Experiment 1

Mücahit Veli Cumart, 21605893

BBM 415

Fundamentals of Image Processing Lab

Hacettepe University

b21605893@cs.hacettepe.edu.tr

Introduction

In this experiment, There are two parts of tasks us to complete. Part1 is about quantization and dithering of an image with different values. Quantization is a discretization method which can cause errors due to limited intensity resolution. An image can lose a lot of visual content when this image quantized. Floyd-Steinberg dithering is an image dithering algorithm .This algorithm achieves dithering using error diffusion. It adds the residual quantization error of a pixel onto its neighboring pixels, to be dealt with later.As a result, we are expected to apply quantization algorithm and Floyd-Steinberg Dithering algorithm to a grayscale image and interpret the results.

Experiment

Part 1

In this part , there are two tasks to implement. Firstly, we are reading a grayscale image for quantization and dithering. After reading the image , we apply quantization to this image with the function we created. Finally, we are reading the image again ,original image, for applying the Floyd-Steinberg Dithering algorithm. To achieve this , we are calling the function which is for Floyd-Steinberg Dithering algorithm.

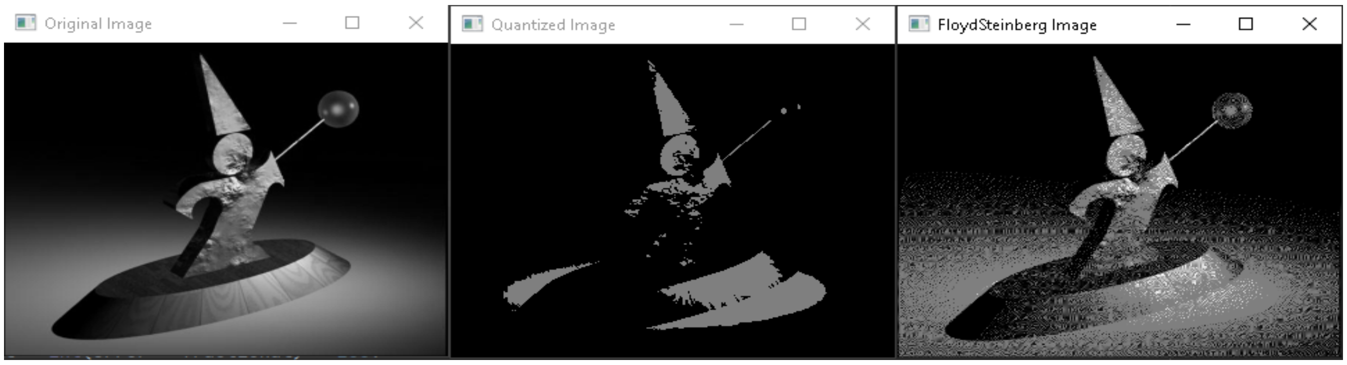
There is a parameter, Q , for the functions of quantization and Floyd-Steinberg. The results in these functions are calculated according to this parameter. The quantization function is changing bit level of an image with the given parameter Q. When the bit level of an image is decreased, the quality of visual content deteriorates and the quantization limits the intensity resolution. The Floyd-Steinberg dithering function calculates the quantized value of the pixels, after this calculatin, with using error diffusion, adds the quantization error onto the neighbour pixels. In this algorithm scans the pixels and error of quantization transferred to the neighbour pixels. That's why the rounding pixels downwards and upwards is nearly same. With this implementation, visual content quality degradation is minimized.

For example, when a pixel of an image is changed by the Floyd-Steinberg Algorithm, the quantized errors which is calculated with numbers in different ratios are added to the neighbour pixels.If the quantized pixel is rounded upwards, the neighbour pixels can be rounded downwards or vice versa. With this situation, the quantizaton error is close to zero in general.

