MCS 585/480
Computer Graphics I

Introduction

Week 1, Lecture 1

William Regli and Maxim Peysakhov
Geometric and Intelligent Computing Laboratory
Department of Computer Science
Drexel University
`http://gicl.mcs.drexel.edu`

Overview

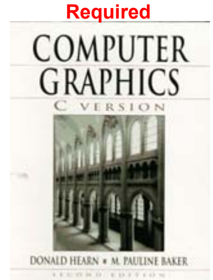
- Course Policies/Issues
- Brief History of Computer Graphics
- The Field of Computer Graphics:
  A view from 66,000ft
- Structure of this Course
- Homework overview

Computer Graphics I: Course Goals

- Provide introduction to ***fundamentals*** of 2D and 3D computer graphics
  - Representation (lines/curves/surfaces)
  - Drawing and viewing
  - Implementation
    - simple frame buffer with XPM format
    - Lines, circles, Bezier and NURBS curves
    - Drawing algorithms

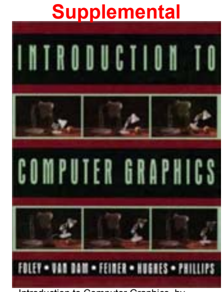
Book

Required



Computer Graphics, C Version, Donald Hearn, M. Pauline Baker, 2nd Edition, Prentice Hall, 1997, ISBN: 0135309247

Supplemental

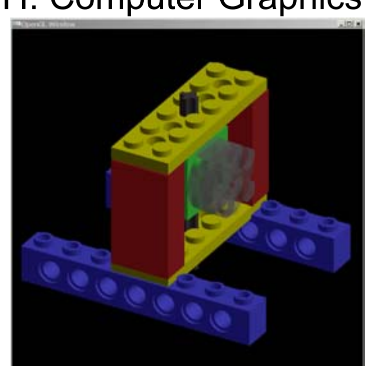


Introduction to Computer Graphics, by James D. Foley, Andies Van Dam, Steven K. Feiner. Addison-Wesley Pub Co; ISBN: 0210609215

FYI: Computer Graphics II

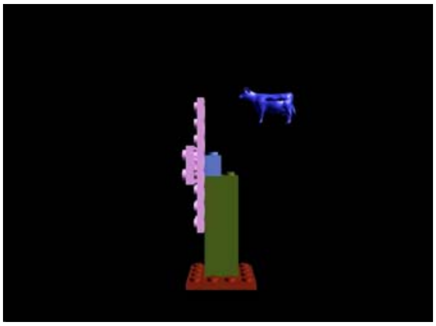
- 3D Computer Graphics
- All OpenGL API
- Lighting, rendering, photorealism
- Complex geometry & polygon soups
- Hierarchical models
- Simulation, animation, collision detection

FYI: Computer Graphics II



Animated by Max Peysakhov @ Drexel University

FYI: Computer Graphics II



Animated by Dmitriy Bespalov @ Drexel University

Computer Graphics I: Technical Material

- Course coverage
  - Geometric transformations, 2D raster graphics, 2D viewing, 2D curves
  - 3D viewing, 3D curves and surfaces
  - Splines, B-Splines and NURBS
  - 3D drawing, hidden line removal, Z-buffering, ray tracing
  - Solid Modeling
- Chapters: 1, 5, 3, 6, 9, 10, 11, 12, 13

Computer Graphics I: Course Highlights

- Bresenham's scan conversion algorithm
- Cohen-Sutherland clipping algorithm
- The De Casteljau Algorithm
- B-Splines, NURBS, De Boor's Algorithm
- Z-buffer algorithm; backface culling
- Ray casting/tracing of 3D shapes and surfaces

Computer Graphics I: Course Management Issues

- All course policies are in the syllabus
- Extensive use PDF handouts
- Must read email list every day
- No cell phones, No AIM
- There will be programming homework each week (*plan on minimum of 8-to-10 hrs*)
- Suggestion: print out handouts before class, use them to take notes
- ***READ THE SYLLABUS!!***

Computer Graphics I: Collaboration Policies

- Thou Shall
  - write your own code
  - do your own math and proofs
  - attribute any work that is not your own
  - talk among yourselves, share ideas
- Thou Shall Not
  - Share code or proofs
  - Use ideas without attribution
- All code will be auto checked for plagiarism
- Violations will result in an automatic ***F***

Computer Graphics: A Brief History

- In The Beginning… 1963 Ivan Sutherland's ***Sketchpad***
- Modified oscilloscope for drawing
- The original CAD system

