IMAGES AND IMAGING SYSTEMS

Computer Graphics

OUTLINE

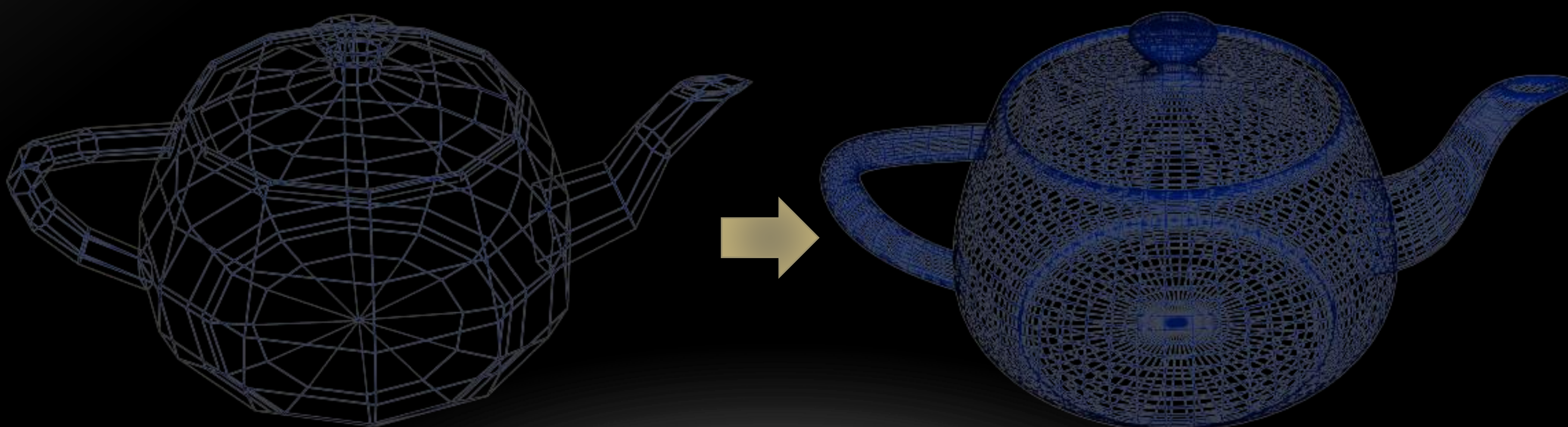
- Fundamental imaging notions
- Physical basis for image formation
  - Light
  - Color
  - Perception
- Synthetic camera model

CLASSICAL GRAPHICS PIPELINE

- Mesh 3D objects with polygons (usually triangles)
- Apply material properties to each object
- Texture map polygons as needed
- Light scene
- Place camera
- Enjoy the view

