Activity 4: Measuring Area from Images

Mary Chris Roperos Go / 2014 - 11122

August 20, 2019

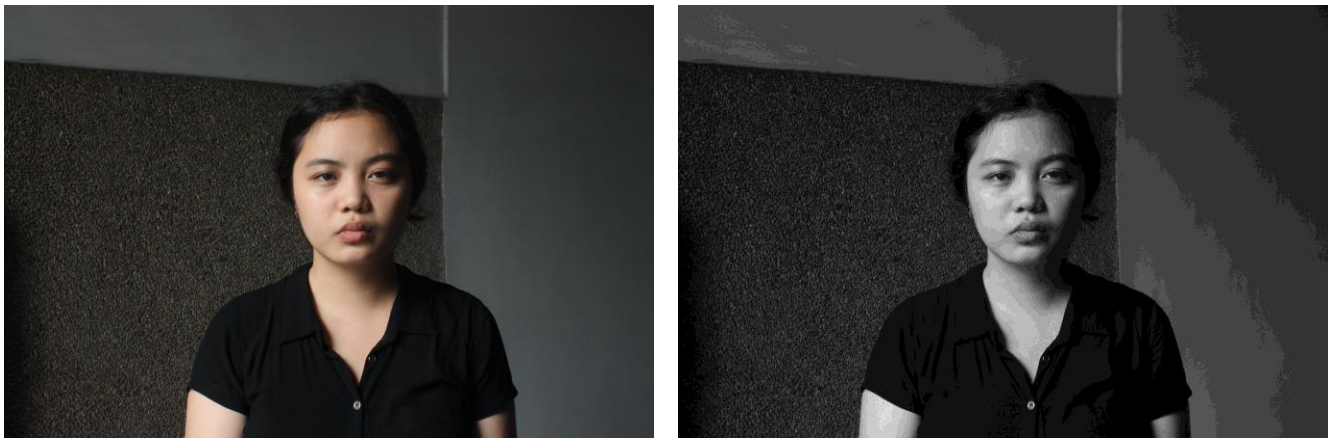

Figure 1: The original image (left) and grayscale original image using Scilab (right).

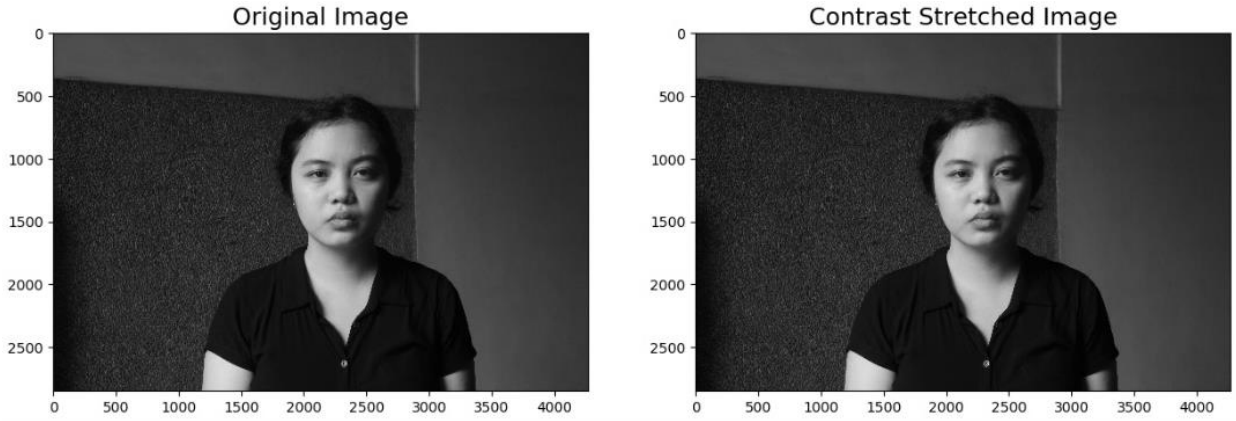

Figure 2: Original image in grayscale mode (left) and contrast stretched image using Scilab (right).

