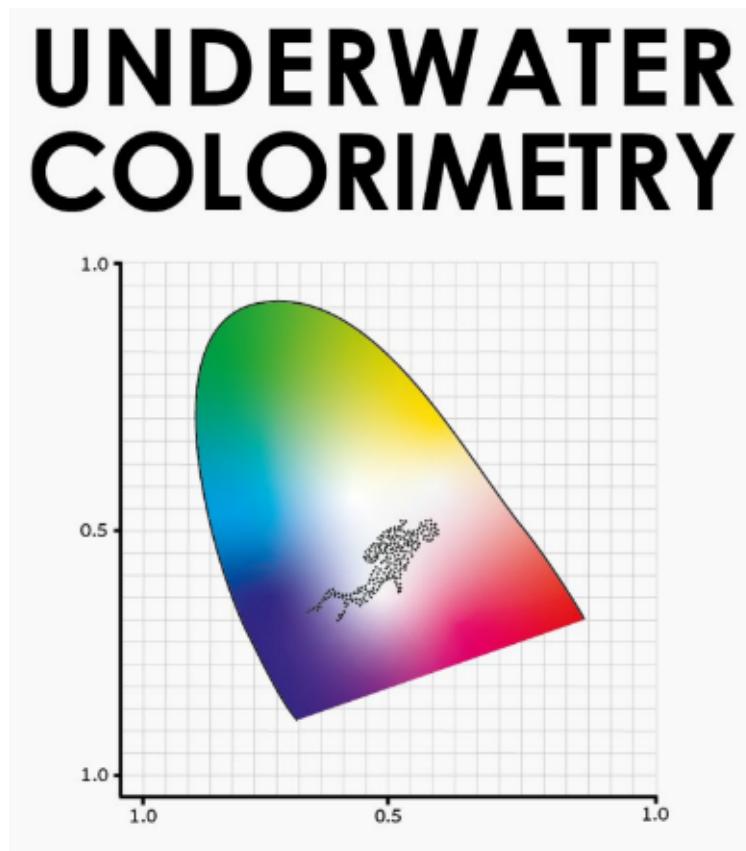


# Underwater Colorimetry

## Lab 2: Basic Colorimetry

IUI

January 2026



## Basic Colorimetry

### Session objectives:

1. Learn how to build an RGB-to-XYZ transformation matrix.
2. Given an image in camera RGB space, convert it to XYZ.
3. Obtain chromaticity  $xy$  values from XYZ tristimulus values.
4. Plot  $xy$  values on the CIE chromaticity diagram.
5. Compare RGB values from two different cameras, before and after transformation to a standard color space.

### Required equipment:

1. Computer - we recommend using the IUI's computers as the code we provide is already tested on them.
2. MATLAB or Python - we provide code in MATLAB but feel free to write your own in another language
3. RAW images

### Provided data:

Underwater Colorimetry GitHub Repository

Download the repository as a .zip file. It is very important to place the repository in a folder whose path **does not** contain any spaces or special characters!

Provided file	Comment
MacbethColorCheckerReflectances.csv	Reflectances of all patches of a Macbeth ColorChecker: The 1-24 corresponds to the patches in the numbering order given here.
illuminant-D65.csv, illuminant-A.csv	Spectral power distributions of two CIE standard illuminants: daylight: D65, incandescent: A.
NikonD90.csv, Canon_1Ds_Mk_II.csv	Spectral responses of two cameras, Nikon D90 and Canon 1Ds II.
CIEStandardObserver.csv	CIE 1931 2-degree standard observer curves.
NikonImage.NEF	An image containing a Macbeth ColorChecker acquired with a Nikon D90 in RAW format.
CanonImage.CR2	An image containing a Macbeth ColorChecker acquired with a Canon 600D in RAW format.

Table 1: Provided files and their descriptions.

## Lab Report Due

Sunday 18.1.26 at 9:00 am, by email

Submit to: [uwcolorimetry@gmail.com](mailto:uwcolorimetry@gmail.com)

Your email title should include the lab number, your name and affiliation!!!