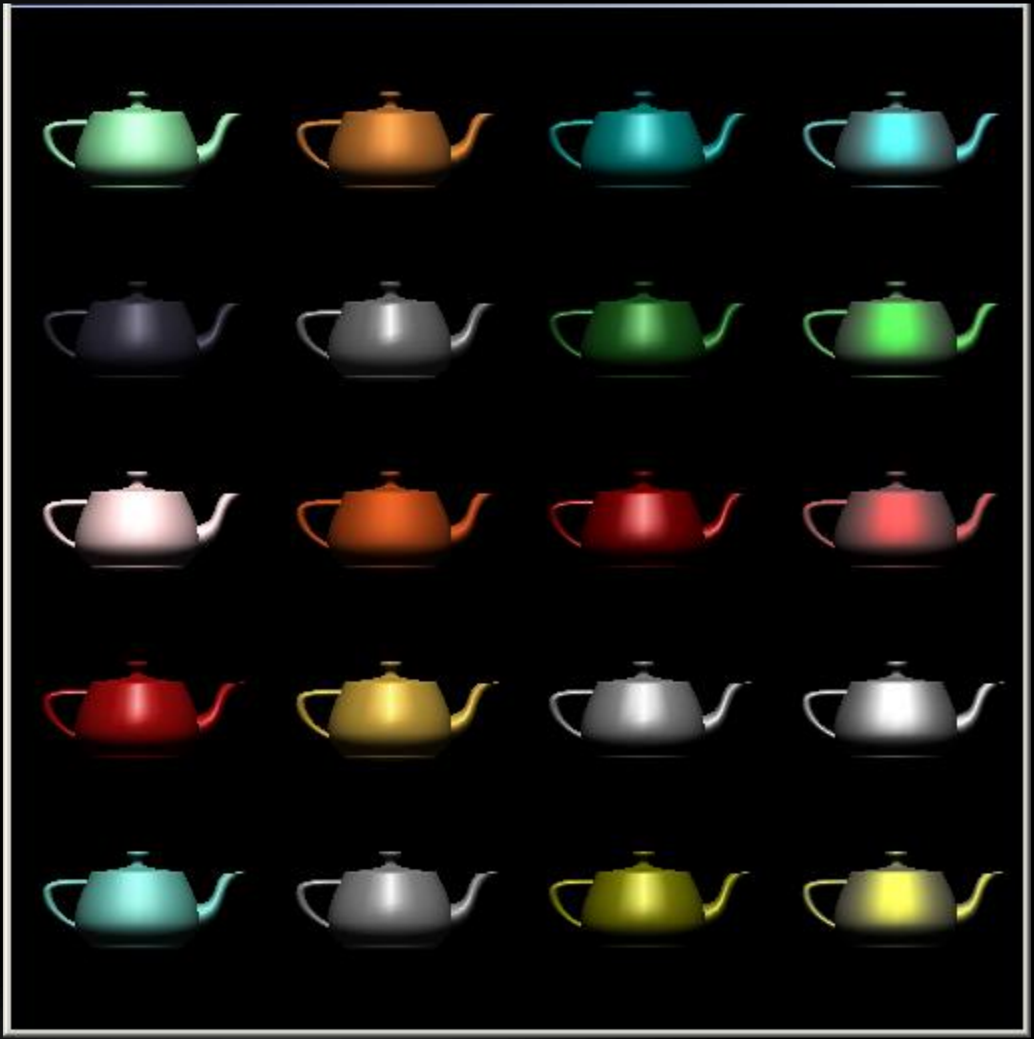
OBJECT MESH

- For better curvature, increase the number of polygons
  - Better shape approximation, more expensive to render
  - Ultimately, still faceted when rendered (higher poly count = less faceted)
  - Adaptive meshing is an improvement - more polygons in areas of high curvature



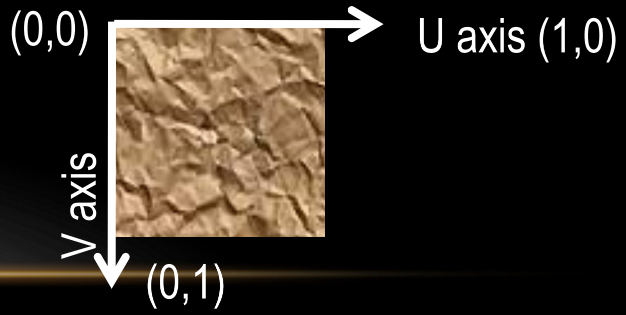
“Utah Teapot” by Martin Newell

MATERIALS



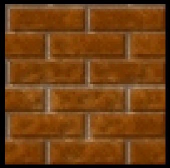
TEXTURE MAPPING

- **Goal:** adding more detail to geometry of scene without adding complexity
- **Solution:** texture mapping
  - Used extensively in video games, e.g., for backgrounds, billboards
  - Also used for many other techniques such as Level of Detail management
  - Cover the mesh's surface in (stretchable) "contact paper" with pattern or image on it
  - In general, difficult to specify mapping from contact paper to every point on an arbitrary 3d surface
  - Mapping to planar polygons is easy: specify mapping for each vertex and interpolate to find mapping of interior points
- Specifying "texture point" mapped to particular vertex
  - Requires coordinate system for referring to positions within texture pixel (texel) map
  - Convention:
    - Points on texel map described in abstract floating-point "texture-coordinate system"
    - Axes labeled u and v, range 0 to 1.
    - Origin located at the upper-left corner of the pixmap

