Texture Mapping

Lecture 10

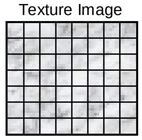
- Increase Object Detail
  - Paint
    - Decal
  - Material
    - Wood Grain
    - Marble
    - Non-Plastic
  - Geometry
    - Surface Normal
  - Surrounding Environment
    - Reflection
    - Transparency (Clouds)
- Increased Computation Time

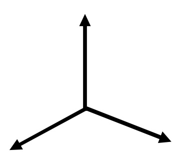
1

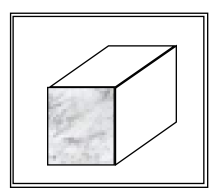
Texture Mapping

Lecture 10

Texture Image







- Texture Space (s, t)
- Object Space (u, v)
- Rendered Image Space (i, j)

- Texels
- Parametric Surface
- Pixels

I. (s,t) —Map to Surface→ (u,v) —Project to Image→ (i,j)

II. (s,t) ←Map to Texture— (u,v) ←Project to Surface— (i,j)

III. Two Pass Method

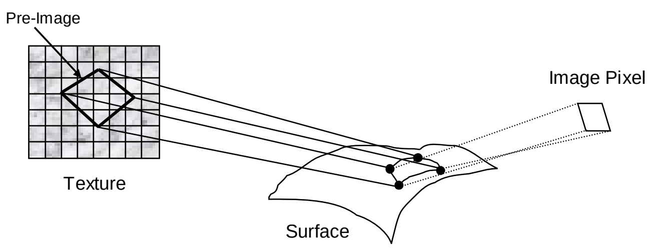
2

EECS 487

Texture Mapping

Lecture 10

II. (s,t) ← Map to Texture ← (u,v) ← Project to Surface ← (i,j)



- Project 4 Pixel Corners Onto Parametric Surface(u,v)
- Transform Parametric Surface Values to Texel Space (s,t)
- Compute Weighted Sum of Texels in Bounding Quadrilateral
- Set Image Pixel Value

3

EECS 487

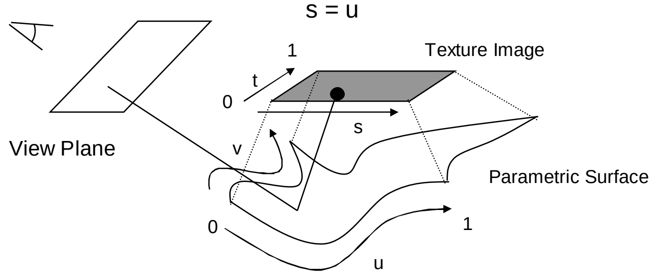
Texture Mapping

Lecture 10

- Shrinkwrapping Image Onto a Parametric Surface

Viewing Pixel (i,,j) → Object Coords (u,v) → Texture Coords (s,t) → Pixel Value (r,g,b)

t = v
s = u

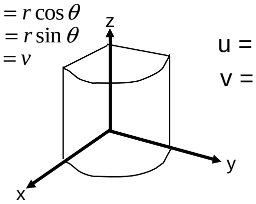


4

Texture Mapping

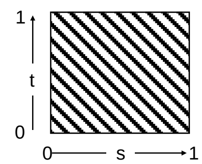
Lecture 10

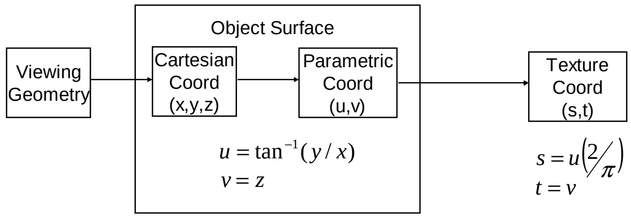
$x = r\cos\theta$
$y = r\sin\theta$
$z = v$



u = θ $0 \le \theta \le \pi/2$

v = z $0 \le z \le 1$





$u = \tan^{-1}(y/x)$
$v = z$

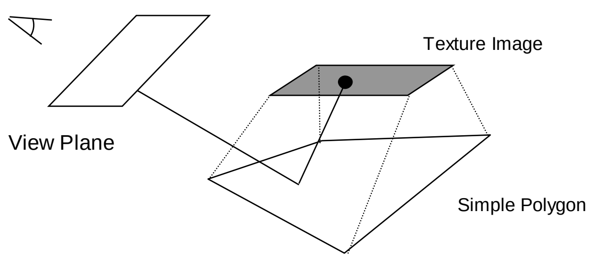
$s = u\left(2/\pi\right)$
$t = v$

Texture Mapping

Lecture 10

- “Vacuum-Form” Image Onto a Polygonal Surface

Viewing Pixel (i,,j) → Object Coords (u,v) → Texture Coords (s,t) → Pixel Value (r,g,b)



