CSCI 440 Computer Graphics

SEMESTER (Spring 2015)

PROFESSOR/CLASS INFORMATION

Dr. Shieu-Hong Lin

(Course) Title: Computer Graphics
Term: Spring, 2015
Location: Busn 209
Office Phone: 562 903-4741
Office Hours: T Th 1:30-2:30pm
E-Mail: shieu-hong.lin@biola.edu
University Website: www.biola.edu

Course Code/#: CSCI 440
Class Days/Time: MW 1:30-2:45pm
Credit Hours/Units: 3
Office Location: White 48
Meetings with Professor: Make Appt via Email
Admin Assistant: Jerrianne Smith, x4741
Dept. Website: http://csci.biola.edu

Class Website: http://csci.biola.edu/csci440graphics/

COURSE ALIGNMENT WITH PROGRAM LEARNING OUTCOMES

CSCI 440 Computer Graphics: This upper-division course is an elective course for Computer Science majors designed to be taken within the junior or senior year of the program. Successful completion of this course (see next section) will prepare students to demonstrate a developing proficiency toward the accomplishment of PLO: programming and system integration.

COURSE OBJECTIVES AND STUDENT LEARNING OUTCOMES

By the completion of this course including class participation, class assignments (referred to as "Tasks"), class readings and group interaction, the following objectives and learning outcomes will be assessed and demonstrated:

IDEA Objective #4: Developing specific skills, competencies, and points of view needed by professionals in the field most closely related to this course (Essential emphasis).

STUDENT LEARNING OUTCOMES (The learner will demonstrate that he or she has satisfactorily fulfilled IDEA Objective #4 by being able to):

- Describe (i) the fundamental principles for modeling 3-dimensional objects and generating renderings efficiently and (ii) how modern graphics software systems are built upon concepts in optics, photometry, algebra, geometry, algorithms, and data structures.

- Develop interactive computer graphics programs using OpenGL as the underlying graphics application programming interface to mode and render scenes of 3D environments.
- Design and implement an elementary 3D virtual world providing interesting scenarios for the user to interactively explore the 3D virtual world.

REQUIRED TEXTS

- OpenGL 1.1 programming guide (Online version: The OpenGL "Redbook" v1.1 HTML format)
- Online OpenGL 1.1 reference guide (Online version: The blue book)

LEARNING TASKS (Assignments) & ASSESSMENT (Grading)

Description and Weighting of Assignments:

Task 1: Weekly Reading and Progress Report

Due Date: Monday of the week **(14 assignments)**
Weighting: 10%
Possible Points: 4 points for each assignment.
Description:
For each reading assignment, the student needs to finish the reading on time and submit the following information online as a report.

Effort (2 points):
Record the information such as (i) a numerical amount of time he/she spent for the reading, (ii) a numerical percentage regarding the percentage of stuff in the reading actually read and understood, and (iii) whether the student has come to the class this week.

Reflection on the reading (2 points):
The student need to put down 1 to 2 paragraphs of his/her thoughts about the reading such as new insight you gained, interesting things encountered, questions of things you don't understand, and so forth.

Assessment:
For the effort part,
the student is expected to **(i)** have attended the class this week at least once (0.5 point), and **(ii) have either** gained a good understanding of 80% or more of the contents **or** have spent at least three hours in the reading (1.5 points).

For the reflection part,
the student is expected to show substantial evidence of understanding or effort of trying to understand the contents in the reading.

Task 2: Programming Assignments (10 assignments)

Due Date: Monday of the week
Weighting: 30%
Possible Points: 6 points each.

Description: There will be about 9 programming assignments, which form the backbone of the course. They require the student to incrementally develop programming skills based on the concepts of data structures and object-oriented programming learned in the class. You need to submit a peer review report together with all your source code files for each assignment. In the self-evaluation report, you should describe results from sufficient test cases that are verified by a peer reviewer.

Integrity rules for programming assignments and programming tests:

- **Peer discussion is encouraged**: Peer discussion is encouraged to cultivate an open learning environment in the class, but you should carefully read the guidelines below to avoid any dishonest behavior and never step over the guidelines explicitly described in the following.
- **Never use code written by others**: Any copy-and-paste of code from other people's programs or from websites is viewed as cheating and you will get 0 points for the assignment.
- **Never circulate your code to others**: Peer discussion of code shown on the screen is acceptable for debugging purpose and for explanation of ideas. But you should never pass around your code (electronically or on paper) to others except for the TA and the instructor. Violating this rule is viewed as cheating in the class and the provider will receive 0 points for the assignment.
- **Never provide false or exaggerated results of test cases**: You need to submit a peer review report together with all your source code files for each assignment**.** Providing false or exaggerated results of test cases in the report is viewed as cheating and you will receive 0 points for the assignment.
- **Demonstrate the credibility of your authorship of the work**: When you submit your code as your own work for points, you should make sure that you are able to explain your code and reconstruct your code from scratch without any outside help when requested. If you are not able to do that on your own when requested, you will get 0 points for the assignment and there will be an investigation.
- **Consequence of cheating in the class**: Cheatings end in 0 points for the assignments followed by discipline actions described in the student handbook.

Assessment: The student needs to submit (all related .cpp and .h files) together with a self-evaluation report to the TA. In the self-evaluation report, you should describe results from sufficient test cases that are verified by a peer reviewer. We'll grade each programming assignment in a 0-6 scale based on the following rubric.

1. Source code is completed but the code fails to compile successfully.
2. Source code can compile and do something required, but has serious bugs or miss a couple of key features.
3. Source code can compile and do most of the features required, but has many minor bugs or miss a key required feature.

