COMPUTER GRAPHICS

Lecture 1
Introduction

Course Information

- Lectures
- Laboratory
- Readings + Quizzes + Presentations (?)
- Assignment (...)
- Project
  - 0 - Team formation & topic choice
  - 1 - Understand the problem space
  - 2 - Exploring the design space
  - 3 - Prototype
  - 4 - Evaluation

Course Information

- Grading
  - Group project (20%)
    - More details soon...
  - Mid-term exam (25% total)
    - Hand written
  - Assignment & Quiz (15% total)
    - Two week to do assignment, likely 1 over the semester
  - Final Exam (40% total)
  - Bonus (5% total)
    - Class involvement and peer lecture
    - Includes project involvement/effort
    - Presentation for special topics

Resources

- Library
- Books
- Web sites
- Standard documents
- Go further
  - Move beyond lectures & book
  - Further courses
  - Step into research

Books

- Computer Graphics with OpenGL, 4rd Edition, Donald Hearn, M. Pauline Baker, and Warren R. Carithers, Prentice Hall, 2011.
- Introduction to Computer Graphics, J.D. Foley, A van Dam, S.K. Feiner, J.F. Hughes, and R.L. Philips, Addison-Wesley, 1994.
- Interactive Computer Graphics: A Top-Down Approach with OpenGL, Edward Angel, Addison-Wesley, 2009
- OpenGL Programming Guide, 5th Edition: The Official Guide to Learning OpenGL, Version 2.0, Addison-Wesley, 2006.
- OpenGL Reference Manual, 4th Edition: The Official Reference Document to OpenGL, Version 1.4, Addison-Wesley, 2004

COMPUTER GRAPHICS

Here we go…

This course is ***not*** about the specification of 3D graphics programs and APIs like Maya, Alias, DirectX but about the concepts underlying them.

Why Study 3D Computer Graphics?

- Applications (discussed next)
- Fundamental Intellectual Challenges

APPLICATIONS OF COMPUTER GRAPHICS

Entertainment

Computer Graphics is about animation (films)

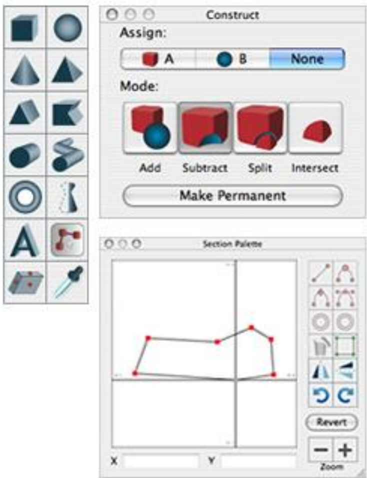
Movie industry

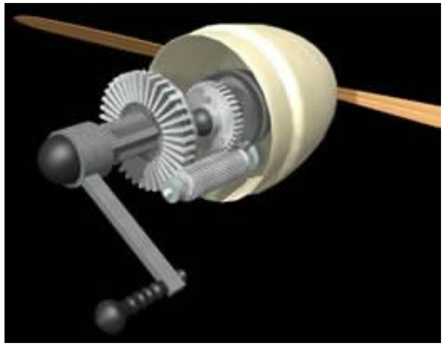
Lighting Simulation


Interior Design


Automobile Visualization

Computer Aided Design

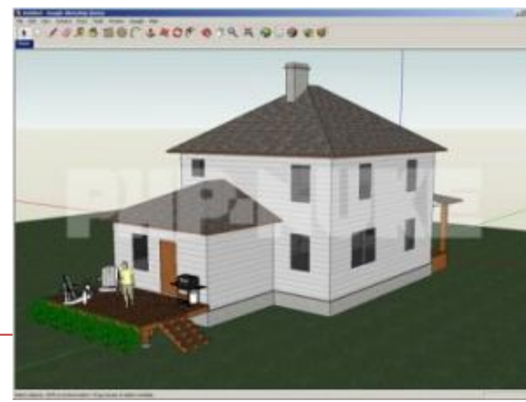




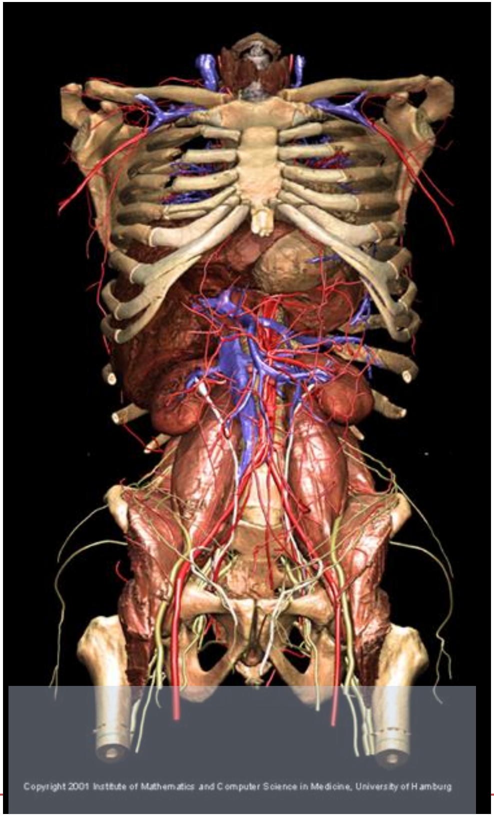
Interiors Professional

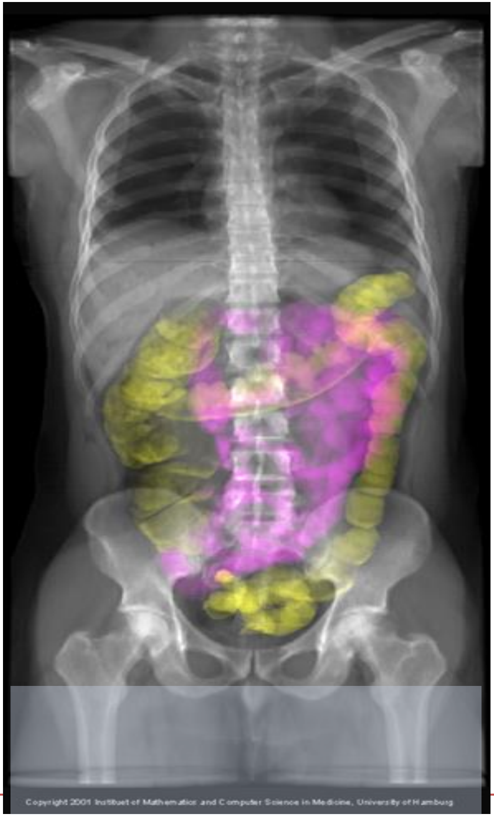
Mechanical CAD
Architectural CAD
Electronics CAD
Casual Users

Google Sketchup

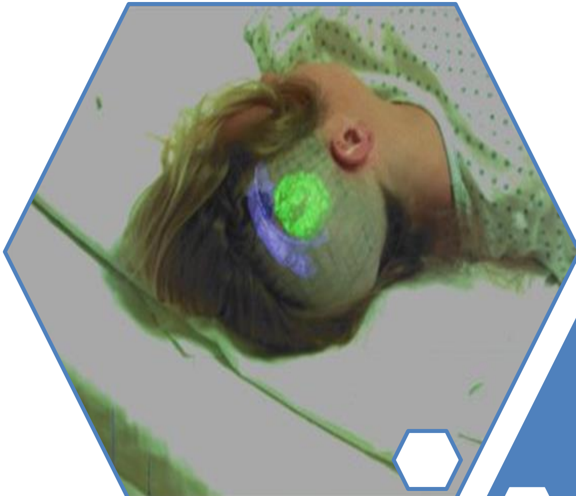


Visualization: Science and Medicine





Medical Imaging

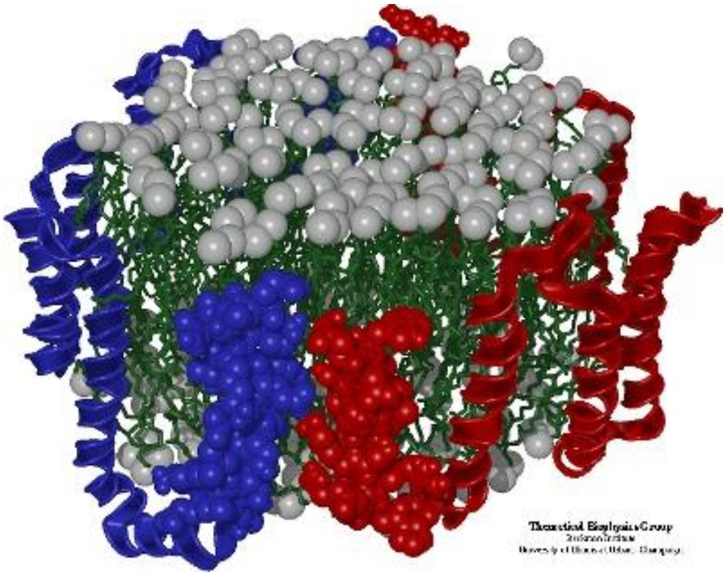


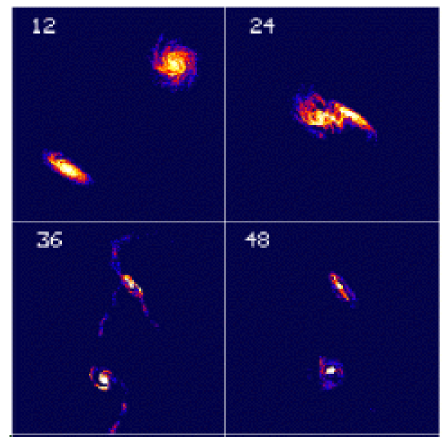
Medical Imaging is a key driving force

Video Gaming

Games are very important in Computer Graphics

Scientific Visualisation





Virtual Reality

- VR for design and entertainment
- Simulators: Surgical, Flight, Driving, Spacecraft

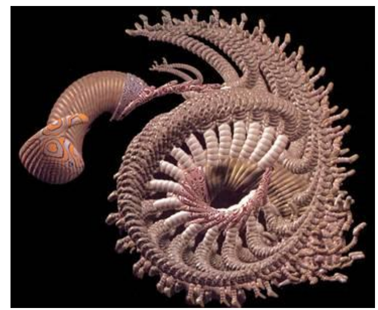
What is Computer Graphics?