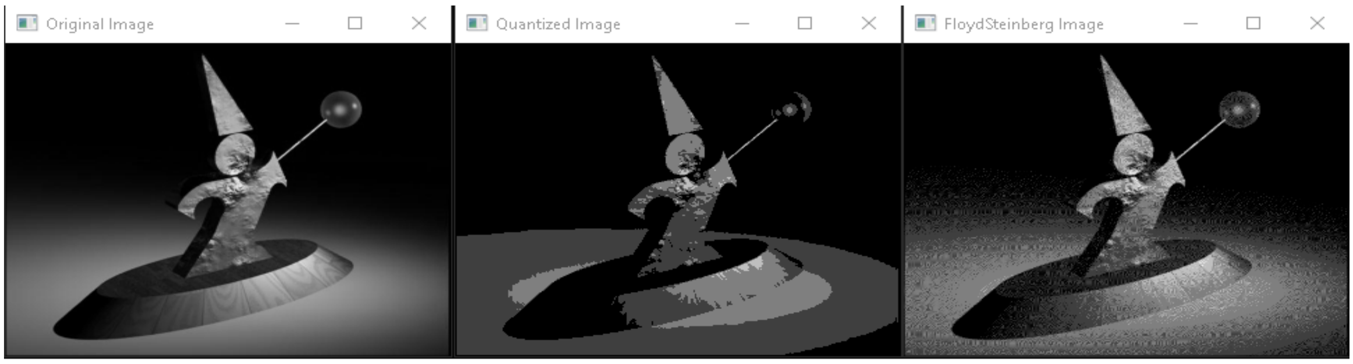
I implemented the given tasks, quantized and dithered a few pictures with different Q parameters. We can see the examples of these trials below.

Different Q Parameters for same image examples below.