For example: Lab 2 - James Bond - University of Integers

## Lab report: Maximum 3 Pages!

Please keep your reports clean and professional

## Lab Overview

**Exercise 1:** *Quantitative color comparison between different cameras*

**Exercise 2:** *Camera RGB to XYZ transformation*

**Exercise 3:** *Calculate the xy white point coordinates of the illuminant you used*

**Exercise 4:** *XYZ to sRGB transformation*

**Exercise 5:** *RAW and JPG comparison*

## Exercise 1

### *Quantitative color comparison between different cameras*

In Exercise 3 of Lab 1, you simulated a “photograph” of a Macbeth ColorChecker by two different cameras (A Nikon and a Canon) under the same illuminant. Now we want to quantitatively check if those two cameras, and a random third one, would capture colors the same way or not.

Use only illuminant A for this exercise (but for your own curiosity, do check what happens with illuminant D65).

Use the Nikon and Canon spectral responses from Lab 1. Choose a third (random) camera from this database and download its spectral sensitivities:

Spectral sensitivities

### Steps

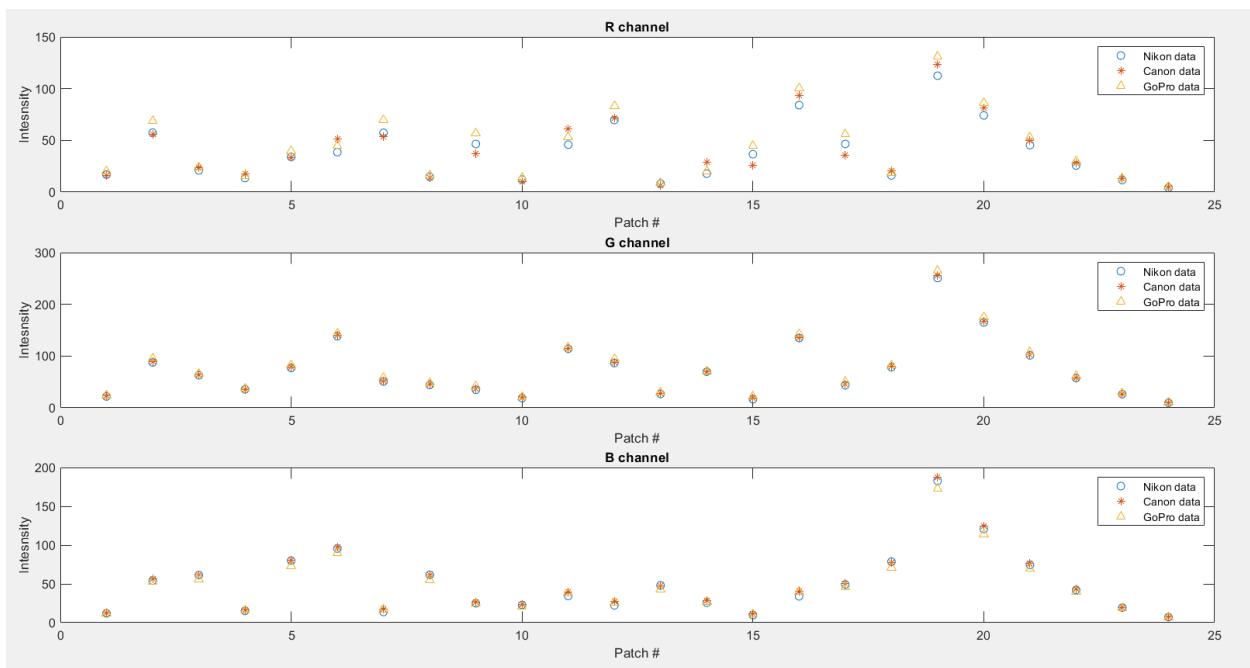
1. Make a simple plot comparing the RGB values of each patch (see example below), as would be captured by each camera. The x-axis will be patch number 1-24 and the Y-axis will be the captured intensity in a given color channel.
2. Since there are 3 color channels, please make 3 subplots and mark each camera with a different marker.
3. Don't forget to add a legend.

## Include in your report - Exercise 1

1. The plots you made showing differences of the captured RGB values for the three different cameras (Nikon, Canon, and random camera - don't forget to include its name!).

Are the large differences?  
small differences?  
Uniform across colors?

