Fundamentos de Procesamiento de Imágenes
Pontificia Univerisidad Católica de Chile
April 2024

Assigment 2

Daniel Martínez Castro
daniel.martinezc@usm.cl

For this task, we are requested to process multiple images to enhance their quality, identify the background, and detect edges. All of these tasks will be accomplished using concepts covered in the course, such as histogram equalization, filters, masks, among others. We will then conclude with an analysis of the resulting images. To achieve this, I will use the Python 3 programming language with the assistance of the OpenCV package.

1. "Waterfall.tif" Image.

   (a) Figure 1 shows the image "waterfall.tif" without any processing, along with its histogram. As can be seen in the histogram, the intensity distribution is concentrated in the low intensities, making the image appear somewhat dark. Equalization achieves a redistribution of these intensities to a more uniform distribution, thus increasing the contrast, which can make objects in the image, such as the sides of the waterfall, more visible.

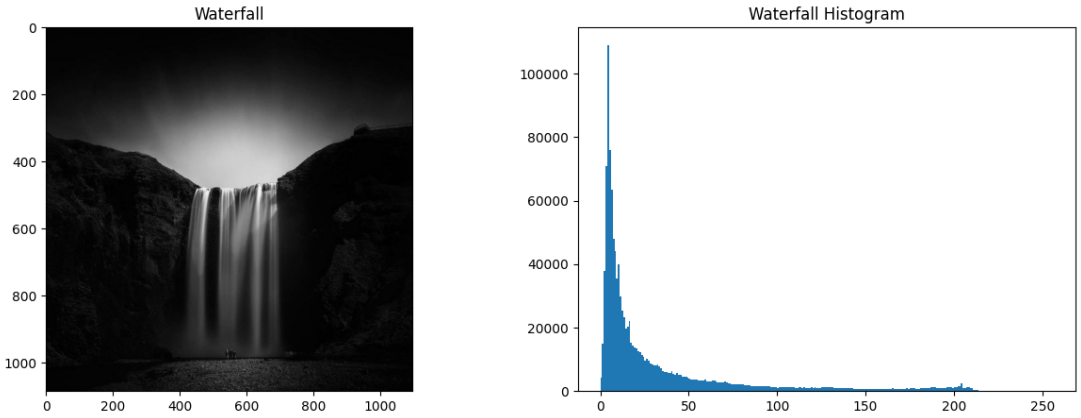

Figure 1: Original Waterfall Image and its Histogram

   (b) Figure 2 shows the image with histogram equalization applied alongside the new histogram. It can be observed how histogram equalization managed to enhance the contrast in the image, improving the visibility of areas that were previously hard to discern. However, a slight saturation can be noticed, especially in the sky and water of the image.

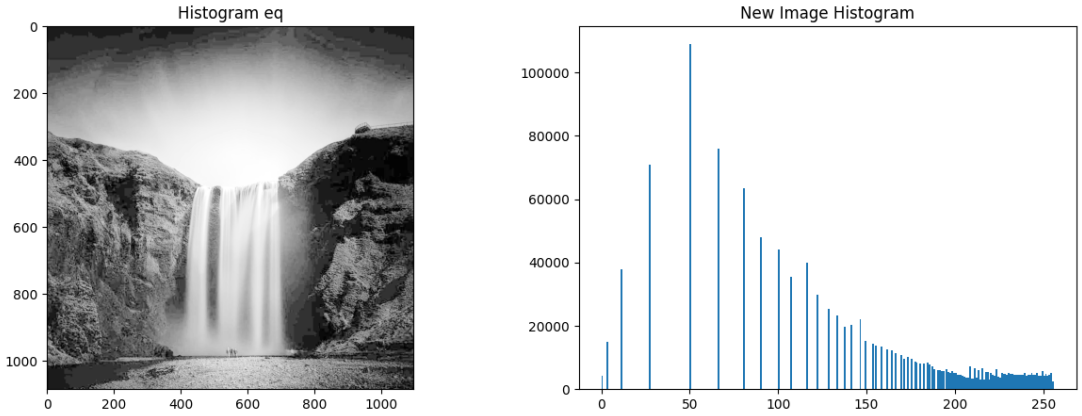

Figure 2: Equalized Image and its Histogram

(c) In Figure 3, the image subjected to linearly decreasing histogram specification. It can be observed that the new histogram's intensity distribution resembles a linearly decreasing histogram. In terms of the resulting image quality, saturation is noticeable throughout the image. Therefore, I consider this transformation of little utility for this specific case.

