Exercise 1: Manipulating RAW images

1. Go to the course repository → **HERE** to access "camera-pipeline-dng-sdk".

   - If using matlab, you can use the `dng2tiff.m` function we provide there to access the SDK.
   - If using python, you will need to write your own version of `dng2tiff` based on the matlab version above. Feel free to use *chatgpt* for this task. There is also a jupyter notebook example HERE which may be helpful (or confusing)!

   MAC users: you need to overwrite the `dng_validate.exe` that came in the downloaded SDK with the one from our course repository.

2. Download Adobe DNG Converter for your operating system from HERE. (https://helpx.adobe.com/camera-raw/using/adobe-dng-converter.html).

3. Take a RAW image with any camera (you can use the examples we provide in the course repository).

4. Follow the detailed instructions on the next page on how to use Adobe DNG Converter to convert your RAW image to .DNG format.

5. Run your `dng2tiff` function on the .DNG file you obtained in Step 4. This step will probably take some time and debugging.

6. You can do all analyses required for this course (and your research!) with the linear .tiff image you obtained in Step 5. However, .tiff images are large in size, so you might want to resize or convert the linear .tiff image to a linear .png image for faster processing. This is optional based on your project and computational resources.

Adobe DNG Converter Settings

Step 0: We suggest a folder structure in the Lab 1 GitHub, feel free to use it.

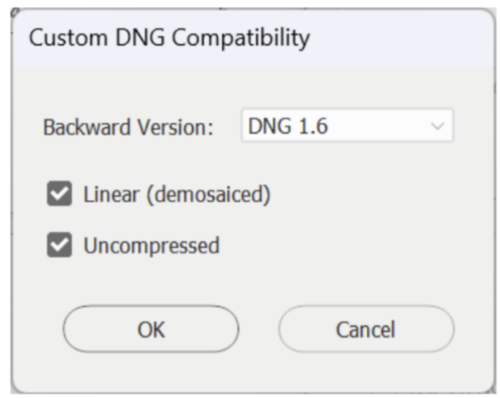
1. Double click and open the program. In section 1, click the "Select Folder" button and point it to the "RAW" folder.

2. In Section 2, click "Select Folder", and choose the "dng" folder.

3. No need to do anything in Section 3.

4. In section 4, click the "Change Preferences" button. In the next screen that opens up, click the "Compatibility" drop down list. Select "Custom" and check both "linear" and "uncompressed".

   Set JPEG preview to Medium Size.



   Leave "Lossy Compression" box UNchecked.

   Leave Embed Original Raw File UNchecked.

   Click OK to close the second window.

5. Click "Convert" to start raw to dng conversion! The dng file(s) should appear in the output folder you specified.

Exercise 2: Linearity Check

1. Convert one of the RAW images we provide (containing a Macbeth colorchecker under natural light) to a linear image.
2. Write a function named **makeMasks** to make masks for each achromatic patch. Each mask should contain the maximum possible number of pixels for that patch. It's always a good idea to save your masks in case you need them later.

   Note: Today you only need to make masks for 6 patches. Later in the course, you will need to make them for 24 patches, and always in a specific order. So think how you can make this function flexible.
3. Write a function named **extractRGB** that uses the masks and extracts the average RGB values of each specified patch.
4. Plot the RGB values of the achromatic patches against their Y value (these are provided for you in the course repository).
5. Now repeat the entire analysis for the JPG versions of the images we provided.

Exercise 3: Basic Image Formation

1. In the course repository, we provide you the spectral responses of two cameras (a Canon and a Nikon), the spectral power distribution of two illuminants (illuminant A and D65), and the reflectances of the 24 patches of a Macbeth ColorChecker.

   Write a function called **simulateRGB** (reference the lecture notes!) that simulates an image by calculating the RGB values a given camera would capture for a reflactance spectrum under a given illuminant. Make sure this function can take any number of reflectances.

   Note: The data will have different wavelength ranges and steps. You will need to interpolate them to a common range, for example $400 : 10 : 700$.

2. Write a function named `visualizeColorChart` that will display the calculated RGB values in a 4×6 grid, just like a color chart.

3. Write a function named `whiteBalance` that will white balance your simulated color chart based on a specific achromatic patch.