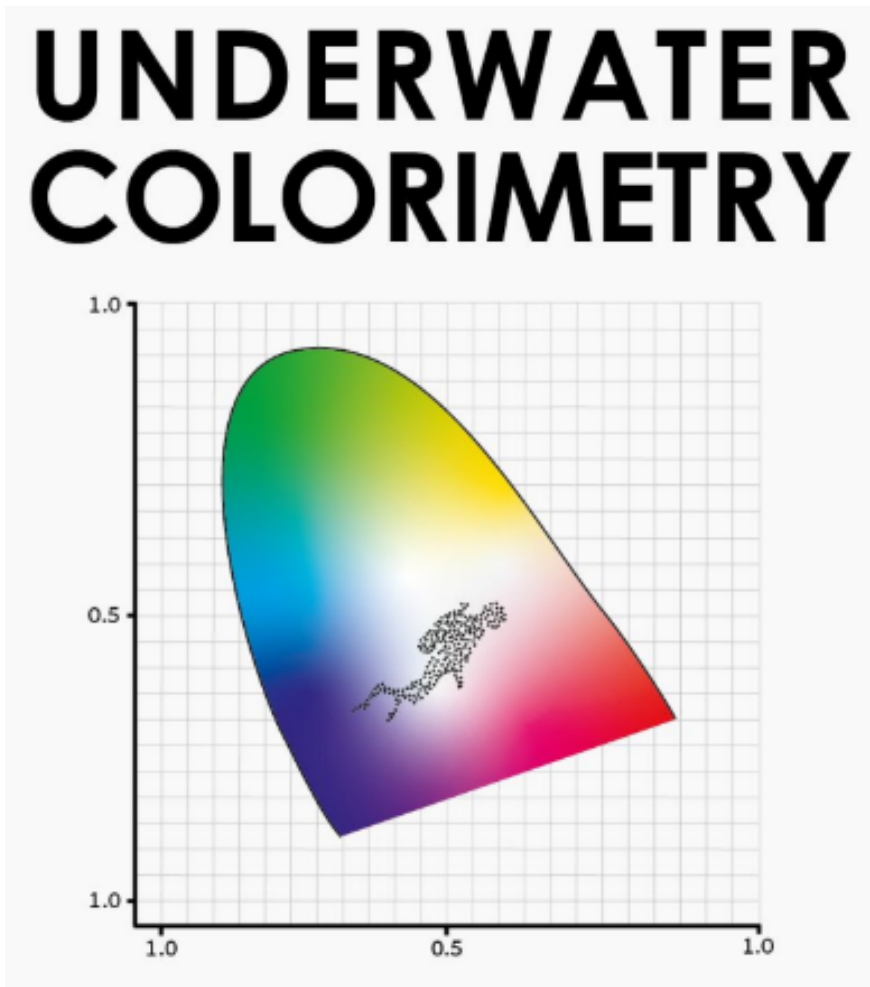


Underwater Colorimetry

Course Project

University of Haifa

January 2025



Contents

1	Underwater Colorimetry 2025 Project - Overview	3
1.1	Goal of the project	3
1.2	Deliverables	3
1.3	Grading	3
1.4	Timeline	3
1.5	Dataset	3
2	Project Instructions	5
2.1	Exercise 1: Extract RGB values from all charts	5
2.2	Exercise 2: Extract D values from all charts	5
2.3	Exercise 3: Calculate backscatter at each chart	5
2.4	Exercise 4: Subtract the backscatter from your raw (linear) image to obtain the direct signal	6
2.5	Exercise 5: Calculate the attenuation coefficient β_c^D for each color chart . . .	7
2.6	Exercise 6: Recover J_c , the image with attenuated colors, for each color chart	7
2.7	Exercise 7: White balance the “ J_c ” images for each chart	8
2.8	Exercise 8: Obtain XYZ values for the reference color chart that is used in the photos	8
2.9	Exercise 9: Build a camera to XYZ color transform	9
2.10	Exercise 10: Apply matrix M to bring your colors to the XYZ color space . .	9
2.11	Exercise 11: Convert colors from all charts in XYZ color space to sRGB . . .	10
2.12	Exercise 12: Calculate errors between different color charts	10
3	Include in your report	11

1 Underwater Colorimetry 2025 Project - Overview

1.1 Goal of the project

Apply principles of underwater image formation and colorimetry we learned in the course to a real-world dataset.