Courtesy Marc Levoy @ Stanford

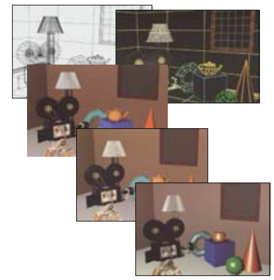
Graphics Hardware History

Display Hardware
- vector displays
  - 1974 – E&S Picture System
- raster displays
  - 1975 – E&S frame buffer
  - 1980s – cheap frame buffers → bit-mapped personal computers
  - 1990s – liquid-crystal displays → laptops
  - 2000s – micro-mirror projectors → digital cinema
- Other developments
  - stereo, head-mounted displays
  - autostereoscopic displays
  - tactile, haptic, sound

Input Hardware
- 2D
  - light pen, tablet, mouse, joystick, track ball, touch panel…
  - 1970s & 80s - CCD analog image sensor + frame grabber
  - 1990s & 2000's - CMOS digital sensor + in-camera processing → high-X imaging (dynamic range, resolution, depth of field,…)
- 3D
  - 3D trackers, 3D scanners
  - multiple cameras
  - active rangefinders
- other
  - data gloves
  - voice

Rendering

- 1960s - the visibility problem
  - Roberts (1963), Appel (1967) - hidden-line algorithms
  - Warnock (1969), Watkins (1970) - hidden-surface algorithms
  - Sutherland (1974) - visibility = sorting
- 1970s - raster graphics
  - Gouraud (1971) - diffuse lighting
  - Phong (1974) - specular lighting
  - Blinn (1974) - curved surfaces, texture
  - Catmull (1974) - Z-buffer hidden-surface algorithm
  - Crow (1977) - anti-aliasing

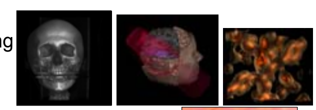


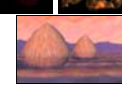
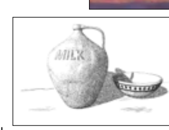
Rendering

Toward Reality
- early 1980s - global illumination
  - Whitted (1980) - ray tracing
  - Goral, Torrance et al. (1984), Cohen (1985) - radiosity
  - Kajiya (1986) - the rendering equation
- late 1980s - photorealism
  - Cook (1984) - shade trees
  - Perlin (1985) - shading languages
  - Hanrahan and Lawson (1990) - RenderMan

Present Developments

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



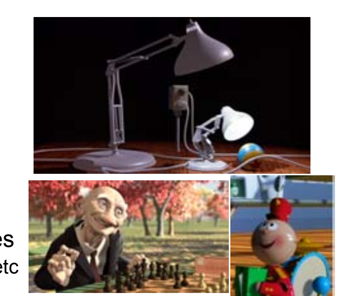
Computer Graphics from 66,000ft

- User Interaction/GUI
- 2D graphics and image processing
- 3D graphics/modeling
- Animation/Simulation
- Photorealism
- Virtual Reality
- Graphics Hardware
- Graphics programming environments

Example of a JPG image.

Application Areas

- ***Entertainment***
- CAD/CAM
- Scientific & Medical visualization
- Training & Education
- Synthetic Realities
  - VR, commerce, etc
- Art and design

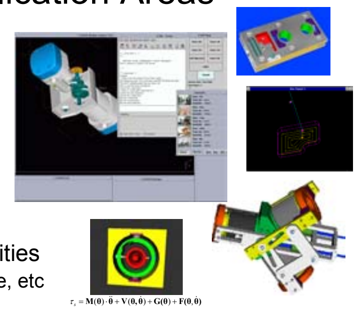


Pixar

Lord Of the Rings Troll

Application Areas

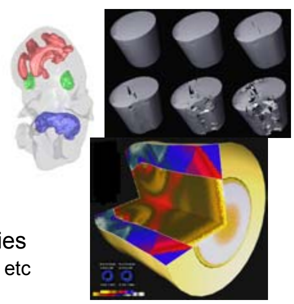
- Entertainment
- ***CAD/CAM***
- Scientific & Medical visualization
- Training & Education
- Synthetic Realities
  - VR, commerce, etc
- Art and design



Regli et al @ Drexel

Application Areas

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

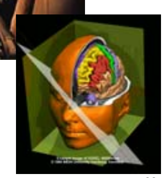


S. Lombeyda @ CalTech

Application Areas

- Entertainment
- CAD/CAM
- Scientific visualization
- ***Training & Education***
- Synthetic Realities
  - VR, commerce, etc
- Art and design

Boeing



Hamburg U, Germany

Application Areas

- Entertainment
- CAD/CAM
- Scientific visualization
- Training & Education
- ***Synthetic Realities***
  - VR, commerce, etc
- Art and design

