Human Vision, Color and Basic Image Processing

Jason Lawrence

CS 4810: Graphics

Acknowledgement: slides by Misha Kazhdan, Allison Klein, Tom Funkhouser, Adam Finkelstein and David Dobkin
Outline

• Human Vision and Color
• Image Representation
• Reducing Color Quantization Artifacts
• Basic Image Processing
Human Vision

Model of Human Visual System
Human Vision

Model of Human Visual System

Vision Components:
- Incoming Light
- The Human Eye

Objects in world

Sun

Human eye
Color

- Two types of receptors:

  Rods and cones

  Cones in fovea
Rods and Cones

• Rods
  • More sensitive in low light: “scotopic” vision
  • More dense near periphery

• Cones
  • Only function with higher light levels: “photopic” vision
  • Densely packed at center of eye: fovea
  • Different types of cones → color vision
Electromagnetic Spectrum

- Visible light frequencies range between ...
  - Red = $4.3 \times 10^{14}$ hertz (700nm)
  - Violet = $7.5 \times 10^{14}$ hertz (400nm)

Figures 15.1 from H&B
Visible Light

- The human eye can “see” light in the frequency range 400nm – 700nm
Visible Light

- The human eye can “see” light in the frequency range 400nm – 700nm

**This does not mean that we can see the difference between the different spectral distributions.**

White Light

Figure 15.3 from H&B
Visible Light

• Color may be characterized by …
  • Hue = dominant frequency (highest peak)
  • Saturation = excitation purity (ratio of highest to rest)
  • Lightness = luminance (area under curve)
Spectral-response functions of each of the three types of cones. This motivates encoding color as a combination of red, green, and blue (RGB).
Tristimulus Color

• Any distribution of light can be summarized by its effect on 3 types of cones
• Therefore, human perception of color is a 3-dimensional space
• Metamerism: different spectra, same response
• Color blindness: fewer than 3 types of cones
  • Most commonly L cone = M cone
Color Models

- RGB
- XYZ
- CMYK
- HSV
- etc...

Different ways of parameterizing 3D space.

RGB most common and used in this class:
R=645.16nm, G=526.32nm, 
B=444.44nm
RGB Color Model

Colors are additive

Plate II.3 from FvDFH
RGB Color Cube

Figures 15.11 & 15.12 from H&B
CMY(K) Color Model

Colors are subtractive

<table>
<thead>
<tr>
<th>C</th>
<th>M</th>
<th>Y</th>
<th>Color</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>White</td>
</tr>
<tr>
<td>1.0</td>
<td>0.0</td>
<td>0.0</td>
<td>Cyan</td>
</tr>
<tr>
<td>0.0</td>
<td>1.0</td>
<td>0.0</td>
<td>Magenta</td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>1.0</td>
<td>Yellow</td>
</tr>
<tr>
<td>1.0</td>
<td>1.0</td>
<td>0.0</td>
<td>Blue</td>
</tr>
<tr>
<td>1.0</td>
<td>0.0</td>
<td>1.0</td>
<td>Green</td>
</tr>
<tr>
<td>0.0</td>
<td>1.0</td>
<td>1.0</td>
<td>Red</td>
</tr>
<tr>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>Black</td>
</tr>
<tr>
<td>0.5</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td>1.0</td>
<td>0.5</td>
<td>0.5</td>
<td></td>
</tr>
<tr>
<td>1.0</td>
<td>0.5</td>
<td>0.0</td>
<td></td>
</tr>
</tbody>
</table>

Plate II.7 from FvDFH
HSV Color Model

<table>
<thead>
<tr>
<th>H</th>
<th>S</th>
<th>V</th>
<th>Color</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1.0</td>
<td>1.0</td>
<td>Red</td>
</tr>
<tr>
<td>120</td>
<td>1.0</td>
<td>1.0</td>
<td>Green</td>
</tr>
<tr>
<td>240</td>
<td>1.0</td>
<td>1.0</td>
<td>Blue</td>
</tr>
<tr>
<td>*</td>
<td>0.0</td>
<td>1.0</td>
<td>White</td>
</tr>
<tr>
<td>*</td>
<td>0.0</td>
<td>0.5</td>
<td>Gray</td>
</tr>
<tr>
<td>*</td>
<td>*</td>
<td>0.0</td>
<td>Black</td>
</tr>
<tr>
<td>60</td>
<td>1.0</td>
<td>1.0</td>
<td></td>
</tr>
<tr>
<td>270</td>
<td>0.5</td>
<td>1.0</td>
<td></td>
</tr>
<tr>
<td>270</td>
<td>0.0</td>
<td>0.7</td>
<td></td>
</tr>
</tbody>
</table>

Figure 15.16&15.17 from H&B
Outline

• Human Vision and Color
• Image Representation
• Reducing Color Quantization Artifacts
• Basic Image Processing
Image Representation

What is an image?
Image Representation

An image is a 2D rectilinear array of pixels:

A width x height array where each entry of the array stores a single pixel.
Image Representation

What is a pixel?

