COMPUTER GRAPHICS

Sławomir Samolej Ph.D.
D102 C, tel.: 865 1766,
email: ssamolej@prz-rzeszow.pl
www: ssamolej.prz-rzeszow.pl

Prerequisities

- Some programming skills in C (or C++)
- Basic Data Structures
  - Linked lists
  - Arrays
- Geometry
- Simple Linear Algebra

Course programme

Lectures:

2. Introduction to Computer Graphics, Windows Programming Principles
3. OpenGL – OpenGL for Windows, Primitives and Attributes,
4. OpenGL – Coordinate transformations,
5. OpenGL – Colour and Shading,
6. OpenGL – Lighting,
7. OpenGL – Texture Mapping
8. OpenGL – Quadrics, Blending, Fog, Curves and Surfaces
9. OpenGL – Some Advanced Tricks, Game Programming
10. Graphic Files Structures,

Labs:

12. Windows Programming
13. Building 3D Models
14. Coordinate transformations
15. Lighting, Colour and Shading
16. Texture Mapping
17. Blending, Fog, Curves and Surfaces

References

1. Angel Edward : „Interactive Computer Graphics: A Top-Down Approach with OpenGL™”, Addison-Wesley 2006.
2. Richard S. Wright jr, Michael Sweet: „ OpenGL™ Superbible”, Sams 2004.
3. „OpenGL Programming Guide” (www.opengl.org).
4. „OpenGL Reference Manual” (www.opengl.org).
5. Kevin Hawkins, Dave Astle: „OpenGL Game Programming”, Premier Press 2004.
6. David M. Bourg: „Physics for Game Developers”, O’Reilly 2001.
7. Petzold Charles, „ Programming Windows”, Microsoft Press 1998.

Web Links

- http://ssamolej.prz-rzeszow.pl

OpenGL:

- http://www.opengl.org
- http://www.codeproject.com/opengl

Game programming:

- http://www.gamedev.net
- http://nehe.gamedev.net
- http://www.gametutorials.com

Software tools

- OpenGL library (a part of Windows)
- Any C/C++ Compiler
- There are OpenGL implementations for most operating systems, including:
  - Linux
  - Unix

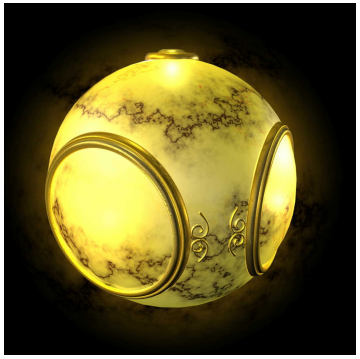
Introduction - Definitions

Computer graphics **deals with all aspects of creating images with a computer**

- **Hardware**
- **Software**
- **Applications**

Example

- **Where did this image come from?**



- **What hardware/software did we need to produce it?**

Introduction - Definitions

Answer

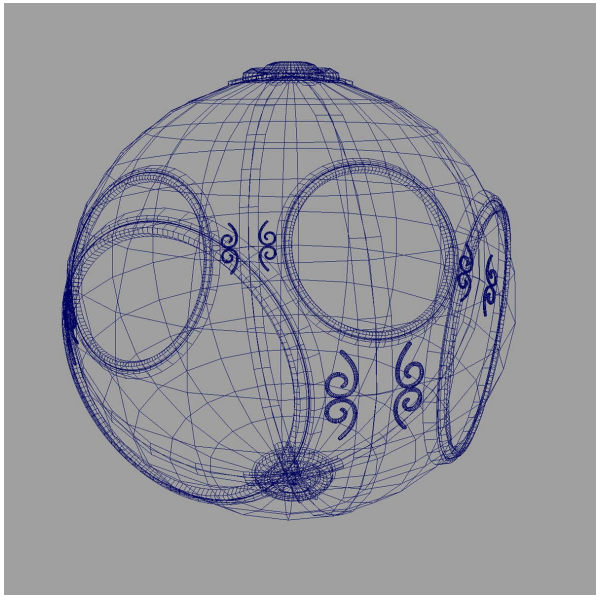
- **Application:** The object is an artist's rendition of the sun for an animation to be shown in a domed environment (planetarium)
- **Software:** Maya for modeling and rendering but Maya is built on top of OpenGL
- **Hardware:** PC with graphics card for modeling and rendering

