Color
“Today’s Lecturer”: Kristi Potter

- Previous:
  - B.S., Univ. of OR: CIS & Fine Arts
  - Ph.D., Univ. of Utah
  - Professional Researcher, Univ. of Utah

- Current:
  - Scientific Programmer, CASSPR, Univ. of OR
  - Courtesy position, CIS
  - Co-founder, CDUX
  - Campus “visualization evangelist”

- Research expertise:
  - Scientific visualization, esp. visualization of ensembles and uncertainty visualization
Physics of Light
The Electromagnetic Spectrum

<table>
<thead>
<tr>
<th>Gamma Rays</th>
<th>X-Rays</th>
<th>Ultraviolet Rays</th>
<th>Infrared Rays</th>
<th>Radar</th>
<th>FM</th>
<th>TV</th>
<th>Shortwave</th>
<th>AM</th>
</tr>
</thead>
<tbody>
<tr>
<td>$1 \times 10^{-14}$</td>
<td>$1 \times 10^{-12}$</td>
<td>$1 \times 10^{-8}$</td>
<td>$1 \times 10^{-4}$</td>
<td>$1 \times 10^{-2}$</td>
<td>$1 \times 10^2$</td>
<td>$1 \times 10^4$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Wavelength (in meters)

Visible Light

Wavelength (in meters)

High Energy

Low Energy
The Visible Spectrum

human visual system sensitive to \(\sim 380-780\) nm
- white light is a combination of all colors of the spectrum
- black is the absence of visible spectrum wavelengths
- separation of visible light into colors is called *dispersion*

**objects appear colored by the character of the light they reflect**
Images
Background on Images

• Definitions:
  – Image: 2D array of pixels
  – Pixel: A minute area of illumination on a display screen, one of many from which an image is composed.

• Pixels are made up of three colors: Red, Green, Blue (RGB)

• Amount of each color scored from 0 to 1
  – 100% Red + 100% Green + 0% Blue = Yellow
  – 100% Red + 0% Green + 100 %Blue = Purple
  – 0% Red + 100% Green + 100% Blue = Cyan
  – 100% Red + 100% Blue + 100% Green = White
Background on Images

• Colors are 0->1, but how much resolution is needed? How many bits should you use to represent the color?
  – Can your eye tell the difference between 8 bits and 32 bits?
  – → No. Human eye can distinguish ~10M colors.
  – 8bits * 3 colors = 24 bits = ~16M colors.

• Red = (255,0,0)
• Green = (0,255,0)
• Blue = (0,0,255)
How to organize a struct for an Image (i.e., 3D arrays)

• 3D array: width * height * 3 color channels

• Color:
  – Choice 1: RGB struct
    • struct rgb { unsigned char r, g, b; };
    • int npixels = width*height;
    • struct RGB *buffer = new RGB[npixels];
    • int p = 21456; // random pixel in the buffer
    • buffer[0].r = 255; buffer[0].g = 0; buffer[0].b = 0;
  – Choice 2: just 3 unsigned chars
How to organize a struct for an Image (i.e., 3D arrays)

• 3D array: width * height * 3 color channels

• Color:
  – Choice 1: RGB struct
  – Choice 2: just 3 unsigned chars
    • int npixels = width*height;
    • unsigned char *buffer = new unsigned char [3*npixels];
    • int p = 21456; // random pixel in the buffer
    • buffer[3*p+0] = 255; buffer[3*p+1] = 0;
    • buffer[3*p+2] = 0;
Project 3

• For project 3, I am doing the management of the buffer
  – I do choice 2
• But you will write functions that work on one pixel
• `void AssignValue (unsigned char *pixel) {
   pixel[0] = 255; pixel[1] = 0; pixel[2] = 0; 
};`
• My code:
  – `for (int i = 0 ; i < npixels ; i++) AssignValue(buffer[3*i]);`
Your Amazing Eyes
Crazy numbers about your eyes (with possibly some exaggeration)

• What is the pixel resolution of your eyes?
  
  50*MegaPixels
  (* = different parts of the eye work differently; 50M pixel is an aggregation)

• What is the frequency your eyes take in information?
  