- Assign Texture Coordinates to Polygon Verticies

Texture Mapping

Lecture 10

- Polygon Mesh Texture Mapping ?
- Two-Part Mapping
  - Map Texture onto an intermediate surface
    - Generally non-planar
    - Possesses an simple parametric definition
  - Map intermediate surface onto object
    - Correspondence between object point and texture point is a 3-D to 3-D transform
  - Handles "global" mapping of ploygon meshes
  - Texture may be distorted by double mapping

7

Two Pass Texture Mapping

Lecture 10

- Map from 2-D Texture Space to a *simple* 3-D intermediate surface
  - Cylinder
  - Sphere
  - Box
  - Plane

$$T(s,t) \rightarrow T'(x_i, y_i, z_i)$$ "S" mapping

- Determine mapping from 3-D intermediate surface to 3-D object surface

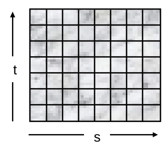
$$T'(x_i, y_i, z_i) \rightarrow O(x, y, z)$$ "O" mapping

8

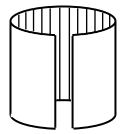
Two Pass Texture Mapping

Lecture 10

- "S" mapping



Texture
(s,t)



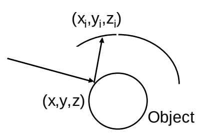
Surface
(θ,h)

$$s = \frac{1}{c}(\theta - \theta_0)$$

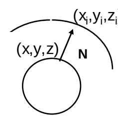
$$t = \frac{1}{d}(h - h_0)$$

c, d are scaling factors
θ_0, h_0 position the texture on the cylinder

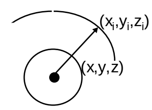
- "O" mapping



Reflected Ray



Object Normal



Object Centroid

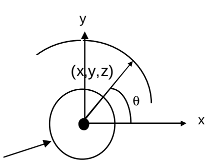
Two Pass Texture Mapping

Lecture 10

1. Map the four pixel corners to the surface of the object (x,y,z)

2. Apply the "O" mapping (θ, h)

$$\theta = \tan^{-1}(y/x)$$
$$h = z$$



Polygon Mesh Object

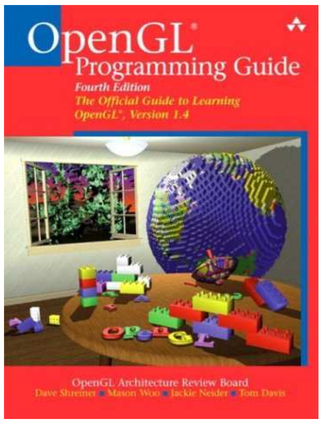
3. Apply "S" mapping to find texture point (s,t)

$$s = \frac{1}{c}(\theta - \theta_0) \qquad t = \frac{1}{d}(h - h_0)$$

EECS 487

OpenGL Implementation

Lecture 10



- Texture Mapping in OpenGL
  - Create a Texture Object
    - Specify Texture for the Object
  - Define How the Texture is to be Applied
    - GL_DECAL, GL_REPLACE
  - Enable Texture Mapping
  - Draw the Scene
    - Supply Texture Coordinates
    - Supply Geometric Coordinates

