

CSCI 4830/7000

Advanced

Computer

Graphics

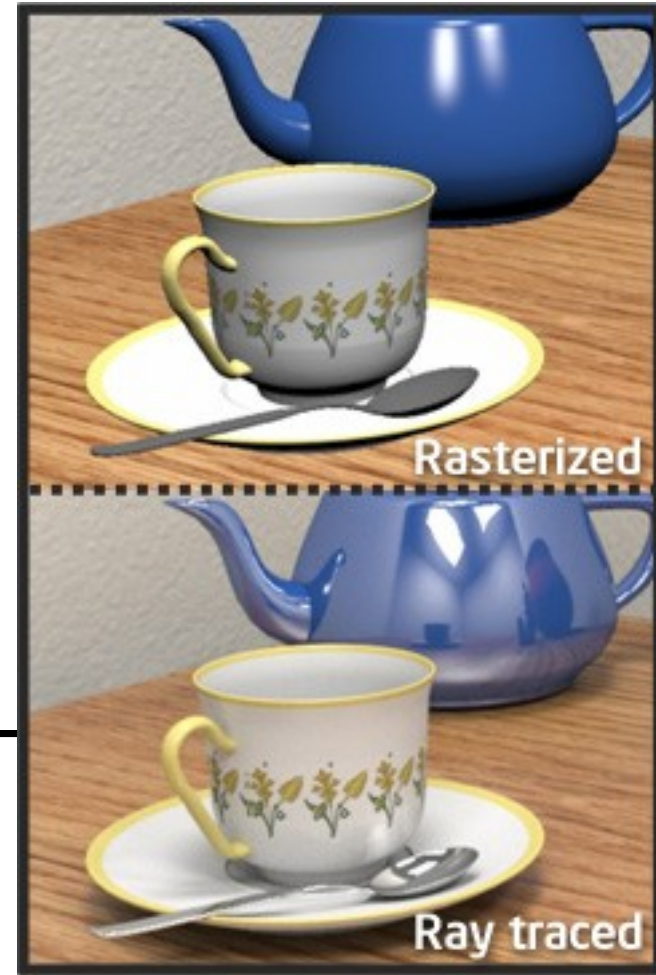
Spring 2010

Instructor

- Willem A (Vlakkies) Schreüder
- Email: willem@prinmath.com
 - Begin subject with 4830 or 7000
 - Resend email not answered promptly
- Office Hours:
 - Before and after Class
 - By appointment
- Weekday Contact Hours: 6:30am - 9:00pm

Course Objectives

- Explore advanced topics in Computer Graphics
 - Pipeline Programming (Shaders)
 - GPU Programming (OpenCL)
 - Ray Tracing
 - Special techniques
 - Shadows
 - Reflections
 - Particle systems
- Assignments: Practical OpenGL
 - Building useful applications



Course Organization and Grading

- Class (50% grade)
 - First hour: Three 20 minute presentations
 - Weekly homework assignments
 - Volunteers and/or round robin
 - Second hour: Introduction of next week's topic
- Semester project (50% grade)
 - Build a significant application in OpenGL
 - 20 minute presentation last two class periods
- No formal tests or final

Assumptions

- You need to be fluent in C
 - Examples are in C (rarely C++)
 - You can do assignments in any language
 - I may need help getting it to work on my system
- You need to be comfortable with OpenGL
 - CSCI 4229/5229 or equivalent
 - You need a working OpenGL environment

Grading

- Satisfactory complete all assignments => A
 - The goal is to impress your friends
- Assignments **must** be submitted on time unless prior arrangements are made
 - Due before next class
- Assignments must be completed individually
 - Stealing ideas are encouraged
 - Code reuse with attribution is permitted
 - Use as starting point
 - Make it your own
 - **Improve it**
- Class attendance HIGHLY encouraged

Text

- OpenGL Shading Language (3ed)
 - Randi J. Rost et. al.
 - “OpenGL Orange Book”
 - Chapters 6 onward discusses Shaders
- Advanced Graphics Programming Using OpenGL
 - Tom McReynolds and David Blythe
 - Not required but a good reference
 - Does not cover all the topics

Other Texts

- OpenGL Programming Guide (5ed)
 - Shreiner, Woo, Neider & Davis
 - “OpenGL Red Book”
 - Download previous editions as PDF
- OpenGL SuperBible: Comprehensive Tutorial and Reference (4ed)
 - Wright, Lipchak & Haemel
 - Good all-round theory and applications