1.2 Deliverables

- Project report (individual)

1.3 Grading

The project makes up 30% of your total grade. Grading will not only be based on your results but also on the effort you put in and your understanding of the concepts. So please take care to explain/discuss your findings.

1.4 Timeline

The project reports will be due on the day of the final presentations (lottery).


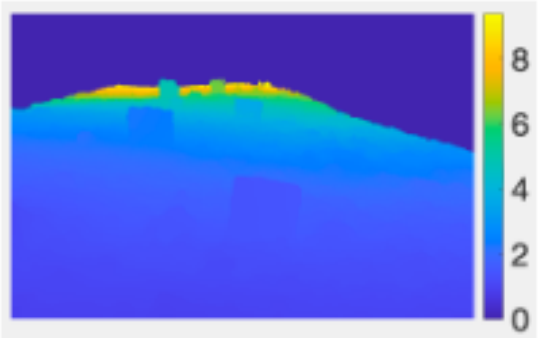
1.5 Dataset

We have provided three RGB-D datasets with multiple colors charts in them:

1. Dataset1_Eilat
2. Dataset2_Eilat
3. Dataset3_Florida

In each dataset, you will find 10 “good” images we picked for you, in which all the color charts are visible. Each student will process one image from a dataset they choose (there is a signup sheet [HERE](#)). **No two students will process the same image.**

In the folder of each photo, there is a linear RGB image (in .png format) and a D image (in .tif) format. In the D image, the distances are scaled in meters (below we show you in false color; the image preview might look black and white but the data are there).

RGB Image (png format)	D Image scaled in meters (tiff format)
	

Some notes about the data

1. The depth maps (“D” images) we provide you were made in Agisoft MetashapePro, exactly the way we did during Lab 3. When Agisoft exports depth maps, it puts a value of zero (0) to pixels for which no depth information was calculated (e.g., water pixels). In our nomenclature, a distance value of 0 denotes the location of the camera and theoretically it is impossible to capture an object at $z = 0$. For your project, you don’t need to access water pixels, just be aware 0 is not a valid value. Filter it out from your data (you can replace it with NaNs) and do not use zero values.
2. Often, the furthest distance values in a D image are inaccurate. If you notice your data behaving weirdly at very large distances, filter out those D values.
3. These are real-world data and they are not perfect. For some datasets, the D image values may not exist for some of the far-away charts. All you need to do these exercises are to have 2 color charts with D information. Therefore, even if you do not have D for ALL color charts, you can still do everything.

