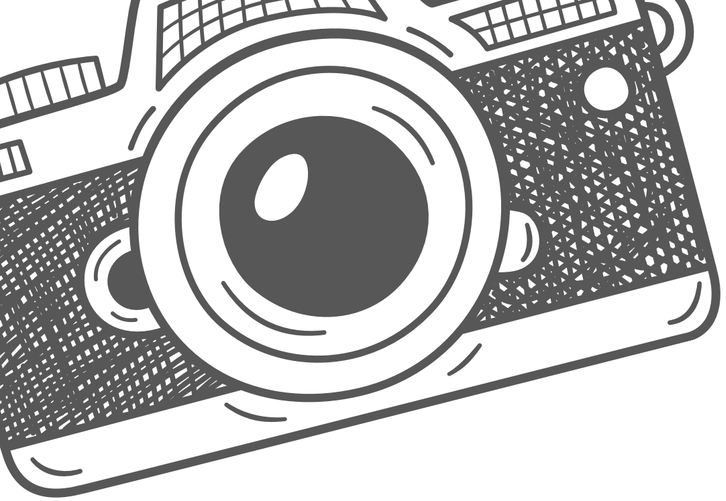


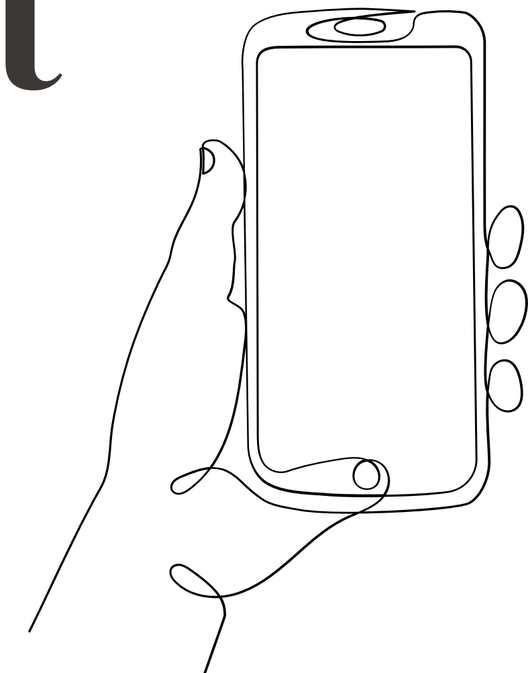
APP PHY 157 WFY-FX-2

LAB REPORT 1

Digital Image Formation and Enhancement

Source code here!

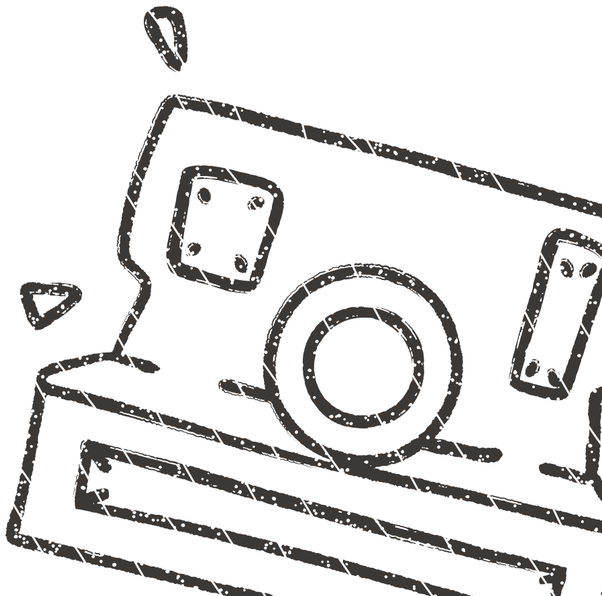
LOVELY L. ANDEO
2020-05405



Objectives

In this day and age, images can be generated, taken, and found almost everywhere. This activity allows us to become more familiar with the processes of dealing with, creating, and enhancing these images - which we can later on use to our advantage.