Some History

1960-1970:

Wireframe graphics - Draw only lines

- **Sketchpad** – first interactive graphic programme; Loop:
  - Display something
  - User moves light pen
  - Computer generates new display
- **Display Processors** - Rather than have the host computer try to refresh display use a special purpose computer called a *display processor* (DPU)
- **Storage tube** – when a portion of the screen is illuminated by the CRT's electron gun, it stays lit until a screen erase command is given



wireframe representation of sun object

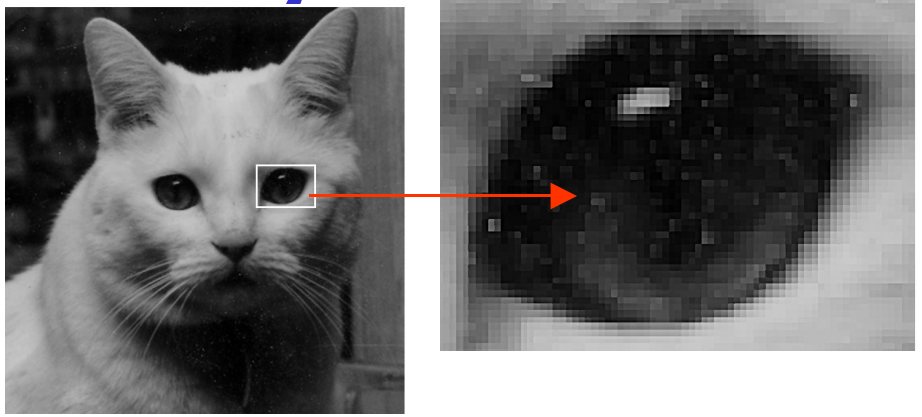
Some History

1970-1980:

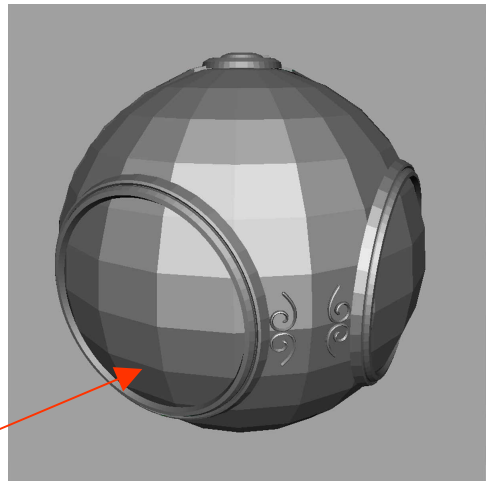
- **Raster Graphics**
- **Beginning of graphics standards**
  - -IFIPS
    - GKS: European effort
      - –Becomes ISO 2D standard
    - Core: North American effort
      - – 3D but fails to become ISO standard
- **Workstations and PCs**

Some History

Raster Graphics:



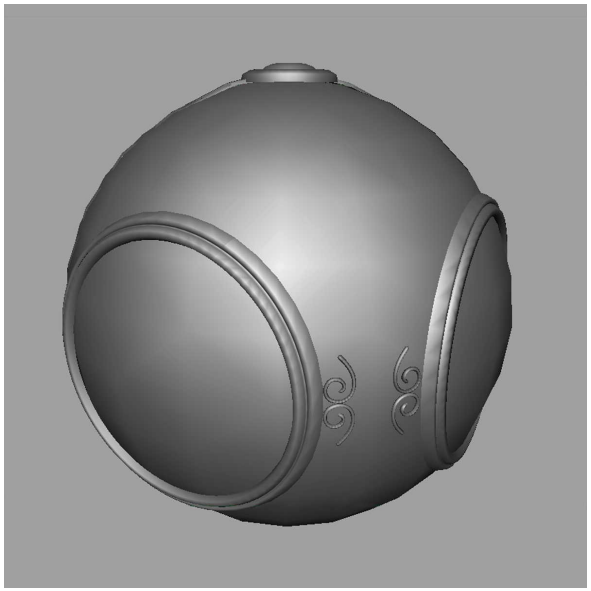
- Image produced as an array (the *raster*) of picture elements (*pixels*) in the *frame buffer*
  - Frame buffer has the depth (number of bits for one pixel) and the resolution (number of pixels in the framebuffer)
  - 1-bit-deep framebuffer allows 2 colors, 8-bit-deep allows 256 colors; full color system has 24 or more bit-deep framebuffer.
- Allows us to go from lines and wire frame images to filled polygons