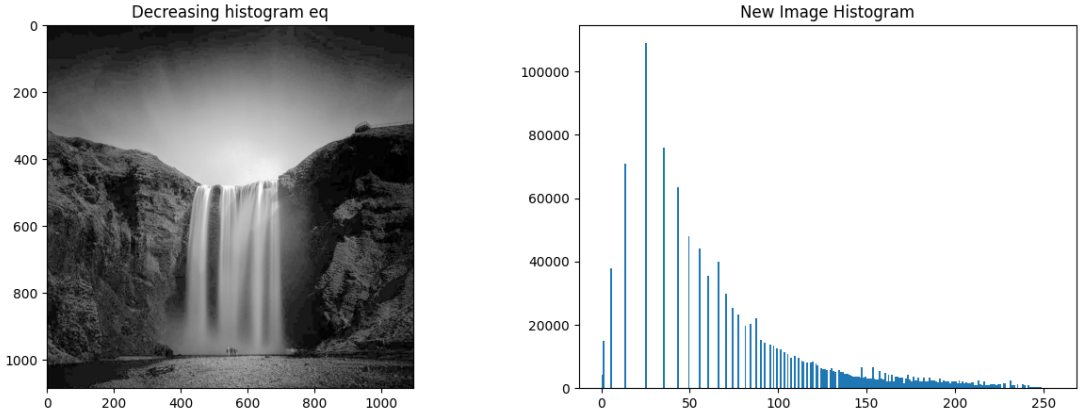

Figure 3: Linearly Decreasing Histogram Equalized Image

In Figure 4, the image undergoing linearly increasing histogram specification. As seen in the histogram, this specification inclines intensity values towards the lighter ones due to the nature of the linearly ascending histogram. Regarding the image quality, it saturates heavily, losing the original overall distribution of the image. Therefore, this transformation is practically useless.

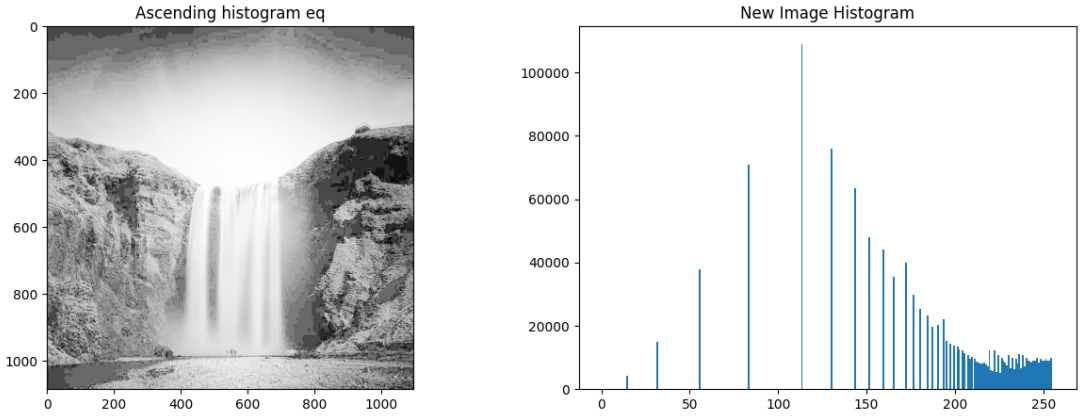

Figure 4: Linearly Increasing Histogram Equalized Image

(d) Figure 5 displays the waterfall image to which a localized histogram transformation was applied, specifically CLAHE (Contrast Limited Adaptive Histogram Equalization) with a clip_limit parameter of 2. This parameter was chosen to make the Python CLAHE function behave like AHE (Adaptive Histogram Equalization).

