Overview

- Course Policies/Issues
- Brief History of Computer Graphics
- The Field of Computer Graphics: A view from 66,000ft
- Structure of this course
- Homework overview
- Introduction and discussion of homework #1

Computer Graphics I:

Course Goals

- Provide introduction to *fundamentals* of 2D and 3D computer graphics
  - Representation (lines/curves/surfaces)
  - Drawing, clipping, transformations and viewing
  - Implementation of a basic graphics system
    - simple frame buffer with XPM format
    - ties together 3D projection and 2D drawing

Interactive Computer Graphics

- Now offered every year
- Learn and program OpenGL
- Computer Graphics I was a pre-requisite
- Useful for Games classes
  - Most final projects are games
- Part of the HCI track

Advanced Rendering Techniques

- Not sure when it will be offered again
- 3D Computer Graphics
- Ray tracing algorithm
- Lighting, rendering, photorealism
- Bezier & polygonal meshes
- Hierarchical models

New Graphics & Vision Track?

Take CS 430 & CS 435 and one other

- CS 430 Computer Graphics *
- CS 431 Advanced Rendering Techniques
- CS 432 Interactive Computer Graphics
- CS 435 Computational Photography *
- Can “double-up” 432 with HCI Track
  * Required
ART Student Images

Computer Graphics I: Technical Material

- Course coverage
  - Mathematical preliminaries
  - 2D lines and curves
  - Geometric transformations
  - Line and polygon drawing
  - 3D viewing, 3D curves and surfaces
  - Splines, B-Splines and NURBS
  - Solid Modeling
  - Color, hidden surface removal, Z-buffering

Computer Graphics I: Course Highlights

- Bresenham’s scan conversion algorithm
- Cohen-Sutherland clipping algorithm
- Sutherland-Hodgman polygon clipping
- The De Casteljau Algorithm
- Polygon filling
- B-Splines, NURBS, De Boor’s Algorithm
- Z-buffer algorithm; backface culling

Computer Graphics I: Course Management Issues

- All course policies are in the syllabus
- Extensive use of PDF handouts
- Must read email every day
- There will be 6 programming assignments
  (plan on 8-to-15 hrs)
- Suggestion: print out handouts before class, use them to take notes
- Grad students submit a paper presentation
- Final exam on material not covered by the programming assignments
- READ THE SYLLABUS!!

Computer Graphics I: Collaboration Policies

- Thou Shall
  - write your own code
  - do your own math
  - attribute any work that is not your own
  - talk amongst yourselves, share ideas
- Thou Shall Not
  - Share/copy code
  - Use ideas without attribution
  - Utilize geometry/graphics libraries
- All code will be auto checked for plagiarism
- Violations will result in an automatic F

Go to class web page
CG Technical Areas

• Geometric Modeling
  – Mathematics and algorithms that define 2D and 3D geometric objects

• Human/Computer Interaction
  – Methods for creating graphics data via user input

Surface Drawing, Steven Schkolne

CG Technical Areas

• Lighting and Shading
  – Math, physics and algorithms that specify how light interacts with matter

• Rendering
  – Algorithms that take geometry, lighting, shading and viewing information and generate an image

CG Technical Areas

• Visualization
  – Techniques for visually communicating and exploring scientific, medical or abstract data

CG Technical Areas

• Perception
  – Study of how humans perceive light and information
CG Technical Areas

• Animation
  – Algorithms for making models change over time

CG Technical Areas

• Simulation
  – Using physics to make models move

CG Technical Areas

• Software and Hardware
  – Designing software and hardware systems to implement graphics algorithms

Computer Graphics: A Brief History

• In The Beginning...
  1963
  Ivan Sutherland’s Sketchpad
  Modified oscilloscope for drawing
  The original CAD system

Graphics Hardware History

• Good summary of the development of the graphics processor
  – http://accad.osu.edu/~wayne/history/lesson15.html

• Part of an excellent on-line history of computer graphics and animation
  – http://accad.osu.edu/~wayne/history/lessons.html

Computer Graphics from 66,000ft

• Display types
• Display/Rendering algorithms
• Application areas
  – Entertainment
  – CAD/CAM
  – Scientific & medical visualization
  – Training & education
  – Synthetic realities
  – Art and design
  – Games
2D Graphics

**Raster:**
- Pixels
  - X11 bitmap, XBM
  - X11 pixmap, XPM
  - GIF
  - TIFF
  - PNG
  - JPG

Lossy, jaggies when transforming, good for photos.

**Vector:**
- Drawing instructions
  - Postscript
  - CGM
  - Fig
  - DWG

Non-lossy, smooth when scaling, good for line art and diagrams.