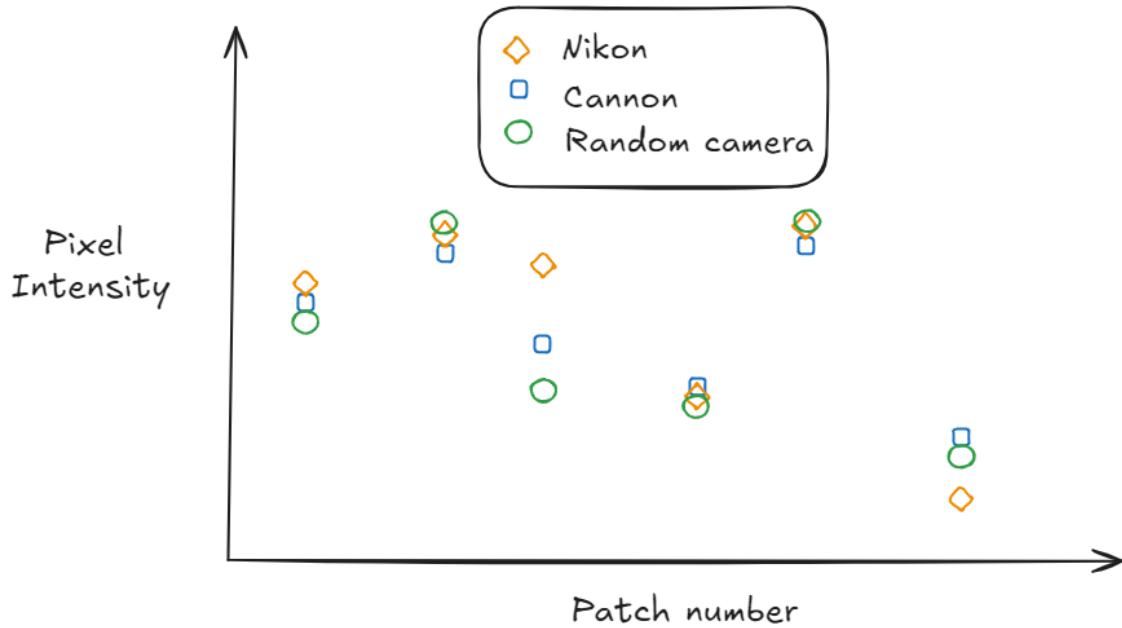
Discuss. Don't forget to note which illuminant and camera you used.



### Guidance for writing the code

*The full code for this exercise is available on the course GitHub.  
If you feel comfortable enough, try to follow this guidance and  
write your own code.*

1. Use the *importdata* function to import:
  - (a) Spectral sensitivities (Nikon, Cannon and random camera)
  - (b) Reflectances of Macbeth ColorChecker
  - (c) Illuminant (D65/A)
2. Create wavelength range of 400 : 700nm and use the *interp1* function to interpolate the imported data to the same range.
3. Use the *getradiance* function from lab 1 to extract, or simulate, the RGB values of each patch given a specific camera under the chosen illuminant.
4. Plot the results.



