Available online at www.sciencedirect.com

SCIENCE DIRECT

Computers & Graphics 29 (2005) 257–265

COMPUTERS & GRAPHICS

www.elsevier.com/locate/cag

Using photography as a metaphor for teaching image synthesis

Joe Geigel*, Nan C. Schaller

Computer Science Department, Rochester Institute of Technology, 102 Lomb Memorial Drive, Rochester, NY 14623 5608, USA

Abstract

The camera has long served as a metaphor for teaching three-dimensional graphics in introductory computer graphics courses. We extend this metaphor to include the complete photographic pipeline as a framework for teaching image synthesis in a second graphics course. We present the correspondence between photographic processes and areas of study in image synthesis, and discuss the success of using this framework in an image synthesis course at our university for the past 3 years.
© 2005 Elsevier Ltd. All rights reserved.

Keywords: Computer graphics education; Image synthesis; Raytracing; RenderMan photography

1. Introduction

The Computer Science Department at Rochester Institute of Technology has offered two computer graphics courses since the early 1980s. These courses are co-listed for undergraduates and graduates. Computer Graphics I introduces two-dimensional and three-dimensional graphics including theory, algorithms and the use of an API, currently OpenGL. Since its inception, Computer Graphics II has been a projects course, using a collaborative learning approach. Students would choose a term project, often working on a team. Each student would research a topic pertinent to their project, write a paper about their findings and present them to the class. While students were happy with this approach, they felt that they needed another, more focused, theoretical course between Computer Graphics I and Computer Graphics II. Three years ago, Computer Graphics II was modified to add a significant lecture component and to specifically focus on the image synthesis pipeline and rendering.

The camera has long served as a metaphor for teaching three-dimensional graphics in introductory courses. In the modification of our second course, we extend this metaphor to use the complete photographic pipeline as a framework for teaching image synthesis. In the following sections, we lay out the correspondence between photographic processes and areas of study in image synthesis, and discuss the success of using this framework in this course at our university for the past 3 years.

2. Computer graphics as virtual photography

Images are a two-dimensional projection of a three-dimensional world. This is true for computer graphics as well as for photography. As the goals of image synthesis are similar to those of photography, computer graphics has borrowed extensively from photography in developing its rendering pipeline.

Photography begins with a physical scene. A camera is used to capture and focus light from the scene and to

*Corresponding author.
E-mail addresses: jmg@cs.rit.edu (J. Geigel), ncs@cs.rit.edu (N.C. Schaller).

0097-8493/$ - see front matter © 2005 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cag.2004.12.012

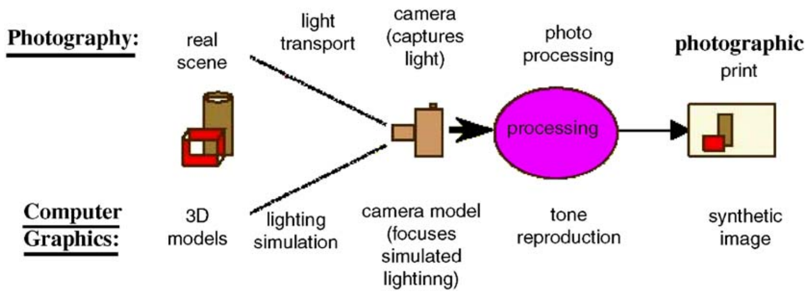

Fig. 1. The photographic and image synthesis pipelines.

project it onto film. The film is then processed, producing a print. The structure of the pipeline is essentially the same in computer graphics. However, in computer graphics the scene is virtual, and both the lighting and the processing are simulated. Fig. 1 illustrates the correspondence of between the two pipelines.

Many of the fundamental algorithms in image synthesis involve the simulation of the transport of light within a virtual scene. Even early graphics techniques, i.e., those developed before global illumination, rely on a photographic metaphor. Students are exposed to the graphics rendering pipeline and the camera metaphor in Computer Graphics I. As most students are somewhat familiar with rudimentary photography, we feel that building on this knowledge by teaching Computer Graphics II using a photographic framework is a natural way to organize and convey the additional complexities of complete image synthesis.

