

CSCI 4229/5229
Computer
Graphics
Summer 2011

Course Objectives

- Class: Theory and principles
 - Attendance is highly encouraged
- Assignments: Practical OpenGL
 - Applications
- No tests or exams
- By the end of the course you will:
 - Be conversant in computer graphics principles
 - Be well versed in the use of OpenGL
 - Understand what OpenGL does internally

Course Outline

- Basics (1/3)
 - Projections, transformations, clipping, rendering, text, color, hidden edge and surface removal, and interaction
- Advanced (1/3)
 - Illumination, shading, transparency, texture mapping, parametric surfaces, shaders
- Project (1/3)
 - Whatever you're interested in: games, modeling, visualization, 'Google Earth',

Why OpenGL?

- Modern, widely used and actively supported
 - Games
 - 3D visualization
- Cross platform
 - Windows
 - Mac
 - *NIX
 - iPhone and Android
- Open source and vendor implementations
 - MESA 3D (source code available)
- Many language bindings

Instructor

- Willem A (Vlakkies) Schreüder
- Email: willem@prinmath.com
 - Begin subject with 4229 or 5229
 - Resend email not answered promptly
- Office Hours:
 - Before and after Class
 - By appointment
- Weekday Contact Hours: 6:30am - 9:00pm

Assumptions

- You need to be fluent in C
 - Examples are in C
 - You need to know how to program and compile
 - You can do assignments in any language
 - I may need help getting it to work on my system
 - Use C or C++ unless you have a good reason
- You need to be comfortable with linear algebra
 - Matrix and Vector multiplication
 - Dot and cross products
 - Rotation matrices

Grading

- Satisfactory complete all assignments => A
 - The goal is to impress your friends
- Assignments **must** be submitted on time unless prior arrangements are made
 - Most due Sunday evening 11:59 pm
 - Grace period until Monday morning at 08:00am
 - CAETE students: Let me know what will work
- Assignments must be completed individually
 - Stealing ideas are permitted
 - OpenGL code fragments from web may be used
 - Make it your own and improve on it

Grading Expectations

- Code reuse is acceptable
 - Give credit where it is due
 - You take responsibility for errors in reused code
 - You need to make a substantial improvement
 - I'm looking to see that you have insight in the material
- No grade => respond to my comments and resubmit
- **Grade <100 means NOT SATISFACTORY**

Text

- OpenGL: A Primer, 3/E
 - Edward Angel
 - An excellent and very accessible introduction to OpenGL
 - Inexpensive
 - Third edition adds new material including shaders
 - Recommended but not required

Other Texts

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

And More Texts

- OpenGL Shading Language (2ed)
 - Randi J. Rost
 - “OpenGL Orange Book”
 - Introduces both OpenGL and Shaders
- OpenGL Reference Manual (4ed)
 - OpenGL Architecture Review Board & Dave Shreiner
 - “OpenGL Blue Book”
 - Official Reference Document to OpenGL, Version 1.4
 - A bit dated, very similar to man pages

Related texts

- OpenGL ES 2.0 Programming Guide
 - Munshi, Ginsburg, Schreiner
 - OpenGL Embedded Systems (iPhone & Android)
 - Subset of OpenGL, 1.3 and 2.0 very different
 - ***Not recommended for beginners***
- Computer Graphics: Principles & Practice (2ed)
 - Foley, van Dam, Feiner & Huges
 - Avoid 1ed (Pascal), 2ed also a bit dated
 - Get it if you want to know more of the theory

OpenGL Resources

- www.google.com
 - Need I say more?
- www.opengl.org
 - Code and tutorials
- nehe.gamedev.net
 - Excellent tutorials
- www.mesa3d.org
 - Code of “internals”
- Class forum

Assignment 0

- Due: **Wednesday** June 1, 2010 at **noon**
- Sign up with moodle.cs.colorado.edu
 - Enrollment key: 42295229
 - A picture will help me learn your names
- Submit
 - Your name and study area
 - Platform (Hardware, Graphics, OS, ...)
 - Background and interests in computer graphics
 - Project ideas (if you have one already)
 - CAETE students propose schedule for homework

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

How to get started

- Get OpenGL to work on your platform
 - *Installing OpenGL* on moodle
 - Compile and run *Hello World* examples
- If you are using Windows
 - Use **glutcu** which adds *glWindowPos*
 - Link in my *glWindowPos* code
- If you are on an X based (*NIX) platform:
 - yum install freeglut-devel
 - apt-get install freeglut3-dev
 - Run glxinfo and check if *direct rendering: yes*
- OS/X based on OpenGL
 - Free SDK