## Exercise 2

### *Camera RGB to XYZ transformation*

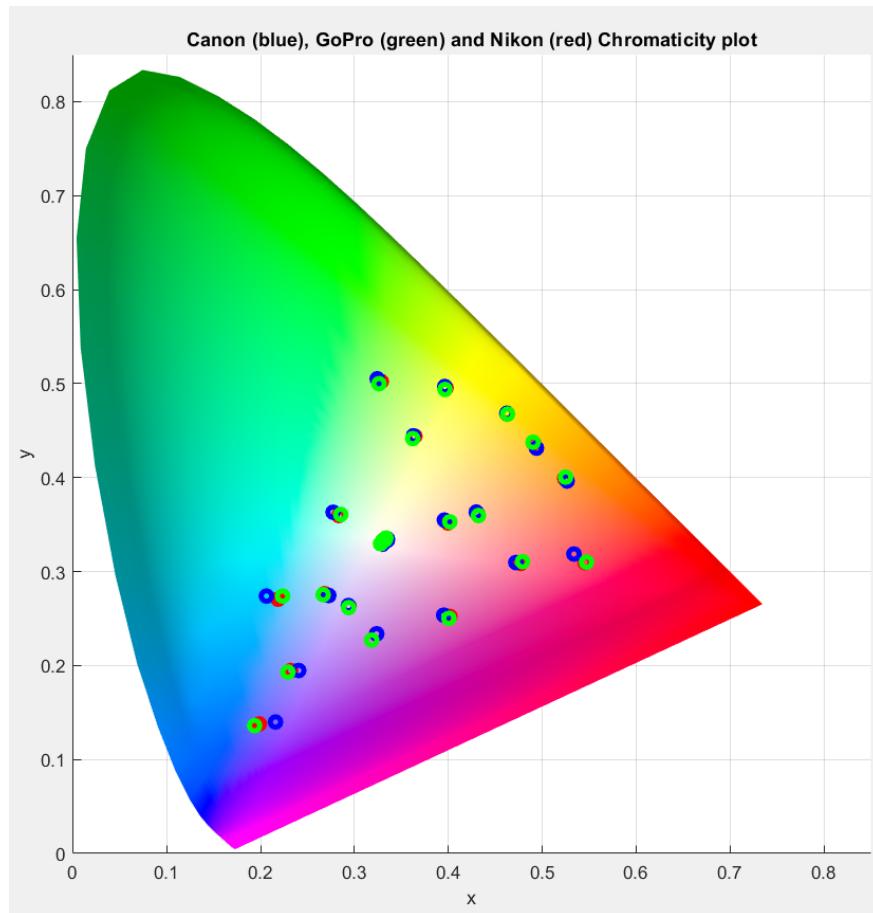
**Steps:**

1. For each of the 3 cameras camera, under illuminant A, build a  $3 \times 3$  transformation from the (white balanced) camera RGB to (white balanced) XYZ values. Calculate the xy-values for both cameras.
2. Plot them on the CIE chromaticity diagram (see example below). Use different colors (or markers) to denote patches that came from each camera.

### Include in your report - Exercise 2

1. The CIE chromaticity diagram that clearly shows the location of the 24 patches of the color chart for each camera (use a different color or different marker to distinguish cameras).

Don't forget to mention which illuminant you used. Are the xy values captured by each camera similar? Identical? Discuss.



## Guidance for writing the code

*The full code for this exercise is available on the course GitHub. If you feel comfortable enough, try to follow this guidance and write your own code.*