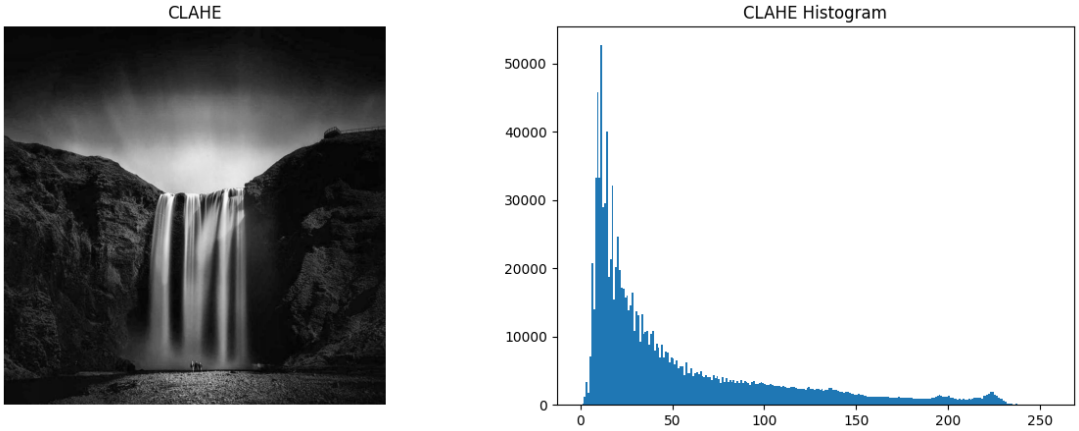

Figure 5: CLAHE (Contrast Limited Adaptive Histogram Equalization) Image

(e) In Figure 6, all the resulting images of the waterfall from the transformations conducted earlier are displayed. In my opinion, the image that underwent CLAHE has the best outcome since it appears balanced without saturating specific areas of the image while maintaining visibility in darker areas.

Figure 6: Comparation of All Images

2. "Car.jpg" Image.

(a) In Figure 7, the original image "car.jpg" is displayed along with its RGB channels. In each channel, the background appears similar, while the car varies in intensity across the different channels. This is because, as the car is blue, the blue channel contains the most information about the car, followed by the green channel with some information, and the red channel with little to no information about the car. By leveraging the blue channel, a mask could be applied to the other channels, thus isolating the background.

Figure 7: Image of the Car and its Color Channels

(b) To remove the background from the blue channel, a threshold at 150 was applied to set values below it to zero, while values equal to or above remained unchanged, as depicted in the transfer curve (Figure 9). This successfully retained the car in the image while eliminating the background. It's worth noting that using this method, it's not possible to remove the sky since this channel contains significant information about the sky's color.