Opening of Image

The image chosen was taken during the first activity for AP 186: Ratio and Proportion since I only have limited pictures in my laptop (Sorry, Krishna!). The first part of the activity was the basic opening of image. I used Scilab with the help of SIVP package which was helpful in image processing activities. Figure 1 showed the usage of **rgb2gray** to convert the rgb-image to grayscale.

Contrast Stretching

For now, we consider figure 2 (left) as the original image for the remaining parts of the activity. In figure 2, I implemented contrast stretching also known as normalization. I simply 'stretched' the range of intensity values. In my observation, the contrast stretched image became lesser intense as compared to the original. I observed the clarity I saw on the original image.

Histogram and CDF of Original Image

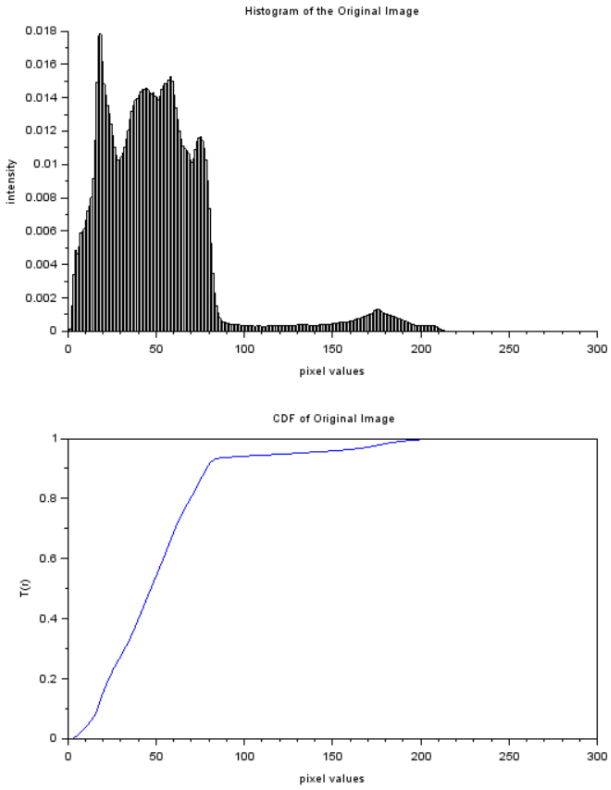

Figure 3: The obtained grayscale histogram and (Cumulative Distribution Function) CDF of the original image.

Figure 3 (top) showed the grayscale histogram using **imhist**. From its PDF, the CDF was obtained. As observed, the histogram has three pronounced peaks. It also showed not well spread values and concentrated towards the black portion. For figure 4 (bottom), the corresponding CDF was generated using **cumsum()** function. The values for vertical values was for the percentage of the overall area of a histogram cover a pixel value. From the graph, there was a positive slope. The peaks in the histogram ranged around 30 – 80 nm. If you noticed, there was a drastic change in CDF on these values. The transition started to appear flat at 100 nm which is expected from the histogram.

Generating various CDFs

```
//desired linear CDF
scf(1);
x = [0:255];
des_CDF = x/255.0;
subplot(311);
xlabel('pixel values');
ylabel('T(r)');
plot(des_CDF);
xtitle("Desired CDF");
//desired quadratic CDF
scf(1);
des_2CDF = x.^2/(255.0^2);
subplot(312);
xlabel('pixel values');
ylabel('T(r)');
plot(des_2CDF);
xtitle("Desired cumulative distribution function (quadratic)");
//desired erf CDF
scf(1);
rangel = linspace(-2,2,256);
des_3CDF = 0.5*(erf(rangel)+1);
subplot(313);
xlabel('pixel values');
ylabel('T(r)');
plot(des_3CDF);
xtitle("Desired CDF (erf)")
```