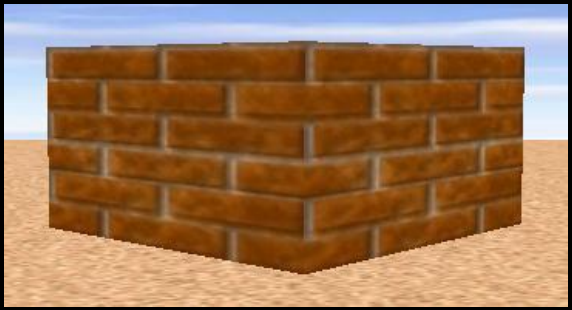


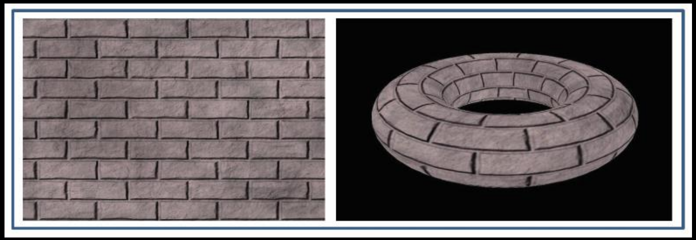
TEXTURE MAPPING (TILING)

- Create a brick wall by applying brick texture to plane

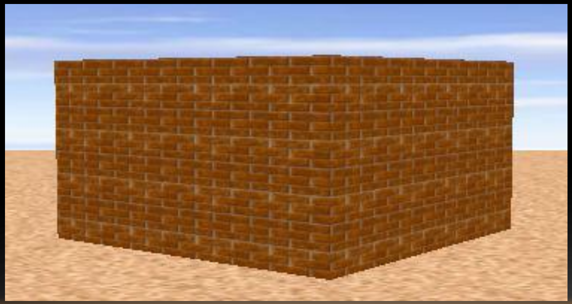


- Produces realistic-looking image, but very few bricks in wall





- Tiling increases number of apparent bricks



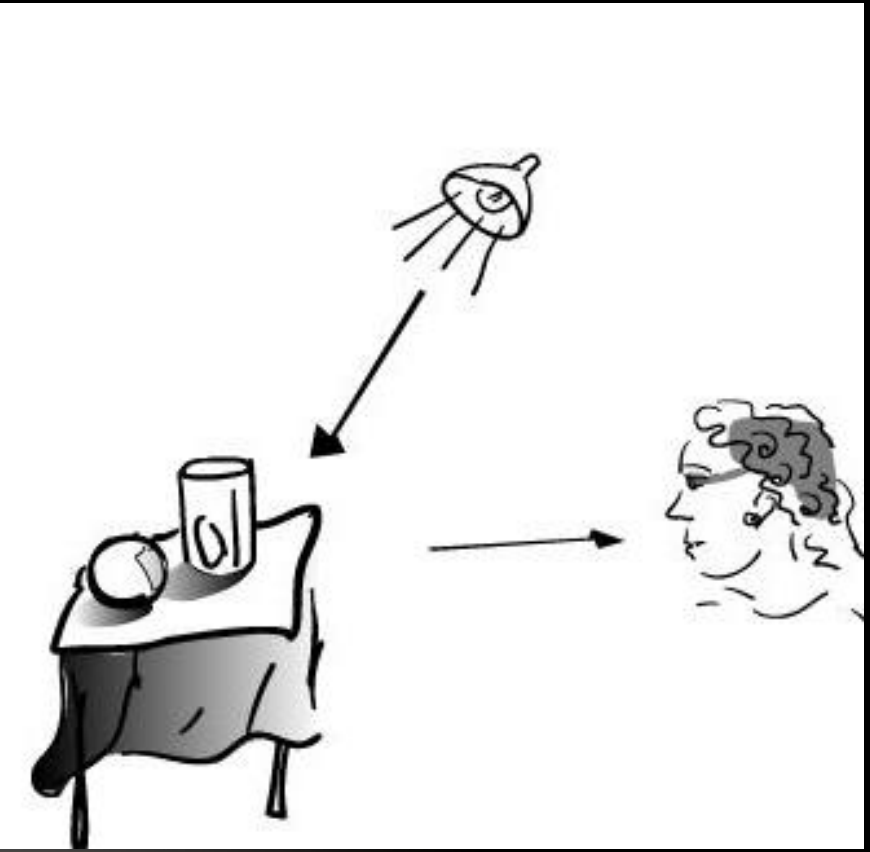
TEXTURE MAPPING (STRETCHING)

