Course Overview

This course introduces advanced visualization techniques for various types of measured or computer-simulated data. Typical applications for these visualization techniques include the study of airflows around car bodies, medical data, and molecular structures.

Course Objectives

The primary goals are to solidify the concepts introduced in ECE105 by studying advanced topics pertaining to raster graphics and to introduce the student to current research topics in the field. Prepare students to understand and conduct research in this area.

I. Review of essential graphics algorithms

II. Advanced scalar field visualization
   A. Multiple, arbitrary slicing surfaces
   B. Transparency paradigms
   C. Multiple, transparent isosurfaces
   D. Internal-based contouring

III. Advanced volume visualization
   A. Review of Levoy’s algorithm
   B. Sabella’s algorithm
   C. Splatting

IV. Advanced vector field visualization
   A. Ray casting vector fields
   B. Topological visualization
   C. Features in vector fields

V. Data compression and reduction
   A. Pyramid hierarchies for structured data
   B. Approximation techniques for hierarchies of fields
   C. Data reduction methods

VI. Animation for visualization
   A. Approximation methods for animation
   B. Animation of field data
   C. Animation of image space data

VII. Tessellation methods
   A. Voronoi diagrams
   B. Delannay triangulations

The design of a comprehensive modeling system is embodied in milestone papers that are required in advance of the project deadlines. The students are required to design and implement all of the system components. Examinations will include questions based on the design concepts learned during the individual projects in addition to formal concepts taught in class.
Prerequisites

In this course students will primarily use C/C++ and OpenGL as the graphics API to study the theoretical foundation and design of computer graphics. ECE105 or instructor consent is required in addition to good knowledge of algorithms, data structures, linear algebra and trigonometry.

Course Organization

First Day of Class: 01/12/2004  
Course Code: 15800
Lectures: M/W 09:00-10:20  
Location: EG3131
Lecture: 3 hours  
Laboratory: 3 hours
Credit Units: 4  
(Design Units: 1)
Homepage: http://graphics.eng.uci.edu/courses/ece205

Instructor Information:

Prof. Falko Kuester  
444D Engineering Tower  
Office Hours 10:30-12:00
Email: fkuester@uci.edu  
Homepage: http://graphics.eng.uci.edu

Course Outcomes

The students will design prototype software systems to be used for the analysis and visualization of scientific and engineering data sets. In particular, systems for the analysis of mechanical engineering, aerospace engineering, ocean and atmosphere modeling, and medical imaging data will be designed. The design will include both interface and algorithm design

• Students are able to write graphics applications using C/C++, OpenGL and GLUT.
• Students are able to write concisely structured and documented application programs.
• Students are able to implement a complete visualization package for scalar and vector field visualization.
• Students are able to work with structured and unstructured time varying datasets.
• Students are able to interpret multi-dimensional data.

Resources

All course material will be provided on a special Web site including lecture notes, laboratory notes, useful links on the web, recommended references, time schedule, contact information for faculty and TAs, guidelines for projects, coding standards and more (http://graphics.eng.uci.edu/courses/ece205).

Textbook

The following two books will be available at the university bookstore and are strongly recommended for this course. The textbook covers all major topics that will be covered in this course and will provide a valuable resource for further advanced study. The other book is the official OpenGL programming guide, which covers most of the OpenGL related topics, required for the successful completion of the course projects.


Course Structure

The quarter is generally organized as follows (tentatively):

<table>
<thead>
<tr>
<th>Week</th>
<th>Monday</th>
<th>Wednesday</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Overview</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Midterm</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>Advanced Topics</td>
<td></td>
</tr>
</tbody>
</table>

Assignments, Exams and Grading

The final course grade will be based on projects, a midterm and a final exam. The projects, as a major part of this course, will contribute 60% to the final grade. Assignments have to be turned in electronically (see details on course web page).

<table>
<thead>
<tr>
<th>Grade [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project 1</td>
</tr>
<tr>
<td>Project 2</td>
</tr>
<tr>
<td>Project 3</td>
</tr>
<tr>
<td>Project 4</td>
</tr>
<tr>
<td>Midterm</td>
</tr>
<tr>
<td>Final</td>
</tr>
<tr>
<td>Total (100%)</td>
</tr>
</tbody>
</table>

Each project will be graded based on completeness, correctness, the students understanding of the algorithms and implementation (80%), and the style (20%). Style considerations include the user interface design, the written project documentation and coding standard compliance. All projects will be graded on a 100-point basis.