OpenGL Resources

- www.google.com
 - Need I say more?
- www.opengl.org
 - Code and tutorials
- nehe.gamedev.net
www.lighthouse3d.com
 - Excellent tutorials
- www.mesa3d.org
 - Code of “internals”
- www.prinmath.com/csci5229
 - Example programs from CSCI 4229/5229

Assignment 0

- Due: **Friday** Jan 15 by 9pm
- Sign up with moodle.cs.colorado.edu
 - Enrollment key: 48307000
 - A picture will help me learn your names
- Submit
 - Your study area
 - Platform (Hardware, Graphics, OS, ...)
 - Any specific interests in computer graphics
 - Specific topics you want to see covered
 - Initial project idea(s)

My information

- Mathematical modeling and data analysis
 - PhD Computational Fluid Dynamics [1986]
 - PhD Parallel Systems (*CU Boulder*) [2005]
 - President of *Principia Mathematica*
- Use graphics for scientific visualization
- Open source bigot
- Program in C, C++, Fortran and Perl
- Outside interests
 - Aviation
 - Amateur radio (voice and digital)

How to get started

- Make sure you have a working OpenGL environment
 - Compile and run examples from CSCI [45]229
- Make sure advanced examples work
 - Windows may need GLEW
 - Linux/nVidia may need driver and library from nVidia
 - OS/X may need Xcode or Fink

OpenGL Extension Wrangler (GLEW)

- Maps OpenGL extensions at run time
 - Provides headers for latest OpenGL
 - Finds vendor support at run time
- Check support for specific functions or OpenGL version at run time
 - Crashes if unsupported features are used
- Use only if you have to (Windows mostly)

Assignment 1

- Due: Thursday January 21, 2009 before class
- Alternate OpenGL environments
 - GLUT is an easy to use and widely supported environment for running OpenGL
 - Created for the Red Book
 - Used by most OpenGL texts
 - Investigate alternative environments and compare it to GLUT, for example
 - Qt (Cross platform C++ application framework)
 - Simple DirectMedia Layer (SDL)
 - iPhone SDK
 - Write a simple visualizer of a 3D scene

Assignment 2

- Due: Thursday January 28, 2009
- Model Loader
 - There are lots of programs for generating three dimensional models, (e.g. Blender, Maya) that can export models in various formats (obj, 3ds, pye, ...)
 - Premade models are available for complex objects
- Write a loader to load a complex model and display it using OpenGL
- If you use my loader you must improve it

Project

- Should be a program with a significant graphics component
 - Something useful in your research/work?
 - Graphical front end to simulation
 - Graphical portion of a game
 - Expect more from graduate students
- Deadlines
 - Proposal: Thursday March 4 (earlier is better)
 - Review: Thursday April 1 (progress report)
 - Final: **Monday** April 26

Nuts and Bolts

- Complete assignments on any platform
 - Assignments reviewed under Ubuntu 9.10
 - Set `#ifdef` so I can compile and run it
- Submit using `moodle.cs.colorado.edu`
 - ZIP or TAR (no RAR)
 - Name executables `hw1`, `hw2`, ...
 - Create a makefile so I can do *make clean;make*
 - Set window title to *Assignment X: Your Name*
- Include number of hours spent on assignment
- ***Check my feedback and resubmit if requested***

A few hints

- My machine runs Linux x86_64
 - gcc/g++ with nVidia & GLX
 - -Wall is a **really** good idea
 - case sensitive file names
 - int=32bit, long=64bit
 - little-endian
 - fairly good performance
- How to make my life easier
 - Try it in CSEL or a Linux box
 - Stick to C/C++ unless you have a good reason
- **Maintain thy backups...**

Class Presentations

- If have a special interest in the topic and have something special to contribute
VOLUNTEER
- If by Sunday morning there are not three volunteers, I will appoint volunteers on a round robin basis (in order by MD5 of names)
 - You can trade places if you can talk somebody into or out of a slot
- Everybody must present at least twice, but you can do more if you want
- Popular topics may have more presenters

What to Present

- Should be (mostly) the assigned topic
 - Feel free to push the envelope
- Show what you did for the assignment
 - Cover principles or theory I omitted
 - Show and describe code of interest
 - Demonstrate “gotchas” you encountered
 - Impress your friends
- Keep it interesting

How to Present

- 20 minutes can be an eternity or over in a wink
 - Plan your time (practice a bit)
 - If you use slides figure 2 minutes per slide
- Plan your presentation
 - What are the key points you want to convey?
 - How do you illustrate the key points?
- The presentation should TEACH
 - Teaching is learning twice
 - Adapt to the questions

How to Listen

- If you don't understand, ask
 - Helps the presenter understand what's new to you
- If you disagree, say so
 - Maybe the presenter misspoke or has an different opinion worth discussing
- Be nice – you may be next!