1. Import the standard observer curves and interpolate to the defined wavelength range.
2. Use the *getradiance* function to get the XYZ values.
3. From the XYZ values derive the xy values.
4. White balance the XYZ values.
5. White balance the RGB values of each one of the 3 cameras.
6. For white balancing - pick the 23rd gray, with 9% reflectance but experiment with different patches!
7. Camera RGB to XYZ using camera specific transformation matrix  $T_{camera}$

$$[XYZ] = T_{camera} \cdot [RGB]'$$

Where:

$$[XYZ] \in 24 \times 3$$

$$T_{camera} \in 3 \times 3$$

$$[RGB]' \in 3 \times 24$$

8. Calculate the XYZ values from each camera using the transformation matrix  $T_{camera}$

$$[XYZ]_{camera} \approx T_{camera} \cdot [RGB]$$

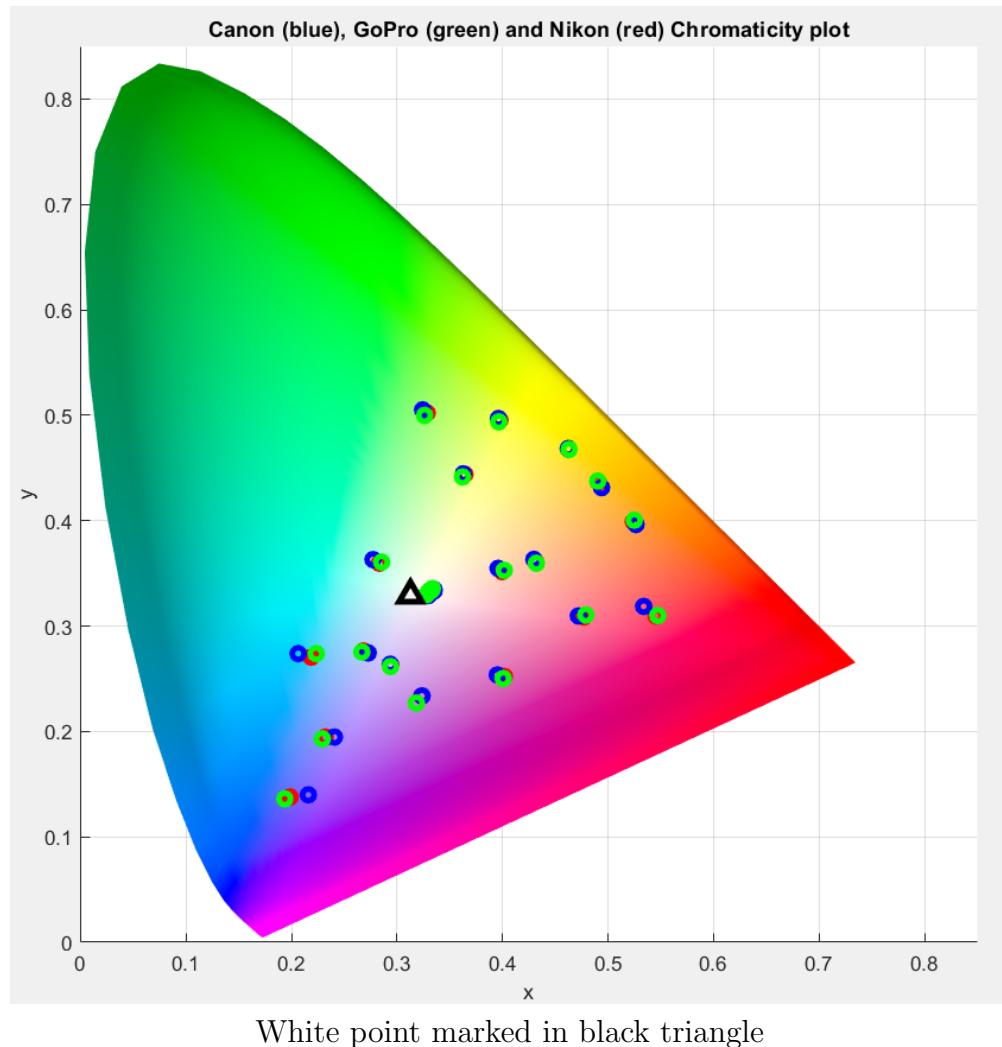
9. Obtain the *xy* coordinates for the 2D chromaticity diagram plots.

