University of British Columbia
CPSC 314 Computer Graphics
Jan-Apr 2010

Tamara Munzner

Viewing/Projection VI, Vision/Color

Week 5, Wed Feb 2

http://www.ugrad.cs.ubc.ca/~cs314/Vjan2010

News

- showing up for your project grading slot is **not** optional
  - 2% penalty for noshows
- signing up for your project grading slot is **not** optional
  - 2% penalty for nosignups within two days of due date
  - your responsibility to sign up for slot
    - not ours to hunt you down if you chose to skip class on signup days
- we do make best effort to accomodate change requests via email to grader for that project
- take a few minutes to review your code/README to reload your mental buffers
  - TA will ask you questions about how you did things

News

- Homework 2 out
  - due Fri Feb 12 5pm
- Project 2 out
  - due Tue Mar 2 5pm
  - moved due date to after break after pleas of pre-break overload with too many assignments due
  - start early, do *not* leave until late in break!!
- reminder
  - extra handouts in lab

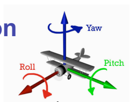
Project 2: RCSS

- solar system
  - planets spin around own axis and sun
  - moon spins around earth
- two spaceships: mothership and scoutship
  - one window for each
  - may see geometry of one spaceship through window of other
- navigation modes
  - solar system coord (absolute) rotate/translate
  - through the lens flying (relative to camera)
  - geosynchronous orbit around planet
    - zoom in/out towards center of planet

Project 2 Hints

- don't forget to keep viewing and projections in their respective stacks
- try drawing scene graphs to help you figure out how to place multiple cameras
  - especially geosynchronous: camera as child of object in world in the scene graph
  - geometric representation of camera vs. what is shown through its window
- disk for Saturn rings: try scaling sphere by 0
- OK to reset camera position between absolute/relative navigation modes
- OK to have camera jumpcut to different orientation when new planet picked in geosync mode

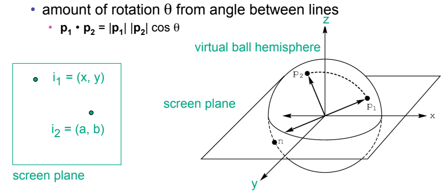
Review/More: Relative Motion



- how to move relative to current camera?
  - what you see in the window
- computation in coordinate system used to draw previous frame is simple:
  - incremental change I to current C
  - each time we just want to premultiply by new matrix
    - p'=ICp
  - but we know that OpenGL only supports postmultiply by new matrix
    - p'=CIp
- use OpenGL matrix stack as calculator/storage!
  - dump out modelview matrix from previous frame with glGetDoublev()
    - C = current camera coordinate matrix
  - wipe the matrix stack with glIdentity()
  - apply incremental update matrix I
  - apply current camera coord matrix C

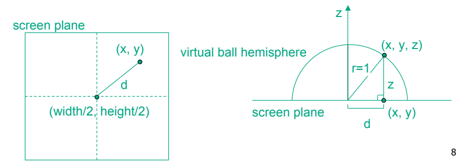
Review/Clarify: Trackball Rotation

- user drags between two points on image plane
  - mouse down at i_1 = (x, y), mouse up at i_2 = (a, b)
- find corresponding points on virtual ball
  - $\mathbf{p}_1$ = (x, y, z), $\mathbf{p}_2$ = (a, b, c)
- compute rotation angle and axis for ball
  - axis of rotation is plane normal: cross product $\mathbf{p}_1 \times \mathbf{p}_2$
  - amount of rotation θ from angle between lines
    - $\mathbf{p}_1 \bullet \mathbf{p}_2 = |\mathbf{p}_1|\,|\mathbf{p}_2| \cos\theta$



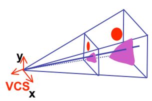
Clarify: Trackball Rotation

- finding location on ball corresponding to click on image plane
  - ball radius r is 1

