

# Computer Graphics in Undergraduate Computational Science Education

Steve Cunningham  
Computer Science  
California State University Stanislaus  
Turlock, CA 95382  
rsc@cs.csustan.edu

Angela B. Shiflet  
Mathematics and Computer Science  
Wofford College  
Spartanburg, SC 29303  
shifletab@wofford.edu

## Abstract

Computer graphics forms an important part of a working scientist's tools. This may be provided by resources with the scientist's group or by capabilities of the toolkits that the scientist uses directly in his or her work. However, in order for the scientist to understand how the computer graphics images represent models and what the overall possibilities are for this representation and for the occasional time when the scientist may want to develop a presentation that is beyond the capability of the standard tools, it is important for the scientist to understand the basic capabilities and processes of computer graphics. This paper outlines how a computational science program can provide students the background needed to have this understanding.

## 1 An Emerging Discipline

Computational science is an emerging graduate and undergraduate discipline, whose primary focus is the application of computational techniques to understanding, applying, and advancing the sciences. The computational techniques cover many areas, including high-performance computing and distributed collaborative work, but most involve modeling and simulation at their base.

Computational science has grown out of the modern approach to science that sees the field supported by a tripod of activities. These are the traditional activities of theoretical and experimental work, together with the new activity of computation. Computational science involves traditional science fields such as chemistry, physics, or biology; computer science; and applied mathematics, as described in Figure 1. In terms of the organizational structure, computational science is mostly seen as having

its home in applied mathematics, and particularly in the Society for Industrial and Applied Mathematics (SIAM) [SIAM]. In computer science, the computational issues range from very straightforward programming to research-level issues in data mining, networking, and highly parallel processing.

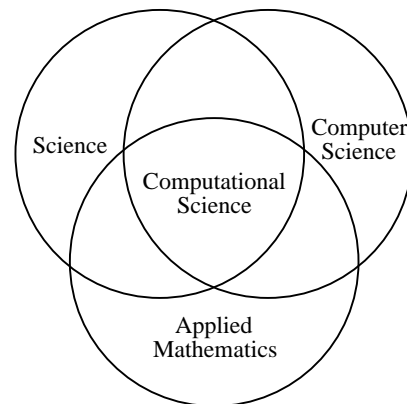


Figure 1: The position of computational science

As computational science becomes better organized, undergraduate and graduate programs are becoming organized in the field. [Turner] [Cunningham 2002a] [Stewart]. For example, in 2001 the SIAM Working Group on CSE published its study of graduate computational science and engineering education [SIAM 2001]. On the undergraduate level, some of these programs are very general, while some are focused in a single science discipline such as the Computational Physics program at Oregon State University [Landau]. In a survey of undergraduate computational science programs sponsored by the Krell Institute, Charles Swanson considered twenty-four undergraduate computational science curricula [Swanson 2001]. Eight of the institutions offer some form of a degree program, such as a minor or a major, while the remaining sixteen have undergraduate computational science courses without a formal degree. The number of programs and courses is expanding rapidly. According to Swanson, Scientific Visualization is among the new courses that often appear in such undergraduate interdisciplinary programs.

## 2 The role of computer graphics in computational science

As we have seen from the rapid growth in the importance of scientific visualization [McCormick], the interactive images and animations that can be produced by computer graphics can be crucial to understand the complexities and subtle details of scientific problems. These images can range from the personal images a scientist would use to probe a detail of a problem to the very sophisticated images developed by a visualization team to present a new theory or solution to the scientific community. The goal is to introduce computer graphics into the problem-solving cycle as shown in the two right-hand boxes in Figure 2; here the geometry and image fill the same role as the experimental process and results would in the classical presentation of the scientific method.

