Game Programming

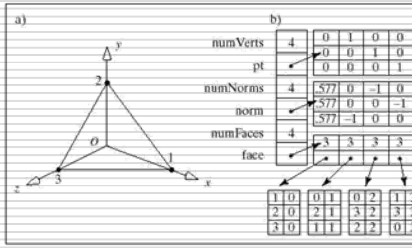
Bing-Yu Chen
National Taiwan University

What is Computer Graphics ?

- Definition
  - the pictorial *synthesis* of real or imaginary objects from their computer-based models

		OUTPUT	
		descriptions	images
INPUT	descriptions		Computer Graphics
	images	Computer Vision Pattern Recognition	Image Processing

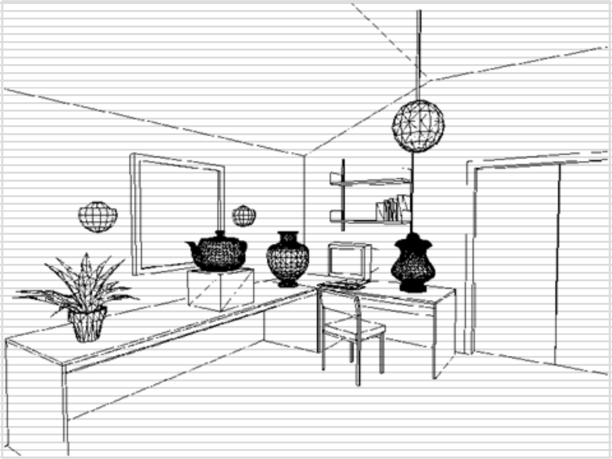
What is Computer Graphics ?



modeling

rendering

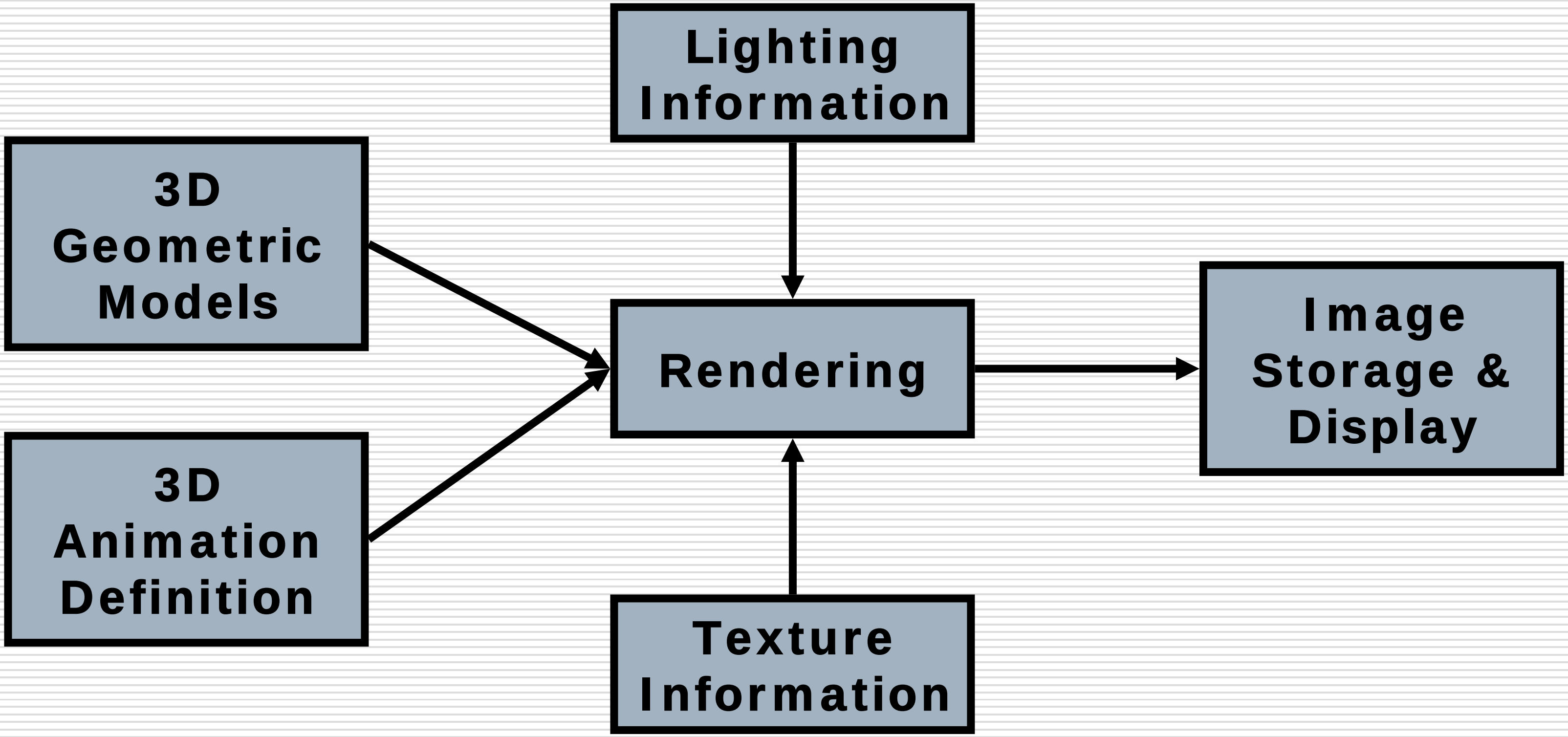
animation



Applications

- Movies
- Interactive entertainment
- Industrial design
- Architecture
- Culture heritage