What is Computer Graphics? (1/2)

- Computer graphics generally means creation, storage and manipulation of models and images
- Such models come from diverse and expanding set of fields including physical, biological, mathematical, artistic, and conceptual/abstract structures

Frame from animation by William Latham, shown at **SIGGRAPH 1992**. Latham creates his artwork using **rules** that govern patterns of natural forms.

Our newest visual computing faculty member, Daniel Ritchie's Ph.d. thesis: **Probabilistic Programming for Procedural Modeling and Design**



What is Interactive* Computer Graphics? (1/2)

- User controls content, structure, and appearance of objects and their displayed images via rapid visual feedback
- Basic components of an interactive graphics system
  - input (e.g., mouse, stylus, multi-touch, in-air fingers.
  - processing (and storage of the underlying represent
  - display/output (e.g., screen, paper-based printer, video recorder...)
- First truly interactive graphics system, Sketchpad, pioneered by Ivan Sutherland 1963 Ph.D. thesis Sketchpad, A Man-Machine Graphical Communication System
- Used TX-2 transistorized "mainframe" at MIT Lincoln Lab

Note CRT monitor, light pen, and function-key panels – the "organ console" showing bi-manual operation

* Sometimes called real-time computer graphics, and in certain contexts, real-time rendering

Why Study 3D Computer Graphics?

- Applications (discussed previously)
- Fundamental Intellectual Challenges
  - Create and interact with realistic virtual world
  - Requires understanding of all aspects of physical world
  - New computing methods, displays, technologies
- Technical Challenges
  - Math of (perspective) projections, curves, surfaces
  - Physics of lighting and shading
  - 3D graphics software programming and hardware

Key terms

- Pixels
- Vertexes

- Modeling
- Animation
- Rendering

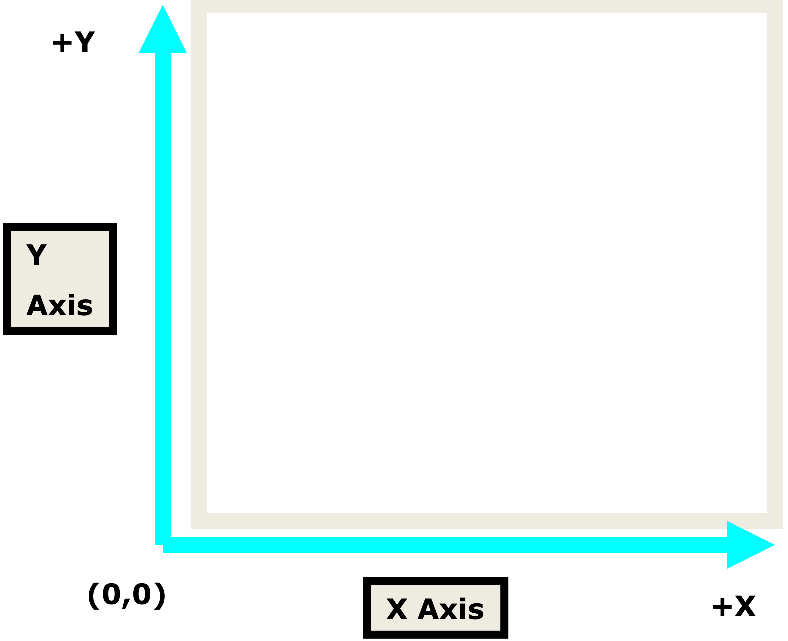
- Transformation

Business of Generating Images

- Images are made up of pixels

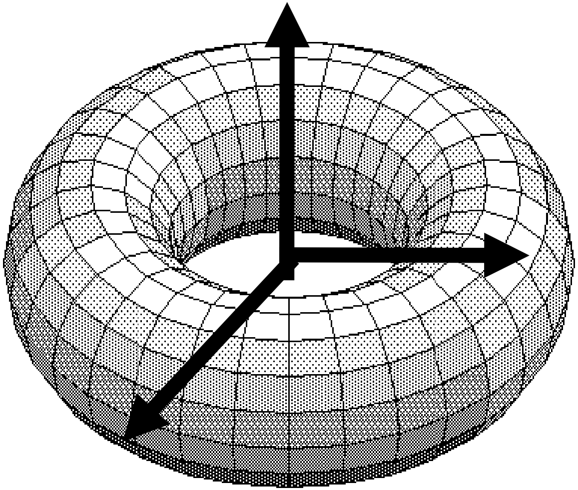
Two Dimensional Images

- Images (at least the ones in this class) are two dimensional shapes.
- The two axes we will label as X (horizontal), and Y (vertical).



Three Dimensional Images

- What if we add one extra axis?
- **Called 3D**



15/04/16

Graphics Definitions

- Point
  - A location in space, 2D or 3D
  - Sometimes denotes one pixel
- Line
  - Straight path connecting two points
  - Infinitesimal width, consistent density
  - Beginning and end on points

Graphics Definitions

- Vertex
  - point in 3D
- Edge
  - line in 3D connecting two vertices
- Polygon/Face/Facet
  - arbitrary shape formed by connected vertices
  - fundamental unit of 3D computer graphics
- Mesh
  - set of connected polygons forming a surface (or object)

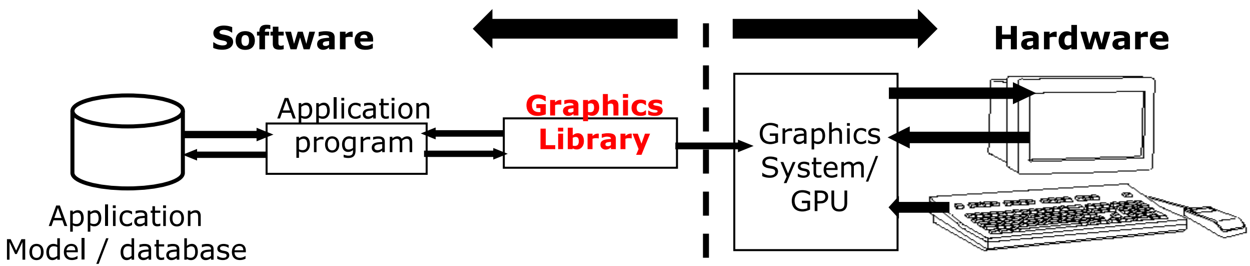
Graphics Definitions

- Rendering : process of generating an image from the model
- Framebuffer : a video output device that drives a video display from a memory containing the color for every pixel

FURTHER MORE

Conceptual Framework for Interactive Graphics

- Graphics library/package is intermediary between application and display hardware (Graphics System)
- Application program maps application objects to views (images) of those objects by calling on graphics library. Application model may contain lots of non-graphical data (e.g., non-geometric object properties)
- User interaction results in modification of image and/or model
- This hardware and software framework is 5 decades old but is still useful

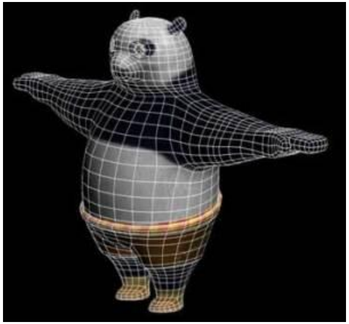


Graphics Library

- Examples: OpenGL™, DirectX™, Windows Presentation Foundation™ (WPF) accessed via XAML, RenderMan™, HTML5 + WebGL™
- Primitives (characters, lines, polygons, meshes,...)
- Attributes
  - Color, line style, material properties for 3D
- Lights
- Transformations
- Immediate mode vs. retained mode
  - immediate mode: no stored representation, package holds only attribute state, and application must completely draw each frame
  - retained mode: library compiles and displays from scenegraph that it maintains, a complex DAG. It is a display-centered extract of the Application Model

Application Distinctions: Two Basic Paradigms

Sample-based graphics *vs* Geometry-based graphics

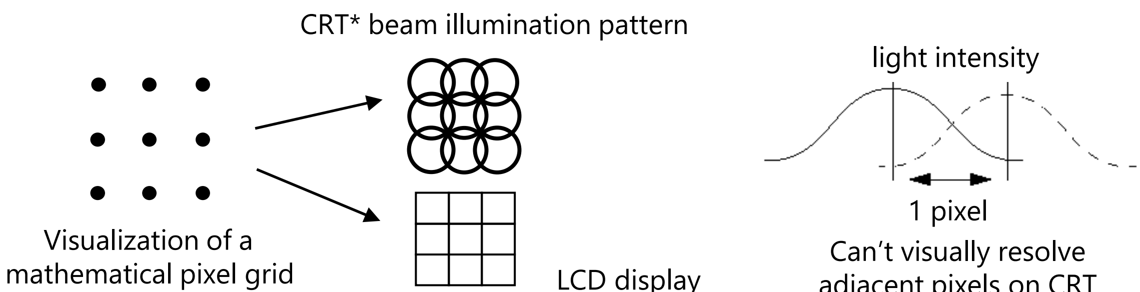


Sample-based Graphics (1/3)

- **Sample-based graphics:** Discrete samples are used to describe visual information
  - pixels can be created by digitizing images, using a sample-based "painting" program, etc.
  - often some aspect of the physical world is sampled for visualization, e.g., temperature across the US
  - example programs: Adobe Photoshop™, GIMP™ , Adobe AfterEffects™ (which came out of CS123/CS224!)