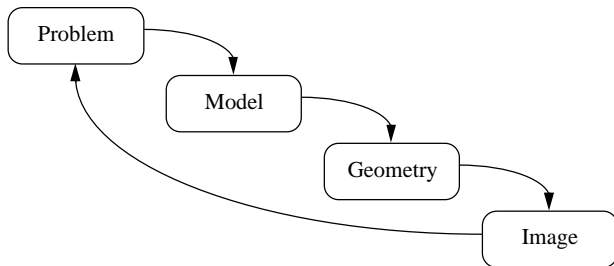


Figure 2: computer graphics in the problem-solving loop

Because computer graphics images are simply part of the problem-solving process and not an end in themselves, there is probably a tendency for computational science to pick tools that work reasonably well and settle for those tools for everyday work. This may be adequate, but tools both empower and limit their users and it is quite possible that using standard tools may limit the way people can think about their science.

## 3 Computer graphics in the computational science curriculum

As students work in the modeling and simulation that are part of computational science studies, they will often find the need to express results in a graphical way. In doing so, students will use appropriate-level tools, not sophisticated graphical techniques, because the goal of their studies is to develop broader skills in the overall scientific process, not to become scientific visualization specialists.

Because of the need to provide appropriate tools for students, a computational science program will probably include several different kinds of computer graphics approaches to familiarize the students with many ways to integrate computer graphics into their work. These may well include Matlab [MathWorks], Mathematica [Wolfram], Maple [Maple], Excel [Microsoft], and STELLA [HPS]. Each of these tools has a specific kind of problem it addresses best, and the graphical displays it presents will be focused on that kind of problem. For example, Figure 3 shows a two dimensional Mathematica

graphics for one time step of a cellular automata simulation of the spread of fire in a forest. Although Figure 3 is shown in grayscale, the original graphic uses green, yellow, and red to indicate a tree, an empty site, or fire, respectively. A student can develop a Mathematica program to perform the simulation, generate an image for each time step, and animate the results. By altering various

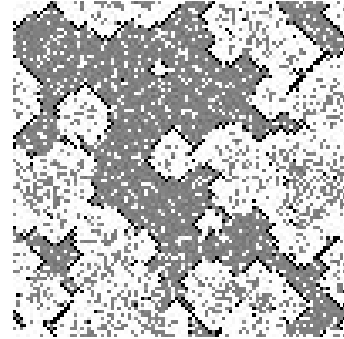


Figure 3 Mathematica cellular automata of the spread of fire

parameters, such as the probability of lightning, the user can investigate various aspects of the spread of fire. MatLab, Maple, and Excel are also suitable for such two-dimensional graphical simulations. The STELLA tool models dynamic systems, such as predator/prey populations (Figure 4). With STELLA, the scientist or student

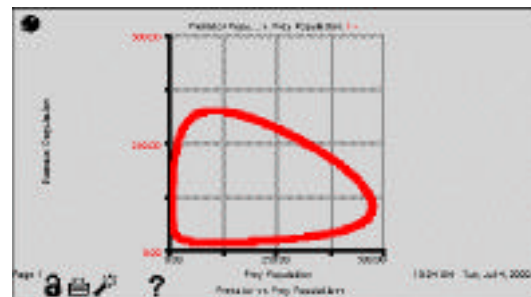
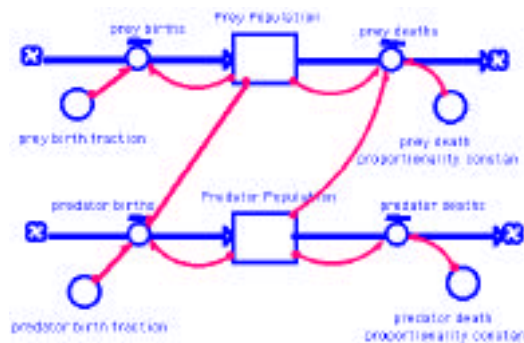


Figure 4 Predator-Prey STELLA model (top) and graph of predator versus prey populations (bottom)

develops a diagram of the model and generates various graphics to depict the results of time-driven simulations.

Tools such as Mathematica, MatLab, and STELLA are often used as part of a course in modeling and simulation.

