Laboration 1

Bildreproduktion och Bildkvalitet
(TNM097)

Device characterization and modeling
Part 1 – Input devices

Preparations

As preparation for this lab, focusing on modeling and characterization for input devices, **it is necessary and very important that you read the theory part of this document (written in black)** prior to the scheduled time for the lab. It will also be very useful to repeat Lab 4 from TNM059 (*Grafisk teknik*). You will partly work with the same data, and you will have good use of the functions and assignments. This lab from TNM059 and the functions are all in the folder ***Course_documents/Labs/preparation_TNM059*** in Lisam. Besides that, it is very useful to read Chapter 6 in Digital Halftoning and Color Reproduction, which is put in the folder ***Course_documents/Literature/0_preparation***. For a deeper understanding of Device Characterization, read Chapter 5 in Digital Color Imaging Handbook under ***Course_documents/Literature/2_Lecture2_Daniel***. Colorimetric computations, spectral computations with surfaces and illuminants, conversions between CIEXYZ and CIELAB, as well as the correct use of normalization factors and white point conversions, are all necessary prerequisites for this lab.

Introduction:

The joint Labs 1 & 2 will focus on modeling and characterization of a complete color reproduction workflow, including both input and output devices. You will investigate both model-based and empirical characterization approaches. The workflow you will work with includes typical input devices with 3 channels (RGB-cameras), and output devices, using additive color mixing (RGB) for reproducing colors. The input data are spectral signals from different light sources and a set of color samples (which you have previously used in TNM059 – Lab3). Much of the data you acquire in Lab 1, focusing on input devices, will be further used in Lab 2, as input to the output devices. Figure 1 shows an overview of the complete color reproduction in these two labs, where the upper part to (XYZ) is done in Lab 1 and the lower part after XYZ is done in Lab 2.

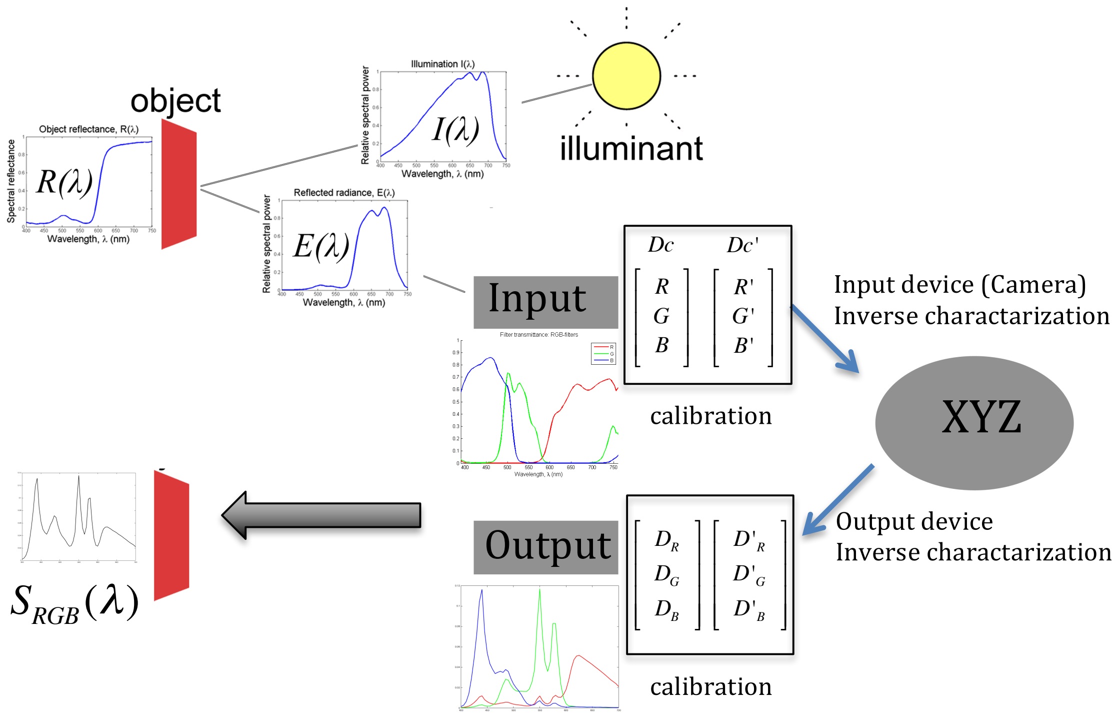

Figure 1. Overview of the complete color reproduction workflow used in Lab 1 & 2, including input- and output- RGB devices.