•Size of Textures Must be a Power of 2

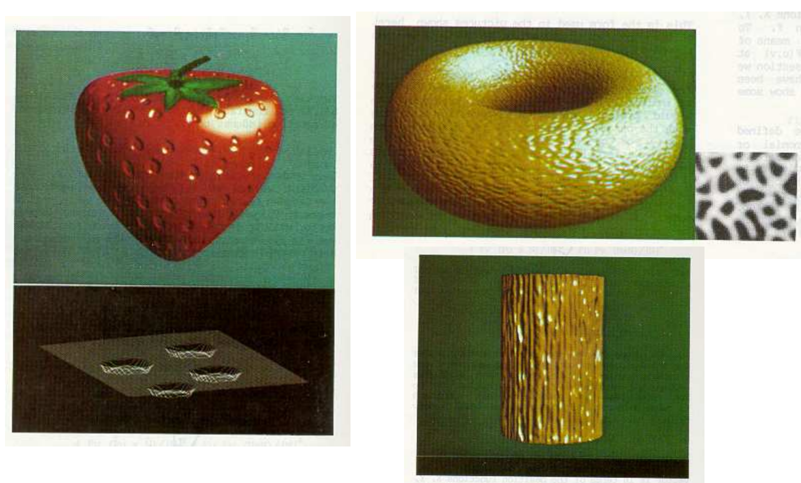
• Minimum Size : 64x64

11

EECS 487

Bump Mapping

Lecture 10

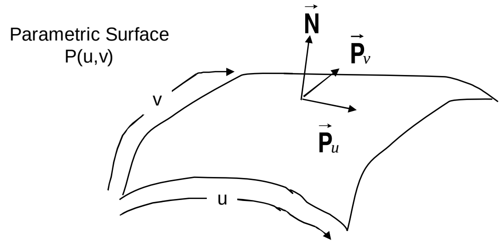


12

Bump Mapping

Lecture 10

- Bump Mapping
  - Texture map image shadows won't change if scene lighting changes
  - Modify surface itself instead of surface color
  - Simulate wrinkled/rough surfaces [Blinn `78]



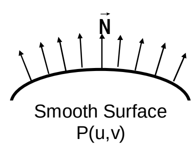
$$\vec{\mathbf{P}}_u = \frac{\partial}{\partial u}\vec{\mathbf{P}}(u,v)$$

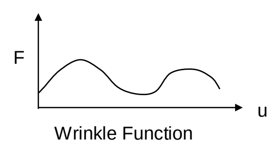
$$\vec{\mathbf{P}}_v = \frac{\partial}{\partial v}\vec{\mathbf{P}}(u,v)$$

$$\vec{\mathbf{N}} = \vec{\mathbf{P}}_u \times \vec{\mathbf{P}}_v$$

Bump Mapping

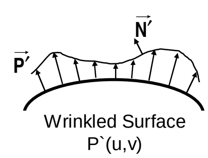
Lecture 10





$$\vec{\mathbf{P}}' = \vec{\mathbf{P}} + F(u,v)\hat{\mathbf{n}}$$

where F is the wrinkle function



$$\vec{\mathbf{N}}' = \vec{\mathbf{P}}'_u \times \vec{\mathbf{P}}'_v$$

Bump Mapping

Lecture 10

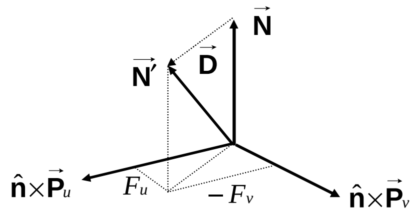
$$\vec{\mathbf{N}}' = \vec{\mathbf{P}}'_u \times \vec{\mathbf{P}}'_v$$

$$\vec{\mathbf{P}}'_u = \frac{\partial}{\partial u}(\vec{\mathbf{P}} + F\hat{\mathbf{n}}) = \vec{\mathbf{P}}_u + F_u\hat{\mathbf{n}} + F\hat{\mathbf{n}}_u$$

$$\vec{\mathbf{P}}'_v = \frac{\partial}{\partial u}(\vec{\mathbf{P}} + F\hat{\mathbf{n}}) = \vec{\mathbf{P}}_v + F_v\hat{\mathbf{n}} + F\hat{\mathbf{n}}_v$$

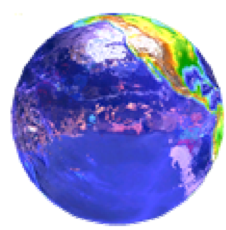
Let *F* be small, so that the last term can be neglected

$$\vec{\mathbf{N}}' = \vec{\mathbf{N}} + \vec{\mathbf{D}} = \vec{\mathbf{N}} + \{F_u(\hat{\mathbf{n}} \times \vec{\mathbf{P}}_v) - F_v(\hat{\mathbf{n}} \times \vec{\mathbf{P}}_u)\}$$



15

Bump maps



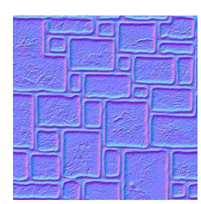
- Modify surface normal
- Produce normals per pixel
  - Store normal maps in textures
  - Generate from height maps
    - Grayscale image stores height
    - Compute

