Advanced Computer Graphics
CSCI B581 - Spring 2017 - course section 30656

Textbooks

Recommended: "OpenGL ES 2.0 Programming Guide" by Aaftab Munshi, Dan Ginsburg, Dave Shreiner - Addison-Wesley 2009

Spring 2017 Topics:
3D & 2D Computer Graphics Programming with OpenGL ES on Android and iOS.

B.Y.O.D. and code!

In Spring 2017, we are going to use OpenGL ES and GLSL to write and run graphics programs on the GPU of mobile devices*: if you have a Android or iOS device (phone, tablet, etc) running a recent version of the operating system, you may test your programs on it: just Bring Your Own Device and code...

(*) bringing your own device is not required: you may complete B581 just as successfully by writing and running the same graphics programs on mobile device simulators that run on STC lab computers, or on your own computer.

Course Description:
• Introduction to graphics hardware and software.
• Two-dimensional graphics methods, transformations, and interactive methods.
• Three-dimensional graphics, transformations, viewing geometry, object modeling and interactive manipulation methods.
• Basic lighting and shading.
• Video and animation methods.
• Credit not given for both B581 and B481.

Course Prerequisites:
Required: [CSCI-C 343 and (MATH-M 301 or M 303)], or equivalent experience: data structures & mathematics for vectors and matrices
Recommended: [CSCI-C 212 and a working knowledge of a C-like language], or equivalent programming experience. We’ll use the GLSL programming language for GPU programming, and there will be "refresher" lectures on linear algebra and geometry.
Course Objectives:

Course Goals:
B581 is the main course in computer graphics for Computer Science graduate students. Graphics is a crucial part of many interfaces and applications. A basic knowledge of computer graphics (CG) is useful for any task that involves a computer. This course concentrates on:

- fundamental principles of displaying 2D information using computer graphics
- 3D programming for geometry modeling, transformation, viewing, lighting and shading
- interactive techniques for real-time computer graphics
- computer graphics for mobile devices

Learning Objectives:
By the end of this semester, students should be able to:

- understand the fundamental stages and mathematical foundation of the 3D polygon projection rendering pipeline
- develop interactive 2D and 3D graphics applications using OpenGL ES and GLSL
- describe the relative advantages and disadvantages of low-, middle-, and high-level computer graphics
- identify state-of-the-art trends in graphics hardware, display devices, and rendering methods

Additional considerations
There will be approximately six programming assignments, and approximately six written-question homework assignments. For programming assignments, you will be writing native code for iOS (in Swift) or Android (in Java) to program the OpenGL ES API, and GLSL for GPU-based code. Other programming languages, APIs or OSes will not be considered for assignments submitted for grading.

Course Policies
Students are expected to know follow the Code of Student Rights, Responsibilities, and Conduct http://www.iub.edu/~code/, in particular the section about academic misconduct and its application to programming courses. The standard penalty for any form of academic dishonesty in a course is failure of the course, i.e. an F grade for the course. Outside of assigned student-team work, providing or receiving help or submitting the work of another as your own constitutes academic dishonesty. There are no "small" offenses.
Homework and programming assignments are to be performed and completed individually, unless otherwise directed: no collaborative work. Students may not use/reuse anyone else’s program code, including code you may find on the internet. We will regularly check assignments against code from classmates and code on the internet. Cheating will result in an F for the class.
Exams and Grades

- Final Exam: 25%
- Midterm: 15%
- Coding assignments and projects: 40%
- Readings, written assignments, in-class presentations: 15%
- Participation, active class presence and journal note-taking: 5%

B581 coursework includes about six programming assignments, six written homework assignments, a written midterm exam and a written final exam. Midterm and final are traditionally pencil-and-paper, closed books exams. There are also about 6 programming assignments, which form an essential part of the course and are to be completed individually. All B581 assignments are written using the OpenGL ES API and GLSL.

Course Topics

*Note: the following outline is a superset of what we’ll actually have time to cover during the semester. It will be refined.* Study guides for mid-term and final exams will specify which material will be tested, such as amount of details and emphasis on specific topics.

**Introduction:**

**Two-Dimensional Concepts:**

2D geometry, vectors, vector products, normals, matrix transformations, and explicit coordinate transformations. Intersection calculations. Know operations and transformations on 2D vectors. Affine transformations, their properties and inverse transformations. Translation, rotation, scaling and shearing in 2D. Explicit equations. Homogeneous coordinates. Matrix representations, the “fixed point rule” not just for displaced fixed point but also rotated coordinate axes. How to manipulate 2D objects in world coordinate system and screen coordinate system; going from "ideal" to world coordinate systems. Line-line and point-line geometry - how to find nearest point. Inside-outside tests. Signed area measurements.

Clipping. Basic motivation and definitions; why is it important? Equations needed to clip lines and polygons. Basic principles used in standard line and polygon clipping, including Cyrus-Beck and Cohen-Sutherland algorithms.

Curves and Splines. Basic properties and ideas. Implicit, explicit and parametric representations of lines and curves. Principles of constructing splines with higher derivatives from linear interpolations (actually linear interpolation is just the simplest spline). Qualitative ideas of matching derivatives on successive curve segments. Contrast Bezier, Catmull-Rom, B-spline cubic spline properties (no need to memorize explicit coefficients). Basis functions.
Scan conversion of lines. Fundamental implicit and parametric equations. Fundamental properties of scan conversion algorithms (integer arithmetic, local recursion equations), construction of the algorithms. Error measure concept and how you use it.

**Three-Dimensional Concepts:**

Math and Geometry for 3D Graphics. 3D vectors, homogeneous coordinates. 3D Transformations - Translation, Scaling, Rotation, Reflection, Shear. Composite Transformations.

3D shape representation. 3D surfaces - polygon, curved (Bezier, spline), surface generation methods.


Hidden Surface and Line Removal Methods.

3D Pipeline and Mid-level APIs. GLSL vs. OpenGL. Special Effects Buffers. API-specific operations. GPU-specific operations.

Light, Surfaces, Scenes and Animation. Color: how are specific colors constructed; additive and subtractive color models; metamers.

Modeling Light and Shading. Light sources and ambient light. Diffuse and specular reflection, refraction. Texture and surface patterns.


High-level APIs. Historical examples. Scene Graph Scripting.

**Advanced Topics:**


Mitja Hmeljak 2016-12-12