The Graphics Process



3D Geometric Models

3D Animation Definition

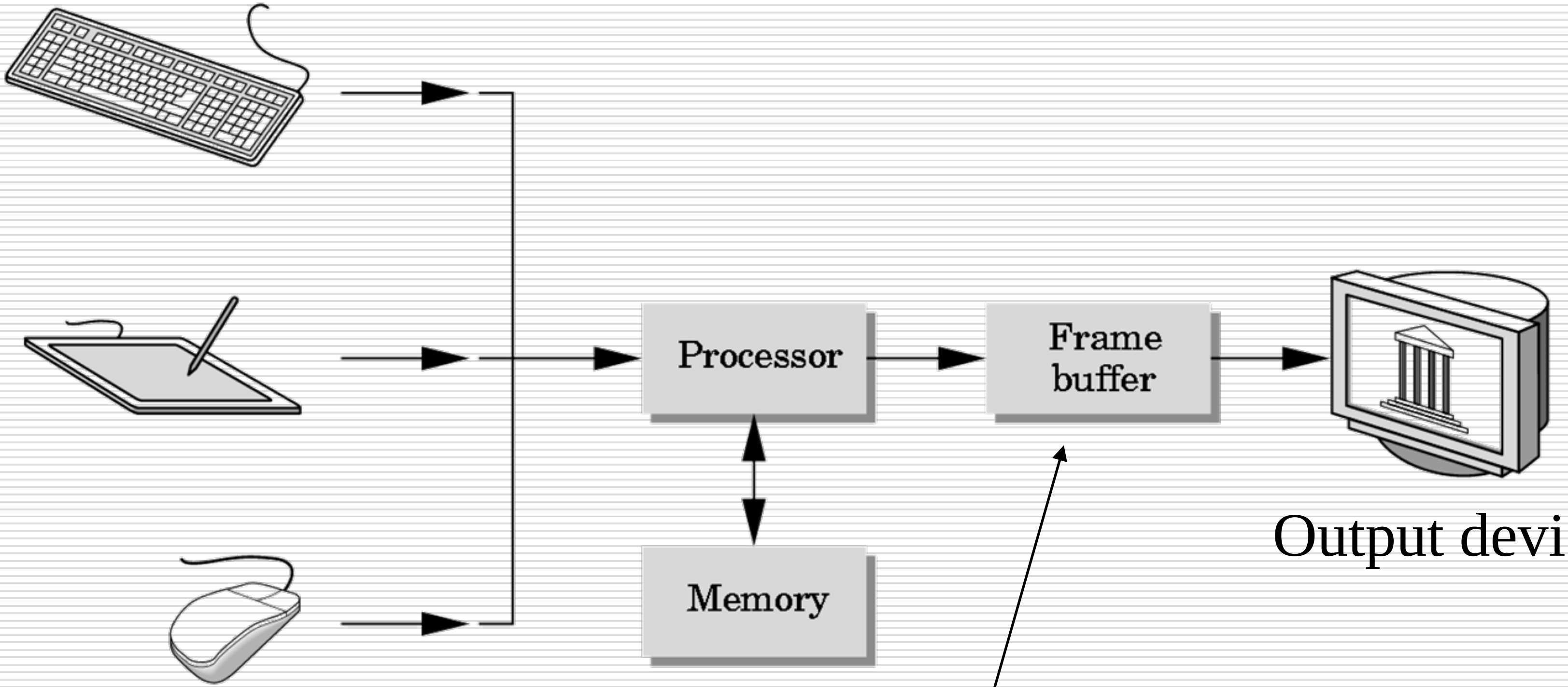
Lighting Information

Rendering

Texture Information

Image Storage & Display

Basic Graphics System

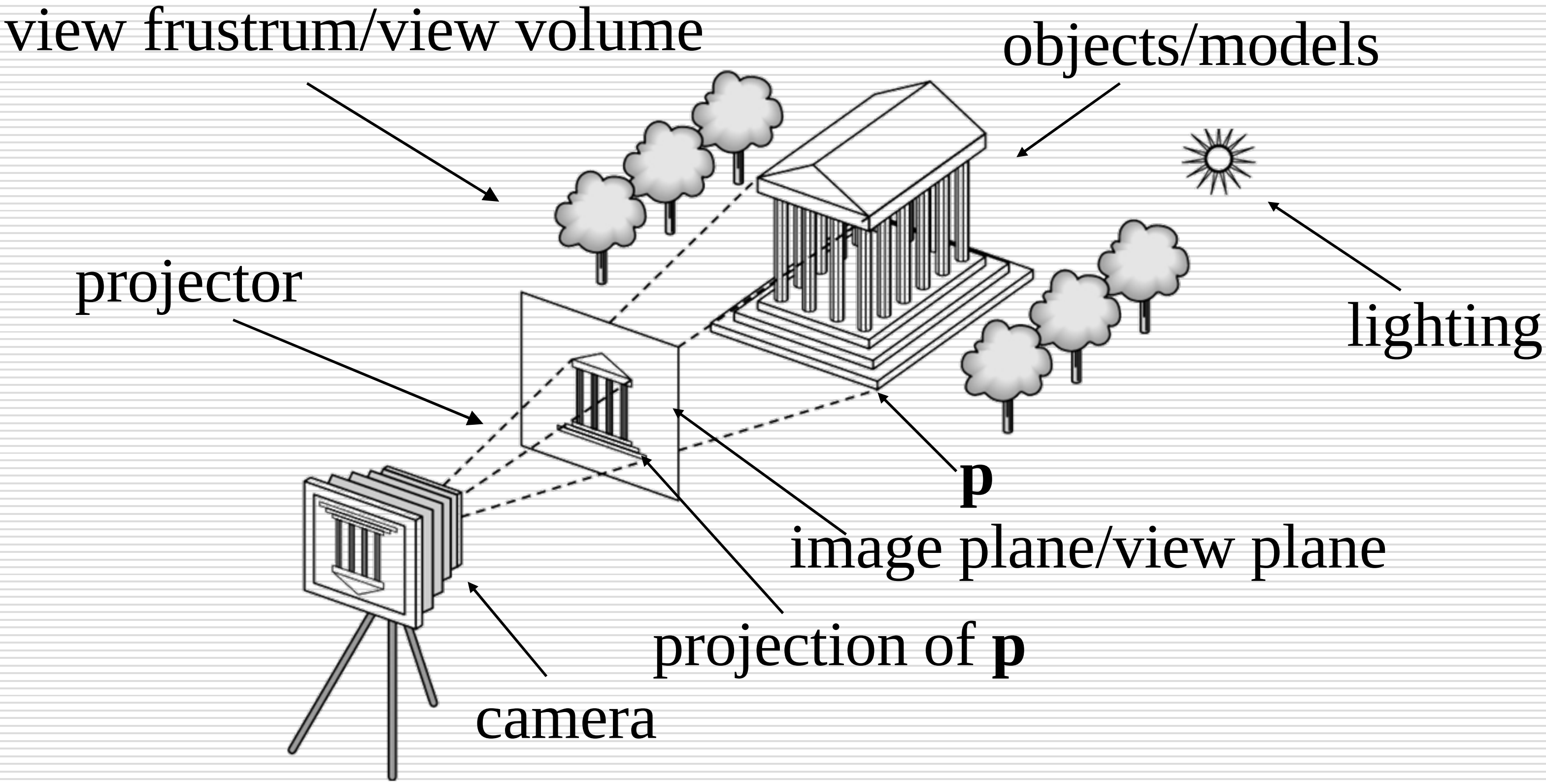


Input devices

Output device

Image formed in FB

Synthetic Camera Model



view frustrum/view volume

objects/models