      Nx = (H[s+1,t] - H[s-1,t])/ds

      Ny = (H[s,t+1] - H[s,t-1])/dt

      Nz = 1

      (R, G, B) = normalized (Nx, Ny, Nz) then packed into bytes



Advanced Texturing

Lecture 10

Mip Mapping

Environment Mapping

Procedural Textures

17

MIP Maps

Lecture 10

MIP from Latin: multim in parvo: “Many thing a small place”

- Provides anti-aliasing
- Pre-filtered, offline
- Multiple levels of detail
- mip maps accessed through u,v,d
- gluBuild2DMipmaps

18

Environment Maps

Lecture 10

- Simulates specular reflected scene information
- Provides a virtual 1-ray ray-trace
- Only provides far field reflection correctly
- Can be mapped to a cube for easy access

19

Environment Maps

Lecture 10

20

Environment Maps

Lecture 10

21

Procedural Textures

Lecture 10

- Textures Based on functions not images
- Simulate the natural world through simplified physics
- Noise generators and fractals
- 3-D textures
- Not only colors but materials, motion …..

22

Procedural Textures

Lecture 10

Advantages

- Very compact when compared to images.
- No fixed resolution
- No fixed area
- Can be parameterized for general classes

Disadvantages

- Difficult to code and debug
- Not always predictable
- Can be slow
- Aliasing can be a problem