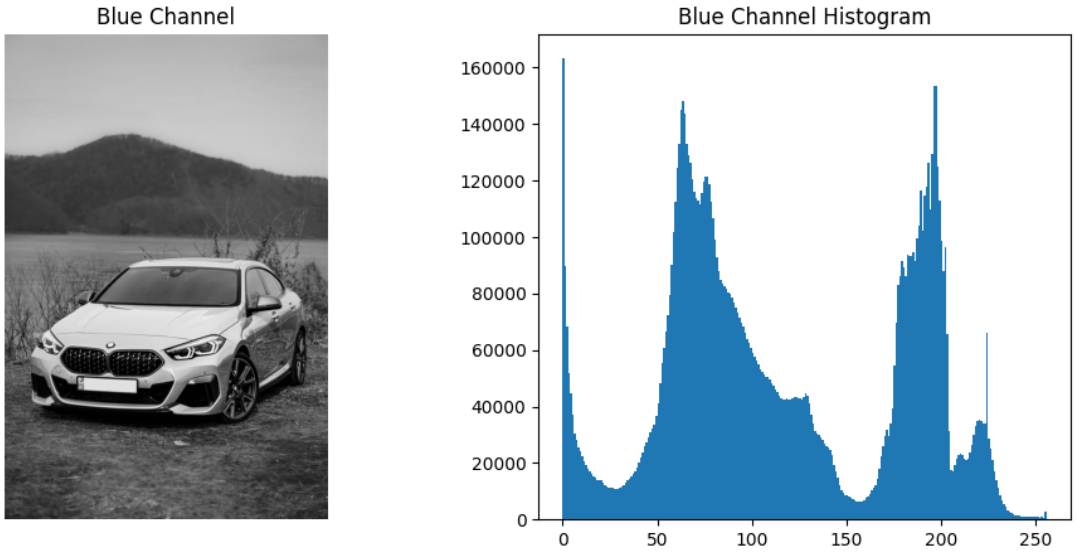

Figure 8: Blue Channel ans Its Histogram

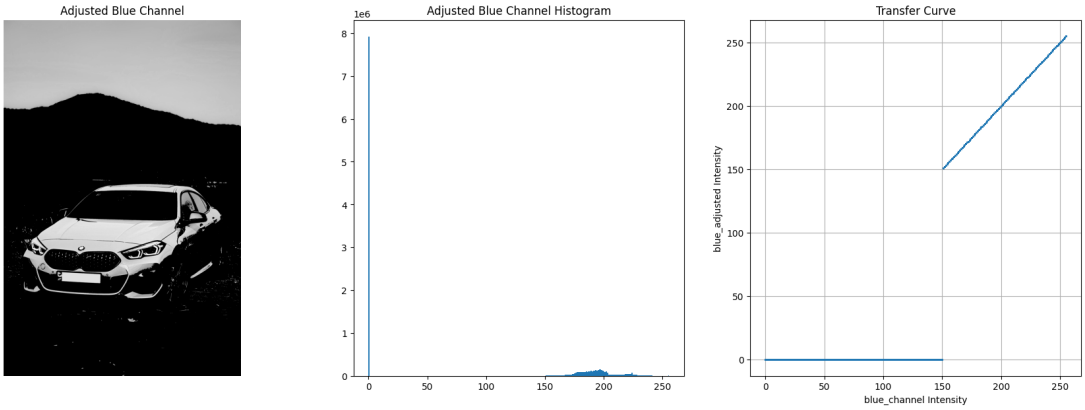

Figure 9: Intensity Adjusted Image its Histogram and Transference Curve

(c) To convert only the background to black and white, the first step was to apply a binary threshold at 180 to isolate the background from the car (Figure 10, Im. 1). Then, the inverse of this image was obtained to use it as a mask for the background (Figure 10, Im. 2). This mask was applied to the original image, which was then converted to black and white (Figure 10, Im. 3). Using the binary threshold image as a mask, the isolated car was obtained (Figure 10, Im. 4). Finally, the two images were blended together, resulting in the final image with the blue car and the background in black and white (Figure 10, Im. 5).

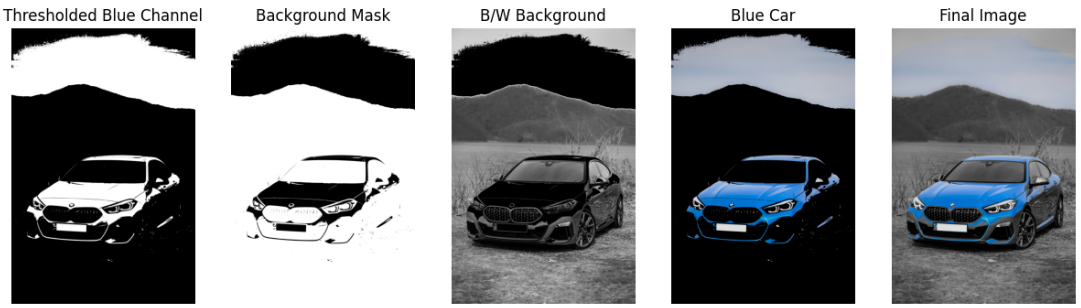

Figure 10: Operations Performed to keep the car blue and the background B/W

3. "KneePD.jpg" Image.

(a & b) In Figure 11, several intensity transformations were performed on the knee image. For the piecewise linear transformation, a binary threshold of 30 was applied. Although this transformation removed the defect from the image, it also resulted in a considerable loss of information. Additionally, histogram equalization was carried out, which, while adjusting intensities more uniformly, saturated the image in the area of the defect and lost crucial information for medical diagnosis. Finally, adaptive local CLAHE equalization was applied, with a clip_limit parameter set to 1. This last transformation slightly improved the area with the defect without saturating other areas. In my opinion, this was the transformation that yielded the best results.

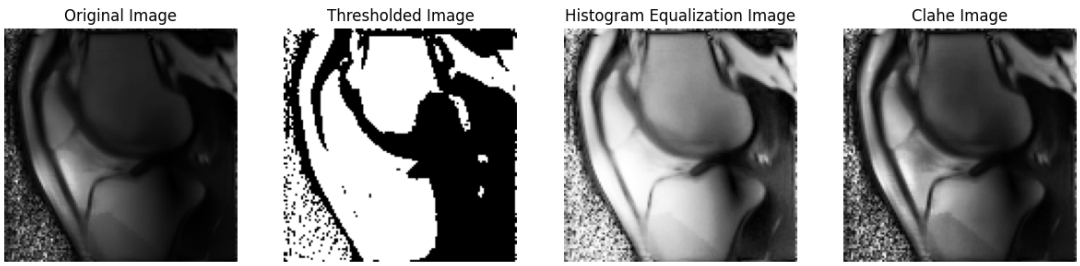

Figure 11: Resulting Images of the different Transformations Performed in the Knee Image