As we look at any particular tool, however, it is clear that it presents only part of the set of images that might be used to represent a given problem. In Figure 5, we diagram the set of all possible images for a problem as the lighter gray circle, and the set of images that a particular tool will produce as the darker gray circle. The location of the smaller circle in the larger circle is the *bias* of the tool; it shows the position of the tool's images in the overall set of images. The size of the smaller circle is the *coverage* of the tool; it shows how broad a set of images the tool will provide in the overall set of possible images. It is reasonable to believe that the bias and coverage of the tool are measures of the way the tool influences the process of understanding the results of a model or simulation, encouraging one way and limiting other ways of thinking about a problem [Green].

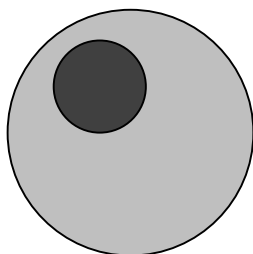


Figure 5: the relation between the images produced by a particular tool and all possible images for a problem

Because of the limitations of pre-existing tools in providing ways to visualize a problem, we believe that it is important that a computational science program include a computer graphics course. This will give students the experience of proceeding from a model to an image and will show them how many variations are possible in the way a model is presented visually. An appropriate kind of course [Cunningham 2000] will also give students experience in creating an interactive and animated presentation of an idea and will teach students important ideas in communicating their science visually. This will allow them to understand just what the limits of pre-written tools are and to begin to be able to develop their own presentations when the tools are inadequate for a given problem.

It is important, however, that the computer graphics course be appropriate for the program. The traditional computer graphics course (for example, [Foley]) focuses on the technology of creating images rather than fundamental graphics concepts and their use in creating effective images for real applications. These fundamental concepts, including the graphics pipeline, modeling, transformations, rendering, color, lighting, shading, interaction, texture mapping, and the like, can all be expressed in a practical way through programming with a current graphics API, such as OpenGL [Cunningham 2002c]. Along with these concepts, the course can also include basic principles of

visual communication for the sciences or another application area, preparing the computational science student both to understand and create effective images for his or her science [Cunningham 2002b].

#### 4 Experiences

Wofford College offers an Emphasis in Computational Science for interested students obtaining a B.S. in a laboratory science or mathematics. Requirements include C++ Programming, Data Structures, Calculus I, two computational science courses—Data and Visualization (CS 370) and Modeling and Simulation (CS 375)—and an internship involving computation in the sciences [Shiflet 2000]. Half of the CS 370 course covers concepts of computer graphics as they apply to scientific visualization.

With a prerequisite of solid skills in C or C++ programming, students quickly learn to create interactive three-dimensional presentations or computer animations by programming in C or C++ with the graphics API OpenGL. The course endeavors to "teach fundamentals, not packages." A commercially available package usually performs standard tasks, while a scientist performing leading-edge research often must write or revise specially tailored programs. Moreover, the graphics tools of today can change significantly in a short amount of time. Learning fundamental graphics concepts and algorithms and how to create scientific visualizations by programming enables these developing scientists to master new visualization tools quickly and to employ these tools to obtain insights and to communicate information effectively.

Using the online draft text *Computer Graphics: Programming, Problem Solving, and Visual Communication* by the first author [Cunningham 2002c], all examples and assignments in Wofford's Data and Visualization course involve scientific applications as well as instruct the student on fundamental graphics concepts.

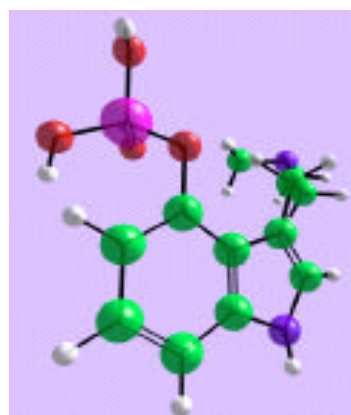


Figure 6 interactive display of the psilocybin molecule from a standard molecule description file

The class examines topics such as viewing, geometric modeling, transformations, color, visual communication,

lighting, shading, and event handling with examples such as diffusion of heat through a metal bar, the spread of malaria, gas laws, and molecular models (see Figure 6). Students develop three-dimensional animations of such scientific applications as a rotating DNA helix and interacting ocean waves.