23

Procedural Textures

Lecture 10

Wood Example

Simulate wood in 3-D

Growth rings

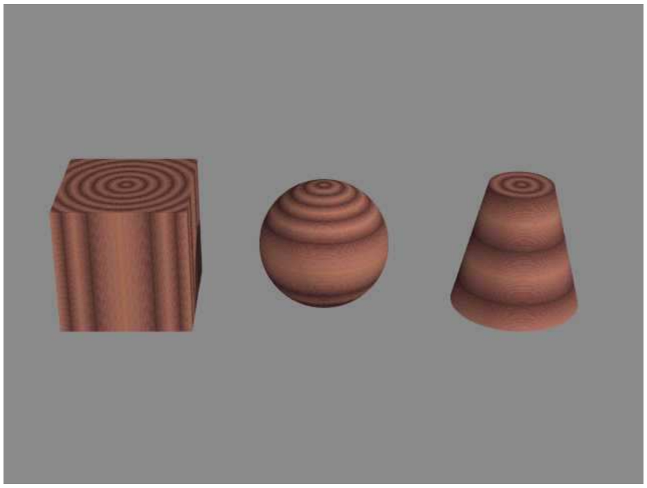
Color

Position

24

Procedural Textures

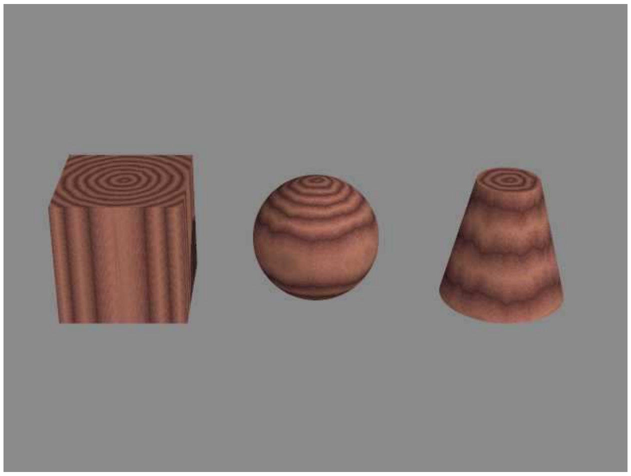
Lecture 10



25

Procedural Textures

Lecture 10



26

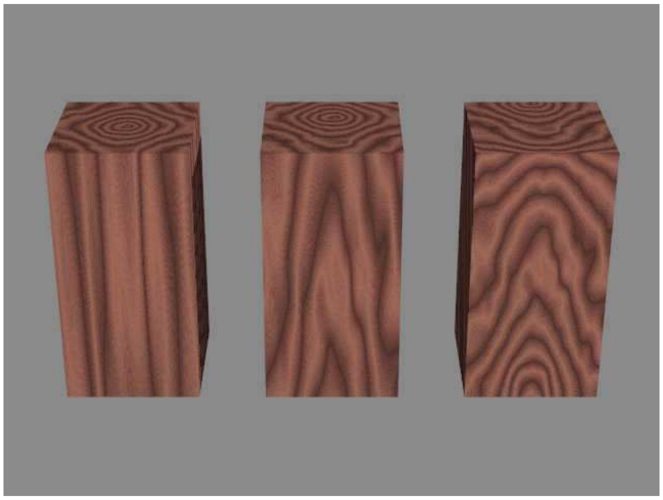
Procedural Textures

Lecture 10

27

Procedural Textures

Lecture 10



28

Procedural Textures

Lecture 10

29

Shadows

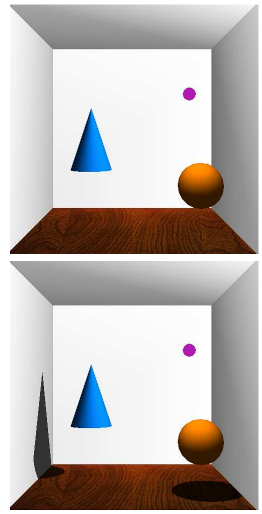
Lecture 10

Shadows

30

Shadows

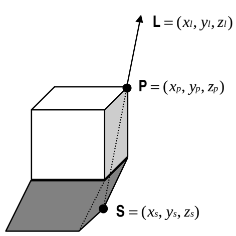
- Important for spatial relations
  - Where the lights are
- Light
  - Point
- Occluders
- Receivers
  - Occluders cast shadows onto receivers
- Easier if something is planar
  - Point and plane projection



31

Shadows

Lecture 10



- Shadows on a Ground Plane
  - Single Point Source
  - Infinite Distance (Parallel Rays)

$$\mathbf{S} = \mathbf{P} - \alpha \mathbf{L}$$

Since $z_s = 0$,
$0 = z_p - \alpha z_l$
$\alpha = z_p / z_l$

$$x_s = x_p - (z_p/z_l)x_l$$
$$y_s = y_p - (z_p/z_l)y_l$$
$$z_s = 0$$

$$\begin{bmatrix} x_s \\ y_s \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & -x_l/z_l & 0 \\ 0 & 1 & -y_l/z_l & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_p \\ y_p \\ z_p \\ 1 \end{bmatrix}$$

32

Projected shadows

Lecture 10

- Simple algorithm
  - Draw projected occluders on top of receivers
    - In dark color
      - Artifacts are usual
      - May try offsetting
- Proper projected shadows
  - First draw receiving planar polygon
  - Disable Z-buffering
    - Draw projected occluder
      - Only where receiver is drawn
      - Use stencil buffer for this

33

Projected shadows

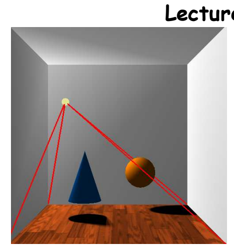
Lecture 10

- Handle simple situations
- Often faster to create shadow textures and move them
  - Changing texture coordinates
  - Or texture coords transform matrix
- To create softer shadows
  - Accumulate several point lights
    - Distributed over an area
  - Can render into a texture and then filter
    - Recompute texture each time light moves

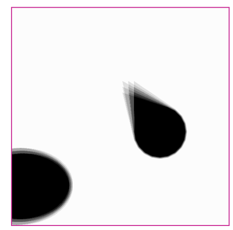
34

Shadow textures

Lecture



- Point light
  - No self-shadowing
  - Occluders
    - Rendered into shadow map
      - From the light's point of view
  - Receivers
    - Per-pixel lookup of shadow texture
      - Modulate with that
    - Lookup based on generated coordinates
      - Like light map



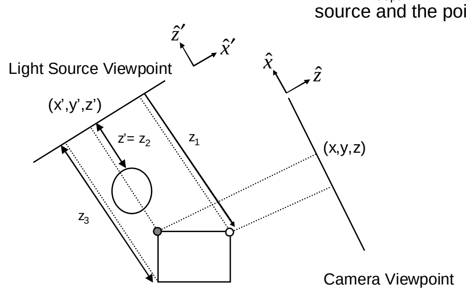
  - Resolution needs to be high for sharp shadows

35

Shadows

Lecture 10

- Zbuffer Transform
  - Simple two-step approach
    - Render depth map from illumination viewpoint
    - Render from camera viewpoint
      - If point is visible…
        - Transform (x,y,z) screen coords from camera view to (x',y',z') screen coords from light view
        - If (z' > z_{depth}) then a surface is between the light source and the point, so shadow the point