  ~20* Hertz
  (* = for VR, 90Hz is sometimes needed. For video games, 30Hz almost always sufficient)

50MegaPixels x 20HZ → your eyes can take in 1GB of data per second
This is why visualization is king for understanding data.
Pseudocolor plot

• First visualization technique we will learn

• Idea: take a scalar field on a mesh and transform it to colors

• Two elements:
  – Define a map
  – Apply the map
Pseudocolor plot

- First visualization technique we will learn
- Idea: take a scalar field on a mesh and transform it to colors
- Two elements:
  - Define a map
  - Apply the map
Color map example ("discrete")

- Example below defines the color for scalar values between 0 and 1.29
- Grouped in 13 ranges ("discrete")

![Color map example](image.png)
Color map example ("continuous")

- Example below defines the color for scalar values between -0.10 and 0.10
- No groupings ("continuous")
- This color map appears to have a fixed number of colors, and then blends between those colors
  - Red, yellow, green, cyan, blue
    - (This is not immediately obvious)
Pseudocolor plot

• First visualization technique we will learn
• Idea: take a scalar field on a mesh and transform it to colors
• Two elements:
  – Define a map
  – Apply the map
Pseudocolor plot

- Defined:
  - Create a mapping, $M$, from scalar values to colors
  - For each pixel:
    - Evaluate $V$, the scalar field at that pixel location
    - Obtain color $C$, by applying $M$ to $V$
      - $C = M(V)$
    - Assign the pixel color to be $C$
Example pseudocolor plot (discrete color map)

Mean Daily Average Temperature
Source: Climate Atlas of the United States
Pseudocolor in practice

• The normal way:
  – Data set is composed of triangles
  – Apply color map to each vertex in the triangle
  – Tell graphics hardware to render each triangle (including color at each of three vertices)
  – Graphics hardware does the work of interpolating color at each pixel
Pseudocolor NOT in practice

• An unusual way:
  – Assume data perfectly overlaps with image
    • Bottom left of data in bottom left pixel
    • Upper right of data in upper right pixel
  – For each pixel
    • Find corresponding spatial location
    • Evaluate scalar field
    • Apply color map
    • Assign pixel

• In effect: doing what graphics card does in software
• And: this is what we will do in Project 3
Choosing Colors
Color Illusions
Cornsweet Illusion

- luminance the same at both ends
- many perceptions more sensitive to abrupt change (luminance, color, motion, depth)
- attributed to center/surround organization
Contrast Effects

- juxtaposition of colors affect our perception of them
- complementary colors often most affected
- simultaneous: how colors can affect each other
- successive: how color can affect future perception
Simultaneous Contrast
Simultaneous Contrast
Equiluminant Colors

- strong contrast
  shapes to be seen by color sensitive cells
- equiluminance
  hides positions from light sensitive cells
- flickering/movement caused by this disconnect
Color Spaces
RGB

Red, Green, Blue channels
- electron guns of CRT monitors
CMYK

subtractive
- Cyan, Magenta, Yellow, Key (black)
- inks used in printing
- assumes white paper (non-existence of ink)
- often required for paper publications
- device dependent
HSV [B, L, I]

additive

- **Hue**, **Saturation**, [**Value**, **Brightness**, **Lightness**, **Intensity**]
- polar coordinate representations of RGB space
- conical or cylindrical shaped space
- more intuitive than RGB for color tuning
CIE LAB/LUV

mathematically defined, perceptually based
- Commission Internationale de l'éclairage)
- Lightness, (a,b)/(u,v) color opponent dimensions
  a: blue/yellow, b: green/red
  lightness (0% black, 100% white)
- space includes all perceivable colors
  (gamut greater than RGB, CMYK)
- device independent
// in: rgb(0.0,1.0) out: h(0,360), v(0.0,1.0), s(0.0,1.0)
rgb2hsv(float r, float g, float b)
    // find the min and max of rgb
    maxColor = max(r, g, b);
    minColor = min(r, g, b);
    float delta = maxColor - minColor;
    float Value = maxColor;
    float Saturation = 0;
    float Hue = 0;
    if (delta == 0)
        Hue = 0; Saturation = 0;
    else
        Saturation = delta / maxColor;
    float dR = 60.f*(maxColor - r)/delta + 180.0;
    float dG = 60.f*(maxColor - g)/delta + 180.0;
    float dB = 60.f*(maxColor - b)/delta + 180.0;
    if (r == maxColor)
        Hue = dB - dG;
    else if (g == maxColor)
        Hue = 120 + dR - dB;
    else
        Hue = 240 + dG - dR;
    if (Hue < 0)
        Hue += 360;
    if (Hue >= 360)
        Hue -= 360;
/in: h(0,360), v(0.0,1.0), s(0.0,1.0) out: rgb(0.0,1.0)