4. Source code can compile and do all the features required, nearly fully functional, only a couple of minor bugs.
5. Source code can compile and do all the features required, fully functional, no bugs.
6. In addition to the points received above, get one more point if (i) in the self-evaluation report, the student does describe sufficient test cases and results that are verified by a peer reviewer and (ii) the source code is well indented and commented to make it visually very readable.

Task 3: Final Project: 40%

Specification: Design and implement an interactive 3D virtual world using the C++ programming language and OpenGL application programming interface for 3D graphics, in which the user can explore and interact with the virtual world.

Assessment: **Basic technical requirements (7 points)**

You can add as many bells and whistles into your final project as you want, but the following are the basic requirements:

- **Requirement A1 (Complexity of the environment)**: It must render scenes of 3D environment. For example, you can model your room, an area of the campus such as the bell tower or the fountain neighborhood, the computer science alcove, or other places of significant complexity. There should be at least **twenty** objects in the scene (for example, desks, chairs, floors, walls, ceiling, lamps, books, bookshelves, windows, curtains, hills outside the windows, doors, closets, teapots, cups, vases, vessels,…)
- **Requirement A2 (Dynamics in the environment)**: Your virtual environment should have at least **two** mobile objects that either automatically move around the environment all the time or will be triggered into dynamic movement when the user interact with the environment in some way. For example, you may have a toy vehicle or a wumpus monster automatically moving around all the time, or you may have the toy vehicle or wumpus monster escaping from user when the user is close to it in the virtual world.
- **Requirement A3 (Rewarding mechanism in the environment)**: Your virtual environment should have a mechanism for rewarding or encouraging the user for exploring the environment. If you want, you can well craft the rewarding mechanism to create an interesting 3D game essentially. For example, the user may receive a number of points to begin with, finding an energy charge may gain points, while jumping over an obstacle or driving away a monster by firing a shot may consume certain some points. Or for simplicity, you focus more on the modeling of the photo-realistic aspect of virtual environment and just include a couple of simple incentives like finding a hidden treasure into the environment .
- **Requirement A4 (Lighting)**: You must have at least one light source in the environment, and set different material properties to some of the geometric objects to create variation of colors, shininess, and or other attributes.
- **Requirement A5 (Texture mapping)**: You must apply texture mapping to some surfaces. For example, you may consider texture mapping used on the ground, the floor, the walls, the ceiling, the desks, and so forth.
- **Requirement A6 (Automatic bird-eye view of the environment)**: You must provide some mechanism of automatic movement of the camera (for example, rotation of camera around a

circle or a sphere while looking at the center of the environment) to provide an easy dynamic overview of the environment.

- **Requirement A7 (User interface supporting manual walk-through)**: You must provide a convenient user interface (through mouse and/or keyboard operations) for the user to manually move around (i.e. change the location of the camera) and to look at the environment in flexible ways (for example be able to change the focus point they look at, to zoom in and room out by changing the perspective angle).

Assessment: **Storyline (6 points):**

We will evaluate storyline of the virtual world project based on how well you design the patterns of interaction between the user and your virtual world to make up a story attracting the user to engage in the game.

i. The events of interaction should be related to one another in a coherent way allowing smooth transition between scenes you create in the virtual world following storyline. (3 points)
ii. The storyline should provide an interesting cause-and-effect relationship between the events and introduces incentives attracting the user to interact with the virtual environment for reward.

Modeling (3 points):

We will evaluate modeling of the virtual world project based on how well you model the objects in the virtual world and render them in a visually appealing way. You should craft the objects in the virtual world with fine details and pleasant lighting and texture settings, and provide convenient ways for the user to navigate in the virtual world and explore the details of the virtual world.

Task 4: Exams (Tests)

Weighting: 20%

Description: Each test may have a written component, or a programming component, or both. The written component mainly tests the conceptual understanding of data structures while the programming component tests skills in object-oriented programming.

Assessment: The written component will be graded based on the answers provided while the programming component will be graded based on the same rubric provided for the weekly programming assignments.

CLASS INFORMATION

Late policy

- **No submission accepted after the submission site on Canvas is closed**: All submissions should be done through the Biola Canvas system. No submission will be accepted after

the submission site on Canvas is closed, except for extremely exceptional situations such as a serious disabling health problem with evidence from the doctors.

- **Penalty for late submission after the due date but before the submission site is closed**:
  For a programming assignment,
  the submission site on Canvas may remain open for 2 more days and 1 point will be deducted from the work for a late submission before the submission is closed.
  For a reading report,
  the submission site on Canvas may remain open for 6 more days and 1 point will be deducted from the report for a late submission before the submission is closed.

Computation of Final Grade:

Weekly Progress Report	10 %
Weekly Programming Assignments	30%
Final Project	45%
Exams	15 %
Total	**100%**

Final grades will be awarded on the following point system:

A	93%
A-	90%
B+	87%
B	84%
B-	80%
C+	77%
C	74%
C-	70%
D+	67%
D	64%
D-	60% to pass class

Tentative Schedule

Weeks 1~7

Introduction

Overview of graphics systems, graphics devices, graphics programming.

Graphics Programming

OpenGL, graphics primitives, 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.

Week 8: Review & Midterm

Weeks 9~15

Using Images

Texture-, bump-, and reflection-mapping.

Implementation Issues

Rasterization, clipping.

Ray tracing

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

Hidden surface removal

Back-face culling, z-buffer method, depth-sort.

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.