- Create a sky backdrop by applying a sky image to a plane

- Would look unnatural if tiled
- Stretch to cover whole plane

COMPLEXITIES OF LIGHT REFLECTION

- Intensities and directions of *all light* that strikes a point on object's surface, directly from lights and also from many inter-object bounces, i.e., global illumination



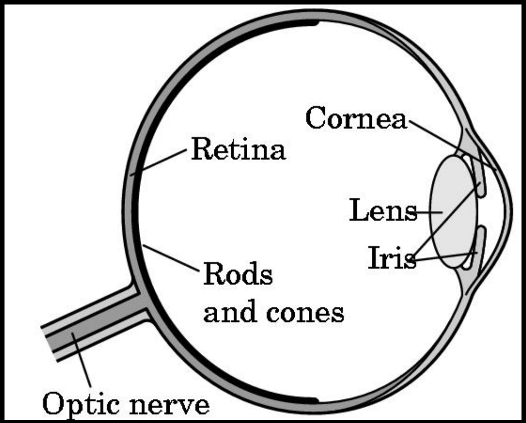
- How an object's surface appears to us as it reflects, absorbs and diffracts light (material properties)
- Location of eye relative to light source
- Distribution of intensity per wavelength of incident light
- Human visual system (HVS) and its differential response to light stimuli
- Lights may have geometry themselves
- **Lighting/illumination models address some of these complexities**

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



LUMINANCE AND COLOR IMAGES

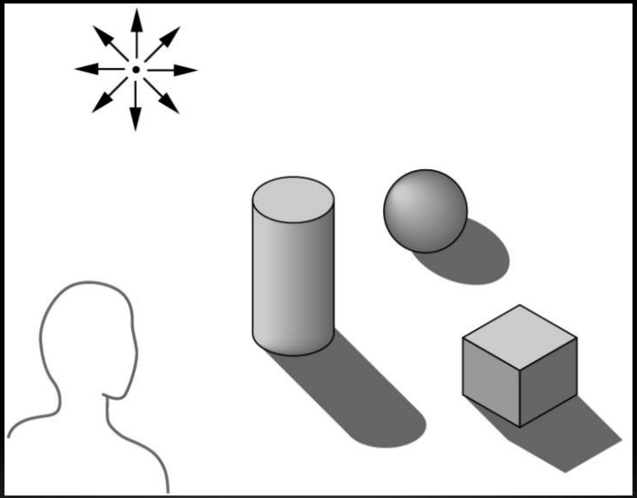
- Luminance Image
  - Monochromatic
  - Values are gray levels
  - Analogous to working with black and white film or television
- Color Image
  - Has perceptional attributes of hue, saturation, and lightness
  - Do we have to match every frequency in visible spectrum? No!

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

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
  - Some objects might be translucent



WHY NOT RAY TRACING?