hsvToRGB(float hue, float saturation, float value)
    if( saturation == 0 ) // achromatic (grey)
        r = g = b = v;
    else{
        hue /= 60.f;
        // sector 0 to 5
        i = floor( hue );
        f = hue - i;
        // factorial part of h
        p = value * ( 1 - saturation);
        q = value * ( 1 - saturation * f );
        t = value * ( 1 - saturation * ( 1 - f ) );
        switch( i ):
            case 0: r = v; g = t; b = p;
            case 1: r = q; g = v; b = p;
            case 2: r = p; g = v; b = t;
            case 3: r = p; g = q; b = v;
            case 4: r = t; g = p; b = v;
            case 5: r = v; g = p; b = q;
    }
Problems with Color

(short list)
Misused Color

“Color used poorly is worse than no color at all.”
-Edward Tufte

- bias a reader’s perception
- cause the wrong information stand out
- hide meaningful information
- cause visual clutter and overload
Color Coding

- hard to pick discernible colors
- can only get $\sim 12$ distinguishable colors
- object size, line width can influence perception

William S. Cleveland and Robert McGill.
"A Color-Caused Optical Illusion on a Statistical Graph".
Rainbow Colormaps

- not ideal for qualitative data
- no inherent ordering (by wavelength?)
- transitions between colors are uneven
- mapping of changing value may not equal the change we see

Best Practices
If you want objects to look the same color, make background colors consistent.
If you want objects to be easily seen, use a background color that contrasts sufficiently.
Use color only when needed to serve a particular communication goal.
Use different colors only when they correspond to differences of meaning in the data.
Use soft, natural colors to display most information and bright and/or dark colors to highlight.
Stick with a monochromatic color scheme when encoding quantitative values.
Non-data components should be displayed just visibly enough to perform their role.
To accommodate for the colorblind, avoid using a combination of red and green in the same display.

Vischeck/Daltonize
http://www.vischeck.com/
Tips
Color Schemes

- monochromatic
  variations in lightness & saturation of a single color
- analogous
  colors adjacent on the color wheel
- complementary
  two colors opposite on the color wheel
Look to Nature
Simplicity

- choose one color to be used in larger amounts
- be selective about the base color
- use other colors to add interest
Avoidance of Color

- use neutrals (work with any scheme)
  black, white, grey
- use diagrammatic marks (may be better encoding channels)
  size, shape, texture, length, width, orientation, curvature and intensity
Tools for Color Scheme Creation
Kuhler

http://kuler.adobe.com
- pick a previously created “theme”
- create a theme from color model
- create a new theme from an image
Color Scheme Designer

http://colorschemedesigner.com
- develop scheme based on a color model
- test out scheme on light and dark pages
Color Brewer

http://colorbrewer2.org/

- pre-defined color schemes for maps
  - sequential
    - optimized for data ordered low to high
  - diverging
    - place equal emphasis on mid-range and extremes
  - qualitative
    - does not imply an order (categorical data)
HCL Color Picker

http://tristen.ca/hcl-picker/
- pick equidistant colors in the HCL colorspace
- test color scheme out on a choropleth map
- addresses issues of HSV colorspace
Questions?

References
- Cinematic Color, Jeremy Selan, Siggraph 2012 Course notes
- Measuring Color [Hunt, 1998]
- Color Science [Wyszecki and Stiles, 1982]
- The Science of Art
  (http://www.imprint.co.uk/rama/art.pdf)
- How the Retina Works
  (http://www.americanscientist.org/libraries/documents/20058313632_306.pdf)
- Color Theory Methods for Visualization, Theresa-Marie Rhyne, Vis 2012 Tutorial
Project 3

- Assigned today, prompt online by tonight
- Due January 26th, midnight (→ January 27th, 6am)
- Worth 6% of your grade
- Provide:
  - Code skeleton online
  - Correct answers provided
- You send me:
  - source code
  - three images from your program
Project 3 in a nutshell

• I give you a 2D data set.
• You will produce 3 images that are 500x500 pixels.
• The 2D data set will be mapped on to the pixels.
• Pixel (0,0): X=-9, Y=-9
• Pixel (499, 0): X=+9, Y=-9, pixel (0,499): X=-9, Y=+9
• Pixel (499,499): X=+9, Y=+9
Project 3 in a nutshell

• For each of the 250,000 pixels (500x500), you will apply 3 color maps to a scalar field