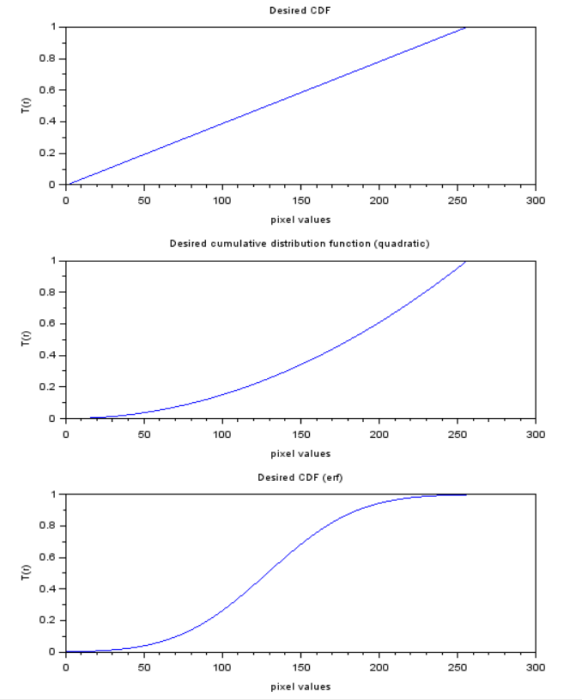

Figure 4: Different CDFs: Linear, quadratic, sigmoid of erf function. These functions were implemented using scilab (left).

A linear, quadratic, and sigmoid via erf function were generated in Scilab. These were all increasing curves. These CDFs were used to produce an actual image.

Backprojecting Technique

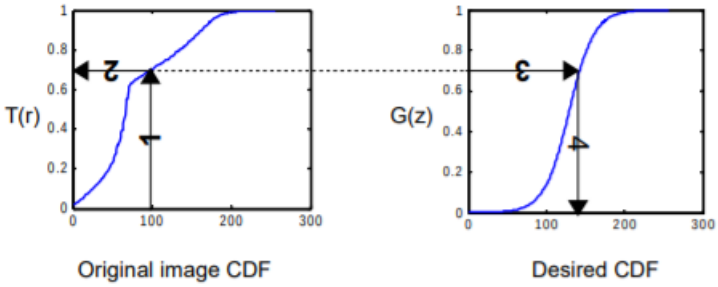

Figure 5: Mechanism behind backprojection (1).

Using technique in figure 5, I extracted the new image from the generated CDF. The original image was retained and used as a reference. A copy of an image was made and was altered for back projecting every pixel value. Pixels with the same shade were located. This value was

mapped to the CDF that was desired getting its corresponding value. The new pixel value will replace the former value. Thus, generating the new image.

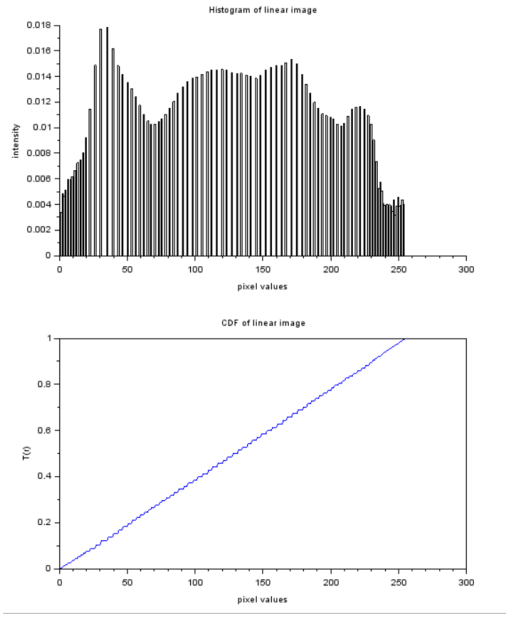

Figure 6: The modified image produced by a linear function CDF.