Assignment 1

- Due: Sunday June 5, 2010 at 23:59
- Write an OpenGL based visualization of the Lorenz Attractor
 - At a minimum show a static line path in 3D
 - Add rotation using cursor keys
 - Use your imagination
- The purpose is scientific visualization
 - Do some science

<http://mathworld.wolfram.com/LorenzAttractor.html>
- Example 6 is your friend

Assignment 2

- Due: Sunday June 12, 2010 at 23:59
- Write an program to visualize a 3D scene
- Scene must consist of solid 3D objects
 - You must create all objects yourself
 - no GLU/GLUT
 - You must replicate some generic objects
- Scene must be viewable from different vantage points under user control
- Generate scene in orthogonal, add perspective

Assignment 3

- Due: Sunday June 19, 2010 at 23:59
- Write an program to visualize a 3D scene with lighting and textures
 - Make the light move to show lighting effects
 - Select solid objects that show lighting effects
- *Add lighting to Assignment 2*
- *Then add textures*
- **WARNING: This homework is a LOT harder than the first two**

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: Monday June 20 (earlier is better)
 - Review: Sunday June 26 (progress report)
 - Final: Wednesday June 29 (show and tell)

Topics for CSCI 4830/700

Advanced Computer Graphics

- Shaders
 - Programing the GPU
- Embedded Systems (iPhone & Android)
- GPU work threads (CUDA & OpenCL)
- Ray Tracing

Nuts and Bolts

- Complete assignments on any platform
 - Assignments reviewed under Ubuntu 11.04
 - Set `#ifdef` so I can compile and run it
- Submit using `moodle.cs.colorado.edu`
 - ZIP or TAR
 - 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 task
- ***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 to use something else
- **Maintain thy backups...**

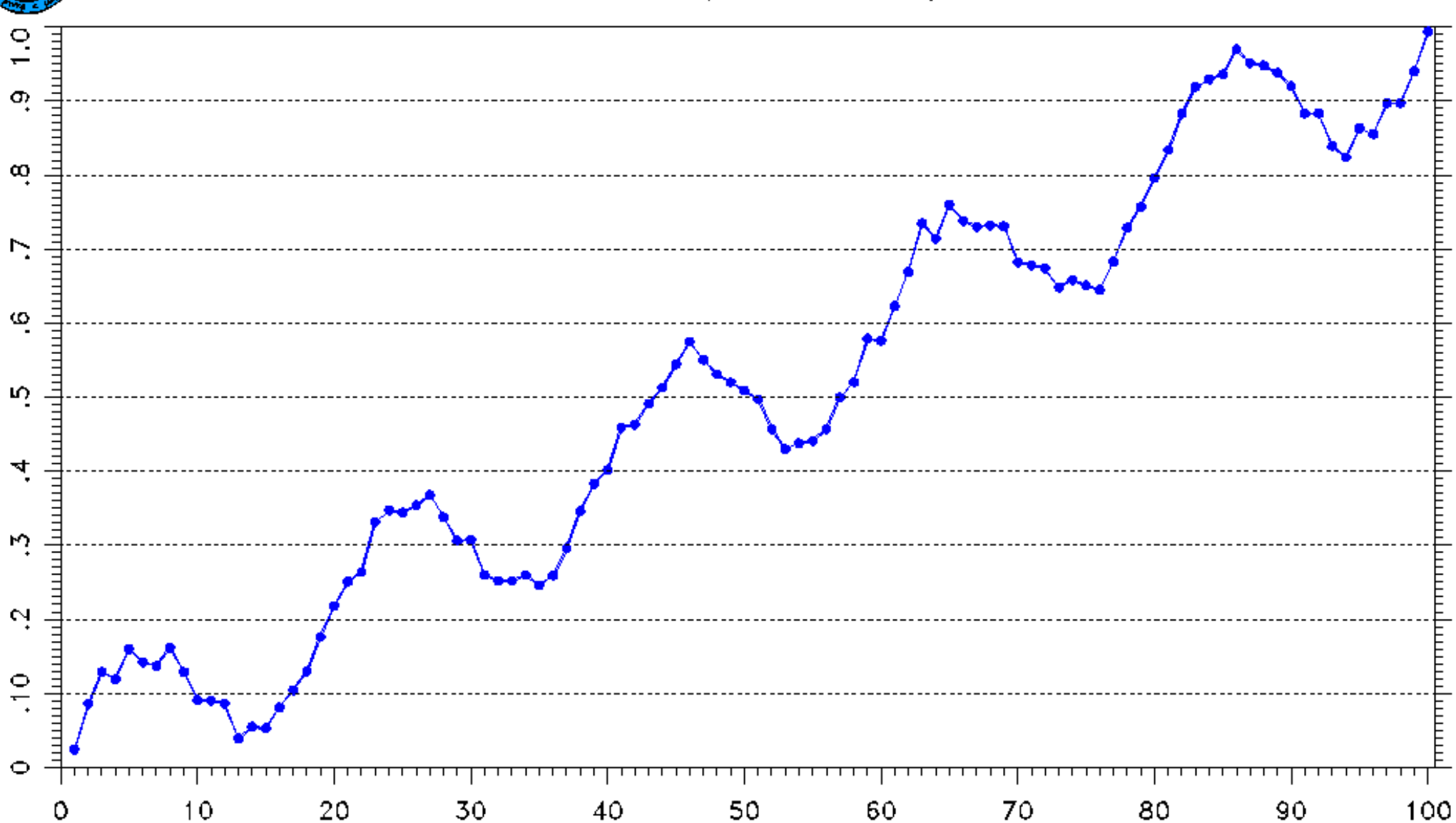
The Importance of Graphics: 100 Values between 0 and 1