Adobe Photoshop:
2D Raster Graphics

Adobe Illustrator:
2D Vector Graphics

2D Raster Graphics

2D Vector Graphics
3D Rendering

- 1960s - the visibility problem
  - Roberts (1963), Appel (1967) - hidden-line algorithms
  - Sutherland (1974) - visibility = sorting

- 1970s - raster graphics
  - Courtois (1971) - diffuse lighting
  - Pfyong (1974) - specular lighting
  - Blinn (1974) - curved surfaces, texture
  - Crow (1977) - anti-aliasing

Toward Reality in the 1980s

- global illumination
  - Whitted (1980) - ray tracing
  - Goral, Torrance et al. (1984), Cohen (1985) - radiosity
  - Kajiya (1986) - the rendering equation

- photorealism
  - Cook & Torrance (1982) - rough surface reflectance
  - Cook (1984) - shade trees
  - Perlin (1985) - shading languages
  - Hannah and Lawson (1990) - RenderMan

Away from Reality

- early 1990s - non-photorealistic rendering
  - Drebin et al. (1988), Levoy (1988) - volume rendering
  - Haebler (1990) - impressionistic paint programs
  - Salesin et al. (1994) - automatic pen-and-ink illustration
  - Meier (1996) - painterly rendering

And Back Again

- late 1990s & 2000s - photon mapping, subsurface scattering and participating medium
  - H. Wann Jensen

Application Areas

- Entertainment
- CAD/CAM
- Scientific & Medical visualization
- Training & Education
- Synthetic Realities
  - VR, AR, etc.
- Art and design
- Games

Pixar
Lord Of the Rings Troll

Application Areas

- Entertainment
- **CAD/CAM**
- Scientific & Medical Visualization
- Training & Education
- Synthetic Realities – VR, AR, etc.
- Art and design
- Games

Regli et al @ Drexel

Application Areas

- Entertainment
- **CAD/CAM**
- Scientific visualization
- Training & Education
- Synthetic Realities – VR, AR, etc.
- Art and design
- Games

Hamburg U, Germany

Application Areas

- Entertainment
- **CAD/CAM**
- Scientific visualization
- Training & Education
- Synthetic Realities – VR, AR, etc.
- Art and design
- Games

FakeSpace Cave

Application Areas

- Entertainment
- **CAD/CAM**
- Scientific visualization
- Training & Education
- Synthetic Realities – VR, AR, etc.
- Art and design
- Games

Kimmel Center
Application Areas

- Entertainment
- CAD/CAM
- Scientific visualization
- Training & Education
- Synthetic Realities – VR, AR, etc.
- Art and design
- Games

Programming Assignments

- No APIs: OpenGL, GLUT, Mesa, DirectX...
- Just line and filled polygon rendering
  - Color in last assignment
- Output in the form of 2D bitmaps
  - XPM: X11Pixmap Format
- Program source (and makefile) turned in via Bb Learn
- Executable MUST RUN on Linux (tux)
- Whatever language you want, so long as you can deliver a program that TA can run (c, c++, java, ...)

For programming assignments

- Use XPM as B/W “software” frame buffers
- We will be implementing parts of the 2D Engine, 3D Engine and Pixel Cache of a graphics accelerator
- Issue: How to translate the mathematics of 2D/3D shapes to the 2D screen?
- Tip: Renew your friendship with your linear algebra textbook
- Read homeworks ahead. It will help you to structure your code for future requirements.

Assignment Dependencies

- Every HW - Read in geometry and write images
- HW1 - Clip 2D lines and draw them
- HW2 - Clip 2D polygons and draw edges with HW1
- HW3 - Use HW2 to clip 2D polygons and fill in interiors
- HW4 - Project 3D polygon edges (3D lines) into 2D. Draw them with HW2
- HW5 - Project 3D triangles into 2D. Combine HW3 and HW4 and extend visible surface calculation
- HW6 – Generate lines and draw with HW1

Programming Assignments Dependencies

<table>
<thead>
<tr>
<th>HW</th>
<th>HW 1</th>
<th>HW 2</th>
<th>HW 3</th>
<th>HW 4</th>
<th>HW 5</th>
<th>HW 6</th>
</tr>
</thead>
<tbody>
<tr>
<td>HW1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HW2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HW3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HW4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HW5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HW6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Data Structures

- Initially B/W, then RGB Frame Buffer
- 2D Lines
- 2D/3D polygons (vertices and edges)
- Triangle mesh
- 3D camera/scene
When it’s all over!

Don’t forget ICG & ART!

Programming assignment 1
• Input PostScript-like file
• Output B/W XPM
• Primary I/O formats for the course
• Create data structure to hold points and lines in memory (the world model)
• Implement 2D translation, rotation and scaling of the world model
• Implement line drawing and clipping
• Due January 20th
• Get started now!