Some History

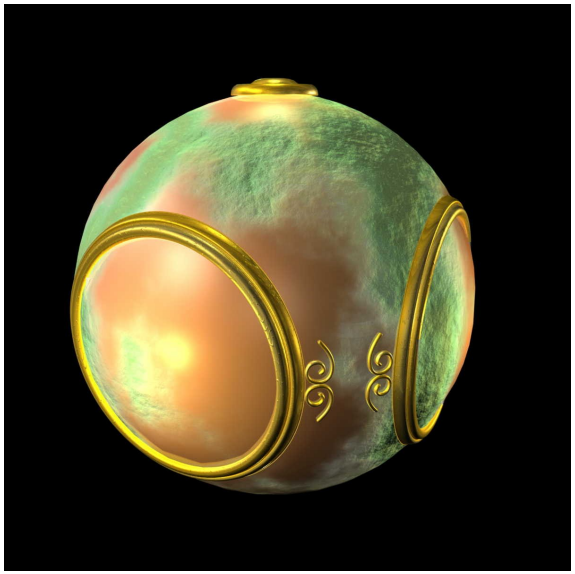
1980-1990 (1):

- **Realism comes to computer graphics**



smooth shading

environment mapping



bump mapping

Some History

1980-1990 (2):

- Special purpose hardware
  - -Silicon Graphics geometry engine
    - VLSI implementation of graphics pipeline
- Industry-based standards
  - -PHIGS
  - -RenderMan
- Networked graphics: X Window System
- Human-Computer Interface (HCI)

Some History

1990-2000:

- **OpenGL API**
- **Completely computer-generated feature-length movies (Toy Story) are successful**
- **New hardware capabilities**
  - -Texture mapping
  - -Blending
  - -Accumulation, stencil buffers

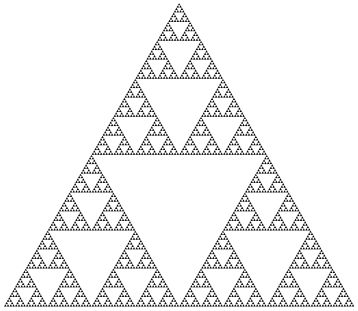
Some History

2000- :

- **Photorealism**
- **Graphics cards for PCs dominate market**
  - **-Nvidia, ATI, 3DLabs**
- **Game boxes and game players determine direction of market**
- **Computer graphics routine in movie industry: Maya, Lightwave**
- **Programmable pipelines**

Polish accents:

- **Wacław Franciszek Sierpiński**
  - **proposed a mathematical object called Sierpiński gasket –** one of the first fraktals
- **Marek Hołyński**
  - **proposed software SGI workstations**



Sierpiński gasket

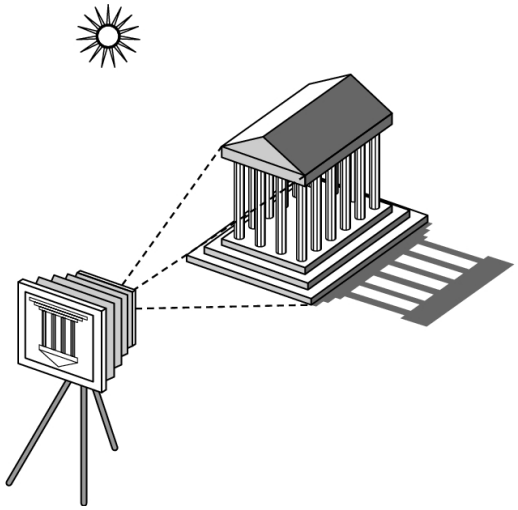
Applications of Computer Graphics

- **Display of information**
  - **Plots; Maps**
  - **Computer tomography, magnetic resonanse imaging, ultrasound**
  - **Molecular biology, phisics, bioinformatics – representing huge amount of data as graphical patterns.**
- **Design**
  - **CAD: Architecture, Mechanics, VLSI circuts,**
- **Simulation and animation**
  - **Flight simulators**
  - **Animated television, motion-picture, advertising industries**
  - **Virtual reality; Games**
- **User Interface**
  - **X Window/Microsoft Windows/Macintosh graphic interfaces**
  - **CAD software interfaces**

