Objectives

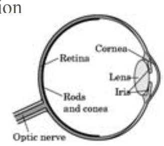
- Fundamental imaging notions
- Physical basis for image formation
  - Light, Color, Perception
- Synthetic camera model
- Other models
- Basic design of a graphics system
- Introduce pipeline architecture
- Examine software components for an interactive graphics system

Luminance and Color Images

- Luminance
  - Monochromatic
  - Values are gray levels
  - Analogous to working with black and white film or television
- Color
  - Has perceptional attributes of hue, saturation, and lightness
  - Do we have to match every frequency in visible spectrum? No!

Three-Color Theory

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

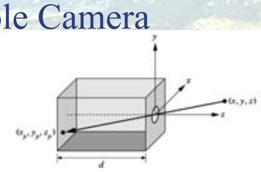


- Need only match these three values
  - Need only three *primary* colors

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

Pinhole Camera

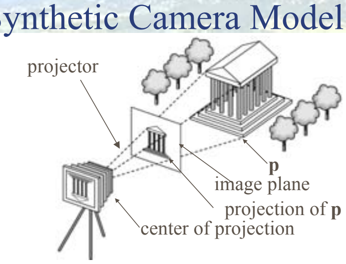


Use trigonometry to find projection of a point

$x_p = -x/z/d \qquad y_p = -y/z/d \qquad z_p = d$

These are equations of simple perspective

Synthetic Camera Model

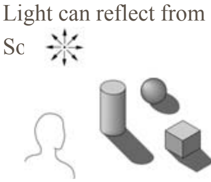


Advantages

- 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

Global vs Local Lighting

- Cannot compute color or shade of each object independently
  - Some objects are blocked from light
  - Light can reflect from object to object
  - Sc translucent



Why not ray tracing?

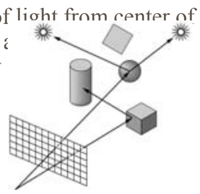
- Ray tracing seems more physically based so why don't we use it to design a graphics system?
- Possible and is actually simple for simple objects such as polygons and quadrics with simple point sources
- In principle, can produce global lighting effects such as shadows and multiple reflections but is slow and not well-suited for interactive applications

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

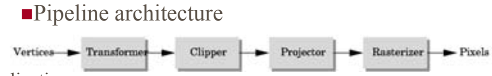
- **Ray tracing**: follow rays of light from center of projection until they either a objects or go off to infinity
  - Can handle global effects
    - Multiple reflections
    - Translucent objects
  - Slow
  - Need whole data base
- **Radiosity**: Energy based approach
  - Very slow



Practical Approach

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

application program

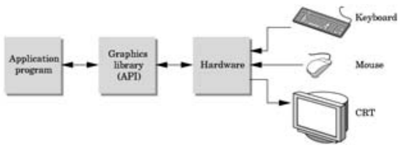


display

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

The Programmer's Interface

- Programmer sees the graphics system through an 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 (1D object)
  - Line segments (2D objects)
  - Polygons (3D objects)
  - Some curves and surfaces
    - Quadrics
    - Parametric polynomial
- All are defined through locations in space or *vertices*

Example

type of object

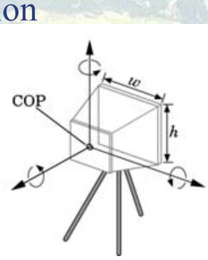
location of vertex

```
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 freedo
  - Position of center of
  - Orientation
- Lens
- Film size
- Orientation of film pl

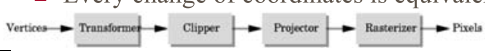


Lights and Materials

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

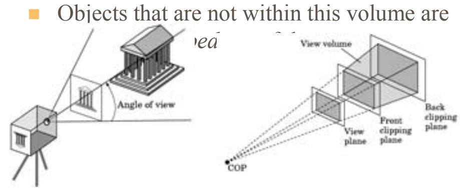
Following the Pipeline: Transformations

- Much of the work in the pipeline is in converting object representations from one coordinate system to another
  - World coordinates
  - Camera coordinates
  - Screen coordinates
- Every change of coordinates is equivalent



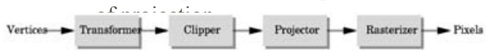
Clipping

- Just as a real camera cannot “see” the whole world, the virtual camera can only see part of the world space
  - Objects that are not within this volume are



Projection

- Must carry out 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



Rasterization

- If an object is visible in the image, the appropriate pixels in the frame buffer must be assigned colors
  - Vertices assembled into objects
  - Effects of lights and materials must be determined
  - Polygons filled with interior colors/shades
  - Must have also determine which objects are in front (hidden surface removal)