Problem Session #2

EE368/CS232
Digital Image Processing
1. Color Reproduction

• *XYZ.mat*: contains the CIE 1931 XYZ spectral matching functions
1. Color Reproduction

• `XYZ.mat`: contains the CIE 1931 XYZ spectral matching functions
• `face.mat`: contains a hyperspectral image of a human face, where the spectral reflectance $\rho$ is measured at each pixel (reflectance $\rho$ is a value between 0 and 1)
1. Color Reproduction

• **XYZ.mat**: contains the CIE 1931 XYZ spectral matching functions
• **face.mat**: contains a hyperspectral image of a human face, where the spectral reflectance $\rho$ is measured at each pixel (reflectance $\rho$ is a value between 0 and 1)
• **light.mat**: contains the spectrum for an illuminant, scaled to result in a peak luminance of 100 cd/m$^2$ (for reflectance $\rho = 1$ across the spectrum) and represented in units of Watts/sr/nm/m$^2$
1. Color Reproduction

• **XYZ.mat**: contains the CIE 1931 XYZ spectral matching functions
• **face.mat**: contains a hyperspectral image of a human face, where the spectral reflectance $\rho$ is measured at each pixel (reflectance $\rho$ is a value between 0 and 1)
• **light.mat**: contains the spectrum for an illuminant, scaled to result in a peak luminance of 100 cd/m$^2$ (for reflectance $\rho = 1$ across the spectrum) and represented in units of Watts/sr/nm/m$^2$
• **display.mat**: contains the spectra for 3 primaries in an LCD display, represented in units of Watts/sr/nm/m$^2$
Part A:
Find the linear mapping from the spectral reflectance of the face to the weights one would apply to the three display primaries such that a human observer sees the same colors on the display as viewing the face directly under the illuminant.

**Illuminant × Face**

\[
X = \int \bar{x}(\lambda) I(\lambda) \rho(\lambda) d\lambda \quad Y = \int \bar{y}(\lambda) I(\lambda) \rho(\lambda) d\lambda \quad Z = \int \bar{z}(\lambda) I(\lambda) \rho(\lambda) d\lambda
\]

\[
X = \sum_{n=1}^{N_\lambda} \bar{x}[n] I[n] \rho[n] \quad Y = \sum_{n=1}^{N_\lambda} \bar{y}[n] I[n] \rho[n] \quad Z = \sum_{n=1}^{N_\lambda} \bar{z}[n] I[n] \rho[n]
\]

\[
\begin{bmatrix}
X \\
Y \\
Z
\end{bmatrix} =
\begin{bmatrix}
\bar{x}[1] I[1] & \cdots & \bar{x}[N_\lambda] I[N_\lambda] \\
\bar{y}[1] I[1] & \cdots & \bar{y}[N_\lambda] I[N_\lambda] \\
\bar{z}[1] I[1] & \cdots & \bar{z}[N_\lambda] I[N_\lambda]
\end{bmatrix}
\begin{bmatrix}
\rho[1] \\
\vdots \\
\rho[N_\lambda]
\end{bmatrix}
\]
Display Primaries

Spectrum of light emitted by a pixel of the display:

\[
X = \sum_{n=1}^{N_\lambda} x[n] \sum_{i=1}^{3} w_i P_i[n] \\
Y = \sum_{n=1}^{N_\lambda} y[n] \sum_{i=1}^{3} w_i P_i[n] \\
Z = \sum_{n=1}^{N_\lambda} z[n] \sum_{i=1}^{3} w_i P_i[n]
\]

\[
\begin{bmatrix}
X \\
Y \\
Z
\end{bmatrix} =
\begin{bmatrix}
\sum_{n=1}^{N_\lambda} x[n] \sum_{i=1}^{3} w_i P_i[n] \\
\sum_{n=1}^{N_\lambda} y[n] \sum_{i=1}^{3} w_i P_i[n] \\
\sum_{n=1}^{N_\lambda} z[n] \sum_{i=1}^{3} w_i P_i[n]
\end{bmatrix} =
\begin{bmatrix}
\sum_{i=1}^{3} w_i \sum_{n=1}^{N_\lambda} x[n] P_i[n] \\
\sum_{i=1}^{3} w_i \sum_{n=1}^{N_\lambda} y[n] P_i[n] \\
\sum_{i=1}^{3} w_i \sum_{n=1}^{N_\lambda} z[n] P_i[n]
\end{bmatrix} = A \begin{bmatrix}
w_1 \\
w_2 \\
w_3
\end{bmatrix}
\]

\[\rightarrow\text{ Find the matrix A}\]
Match Display Primaries to Illuminant \( \times \) Face

\[
\begin{bmatrix}
w_1 \\
w_2 \\
w_3 \\
\end{bmatrix} = A^{-1} \begin{bmatrix}
X \\
Y \\
Z \\
\end{bmatrix} = A^{-1} \begin{bmatrix}
\end{bmatrix} \begin{bmatrix}
\rho[1] \\
\vdots \\
\rho[N_\lambda] \\
\end{bmatrix} \\
\begin{array}{c}
T
\end{array}
\]
Part B:
Generate and display an RGB image, assuming a display $\gamma$ of 2.2.