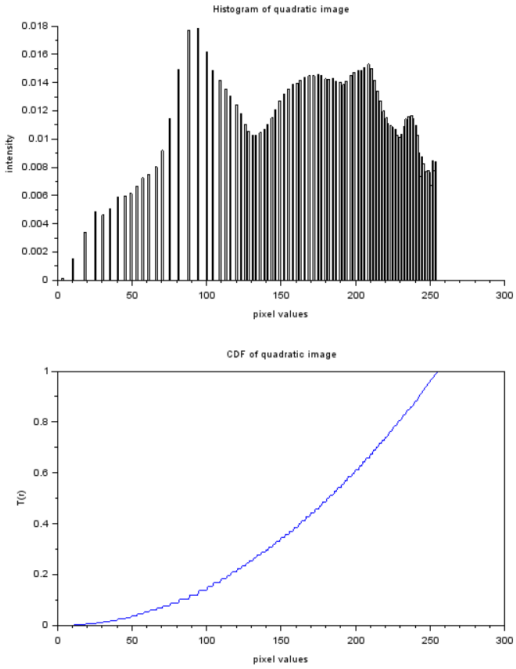

Figure 7: The modified image produced by a quadratic function CDF.

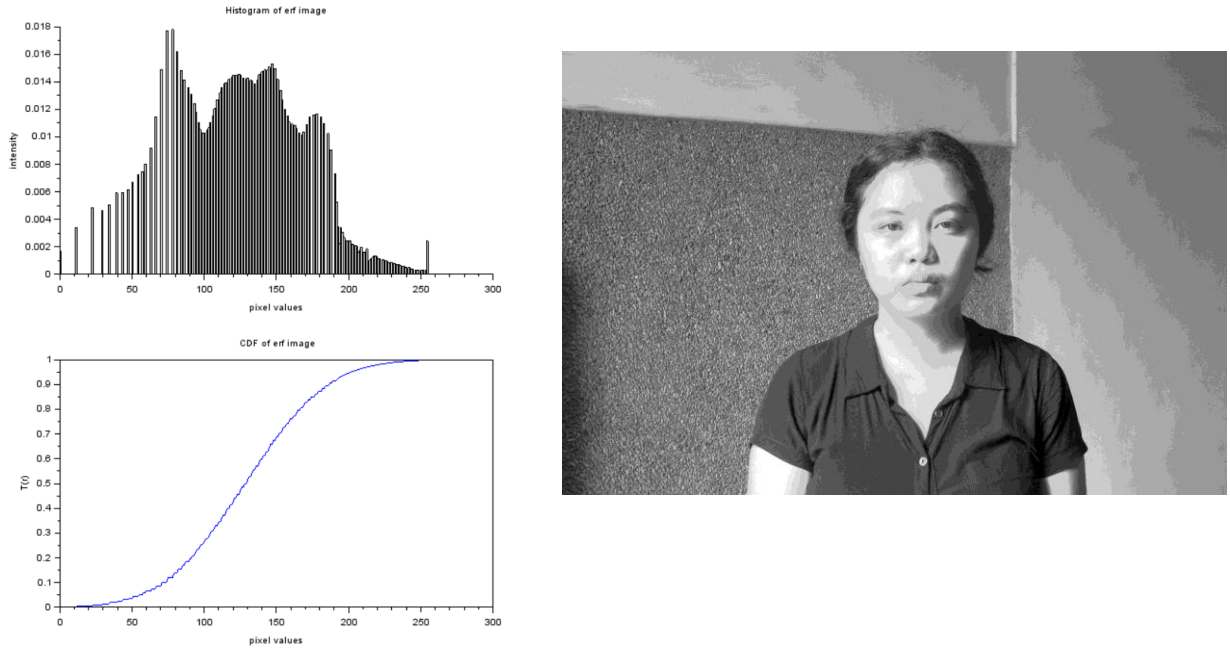

Figure 8: The modified image produced by sigmoid generated via erf function.