## Exercise 3

*Calculate the xy white point coordinates of the illuminant you used*

Plot the white point on top of the previous chromaticity diagram from exercise 2.

**Include in your report - Exercise 3**



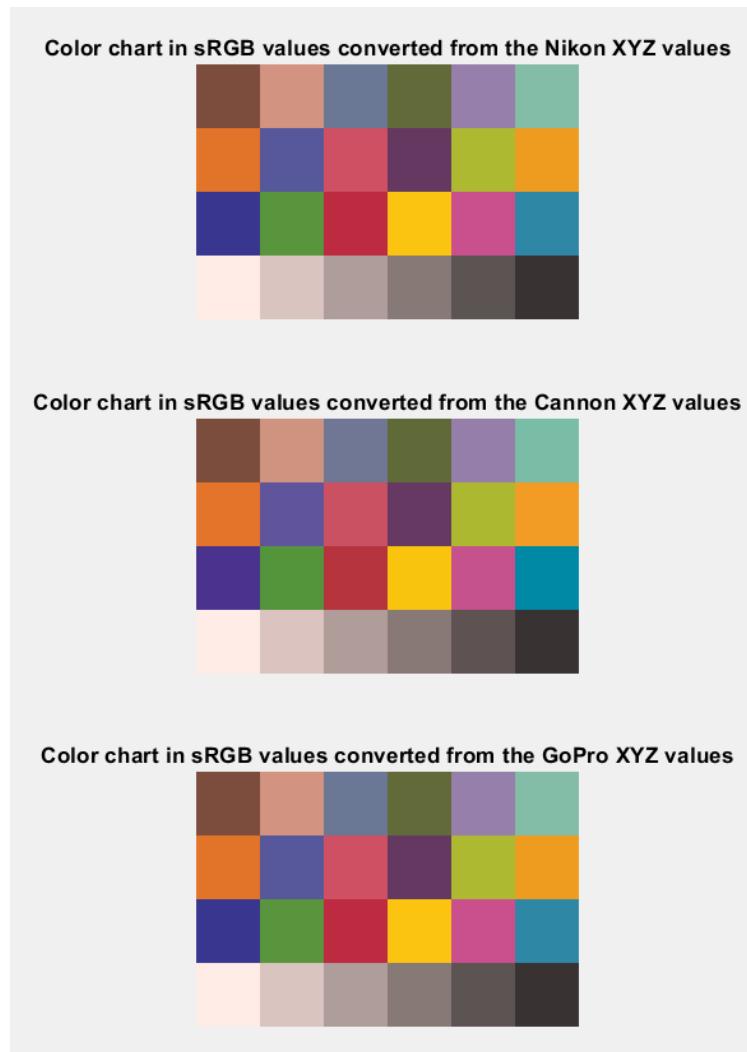
## Exercise 4

### *XYZ to sRGB transformation*

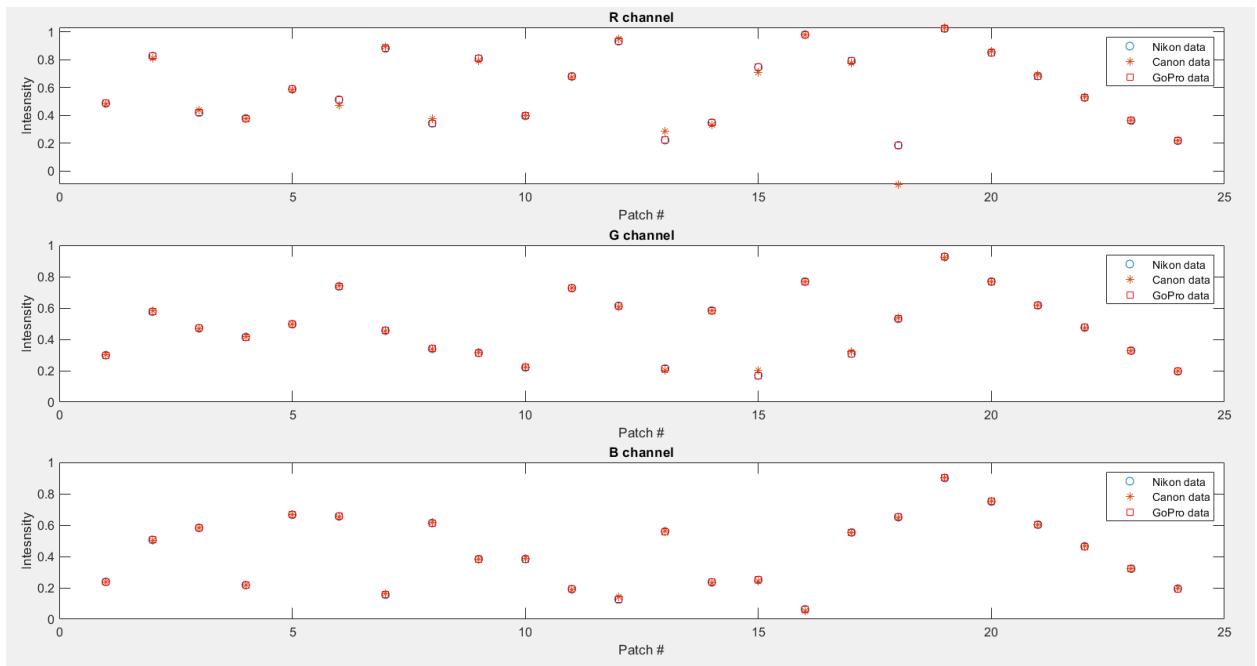
Convert XYZ values from the 3 cameras to sRGB color space. Don't forget to check if chromatic adaptation is needed (if the XYZ is already white balanced, it is not needed).

### **Include in your report - Exercise 4**

1. The sRGB appearance of your color chart as "photographed" by each camera.



2. Similar plot to the one in Exercise 1, but now using the sRGB values for all 3 cameras.



3. Discuss what differences you see.

### Guidance for writing the code

*The full code for this exercise is available on the course GitHub. If you feel comfortable enough, try to follow this guidance and write your own code.*

1. Use the `xyz2rgb()` function to obtain sRGB values XYZ values for each image.
2. Visualize the simulated sRGB color-chart for each camera in **similar way you did in lab 1 - exercise 3**
3. Plot the three simulated color-charts in sRGB using the `subplot` function, one simulated color-chart per camera.
4. In new figure, plot the sRGB values for each patch in all cameras.

## Lab Report Due

Sunday 18.1.26 at 9:00 am, by email

Submit to: [uwcolorimetry@gmail.com](mailto:uwcolorimetry@gmail.com)

Your email title should include the lab number, your name and affiliation!!!

For example: Lab 2 - James Bond - University of Integers

## Lab report: Maximum 3 Pages!

Please keep your reports clean and professional

## Lab Overview

**Exercise 1:** *Quantitative color comparison between different cameras*

**Exercise 2:** *Camera RGB to XYZ transformation*

**Exercise 3:** *Calculate the xy white point coordinates of the illuminant you used*

**Exercise 4:** *XYZ to Standard RGB transformation*

**Exercise 5:** *RAW and JPG comparison*