All spectral data used in these two labs are given at the visible wavelength interval [400 nm, 700 nm], with a step of 5 nm. Therefore, all spectral data includes 61 elements.

The data you need is available in the file Lab1.zip, including:

- The file *chips20.mat* contains spectral reflectance data for 20 different surfaces, representing **object** in Fig. 1 and 2. *chips20* is a *20 x 61* matrix, in which, each row contains the spectral reflectance data for each object. The data are given at wavelengths 400:5:700 nm, giving 61 elements for each row.
- The file *illum.mat* contains spectral data for eight different light sources, i.e. CIEA, CIEB, CIED65, Halogen 75W, Tungsten60W, plank50k, plank65k and plank90k. Each light source is defined by a *1 x 61* matrix, representing the light intensity value at wavelengths 400:5:700. In this file, there is also a vector *1 x 61* called *waverange*, representing the wavelengths 400:5:700.
- The file *xyz.mat* contains the color matching functions (CMFs), used for computing tristimulus values (XYZ) from spectral data. *xyz* is a *61 x 3* matrix, in which column 1, 2 and 3 represent the x, y and z color matching functions, respectively.
- The files *Ad.mat* and *Ad2.mat* contain spectral sensitivity functions for two input devices (RGB-cameras), mapping input spectral radiance to RGB. *Ad* and *Ad2* are both *61 x 3* matrices, in which column 1, 2 and 3 represent the R, G and B sensitivity functions of the camera.

Input device calibration and characterization

1. Spectral model of color image acquisition

Let's start by looking at the forward spectral model for color image acquisition, expressing the relationship between the spectral input and the resulting camera response. Remember that the general model that describes the response for each channel, *i*, of an image capture device with *M* color channels is given by:

$$D_i = \int_{\lambda \in V} E(\lambda) S_i(\lambda)\, d\lambda + n_i \qquad (1)$$

where $i = 1,\ldots,M$ ($M = 3$ for RGB devices), D_i is the output sensor response, $E(\lambda)$ is the input spectral radiance, $S_i(\lambda)$ is the spectral sensitivity of the *i*:th sensor, n_i is the measurement noise and V is the spectral sensitivity region of the device. For more detail, please see **Section 6.4.1** in Digital Halftoning and Color Reproduction (in the folder Course_documents/Labs/preparation_TNM059 in Lisam).

Figure 2 illustrates the spectral image acquisition model for an RGB-camera. The input to the camera, *E(λ)* is spectral radiance, given by the product of the light source *I(λ)* and the reflectance of the surface*, R(λ). S(λ)* shows the cameras spectral sensitivity functions for the channels R, G and B. The resulting camera output is 3 signals R, G and B, corresponding to D_i in Eq. 1.

Based on the data you have in this lab, *R(λ)* is represented by the rows in *chips20* (representing 20 different objects in 20 rows), *I(λ)* is represented by the variables in *illum.mat* (eight different variables or light sources), and *S(λ)*, i.e. camera characteristic, is represented by *Ad* and *Ad2* for two different cameras, respectively.

