COMP2005 Laboratory Sheet 1B: Colour and Point Operations

Colour images are read and stored as 3D NumPy, and the indexing scheme is *image[row, column, colour]*. Please note that OpenCV reads images in **BGR** rather than RGB, so the colours are arranged in **B, G, R** order.

1. NumPy

As mentioned in the previous lab, NumPy is a library which provides convenient operations for multi-dimensional arrays and matrices. Take some time to look through the official guide of NumPy to better understand how NumPy works and the functionalities provided https://numpy.org/doc/stable/user/absolute_beginners.html.

- Install the NumPy library and import it using *import numpy as np*. NumPy is usually imported under the alias *np*.

2. Colour Space Conversion

Careful choice of colour space can make image processing tasks much easier. To gain some familiarity with colour and colour space conversion algorithms, read in an RGB colour image (You may use the RGB image of the Computer Science Atrium on the Moodle page).

REMINDER: Images read in Python are arrays of type uint8 so some operations can cause overflows. You may find it useful to convert to int or float before applying some processes and then convert it back to uint8.

- Display its red, green and blue components as three separate grey-level images.
- Convert the RGB image to greyscale. First, by simple averaging, then using the weighting function described in Lecture 3. Perform this by explicitly writing a loop. Then, compare your results with the OpenCV function *cvtColor.* (Hint: Use *empty* or *zeros* from NumPy)
- Perform the above again but using NumPy's array manipulation facilities. (Hint: It may be helpful to create a Python function for the weighted function)
- Produce and display an image of the atrium showing the 'greenness' value described in the lecture. This can be done either by explicitly writing a loop or (more efficiently) using the NumPy library.
- Convert the RGB image to HSV and display each channel as separate images.

3. Point Image Processing

Gamma correction is used to adjust the brightness and contrast of an image by applying a function which maps the intensity values in the image I to new values in O. The formula is shown in (1), where γ is the gamma value.

$$O = \frac{I}{255}^{\gamma} \times 255 \quad (1)$$

The gamma value specifies the shape of the curve that maps the intensity values I to create O (see Figure 1). If gamma is less than 1, the mapping is weighted towards brighter output values. If gamma is greater than 1, the mapping is weighted towards darker output values. If gamma is equal to 1, the mapping is linear, and there is no correction.

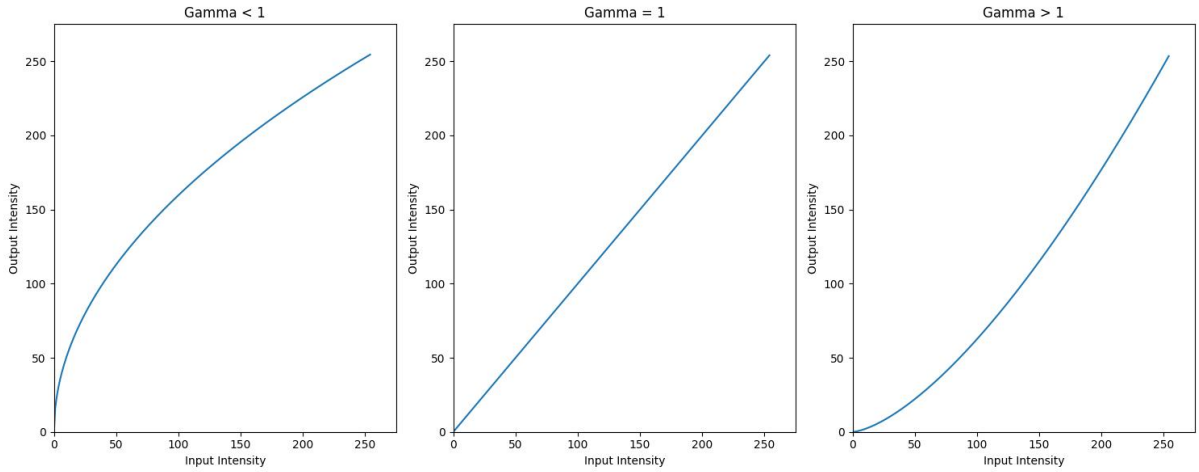

Figure 1

Using the cameraman.tif image from the Moodle page, perform the following:

- Using (1), apply gamma correction to the image with gamma values 0.5, 1 and 1.5 and display the results.
- Negate the image and display the results.

Gain and bias adjustments are used to control the brightness of an image. Gain adjustment multiplies the pixel values by a scale factor, whereas bias adjustment adds a constant value. This is done using (2) where α is the gain parameter and β is the bias parameter.

$$g(x, y) = \alpha f(x, y) + \beta \quad (2)$$

The effects of gain and bias adjustment can be observed through the histograms of the image. Histograms are commonly used in image processing to view the properties of an image and the effects of image processing techniques. We will explore more on histograms in Lab 3. Changing the bias shifts the image's histogram but maintains its shape while changing the gain stretches the histogram in the x-direction and flattens it (Figure 3).