CAETE Students

- This is an experiment
- Suggest ways you present remotely
- Provide MOV or similar demonstration
- Stick to the class schedule if possible

Parallel Flight Simulator Project

- Consider joining a project with many members
 - Each member has a specific subtask
 - World visualization
 - Special effects
 - Flight dynamics
 - Multi-function displays (instruments)
 - Networking
 - Flight controls
 - Sound
 - Rotating project manager
 - Responsible for managing the project for a week
 - Provide concise report of what was done the last week
 - Lay out a plan for what should be done the next week
- Somewhat like a real software project
 - I will be the client

Review of OpenGL

- OpenGL is an API or library
 - GLU for some utilities and convenience functions
 - GLUT or other library for OS interface
 - Just gets a picture inside an OS supplied window
- OpenGL uses a state machine
 - Function calls set attributes that apply to subsequent objects
 - Objects are built from primitive elements
 - Vertexes define points in 3D space
 - Build lines and polygons from vertexes
 - Build 3D objects from polygons (skin only)

Vertex Attributes

- Position (x,y,z,w)
 - glVertex
- Color (R,G,B,α)
 - glColor
- Normal (x_n, y_n, z_n, w_n)
 - glNormal
- Texture Coordinates (s,t,r,q)
 - glTexture
 - glMultiTex
- User defined attributes

Points

- One at each vertex
 - glBegin(GL_POINTS)
- Point size
 - glPointSize()