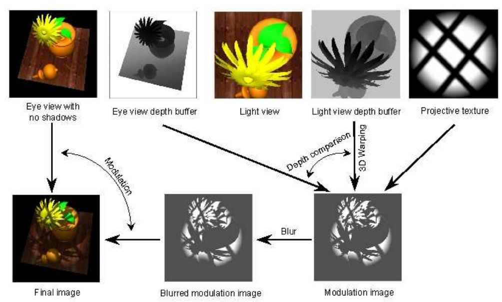


36

Shadow maps

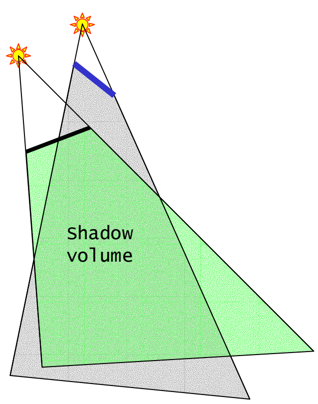
Lecture 10

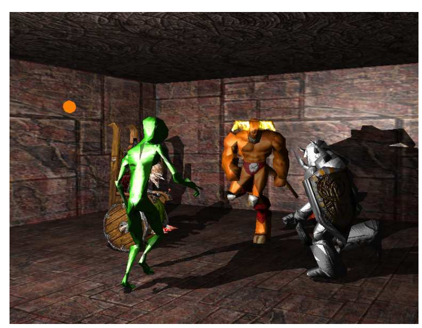
- Williams 1978
  - Like shadow texture
  - But render DEPTH from light
  - Use it when shading pixels

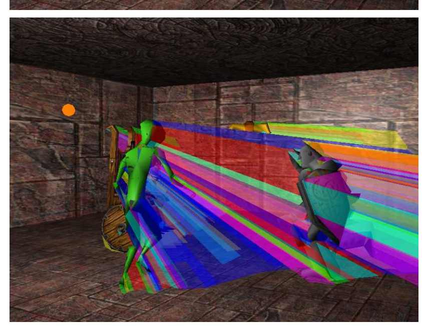


Shadow volumes

- Arbitrary occluders
- Self-shadowing

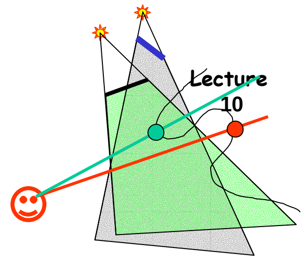






Shadow volumes

Lecture 10



- Several stages
  - Clear stencil buffer
  - Render scene in ambient and depth
  - Z-buffer updates and color off
    - Z-test still on
  - Draw front facing polygons of shadow volume
    - Increment stencil values
  - Draw back facing
    - Decrement stencil values
  - Render full scene diffuse and specular
    - Where stencil is zero

39

Color Models

Lecture 10

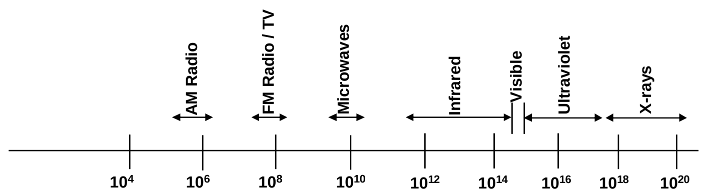
Color Models

40

Color Models

Lecture 10

• Electromagnetic Spectrum



• Visible Wavelengths (ROY G BIV)
- Red ~ 670 nm ~ $4.3\text{x}10^{14}$ Hz
- Violet ~ 420 nm ~ $7.5\text{x}10^{14}$ Hz

41

Color Models

Lecture 10

•Color of an Object Determined by Reflected Wavelengths

• White = All Wavelengths/Frequencies

$$c = \lambda \cdot \upsilon$$

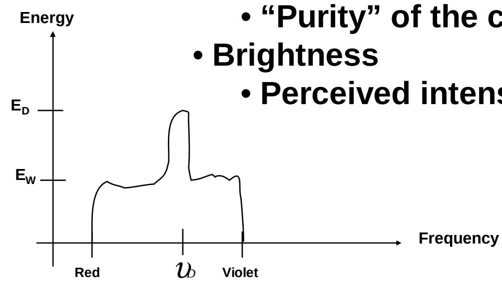
$$c = 3 \times 10^{10} \text{ cm/sec}$$

42

Color Models

Lecture 10

- Color Description
  - Hue
    - Dominant wavelength/frequency
  - Saturation
    - "Purity" of the color
  - Brightness
    - Perceived intensity