Sample-based Graphics (2/3)

- Pixels are point locations with associated sample values, usually of light intensities/colors, transparency, and other control information
- When we sample an image, we sample the point location along the continuous signal and we cannot treat the pixels as little circles or squares, though they may be displayed as such



* Cathode Ray Tube, like those really old TVs

Sample-based Graphics (3/3)

- Samples created directly in Paint-type program, or by sampling of continuous (analog) visual materials (light intensity/color measured at regular intervals) with many devices including:
  - flatbed and drum scanners
    (e.g., https://luminous-landscape.com/drum-scans/)
  - digital still and motion (video) cameras
- Sample values can also be input numerically (e.g., with numbers from computed dataset)
- Once an image is defined as pixel-array, it can be manipulated
  - **Image editing:** changes made by **user**, such as cutting and pasting sections, brush-type tools, and processing selected areas
  - **Image processing:** algorithmic operations that are performed on image (or pre-selected portion of image) without user intervention. Blurring, sharpening, edge-detection, color balancing, rotating, warping. These are front-end processes to Computer Vision, Computational Photography

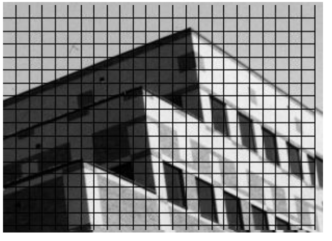
Sampling an Image

- Lets do some sampling of CIT building

3D scene

- A color value is measured at every grid point and used to color corresponding grid square

0 = white, 5 = gray, 10 = black



- Crude sampling and image reconstruction method creates blocky image

What's the Advantage?

- Once image is defined in terms of colors at (x, y) locations on grid, can change image easily by altering location or color values
- E.g., if we reverse our mapping above and make 10 = white and 0 = black, the image would look like this:

- Pixel information from one image can be copied and pasted into another, replacing or combining with previously stored pixels

What's the Disadvantage?

- WYSIAYG (What You See Is All You Get): No additional information
  - No depth information
  - Can't examine scene from different point of view
  - At most can play with the individual pixels or groups of pixels to change colors, enhance contrast, find edges, etc.
  - But increasingly great success in image-based rendering to fake 3D scenes and arbitrary camera positions. New images constructed by interpolation, composition, warping and other operations.
  - For a computational and cognitive science perspective, take James Tompkin's Computer Vision (CSCI1430)

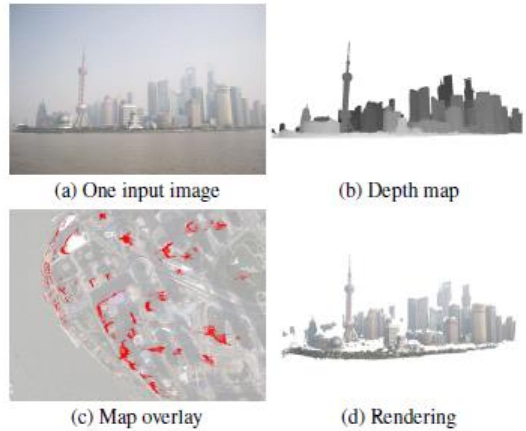
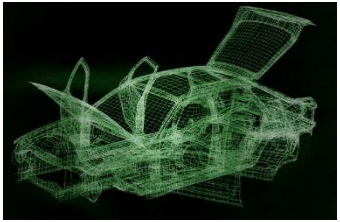


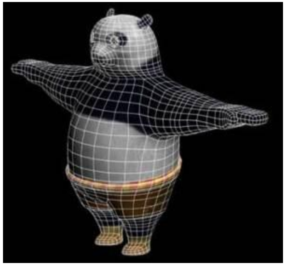
Figure 15: *Results on a challenging unstructured light field, obtained by hand-held capture (a) from a floating boat. (b) A resulting depth map. (c) Overlay of our reconstruction on a satellite image ©2013 DigitalGlobe, Google. (d) Rendering from a novel viewpoint.*

"Scene Reconstruction from High Spatio-Angular Resolution Light Fields" by Kim, Zimmer et al., 2013

Geometry-Based Graphics (1/2)

- Geometry-based graphics (also called scalable vector graphics or object-oriented graphics): geometrical model is created, along with various appearance attributes, and is then sampled for visualization (rendering, a.k.a image synthesis)
  - often some aspect of physical world is visually simulated, or "synthesized"
  - examples of 2D apps: Adobe Illustrator™ and Corel CorelDRAW™
  - examples of 3D apps: Autodesk's AutoCAD™, Autodesk's (formerly Alias|Wavefront's) Maya™, Autodesk's 3D Studio Max™



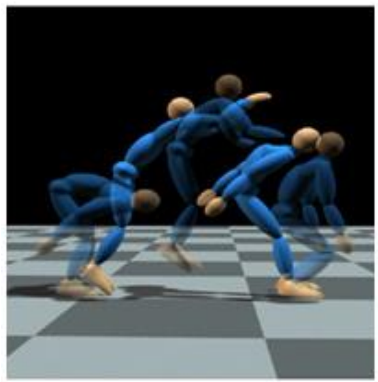


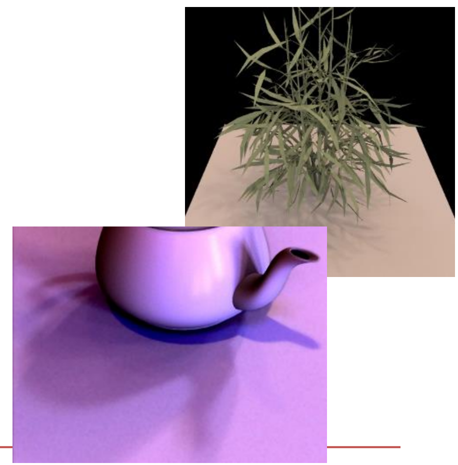
Geometry-Based Graphics (2/2)

- Geometry-based graphics applications
  - Store mathematical descriptions, or "**models**," of geometric elements (lines, polygons, polyhedrons, polygonal meshes…) and associated attributes (e.g., color, material properties).
  - Geometric elements are primitive shapes, **primitives** for short.
  - Images are created via sampling of geometry for viewing, but not stored as part of model.
  - Users cannot usually work directly with individual pixels in geometry-based programs; as user manipulates geometric elements, program resamples and redisplays elements
- Increasingly rendering combines geometry- and sample-based graphics, both as performance hack and to increase quality of final product
  - CG animated characters (geometry) on painted or filmed scene images (samples)

3D Graphics Pipeline

Modeling → **Animation** → **Rendering**





What is Geometric Modeling?

- What is a model?
- Captures salient features (data, behavior) of object/phenomenon being modeled
  - data includes geometry, appearance, attributes…
  - note similarity to OOP ideas
- Modeling allows us to cope with complexity
- Our focus: modeling and viewing simple everyday objects
- Consider this:
  - Through 3D computer graphics, we have abstract, easily changeable 3D forms, for the first time in human history
    - Has revolutionized working process of many fields – science, engineering, industrial design, architecture, commerce, entertainment, etc. Profound implications for visual thinking and visual literacy
    - "Visual truth" is gone in the Photoshop and FX-saturated world (but consider painting and photography…) – seeing no longer is believing…(or shouldn't be!)

Modeling vs. Rendering

- Modeling
  - Create models
  - Apply materials to models
  - Place models around scene
  - Place lights in scene
  - Place the camera
- **Rendering**

  Take “picture” with camera
- Both can be done with commercial software:

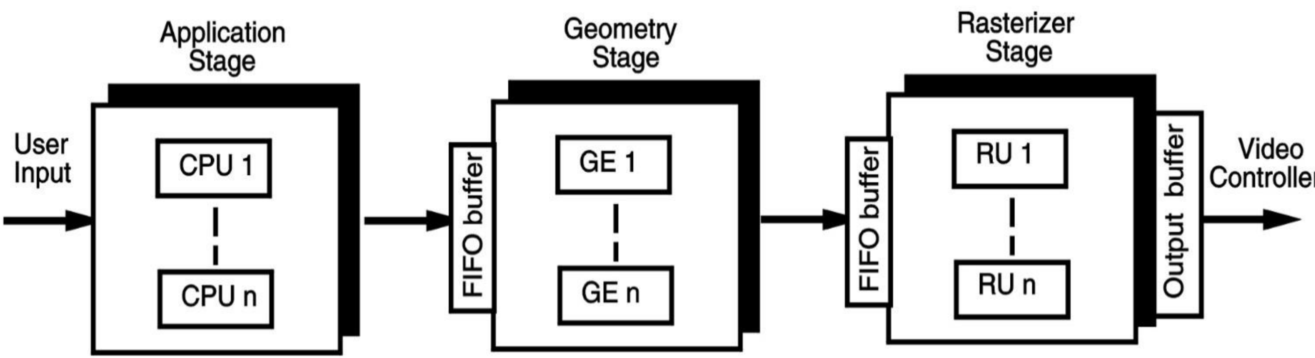
  Autodesk Maya™ ,3D Studio Max™, Blender™, etc.