Ten years ago, the ability to create such animations required a supercomputer and sophisticated software. Today, students at even small liberal arts colleges can study scientific visualization and generate quality work with readily available compilers and personal computers.

## 5 References

- [Cunningham 2000] Steve Cunningham, "Powers of 10: The Case for Changing the First Course in Computer Graphics," *Proceedings*, ACM SIGCSE Technical Symposium on Computer Science Education, Austin, TX, March 2000, 293-296
- [Cunningham 2002a] Steve Cunningham et al., "Computational Science and Engineering -- Tools and Techniques for Teaching," *Proceedings*, ACM SIGCSE Technical Symposium on Computer Science Education, Covington, KY, February 2002, pp. 135-136
- [Cunningham 2002b] Steve Cunningham, "Graphical Problem Solving and Visual Communication in the Beginning Computer Graphics Course," *Proceedings*, ACM SIGCSE Technical Symposium on Computer Science Education, Covington, KY, February 2002, pp. 181-185
- [Cunningham 2002c] Steve Cunningham, *Computer Graphics: Programming, Problem Solving, and Visual Communication*, draft manuscript available online at <http://www.cs.csustan.edu/~rsc/NSF>
- [Foley] James D. Foley et al., *Computer Graphics Principles and Practice*, second edition, Addison-Wesley, 1996
- [Green] Mark Green, "VR Authoring Tools for Non-Programmers: A Case Study," presentation at VRAI 2002, Hangzhou, China, April 2002
- [HPS] HPS Inc., developers of STELLA, <http://www.hps-inc.com>
- [Landau] Rubin H. Landau and Manuel J. Páez, *Computational Physics: Problem Solving with Computers*, Wiley, 1997
- [Maple] Waterloo Maple, developers of Maple, <http://www.maplesoft.com>
- [MathWorks] The MathWorks, developers of MATLAB, <http://www.mathworks.com>
- [McCormick] Bruce McCormick, Thomas A. DeFanti, and Maxine D. Brown, *Visualization in Scientific Computing*, *Computer Graphics* 21(6), 1987
- [Microsoft] Microsoft Corp., developers of Excel, [www.microsoft.com](http://www.microsoft.com)
- [Shiflet 2000] Angela B. Shiflet, Description of Wofford College's Emphasis in Computational Science, online at [http://www.wofford.edu/computerscience/emphasis\\_in\\_cs.htm](http://www.wofford.edu/computerscience/emphasis_in_cs.htm)
- [Shiflet 2001] Angela B. Shiflet, *Computational Science Modules*, available online at <http://www.wofford.edu/ecs>
- [Stewart] Kris Stewart and Jose Castillo, "Computer Graphics and the Continuum of Computational Science Education," SIAM Conference on Computational Science & Engineering, Sept. 21-24, 2000, Washington, D.C.
- [Swanson] Charles D. Swanson, "Computational Science Education," Krell Institute, available online at <http://www.krellinst.org>
- [SIAM] Society for Industrial and Applied Mathematics, activity group on computational science and engineering, <http://www.siam.org/siags/siagcse.htm>
- [SIAM 2001] SIAM Working Group on CSE Education, "Graduate Education in Computational Science and Engineering," *SIAM Review*, Vol. 43, No. 1, pp. 163-177, March 2001, available online at <http://www.siam.org/journals/sirev/43-1/37974.html>
- [Turner] Peter R. Turner et al., "Undergraduate Computational Science and Engineering Programs and Courses," *Proceedings*, ACM SIGCSE Technical Symposium on Computer Science Education, Covington, KY, February 2002, pp. 96-97
- [Wolfram] Wolfram Research, developers of Mathematica, <http://www.wolfram.com>