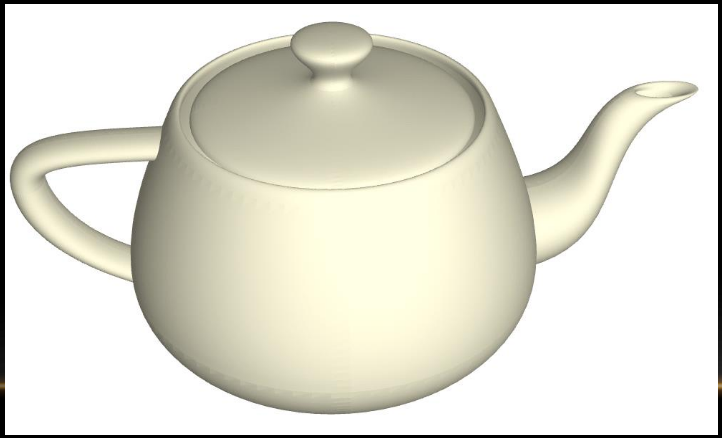
- Ray tracing seems more physically based so why don't we use it to design a graphics system?
- Possible and is actually simple for simple objects such as polygons and quadrics with simple point sources
- In principle, can produce global lighting effects such as shadows and multiple reflections but ray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

LIGHTING

- Get this:

faceted shading

- Want this:



smooth shading

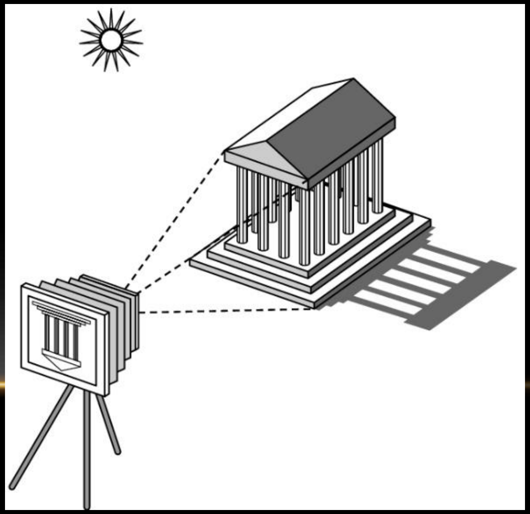
IMAGE FORMATION

- 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)

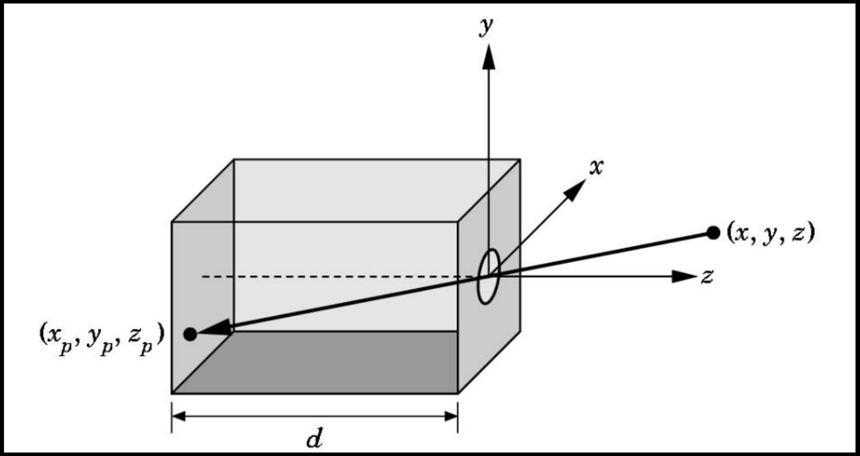
- Attributes that govern how light interacts with the materials in the scene
- Note the independence of the objects, the viewer, and the light source(s)