CS128 lighting assignment by Patrick Doran, Spring 2009

RENDERING

GRAPHICS / RENDERING PIPELINE

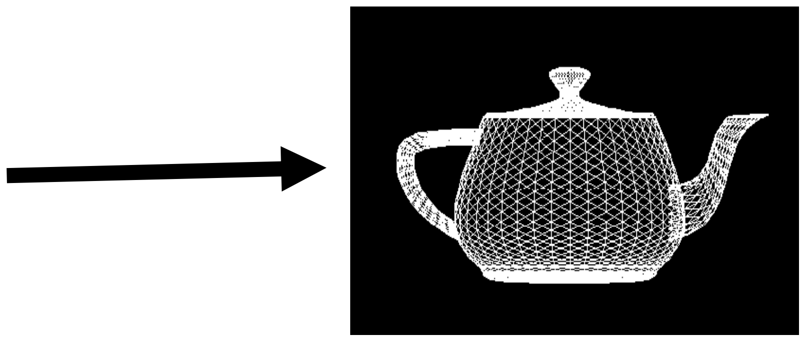
- There are three stages
  - Application Stage
  - Geometry Stage
  - Rasterization Stage



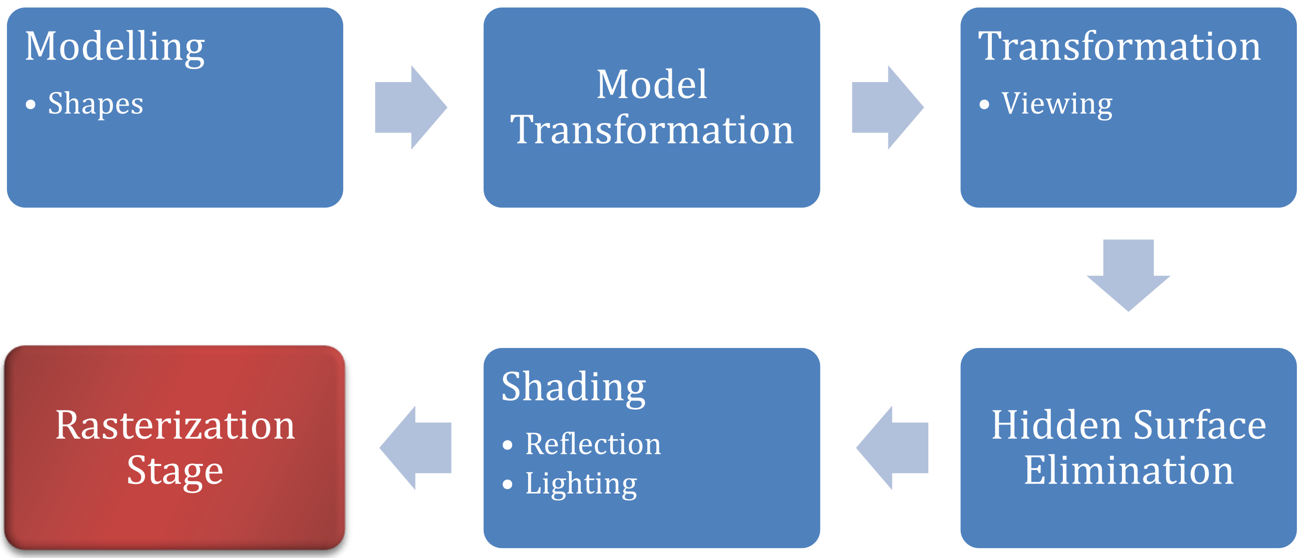
APPLICATION STAGE

- Entirely done in software by the CPU
- Read Data
  - the world geometry database,
  - User's input by mice, trackballs, trackers, or sensing gloves
- In response to the user's input, the application stage change the view or scene

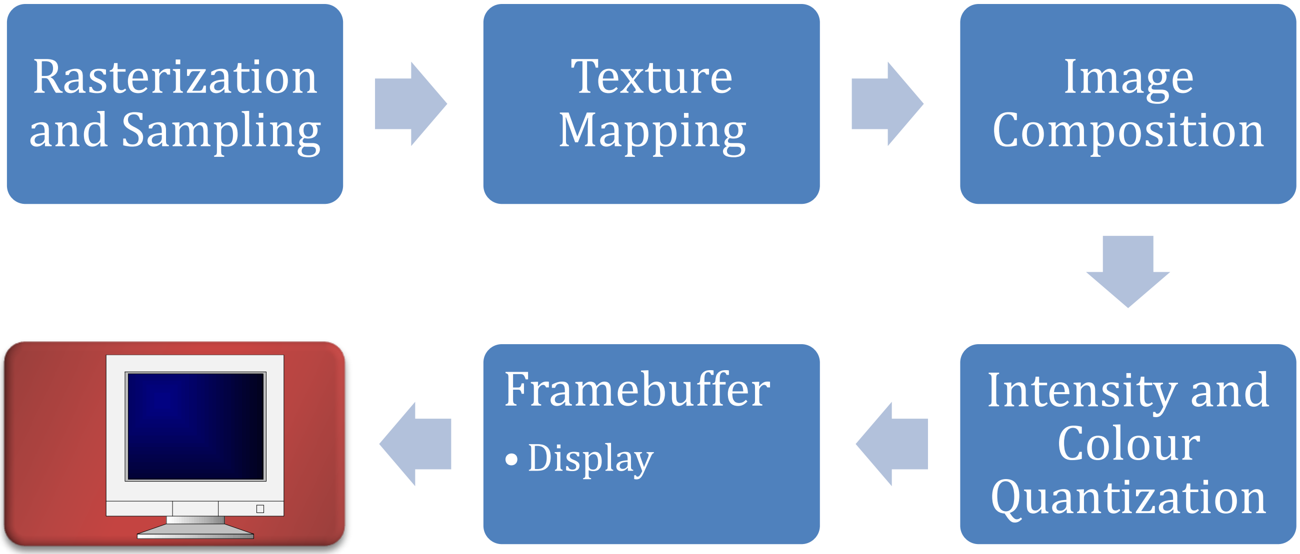
```
3.382035 2.446498 -0.064692
3.382035 2.446498 0.064692
3.392006 2.474995 -0.050004
3.392006 2.474995 0.050004
3.400000 2.446800 0.000000
3.406947 2.462176 -0.061668
3.406947 2.462176 0.061668
3.408000 2.475600 0.000000
3.408000 2.475600 0.000000
3.411237 2.476753 -0.054000
3.411237 2.476753 0.054000
3.416450 2.472371 -0.057996
3.416450 2.472371 0.057996
3.424875 2.462606 0.000000
3.428125 2.477344 0.000000
3.428125 2.477344 0.000000
3.434000 2.472900 0.000000

2909 2921 2939
2939 2931 2909
2869 2877 2921
2921 2909 2869
2819 2827 2877
2877 2869 2819
2737 2747 2827
2827 2819 2737
2669 2673 2747
2747 2737 2669
2567 2575 2673
2673 2669 2567
```