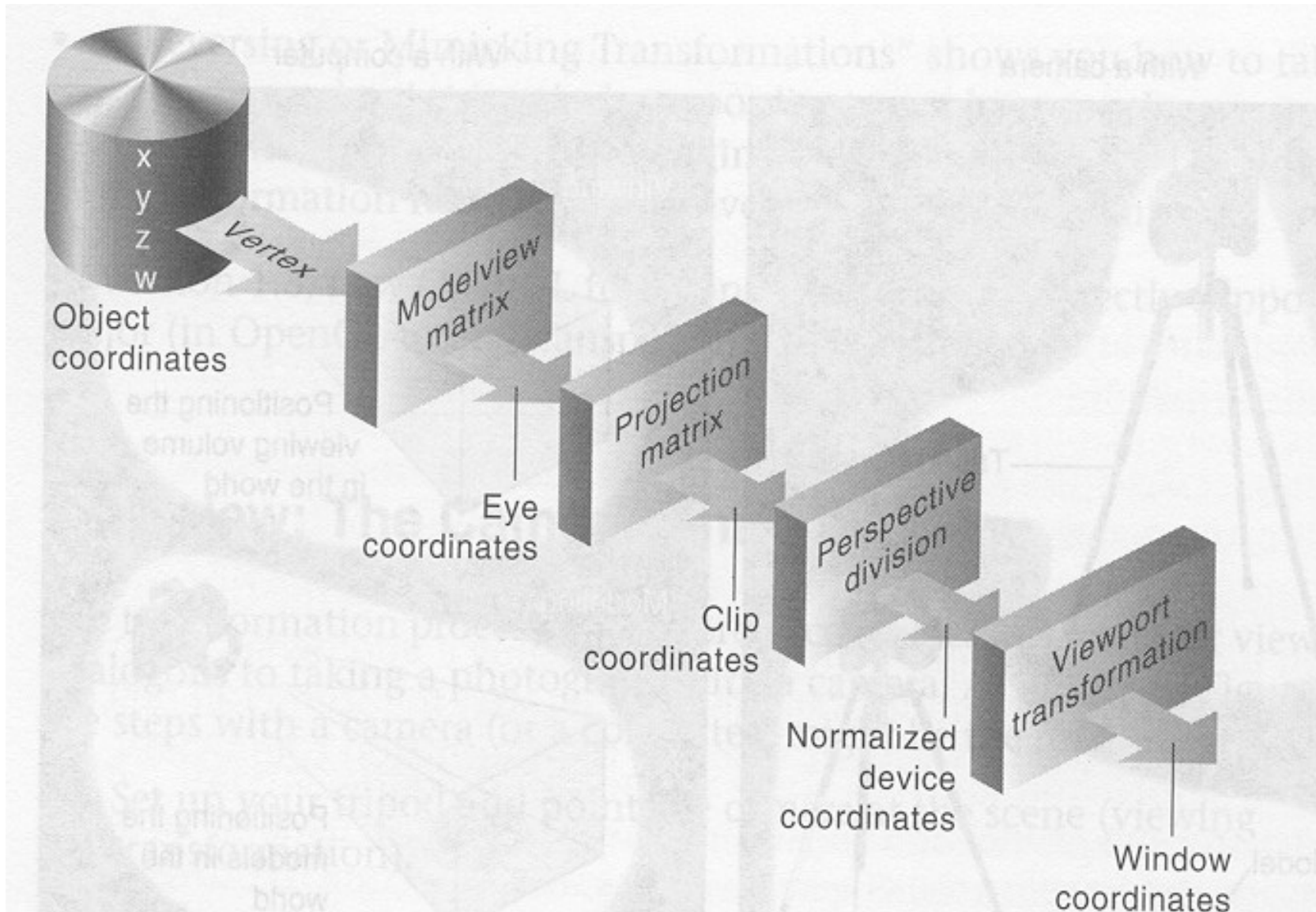
Lines

- Built from vertexes
 - glBegin(GL_LINES)
 - glBegin(GL_LINE_STRIP)
 - glBegin(GL_LINE_LOOP)
- Line pattern
 - glLineStipple()
- Line width
 - glLineWidth()

Polygons

- Build from vertexes
 - glBegin(GL_POLYGON)
 - glBegin(GL_TRIANGLES)
 - glBegin(GL_QUADS)
 - many more
- Must be planar
- Fill pattern
 - glPolygonStipple()
 - glShadeModel()

OpenGL Transformation Pipeline



Transformations

- `glMatrixMode()`
- Primitive transforms
 - `glTranslate`
 - `glScaled`
 - `glRotated`
- Utilities
 - `gluPerspective()`
 - `gluLookat()`
- Direct manipulation (homogeneous coordinates)
 - `glMultMatrix`

Modes

- glEnable(), glDisable()
 - GL_DEPTH_TEST
 - GL_CULL_FACE
 - GL_LIGHTING
 - GL_TEXTURE_2D
 - GL_SHADOWS

Blinn-Phong Lighting

- Enable `GL_LIGHTING` & `GL_LIGHTx`
- `glLightXXX`
 - Ambient, Diffuse, Specular
 - Position, Direction, Spot Angle
 - Local/Global, One/Two sided
- `glMaterialXXX`
 - Emission, Ambient, Diffuse, Specular, Shininess
- `glNormal`

Textures

- `glEnable(GL_TEXTURE_xD)`
- `glGenTexture()`
- `glBindTexture()`
- `glTexImage2D()`
 - you need to load the image yourself
- `glTexParameter()`
- `glTexEnv()`

Blending and Transparency

- glEnable(GL_BLEND)
- glBlendFunc(source, destination)
 - GL_ZERO
 - GL_ONE
 - GL_DST_COLOR
 - GL_ONE_MINUS_DST_COLOR
 - GL_SRC_ALPHA
 - GL_ONE_MINUS_SRC_COLOR
 - GL_DST_ALPHA
 - GL_ONE_MINUS_DST_ALPHA
 - GL_SRC_ALPHA_SATURATE
- Anti-aliasing

Other Features

- Display Lists
- Polygon Offset
- Fog
- Bezier curves and surfaces
- Masking
 - Color, Z-buffer, etc can be made read-only
 - Stencil buffer
- Shadows (Shadow Volume & Shadow Map)
- Many more..

Interfacing with the OS

- Displaying the image
 - Request window features
 - Redraw window
 - Window resized
- User interaction
 - Keyboard input
 - Mouse input
- Other OS functions
 - Elapsed time
 - Sound

GLUT (OpenGL Utility Toolkit)

- Used by many textbooks
 - Implementations for Unix, Windows, MacOS, ...
- Hides OS specific complexity
 - Create window and request properties
 - Callback functions to process events
 - Display
 - Keyboard
 - Mouse
 - glutGet to access time, dimensions, ...
- Limited support for menus

Alternatives to GLUT

- Simple Directmedia Layer
 - Similar to GLUT
 - Add sound, menus, etc..
- Nokia/Trolltech Qt
 - Cross Platform C++ Application Framework
 - Comprehensive sets of widgets
 - OpenGL window is a widget
- iPhone SDK
- Many others