2 Project Instructions

2.1 Exercise 1: Extract RGB values from all charts

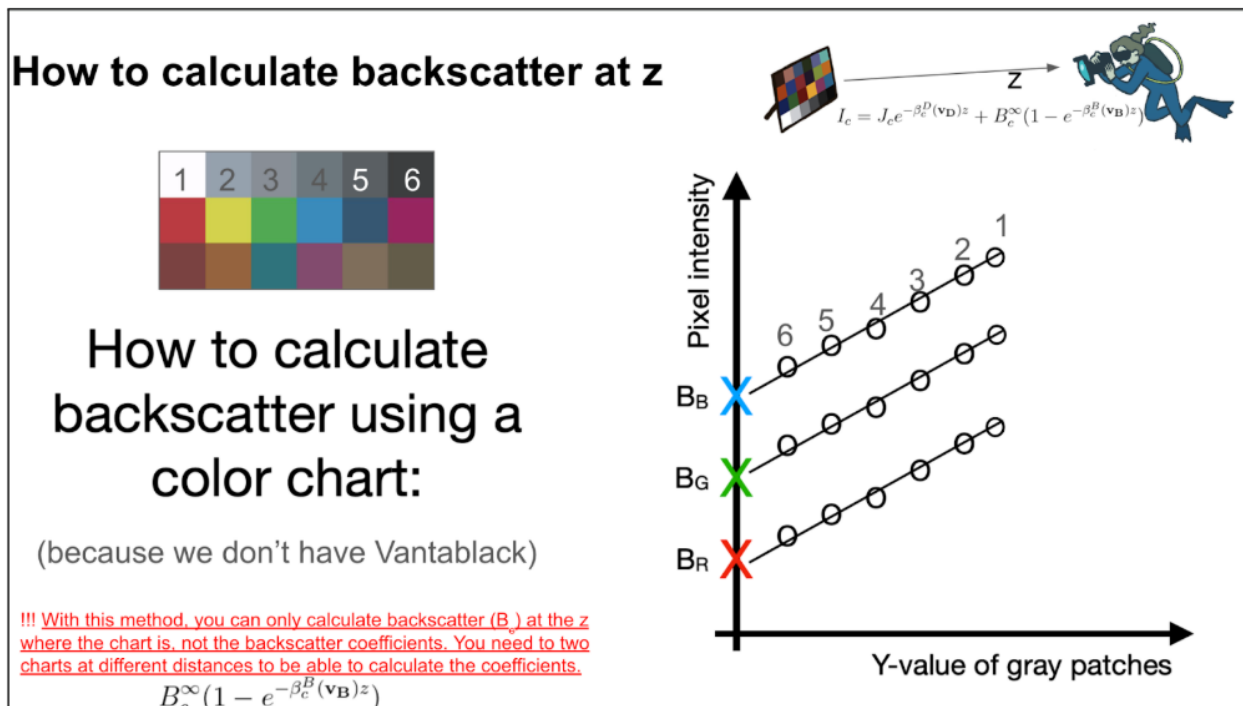
The linear-PNG images are in the camera-specific RGB color space. From these images, extract RGB values for each patch of each color chart. Our underwater color charts have 18 patches. So let's say if you have 5 color charts in an image, you will extract 5 matrices of 18×3 . Save these because you will need to reuse them.

2.2 Exercise 2: Extract D values from all charts

Using the D-images, extract the depth (range) value for each color chart. Here, there is no need to do it at patch level, because we can assume all patches on a given chart are at the same distance from the camera. So if you have 5 color patches in your scene, you will end up with a vector of 1×5 . Save these because you will need to reuse them.

2.3 Exercise 3: Calculate backscatter at each chart

Following the method described in the lecture notes, calculate Backscatter B_c at each color chart. If you have 5 color charts in the scene, you should end up with 5 backscatter values, so a matrix of 5×3 . Save these values.



Include in your report (Exercise 3)

1. A plot of B_c (y-axis) as a function of z (x-axis). Color code color channels (e.g., use red markers for the red channel, green markers for the green channel, and blue markers for the blue channel).
2. A visualization of the B_c values in the format of the color chart. You can use the `visualizeColorChecker()` function provided to you in Lab 2, but you will need to adjust it to this color chart which only has 18 patches.

2.4 Exercise 4: Subtract the backscatter from your raw (linear) image to obtain the direct signal

From the raw-RGB values you obtained in **Exercise 1** for each chart, subtract the B_c to obtain D_c .

Include in your report (Exercise 4)

1. A plot of D_c (y-axis) as a function of z (x-axis). Color code color channels (e.g., use red markers for the red channel, green markers for the green channel, and blue markers for the blue channel).
2. A visualization of the D_c values in the format of the color chart.

2.5 Exercise 5: Calculate the attenuation coefficient β_c^D for each color chart

Following the method described in the lecture notes, calculate the attenuation coefficient β_c^D using pairs of color charts (one at distance z and one at distance $z + \Delta z$). Remember that β_c^D is a function of distance and it will have a different value depending on which pairs of charts you use.