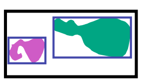


Review: Picking Methods

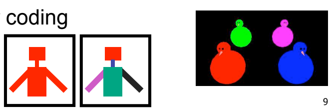
- manual ray intersection



- bounding extents



- backbuffer coding

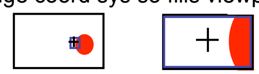


Review: Select/Hit Picking

- use small region around cursor for viewport
- assign per-object integer keys (names)
- redraw in special mode
- store hit list of objects in region
- examine hit list
- OpenGL support

Viewport

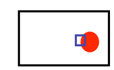
- small rectangle around cursor
  - change coord sys so fills viewport



- why rectangle instead of point?
  - people aren't great at positioning mouse
    - Fitts' Law: time to acquire a target is function of the distance to and size of the target
  - allow several pixels of slop

Viewport

- nontrivial to compute
  - invert viewport matrix, set up new orthogonal projection
- simple utility command
  - gluPickMatrix(x,y,w,h,viewport)
    - x,y: cursor point
    - w,h: sensitivity/slop (in pixels)
  - push old setup first, so can pop it later



Render Modes

- glRenderMode(mode)
  - GL_RENDER: normal color buffer
    - default
  - GL_SELECT: selection mode for picking
  - (GL_FEEDBACK: report objects drawn)

Name Stack

- again, "names" are just integers
  glInitNames()
- flat list
  glLoadName(name)
- or hierarchy supported by stack
  glPushName(name), glPopName
  - can have multiple names per object

Hierarchical Names Example

```
for(int i = 0; i < 2; i++) {
  glPushName(i);
  for(int j = 0; j < 2; j++) {
    glPushMatrix();
    glPushName(j);
    glTranslatef(i*10.0,0,j * 10.0);
      glPushName(HEAD);
      glCallList(snowManHeadDL);
      glLoadName(BODY);
      glCallList(snowManBodyDL);
      glPopName();
    glPopName();
    glPopMatrix();
  }
  glPopName();
}
```

http://www.lighthouse3d.com/opengl/picking/

Hit List

- glSelectBuffer(buffersize, *buffer)
  - where to store hit list data
- on hit, copy entire contents of name stack to output buffer.
- hit record
  - number of names on stack
  - minimum and minimum depth of object vertices
    - depth lies in the NDC z range [0,1]
    - format: multiplied by 2^32 -1 then rounded to nearest int

Integrated vs. Separate Pick Function

- integrate: use same function to draw and pick
  - simpler to code
  - name stack commands ignored in render mode
- separate: customize functions for each
  - potentially more efficient
  - can avoid drawing unpickable objects

Select/Hit

- advantages
  - faster
    - OpenGL support means hardware acceleration
    - avoid shading overhead
  - flexible precision
    - size of region controllable
  - flexible architecture
    - custom code possible, e.g. guaranteed frame rate
- disadvantages
  - more complex

Hybrid Picking

- select/hit approach: fast, coarse
  - object-level granularity
- manual ray intersection: slow, precise
  - exact intersection point
- hybrid: both speed and precision
  - use select/hit to find object
  - then intersect ray with that object

High-Precision Picking with OpenGL

- gluUnproject
  - transform window coordinates to object coordinates given current projection and modelview matrices
  - use to create ray into scene from cursor location
  - call gluUnProject twice with same (x,y) mouse location
    - z = near: (x,y,0)
    - z = far: (x,y,1)
    - subtract near result from far result to get direction vector for ray
- use this ray for line/polygon intersection

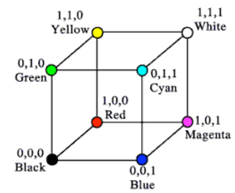
Vision/Color

Reading for Color

- RB Chap Color
- FCG Sections 3.2-3.3
- FCG Chap 20 Color
- FCG Chap 21.2.2 Visual Perception (Color)

RGB Color

- triple (r, g, b) represents colors with amount of red, green, and blue
  - hardware-centric
  - used by OpenGL