1. Mathematically create images.
2. Save an image in an appropriate file format.
3. Open and capture images using Python or Matlab
4. Improve the appearance of gray level and color images
5. Use the backprojection technique to transform the histogram of an image to a desired distribution



Results and Analysis

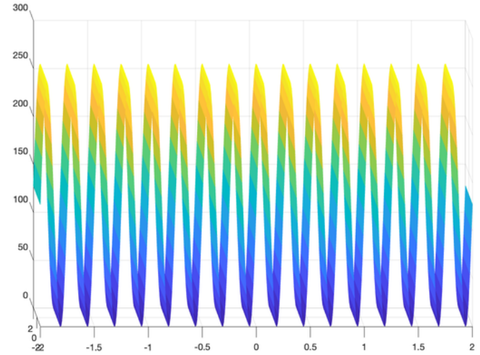


Figure 1.1 Sinusoid

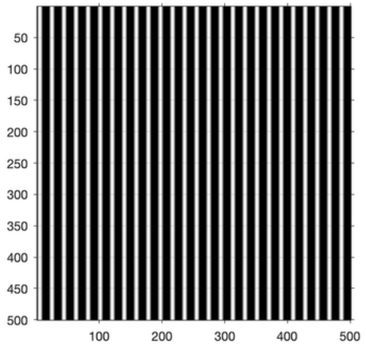


Figure 1.2 Grating

The idea is to create images mathematically. First, I created a sinusoid along the x-direction which has 4 cycles per cm frequency shown in Figure 1 using the meshgrid function. In order to generate the image above, I also had to rescale the axis from 0 to 255. The next figure (2) is grating, where I created a line pair of black (0) and white (1) making sure that there are 5 line pairs pero cm (100 in the figure).

Image DIY

The next step is to recreate the primary mirror of the Hubble Space Telescope (Figure 1.3) which is technically an annulus. I also tried using the ringAnnular function of Matlab to create the same image (Figure 1.4), I found that with a more concise code, it was able to replicate the same thing. Lastly, creating the JWST mirrors was definitely an experience. It was important to take note of the coordinates of the hexagon/s to make sure that the next one you plot fits perfectly the desired shape.

Takeaway: Simulating images allows us to specify parameters creating so much more accurate results rather than just manually drawing them using softwares.

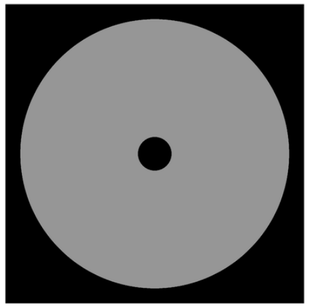


Figure 1.3 Hubble

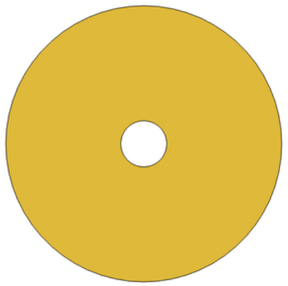


Figure 1.4 Hubble

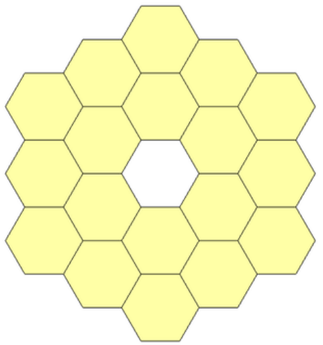


Figure 1.5 JWST

Color Image

In this part, I tried to recreate the Olympic logo. I noticed that it was really important to plot the rings in such a way wherein the ring plotted last is the one on top. Then the colors of the rings were set accordingly to pattern that of the Olympic logo. The result is shown in Figure 2.1.

After the image was made, along with the previous images, it was then saved as JPEG, PNG, BMP, and TIFF, to be used in future activities. What I found is that file size-wise, TIFF > BMP > PNG > JPEG. Although PNGs and JPEGs are commonly used in lots of web-based projects, PNGs are still more often preferred as it is based on lossless compression, unlike JPEG which disregards some of the pixels. But if you have the luxury of not caring about your storage and wanna have the highest quality photo, definitely go for TIFF.

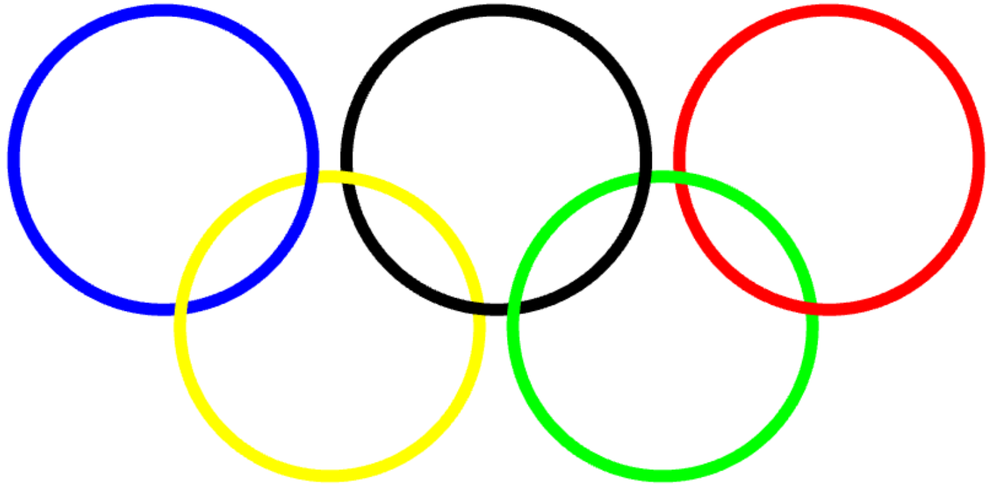


Figure 2.1 Olympic Logo

Takeaway: Images must be saved in a format that is appropriate for its purpose.