3. Course content

An extensive web search has led us to conclude that, although there is an overlap in the topics we cover and those in image synthesis courses taught at other universities, our organization and presentation with the analogy to the complete photographic pipeline is unique. This is true as well with respect to our emphasis on tone reproduction. As such, the course takes the student beyond the traditional computer graphics treatment of light and color. Many Computer Graphics APIs and toolkits represent color using 8-bit integers or floating point values that range between 0 and 1. Instead we emphasize that light and color can be measured using a well-defined set of scientifically established standards and units that do not conform to this treatment. One of our goals is that students gain an appreciation of physical light units and the potential for high dynamic range imagery, as well as be able to recognize the limitations imposed by the traditional computer graphics representation of color and light intensity.

The course content is divided into four units, each corresponding to an area of the pipeline. These units, along with the topics covered in them, are summarized below.

Unit 1—setting the scene: This unit focuses on object modeling and transformations. Some of the topics in this section are review from Computer Graphics I. They not only serve as a refresher for those who have already taken the first course, but also as a means to ensure that those who have entered the course through a non-traditional path have a similar knowledge base. Topics in this unit include object modeling and object transformations. On the modeling end, students who have previously dealt with polygonal models and geometric primitives are introduced to polygonal meshes, parametric surfaces using splines and NURBs, and quadratic surfaces, such as surfaces of revolution. Hierarchical modeling is encouraged and students are introduced to other modeling approaches such as constructive solid geometry and procedural models, including hypertextures [1] and fractals.

Unit 2—the camera: Most students will have been exposed to the pinhole camera model as well as to perspective and orthographic projections. This unit introduces more advanced camera models, e.g., Kolb's [2], developed from photographic science. Featured camera models include those with one or more lenses and aperture control. Visual effects, such as focus, depth of field, and motion blur, that are made possible by using these extended models, are discussed. This unit also emphasizes the point sampling nature of the image synthesis process, the aliasing problem inherent in this method, as well as approaches to solving that problem.

Unit 3—illumination: Photographs are the result of focusing light on film. In computer graphics, images are produced through the *simulation* of light transport. In this unit, we explore methods by which this simulation is performed. Topics include the physics of light, the basics of color science, shading models, and illumination models. The distinction between shading and illumination is emphasized. The shading discussion reintroduces the three traditional shading models, i.e., flat, Gouraud, and Phong, and focuses on types of texture mapping and procedural shading. The illumination discussion begins with an introduction of bi-directional reflection distribution functions (BRDFs). This is followed by a discussion of several local illumination models: Lambertian, Phong, Cook-Torrance [3], and Ward's anisotropic model [4], and how these approximate general BRDFs. Kajiya's rendering equation [5] is introduced as a frame work for discussing traditional global illumination algorithms, including ray tracing, radiosity, distributed ray tracing, and photon mapping. Advanced methods such as the two pass global illumination method [6] and REYES [7] are introduced as well.

Unit 4–tone reproduction: The dynamic range of a scene can far exceed the dynamic range of the device that will display its image; thus, it is essential to address the issue of tone reproduction. Indeed, tone reproduction is fundamental to photography, being a driving force in its development. In computer graphics, it has been given increasing attention over the past decade [8]. Keeping with the photographic analogy and our emphasis on physical lighting units and measurements, tone reproduction must be treated as an integral part of the image synthesis pipeline, rather than just as another advanced topic in computer graphics. Topics in this unit include human perception, color management, photographic tone reproduction, and tone reproduction in computer graphics based on both photographic and perceptual models.

Each of these four units is presented first by focusing on general theory and then by progressing to specific topics and algorithms. For example, in presenting material properties and illumination, the theory and definition of BRDFs is covered first. Then, traditional illumination models are presented as examples of generalized BRDFs. When discussing global illumination, the rendering equation is discussed first, and then ray tracing and radiosity are presented as example approximate solutions to this equation. The rationale behind this approach is to separate the conceptual components of the framework from any particular implementation. One of our goals is for the students to be able to use the conceptual framework provided to identify where to "plug in" the algorithms and techniques that are discussed. It should be noted that units may not necessarily be covered in a linear fashion, but rather are visited as is required in preparation for programming assignments.