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

PINHOLE CAMERA

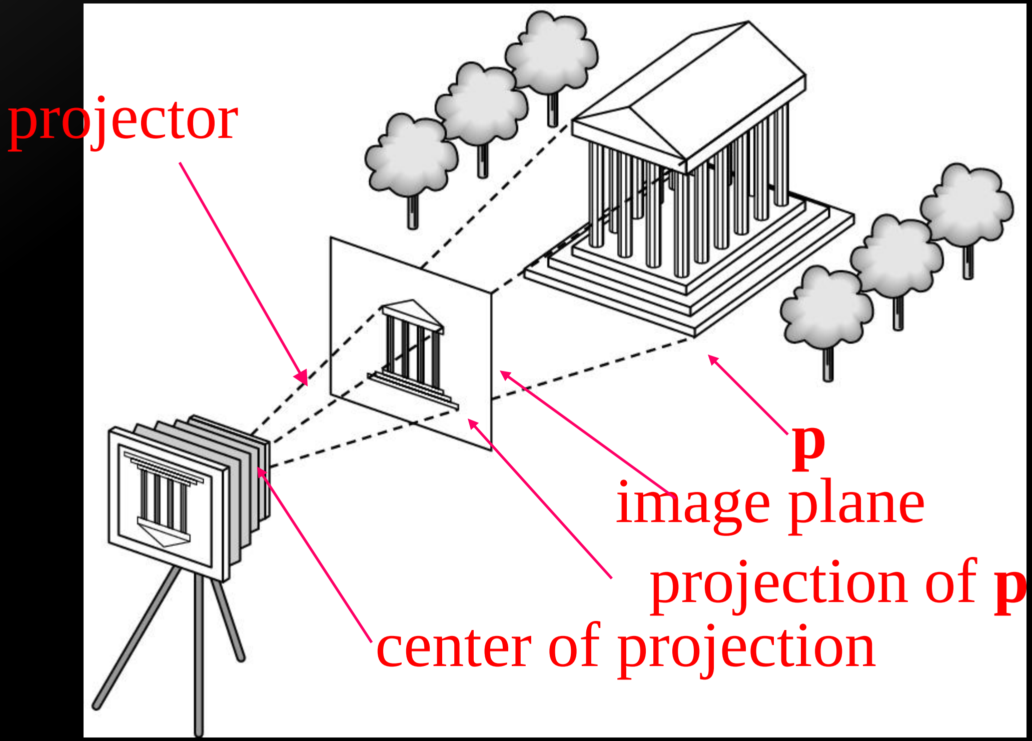


Use trigonometry to find projection of point at (x,y,z)

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

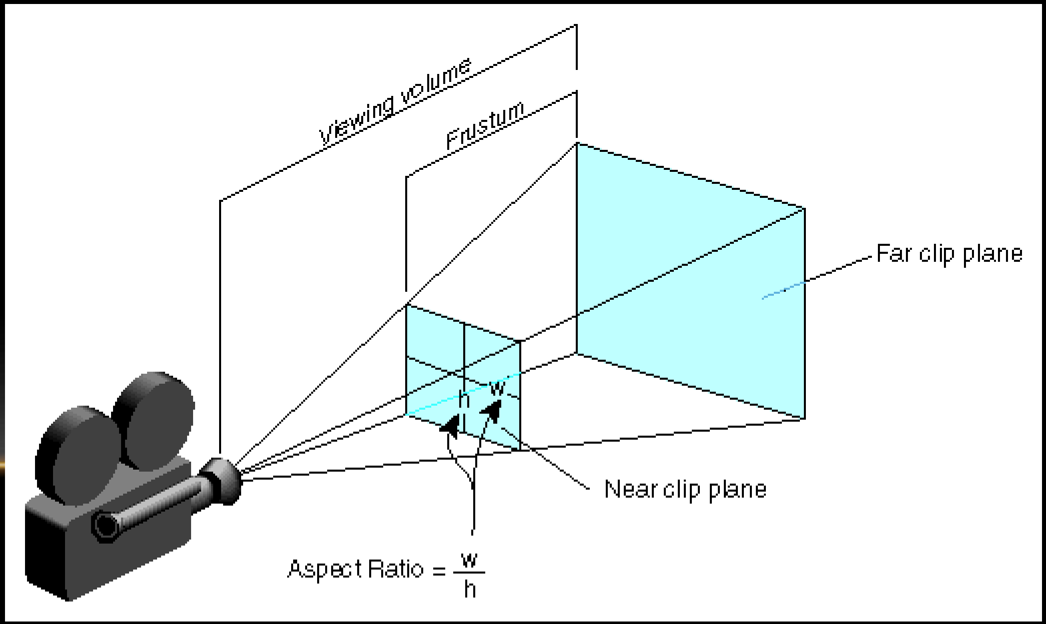
These are equations of simple perspective

