Introduction to Computer Graphics

Nicolas Holzschuch
University of Cape Town
e-mail: holzschu@cs.uct.ac.za
Uses of Computer Graphics

• Graphical user interfaces
  – mouses, windows

• Virtual Reality
  – input: gloves, head-tracker
  – output: simulation

• Digital Media Technologies
  – digital publishing
  – multi-media
Uses of Computer Graphics (2)

• Visualization
  – medicine
  – financial data

• Simulation
  – engineering, GIS, architecture,…

• Entertainement
  – Movies (Toy Story, Jurassic Park, …)
  – Games (Doom, Marathon, Descent…)

Interdisciplinary

• Theory:
  – Mathematics of curves, surfaces, matrices
  – Physics of lights, materials, colors

• Practice:
  – Hardware: display, processors
  – Software: graphics libraries, windows

• Aesthetics: how does it look?
What’s in this course?

• OpenGL for the practicals
• Basic Computer Graphics:
  – what everybody must know before doing Computer Graphics.
• Advanced Computer Graphics:
  – more advanced topics, demand-driven.
When is the course?

• Video-conferencing, Thursdays at 17h30
• Drill session, 26-27 March (12 hours)
• ≈ 25 lectures (3+12+10)
• Slides and course information on the WWW:

  http://www.cs.uct.ac.za/~holzschu/Pretoria.html
The Textbook

• Foley, van Dam, Feyner et al.

• Two textbooks:
  – one is a reference for CG.
  – the other is an abridged version, easier for beginners.

• Which should I buy?
  – the small one is enough, cheaper and easier
  – except if you are going to work in CG for several more years.
OpenGL for the practicals

- Graphics Library
- Multi-platform
  - runs on SGI and on all X11 machines
  - also on Windows 32
  - gets all the power of the SGI
- Easy to use
  - one or two lectures of introduction
Basic Computer Graphics

• Graphics Primitives
  – rasterization, clipping
• Transformations
• Hierarchical Modelling
• Color Spaces and transformations
• Hidden Surface Removal
• Lighting Models
Advanced Computer Graphics

- Demand-driven
- Curves and surfaces (splines, Bézier)
- Texture-mapping
- Quaternions
- Radiosity and ray-tracing
The Practicals

• Subject available on the WWW: http://www.cs.uct.ac.za/~holzschu/Pretoria.html
• Collective e-mail when subject is ready
• Send your prac's by e-mail on due date
• 2 practicals, three-four weeks per prac.
• Subjects tend to be writing an application, with progressive steps
• Suggestions?
The Marks

- 2 practicals, one final exam
- Final exam is 2 hours, 5 exercises
- Pick 3 out of 5
- Open-book exam, lots of thinking
- Probably 0.35 for each practical, 0.3 for the exam
Life after the exam

• Hey, I want to do more Computer Graphics!
• Masters studies
  – in SA: UCT, Rhodes,…
  – abroad: France, US, Germany…
• Siggraph Student Volunteers Program:
  http://www.siggraph.org
Framework of Computer Graphics

• Graphics Pipeline:

Application Model → Application Program → Graphics System → Output:

Input: Keyboard, Mouse → Display
Application Modeling

- Description of the objects
  - shape
  - attributes
  - behaviour and properties
- Stored in the application memory
- Can be modified by the application
- Sent to display
Display of the Model

- From internal representation to standard graphics primitives
  - lines, circles, polygons
  - geometric transformations
  - using the graphic library

- Interactions:
  - event-driven loop
  - queue of events to process
Graphics Primitives

• Pixels (=picture elements)

• Screen coordinates

(6,5)
Screen Resolution

• Number of pixels per square cm
  – Measured in dots per inch (dpi)

• Number of addressable colors per pixel
  – measured in bits

• Depends on the medium:
  
  TV Screen: 30 dpi, 8 bits color
  Computer Screen: 70-100 dpi, up to 24 bits color
  Laser Printer: 300-2400 dpi, 3 bits color (8 colors)
  Photo: ≈ 800 dpi, 36 bits color
Simple Graphics Primitives

- **Lines:**
  
  ```c
  void LineCoord(int xmin, int ymin, int xmax, int ymax);
  void Line(point pt1, point pt2);
  ```

- **Polygons:**
  
  ```c
  void polyLine(int count, point* vertices);
  void polygon(int count, point* vertices);
  ```

- **Circles and ellipses:**
  
  ```c
  void circle(point center, int radius);
  void ellipse(point center, int radiusX, int radiusY);
  void ellipseArc(point center, int radiusX, int radiusY, 
                   float startAngle, float endAngle);
  ```
Filled Graphics Primitives

• Same primitives, but filled:
• Filling can be:
  – solid vs bitmap pattern
  – opaque vs transparent
• Where is the interior?

Odd-even rule

Edge orientation rule
Clipping Graphics Primitives

- Drawing outside the window:
- Graphics have to be clipped.
Rasterization of Primitives

• How to draw primitives?
  – Convert from geometric definition to pixels
  – *rasterization* = selecting the pixels

• Will be done frequently
  – must be fast:
    • use integer arithmetics
    • use addition instead of multiplication