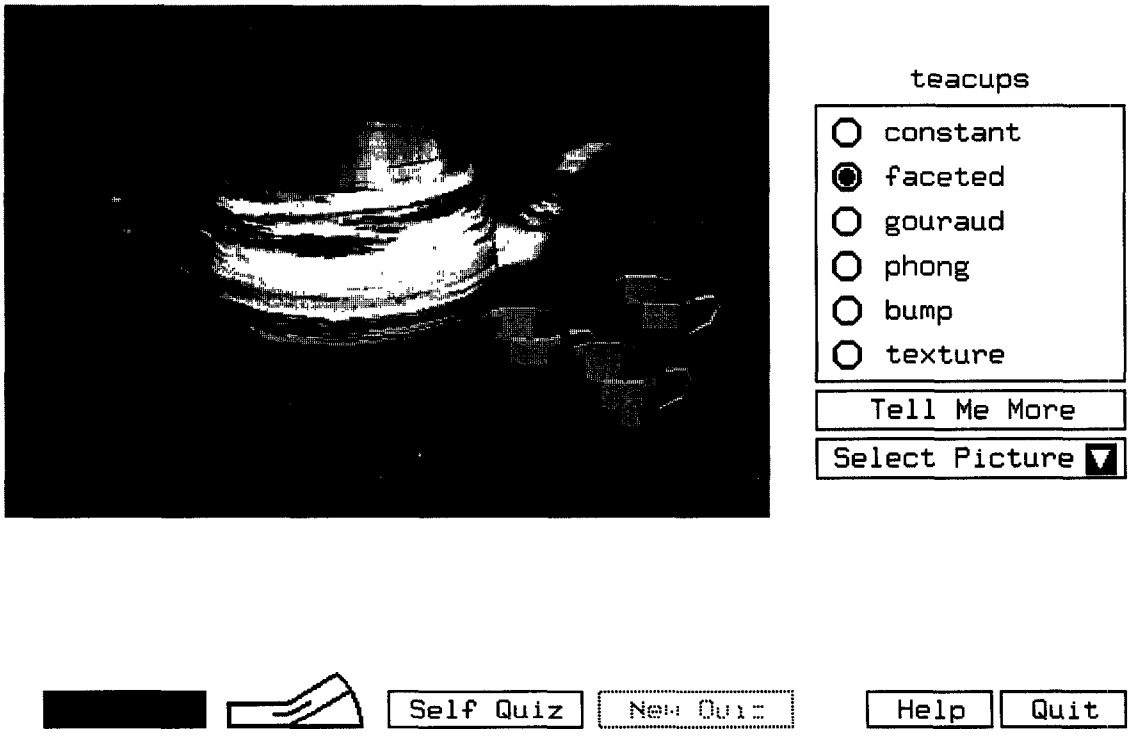

Figure 1: TERA in use

Visual Effect	Algorithm
Outlined polygons. Object's far side is visible. Objects do not occlude each other.	wireframe
Outlined polygons. An object's far side is not visible. Objects occlude one another.	hidden line removal
Opaque objects. An object appears flat, as if cut from paper.	hidden-surface removal, constant shading
Opaque objects. Each polygon has a single color, making each polygon clearly visible.	faceted shading
Opaque objects. Although the object may have a jagged profile, its shading is smooth. Specular highlights follow polygon edges.	Gouraud shading
Objects are opaque. Object is smoothly shaded, although its profile may be jagged. Highlights are white and elliptical. "Plastic look".	Phong shading
Object has a single 'base' color, but the surface appears rough or wrinkled.	bump mapping
Object appears to be made of a material like marble or wood, or appears to have a decal pasted on it.	texture mapping
Transparent object. Other opaque objects are visible through the glass object.	If the glass material appears to bend the light, hidden-surface algorithm is raytracing.
Reflective object	Probably raytracing. Check for interreflections between objects to confirm.
Color bleeding, soft shadows	radiosity

Table 1: Visual Effects of Rendering Algorithms