SYNTHETIC CAMERA MODEL



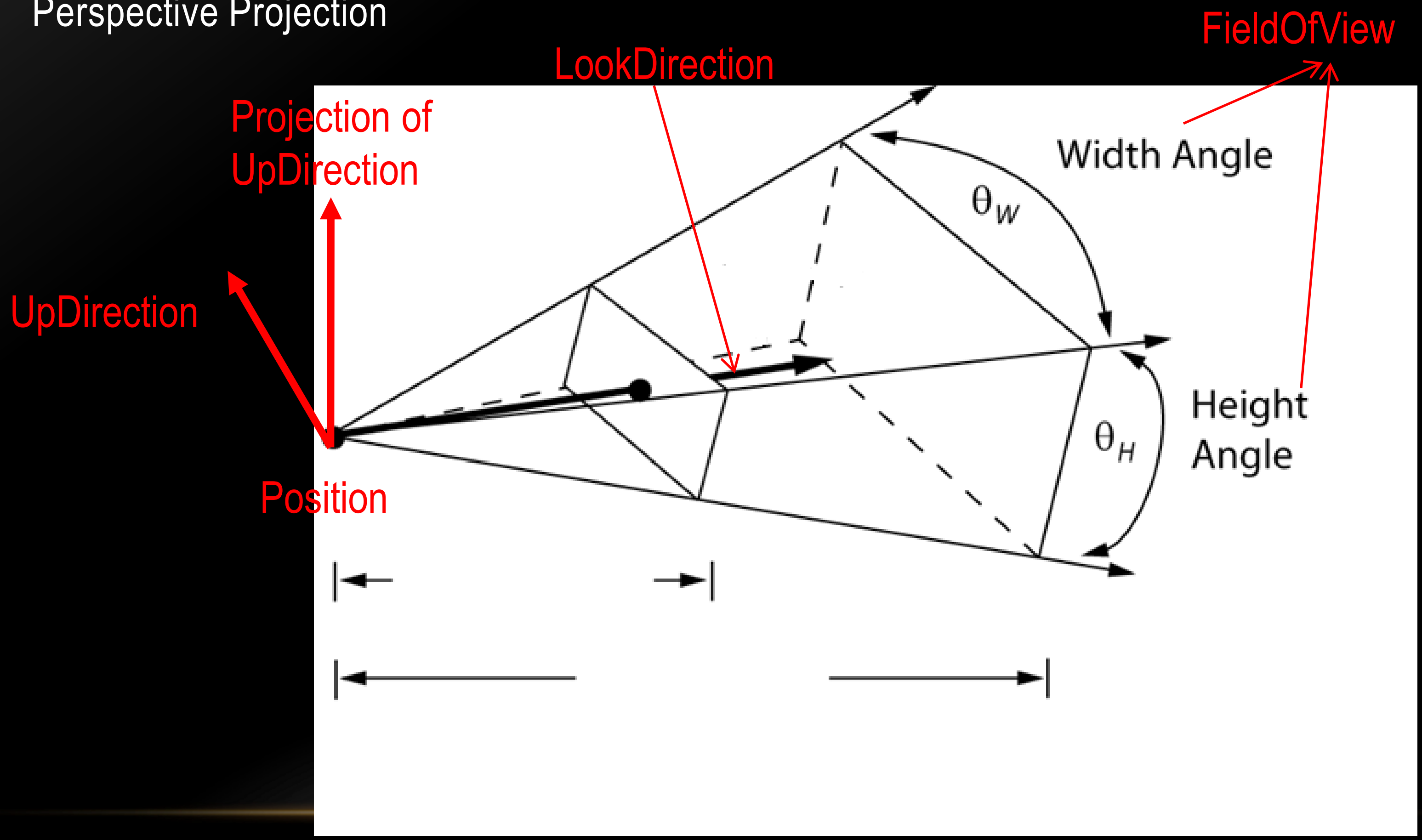
CAMERA

- Camera Properties:
  - **PerspectiveCamera** or **OrthographicCamera**
  - **Position**: placement of camera
  - **LookDirection**: direction camera is aimed (vector determining lens axis)
  - **UpDirection**: rotates camera about look vector, specifying which way is "up"-- must not be collinear to the look vector
  - **FarPlaneDistance**: objects behind do not appear
  - **NearPlaneDistance**: objects in front do not appear
  - **FieldOfView**: (Width angle, Height Angle)