(b) Figure 12 displays the histograms of the transformations conducted earlier. For the histogram of the original image, it can be observed that it is skewed towards darker color intensities. On the other hand, for the histogram of the image subjected to thresholding, it can be seen that the only values on the x-axis of the histogram are 0 and 1. This is because binary thresholding was applied, and it can also be noted that the quantity of ones and zeros is relatively even. In the case of the histogram of the image that underwent histogram equalization, it can be observed that the distribution was uniformly adjusted to enhance the contrast. Finally, for the histogram of CLAHE, the general distribution resembles that of the original image but is stretched to improve the contrast without saturating the image.

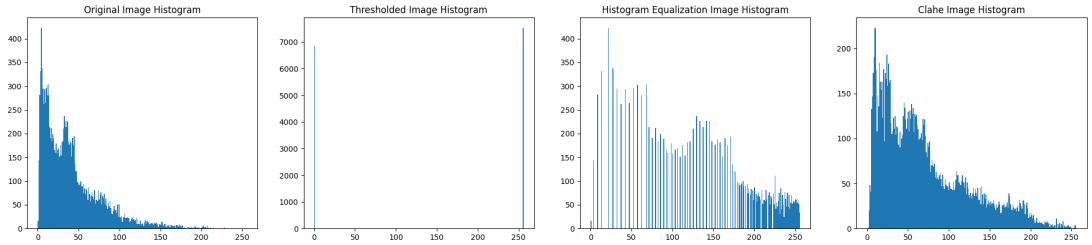

Figure 12: Histograms of the Different Transformations Performed in the Knee Image

(c) For this item, it is proposed to fine-tune the hyperparameter clip_limit in the OpenCV CLAHE function in order to behave like an AHE. To achieve this, the values 1, 2, 5, 10, 20, 40, 80, and 160 were utilized. For the values of 1 and 2, no improvements were observed compared to the previous one, as the values are close to those used previously. In the cases of 5 and 10, it can be observed that while the visualization of the right side of the image, which was previously very dark, is improved, some saturation is introduced in the area with the defect and the center of the image. For clip_limit higher than 20, they are similar to each other, and only a decrease in image quality and extreme saturation in intensity are observed. In my opinion, the clip_limit parameter at 10 improved the visualization of the image as previously invisible structures can now be seen without excessively compromising the image quality.

