

Assignment: Light, Cameras, and Image Formation

Erik G. Learned-Miller

February 11, 2014

Problem 1. Linearity. (10 points) Alice has a chandelier with 5 light bulbs sockets. Currently, she has 5 100-watt incandescent bulbs in the sockets. Each incandescent bulb is characterized by a spectrum $S_{\text{INC}}(\lambda)$, for integer λ from 380 to 779, which gives the fraction of the total power output for each interval with wavelengths between λ and $\lambda + 1$ nanometers. For example, if $S_{\text{INC}}(523) = 0.002$, then it means that the incandescent bulb outputs $0.002 \times 100 = .2$ watts in the range of 523 – 524 nanometer wavelengths.

Eventually, she wants to replace the incandescent bulbs with fancy new light emitting diode (LED) bulbs (also 100-watt bulbs) that have a different spectrum $S_{\text{LED}}(\lambda)$, but right now, she only has 3 of these. She replaces 3 of the 5 incandescent bulbs with the LED bulbs. Give a formula for the chandelier's new spectrum S_{TOTAL} in terms of the components of S_{INC} and S_{LED} . The new spectrum should give the fraction of the total power at each 1 nanometer portion of the visible spectrum.

Problem 2. Inverse Square Law (10 points). Melon Husk, the CEO of Space OX, has just launched a new satellite into orbit. The satellite has a device which measures the amount of electromagnetic radiation (light), in wavelengths between 100nm and 1000nm, coming from a particular small region of the sky. It restricts the light detection to this small region by using a *collimator* to block light coming from other sources (such as the sun, the moon, and other stars). He aims the satellite's detector at a small region of the sky containing the binary star system Alpha Centauri. On the satellite's sensor, which is 1cm by 1cm, he estimates the light power is about $1.5 * 10^{-12}$ watts.

Alpha Centauri is about 4.37 light years from earth. How much power is it producing in the 100-1000 nanometer range? (Warning: the numbers in this problem may not correspond to the star's true distance or power, so if you look them up on the internet, you will be wrong.) Draw a picture to illustrate your solution.

Problem 3. Tristimulus theory (30 points). Let F_1 , F_2 , and F_3 below represent the power spectra of 3 different colored flashlights, where each of the 10 values is the fraction of power produced in a 40nm range from 380nm to 780nm. (Note: some of the power will be in the non-visible spectrum).

$$F_1 = [0.0, 0.01, 0.01, 0.02, 0.01, 0.02, 0.17, 0.29, 0.35, 0.02]$$

$$F_2 = [0.0, 0.01, 0.02, 0.11, 0.20, 0.30, 0.19, 0.10, 0.01, 0.0]$$

$$F_3 = [0.01, 0.12, 0.25, 0.27, 0.13, 0.02, 0.01, 0.01, 0.0, 0.0]$$

For example, flashlight F_1 produces 2 percent of its power in the range 740-780nm.

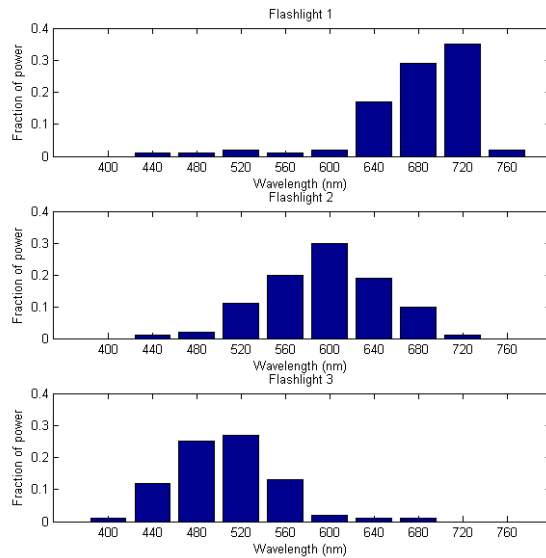


Figure 1: **Three Flashlight Spectra.** This image shows the three spectra of F_1 , F_2 , and F_3 .