4. Student learning

Students are expected to master the fundamental rendering framework and illustrate acquired depth in an advanced technique of their choice. There are four learning components used to achieve this goal: lectures, assignments, readings and projects. Each is briefly described below:

4.1. Lecture

The lecture component conveys basic knowledge about image synthesis. Lectures are used to present the material at three different levels of detail: conceptual, technical, and advanced topics. The conceptually focused lecture provides an overview of a topic, explaining the purpose of an area of the pipeline or of a class of algorithms. The lectures with a technical focus present fundamental techniques or algorithms in an area, including the details required to complete programming assignments. The advanced topic lectures provide a survey of more sophisticated techniques, intended to provide breadth, as well as potential motivation for student projects.

4.2. Programming assignments

As it is our firm belief that the best way to fully understand a technique in computer graphics is to program it, the programming assignments in the course have evolved the most and, are indeed, still evolving.

The first offering of the revised course maintained the single term project approach, using a final examination to assess learning from lecture topics. However, since lecture topics were not necessarily related to their projects, students found the final examination jarring. Choosing examination questions was difficult as well: Should they be easy enough to answer without having had reinforcing learning activities, or difficult enough to require extensive and intensive study? As neither seemed reasonable, programming assignments replaced the examination in the offerings that followed as a mechanism to reinforce the concepts discussed during lecture.

Initially, the elements of the (virtual) photography pipeline were discussed in order. This naturally placed most of the programming assignment topics in the last half of the term. Having all programming assignments due then caused a severely unbalanced workload. Topic order was then adjusted to allow meaningful programming assignments earlier in the course. The next pass added some intermediate checkpoints for the main programming assignment. The current offering is fine-tuning the topic order and adding even more checkpoints to facilitate the alignment of lecture topics and assignments.

4.2.1. The ray tracer assignment

The main programming assignment is designed to step students through the image synthesis pipeline with the goal that they will understand it well enough to code it. Like the course presented in [9], we choose ray tracing to motivate this process. Conceptually, the ray tracing algorithm is elegant and easy to understand. In addition, the code for a ray tracer can be naturally structured and extended thus allowing advanced techniques to be easily added.

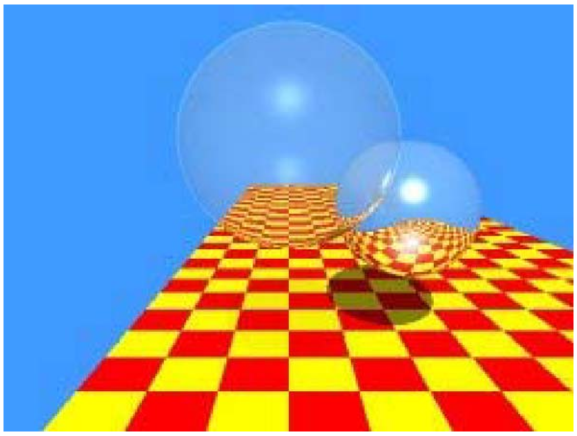
Fig. 2. A classic ray traced image.

This assignment is divided into a number of checkpoints; each checkpoint builds on the previous and focuses on a particular topic. Collectively, the goal of this assignment is to reproduce the classic image first seen in the seminal ray tracing paper [10] and shown in Fig. 2, and to experiment with tone reproduction operators.

A list of the checkpoints for this programming assignment follow and include a discussion of the challenges that students face at each step. The expected output for each checkpoint is shown in Fig. 3.

(1) *Setting the scene*: Construct the scene using a three-dimensional API. The goal is to determine the correct locations and transformations of the objects, as well as the correct camera parameters necessary to produce the model for this image. This step is deceptively challenging as it sets the stage for what is to follow. It is important that students not only set up the model but also think about the kind of data