Calculate it for several pairs of charts because due to the real-world nature of the data, it might work better with some pairs than others.

Each β_c^D will be a vector of size 1×3 .

How to calculate attenuation at z

This is what is left after subtracting B_c :

$$D_c = J_c e^{-\beta_c^D z}$$

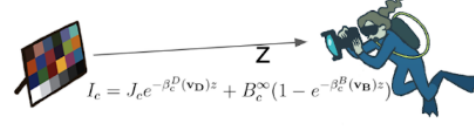
Now need to calculate β_c^D to be able to recover J_c

$$J_c = D_c e^{\beta_c^D z}$$

$$\beta_c^D = \ln \frac{D_c(z)}{D_c(z + \Delta z)} / \Delta z$$

You need two color charts, at distances z and $z + \Delta z$

Akkaynak et al. "What is the space of attenuation coefficients in underwater computer vision?" IEEE CVPR 2017.



$$I_c = J_c e^{-\beta_c^D (vD)z} + B_c^\infty (1 - e^{-\beta_c^B (vB)z})$$

2.6 Exercise 6: Recover J_c , the image with attenuated colors, for each color chart

Using the attenuation coefficient(s) you got in **Exercise 5**, recover J_c as described in the lecture notes. For each chart, J_c will be of size 18×3 .

Include in your report (Exercise 6)

1. A plot of J_c (y-axis) as a function of z (x-axis) (should be more or less constant across z). Color code color channels (e.g., use red markers for the red channel, green markers for the green channel, and blue markers for the blue channel).
2. A visualization of the J_c values in the format of the color chart.

2.7 Exercise 7: White balance the “ J_c ” images for each chart

White balance your resulting J_c image using any gray patch in any chart. This should be possible because at this point, you have compensated for all the effects of water due to z , and now you only need to correct global illumination.

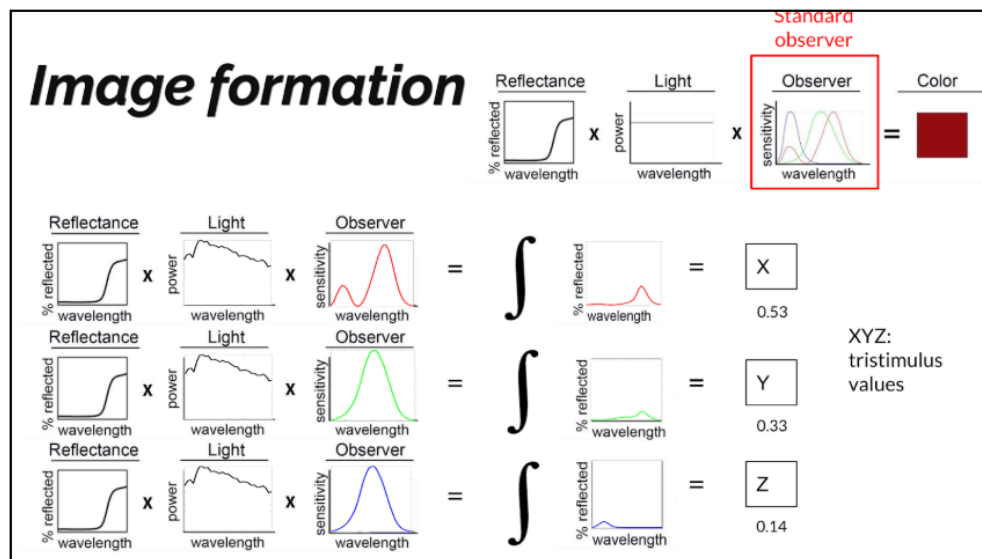
Include in your report (Exercise 7)

1. A plot of white-balanced J_c (y-axis) as a function of z (x-axis). Color code color channels (e.g., use red markers for the red channel, green markers for the green channel, and blue markers for the blue channel).
2. A visualization of the white-balanced J_c values in the format of the color chart. Remember to note with which patch you white balanced!