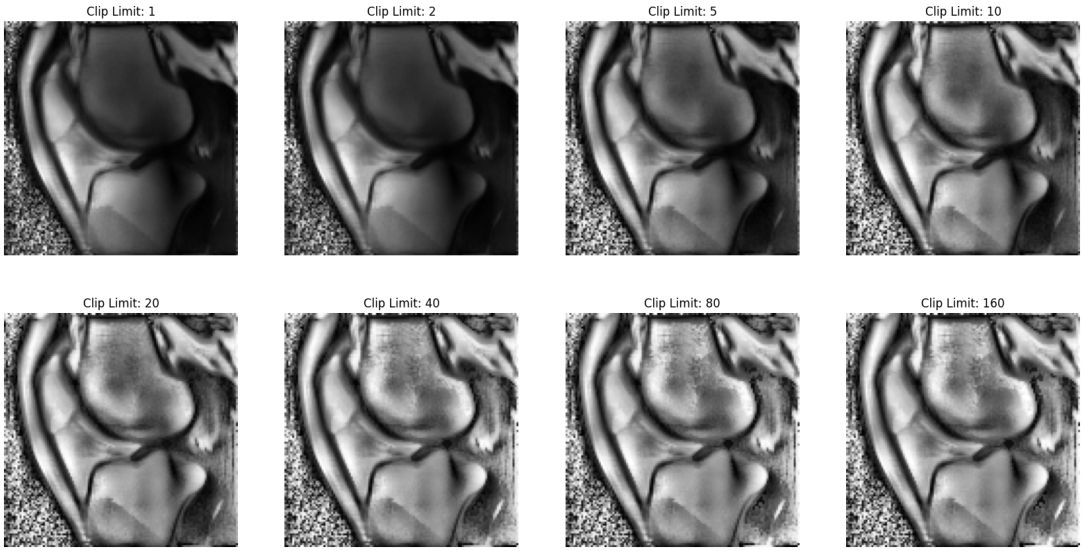

Figure 13: Hyperparameter Tuning (Clip Limit) CLAHE

4. "Star.jpg" Image

   (a) In this item, semi-automated segmentation of a star in the file "star.jpg" (Figure 14) is requested. To accomplish this, I first separated the image into its RGB channels (Figure 15) to identify the most useful channel. I chose the blue channel because it has the least amount of noise in the star. Then, a binary threshold at 210 was applied to eliminate the background, and the image size was reduced while maintaining the aspect ratio so that the width of the image is 510 pixels, aiming to avoid the need to resize kernels for filters to function properly in future steps (Figure 16). Next, a Gaussian blur with a 5x5 kernel and sigma 3 was applied to reduce noise in the star. Finally, the Canny algorithm with threshold_1 of 300 and threshold_2 of 365 from OpenCV was used, which utilizes Sobel filters among other techniques to detect the edges of the star, resulting in the displayed outcome shown in Figure 17.

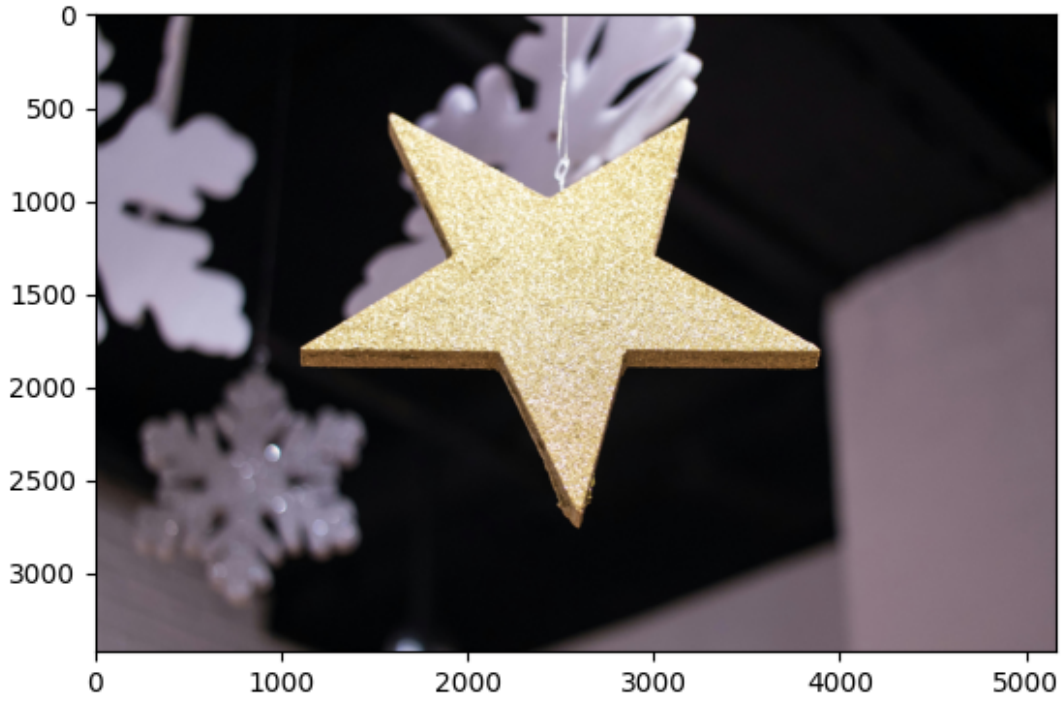

Figure 14: Original Star Image

Figure 15: RGB Channels of "star.jpg"