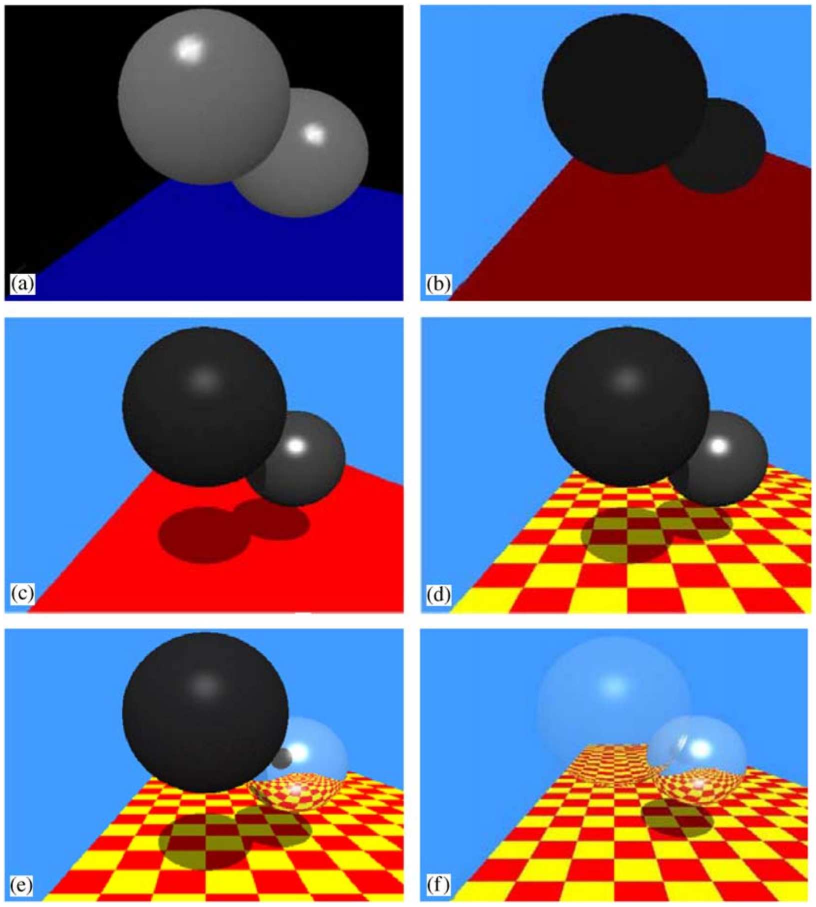

Fig. 3. Sample output for ray tracing assignment: (a) Scene rendered using OpenGL; (b) First level ray tracing through camera model; (c) Phong illumination model; (d) Procedural checkerboard texture; (e) Reflection; (f) Transmission.

structure that will contain it, as their choice will either help or hinder them in succeeding steps. Fig. 3a shows a typical output from this task rendered using OpenGL.

(2) *Camera*: Cast rays using a simple perspective pinhole camera model for visible surface determination. It is during this checkpoint that the basic elements of ray spawning, including definition and storage, are tested. The challenge here is making sure that the camera and model are specified in the same coordinate system, as well as performing the ray-object intersection calculations correctly. The issues of inherent floating-point round-off errors in the intersection calculations come to bear during this checkpoint and must be handled appropriately. Fig. 3b shows the output from this step.

(3) *Illumination*: Add basic Phong illumination. During this checkpoint, students must decide how illumination at ray-object intersection points is to be calculated. Although implementation of the Phong illumination model is assigned for this checkpoint, it is emphasized that their code should be structured such that any illumination model could be substituted. In addition to properly structuring the code, other challenges in this step include: working with vectors, getting in the habit of normalizing them, determining the direction of the perfect reflection of light from the light source and spawning of the shadow ray. Here, students begin to determine the necessary material property constants as well. While they are encouraged to experiment, they are also referred to material property examples from Computer Graphics I. The output of this step is shown in Fig. 3c.

(4) *Procedural shading*: Implement the checkerboard floor as a procedural texture. This is a relatively simple step with the biggest issues being where to anchor the pattern and how large to make it. Although the checkerboard pattern is a relatively simple procedural texture, the point of this checkpoint is to illustrate how procedural textures can be built into the ray tracing process and to compare them with static illumination models, such as the Phong model that was implemented for Checkpoint 3. Fig. 3d shows the output from this step.

(5) *Reflection*: Recursively trace the reflection rays. During this checkpoint, the students begin to deal with the recursive nature of the ray tracing process. They face three challenges: The reflection ray in this case is the reflection of an incoming ray rather than the light vector that is used in Phong illumination. The second challenge is to recognize that this incoming ray changes with each recursive call. The third challenge is making sure that the intersection point found by the ray-object intersection routine for this reflection ray is not the same point as where the ray originated. This problem can occur because of floating point round-off errors. Fig. 3e shows the output from this step.