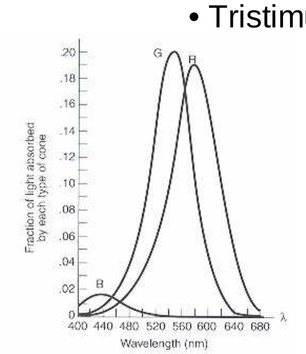


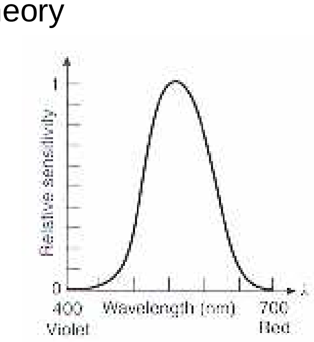
43

Color Models

Lecture 10

- Colors Not in a Rainbow?
  - Color combination from multiple sources
  - Human perception
    - Tristimulus Theory



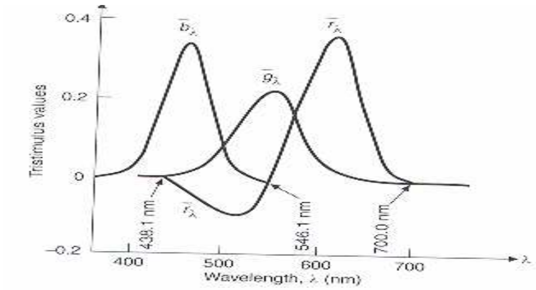


44

Color Models

Lecture 10

- Color Matching
  - Choose weights for three sources (primaries)
  - Combine sources to produce sample color
  - If no match can be obtained, add a primary to sample
    - Negative weight



45

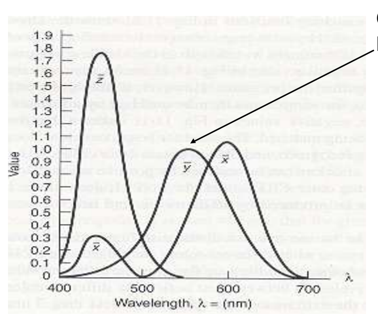
Color Models

Lecture 10

- International Commission on Illumination (CIE)
  - Define three primaries (imaginary colors)
  - Combine primaries with positive weights

$$C_\lambda = X\mathbf{X} + Y\mathbf{Y} + Z\mathbf{Z}$$

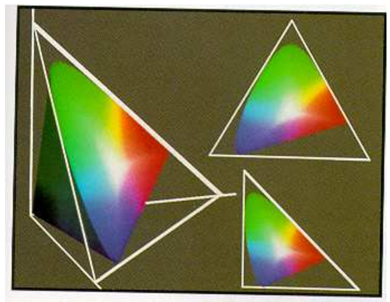
Chosen to match Luminous efficiency



46

Color Models

Lecture 10



• Normalize against Luminance (X+Y+Z)

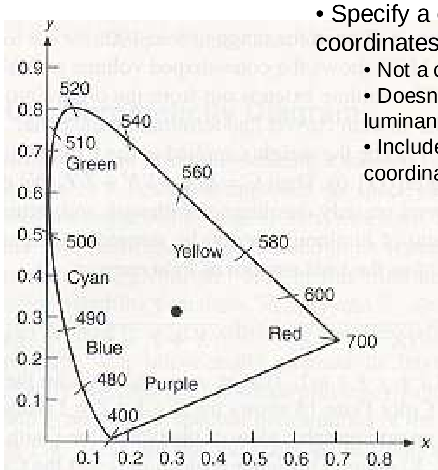
$$x = \frac{X}{X+Y+Z} \qquad y = \frac{Y}{X+Y+Z} \qquad z = \frac{Z}{X+Y+Z}$$

$$(x + y + z = 1)$$

47

Color Models

Lecture 10



• Specify a color with chromaticity coordinates (x,y)
  • Not a complete color palette
  • Doesn't account for color changes due to luminance
  • Include luminance information in coordinate (x,y,Y)