GEOMETRY STAGE

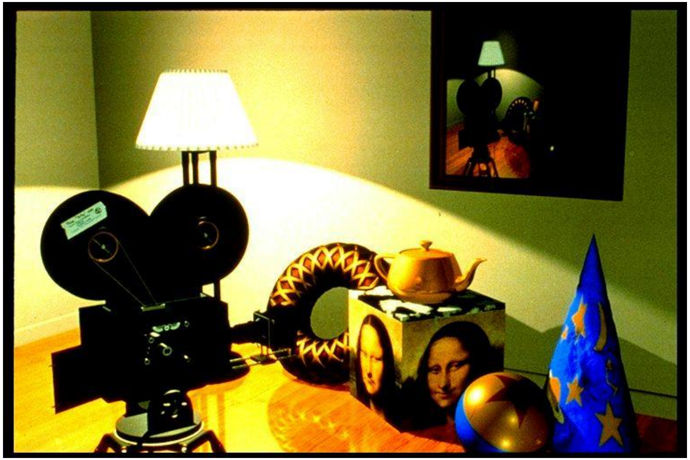


RASTERIZATION STAGE

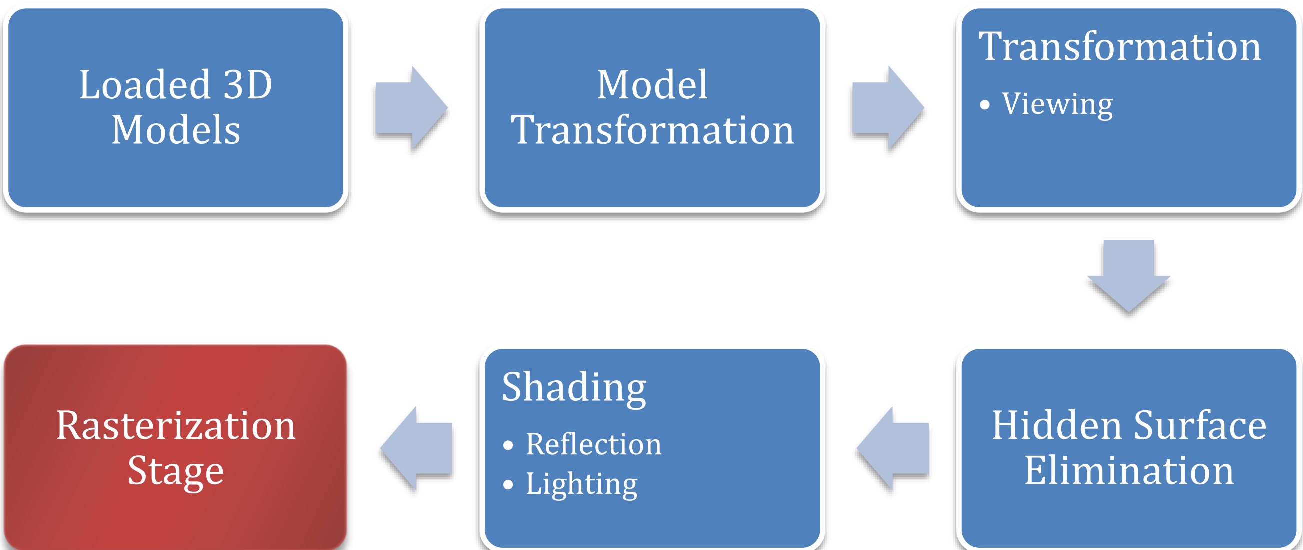


EXAMPLE

The scene we are trying to represent:

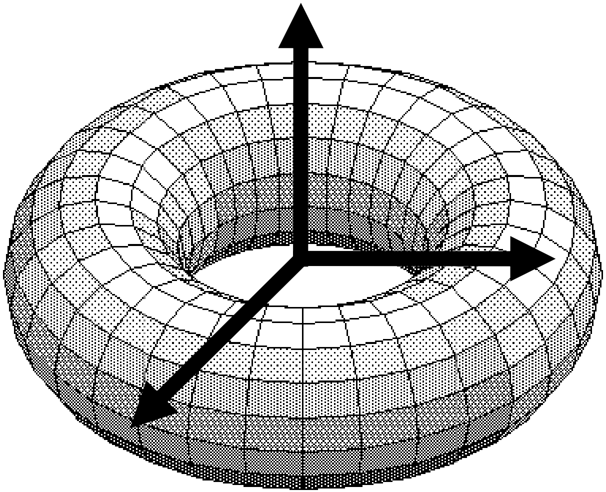


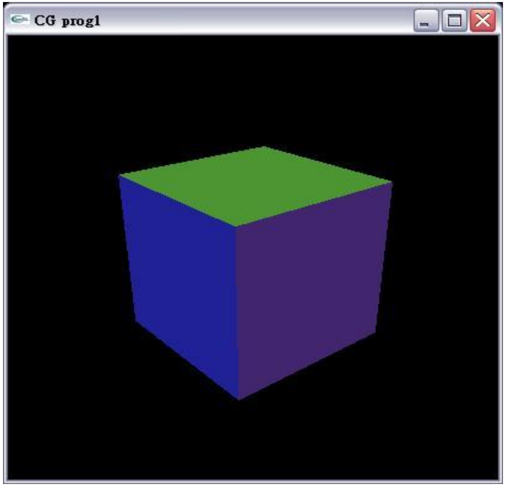
GEOMETRY PIPELINE



PREPARING SHAPE MODELS

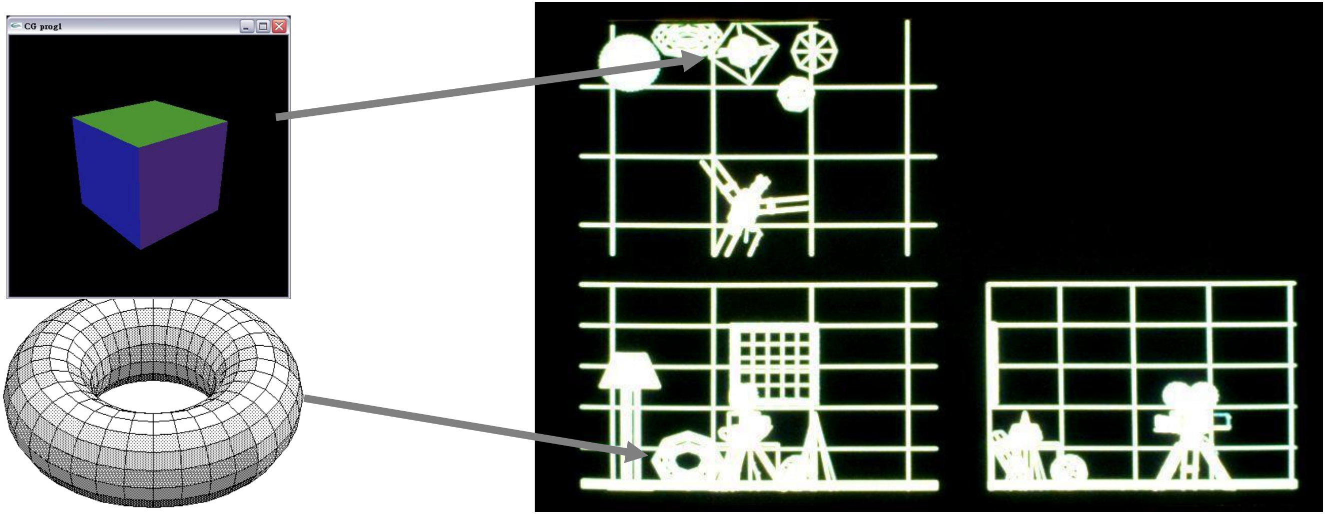
- Designed by polygons, parametric curves/surfaces, implicit surfaces and etc.
- Defined in its own coordinate system





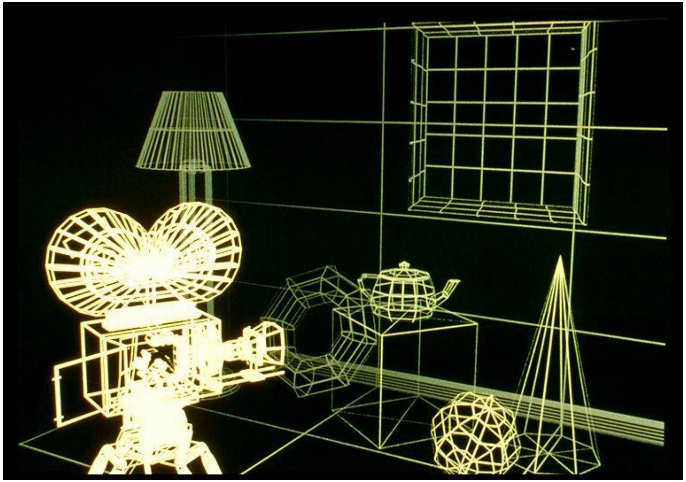
MODEL TRANSFORMATION

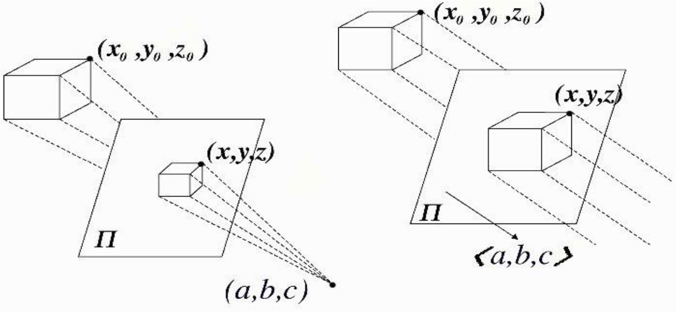
- Objects put into the scene by applying
  - Translation, scaling and rotation

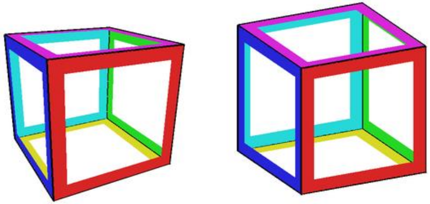


PERSPECTIVE PROJECTION

- We want to create a picture of the scene viewed from the camera
- We apply a perspective projection to convert the 3D coordinates to 2D coordinates of the screen
- Objects far away appear smaller, closer objects appear bigger