(6) *Transmission*: Recursively trace the transmission rays. The recursive nature of the ray tracing process is further explored in this checkpoint by considering rays transmitted through objects and bent based on Snell's Law. In this step, besides needing to continue to utilize the good vector practices established in the previous steps, students must recognize that transmission rays are bend twice, once upon entering a medium and once upon leaving it. They also must decide how to detect and handle total internal reflection. Finally, the image in Fig. 2 is completely realized as shown again in Fig. 3f.

(7) *Tone reproduction*: Implement tone reproduction operators. Up until this checkpoint, floating point numbers in the range of 0 to 1 are used to represent lighting values for the ray tracer. In this checkpoint, pixel values of the ray traced image are scaled to represent more physically correct scene luminances. Two different tone reproduction operators, one based on photography [11] and the other on human perception [12], are implemented and applied to the scaled luminances. Three different scaling values are used for each tone reproduction operator. The main challenge is not the implementation of the operators, which are relatively straightforward to code, but rather in explaining and comparing the resulting effects. Fig. 4 illustrates the effect of the operators on the scene. The photographic based operator maps the average luminance in the scene to middle grey. Thus, as luminance is increased, the average luminance increases proportionately, and as a result, the tone mapped images appear to be the same regardless of how they are scaled. This is demonstrated in Figs. 4a–c. The other operator mimics human visual perception. Here, as luminance is increased, the perceived brightness is also increased and, as a result, the image appears brighter, as demonstrated in Figs. 4d–f.

4.2.2. Procedural shading assignment

With the current focus on the advances in procedural shading, the second programming assignment was chosen to provide students further experience with procedural shading. Noting the successes presented in [13], we chose RenderMan™ as the means for exploring this topic. In this assignment, students are asked to create three simple shaders using the RenderMan™ shader language [14], either from scratch or by modifying existing shaders, and to apply them to a simple scene. They are encouraged to go beyond this with bonus points being awarded to that student who produces the "best image" as selected by a vote of the class. Figs. 5 and 6 show two recent winning images.

4.3. Projects

Whereas the programming assignments are designed to give the students a firm foundation in the basic rendering pipeline, the project component allows

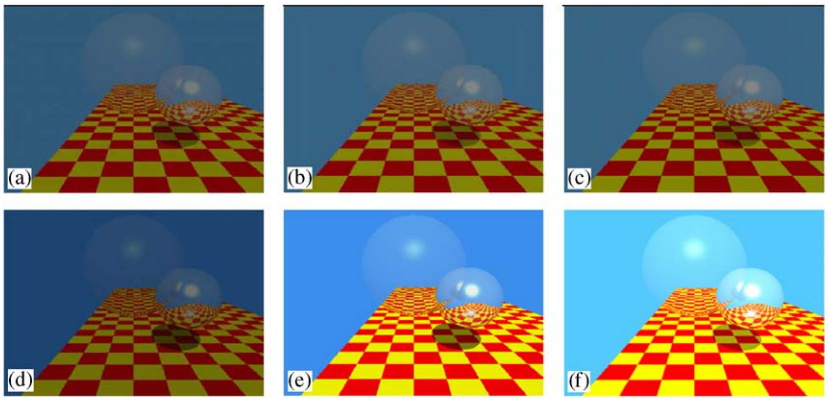

Fig. 4. Application of tone reproduction operators: (a) Photographic operator max luminance: 10 nits; (b) Photographic operator max luminance: 100 nits; (c) Photographic operator max luminance: 1000 nits; (d) Perceptual operator max luminance: 10 nits; (e) Perceptual operator max luminance: 100 nits; (f) Perceptual operator max luminance: 1000 nits.

Fig. 5. Shader assignment winner.

