CMSC 427: Computer Graphics

Spring 2004

http://www.cs.umd.edu/~mount/427/

Instructor: Dave Mount. Office: AVW 3373. Email: mount@cs.umd.edu. Office phone: (301) 405–2704. Office hours: Mon 2:30-3:30, Wed 2:30-3:30. I am also available immediately after class for questions. Please send me email if you cannot make these times. If the question is short (a minute or so) drop by my office any time. Please send me email if you cannot make these times. (Don't be shy about doing this. I always set aside at least one hour each week for "unscheduled" office hours.)

Teaching Assistant: Pooja Nath. Office: AVW 1112. (If you do not see her there, try her office, AVW 3444.) Email: pooja@cs.umd.edu. Office hours: (TBA, see the class web page). If you cannot make these times, please feel free to contact her to set up another time.

Class Time: Tue, Thu 2:00-3:15 in CSI 3117.

Course Objectives: This course provides an introduction to the principles of computer graphics. In particular, the course will consider methods for modeling 3-dimensional objects and efficiently generating photorealistic renderings on color raster graphics devices. The emphasis of the course will be placed on understanding how the various elements that underlie computer graphics (algebra, geometry, algorithms and data structures, optics, and photometry) interact in the design of graphics software systems.

Texts: It is strongly recommended that you buy the required text. The reference book is a good source of advanced information, if you intend to do advanced graphics programming.

Required: *Computer Graphics with OpenGL* (3rd edition), D. Hearn and M. P. Baker, Prentice Hall, 2004.

Reference:

- *OpenGL Programming Guide: The Official Guide to Learning OpenGL* (Fourth Edition), by OpenGL Architecture Review Board, *et al.*, Addison-Wesley, 2003.
- *OpenGL Reference Manual: The Official Reference Document to OpenGL* (3rd Edition), by Dave Shreiner, *et al.*, Addison-Wesley, 1999.

Prerequisites: MATH 240 (Linear Algebra) and CMSC 420 (Data Structures). Knowledge of C, C++, or Java programming. The course involves a considerable amount of mathematical reasoning involving 3-dimensional objects (points, lines, spheres, and polygons). Knowledge of and the ability to solve problems in linear algebra and (primarily differential) calculus will be required. We will give an overview of linear algebra in class, but if you are unfamiliar in your understanding of concepts from linear algebra (such as vector spaces, linear independence, bases, linear transformations, determinants, and inner products) and differential calculus (including partial derivatives), I recommend that you review this material. The course involves some nontrivial programming projects. Although a specific knowledge of data structures is not essential, I will assume that you are capable of writing and debugging moderately sophisticated programs in either C, C++, or Java.

Course Work: Course work will consist of a combination of written homework assignments and three programming projects. Homeworks are due at the start of class. Late homeworks are not allowed (so just turn in whatever you have done by the due date). Programming assignments will typically be due at midnight of the due date. They are subject to the following late penalties: up to six hours late: 5% of the total; up to 24 hours late: 10%, and then 20% for every additional day late.

There will be two exams: a midterm and a comprehensive final. Tentative weights: Homeworks and projects 35%, midterm 25%, final exam 40%. The final exam will be Mon, May 17, 10:30–12:30.

As a courtesy to the grader, homework assignments are to be written up neatly and clearly, and programming assignments must be clear and well-documented. Although you may develop your program on whatever system you like, for final grading your program must execute either on a workstation (Microsoft Windows, Linux, or Sun Solaris) either in the WAM, Glue, or Linux labs. If you develop your program on some other platform, it is your responsibility to see that it can be compiled and executed on one of these machines by the due date. If not, you will be asked to make whatever changes are needed and will be assessed a late penalty as a result.

Some homeworks and projects will have a special challenge problem. Points from the challenge problems are *extra credit*. This means that I do not consider these points until *after* the final course cutoffs have been set. Each semester extra credit points usually account for at least few students getting one higher letter grade.

Academic Dishonesty: All class work is to be done independently. You are allowed to discuss class material, homework problems, and general solution strategies with your classmates. When it comes to formulating/writing/programming solutions you must work alone. If you make use of other sources in coming up with your answers you *must* cite these sources clearly (papers or books in the literature, friends or classmates, information downloaded from the web, whatever).

It is best to try to solve problems on your own, since problem solving is an important component of the course. But I will not deduct points if you make use of outside help, provided that you cite your sources clearly. Representing other people's work as your own, however, is plagiarism and is in violation of university policies. Instances of academic dishonesty will be dealt with harshly, and usually result in a hearing in front of a student honor council, and a grade of XF.

Topics: The topics and order listed below are tentative and subject to change.

Introduction: Overview of graphics systems, graphics devices, graphics programming.

Graphics Programming: OpenGL, graphics primitives, color, viewing, event-driven I/O, GL toolkit, frame buffers.

Geometric Programming: Review of linear algebra, affine geometry, (points, vectors, affine transformations), homogeneous coordinates, change of coordinate systems.

3-d transformations and perspective: Scaling, rotation, translation, orthogonal and perspective transformations, 3-d clipping.

Light and shading: Diffuse and specular reflection, Phong and Gouraud shading.

Using Images: Texture-, bump-, and reflection-mapping.

Implementation Issues: Rasterization, clipping.

Ray tracing: Ray-tracing model, reflective and transparent objects, shadows, light transport and radiosity.

Hidden surface removal: Back-face culling, z-buffer method, depth-sort.

Color: Gamma-correction, halftoning, and color models.

Modeling: Hierarchical models, fractals and fractal dimension.

Curves and Surfaces: Representations of curves and surfaces, interpolation, Bezier, B-spline curves and surfaces, NURBS, subdivision surfaces.