Input-Output Curve

Figure 3.1 Original

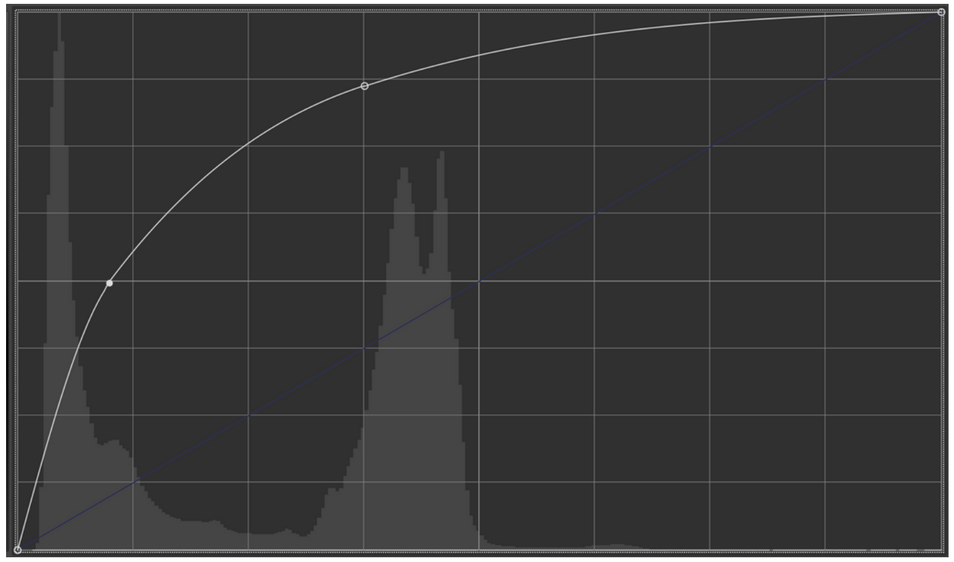


Figure 3.2 Input-output curve

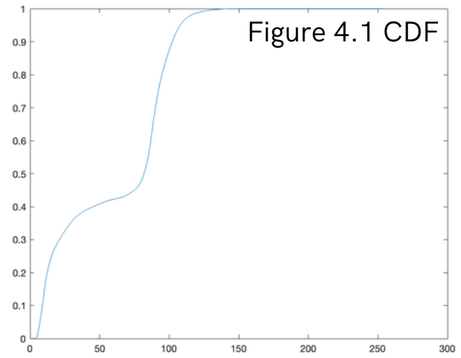
Figure 3.3 Enhanced

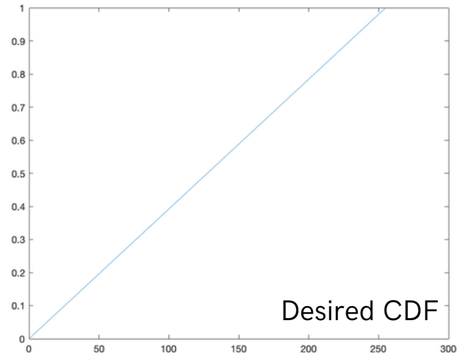
As someone who loves taking pictures, I always find myself thinking '*what if cameras are able to capture this scenery as beautifully as human eyes do*'. Because I feel like cameras are just not able to give justice (most of the time) to how images actually look in real life. As in the example above, figure 3.1 is the photo I took when I first went to UP Diliman as officially a student. The sensors of my camera were only able to pick up the shadows making the image register darker than intended. Using GIMP, I was able to inspect the histogram of the image. We can see that the peaks of the histogram lie on the left side which means that it really is skewed. After playing around with its input and output curve (figure 3.2) to adjust the brightness of the darker parts of the image, figure 3.3 now shows the enhanced version of it, showing more of the trees and plants.