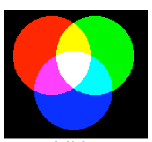
Alpha

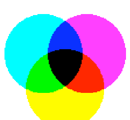
- fourth component for transparency
  - (r,g,b,α)
- fraction we can see through
  - c = $\alpha c_f + (1-\alpha)c_b$
- more on compositing later

Additive vs. Subtractive Colors

- additive: light
  - monitors, LCDs
  - RGB model
- subtractive: pigment
  - printers
  - CMY model
  - dyes absorb light

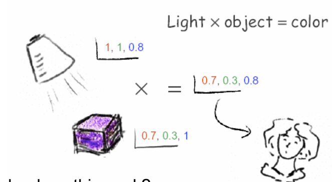
$$\begin{bmatrix} C \\ M \\ Y \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} - \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$


additive


subtractive

Component Color

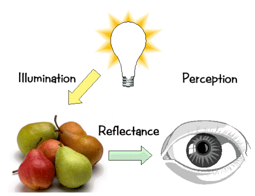
- component-wise multiplication of colors
  - (a0,a1,a2) * (b0,b1,b2) = (a0*b0, a1*b1, a2*b2)



- why does this work?
  - must dive into light, human vision, color spaces

Basics Of Color

- elements of color:



Basics of Color

- physics
  - illumination
    - electromagnetic spectra
  - reflection
    - material properties
    - surface geometry and microgeometry
      - polished versus matte versus brushed
- perception
  - physiology and neurophysiology
  - perceptual psychology

Light Sources

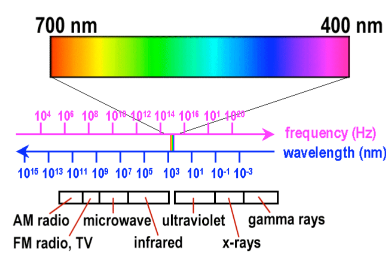
- common light sources differ in kind of spectrum they emit:
  - continuous spectrum
    - energy is emitted at all wavelengths
      - blackbody radiation
      - tungsten light bulbs
      - certain fluorescent lights
      - sunlight
      - electrical arcs
  - line spectrum
    - energy is emitted at certain discrete frequencies

Blackbody Radiation

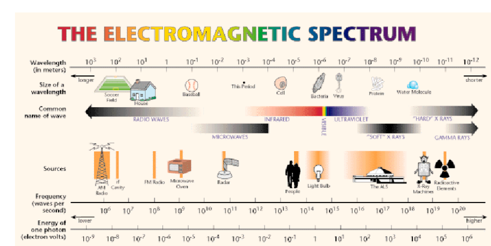
- black body
  - dark material, so that reflection can be neglected
  - spectrum of emitted light changes with temperature
    - this is the origin of the term "color temperature"
      - e.g. when setting a white point for your monitor
    - cold: mostly infrared
    - hot: reddish
    - very hot: bluish
  - demo:

http://www.mhhe.com/physsci/astronomy/applets/Blackbody/frame.html

Electromagnetic Spectrum

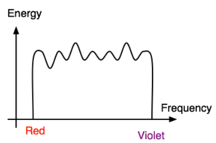


Electromagnetic Spectrum



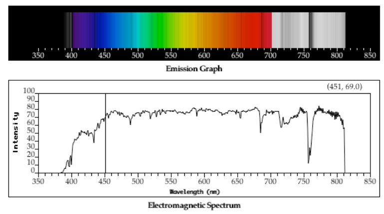
White Light

- sun or light bulbs emit all frequencies within visible range to produce what we perceive as "white light"