| | | | | | | | | | |
|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| 0.024 | 0.086 | 0.129 | 0.119 | 0.160 | 0.142 | 0.137 | 0.162 | 0.129 | 0.091 |
| 0.090 | 0.086 | 0.039 | 0.055 | 0.053 | 0.081 | 0.104 | 0.130 | 0.176 | 0.218 |
| 0.251 | 0.264 | 0.331 | 0.347 | 0.344 | 0.354 | 0.368 | 0.338 | 0.306 | 0.307 |
| 0.260 | 0.252 | 0.252 | 0.260 | 0.246 | 0.259 | 0.296 | 0.346 | 0.383 | 0.402 |
| 0.459 | 0.463 | 0.491 | 0.513 | 0.544 | 0.575 | 0.550 | 0.531 | 0.520 | 0.509 |
| 0.497 | 0.457 | 0.430 | 0.438 | 0.441 | 0.457 | 0.500 | 0.520 | 0.579 | 0.576 |
| 0.623 | 0.669 | 0.735 | 0.714 | 0.760 | 0.738 | 0.730 | 0.732 | 0.731 | 0.682 |
| 0.678 | 0.674 | 0.648 | 0.658 | 0.651 | 0.645 | 0.683 | 0.729 | 0.757 | 0.796 |
| 0.834 | 0.883 | 0.919 | 0.929 | 0.936 | 0.970 | 0.951 | 0.948 | 0.938 | 0.920 |
| 0.883 | 0.883 | 0.839 | 0.824 | 0.863 | 0.855 | 0.897 | 0.897 | 0.940 | 0.994 |



100 Values between 0 and 1

The Importance of Graphics



Graphic Design

- 2D vs. 3D
 - Cool vs. informative
- Edward R. Tufte
 - Visual Explanations
 - Envisioning Information
 - The Visual Display of Quantitative Information
 - Beautiful Evidence