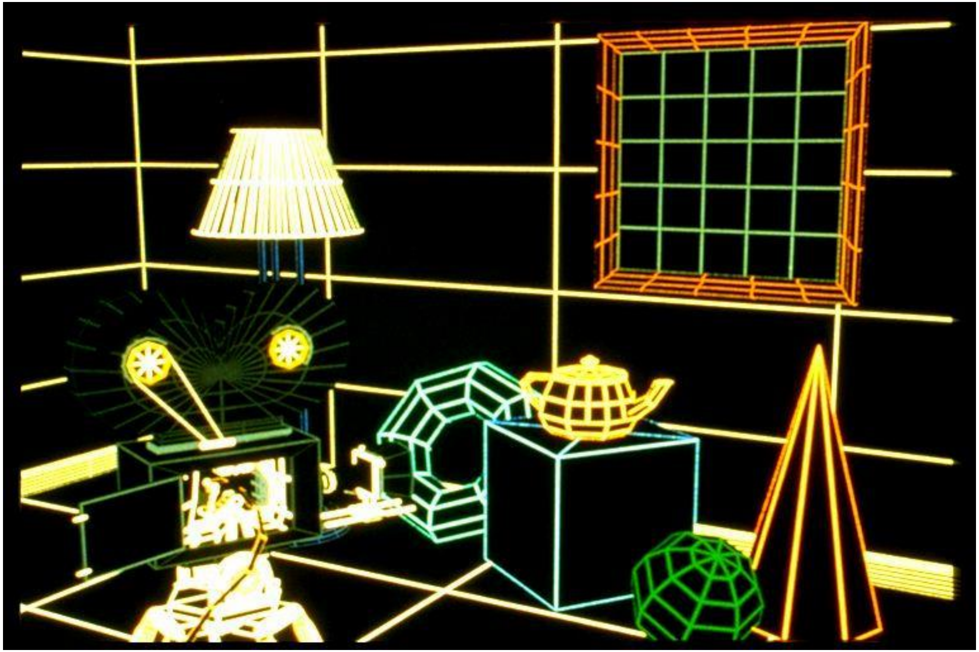






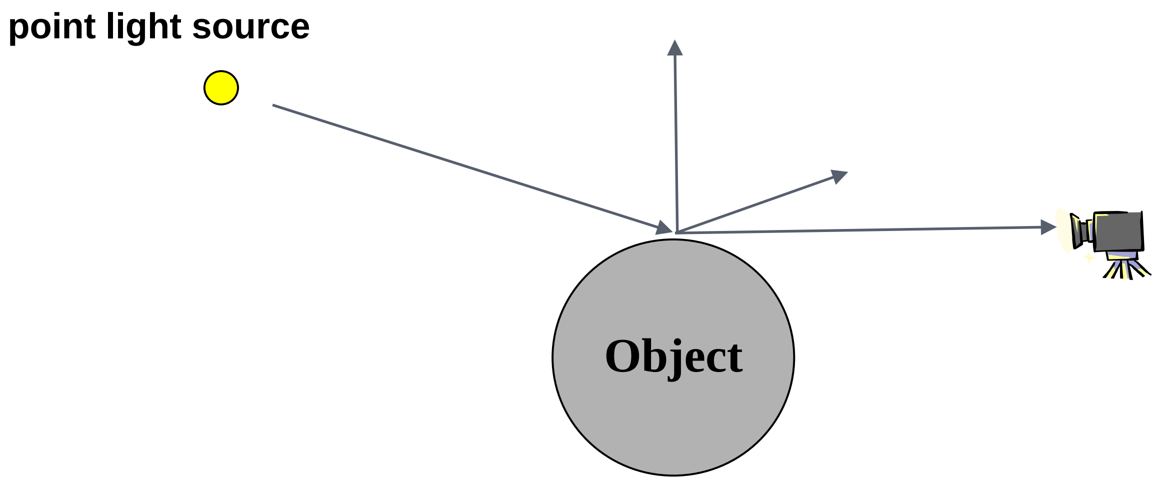
HIDDEN SURFACE REMOVAL

- Objects occluded by other objects must not be drawn



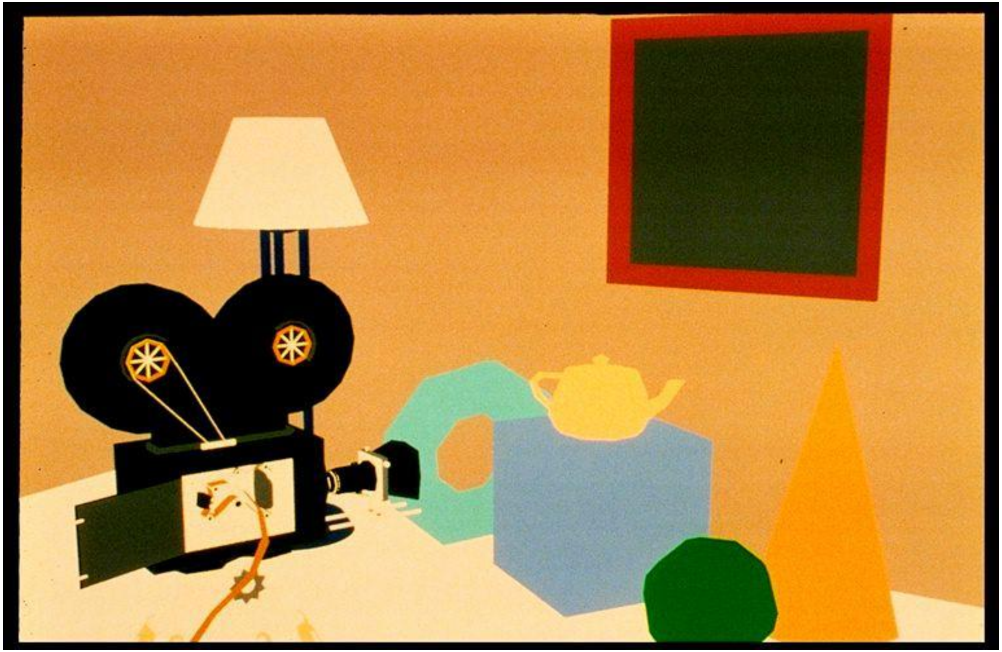
SHADING

- Now we need to decide the colour of each pixels taking into account the object's colour, lighting condition and the camera position



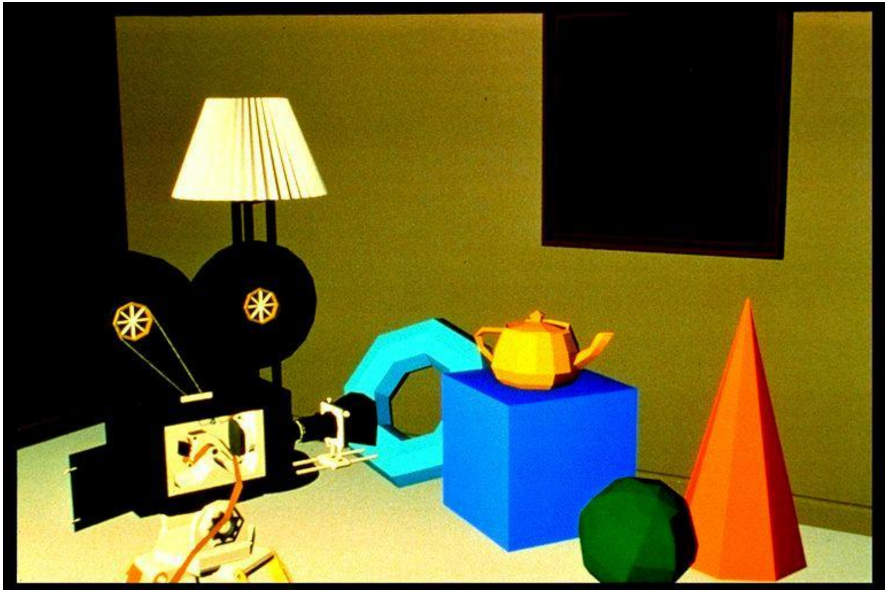
SHADING : CONSTANT SHADING - AMBIENT

- Objects colours by its own colour



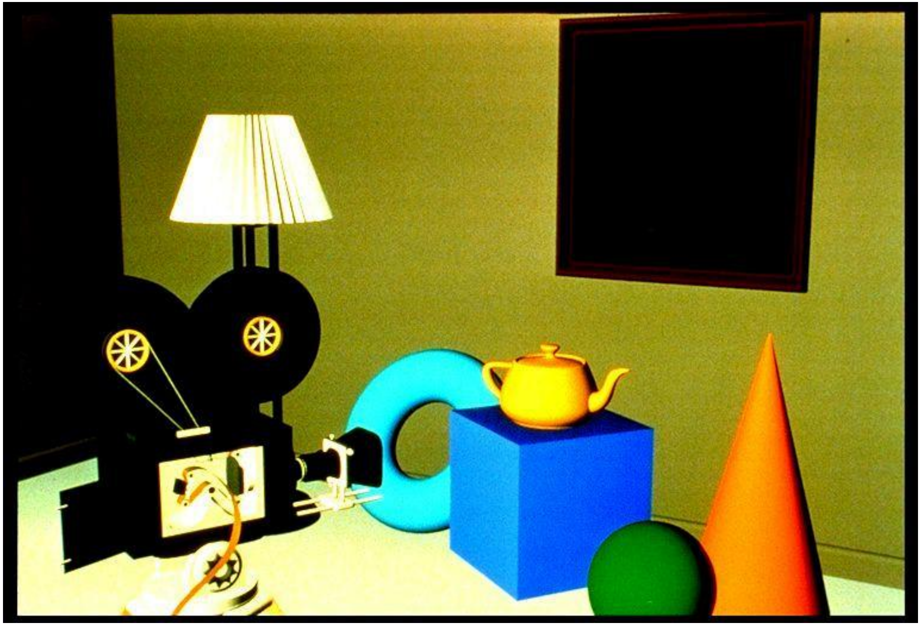
SHADING – FLAT SHADING

- Objects coloured based on its own colour and the lighting condition
- One colour for one face