Image Formation

- In computer graphics, we form images which are generally two dimensional using a process analogous to how images are formed by physical imaging systems
  - -Cameras
  - -Microscopes
  - -Telescopes
  - -Human visual system

Elements of Image Formation



- Objects
- Viewer
- Light source(s)

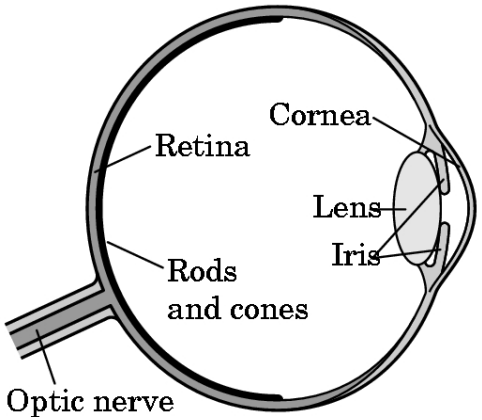
- Attributes that govern how light interacts with the materials in the scene
- Note the independence of the objects, the viewer, and the light source(s)

Light

- *Light* is the part of the electromagnetic spectrum that causes a reaction in our visual systems
- Generally these are wavelengths in the range of about 350-750 nm (nanometers)
- Long wavelengths appear as reds and short wavelengths as blues

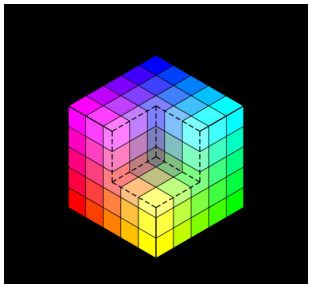
Three-Color Theory

- Human visual system has two types of sensors
  - -Rods: monochromatic, night vision
  - -Cones
    - Color sensitive
    - Three types of cones
    - Only three values (the *tristimulus* values) are sent to the brain