projector

lighting

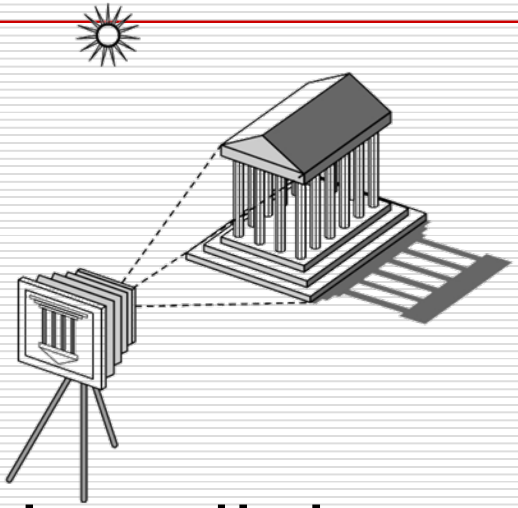
p

image plane/view plane

projection of **p**

camera

Elements of Image Formation



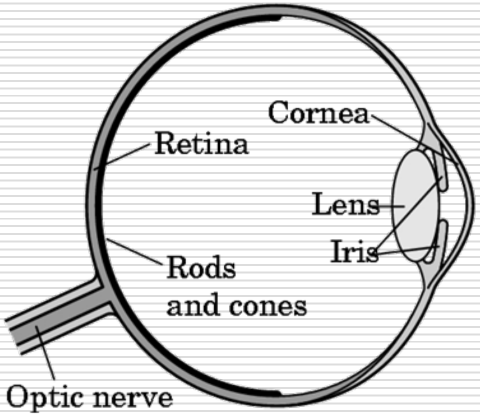
- Objects
- Viewer
- Light source(s)
- Attributes that govern how light interacts with the materials in the scene
- Note the independence of the objects, viewer, and light source(s)