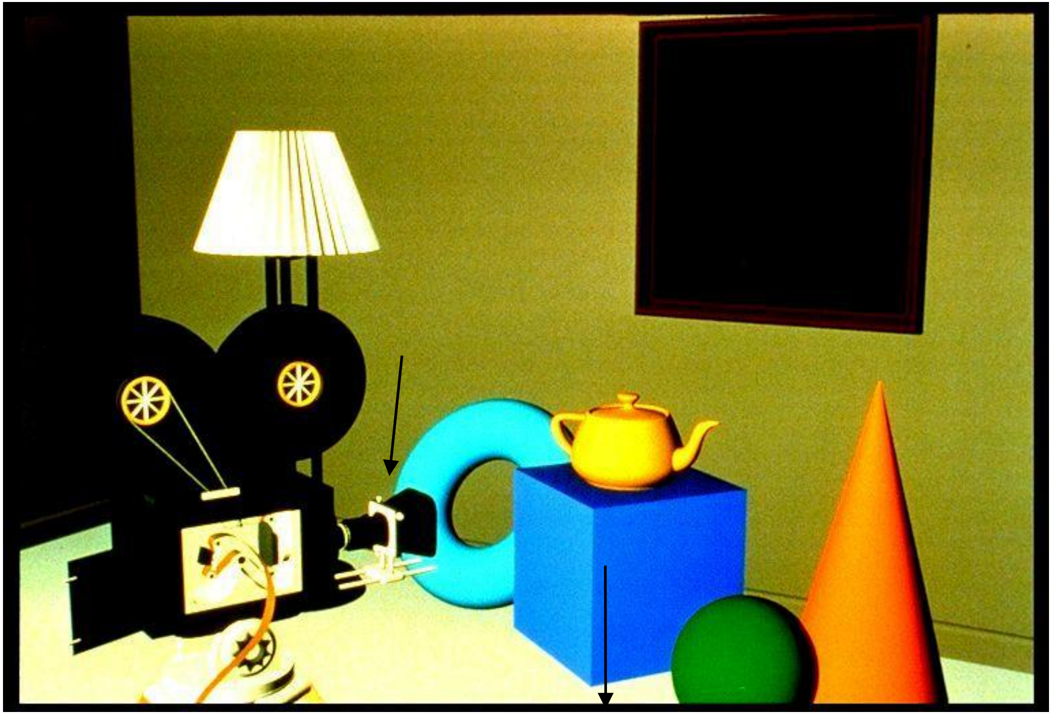


GOURAUD SHADING, NO SPECULAR HIGHLIGHTS

- Lighting calculation per vertex

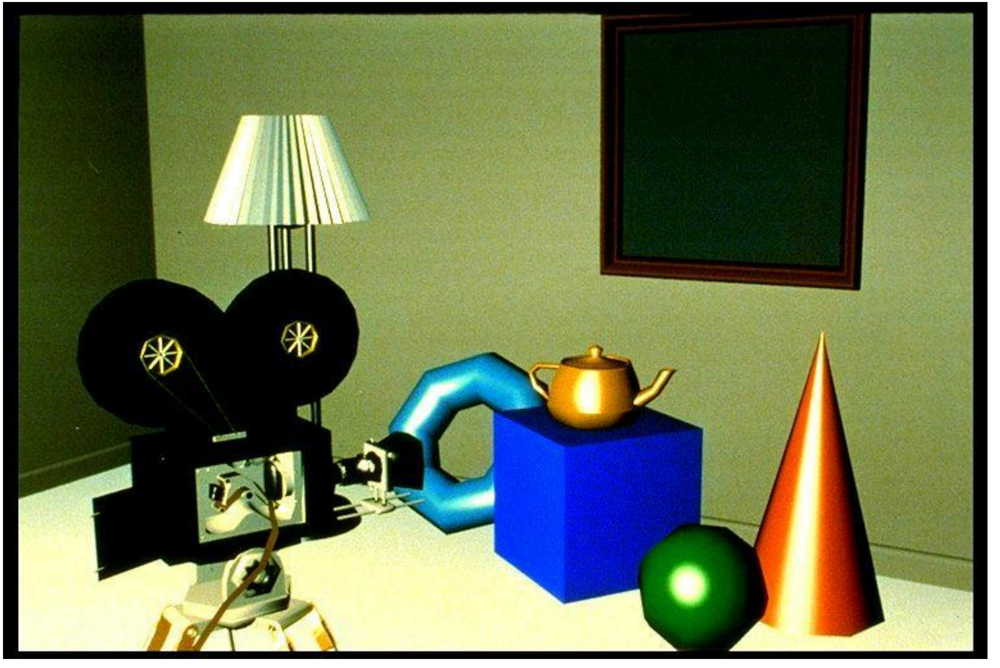


SHAPES BY POLYNOMIAL SURFACES

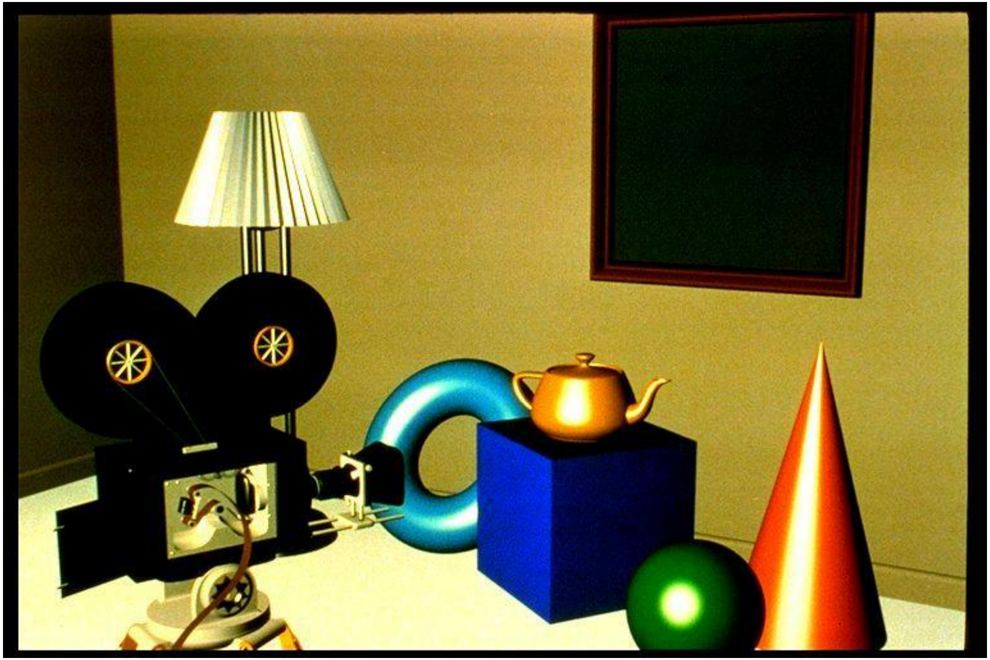


SPECULAR HIGHLIGHTS ADDED

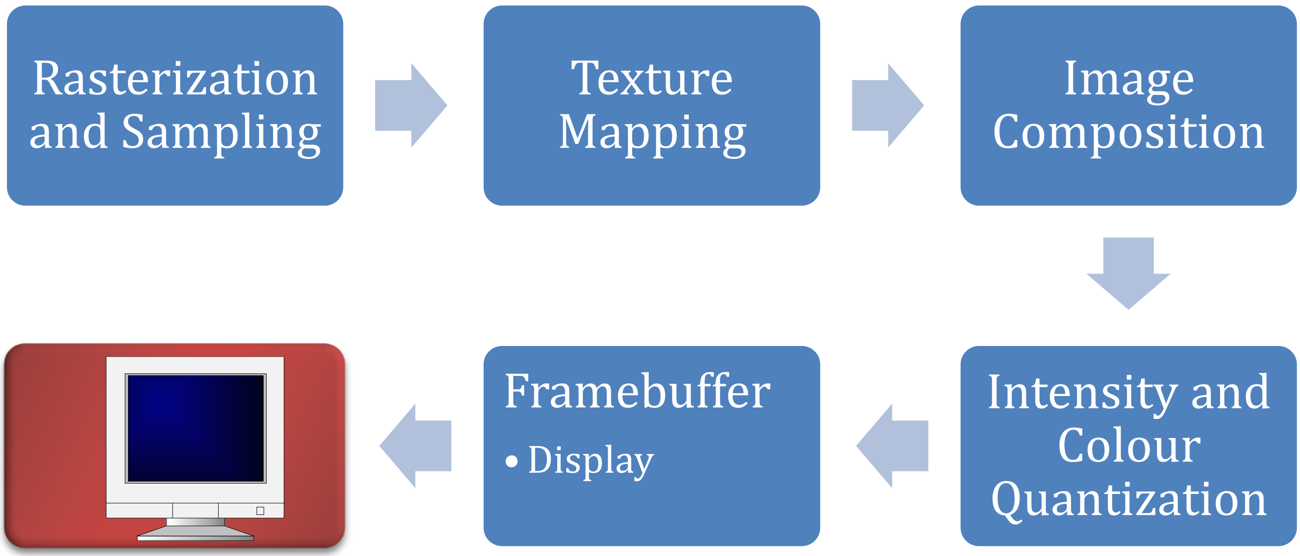
- Light perfectly reflected in a mirror-like way