The first one was the linear function as seen in figure 6. This function was the ideal to enhance the image. But the chosen picture was already light, so doing this saturated the white pixels of the image. The second one was a quadratic function show in figure 7. It brightened the image since we observed a slow accumulation of dark pixels. In figure 8, a sigmoid via erf function was generated. This nonlinear shape suited a nonlinear response. Looking at the histogram, it converted the PDF to somehow concentrate into the center. Although from the picture generated, it did not suit well. This output quality might depend on the quality of the input image. Since I am only imitating what a human eye vision can see, the image shown is just an approximation.

Initiative – Manipulation via GIMP

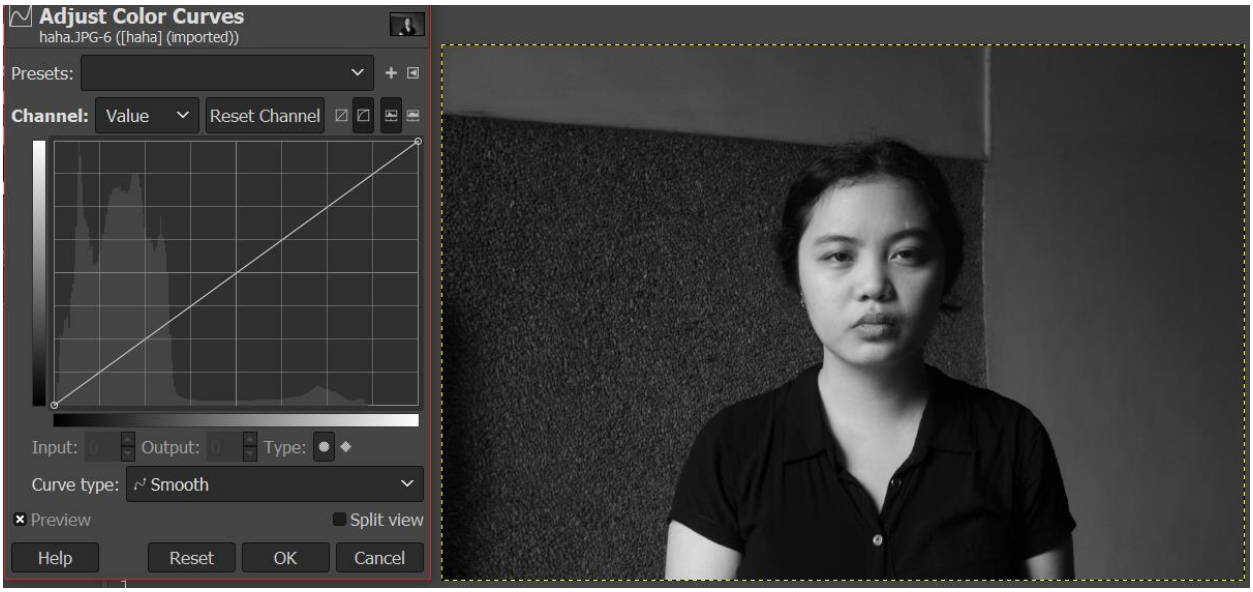

Figure 9 : Enhanced image (linear) of IO curved manipulation via Gimp.