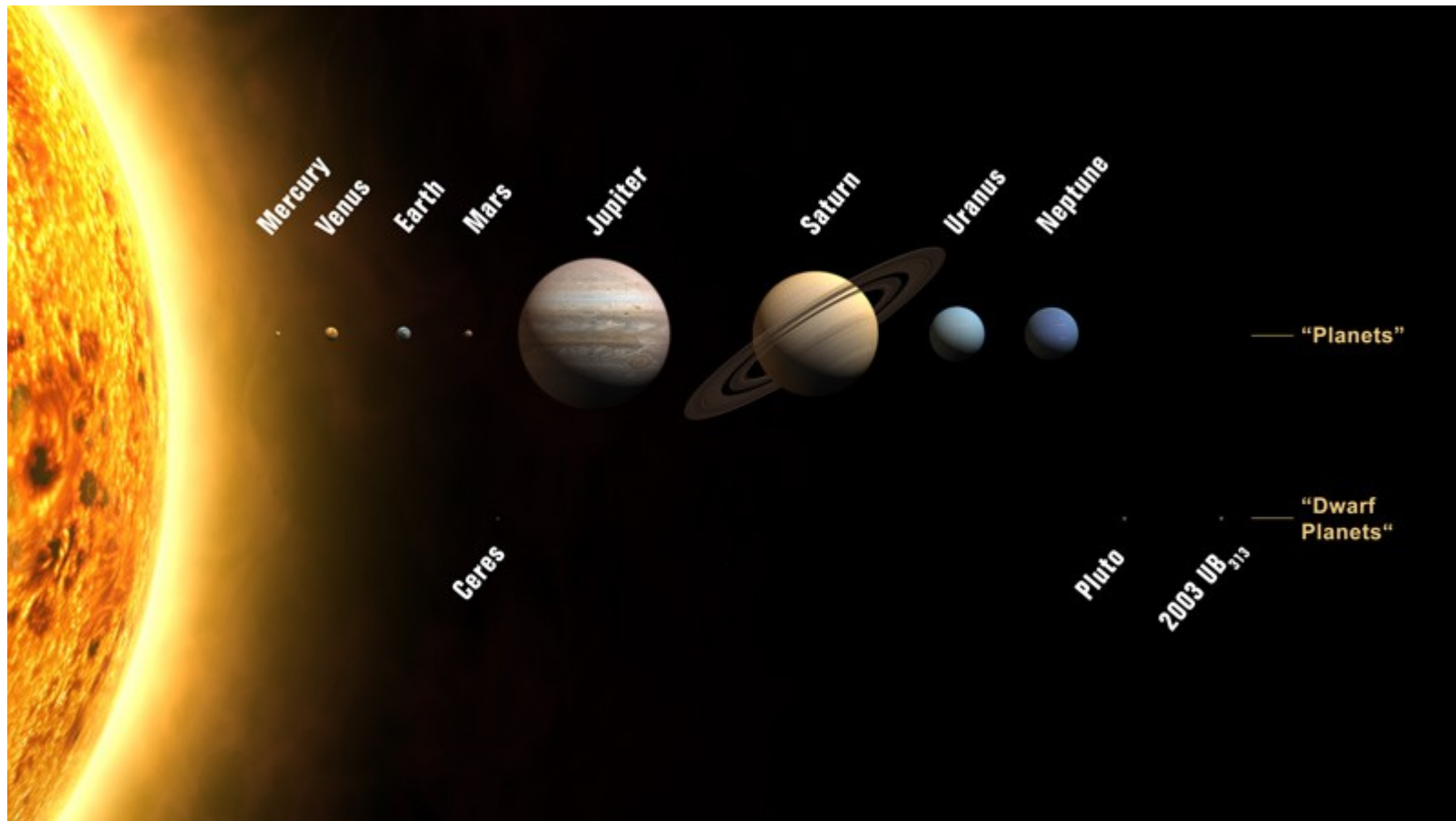
Saturn from Cassini Probe



Colorado Fall Colors



What is wrong with this picture?