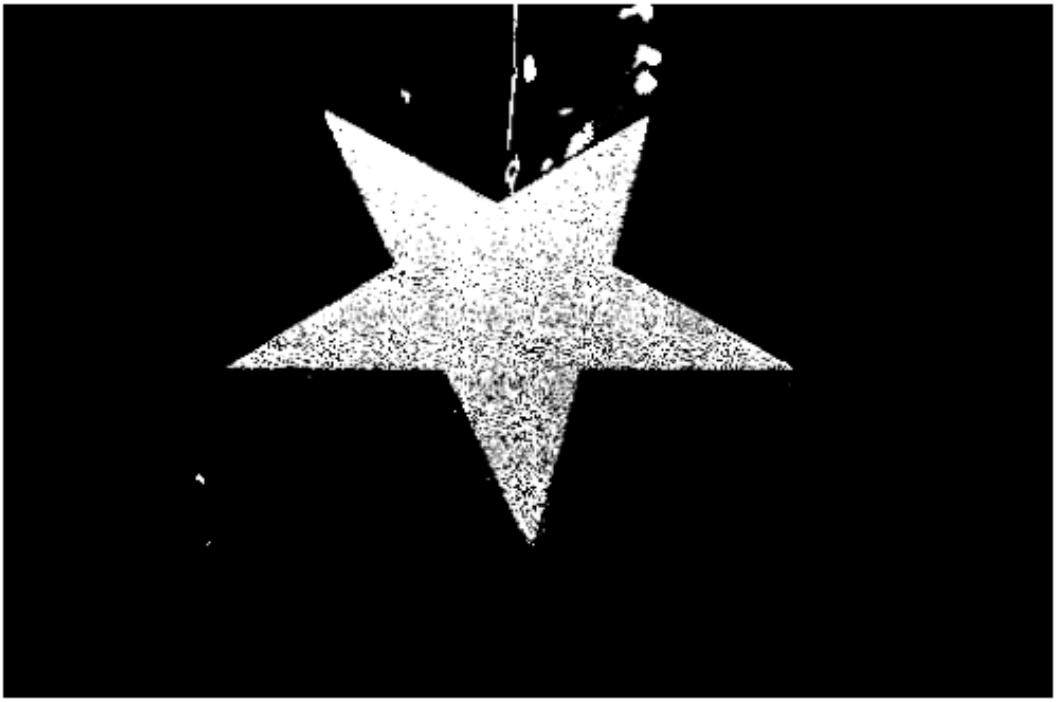

Figure 16: Processed Blue Channel of the Star Image

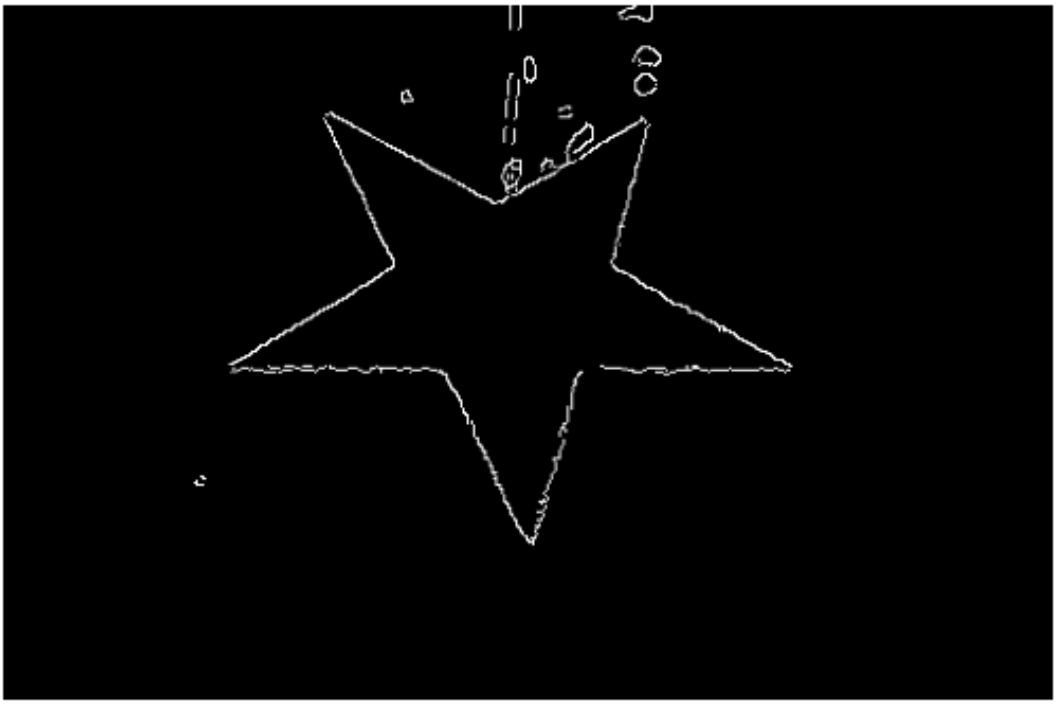

Figure 17: Image of the Detected Borders