Principles of Virtual Reality

EECS 208
Course Syllabus

Course Overview

This course introduces cutting-edge virtual reality technology currently available in academia and industry. It provides an introduction to the physical principles, technological challenges, possibilities and limitations for the creation of virtual environments. Programming projects emphasize the visualization, exploration and modification of scientific data in virtual environments.

Course Objectives

This course will focus on the fundamentals of Virtual Reality (VR) and provide extended laboratory and programming exposure. The course objective is to promote the understanding of this technology, underlying principles, its potential and limits and to learn about the criteria for defining useful applications. Furthermore, students will be exposed to the process of creating virtual environments, by developing a complete VR application as members of a development group. Project groups will include students from different disciplines, thereby, complementing technical skills with imagination, creativity, and innovative ideas. Engineering students will work side-by-side with liberal arts and other students and practice problem solving in a team environment. "Principles of Virtual Reality" was developed for facilities available in the department of Electrical and Computer Engineering and its graphics group consisting out of a wide range of high-end workstations and a state-of-the-art virtual reality hardware located in the Visualization and Interactive Systems Laboratory (VIS). VR is changing the interface between people and information technology by offering new ways for the communication of information, the visualization of processes, and the creative expression of ideas. Course topics include:

I. Review of essential principles
II. Classification
   A. Semi-immersive environments
   B. Fully-immersive environments
   C. Augmented environments
III. Stereo-vision
   A. The human vision system
   B. Technological implementation
IV. Human-computer interaction
   A. 3D interaction paradigms
   B. Hardware interfaces
V. Virtual reality device technology
   A. Display devices
   B. Position tracking
VI. Multi-modal interaction feedback
   A. Visual input/output
   B. Acoustic input/output
   C. Haptic input/output
   D. Others
VII. Real-time visualization
   A. Data simplification and compression
   B. Real-time algorithms
VIII. Related Issues
IX. Health hazards
The design of a comprehensive visualization system is embodied in milestone specifications 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. EECS104 or consent by the instructor is required in addition to good knowledge of algorithms, data structures, basic linear algebra and trigonometry.

**Course Organization**

First Day of Class: 04-03-2006  
Course Code: 15810 (13096, 37130, 13096)  
Lectures: M/W 09:00-10:20  
Lecture: 3 hours  
Credit Units: 4  
(Design Units: 1)  
Location: EG 3131  
Laboratory: 3 hours  
Homepage: [http://vis.eng.uci.edu/courses/eecs208](http://vis.eng.uci.edu/courses/eecs208)

Instructor Information:

Prof. Falko Kuester  
Email: fkuester@uci.edu  
444D Engineering Tower  
Homepage: [http://vis.eng.uci.edu/~fkuester](http://vis.eng.uci.edu/~fkuester)  
Office Hours M/W 10:30-11:30, ET 444D

Co. Instructor:

Dr. Kai-U. Doerr  
Email: kdorr@uci.edu  
537 Engineering Tower  
Homepage: [http://vis.eng.uci.edu/~kdoerr](http://vis.eng.uci.edu/~kdoerr)  
Office Hours M/W 10:30-11:30, ET 537

**Course Outcomes**

EECS 208 is an application- and project-oriented design class. Students learn about various scientific and engineering applications requiring the use of computer visualization methods for complex data exploration and analysis. The student will design and prototype virtual environments for the visualization, exploration and modification of scientific data sets. In particular, systems for the analysis of medical, biological and environmental data as well as mechanical engineering tasks in the automotive and aerospace industry will be emphasized. Programming projects are concerned with the development of rather application-dependent visualization methods, creation of stereoscopic images and integration with multi-modal input devices. This course requires students to build on and apply their knowledge of object-oriented programming (languages), complex data structure design, and modular system 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 virtual reality application supporting stereoscopic vision and multimodal input devices such as six degree-of-freedom trackers and data gloves.
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.


Textbook

A collection of research papers will be made available to students and serve as a substitute for a textbook. The following two books will be available at the university bookstore and are recommended for this course. Both books provide detailed resources for further advanced study and important insight about the OpenGL graphics API.


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>Introduction to Virtual Reality</td>
</tr>
<tr>
<td>2</td>
<td>Human Factors and Perception</td>
<td>Device Technology (input)</td>
</tr>
<tr>
<td>3</td>
<td>Device Technology (output – displays)</td>
<td>Device Technology (output – sonification/haptics)</td>
</tr>
<tr>
<td>4</td>
<td>3D Interaction Paradigms</td>
<td>Applications</td>
</tr>
<tr>
<td>5</td>
<td>Data Simplification and Compression</td>
<td>Data Presentation</td>
</tr>
<tr>
<td>6</td>
<td>MIDTERM</td>
<td>Stereoscopic Imaging I</td>
</tr>
<tr>
<td>7</td>
<td>Stereoscopic Imaging II</td>
<td>Spatial Tracking and Interaction I</td>
</tr>
<tr>
<td>8</td>
<td>Spatial Tracking and Interaction II</td>
<td>Sonification</td>
</tr>
<tr>
<td>9</td>
<td>Real-time algorithms I</td>
<td>Real-time algorithms II</td>
</tr>
<tr>
<td>10</td>
<td>Distributed Environments</td>
<td>Advanced Topics</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>Midterm</td>
</tr>
<tr>
<td>Final</td>
</tr>
<tr>
<td>Total</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.
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:

<table>
<thead>
<tr>
<th></th>
<th>90%</th>
<th>80%</th>
<th>70%</th>
<th>60%</th>
<th>50%</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>B</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>F</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

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?

EECS208 does require computer graphics knowledge, and either C, C++, Java or Python programming skills. Assignments and group projects require programming in either one of the mentioned languages or scripting in VRML. Most importantly, a certain addiction to programming and familiarity with UNIX/Linux, the World Wide Web and geometric modeling are important. If you are not familiar with these topics, you should be prepared to spend a reasonable amount of time on acquiring these skills. Project groups should include students from different disciplines, thereby, complementing technical skills with imagination, creativity, and innovative ideas. Engineering students should work side-by-side with liberal arts and other students and practice problem solving in a team environment. In order to hit the critical mass for joint project, groups should consist out of 2 or 3 students. Lectures and lab sessions present a considerable amount of technical information required for the successful development of a VR application. You should not only be interested in the fascinating aspects of VR, but also willing to learn and apply the concepts. 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
Principles of Virtual Reality

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 Spring 2006*.

**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.