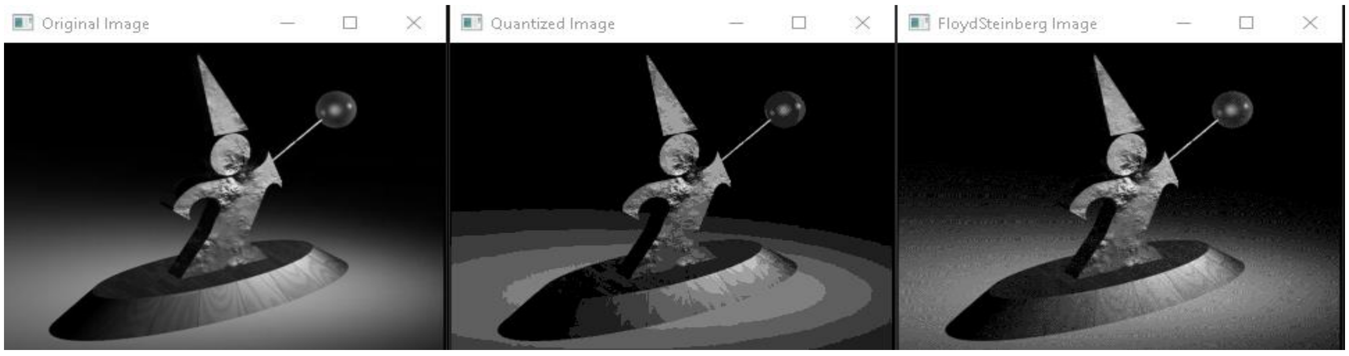
Q = 2



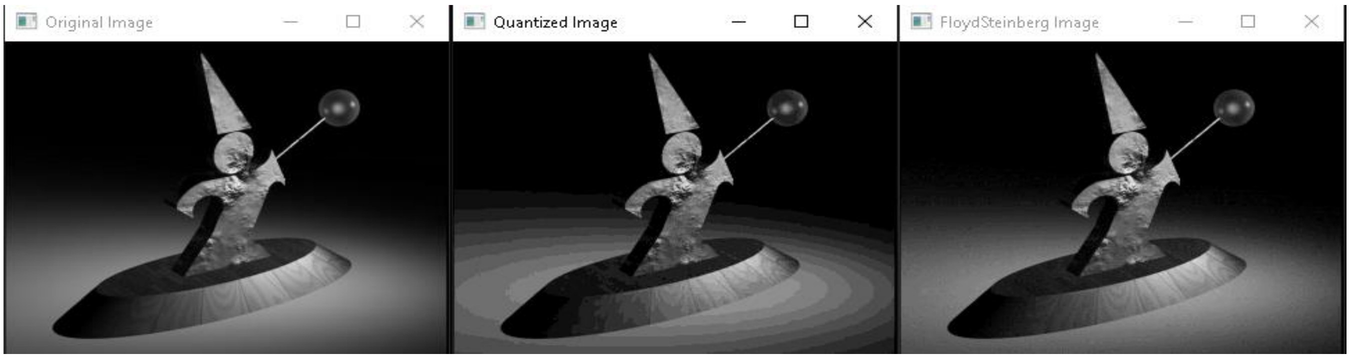
Q = 4



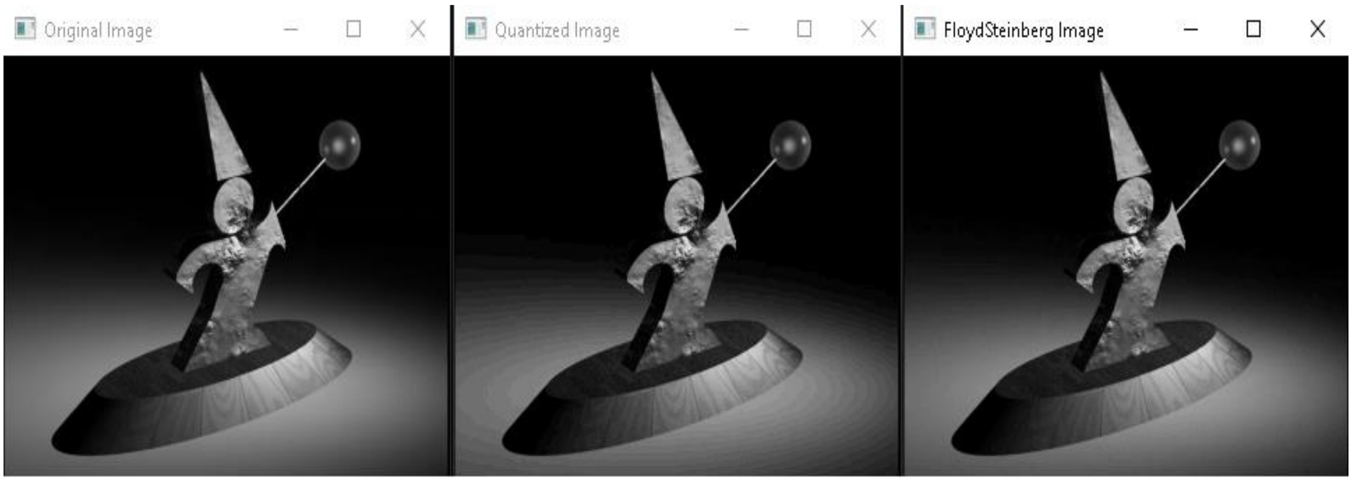
Q = 8



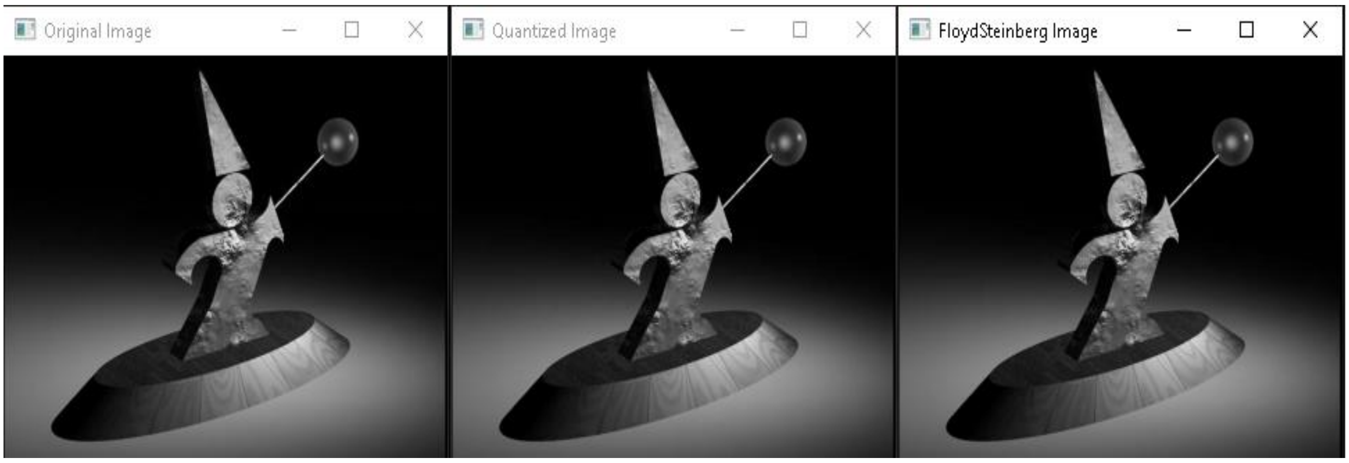
Q = 16


Original Image
Quantized Image
FloydSteinberg Image

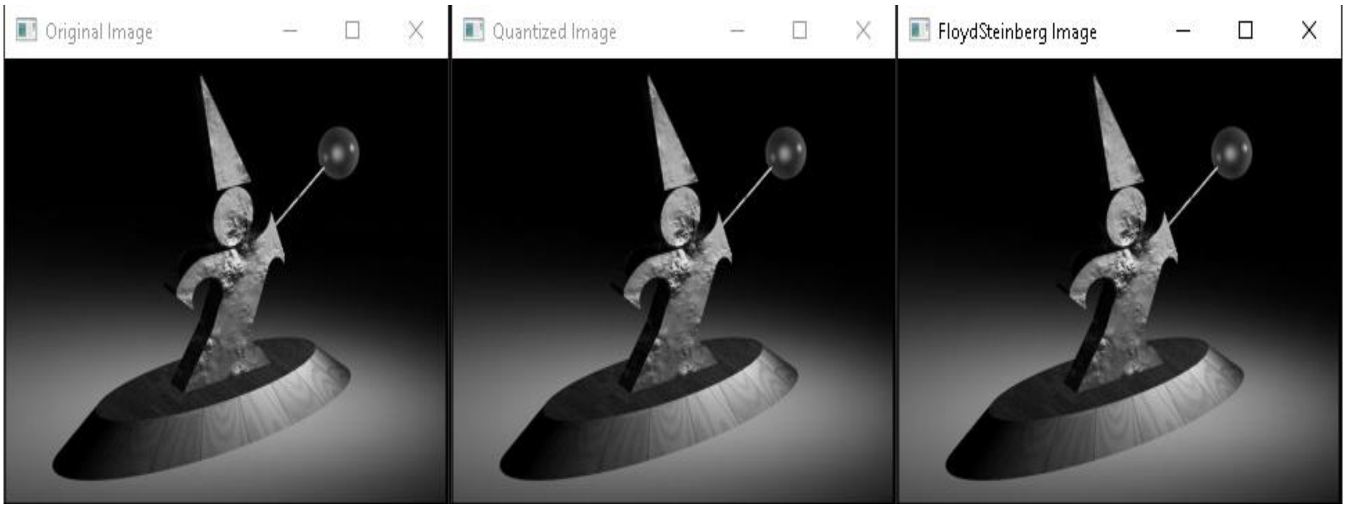
Q = 32


Original Image
Quantized Image
FloydSteinberg Image

Q = 128


Original Image
Quantized Image
FloydSteinberg Image

Q = 256



Behaviour of the Floyd-Steinberg Algorithm is changing when the q parameter changed. Firstly, I give 2 to the Q parameter. Because of the implementation of the algorithm, the number of each pixels rounded to the quantized value of this pixel and the calculated error added to the new value. When the image is quantized with value 2(q), the bit level is decreasing two. With this bit level, the visual of the image becomes black and white. After the quantization, the Floyd-Steinberg Algortihm is applied to the image and we see that the quantized image improved and the quality is increased.