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

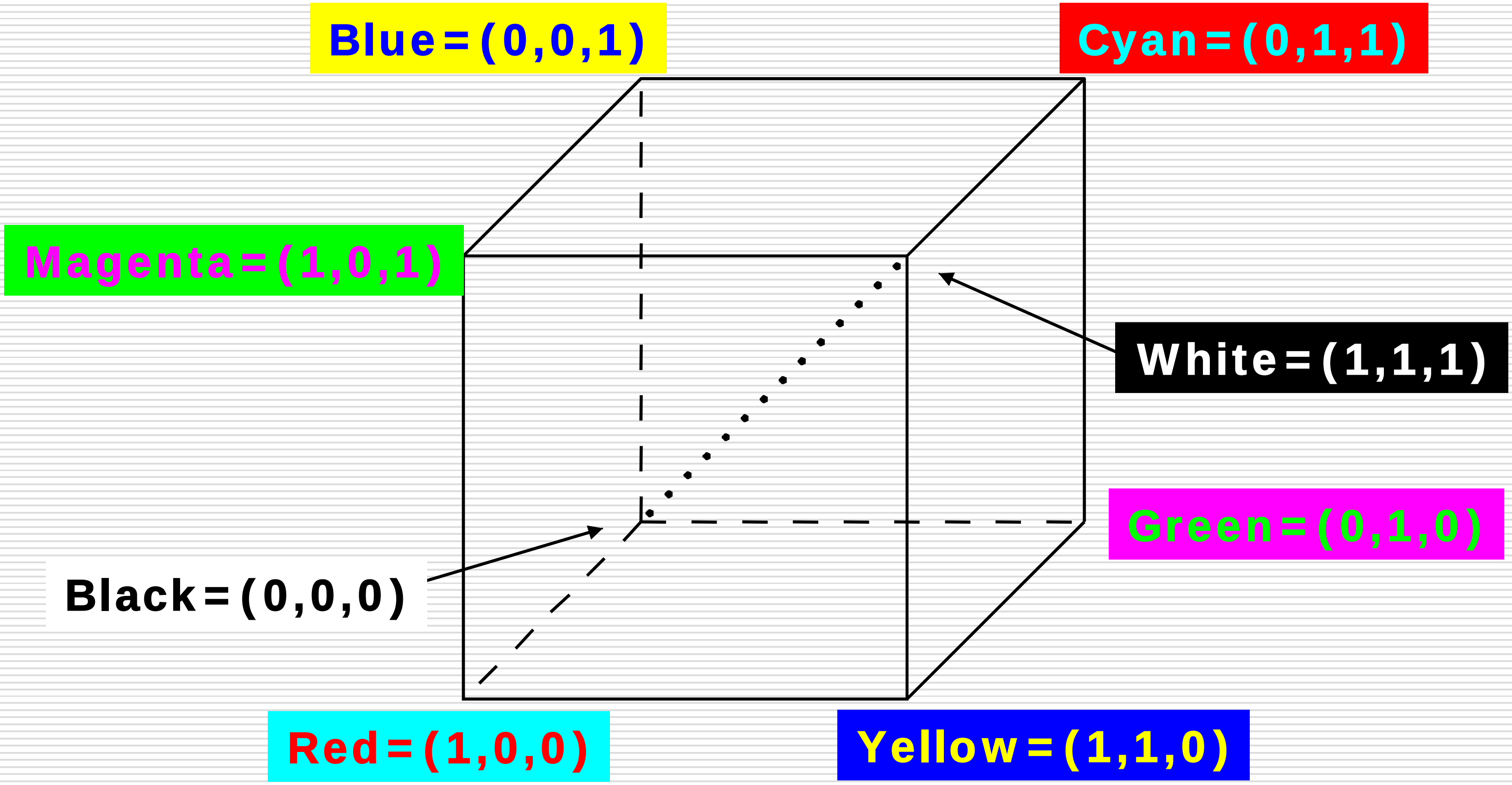


- Need only match these three values
  - Need only three *primary* colors

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

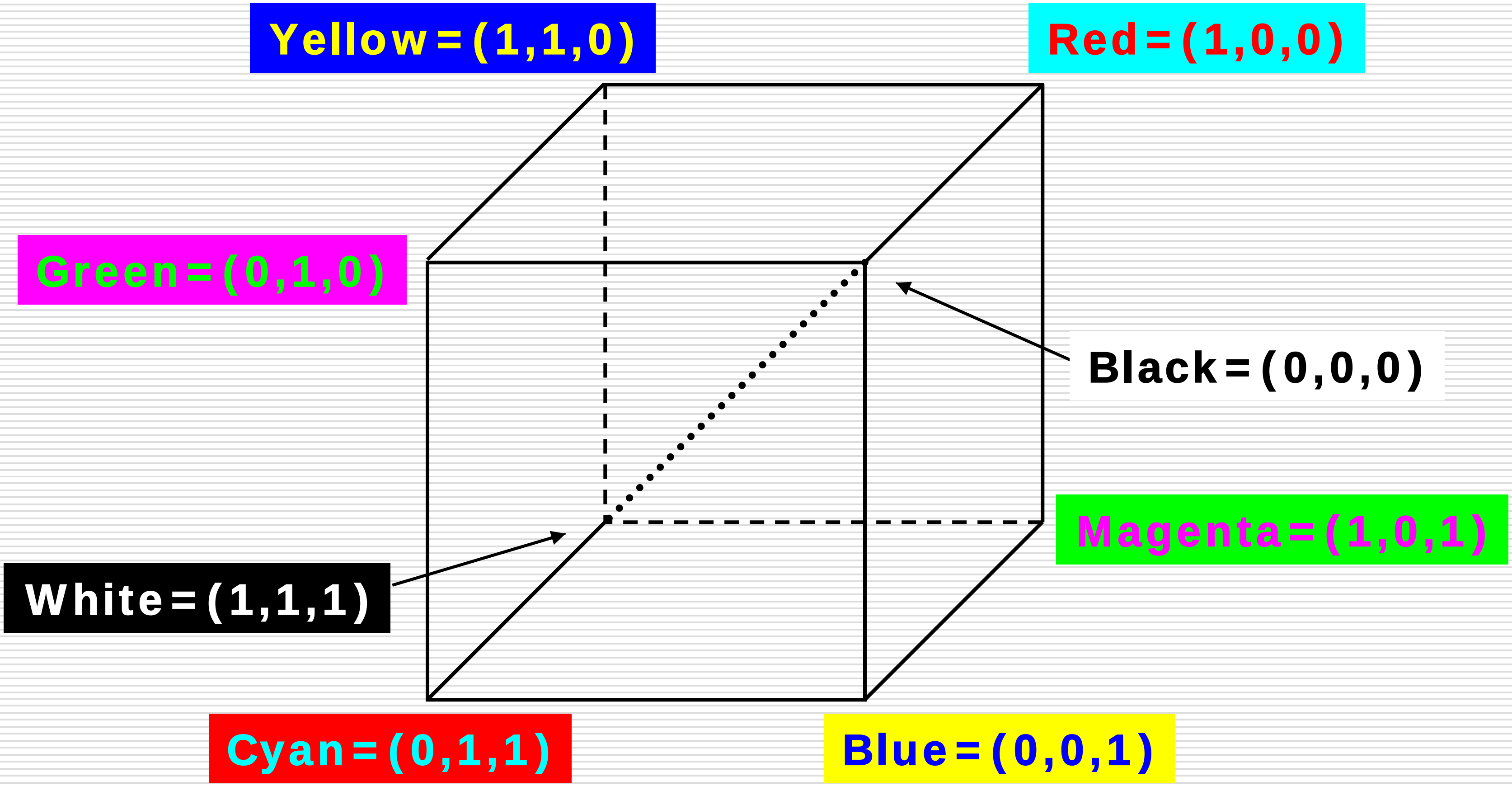
The RGB Color Model – for CRT



Color Depth

- Can choose number of bits for each of *r*, *g* and *b*
  - More bits per component means more colors can be distinguished, but image files will be larger
  - 8 bits (1 byte) per component: *24-bit color*, millions of colors
- If *r* = *g* = *b*, color is a shade of gray, so grayscale can be represented by a single value
  - 8 bits permits 256 grays