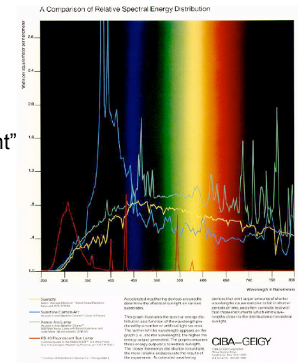
Sunlight Spectrum

- spectral distribution: power vs. wavelength



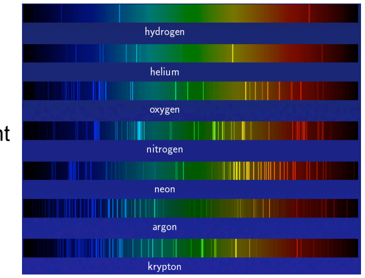
Continuous Spectrum

- sunlight
- various "daylight" lamps



Line Spectrum

- ionized gases
- lasers
- some fluorescent lamps

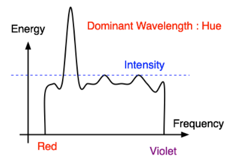


White Light and Color

- when white light is incident upon an object, some frequencies are reflected and some are absorbed by the object
- combination of frequencies present in the reflected light that determines what we perceive as the color of the object

Hue

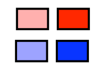
- hue (or simply, "color") is dominant wavelength/frequency

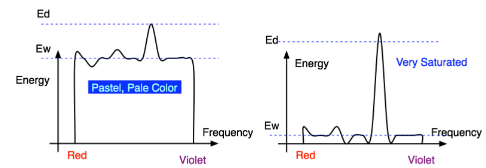


- integration of energy for all visible wavelengths is proportional to intensity of color

Saturation or Purity of Light

- how washed out or how pure the color of the light appears
  - contribution of dominant light vs. other frequencies producing white light
  - saturation: how far is color from grey
    - pink is less saturated than red
    - sky blue is less saturated than royal blue





Intensity vs. Brightness

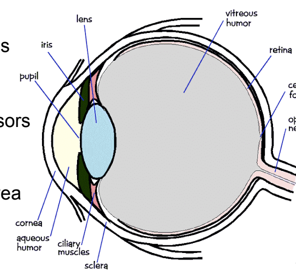
- intensity : physical term
  - measured radiant energy emitted per unit of time, per unit solid angle, and per unit projected area of the source (related to the luminance of the source)
- lightness/brightness: perceived intensity of light
  - nonlinear

Perceptual vs. Colorimetric Terms

- Perceptual
  - Hue
  - Saturation
  - Lightness
    - *reflecting objects*
  - Brightness
    - *light sources*
- Colorimetric
  - Dominant wavelength
  - Excitation purity
  - Luminance
  - Luminance

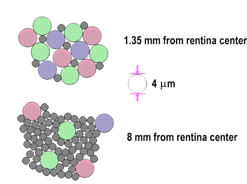
Physiology of Vision

- the retina
  - rods
    - b/w, edges
  - cones
    - 3 types
    - color sensors
  - uneven distribution
    - dense fovea



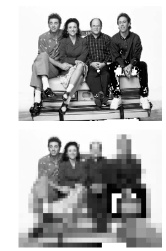
Physiology of Vision

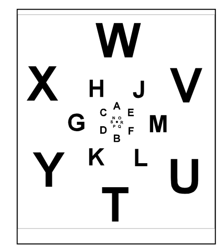
- Center of retina is densely packed region called the *fovea*.
  - Cones much denser here than the *periphery*



Foveal Vision

- hold out your thumb at arm's length



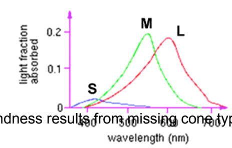


Tristimulus Theory of Color Vision

- Although light sources can have extremely complex spectra, it was empirically determined that colors could be described by only 3 primaries
- Colors that look the same but have different spectra are called metamers

Trichromacy

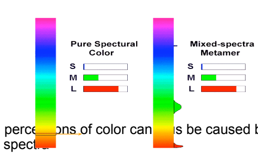
- three types of cones
  - L or R, most sensitive to red light (610 nm)
  - M or G, most sensitive to green light (560 nm)
  - S or B, most sensitive to blue light (430 nm)



- color blindness results from missing cone type(s)

Metamers

- a given perceptual sensation of color derives from the stimulus of all three cone types

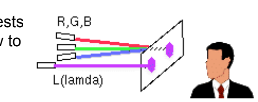


- identical perce ns of color can s be caused by very different spect
- demo

http://www.cs.brown.edu/exploratories/freeSoftware/catalogs/color_theory.html

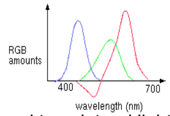
Color Spaces

- three types of cones suggests color is a 3D quantity. how to define 3D color space?