When we increased the value of Q parameter to 4, the bit level of quantized image changed and we can see that four gray level of color on the image. Also, after the Floyd-Steinberg dithering applied to the image with q which is increased to 4, the error of image is decreased compared to the q parameter which is 2 .

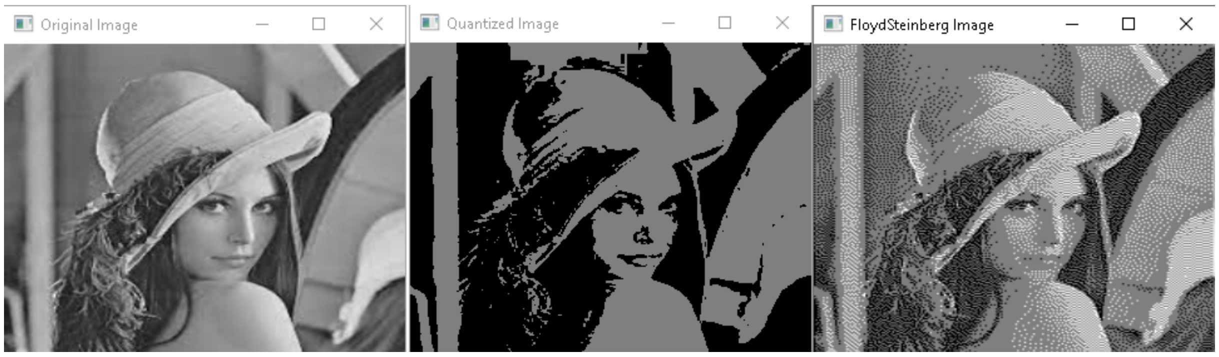
quant_error = Old Pixel Value – New Pixel Value

Dithered pixels = pixels[*x* + 1][*y*] := pixels[*x* + 1][*y*] + quant_error × 7 / 16
pixels[*x* - 1][*y* + 1] := pixels[*x* - 1][*y* + 1] + quant_error × 3 / 16
pixels[*x*][*y* + 1] := pixels[*x*][*y* + 1] + quant_error × 5 / 16
pixels[*x* + 1][*y* + 1] := pixels[*x* + 1][*y* + 1] + quant_error × 1 / 16

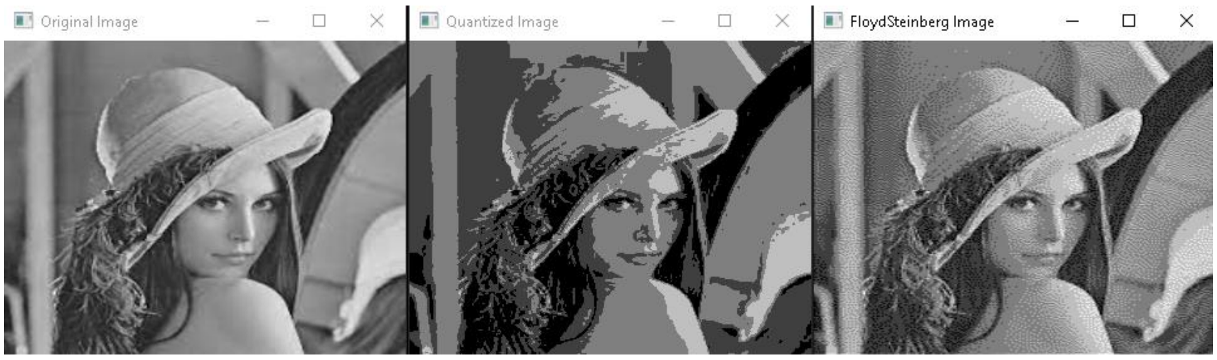
Because of the quantization and dithering calculations, if the q value increases , the error of the quantization decreases. That’s why when we increase the q value, the losing of visual content quality is less than before. In the example above, as the q value increases, we can see that the image quality deteriorates less.