The CMY Color Model – for hardcopy



Yellow=(1,1,0)

Red=(1,0,0)

Green=(0,1,0)

Black=(0,0,0)

Magenta=(1,0,1)

White=(1,1,1)

Cyan=(0,1,1)

Blue=(0,0,1)

Undercolor Removal: CMYK System

- Real inks do not correspond to ideal subtractive primaries
- Combining three inks for black is undesirable
- Printers use *four process colors*, cyan, magenta, yellow and black
- CMYK gamut is not the same as RGB
  - Implications for using images prepared for print (CMYK) on the Web (RGB)

The CMYK Color Model – for hardcopy

- $C = G+B = W-R$
- $M = R+B = W-G$
- $Y = R+G = W-B$

⇩

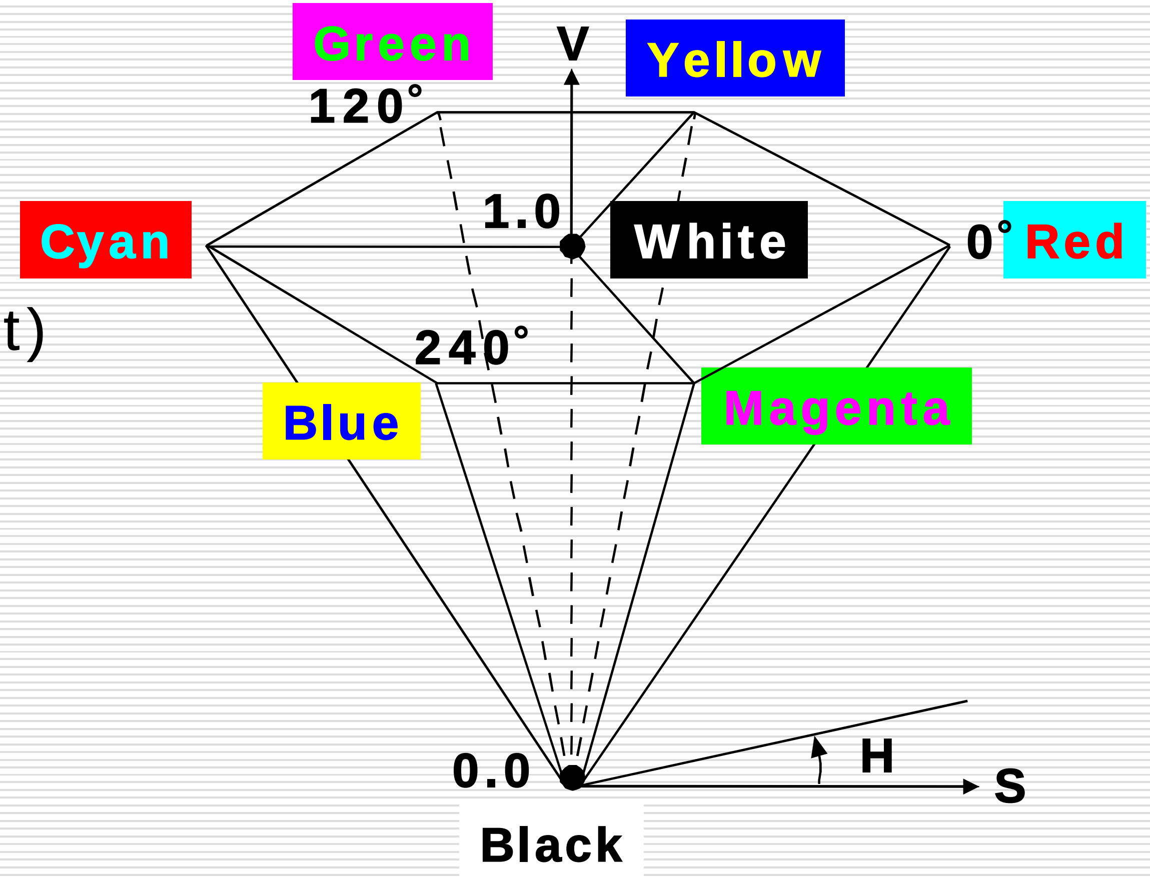
- $K = \min(C,M,Y)$
- $C \leftarrow C-K$
- $M \leftarrow M-K$
- $Y \leftarrow Y-K$

The HSV Color Model – for user-oriented

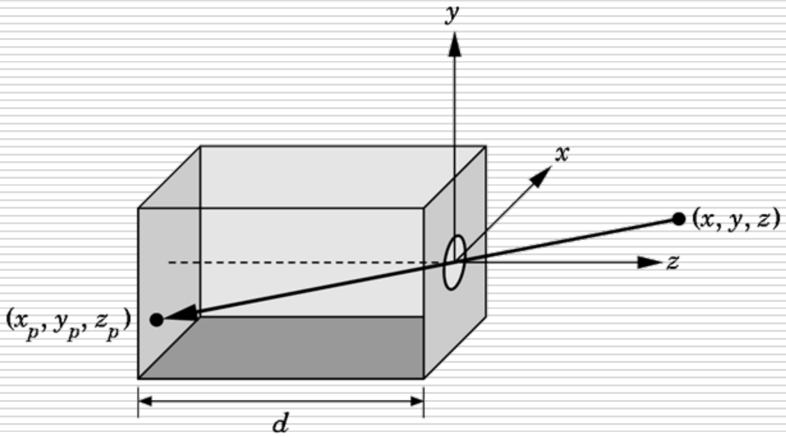
- □ Alternative way of specifying color
- □ *Hue* (roughly, dominant wavelength)
- □ *Saturation* (purity)
- □ *Value* (brightness)
- □ Model HSV as a cylinder: *H* angle, *S* distance from axis, *V* distance along axis
- □ Basis of popular style of *color picker*