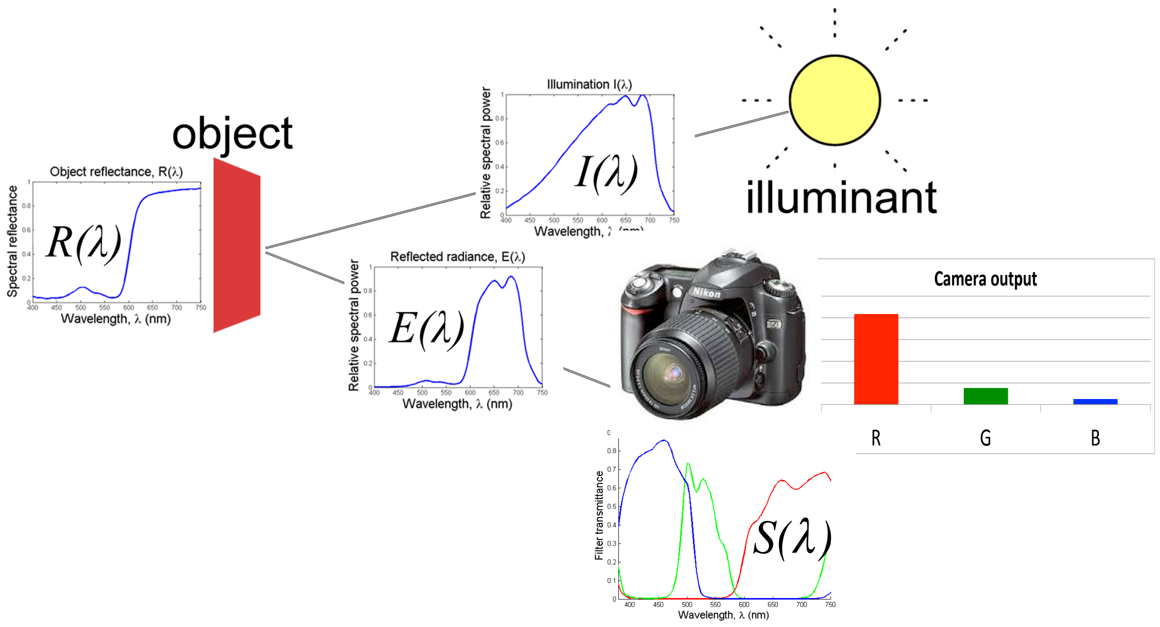

Figure 2. Color image acquisition for an input device (RGB camera).

Assume that, we want to find the camera response for the Red-sensor when capturing surface 1 ($R_1(\lambda)$) under illuminant 1 ($I_1(\lambda)$). Using Eq. 1, we will have:

$$R_{camera} = \int_{400}^{700} R_1(\lambda) I_1(\lambda) S_R(\lambda) d\lambda, \tag{2}$$

where, $S_R(\lambda)$ represents the camera characteristic for the Red-sensor of the camera. The G and B responses of the camera can be calculated by replacing $S_R(\lambda)$ by $S_G(\lambda)$ and $S_B(\lambda)$, respectively. As discussed before, the spectral signals are sampled, being represented by vectors, and therefore Eq. 2 can be rewritten as,

$$R_{camera} = \sum_{i=1}^{61} R_1(i) I_1(i) S_R(i), \tag{3}$$

where, R_1, I_1 and S_R are vectors representing object 1, illuminant 1 and the characteristic of the Red-sensor of the camera. The G and B responses can be calculated accordingly. Using Matlab notions, Eq. 3 can be rewritten as,

$$R_{camera} = sum(R_1.* I_1.* S_R), \tag{4}$$

where, *sum* returns the sum of the elements in a vector and *.** represents element-wise multiplication between vectors. The G and B camera responses are found by:

$$G_{camera} = sum(R_1.* I_1.* S_G), \tag{5}$$
$$B_{camera} = sum(R_1.* I_1.* S_B), \tag{6}$$

Notice now that, in the data you have, the reflectance spectrum of the first object (R_1) is found in row 1 in chips20. The illuminant (I_1) is one of the eight available light sources in *illum.mat* and the R, G and B sensor characteristic of a camera is found in column 1, 2 and 3 in *Ad* (or *Ad2*), respectively.