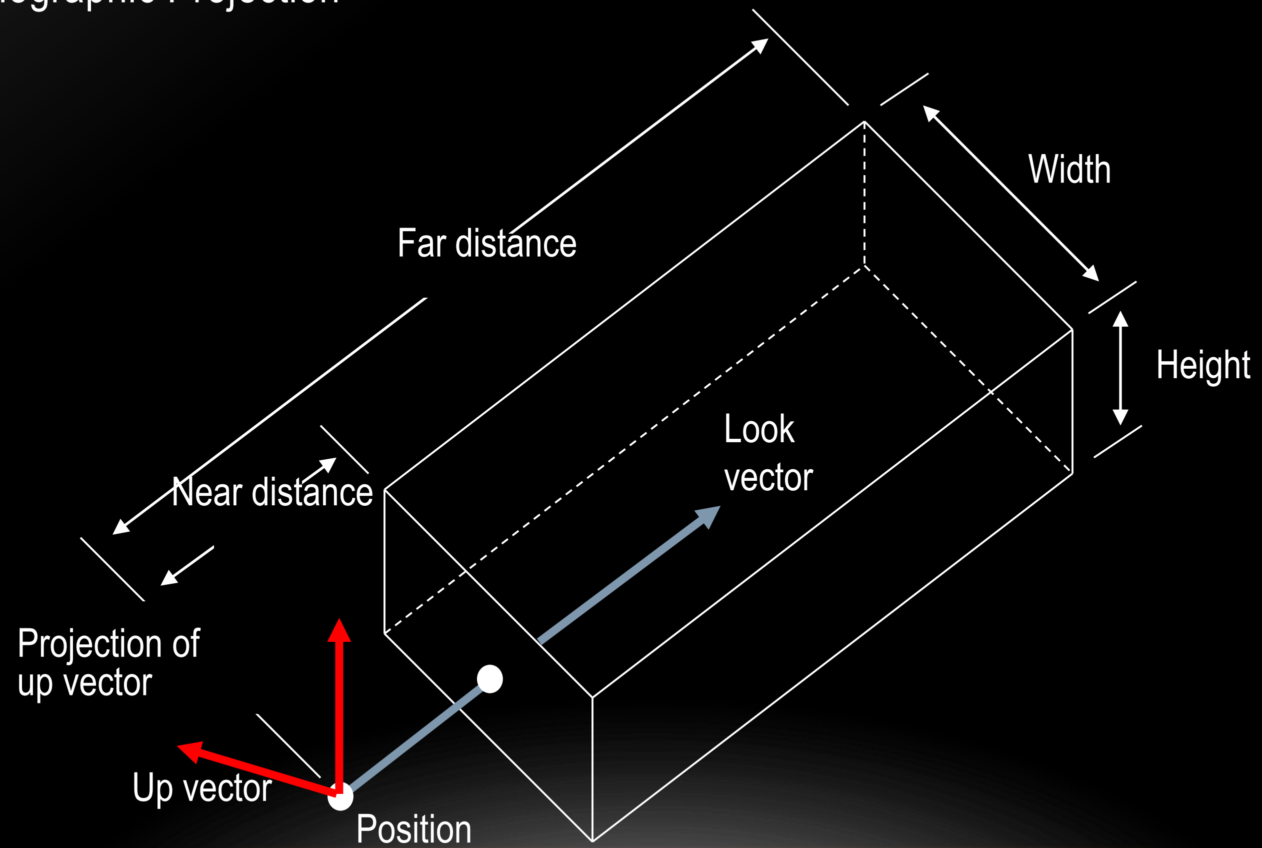
CAMERA

- Perspective Projection



CAMERA

- Orthographic Projection



SUMMARY

- Fundamental imaging notions
- Physical basis for image formation
  - Light
  - Color
  - Perception
- Synthetic camera model