Let $S_r(\lambda)$, $S_g(\lambda)$, and $S_b(\lambda)$ represent the *relative* absorption spectra of the cone cells in your eye. This means that, of the power absorbed by a given type of cone cell, the fraction absorbed in a given range is given by these numbers.

$$S_r(\lambda) = [0.16, 0.26, 0.28, 0.15, 0.10, 0.03, 0.02, 0.0, 0.0, 0.0]$$

$$S_g(\lambda) = [0.0, 0.0, 0.04, 0.23, 0.34, 0.23, 0.15, 0.01, 0.0, 0.0]$$

$$S_b(\lambda) = [0.0, 0.0, 0.0, 0.0, 0.01, 0.04, 0.08, 0.23, 0.35, 0.29]$$

When a flashlight is 5 meters from a white screen, assume that it stimulates a cone cell response (relative to the maximum possible response from that cone) given by

$$R_c^f = \sum_{i=1}^{10} F_f(i) \times S_c(i).$$

Here F_f is flashlight f , S_c is the absorption spectrum for cones of type c , and R_c^f is the response for cone cell type c and flashlight f . For example, from 5 meters away, the third flashlight F_3 generate a response from the green cone cells of

$$\begin{aligned} & \sum_{i=1}^{10} F_3(i) \times S_g(i) \\ = & (0.01)(0.0) + (0.12)(0.0) + (0.25)(0.04) + (0.27)(0.23) + (0.13)(0.34) + \\ & (0.02)(0.23) + (0.01)(0.15) + (0.01)(0.01) + (0.0)(0.0) + (0.0)(0.0) \\ = & 0.1225. \end{aligned}$$

Consider the response of all 3 cone cell types at once to a given flashlight f at 5 meters from the white screen. We will call this variable R^f , and it will be a vector of 3 values:

$$R^f = \begin{bmatrix} R_r^f \\ R_g^f \\ R_b^f \end{bmatrix}.$$

Of course, the “color” that you see for a particular flashlight will be related to the cone cell responses R^f produced by that flashlight. Due to the linearity of light and the approximate linearity of cone cell responses, you can model the responses of the cone cells to two flashlights, say 1 and 3, as

$$R^1 + R^3 = \begin{bmatrix} R_r^1 + R_r^3 \\ R_g^1 + R_g^3 \\ R_b^1 + R_b^3 \end{bmatrix}.$$

Problem 3.A. Let the matrix R be defined to be

$$R = \begin{bmatrix} R_r^1 & R_r^2 & R_r^3 \\ R_g^1 & R_g^2 & R_g^3 \\ R_b^1 & R_b^2 & R_b^3 \end{bmatrix}.$$

Write a Matlab function to compute the matrix R given F_1, F_2, F_3, S_r, S_g , and S_b . Give the value of the matrix R here:

If we had a way to make each flashlight brighter or dimmer, then we could create novel color combinations. In particular, if we let b_1 be the brightness multiplier for flashlight 1, b_2 the brightness multiplier for flashlight 2, and b_3 the brightness multiplier for flashlight 3, then we can create arbitrary linear combinations of the 3 flashlights as

$$b_1 R^1 + b_2 R^2 + b_3 R^3 = \begin{bmatrix} b_1 R_r^1 + b_2 R_r^2 + b_3 R_r^3 \\ b_1 R_g^1 + b_2 R_g^2 + b_3 R_g^3 \\ b_1 R_b^1 + b_2 R_b^2 + b_3 R_b^3 \end{bmatrix}.$$

In particular, a brightness multiplier value of 1 would result in a brightness equal to the brightness of the flashlight at the default distance of 5 meters. By the inverse square law of brightness, the brightness would be 0.25 if the flashlight were twice as far from the screen (10 meters away).

Problem 3.B. To achieve various brightness multiplier values, we will move a flashlight closer to or further from the screen than the 5 meter starting distance. Give a formula for the distance to achieve a certain brightness multiplier b using the inverse square law:

Now suppose you want to create a combination of lights that results in a particular perceived color. For example, the color turquoise is perceived when the RGB cone cells have a response of

$$C_{\text{turquoise}} = \begin{bmatrix} 0.2896 \\ 0.8862 \\ 0.7471 \end{bmatrix}.$$

We need to solve the following set of equations

$$\begin{aligned} b_1 R_r^1 + b_2 R_r^2 + b_3 R_r^3 &= C_r \\ b_1 R_g^1 + b_2 R_g^2 + b_3 R_g^3 &= C_g \\ b_1 R_b^1 + b_2 R_b^2 + b_3 R_b^3 &= C_b \end{aligned}$$

This can be written in matrix form as

$$Rb = C,$$

where R is the matrix defined previously, b is a vector of 3 multiplier values, and C is the 3 color components for the desired color. Fortunately, this is extremely easy to solve in Matlab! By multiplying both sides by the matrix inverse of R , we obtain

$$R^{-1}Rb = R^{-1}C,$$

and then, simplifying,

$$b = R^{-1}C.$$

In Matlab, the inverse of a matrix can be obtained by simply raising the matrix to the power -1.