$$X = \frac{x}{y} Y$$

$$Y = Y$$

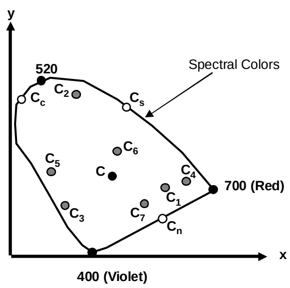
$$Z = \frac{(1 - x - y)}{y} Y$$

48

Color Models

Lecture 10

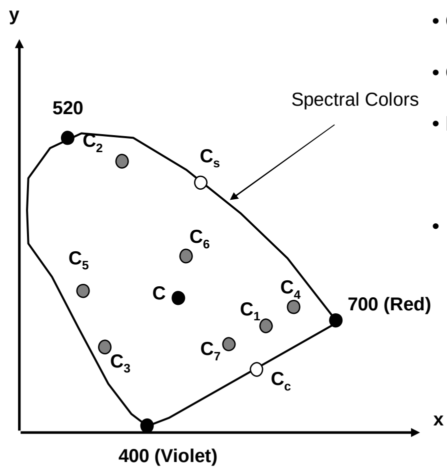
- Chromaticity Diagram
  - Compare color gamuts for different primaries
  - Identify complementary colors
  - Determine dominant wavelength and purity of a given color



49

Color Models

Lecture 10



- **Color Gamut ==> C_1, C_2, C_3**
- **Complementary Colors ==> C_4, C_5**
- **Dominant Wavelength**
  - **C_6 ==> C_s**
  - **C_7 ==> C_c**
- **Purity ==> Distance from C**
  - **C_4 very pure**
  - **C_6 not very pure**

C_n

50

Color Models

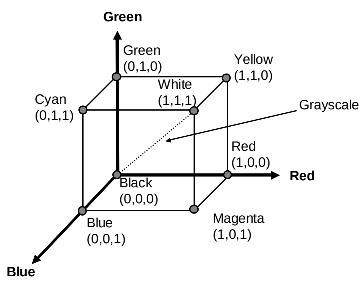
Lecture 10

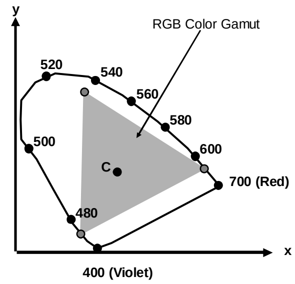
- RGB Color Model
  - Additive Primaries

$$C_\lambda = R\mathbf{R} + G\mathbf{G} + B\mathbf{B}$$

<u>Chromaticity Coordinates</u>

R	(0.735, 0.265)
G	(0.274, 0.717)
B	(0.167, 0.009





51

Color Models

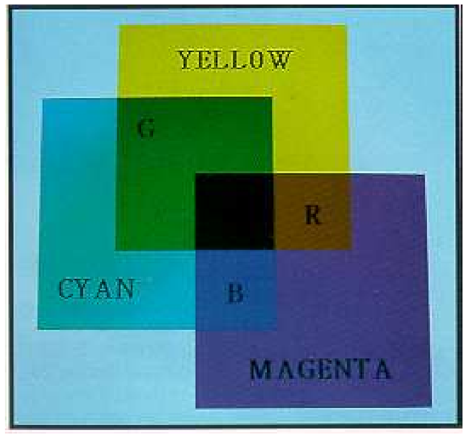
Lecture 10

52

Color Models

Lecture 10

- CMY Color Model
  - Cyan, Magenta, Yellow
  - Complements of RGB
  - Subtractive Primaries
  - Hardcopy Devices
  - Combining Pigments



53

Color Models

Lecture 10

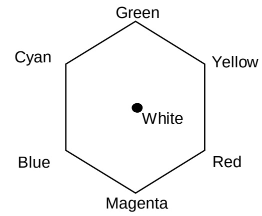
- YIQ Color Model (there is also YUV)
  - Re-coded RGB for NTSC transmission efficiency
  - Y component is luminance (CIE)
  - Black and white TV displays only the Y component
    - Largest bandwidth in the NTSC video signal (4 MHz)
  - Chromaticity encoded in I and Q

$$\begin{bmatrix} Y \\ I \\ Q \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.144 \\ 0.596 & -0.275 & -0.321 \\ 0.212 & -0.528 & 0.311 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

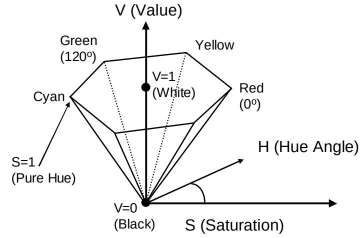
54

Color Models

- HSV Color Model
  - User intuitive
  - Shades
    - Addition of black pigment to a given hue
  - Tints
    - Addition of white pigment to a given hue
  - Tones
    - Addition of both black and white to a hue



- Vary H (Hue) ==> Select Color
- Decrease S (Saturation) ==> Add white
- Decrease V (Value) ==> Add Black



55