Computer Graphics
CS 4731 – Midterm Review

Prof Emmanuel Agu

Computer Science Dept.

Worcester Polytechnic Institute (WPI)

Exam Overview

- Thursday, February 7, 2013 in-class
- Will cover up to lecture 14 (projection)
- Can bring:
  - One page cheat-sheet, hand-written (not typed)
  - Calculator
- Will test:
  - Theoretical concepts
  - Mathematics
  - Algorithms
  - Programming
  - OpenGL/GLSL knowledge (program structure and some commands)

What am I Really Testing?

- Understanding of
  - concepts (NOT only programming)
  - programming (pseudocode/syntax)
- Test that:
  - you can plug in numbers by hand to check your programs
  - you did the projects
  - you understand what you did in projects

General Advise

- **Read your projects** and refresh memory of what you did
- **Read the slides**: worst case – if you understand slides, you're more than 50% prepared
- Focus on **Mathematical results, concepts, algorithms**
- Plug numbers: calculate by hand
- Try to **predict subtle changes** to algorithm.. What ifs?..
- **Past exams**: One sample midterm is on website
- All lectures have references. Look at refs to focus reading
- Do all readings I asked you to do on your own

Grading Policy

- I try to give as much partial credit as possible
- In time constraints, laying out outline of solution gets you healthy chunk of points
- Try to write something for each question
- Many questions will be easy, exponentially harder to score higher in exam

Introduction

- Motivation for CG
- Uses of CG (simulation, image processing, movies, viz, etc)
- Elements of CG (polylines, raster images, filled regions, etc)
- Device dependent graphics libraries (OpenGL, DirectX, etc)

OpenGL/GLUT

- High-level:
  - What is OpenGL?
  - What is GLUT?
  - What is GLSL
  - Functionality, how do they work together?
- Design features: low-level API, event-driven, portability, etc
- Sequential Vs. Event-driven programming
- OpenGL/GLUT program structure (create window, init, callback registration, etc)
- GLUT callback functions (registration and response to events)

OpenGL Drawing

- Vertex Buffer Objects
- glDrawArrays
- OpenGL :
    - Drawing primitives: GL_POINTS, GL_LINES, etc (should be conversant with the behaviors of major primitives)
    - Data types
    - Interaction: keyboard, mouse (GLUT_LEFT_BUTTON, etc)
    - OpenGL state
- GLSL Command format/syntax
- Vertex and fragments shaders
- How GLSL works

2D Graphics: Coordinate Systems

- Screen coordinate system/Viewport
- World coordinate system/World window
- Setting Viewport
- Tiling, aspect ratio

Fractals

- What are fractals?
  - Self similarity
  - Applications (clouds, grass, terrain etc)
- Mandelbrot set
  - Complex numbers: s, c, orbits, complex number math
  - Dwell function
  - Assigning colors
  - Mapping mandelbrot to screen
- Koch curves, gingerbread man, hilbert transforms

Points, Scalars Vectors

- Vector Operations:
  - Addition, subtraction, scaling
  - Magnitude
  - Normalization
  - Dot product
  - Cross product
  - Finding angle between two vectors
- Standard unit vector
- Normal of a plane

Transforms

- Homogeneous coordinates Vs. Ordinary coordinates
- 2D/3D affine transforms: rotation, scaling, translation, shearing
- Should be able to take problem description and build transforms and apply to vertices
- 2D: rotation (scaling, etc) about arbitrary center:
  - $T(Px,Py)\ R(\theta)\ T(-Px,-Py) * P$
- Composing transforms
- OpenGL transform commands (Rotate, Translate, Scale)
- 3D rotation:
  - x-roll, y-roll, z-roll, about arbitrary vector (Euler theorem) if given azimuth, latitude of vector or (x, y, z) of normalized vector
- Matrix multiplication!!

Modeling and 3D Viewing

- Drawing with Polygonal meshes
- Finding vertex normals
- Lookat(Eye, COI, Up) to set camera
  - How to build 3 new vectors for axes
  - How to build world-to-eye transformation
  - Pitch: nose up-down
  - Roll: roll body of plane
  - Yaw: move nose side to side

Projection

- Projection:
  - View volume, near plane, far plane
  - Perspective(fovy, aspect, near, far) **or**
  - Frustum(left, right, bottom, top, near, far)
  - Ortho(left, right, bottom, top, near, far)
  - How to build Perspective and Ortho matrices