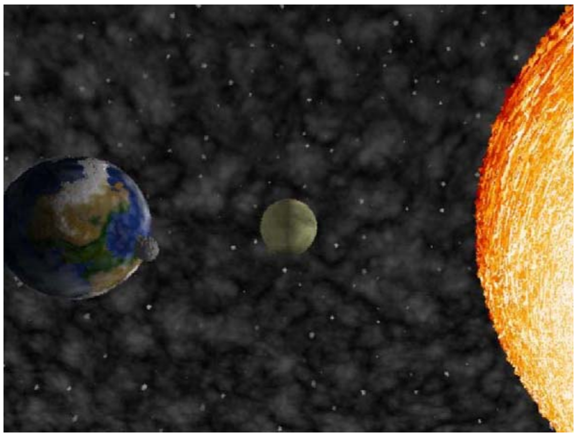

Fig. 6. Shader assignment winner.

students to explore an advanced topic of their choice in depth. Projects are proposed at the beginning of the term and worked on throughout. The final product includes a report and a presentation. These projects may be done in groups of two to three. To help those students who do not have any idea of what they would like to do, suggestions are solicited from faculty from other departments. In the recent past, faculty members from Graphic Design, Imaging Science, and Chemistry, among others, have mentored such interdisciplinary projects. In addition, graduate students are encouraged to consider choosing a project that is relevant to their M.S. project or thesis topic.

4.4. Readings

We do not require a textbook for the course. Instead, we provide copies of notes and an extensive required reading list consisting of seminal graphics papers. Supplemental textbooks are available on reserve at the campus library, e.g., [15–17]. Details for the lectures come from these required readings as well as from the supplemental texts.

One learning outcome for graduate students is preparation for research and advanced study in computer graphics, specifically in image synthesis. This involves surveying the work done in the field to identify problems and areas for future research. As such, graduate students are required to summarize and evaluate a paper of their choice from the reading list for each class. Undergraduates are invited to do the same for extra credit, and many opt to do so. The required readings serve three purposes: First, they reinforce the material presented in lecture. Secondly, they provide motivation and ideas for student projects.

Finally, they introduce students to the graphics literature and help them learn to comprehend technical papers from scholarly journals.

5. Results

Student interest in the course has increased with the change in format. Formerly, the course was offered twice a year, in the winter and summer quarters, but due to popular demand, it is now offered every quarter. While not all of the lecture material may be pertinent to each project, it serves to provide a complete overview of image synthesis; something that was lacking in the projects only course.

We believe that the ray tracing assignment has been effective in student understanding of the rendering pipeline and its relationship to that of photography. Assigning the ray tracer as a number of smaller checkpoints has not only made the task of writing a ray tracer more manageable, but also has aided in the understanding of how each component fits in the pipeline and where advanced methods presented in lecture, but not assigned for implementation, could be added. As a result, each quarter will find at least one student restructuring their code at Checkpoint 5 or 6 based on a deeper understanding of and appreciation for the entire pipeline. In addition, we feel that the ray tracing assignment has engendered an appreciation for physically accurate illumination units and the need for tone reproduction. The inevitable question arises, usually around checkpoint 4, 5, or 6: "What do you do when the illumination calculation exceeds 1.0?" It is then that the realization can occur that such limitations do not exist in the physical world, but are merely an artifact of the traditional treatment of light and color in computer graphics.

As shown by Figs. 5 and 6, the RenderMan™ assignment generates a great deal of enthusiasm. With little guidance, the students are able to create complex and beautiful imagery in a relatively short time. The passion around the RenderMan™ assignment has encouraged us to consider the creation of a follow up course focusing specifically on procedural shading. With procedural shading becoming so pervasive in both real time and off line rendering, this would be a timely and valuable addition to our current graphics offerings.

The required reading assignments have generated some interesting comments. One graduate student said: "The paper summaries are the best part of the course". An undergraduate, who did them for extra credit, stated: "I like that I can now take a research paper and nderstand it, and if I want to implement the authors' ideas in a project that I am doing, I now know that I can!"

The favorite component of the course, however, is the project. Many students indicate that the best thing about the course is the freedom to focus on what they find most interesting. The project presentation days are the most anticipated classes. Student enthusiasm for their work is contagious! And, the projects are generally quite interesting and well done.