\[
\begin{bmatrix}
w_1 \\
w_2 \\
w_3 \\
\end{bmatrix}
= A^{-1}
\begin{bmatrix}
X \\
Y \\
Z \\
\end{bmatrix}
= A^{-1}
\begin{bmatrix}
\bar{x}[1]I[1] & \cdots & \bar{x}[N_\lambda]I[N_\lambda] \\
\bar{y}[1]I[1] & \cdots & \bar{y}[N_\lambda]I[N_\lambda] \\
\bar{z}[1]I[1] & \cdots & \bar{z}[N_\lambda]I[N_\lambda] \\
\end{bmatrix}
\begin{bmatrix}
\rho[1] \\
\vdots \\
\rho[N_\lambda] \\
\end{bmatrix}
\]
Part C:
Plot the product of the spectral reflectance and the spectrum of the illuminant at pixel position \([x,y] = [47,149]\) in the image. Then, plot the corresponding spectrum emitted by the display. Compare the two spectra. Are the spectra metamers or isomers?


2. Color Gamut and Saturation

In this problem, we study the effects of saturation on the display of different colors. From the homework webpage, please download the following MATLAB files:

- **Chroma_Solid.fig**: contains a figure showing the boundaries of several cross-sections of the color gamut in XYZ space
- **Chroma_Map.mat**: contains a binary image marking the interior of the CIE chromaticity diagram in xy space
Part A:
Suppose colors from the color gamut are represented in the CIE RGB color space. In practical applications, the weights for the R, G, and B primaries must be (i) nonnegative and (ii) less than a maximum value. Suppose the individual RGB values must lie in the range [0,1] and values outside this range are clipped to 0 or 1. Perform a linear transformation of the unit cube \( \{0 \leq R, G, B \leq 1\} \) from the CIE RGB color space into the CIE XYZ color space. Display the edges of the resulting parallelepiped in XYZ space on the same axes as the cross-sections of the color gamut and submit the figure. In relation to the parallelepiped, which colors in the color gamut would be clipped/saturated in the RGB color space?
\[
\begin{bmatrix}
X \\
Y \\
Z
\end{bmatrix} =
\begin{bmatrix}
0.490 & 0.310 & 0.200 \\
0.177 & 0.813 & 0.011 \\
0.000 & 0.010 & 0.990
\end{bmatrix}
\begin{bmatrix}
R \\
G \\
B
\end{bmatrix}
\]
Part B:
Render and submit xy-chromaticity diagrams corresponding to the following cross-sections of the color gamut: \( X + Y + Z = S \) for \( S = 0.5, 1.0, 2.0 \). Assume a display \( \gamma \) of 2.2. For each chromaticity diagram, display and submit a binary image where regions of R, G, or B clipping/saturation are marked by white pixels and all other regions are marked by black pixels. Interpret the shape of the clipped/saturated regions in relation to the parallelepiped generated in (a).

\[
x = \frac{X}{X + Y + Z} = \frac{X}{S}
\]

\[
X = Sx
\]

\[
Y = Sy
\]

\[
Z = S - X - Y
\]

\[
\begin{bmatrix}
R \\
G \\
B
\end{bmatrix} =
\begin{bmatrix}
0.490 & 0.310 & 0.200 \\
0.177 & 0.813 & 0.011 \\
0.000 & 0.010 & 0.990
\end{bmatrix}^{-1}
\begin{bmatrix}
X \\
Y \\
Z
\end{bmatrix}
\]
Chromaticity Diagram: $X + Y + Z = 0.50$