If we now assume $e = R_1.* I_1$, which is a *1 x 61* matrix, and the characteristic of the RGB-sensors of the camera to be represented by A_d, which is a *61 x 3* matrix, then Equations 4, 5 and 6 can be calculated at the same time by,

$$d = A_d^t e^t, \tag{7}$$

where A_d^t and e^t are the transpose of A_d and e being *3 x 61* and *61 x 1* matrices, respectively. The vector **d** is now a *3 x 1* matrix (vector), containing the R, G and B response of the camera. To simplify, the measurement noise term in Eq. 1 is ignored.

For example, if you want to calculate the camera response for object number 10, under illuminant CIEA, for the camera represented by Ad, in Matlab you simply write:

$$d = Ad' * (chips20(10,:).* CIEA)'. \tag{8}$$

1.1

In the files provided, you will find the matrices $\mathbf{A}_d$ and $\mathbf{A}_{d2}$ containing spectral sensitivity functions for two RGB-cameras. As discussed, the sensitivity functions, as well as the spectral data for illuminants and surfaces, are all in the interval 400:5:700 nm (giving L=61).

Start by comparing the spectral sensitivity functions for the two cameras by plotting $\mathbf{A}_d$ and $\mathbf{A}_{d2}$ against the wavelengths 400:5:700 nm. Based on the sensitivity functions, do you predict that the output from the two cameras will be the same for the same input? Note that these curves actually correspond to two real RGB-cameras.

………………………………………………………………………………………………………
………………………………………………………………………………………………………
………………………………………………………………………………………………………

1.2

Compute the resulting RGB-output for the first camera, for the 20 different color samples in the file chips20. Use the illuminant CIED65 (which corresponds to the standard illuminant D65, representing daylight) for the computations. Corresponding to Eq. 7, **d** is the output RGB-values, $\mathbf{A}_\mathbf{d}$ is the 61x3 matrix representing the camera sensitivity functions for the channels R, G and B, and **e** is the input spectral radiance, given by the product of the illuminant and the surface reflectance in chips20. The resulting 20 RGB-triplets represent the raw (i.e. un-calibrated) camera output for the 20 surfaces, captured under the illuminant D65. Save your data as RGB_raw_D65 (or similar) for later use. Note that the RGB data is in double format (64 bits), in the range [0,1], while for 8-bit devices it would be converted to integers in the interval [0,255]. You can use the provided file showRGB to display the RGB-values for the 20 color samples.

Now compute the corresponding RGB-output for the second camera (represented by $\mathbf{A}_{d2}$) using the same 20 color samples and D65 illuminant. What you have now is device dependent RGB-output for two different RGB-cameras, representing the same 20 color samples under the same illuminant. Compare the resulting RGB-values for the two cameras (you can also use showRGB). Considering identical input, what is your comment to the output RGB-values for the two (real) RGB-cameras? Can you relate the difference in the device-dependent RGB values to the spectral sensitivity curves for the two cameras?

………………………………………………………………………………………………………
………………………………………………………………………………………………………

………
………
………

So far, you have calculated the un-calibrated (raw) camera response, denoted by D_c in Fig. 1. The next step in the workflow is to calibrate the camera response.

2. Calibration of input devices

The spectral model you used for the input device defines the forward function, determining the device response, RGB, for a spectral input signal, *E(λ)* (product of illuminant and the surface reflectance). The results have (hopefully) illustrated the need to further process the raw RGB-signals using device calibration and characterization, to achieve a consistent color reproduction workflow. Input device calibration is performed prior to the characterization, to compensate for the device characteristics, and keeping it in a fixed, known state. Device characterization then derives the relationship between device-dependent, **d'**, and device-independent data, **c**, for the calibrated device (Fig. 3).