- idea: perceptually based measurement
  - shine given wavelength (λ) on a screen
  - user must control three pure lights producing three other wavelengths
    - used R=700nm, G=546nm, and B=436nm
  - adjust intensity of RGB until colors are identical
    - this works because of metamers!
    - experiments performed in 1930s

Negative Lobes

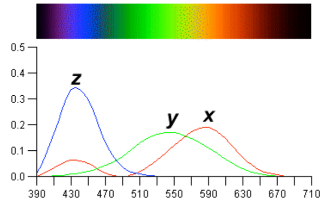


- sometimes need to point red light to shine on target in order to match colors
  - equivalent mathematically to "removing red"
    - but physically impossible to remove red from CRT phosphors
- can't generate all other wavelenths with any set of three positive monochromatic lights!
- solution: convert to new synthetic coordinate system to make the job easy

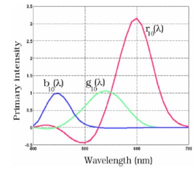
CIE Color Space

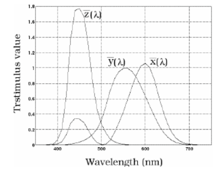
- CIE defined 3 "imaginary" lights X, Y, Z
  - any wavelength λ can be matched perceptually by positive combinations

Note that:
X ~ R
Y ~ G
Z ~ B



Measured vs. CIE Color Spaces

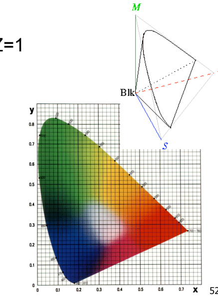




- measured basis
  - monochromatic lights
  - physical observations
  - negative lobes
- transformed basis
  - "imaginary" lights
  - all positive, unit area
  - Y is luminance, no hue
  - X,Z no luminance

CIE and Chromaticity Diagram

- X, Y, Z form 3D shape
- project X, Y, Z on X+Y+Z=1 plane for 2D color space
  - chromaticity diagram
    - separate color from brightness
    - x = X / (X+Y+Z)
    - y = Y / (X+Y+Z)



CIE "Horseshoe" Diagram Facts

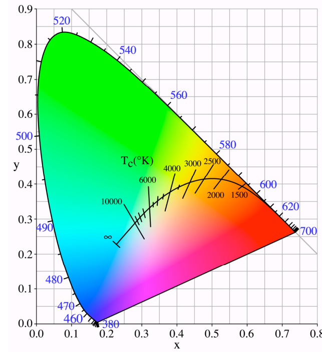
- all visible colors lie inside the horseshoe
  - result from color matching experiments
- spectral (monochromatic) colors lie around the border
  - straight line between blue and red contains purple tones
- colors combine linearly (i.e. along lines), since the xy-plane is a plane from a linear space

CIE "Horseshoe" Diagram Facts

- can choose a point C for a white point
  - corresponds to an illuminant
  - usually on curve swept out by black body radiation spectra for different temperatures

Blackbody Curve

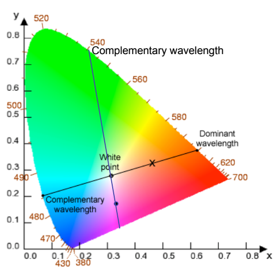
- illumination:
  - candle 2000K
  - A: Light bulb 3000K
  - sunset/ sunrise 3200K
  - D: daylight 6500K
  - overcast day 7000K
  - lightning >20,000K