2.8 Exercise 8: Obtain XYZ values for the reference color chart that is used in the photos

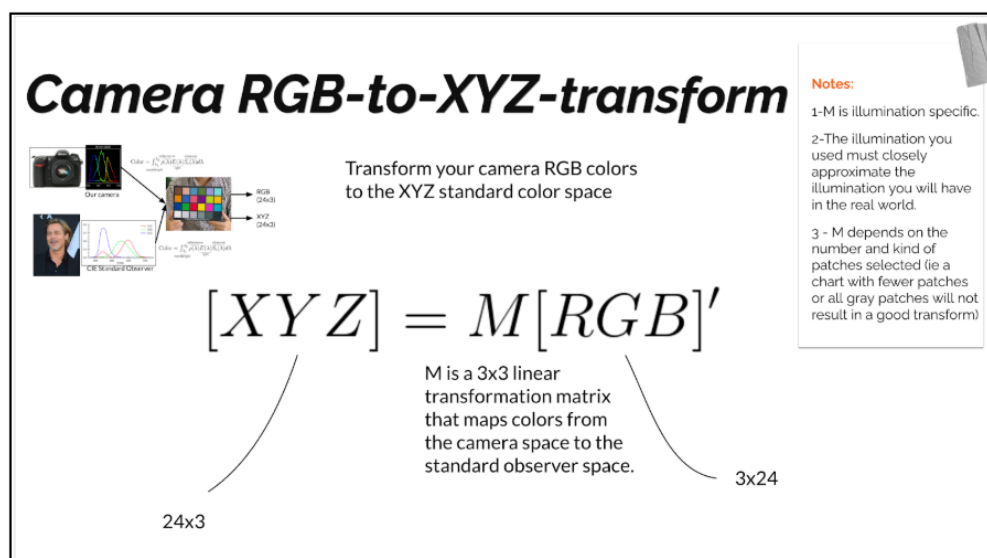
The chart we take underwater is called the DGK color chart. It has 18 patches. We provide you reflectance spectra for each patch in a file named `DGKcolorchart_reflectances.csv`. Following the lecture notes, using $D65$ illuminant and the standard observer curves, obtain the XYZ values for this chart. Remember to white balance these values with the same patch you used in **Exercise 7**.





2.9 Exercise 9: Build a camera to XYZ color transform

At this point, if everything worked well to **Exercise 7**, all your color chart J_c images should be identical. You can use any of them to build your $RGB2XYZ$ color transformation matrix M .



2.10 Exercise 10: Apply matrix M to bring your colors to the XYZ color space

Apply the M and obtain the XYZ versions of your color chart RGB values. You have code to do this with from Lab 2.

2.11 Exercise 11: Convert colors from all charts in XYZ color space to sRGB

Convert your chart XYZ values to sRGB color space. You can look at Matlab's built-in functions `makecform()` and `applycform()` to use a standard transformation from `xyz2srgb`. If you are working in a different programming language/platform, you can look up the conversion matrices from XYZ to sRGB as these are standard and can be found online or in Colorimetry books.