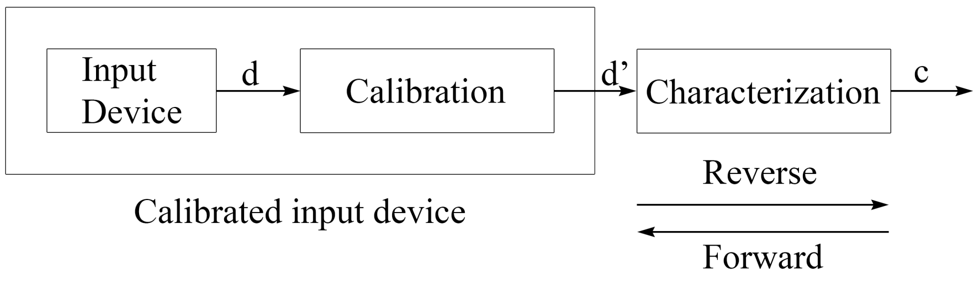

Figure 3. Calibration and characterization for an input device (e.g. an RGB-camera).

Calibration of input devices, such as RGB-cameras, generally involves determining the relationship between scene radiance and camera response, in order to linearize and gray-balance the camera. For this camera, it has previously been established that the response is linear to scene radiance, and it is not necessary to compute linearization curves. The calibration thus reduces to computing normalization factors to scale the response for the R, G and B channels, compensating for the filter characteristics. In Fig. 3, this corresponds to converting the raw RGB-signals **d** (calculated in assignment 1.2) to calibrated RGB-signals **d'**.

2.1

Start by computing normalization factors for the two cameras, to compensate for the sensor characteristics. For each of the R, G and B channels, find the factor that can be used for scaling the channel so that R'=G'=B'=1 for an ideal "white" input spectrum (i.e. *e*=ones(1,61)). This means that the normalization factors can be found by replacing *e* in Eq. 7 by a vector of ones. Can you relate the normalization factors to the spectral sensitivity curves for the two cameras?

………
………
………

2.2

Now use your normalization factors found in assignment 2.1 to compute calibrated R'G'B'-data for the 20 color samples, by scaling your RGB_raw data. The result is calibrated device-dependent data, corresponding to **d'** in Fig. 3. For Camera 1, make sure to save the data as RGB_cal_D65 (or similar), for later use. Now compare the resulting RGB-signals after calibration, for the two cameras, in the same way as in 1.2. Comments?

....
....
....
....

2.3

So far, all experiments have been made using the same light source, standard illuminant D65 (representing daylight). In practice, image acquisition is not always made under a controlled and constant light source. In the file *illum.mat*, you will also find data for CIEA, which is a standard illuminant representing indoor light (tungsten filament light). Compare the illuminants D65 and A by plotting their spectral power distributions against the wavelengths 400:5:700 nm. What can you say about the difference between typical outdoor and indoor light?

....
....
....
....

2.4

Now compute the resulting RGB-output for Camera 1 under Illuminant A (the same way as you did in 1.2 for D65). Then use your normalization factors to compute the calibrated camera response. Compare the calibrated RGB-signals for Illuminant A to the corresponding ones for D65 (RGB_cal_D65). This corresponds to images of the same 20 surfaces, captured by the same calibrated camera, using two different light sources. Comments?

....
....
....

The reason for the results is that the camera calibration is based on the camera characteristics alone. The normalization factors are computed to generate a neutral RGB response for a neutral input spectrum, E(λ). Since E(λ) is the product of the illuminant I(λ) and the object R(λ) (see Fig. 2), the characteristics of the illuminant will inherently affect the resulting color balance in the resulting RGB-image. This is suitable for absolute camera calibration, compensating for device characteristics and keeping it in a fixed state, independent of the light source. However, in many practical applications, we actually do want to compensate for the illumination, using white point normalization. The aim is then to mimic human color vision, which automatically compensates for the scene illumination. If you instead would compute the normalization factors based on a specific illumination and a reference white (i.e. using R=ones(1,61) in Eq. 8 instead of *chips20(10,:)*), the system would be calibrated for that specific light source, which would be a type of white point normalization.