Framing the project in the context of the complete photographic pipeline has clearly affected the definition of and the direction of the projects. Some students choose to extend programming assignments, such as adding caustics to the ray tracer as shown in Fig. 7, changing the camera model as illustrated in Fig. 8, or by parallelizing it. Other projects are defined outside of the scope of the assignments yet clearly fall within the rendering framework. For example, one student's urban landscape builder, *City Builder*, shown in Fig. 9, explored the modeling area of pipeline. Our goal is to provide as much latitude and flexibility as possible to enable students to work on something they find particularly interesting.

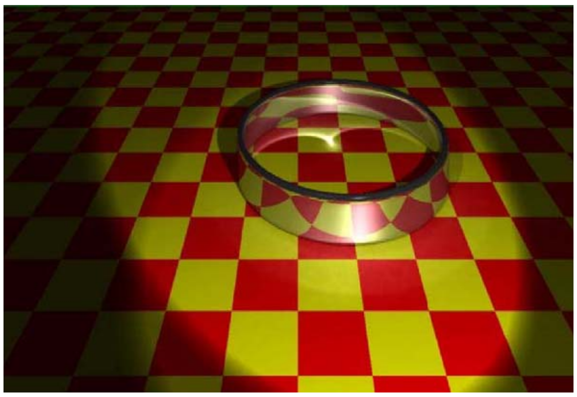
Fig. 7. Adding caustics to the ray tracing assignment.

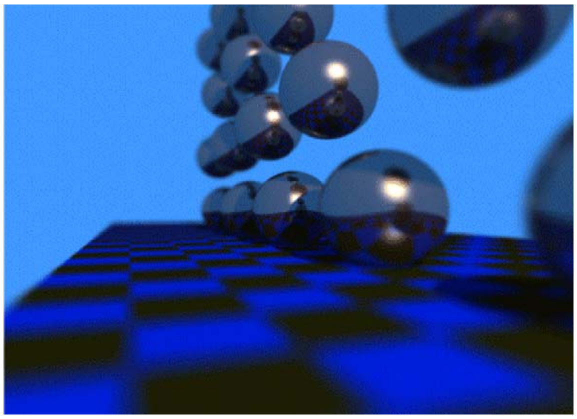
Fig. 8. Replacing the camera model in the ray tracing assignment.

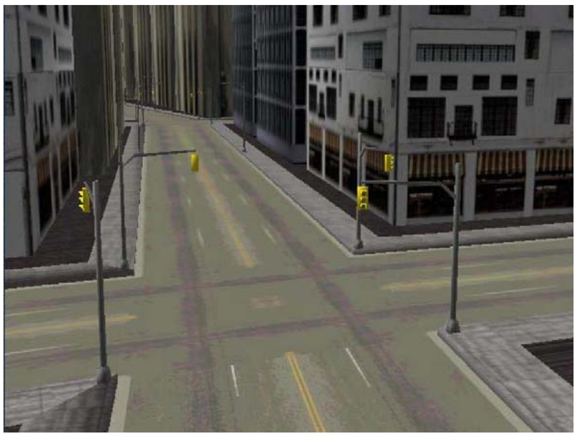
Fig. 9. Screenshot from *City Builder*.

Work on projects often continues after the course is completed. For example, *City Builder* grew into an open source project (*http://citybuilder.sourceforge.net/*), freely available on the web. A three-person team collaborated with eleven others to produce *MegaMonkey* (*http://www.rit.edu/~jpw9607/m3/*), a game based on their project that was entered in the annual International Game Developer Conference's (IGDC) student game competition. For others, their project forms the basis for further exploration through independent study or through interdisciplinary collaboration with other RIT departments. Finally, as previously mentioned, graduate students frequently use the project in this course as a starting point to define their M.S. project or thesis.

6. Conclusions

We have found that virtual photography provides an effective framework for teaching image synthesis. The mix of learning components, presented in the context of the basic and familiar photographic framework has provided a rich, intuitive, and exciting learning experience for students.

Course materials for the most recent offering are available at *http://www.cs.rit.edu/~graphics/cgII*. We invite others to try it and provide us with feedback. Student excitement and involvement in the course has encouraged us to create and apply similar methods in new graphics seminars, including those focusing on computer animation and the application of virtual reality technologies.