The another examples are below:

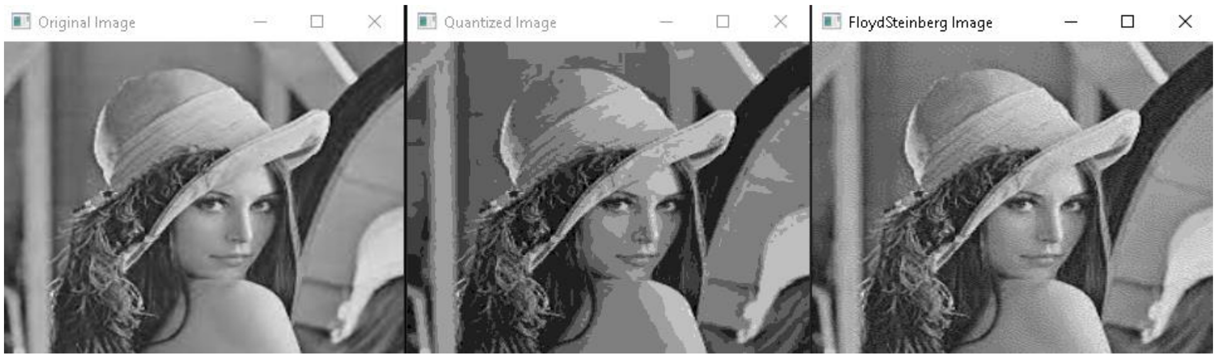
Q = 2



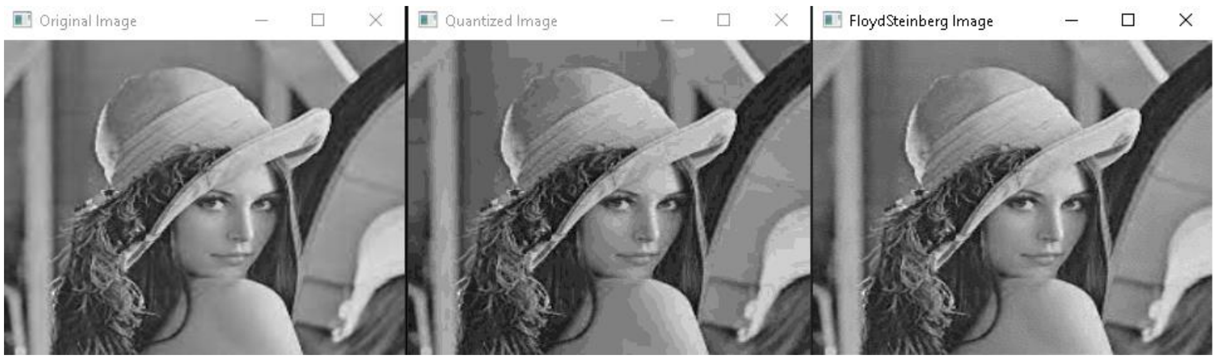
Q = 4