2.5

Now compute new normalization factors based on the specific light sources D65 and A for Camera 1 (in the same way as in 2.1, but instead using R=ones(1,61)). Use the new factors to scale the RGB_raw data, for D65 and A, respectively, and compare the results. This corresponds to the same set-up as in 2.4 (same 20 surfaces under 2 light sources), but including the light source in the calibration. Comments?

....
....
....

3 Characterization of input devices

Now you have used the forward spectral model to simulate the camera sensor, generating RGB-signals from a spectral input, and applied calibration routines to modify the RGB-signals. However, the calibrated RGB-signals are still device-dependent. The next step in a color management workflow is device characterization, deriving the relationship between camera RGB and the device-independent color space CIEXYZ. For an input device, this refers to the inverse function, which compensates for the device characteristics and determines colorimetric data (XYZ) from the recorded device signals (RGB). In Fig. 3, this corresponds to deriving the relationship between the calibrated RGB-values, **d'**, and colorimetric values, **c.**

Figure 4 shows the workflow for evaluating the accuracy of device characterization. The input is the spectral signals $E(\lambda)$, which after image acquisition generate the device response, **d**, modified by camera calibration to **d'** (lower row in Fig. 4). The characterization function then estimates XYZ-values, from the calibrated RGB-values **d'**. To evaluate the result, we need to compare the estimated XYZ-values to reference values. The reference values are simply computed by applying the color matching functions (CMFs) to the spectral input $E(\lambda)$, which gives us XYZ-values, representing the color appearance for a "standard observer" (upper row in Fig. 4). However, since XYZ is not a perceptually uniform color space, it is not optimal for describing color differences. Instead, the XYZ-values are converted into CIELAB color space, using standard transformations. In CIELAB, the perceptually relevant ΔE_{ab} color difference can be computed, as the Euclidian distance between the CIELAB values for the references and the estimations (Fig. 4).

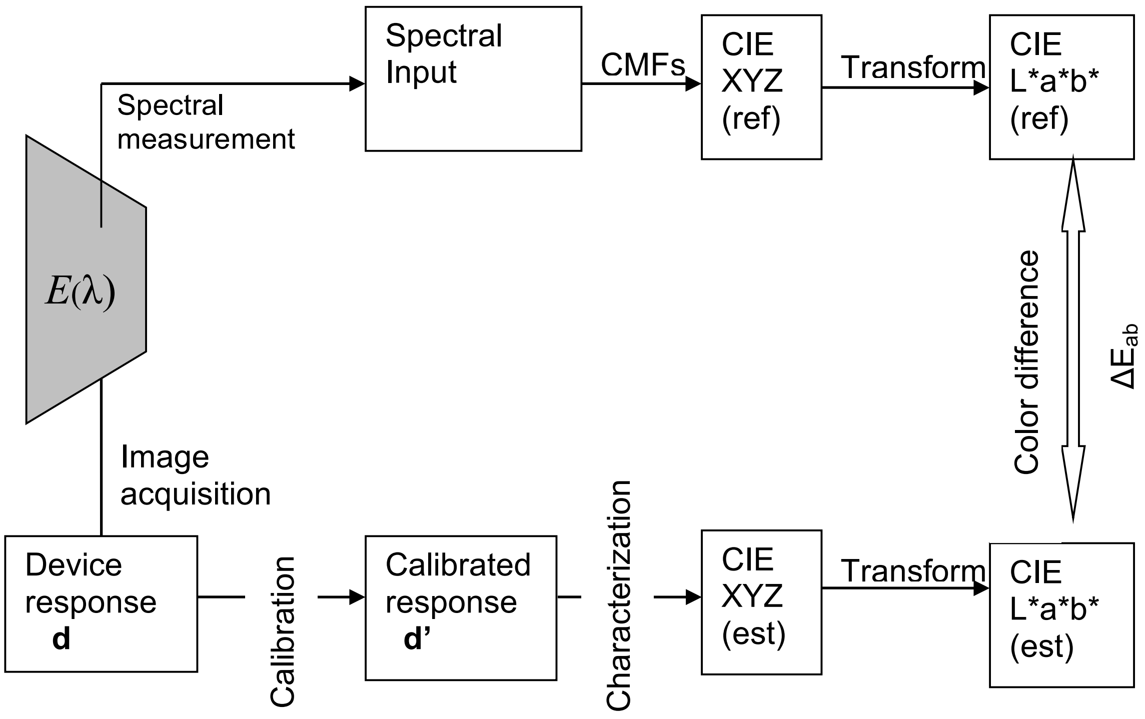

Figure 4. Evaluation of input device characterization.