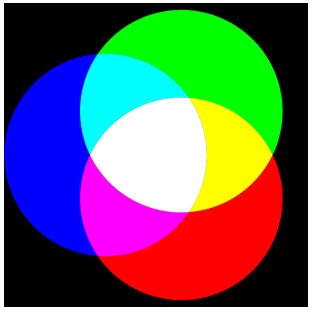


- Need only match these three values
  - -Need only three *primary* colors

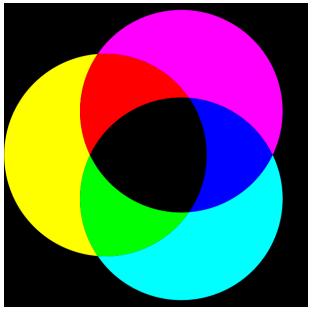
Additive and Subtractive Color



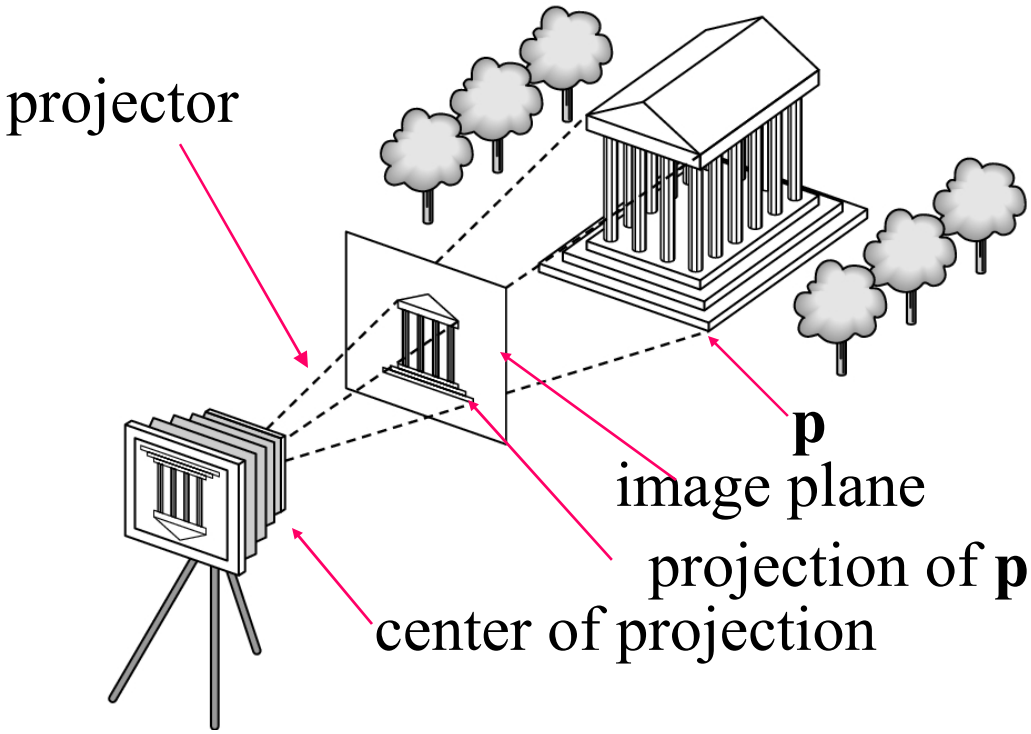
- Additive color
  - -Form a color by adding amounts of three primaries
    - CRTs, projection systems, positive film
  - -Primaries are Red (R), Green (G), Blue (B)



- Subtractive color
  - -Form a color by filtering white light with cyan (C), Magenta (M), and Yellow (Y) filters
    - Light-material interactions
    - Printing
    - Negative film



Synthetic Camera Model



Advantages

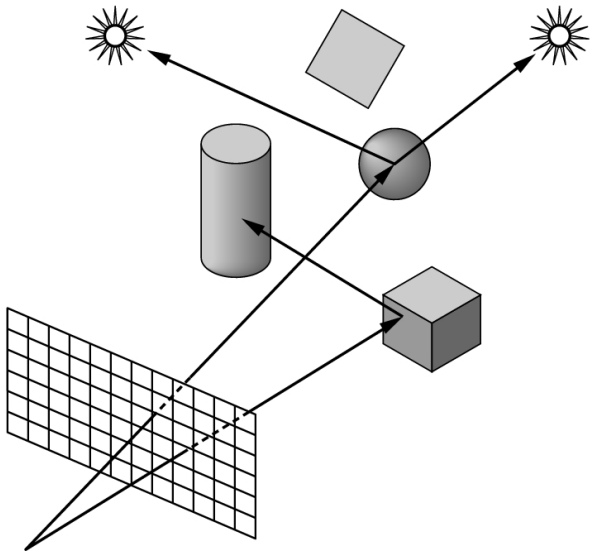
- Separation of objects, viewer, light sources
- Two-dimensional graphics is a special case of three-dimensional graphics
- Leads to simple software API
  - Specify objects, lights, camera, attributes
  - Let implementation determine image
- Leads to fast hardware implementation

Image Formation Revisited

- Can we mimic the synthetic camera model to design graphics hardware software?
- Application Programmer Interface (API)
  - Need only specify
    - Objects
    - Materials
    - Viewer
    - Lights
- But how is the API implemented?