Acknowledgements

We want to express our appreciation to Pixar for donating their suite of RenderMan™ software, and to our students, Matthew Bellinger and Patrick McCormick (Fig. 6), Michael Murdoch (Fig. 8), Kevin Pazirandeh (Fig. 7), Kristina Rozanski (Fig. 5), and Andrew Sampson (Fig. 9) for allowing us to share their images.

References

[1] Perlin K, Hoffert EM. Hypertexture. In: Beech RJ, editor. Proceedings of the 16th annual international conference on computer graphics and interactive techniques, Boston, MA, USA, vol. 23, no. 3, 31 July–4 August 1989. p. 253–62.

[2] Kolb C, Mitchell D, Hanrahan P. A realistic camera model for computer graphics. In: Mair SG, Cook R, editors. Proceedings of the 22nd annual international conference on computer graphics and interactive techniques, Los Angeles, CA, USA, 06–11 August 1995. New York: ACM Press; 1995. p. 317–24.

[3] Cook RL, Torrance KE. A reflectance model for computer graphics. ACM Transactions on Graphics (TOG) 1982; 1(1):7–24.

[4] Ward GJ. Measuring and modeling anisotropic reflection. Proceedings of the 19th annual international conference on computer graphics and interactive techniques, Chicago, IL, USA, vol. 26, no. 2, 26–31 July 1992. p. 265–72.

[5] Kajiya J. The rendering equation. In: Evans DC, Athay RJ, editors. Proceedings of the 13th annual international conference on computer graphics and interactive techniques, Dallas, TX, USA, vol. 20, no. 4, 18–22 August 1986. p. 143–50.

[6] Wallace JR, Cohen MF, Greenberg DP. A two-pass solution to the rendering equation: a synthesis of ray tracing and radiosity methods. Proceedings of the 14th annual international conference on computer graphics and interactive techniques, Anaheim, CA, USA, vol. 21, no. 4, 27–31 July, 1987. p. 311–20.

[7] Cook RL, Carpenter L, Catmull E. The Reyes image rendering architecture. Proceedings of the 14th annual international conference on computer graphics and interactive techniques, Anaheim, CA, USA, vol. 21, no. 4, 27–31 July, 1987. p. 95–102.

[8] Devlin K, Chalmers A, Wilkie A, Purgathofer, W. Tone reproduction and physically based spectral rendering. In: Fellner D, Scopignio R, editors. State of the art reports, Eurographics 2002, Geneva, Eurographics Association, 2002. p. 101–23.

[9] Schweitzer D. Ray tracing: a means to motivate students in an introductory graphics course. In: Miller JE, Joyce DT, editors. SIGCSE bulletin, Proceedings of the 21st SIGCSE technical symposium on computer science education, Washington, DC, USA, vol. 22, no. 1, 22–23 February 1990. New York: ACM Press; 1990. p. 157–61.

[10] Whitted T. An improved illumination model for shaded display. Communications of the ACM 1980;23(6):343–9.

[11] Reinhard E, Stark M, Shirley P, Ferwerda J. Photographic tone reproduction for digital images. In: Hughes J, editor. ACM transactions on graphics (TOG), Proceedings of the 29th annual international conference on computer

graphics and interactive techniques, San Antonio, TX, USA, vol. 21, no. 3, 23–26 July 2002. New York: ACM Press; 2002. p. 267–76.

[12] Ward G. A contrast-based scalefactor for luminance display. In: Heckbert PS, editor. Graphics gems IV. Boston: Academic Press Professional; 1994. p. 415–21.

[13] Owen GS. Teaching computer graphics using RenderMan. In: Miller JE, editor. SIGCSE bulletin, Proceedings of the 23rd SIGCSE technical symposium on computer science education, Kansas City, Missouri, USA, vol. 24, no. 1, 05–06 March 1992. New York: ACM Press; 1992. p. 304–8.

[14] Upstill S. The RenderMan™ companion. A programmer's guide to realistic computer graphics. New York: Addison-Wesley; 1990.

[15] Watt A, Watt M. Advanced animation and rendering techniques: theory and practice. New York: ACM Press; 1992.

[16] Glassner A. Principles of digital image synthesis, vols. 1–2. Amsterdam: Morgan Kaufmann; 1995.

[17] Pharr M, Humphreys G. Physically based rendering. Amsterdam: Morgan Kauffman; 2004.