CIE "Horseshoe" Diagram Facts

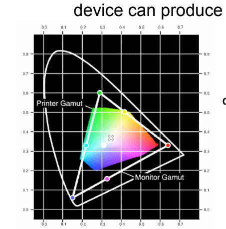
- can choose a point C for a white point
  - corresponds to an illuminant
  - usually on curve swept out by black body radiation spectra for different temperatures
  - two colors are complementary relative to C when are
    - located on opposite sides of line segment through C
      - so C is an affine combination of the two colors
  - find dominant wavelength of a color:
    - extend line from C through color to edge of diagram
    - some colors (i.e. purples) do not have a dominant wavelength, but their complementary color does

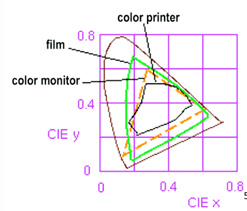
Color Interpolation, Dominant & Opponent Wavelength



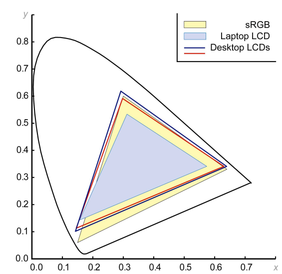
Device Color Gamuts

- gamut is polygon, device primaries at corners
  - defines reproducible color range
  - X, Y, and Z are hypothetical light sources, no device can produce entire gamut



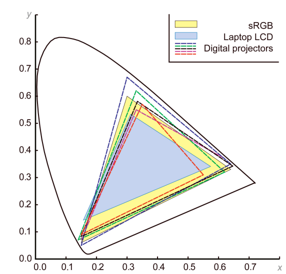


Display Gamuts



From A Field Guide to Digital Color, © A.K. Peters, 2003

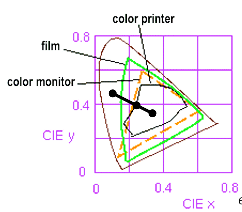
Projector Gamuts



From A Field Guide to Digital Color, © A.K. Peters, 2003

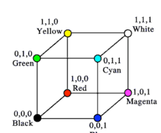
Gamut Mapping

- how to handle colors outside gamut?
  - one way: construct ray to white point, find closest displayable point within gamut

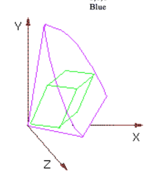


RGB Color Space (Color Cube)

- define colors with (r, g, b) amounts of red, green, and blue
  - used by OpenGL
  - hardware-centric

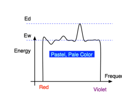


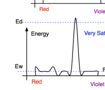
- RGB color cube sits within CIE color space
  - subset of perceivable colors
  - scale, rotate, shear cube

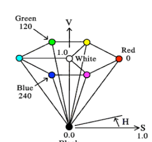


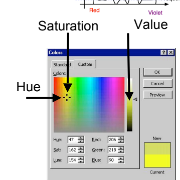
HSV Color Space

- more intuitive color space for people
  - H = Hue
    - dominant wavelength, "color"
  - S = Saturation
    - how far from grey/white
  - V = Value
    - how far from black/white
    - also: brightness B, intensity I, lightness L









HSI/HSV and RGB

- HSV/HSI conversion from RGB not expressible in matrix
  - H=hue same in both
  - V=value is max, I=intensity is average

$$H = \cos^{-1}\left[\frac{\frac{1}{2}\left[(R-G)+(R-B)\right]}{\sqrt{(R-G)^2+(R-B)(G-B)}}\right] \quad \text{if } (B > G),\ H = 360 - H$$

HSI: $S = 1 - \frac{\min(R,G,B)}{I}$ $\quad I = \frac{R+G+B}{3}$

HSV: $S = 1 - \frac{\min(R,G,B)}{V}$ $\quad V = \max(R,G,B)$