Physical Approaches

- **Ray tracing**: follow rays of light from center of projection until they either are absorbed by objects or go off to infinity
  - -Can handle global effects
    - Multiple reflections
    - Translucent objects
  - -Slow
  - -Must have whole data base available at all times



- **Radiosity**: Energy based approach
  - -Very slow

Practical Approach

- Process objects one at a time in the order they are generated by the application
  - -Can consider only local lighting
- Pipeline architecture

- All steps can be implemented in hardware on the graphics card

Vertex Processing

- Much of the work in the pipeline is in converting object representations from one coordinate system to another
  - -Object coordinates
  - -Camera (eye) coordinates
  - -Screen coordinates
- Every change of coordinates is equivalent to a matrix transformation
- Vertex processor also computes vertex colors

Projection

- *Projection* is the process that combines the 3D viewer with the 3D objects to produce the 2D image

  -Perspective projections: all projectors meet at the center of projection

  -Parallel projection: projectors are parallel, center of projection is replaced by a direction of projection

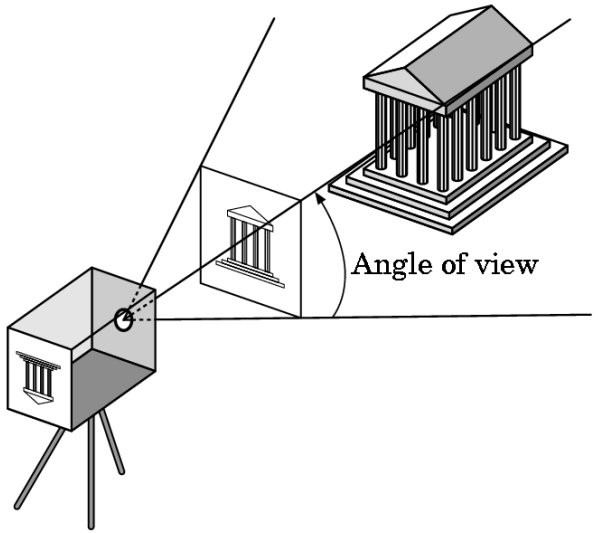
Primitive Assembly

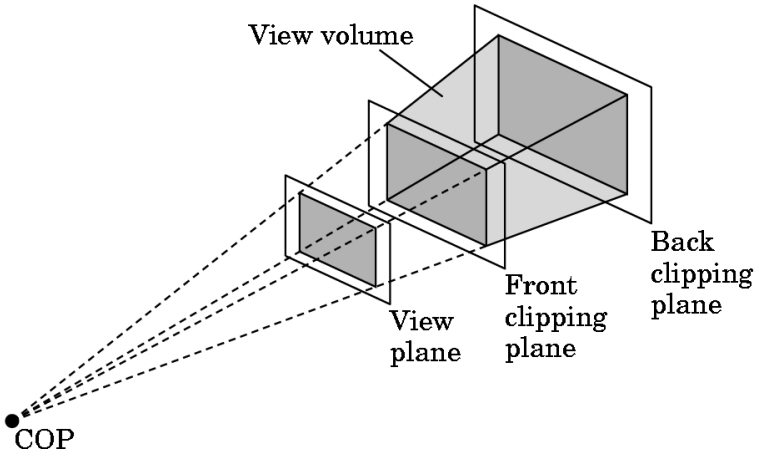
- Vertices must be collected into geometric objects before clipping and rasterization can take place
  - -Line segments
  - -Polygons
  - -Curves and surfaces

Clipping

- Just as a real camera cannot “see” the whole world, the virtual camera can only see part of the world or object space

  -Objects that are not within this volume are said to be *clipped* out of the scene