Continuous image

Digital image
Image Representation

A pixel is something that captures the notion of “intensity” and possibly “color”

- Luminance pixels
  - Grey-scale images (aka “Intensity images”)
    - 0 – 1.0 or 0 – 255
  - Red, Green, Blue pixels (RGB)
    - Color images
      - 0 – 1.0 or 0 – 255
# Image Resolution

- **Spatial resolution:** $width \times height$ pixels
- **Intensity/Color resolution:** $n$ bits per pixel
- **Temporal resolution:** $n$ Hz (fps)

<table>
<thead>
<tr>
<th></th>
<th>Width x Height</th>
<th>Bit Depth</th>
<th>Hz</th>
</tr>
</thead>
<tbody>
<tr>
<td>NTSC</td>
<td>640 x 480</td>
<td>8</td>
<td>30</td>
</tr>
<tr>
<td>Handheld</td>
<td>640 x 480</td>
<td>16</td>
<td>45</td>
</tr>
<tr>
<td>Monitor</td>
<td>1920 x 1200</td>
<td>24</td>
<td>75</td>
</tr>
<tr>
<td>CCDs</td>
<td>3000 x 2000</td>
<td>36</td>
<td>-</td>
</tr>
<tr>
<td>Laser Printer</td>
<td>6600 x 5100</td>
<td>1</td>
<td>-</td>
</tr>
</tbody>
</table>
Image Quantization Artifacts

• With only a small number of bits associated to each color channel of a pixel there is a limit to intensity resolutions of an image
  - A black and white image allocates a single bit to the luminance channel of a pixel.
    » The number of different colors that can be represented by a pixel is 2.
  - A 24 bit bitmap image allocates 8 bits to the red, green, and blue channels of a pixel.
    » The number of different colors that can be represented by a pixel is 16,000,000.
Outline

• Human Vision
• Image Representation
• Reducing Color Quantization Artifacts
  oHalftoning and Dithering
• Basic Image Processing
Quantization

Image with decreasing bits per pixel

- Note contouring!
Quantization

- When you have a small number of bits per pixel, you can coarsely represent an image by quantizing the color values:

\[ P(x, y) = Q(I(x, y)) = \text{floor}\left(\frac{I(x, y)}{256} \cdot 2^b\right) \]

\( b \) is the number of bits per pixel
Reducing Effects of Quantization

Trade spatial resolution for intensity resolution:

• Halftoning

• Dithering
  • Random dither
  • Ordered dither
  • Error diffusion dither
Classical Halftoning

- Varying-size dots represent intensities
- Area of dots inversely proportional to intensity

$I(x, y)$

$P(x, y)$
Classical Halftoning

Newspaper Image

From New York Times, 9/21/99
Digital Halftoning

- Use cluster of pixels to represent intensity
- Trades spatial resolution for intensity resolution
- Note that halftoning pattern matters
  - Want to avoid vertical, horizontal lines

\[
0 \leq I \leq 0.2 \quad 0.2 < I \leq 0.4 \quad 0.4 < I \leq 0.6 \quad 0.6 < I \leq 0.8 \quad 0.8 < I \leq 1.0
\]
Digital Halftoning

- Use cluster of pixels to represent intensity
- Trades spatial resolution for intensity resolution
- Note that halftoning pattern matters

Original (8 bits)  Quantized (1 bit)  Halftoned (1 bit)
Dithering

• Distribute errors among pixels
  • Exploit spatial integration in our eye
  • Display greater range of *perceptible* intensities
Random Dither

- Randomize quantization errors
- Errors appear as noise

\[ P(x,y) = Q(I(x,y) + \text{noise}(x,y)) \]
Random Dither

- Randomize quantization errors
- Errors appear as noise

$P(x,y)$

$P(x,y) = Q(I(x,y) + \text{noise}(x,y))$

If a pixel is black, then adding random noise to it, you are less likely to turn it into a white pixel than if the pixel were dark gray.
Random Dither

- Randomize quantization errors
- Errors appear as noise

If a pixel is black, then adding random noise to it, you are less likely to turn it into a white pixel than if the pixel were dark gray.

\[ P(x, y) = Q(I(x, y) + \text{noise}(x, y)) \]
Random Dither

- Randomize quantization errors
- Errors appear as noise

If a pixel is black, then adding random noise to it, you are less likely to turn it into a white pixel than if the pixel were dark gray.