Saturated Regions
Chromaticity Diagram: $X + Y + Z = 1.00$

Saturated Regions
Chromaticity Diagram: $X + Y + Z = 2.00$

Saturated Regions
3. Binarization of Scanned Book Pages

Services like the Guttenberg Project and Internet Archive have digitized large collections of books. Typically, the book pages are scanned, and then the scanned images are binarized and processed through an optical character recognition (OCR) engine. For modern books, the book spine can be removed to separate the book pages for more efficient scanning. For vintage books, however, destroying the original binding of the book is often undesirable. If a book is scanned with its spine left intact, the curvature of the pages causes uneven illumination in the resulting images. This effect can be observed in the images hw2_book_page_1.jpg and hw2_book_page_2.jpg.
Part A:
For each image, generate a binary image by performing global thresholding. Use a threshold chosen by Otsu’s method (function: graythresh). Submit the binary image. Also, submit a histogram of the original image’s gray values and clearly mark the threshold of Otsu’s method on this histogram. Comment on the quality of the binary image.
Part A:
For each image, generate a binary image by performing global thresholding. Use a threshold chosen by Otsu’s method (function: graythresh). Submit the binary image. Also, submit a histogram of the original image’s gray values and clearly mark the threshold of Otsu’s method on this histogram. Comment on the quality of the binary image.
Part B:
Now, perform locally adaptive thresholding. Treat uniform and non-uniform regions differently based on the local variance. Submit the binary image. Comment on the quality of the binary image compared to the result from part (a).
Part B:
Now, perform locally adaptive thresholding. Treat uniform and non-uniform regions differently based on the local variance. Submit the binary image. Comment on the quality of the binary image compared to the result from part (a).
4. Traffic Cone Detection

Traffic cones are set up by law enforcement and construction workers to block a portion of the road. Unlike more permanent road signs whose locations are indexed in digital maps and hence available to navigation systems on automobiles, traffic cones are missing from the digital maps and must be dynamically detected. Please download the following images from the handouts webpage:

- hw2_cone_training_{1,2,3,4,5}.jpg: RGB images for training
- hw2_cone_training_map_{1,2,3,4,5}.png: binary masks of cones in the training images
- hw2_cone_testing_{1,2}.jpg: RGB images for testing
Part A:
Using the 5 training images, generate and submit a 3-d scatterplot in RGB space of a small randomly chosen subset of the training RGB samples, where the cone and non-cone RGB samples are plotted with different markers (function: scatter3). Comment on how effectively cone and non-cone samples are separated in RGB space.
Part B:
Using the 5 training images, train a multidimensional MAP detector in RGB space for distinguishing between traffic cones and other parts of the image. Assuming RGB values lie in the range [0,255], use 16x16x16 uniformly spaced bins for the RGB values. Report the fraction of bins in the 16x16x16 grid which are labeled as belonging to the cone class.
Part C:
Use the multidimensional MAP detector to classify pixels in each testing image as “cone” or “non-cone”. After MAP classification, you can also perform any post-processing (e.g., small region removal) to make the final result more accurate. For each testing image, submit a binary image where the “cone” pixels are shown as white and the “non-cone” pixels are shown as black.
Bonus #1: Android Mobile Image Processing

Mean (R,G,B): 121.2, 112.0, 110.9
Std Dev (R,G,B): 119.6, 120.3, 119.8

Mean (R,G,B): 113.0, 84.15, 77.19
Std Dev (R,G,B): 53.16, 65.96, 70.87
Bonus #2: Color Balancing

In this problem, we explore the effectiveness of some popular color balancing algorithms. From the homework webpage, please download macbeth.jpg and leaf.jpg, which contain values in the γ-predistorted domain.
Part A:
Convert the images to the linear RGB domain assuming a γ of 2.2. Then, perform color balancing on each image in the linear RGB domain using each of the following algorithms:

• Gray-world: Choose the scale factors $[k_r, k_g, k_b]$ so that $k_g = 1$.

$$k_L \sum_{x,y} L[x,y] = k_M \sum_{x,y} M[x,y] = k_S \sum_{x,y} S[x,y]$$

• Scale-by-max: Choose the scale factors $[k_r, k_g, k_b]$ so that after scaling, the max value in each color component in the image is 1.

$$k_L \max_{x,y} L[x,y] = k_M \max_{x,y} M[x,y] = k_S \max_{x,y} S[x,y]$$

• Shades-of-gray. Choose the scale factors $[k_r, k_g, k_b]$ so that $k_g = 1$. Use $p = 6$.

$$k_L \left( \sum_{x,y} L^p[x,y] \right)^{\frac{1}{p}} = k_M \left( \sum_{x,y} M^p[x,y] \right)^{\frac{1}{p}} = k_S \left( \sum_{x,y} S^p[x,y] \right)^{\frac{1}{p}}$$
Part B:
Perform color balancing on each image using the shades-of-gray algorithm for the different values $p = 1, 2, 6, 10, 100$. Note that the $p = 1$ generates a result equivalent to that of gray-world, while $p = 100$ generates a result very similar to that of scale-by-max. In each case, plot and submit the histograms for each color channel after color balancing. Also plot and submit the histogram before color balancing. How are the histograms transformed through the color balancing operation for the different $p$ values?