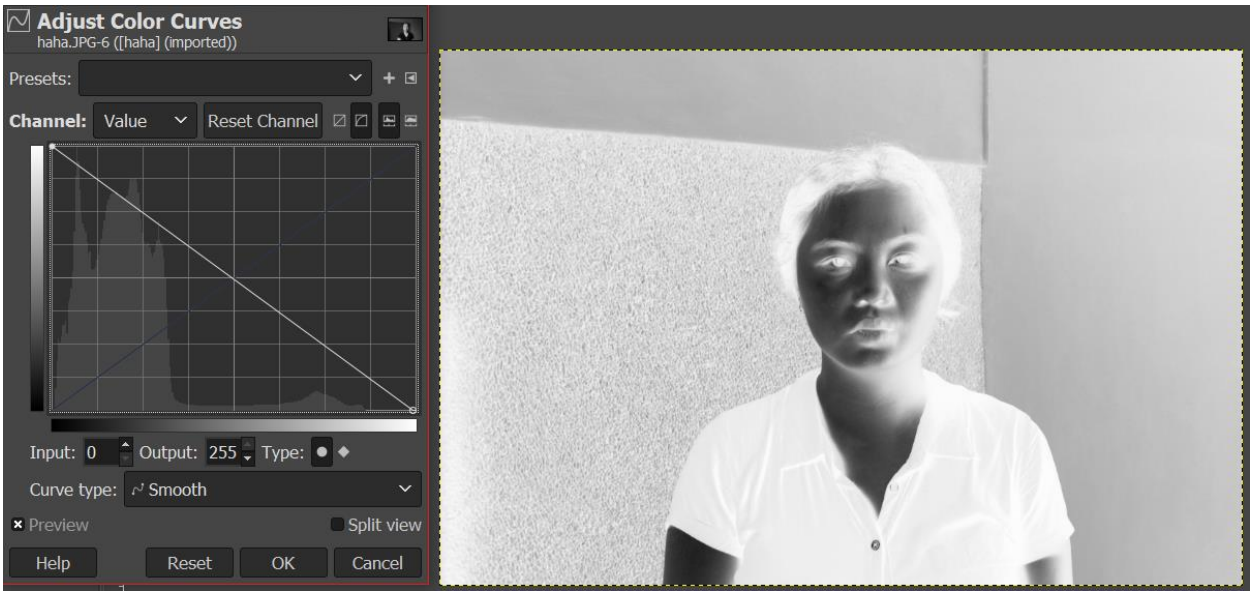

Figure 10: Enhanced image (mirrored pixel values) of IO curved manipulation via Gimp.

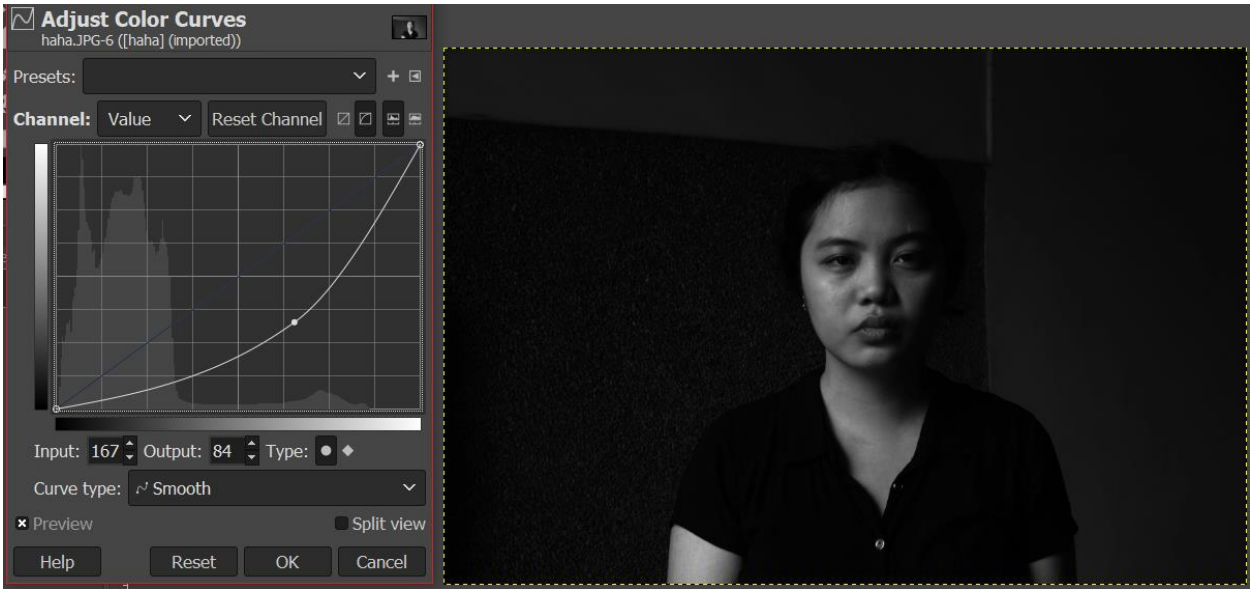

Figure 11: Enhanced image (upward concavity) of IO curved manipulation via Gimp.