How much noise should we add?

Enough so that we can effect rounding, but not so much that we overshoot:

\[ [-0.5,0.5] \]

\[ P(x,y) = Q(I(x,y) + \text{noise}(x,y)) \]
Random Dither

Original (8 bits)

Uniform Quantization (1 bit)

Random Dither (1 bit)
Ordered Dither

• Pseudo-random quantization errors
• Matrix stores pattern of thresholds

For Binary Displays

\[
i = x \mod n \\
j = y \mod n \\
\text{if } \frac{l(x,y)/255}{D(i,j)/(n^2+1)} > 1 \text{ then} \\
\quad P(x,y) = 1 \\
\text{else} \\
\quad P(x,y) = 0
\]

\[
D_2 = \begin{bmatrix}
1 & 3 \\
4 & 2
\end{bmatrix}
\]
Ordered Dither

- Pseudo-random quantization errors
- Matrix stores pattern of thresholds

For b bit displays

\[
i = x \mod n \\
j = y \mod n \\
c = (I(x,y)/255)*(2^b-1) \\
e = c - \text{floor}(c) \\
\text{if } (e > D(i,j) / (n^2+1) ) \\
P(x,y) = \text{ceil}(c) \\
\text{else} \\
P(x,y) = \text{floor}(c)
\]

\[
D_2 = \begin{bmatrix}
1 & 3 \\
4 & 2
\end{bmatrix}
\]
Ordered Dither

Original (8 bits)

Random Dither (1 bit)

Ordered Dither (1 bit)
Error Diffusion Dither

- Spread quantization error over neighbor pixels
  - Error dispersed to pixels right and below
- Below we see Floyd-Steinberg Method

\[
\alpha + \beta + \gamma + \delta = 1.0
\]

Figure 14.42 from H&B
Error Diffusion Dither

for (i = 0; i < height; i++)
    for (j = 0; j < width; j++)
        Dest\textsubscript{i,j} = quantize(Source\textsubscript{i,j})
        error = Source\textsubscript{i,j} − Dest\textsubscript{i,j}
        Source\textsubscript{i,j+1} = Source\textsubscript{i,j+1} + \alpha \times error
        Source\textsubscript{i+1,j-1} = Source\textsubscript{i+1,j-1} + \beta \times error
        Source\textsubscript{i+1,j} = Source\textsubscript{i+1,j} + \gamma \times error
        Source\textsubscript{i+1,j+1} = Source\textsubscript{i+1,j+1} + \delta \times error

\alpha = \frac{7}{16}
\beta = \frac{3}{16}
\gamma = \frac{5}{16}
\delta = \frac{1}{16}

Floyd–Steinberg Dither
Error Diffusion Dither

- **Original** (8 bits)
- **Random Dither** (1 bit)
- **Ordered Dither** (1 bit)
- **Floyd–Steinberg Dither** (1 bit)
Outline

• Human Vision
• Image Representation
• Reducing Color Quantization Artifacts

• Basic Image Processing
  o Single Pixel Operations
  o Multi-Pixel Operations
Computing Grayscale

• The human retina perceives red, green, and blue as having different levels of brightness.

• To compute the luminance (perceived brightness) of a pixel, we need to take the weighted average of the RGBs:

\[ L = 0.30*r + 0.59*g + 0.11*b \]

Original

Grayscale

Figure 13.18 from FvDFH
Adjusting Brightness

• Simply scale pixel components
  o Must clamp to range (e.g., 0 to 255)
Adjusting Contrast

- Compute mean luminance \( L \) for all pixels
  \[
  \mathbf{L} = 0.30*r + 0.59*g + 0.11*b
  \]

- Scale deviation from \( L \) for each pixel component
  Must clamp to range (e.g., 0 to 255)
Adjusting Saturation

• Compute luminance $L_p$ for each pixel
  $$\mathbf{oL} = 0.30^*r + 0.59^*g + 0.11^*b$$

• Scale deviation from $L_p$ for each pixel component
  Must clamp to range (e.g., 0 to 255)
Image Processing by Interpolation

• Nice discussion of these operations: http://www.graficaobscura.com/interp/index.html
Image Processing by Interpolation

\[ \text{out} = (1 - \alpha) \cdot \text{in0} + \alpha \cdot \text{in1} \]
Image Processing by Interpolation

\[ \text{out} = (1-\alpha)\text{in0} + \alpha\text{in1} \]
Image Processing by Interpolation

\[ \text{out} = (1 - \alpha) \cdot \text{in0} + \alpha \cdot \text{in1} \]