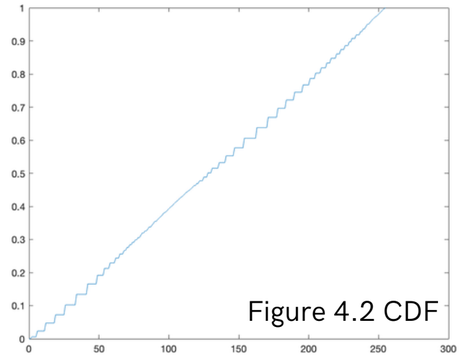
Takeaway: Sometimes, the sensors of our cameras do not perfectly capture the brightness of the object, but it doesn't mean that its details are lost.

Figure 4.1 Dark image

Figure 4.2 Backprojected image







In this activity, I converted the dark image (figure 4.1) into a probability distribution function (PDF) and then used backprojection to modify the values of the grayscale image to get as close as possible to the desired cumulative distribution function (CDF). The steps involve finding the CDF value of each pixel of the grayscale image, then finding its corresponding value on the desired CDF.

As a result of the backprojection, the image becomes that of figure 4.2. It is now much brighter and you can also notice the tiny pixels. What essentially happened was that the dark pixels were replaced by the x-values from the desired PDF.

Takeaway: Backprojection is a great technique that can be used to brighten grayscale images.

Histogram Projection on a Grayscale Image

Contrast Enhancement

Figure 5.1 Contrast strected image

Here, I basically did contrast stretching on the same grayscale image of the picture. To do that, I performed normalization of the values using a certain equation. Comparing the resulting image to that of the original image, the brighter areas became lighter and the darker areas became darker which is expected. The initial percentile value of thie image is 10.

Next, I tried to vary the percentile value of the image into 0, 25, and 50 (same value for the minimum and maximum). The results are shown below. Upon inspection, we can see that the higher the percentile value, the more it looks like the original dark grayscale image. The lower the percentile, the brighter the pixels become.

Takeaway: Finding the appropriate parameters is very important when doing contrast stretching to ensure that the pixels are enhanced and brighten just enough that it's not overpowering.

0 prcntl | 25 prcntl | 50 prcntl

The goal here is to apply different white balancing algorithms to restore the original colors of an unbalanced or faded image. Figure 6.1 shows the results of these algorithms in comparison to the. To execute the contract stretching, I basically had to contrast stretch each color channel such as red, green, and blue (results shown in figure 6.2). Slight differences can be observed since different intensity values were stretched for each color. For the white patch algorithm, the RGB space had to be at its maximum value in order to observe the lightest patch to be used as a reference. In the gray world algorithm, assuming that the world is acromatic, it estimates a reflectance using the average of the RGB values.

Takeaway: For the picture I used, the gray world algorithm seems to have worked the best but depending on the picture, other algorithms might perform better.

Restoring Faded Colored Photographs

Figure 6.1

Original | Contrast stretching in RGB | White Patch Algorithm | Gray World Algorithm

Figure 6.2

Red | Green | Blue

One of the most obvious applications of image enhancement is on social media such as Instagram,Snapchat, etc. which utilize different white balancing algorithms as well to generate image 'filters' that its users can choose from.

Applications

Another application is in remote sensing, especially with satellite images with blurry or low resolution, contrast enhancement is commonly used to enhance such pictures so that the stakeholders or the public can appreciate it more.

Reflection

This activity is really fun and fulfilling since I think I got the expected results for each part. And as a person who likes taking pictures and editing them, I really appreciate getting into the nitty-gritty aspect of creating and enhancing images. I would also like to acknowledge my very helpful classmates for helping me get through using Matlab for the first time and for being patient in debugging some of my codes with me. It was a challenge but definitely a worthy one, as I found Matlab to be very concise, in terms of the codes, compared to Python. I am looking forward to getting more used to it in the next activities.

Self-evaluation

100/100

+ 5 bonus points

I do believe that I really stepped out of my comfort zone for this activity by taking on the challenge of using a language in which I have little to zero background and learning its syntax along the way. I also added variations and additional steps on some parts that were not necessarily required

References

Here are the materials and the code templates I used to accomplish this activity:

Dr. Maricors Soriano on Digital Image Formation and Enhancement. https://uvle.upd.edu.ph/pluginfile.php/834929/mod_resource/content/1/Digital%20Image%20Formation%20and%20Enhancement.pdf

Brush up on the Difference between Image File Formats. https://creativemarket.com/blog/difference-between-jpg-png-bmp-tiff-images

Loren Shure sample code. https://creativemarket.com/blog/difference-between-jpg-png-bmp-tiff-images

Juan on Color Constancy Algorithms. https://www.mathworks.com/matlabcentral/fileexchange/41341-color-constancy-algorithms-gray-world-white-patch-modified-white-patch-etc