Telepresence

UCLA

Id Software

FakeSpace Cave

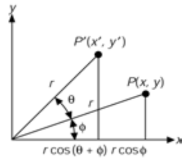
Application Areas

- Entertainment
- CAD/CAM
- Scientific visualization
- Training & Education
- Synthetic Realities
  - VR, commerce, etc
- ***Art and design***

Kimmel Center

Structure of this course…

- Mathematical Preliminaries
- 2D Viewing, Raster Graphics, 2D Transformations
- Representing and Drawing Lines, Arcs and Curves
- 3D modeling and drawing, surfaces
- More advanced topics
  - Hidden lines, Z-buffering, ray tracing, fractals



Programming Assignments

- **No APIs**: OpenGL, GLUT, Mesa, DirectX… (APIs will be saved for Graphics II)
- Just grayscale rendering
- Output in the form of 2D bitmaps
  - XPM: X11 Pixmap Format
  - You will need to write your own code to read and write XPM
- Program executable + source to be handed in with SUBMIT command
- ***Executable MUST RUN on Solaris***
- ***Whatever language you want***, so long as you can deliver a ***statically linked*** executable (lisp, 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.

Programming Assignments Dependency

		HW 1	HW 2	HW 3	HW 4	HW 5	HW 6	HW 7	HW 8	HW 9
HW 1	Read XPM file									
	Write XPM file									
	2D Transformations									
	Window to Viewport Transformation									
HW2	Line Drawing									
	2D Line Clipping									
HW 3	Polygon Filling									
	Polygon Clipping									
HW 4	Circle Drawing									
	Circle Clipping									
HW 5	Bezier Curve Drawing									
HW 6	Anti-aliasing									
	Thick Primitives									
HW 7	3D Viewing									
	3D Clipping									
HW 8	Z-Buffering									
	Backface Culling									
HW 9	Bezier Surfaces									

And If You Lucky

2D Graphics

- **Raster:**

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

- **Vector:**

  Drawing instructions
  - Postscript
  - CGM
  - Fig
  - DWG

Lossy, jaggies when transforming, good for photos.

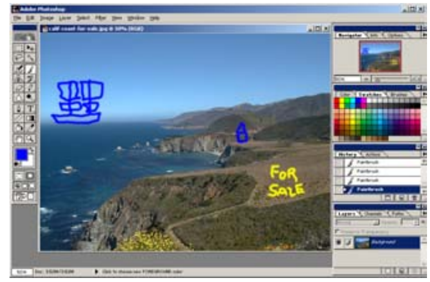
Non-lossy, smooth when scaling, good for line art and diagrams.

2D Graphics

- **Raster:**

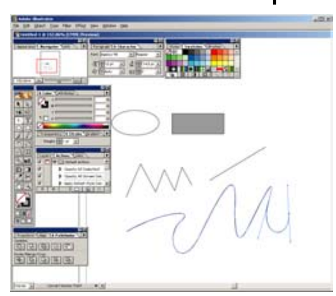
- **Vector:**

Adobe Photoshop: 2D Raster Graphics

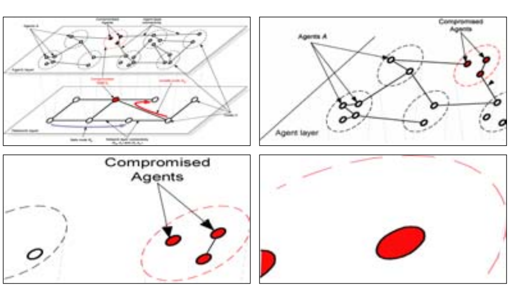


2D Raster Graphics

Adobe Illustrator: 2D Vector Graphics

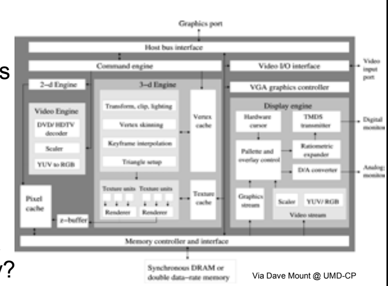


2D Vector Graphics



Graphics Displays

- Nearly all video displays today are RASTER
- How do we go from bits and shapes to pixels on a raster display?



Via Dave Mount @ UMD-CP

The Frame Buffer

- Video memory holding pixels from which the video display is refreshed
  - I.e. essentially a pix/bit map, a Raster image
- Usually implemented on hardware cards
  - Smart frame buffers
    - Accelerated 2D and 3D interaction
    - Color depth (1, 8, 16, 24, 32 bit), Z-buffering
- Double buffering: use of a second memory space to reduce visual artifacts, i.e. swap in-and-out screen buffers