Rasterization

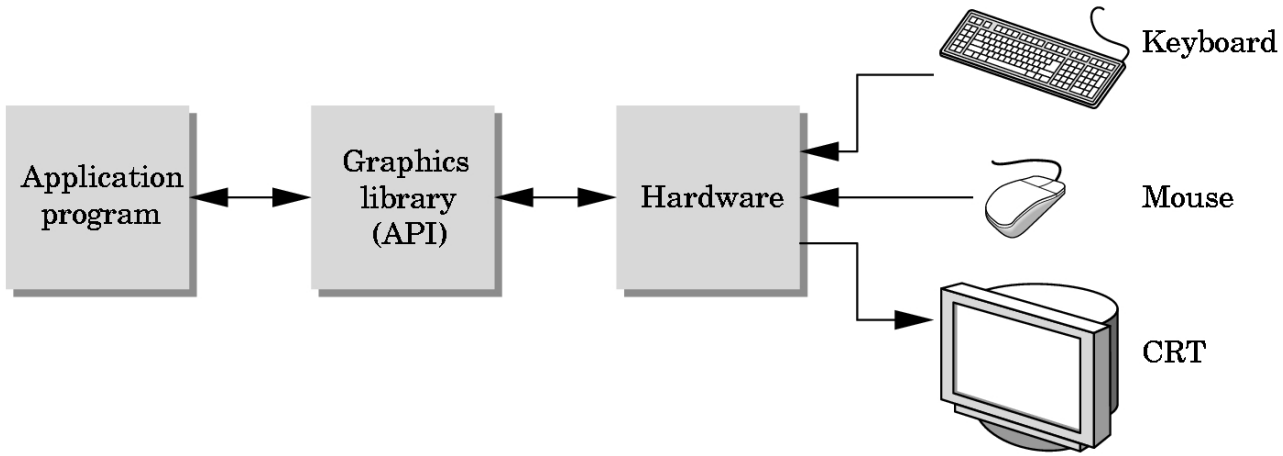
- If an object is not clipped out, the appropriate pixels in the frame buffer must be assigned colors
- Rasterizer produces a set of fragments for each object
- Fragments are "potential pixels"
  - -Have a location in frame bufffer
  - -Color and depth attributes
- Vertex attributes are interpolated over objects by the rasterizer

Fragment Processing

- Fragments are processed to determine the color of the corresponding pixel in the frame buffer
- Colors can be determined by texture mapping or interpolation of vertex colors
- Fragments may be blocked by other fragments closer to the camera
  - -Hidden-surface removal

The Programmer's Interface

- Programmer sees the graphics system through a software interface: the Application Programmer Interface (API)