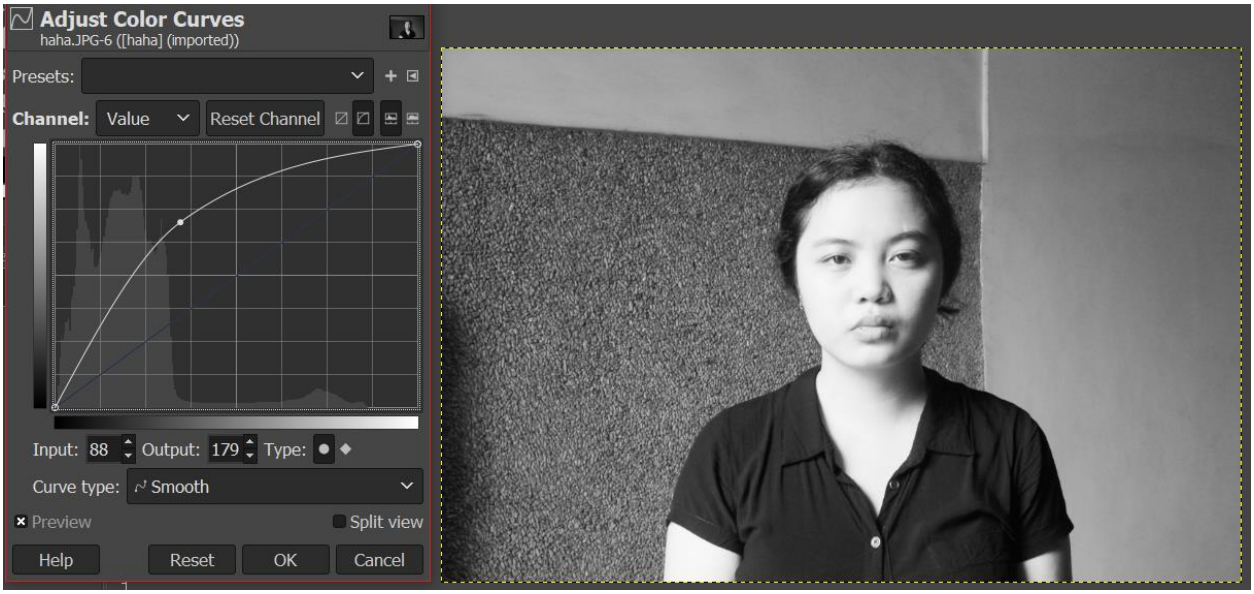

Figure 12: Enhanced image (downward cavity) of IO curved manipulation via Gimp.

Figures 9 – 12 showed various kinds of manipulation for the chosen photos. But instead of using Scilab, the following changes were done using an image processing tool called GIMP. The linear just showed the better enhancement of the image, inversing the line showed mirrored pixel values where dark pixels were replaced with bright pixel shades, producing a ‘weird’ image. For the concavity, the upward concavity (figure 11), darkens the image since the pixel values were placed towards the darker pixel. The opposite phenomenon happens for the downward concavity.

All in all, this was the most fun activity so far. Although the coding part was the challenging, specifically back projection. But once you get the gist of it, the rest will be easy and enjoying.

Reference:

1. M. Soriano – Activity 5: Enhancement by Histogram Manipulation, 2019

Self-evaluation

Technical correctness	5
Quality of presentation	5
Initiative	2
Total	12