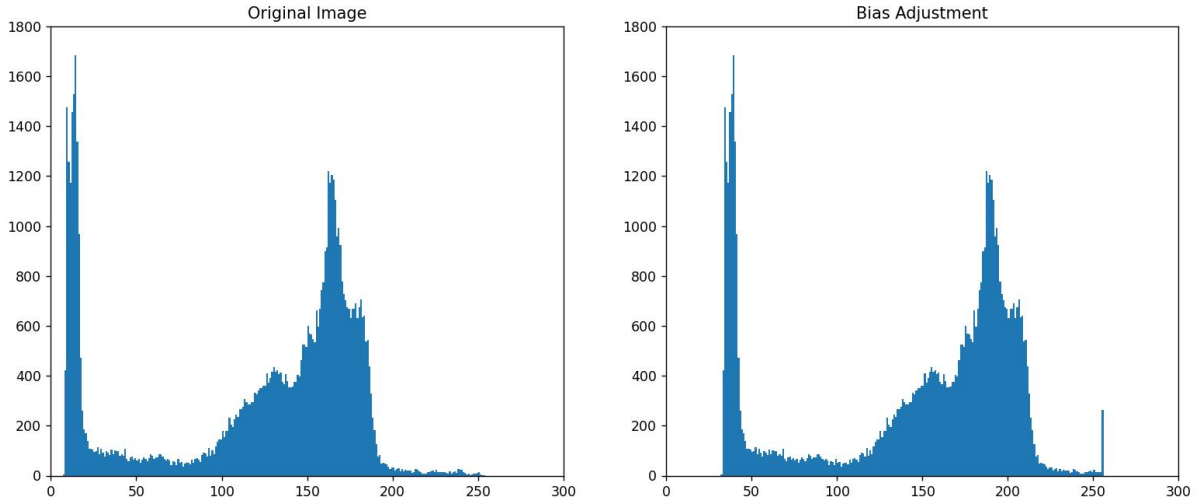

Figure 2

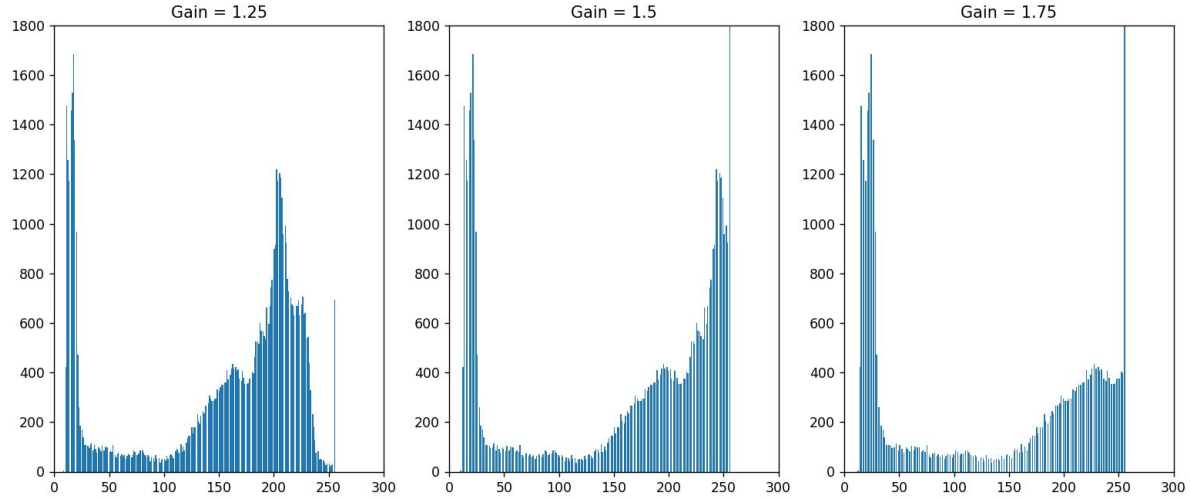

Figure 3

- Produce and display the result of changing the gain and the bias of an image.

4. **Expected Results**

Figure 4: Blue Channel

Figure 5: Green Channel

Figure 6: Red Channel

Figure 7: Using Simple Averaging

Figure 8: Using the Weighted Function

Figure 9: Using OpenCV's Built-In Function

Figure 10: Greenness Value

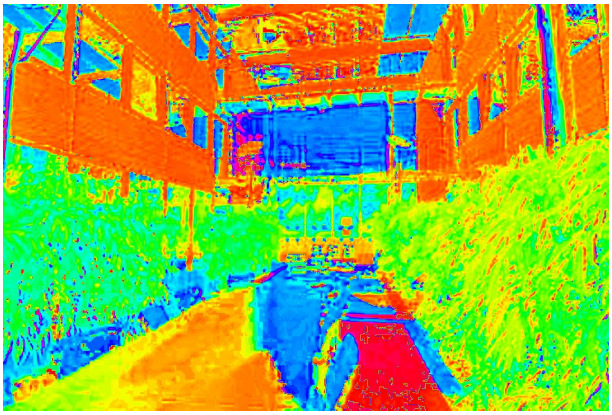

Figure 11: Hue Channel

Figure 12: Saturation Channel

Figure 13: Value Channel

Figure 14: Gamma < 1

Figure 15: Gamma = 1

Figure 16: Gamma > 1

Figure 17: Negated Image

Figure 18: Gain Adjustment

Figure 19: Bias Adjustment