PHONG SHADING

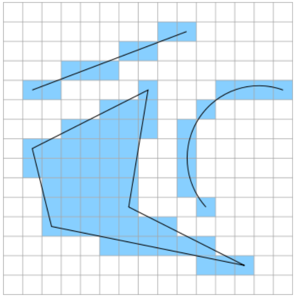


NEXT, THE IMAGING PIPELINE



RASTERIZATION

- Converts the vertex information output by the geometry pipeline into pixel information needed by the video display
- Aliasing: distortion artefacts produced when representing a high-resolution signal at a lower resolution.
- Anti-aliasing : technique to remove aliasing

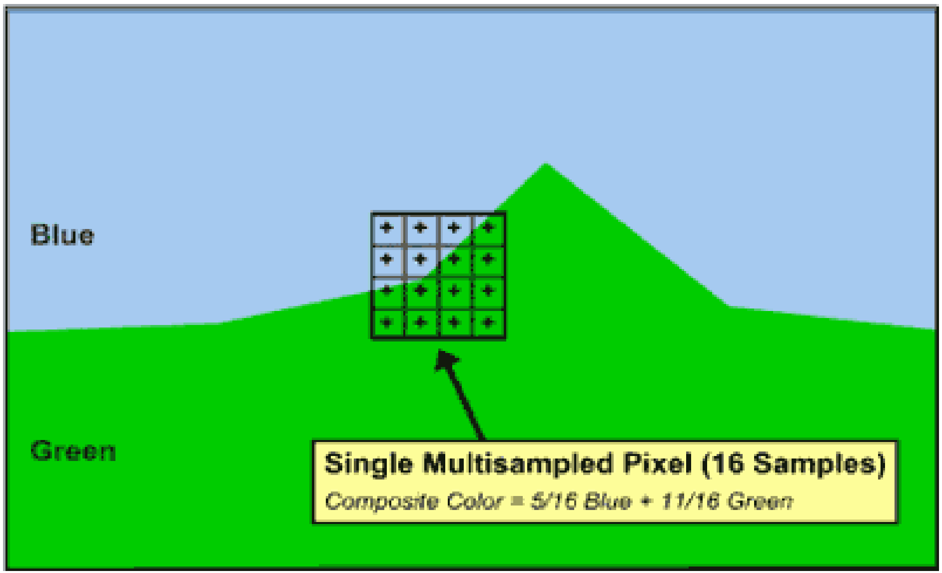


ANTI-ALIASING

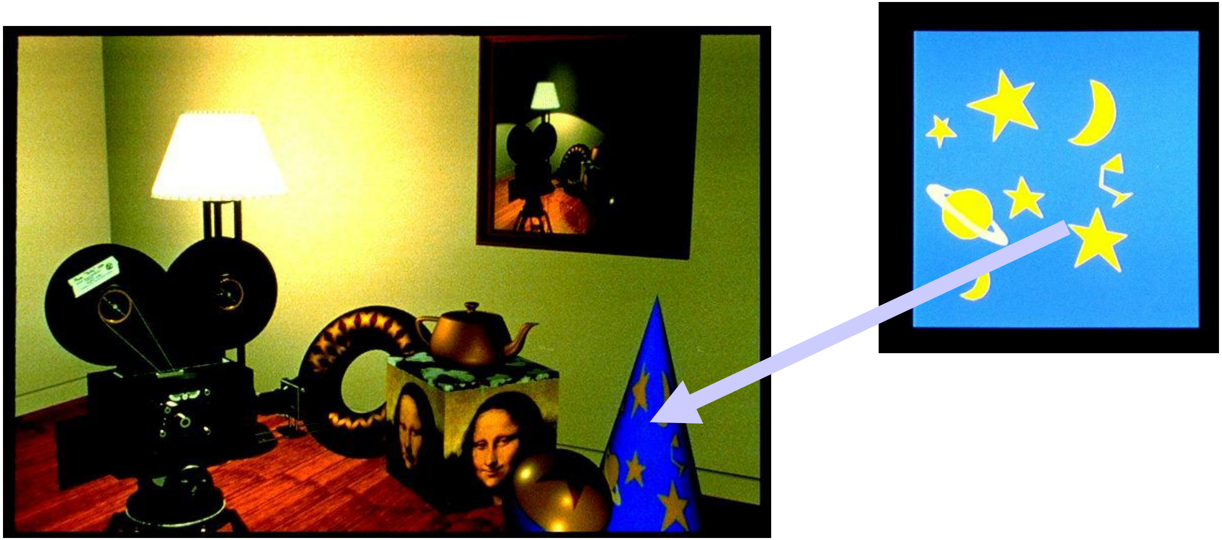
Aliased polygons (jagged edges)

Anti-aliased polygons

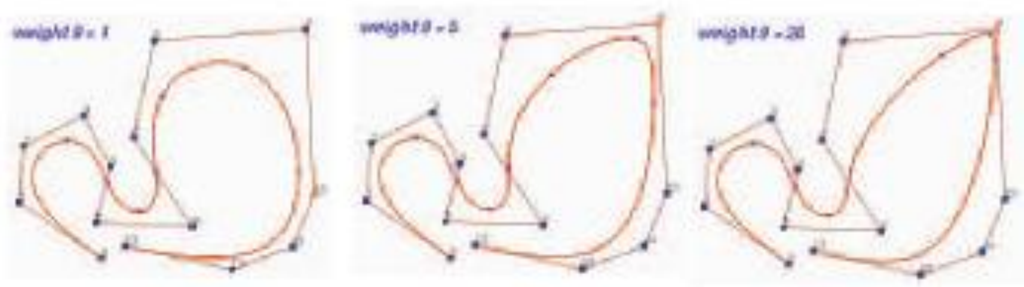
- How is anti-aliasing done? Each pixel is subdivided
- (sub-sampled) in n regions, and each sub-pixel has a color;
- Compute the average color value

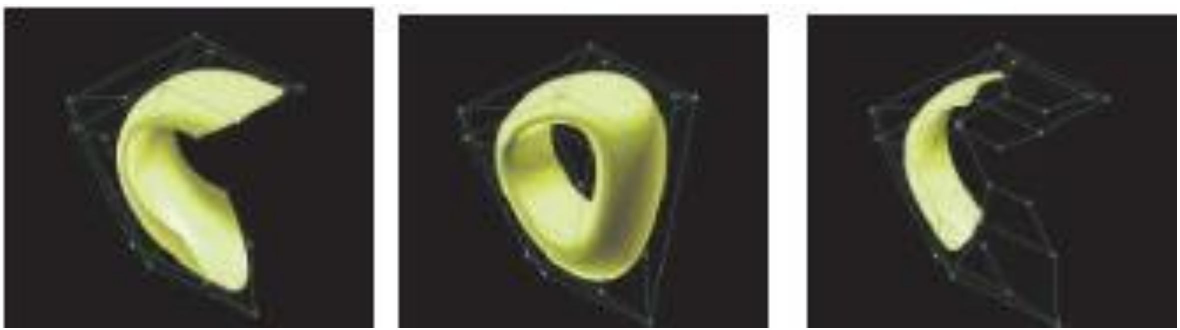


TEXTURE MAPPING



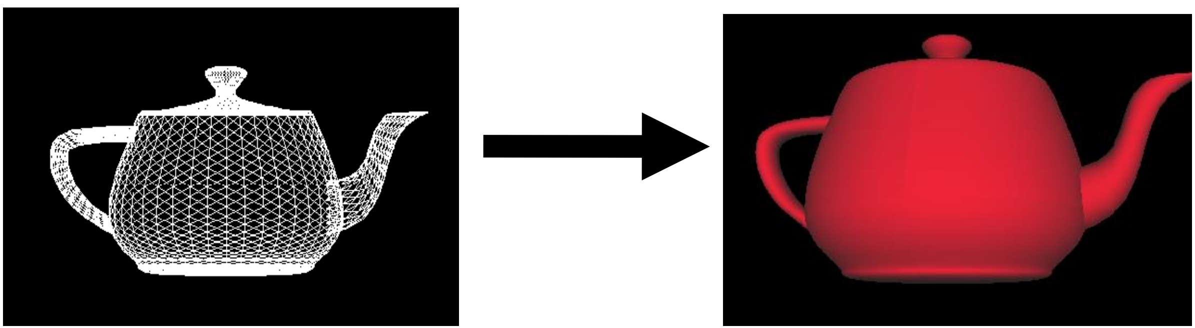
POLYNOMIAL CURVES, SURFACES





First Practical

- Write a program that renders an image of a teapot and outputs it into an image file
- I prepared a demo program to load a 3D model and draw the edges
- You update it so that the surface appears shaded
- See the course website for the details



Summary

- The course is about algorithms, not applications
  - Lots of mathematics
- Graphics execution is a pipelined approach
- Basic definitions presented
- Some support resources indicated

END

Any Question?