The HSV Color Model – for user-oriented

- H : hue
- S : saturation
- V : value
  - (or B for blight)



Pinhole Camera



Use trigonometry to find projection of a point

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

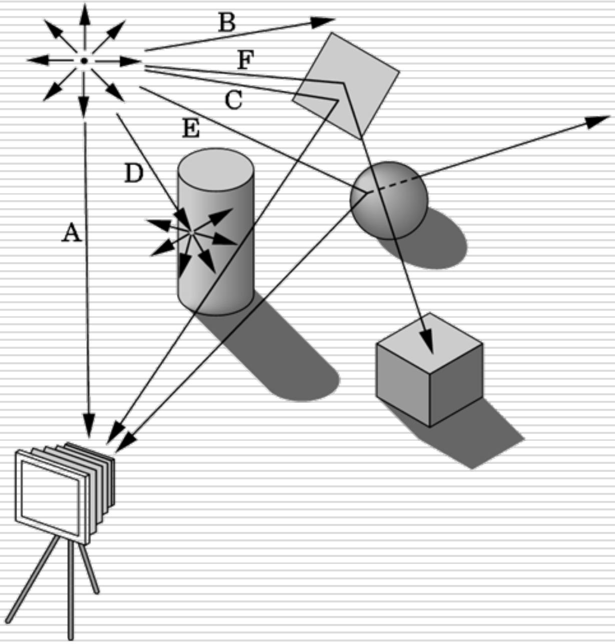
These are equations of simple perspective

Basics of Rendering

- Pipeline Based Rendering
  - Objects in the scene are rendered in a sequence of steps that form the Rendering Pipeline.
- Ray-Tracing
  - A series of rays are projected thru the view plane and the view plane is colored based on the object that the ray strikes

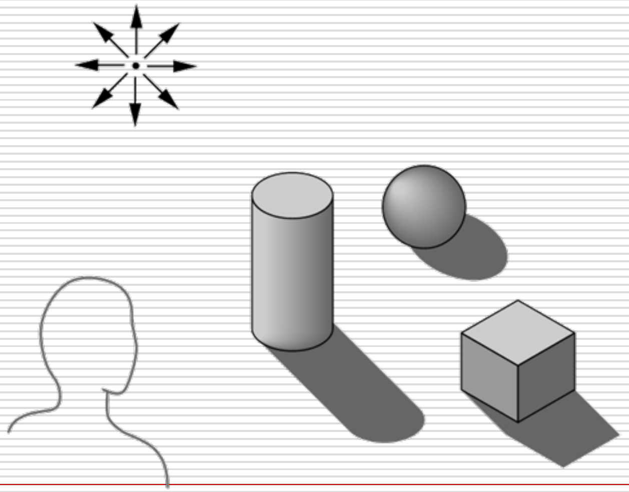
Ray Tracing and Geometric Optics

One way to form an image is to follow rays of light from a point source determine which rays enter the lens of the camera. However, each ray of light may have multiple interactions with objects before being absorbed or going to infinity.



Global vs. Local Lighting

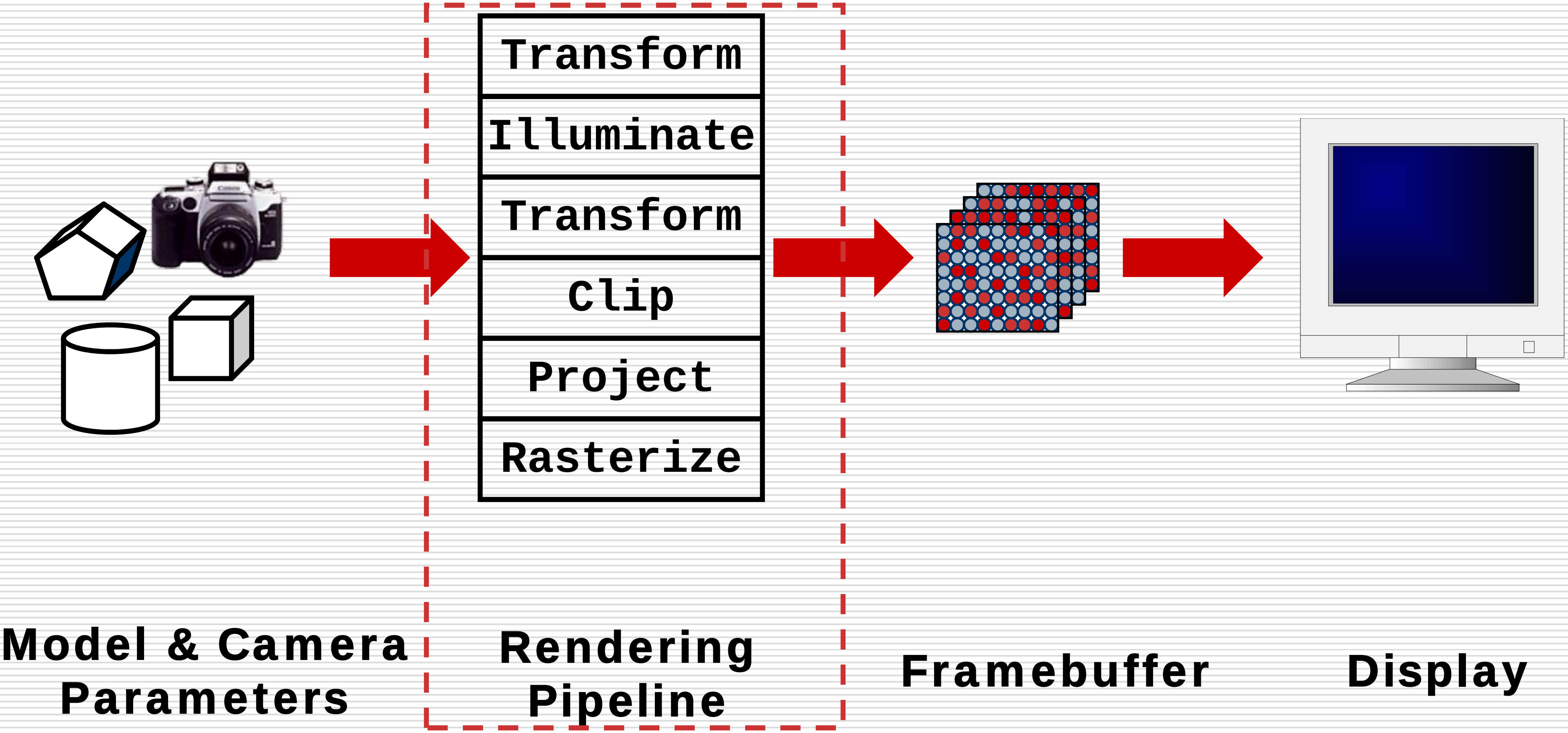
- Cannot compute color or shade of each object independently
  - Some objects are blocked from light
  - Light can reflect from object to object
  - Some objects might be translucent