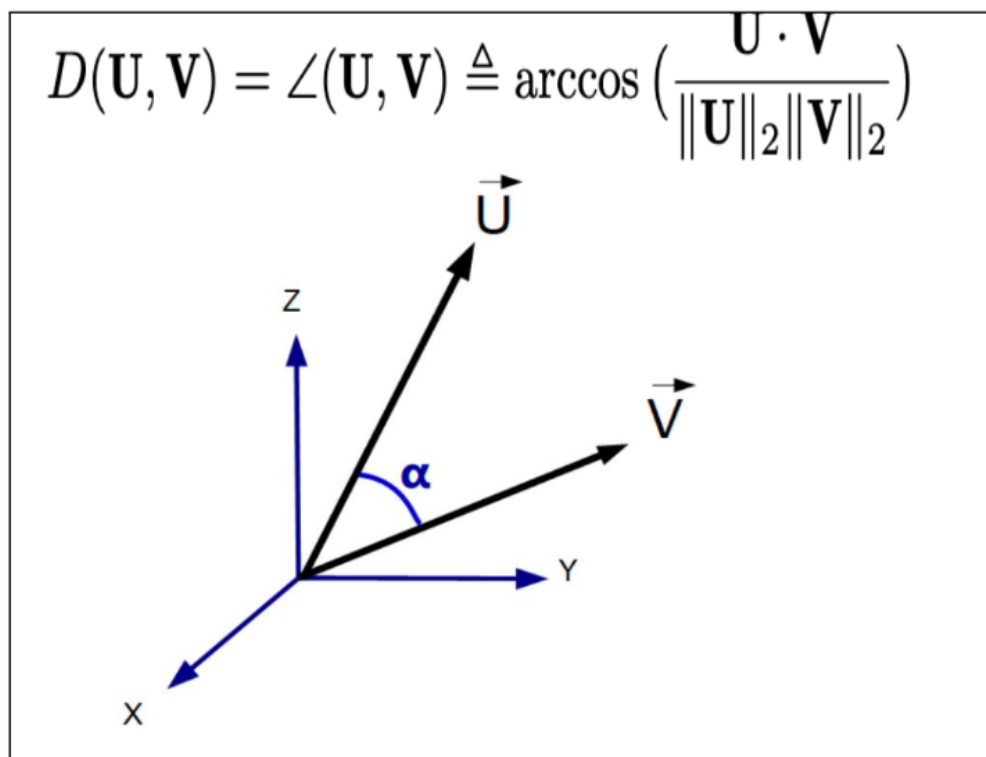
Include in your report (Exercise 11)

1. A plot of J_c in sRGB color space (y-axis) as a function of z (x-axis). Color code color channels (e.g., use red markers for the red channel, green markers for the green channel, and blue markers for the blue channel).
2. A visualization of the sRGB J_c values in the format of the color chart. Remember to note with which patch you white balanced!

2.12 Exercise 12: Calculate errors between different color charts

Compare, using the color angle error, the errors between **each patch across different charts**.

For example: compute the error between white patch from chart 1 and white patch from chart 2, and then chart 3, etc (do all combinations). Compute the average error across all charts for the white patch. Repeat for all patches.



Include in your report (Exercise 12)

A plot of average errors (y-axis) per patch (x-axis).

Feel free to include any other details or questions/observations in your report, or if you have suggestions for computing things differently. Underwater Colorimetry is a new and emerging field and there is a lot of work to be done!

Extra credit of 10% and an offer to join the COLOR Lab: present results from all images from all datasets.

3 Include in your report

1. A plot of B_c (y-axis) as a function of z (x-axis). Color code color channels (e.g., use red markers for the red channel, green markers for the green channel, and blue markers for the blue channel).
2. A visualization of the B_c values in the format of the color chart. You can use the `visualizeColorChecker()` function provided to you in Lab 2, but you will need to adjust it to this color chart which only has 18 patches.
3. A plot of D_c (y-axis) as a function of z (x-axis). Color code color channels (e.g., use red markers for the red channel, green markers for the green channel, and blue markers for the blue channel).
4. A visualization of the D_c values in the format of the color chart.
5. A plot of J_c (y-axis) as a function of z (x-axis) (should be more or less constant across z). Color code color channels (e.g., use red markers for the red channel, green markers for the green channel, and blue markers for the blue channel).
6. A visualization of the J_c values in the format of the color chart.
7. A plot of J_c in sRGB color space (y-axis) as a function of z (x-axis). Color code color channels (e.g., use red markers for the red channel, green markers for the green channel, and blue markers for the blue channel).
8. A visualization of the sRGB J_c values in the format of the color chart. Remember to note with which patch you white balanced!
9. A plot of average errors (y-axis) per patch (x-axis).