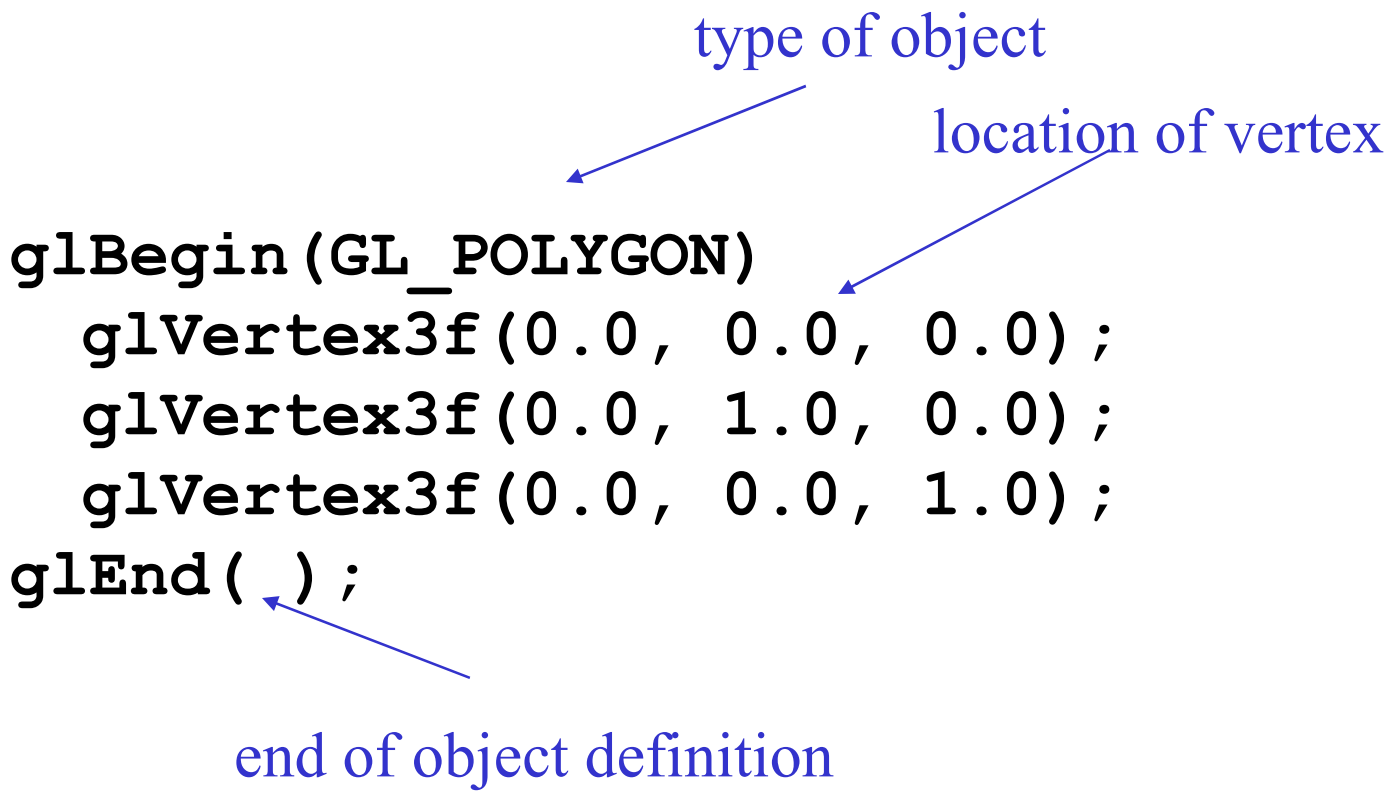
API Contents

- Functions that specify what we need to form an image
  - -Objects
  - -Viewer
  - -Light Source(s)
  - -Materials
- Other information
  - -Input from devices such as mouse and keyboard
  - -Capabilities of system

Object Specification

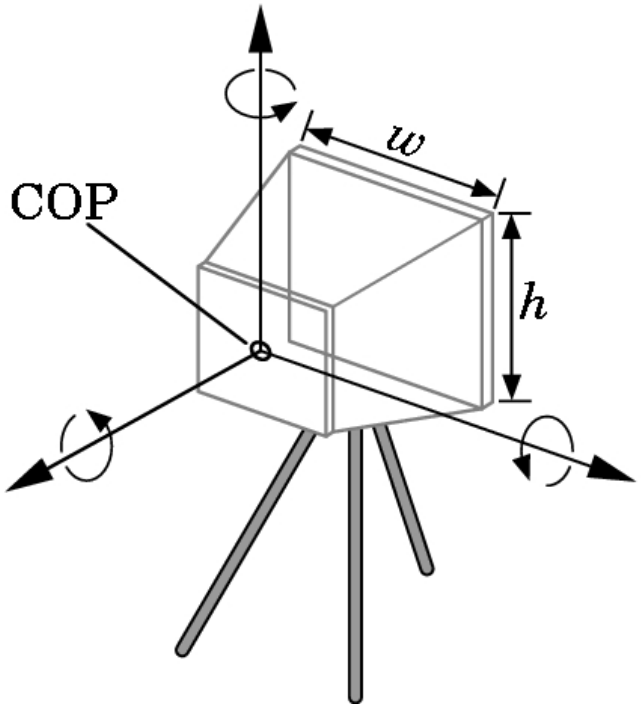
- Most APIs support a limited set of primitives including
  - -Points (0D object)
  - -Line segments (1D objects)
  - -Polygons (2D objects)
  - -Some curves and surfaces
    - Quadrics
    - Parametric polynomials
- All are defined through locations in space or *vertices*

Example


type of object
location of vertex
glBegin(GL_POLYGON)
glVertex3f(0.0, 0.0, 0.0);
glVertex3f(0.0, 1.0, 0.0);
glVertex3f(0.0, 0.0, 1.0);
glEnd( );
end of object definition

Camera Specification

- Six degrees of freedom
  - -Position of center of lens
  - -Orientation
- •Lens
- •Film size
- •Orientation of film plane



Lights and Materials

- Types of lights
  - -Point sources vs distributed sources
  - -Spot lights
  - -Near and far sources
  - -Color properties
- Material properties
  - -Absorption: color properties
  - -Scattering
    - Diffuse
    - Specular

Interactive vs. „off-line” graphics

- Interactive graphic libraries
  - OpenGL; Java3D/JOGL; DirectX
  - The rendering may be done in real-time with respect to external stimuli (Games, Simulators, Visualisation)
- Off-line graphic tools
  - 3D Studio Max; Lightwave; Maya
  - The result of rendering is usually a film
  - Off-line graphic are often used for the model production
  - The models are often used in interactive programmes