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 is slow and not well-suited for interactive applications

Pipeline Rendering



Transform

Illuminate

Transform

Clip

Project

Rasterize

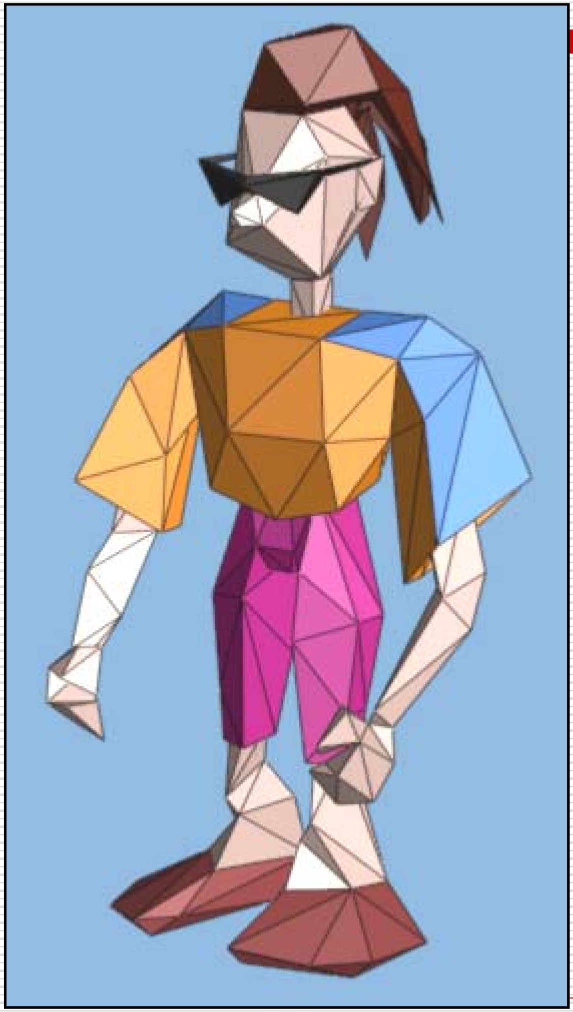
Model & Camera Parameters

Rendering Pipeline

Framebuffer

Display

Definitions of Triangle Meshes



[Hoppe 99']

$\{f_1\} : \{ v_1 , v_2 , v_3 \}$

$\{f_2\} : \{ v_3 , v_2 , v_4 \}$

…

$\{v_1\} : (x,y,z)$

$\{v_2\} : (x,y,z)$

…

$\{f_1\}$: *"skin material"*

$\{f_2\}$: *"brown hair"*

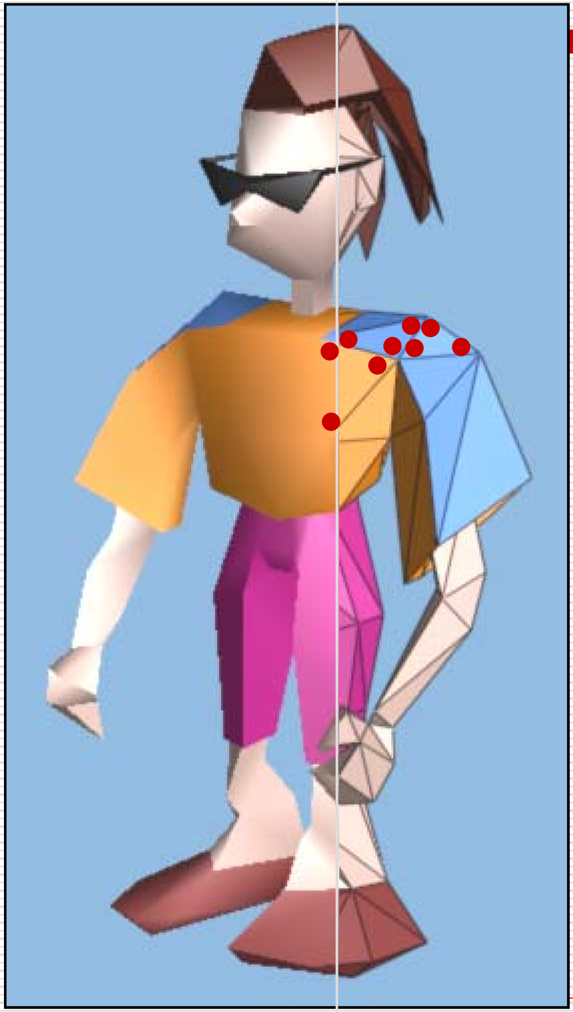
…

connectivity

geometry

face attributes

Definitions of Triangle Meshes



[Hoppe 99']