3.1

Start by computing XYZ-values for the 20 surfaces in *chips20*, under the illuminant D65, which you will use as references (Fig. 4). Save it as XYZ_D65_ref (or similar). When you compute XYZ from spectra, always make sure that you use the correct normalization factor, so that Y=100 for a reference white (i.e. R=ones(1,61)). You can simply use Eq. 7, replacing the camera sensitivity functions with the CMFs (available in the file *xyz.mat*) and then multiplied by the normalization factor.

3.2

In the previous labs in TNM059, you were given a 3x3 matrix that you used for conversions between RGB and XYZ. This is a general conversion, based on sRGB, and not always optimal for all devices. In the data, you have the matrix MXYZ2RGB. Let's start to evaluate the performance of this general matrix conversion, for this specific camera. Use your calibrated R'G'B' values (RGB_cal_D65) to compute the corresponding XYZ values for the 20 objects, using the **inverse** of the matrix MXYZ2RGB. To evaluate the result, you need to first convert the XYZ-values to CIELAB and then compute the ΔE_{ab} color difference between the references and the estimations. For the CIELAB conversion you can use the function *xyz2lab.* What is the mean and maximum ΔE_{ab} color difference for the 20 colors/objects? Are the results acceptable?

(Hint: you will perform the operation of converting XYZ-values to CIELAB and computing mean and maximum ΔE_{ab} several times (in this lab and the next), so writing a function for this could prove useful.)

....
....
....
....
....

To have an exact characterization using a simple 3x3 matrix is only possible if there exists a unique non-singular transform between the sensor sensitivities and the color matching functions (this is called the *Luther Ives condition*). In such cases the device is called a *colorimetric device*, and the characterization is simple and exact. However, in practice, it is difficult to construct color filters and sensors that fulfill this condition.

3.3

Compare the spectral sensitivity functions for the camera ($\mathbf{A}_d$) to the CMFs of the standard observer (**xyz**) by plotting them against the wavelengths 400:5:700 nm. Comments?

....
....

Since the camera is not a colorimetric device, it is difficult to derive the inverse function, based on the camera characteristics. Instead, we will use empirical techniques for the inverse characterization. Least-squares regression techniques are widely used in color imaging, device characterization and modeling. The simplest form is Linear least squares regression, where the characterization function is approximated by a linear relationship $\mathbf{c} = \mathbf{d} \cdot \mathbf{A}$, where $\mathbf{d}$ is a 1 x *m* input color vector and $\mathbf{c}$ is a *1 x n* output color vector. The *m x n* matrix $\mathbf{A}$ is derived by minimizing the mean squared error of the linear fit to a set of training samples ($\{d_i\}$,$\{c_i\}$), as:

$$\mathbf{A}_{opt} = \arg\min_{A} \left\{ \frac{1}{T} \sum_{i=1}^{T} \left\| \mathbf{c}_i - \mathbf{d}_i \mathbf{A} \right\|^2 \right\} \qquad (9)$$