Problem 3.C. Write a Matlab function, which, given the matrix R and a set of desired color responses in the form of a vector C calculates the proper brightness multiplier weights b and returns them as a single vector of 3 values.

Problem 3.D. Finally, use your results from the previous parts to figure out how far away each flashlight should be from the screen to obtain these color values. As an intermediate step, you will want to compute the multiplier weights for each flashlight that you want to obtain:

Turquoise:

$$C_{\text{turquoise}} = \begin{bmatrix} 0.2896 \\ 0.8862 \\ 0.7471 \end{bmatrix}.$$

b_1 : _____ b_2 : _____ b_3 : _____

Goldenrod (a slightly brownish yellow):

$$C_{\text{goldenrod}} = \begin{bmatrix} 0.8567 \\ 0.6874 \\ 0.1408 \end{bmatrix}.$$

b_1 : _____ b_2 : _____ b_3 : _____

For each color, how far from the wall should you place the 3 flashlights?

Turquoise:

Flashlight 1 dist: _____ Flashlight 2 dist: _____ Flashlight 3 dist: _____

Goldenrod:

Flashlight 1 dist: _____ Flashlight 2 dist: _____ Flashlight 3 dist: _____

Problem 4. Simulated Bayer filters (30 points). Bayer filters or patterns are used in many camera designs to allow the capture of different parts of the visible light spectrum across the image and hence gather color information while still only using a single sensor array (such as a CCD array). Bayer patterns consist of a pattern of red, green, and blue filters arranged in a grid as shown in Figure 2. For more information about Bayer filters, check the Wikipedia article. For this question, you will be using the M&M's image to test your code.

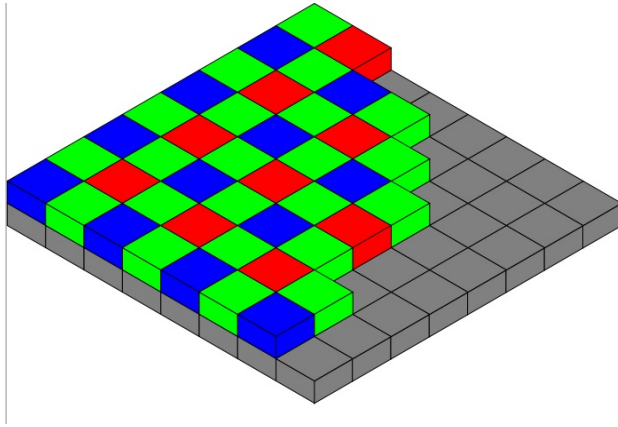


Figure 2: **Bayer pattern.** (Taken from Wikipedia page.) This shows the pattern of filters typically used on CCD cameras to measure different aspects of the incoming spectrum at different pixels.

Problem 4.A. Write a Matlab function called *simulateBayerSampling* that takes an RGB image as input and outputs the gray-level Bayer filtered image. The gray-level image should be the selected elements of the RGB image, placed in a single layer (gray) image. Use the grid shown in Figure 2 as an example, with the blue pixel in the left corner as (1,1), and the first row of the image proceeding across the alternating blue and green pixels.

Problem 4.B. Now write a Matlab function called *rebuildRGBFromBayer* that takes a gray-level Bayer filter image as input and outputs an RGB image. There are a variety of methods to estimate the RGB image from Bayer filter image. Briefly describe a couple simple algorithms (you can research these using the web, but do not copy the code from the web), then choose one to implement in this function.

Problem 4.C. The estimate produced in *rebuildRGBFromBayer* will have errors when compared to the original, as some of the information in the original image was lost when producing the Bayer filtered image. Write a Matlab function called *computeErrorImage* that takes the original image and the estimated RGB image as input and produces an image representing the error between the original and estimate images at each pixel location. The formula you use for the error at each pixel should be

$$error = \sqrt{(r_1 - r_2)^2 + (g_1 - g_2)^2 + (b_1 - b_2)^2},$$

where the r_1 is the red channel value of the original image and r_2 is the red channel value of the reconstructed image, and so on. Display the “error image” by creating an image whose value at each pixel is the error at that pixel. Briefly comment on where the errors are largest and why that may be.

What to turn in

This assignment asks for various written answers, code, and images. Please turn these in however you see fit, but please make it as clear as possible to the grader. I suggest that you turn in your code in separate files from the remainder of your answers. -Erik