$\{f_1\} : \{ v_1 , v_2 , v_3 \}$

$\{f_2\} : \{ v_3 , v_2 , v_4 \}$

…

connectivity

$\{v_1\} : (x,y,z)$

$\{v_2\} : (x,y,z)$

…

geometry

$\{f_1\}$: *"skin material"*

$\{f_2\}$: *"brown hair"*

…

face attributes

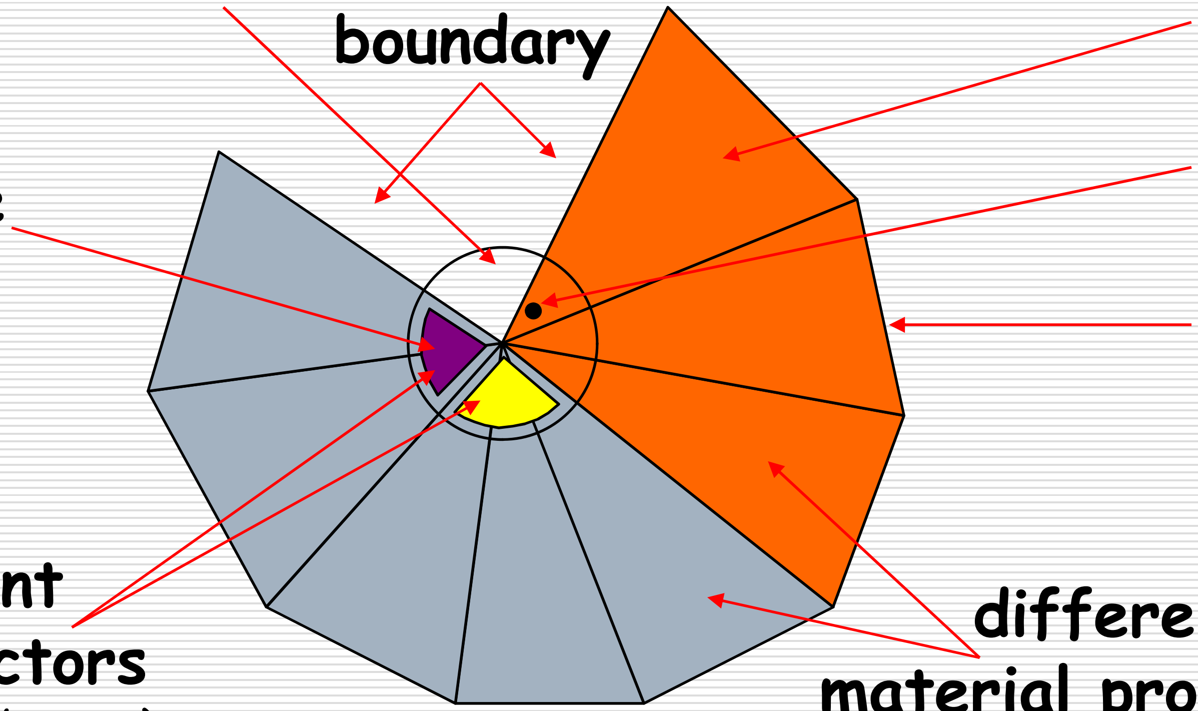
$\{v_2,f_1\} : (n_x,n_y,n_z)\ (u,v)$

$\{v_2,f_2\} : (n_x,n_y,n_z)\ (u,v)$

…

corner attributes

Definitions of Triangle Meshes



vertex

boundary

face

corner

wedge

edge

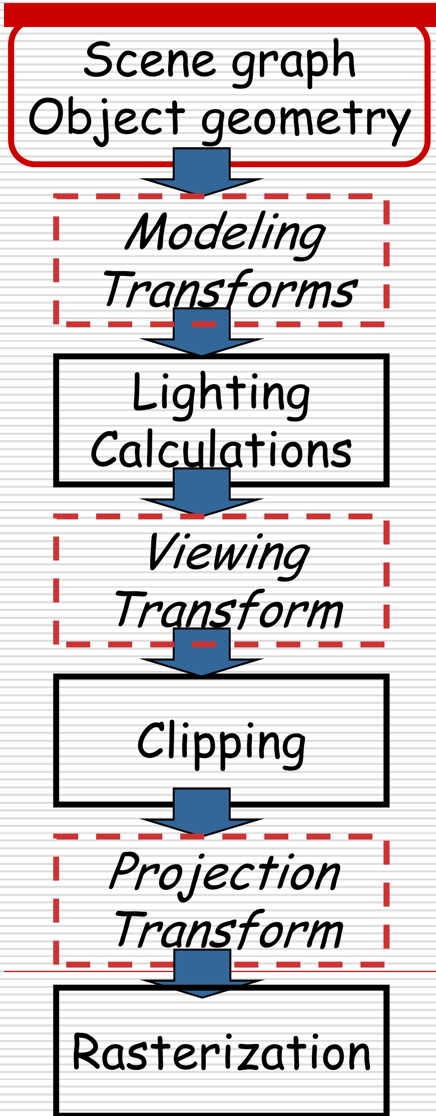
different
normal vectors
(corner attributes)

different
material properties
(face attributes)

Rendering: Transformations

- So far, discussion has been in *screen space*
- But model is stored in *model space* (a.k.a. object space or world space)
- Three sets of geometric transformations:
  - Modeling transforms
  - Viewing transforms
  - Projection transforms

The Rendering Pipeline



Scene graph
Object geometry

Modeling Transforms

Lighting Calculations

Viewing Transform

Clipping

Projection Transform

Rasterization