For a conversion from RGB (m=3) to XYZ (n=3), **A** is 3x3. If the XYZ-values for T samples are collected into a T x *3* matrix **C** = [$\mathbf{c}_1$; …;$\mathbf{c}_T$] and the corresponding RGB-values into a T x *3* matrix **D** = [$\mathbf{d}_1$; …;$\mathbf{d}_T$], the linear relationship is given by **C** = **D** · **A**. The optimal **A** is then given by **A** = **D**† **C,** where **D**† is the Moore-Penrose pseudo-inverse of **D** (in Matlab: *pinv*). This requires for $T \geq m$, i.e. we need at least as many sample points as the dimensionality of the input data. For the preferred case, $T > m$, we have an overdetermined system of equations for which we derive the least-squares solution.
To summarize, the conversion matrix **A** is calculated by **A**=pinv(**D**)*C. The matrix **D** is a *20 x 3* matrix containing the RGB values (the matrix you found in assignment 2.2, notice that you might need to transpose that matrix to make it *20 x 3*). The matrix **C** is a *20 x 3* matrix containing the corresponding XYZ-values (the matrix you found in assignment 3.1, notice that you might need to transpose that matrix to make it *20 x 3*). When **A** is found, the conversion from RGB to XYZ could be found by **D*A**, where **D** and **A** are the matrices specified before.

3.4

Use your calibrated R'G'B' values (RGB_cal_D65) to compute the optimal matrix **A**, using linear regression. This 3x3 matrix defines the conversion between device RGB and XYZ, based on the relation between measured XYZ-vales and recorded RGB-signals, for the specific 20 color samples. Now use your 3x3 matrix **A** to estimate XYZ-values for the 20 color samples. As earlier, compute the ΔE_{ab} color difference between the references and the estimations and report the mean and maximum ΔE_{ab} for the 20 objects. Comments?

………
………
………
………
………

Polynomial regression is a special case of least squares fitting, where the characterization function is approximated by a polynomial. The inverse characterization function of a 3-channel system, mapping RGB values to XYZ tristimulus values, is obtained by expressing XYZ as a polynomial function of R, G and B. As an example, a second order polynomial approximation is given by:

$$\left[X\,Y\,Z \right] = \left[1, R, G, B, R^2, RG, RB, G^2, GB, B^2, RGB \right] \begin{bmatrix} w_{X,1} & w_{Y,1} & w_{Z,1} \\ w_{X,2} & w_{Y,2} & w_{Z,2} \\ & \cdots & \\ w_{X,11} & w_{Y,11} & w_{Z,11} \end{bmatrix} \qquad (10)$$

By rearranging the data into a polynomial vector, the polynomial regression is reduced into a linear least squares regression, to determine the optimal weights, w.

3.5

You are provided with the function *Optimize_poly.m* that can be used to compute the optimal 11x3-matrix **A**, containing weights for polynomial regression, according to Eq. 10. The function *Polynomial_regression.m* then uses the matrix **A** to compute XYZ-values from RGB. Make use of these functions to compute XYZ-values for the 20 samples, using polynomial regression. Compare the results (i.e. the mean and max ΔE_{ab} color difference) to your previous results.

....
....
....
....
....

Note that in a “real” camera characterization set-up, the training set used for deriving the inverse characterization by regression, should optimally be larger than 20 samples. The result should also be evaluated using a different set of color samples, for verification. Since the parameters have been optimized for these specific color samples, the estimation errors will always be smaller for the training set.

Summary

In this lab, you have used a model-based approach for the forward image acquisition model, i.e. you have simulated the camera response based on the interaction between light-sources, object reflectance and the characteristics of two real RGB-camera sensors. The resulting RGB-signals have then been calibrated, to compensate for the characteristics of the camera sensors, and also to compensate for the illumination, using white point normalization.

For the inverse device characterization function, describing the transformation from the device-dependent RGB to the device-independent CIEXYZ color space, you used empirical characterization. This is a kind of “black-box”-approach, where the characterization functions are based only on the input and output for a training set of color samples, thus ignoring the characteristics of the device. The drawback of least-squared regression optimization is that the characterization will be accurate only for these specific conditions, and e.g. changing the illuminant would require a new characterization process.

Much of the data that you have required in this lab, such as the camera RGB-response (both un-calibrated and calibrated), the reference XYZ-values and the different XYZ-estimations from device characterization, will be further used in the next lab, focusing on characterization and modeling of output devices.