<table>
<thead>
<tr>
<th>Category</th>
<th>[%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Completeness &amp; correctness</td>
<td>80%</td>
</tr>
<tr>
<td>User Interface</td>
<td>10%</td>
</tr>
<tr>
<td>Documentation</td>
<td>5%</td>
</tr>
<tr>
<td>Coding standard compliance</td>
<td>5%</td>
</tr>
<tr>
<td>Total</td>
<td>(100%)</td>
</tr>
</tbody>
</table>

Only programs that can be compiled and executed on the laboratory machines will be graded. Incomplete projects will only be graded if a detailed description of the available and missing functionality is provided.

Each class is different, and no absolute grading scheme can be defined in advance. However, the following guarantees will always be made

90%  80%  70%  60%  50%
A     B     C     D     F

A grade of ‘Incomplete’ is not given unless extreme circumstances are presented. Students generally should be prepared to retake the course completely.
Late Policy

To receive full credit on a project, the project must be turned in on the official due date. Projects that are turned in late will be subject to a 10-point daily penalty. A weekend counts as one day.

Course Load

The projects require significant effort and in general substantial time commitment, specifically during the final weeks of the quarter. Keep this in mind when you plan your total course load. If you plan on taking other courses that require finishing a project by the end of the quarter, you should consider this in your course strategy.

Is this the right course for me?

ECE205 does require C, C++, Java or Python programming skills and a good background in mathematics (see the course prerequisites for more information). All of the assignments and projects require programming in either one of the mentioned languages in combination with OpenGL as the graphics API of choice. OpenGL knowledge is required before taking this class. Most importantly, a certain addiction to programming and familiarity with UNIX/Linux or Windows environments are important. If you are not familiar with these topics, you should be prepared to spend a considerable amount of time on acquiring these skills that a crucial for the implementation of the individual group projects. Lectures and lab sessions present a substantial amount of technical information required for the understanding of the material and successful development of the projects. You should not only be interested in the fascinating aspects of computer graphics, but also be willing to learn and apply the concepts. The goal of the course is to do computer graphics and not just to learn about it. The completion of the course projects is a rewarding experience and a valuable step towards applying your cumulative theoretical knowledge in mathematics, physics and other engineering disciplines.

Academic Integrity

Academic integrity (honesty) is taken very seriously. It is the responsibility of each student to be familiar with UCI's current academic honesty policies. Please take the time to read the current UCI Senate Academic Honesty Policies. Please note that any form of academic dishonesty could result in a grade of “F” in the class in addition to disciplinary action from the department or university. Detailed information about the Academic Senate Policy on Academic Honesty can be accessed through the references section on the course web page.

However, we encourage cooperation within well-defined bounds. This is a very challenging and work intensive course that will require you to refine and improve your mathematics, computer language and programming skills. At times it is difficult to learn all of the necessary programming techniques and tricks, implementation philosophies and debugging strategies on your own. I encourage you to draw from different sources including your classmates to improve these skills and to learn about available tools.

At the same time all assignments (source code) that you turn in has to be your own. You are allowed to talk to other students, the TAs or the instructor, share ideas, strategies and design philosophies, but in the end do your own work and write your own code. This means, you may not use material/code that anyone other than yourself has written. Material/code explicitly forbidden includes code taken from the web, from books or from any source other than yourself.

The only exception to this rule is that you should feel free to use any of the routines that are provided by the instructor or TA.
Add/Drop Policy

You can add or drop this course via TELE during the first two weeks of the Spring Quarter. The course instructor will not sign any Add/Drop cards. If you have trouble enrolling in this course via TELE due to restrictions or prerequisites, go to the Engineering Student Affairs Office for authorization. After the second week of the Spring Quarter, you will need to provide justification for adding or dropping the course. Requests to add or drop this course after the first two weeks will be handled by the Student Affairs office.

Extract from the Campus Final Examination Policy

Final examinations are to be administered during the examination week at the time announced on page 23 in the Schedule of Classes Fall 2002.

Computer Usage

Students implement their term projects with the C and/or C++ programming language, using the computer systems available in the Henry Samueli School of Engineering instructional facilities.

Laboratory Projects

The programming projects for this class are chosen to enhance the lecture material in the course.

Estimated ABET Category Content

<table>
<thead>
<tr>
<th>Engineering Science</th>
<th>2 credit units or 66.6 %</th>
</tr>
</thead>
<tbody>
<tr>
<td>Engineering Design</td>
<td>1 credit units or 33.3 %</td>
</tr>
</tbody>
</table>

Prepared by: Falko Kuester       Date: May 10, 2002

Contact person for questions: Dr. Falko Kuester, 4-9320, fkuester@uci.edu

Last Modified: 1/12/2004