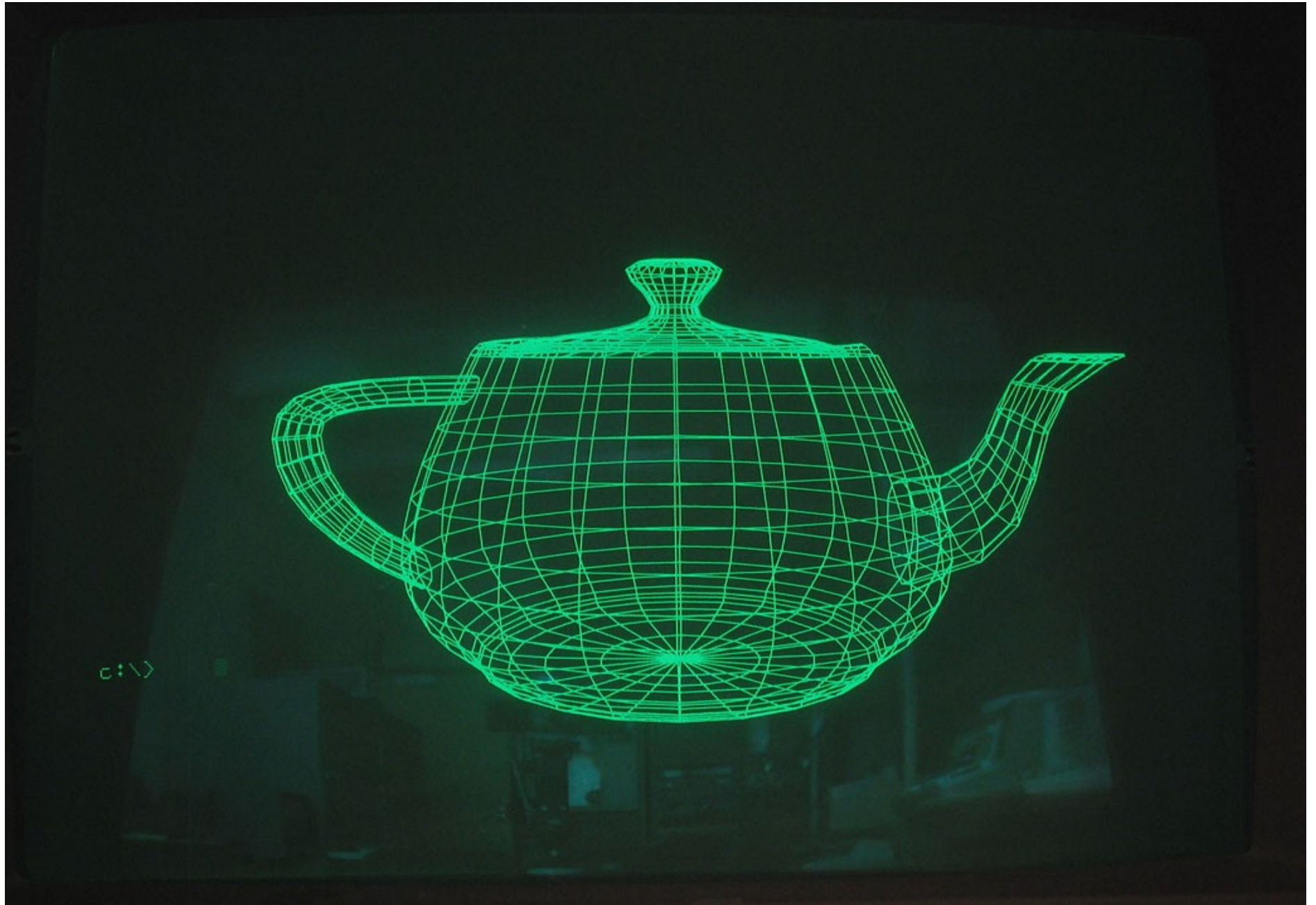
In the beginning....



Storage Tube Terminals



Storage Display Images



Color: Multiple Pen Plotters



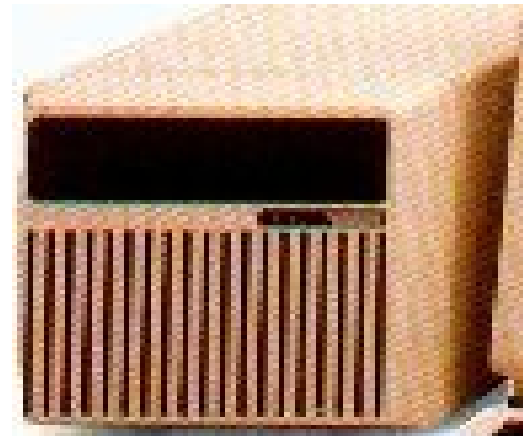
Raster Graphic Terminals



Color Inkjets



Workstations: Apollo DN 330 12 MHz 68020, 3MB RAM, 70MB disk



Workstation, Desktop, Laptop, Phone, Communicator..



Plotting Packages

- PLOT-10: Tektronix 4010 graphics
- PLOT88: PC graphics
- DISSPLA: NCAR graphics
- GINO: Portable graphics
- DIGLIB: LLNL device-independent, open source
- GKS: Graphics Kernel System (2D vector)
- PHIGS: 3D Interactive Graphics
- OpenGL and DirectX

The rise of OpenGL

- Originated as SGI IrisGL
- Vendor-neutral OpenGL controlled by ARB
- Hides the details of hardware
 - Software emulation when necessary
 - Hardware acceleration when possible
- Supports 2D to advanced 3D graphics
- Portable to most hardware and OS with WGL, AGL and GLX
- Hardware range from phones to Big Iron

Focus of OpenGL

- OpenGL 1 (1992)
 - Hardware abstraction
- OpenGL 2 (2004)
 - Add Shaders (Programming the GPU)
- OpenGL 3 (2008)
 - Focus on shaders and new hardware
 - Deprecates many features
- OpenGL 4 (2010)
 - Core & Compatibility Profiles
 - Merge desktop and embedded versions

Gaming and Graphics

- Text based/ASCII graphics (Pong, PacMan)
- 2D monochrome line graphics (Astroids)
- 2D images & sprites (Mario)
- 3D graphics
 - Flight Simulators (2D -> 3D)
 - First Person Shooters
 - Multi-player games
- Games push the envelope
 - Realism
 - Speed