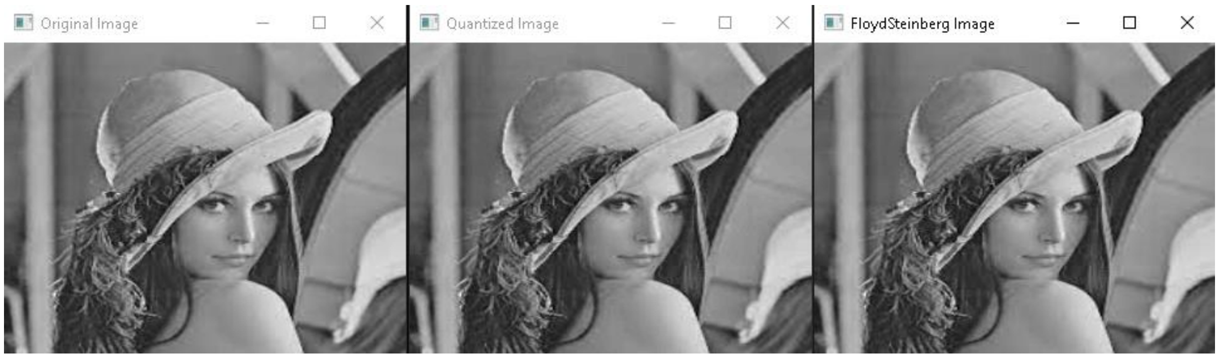
Q = 8



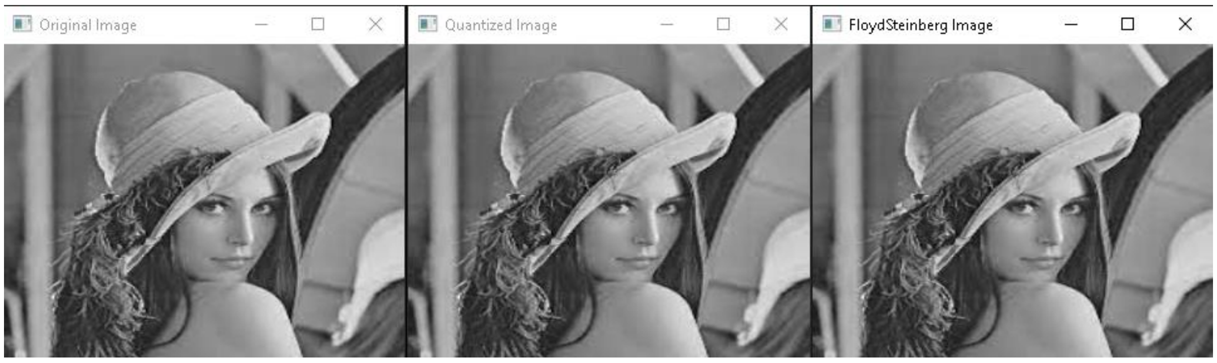
Q = 16



Q = 64



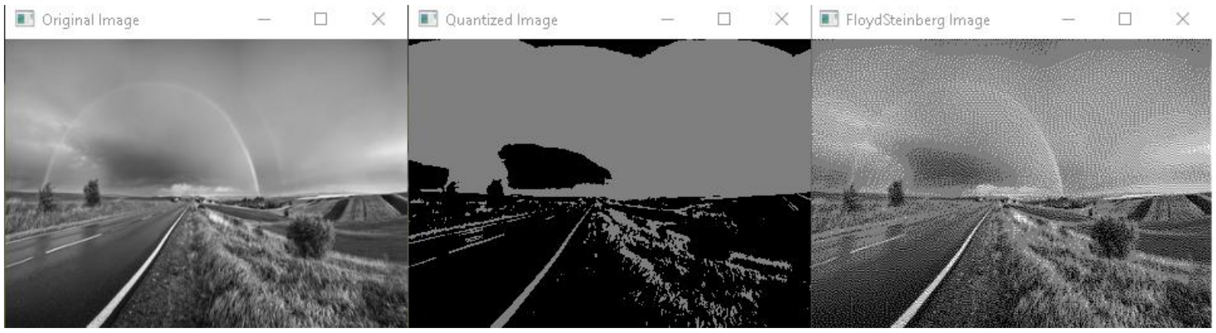
Q = 128



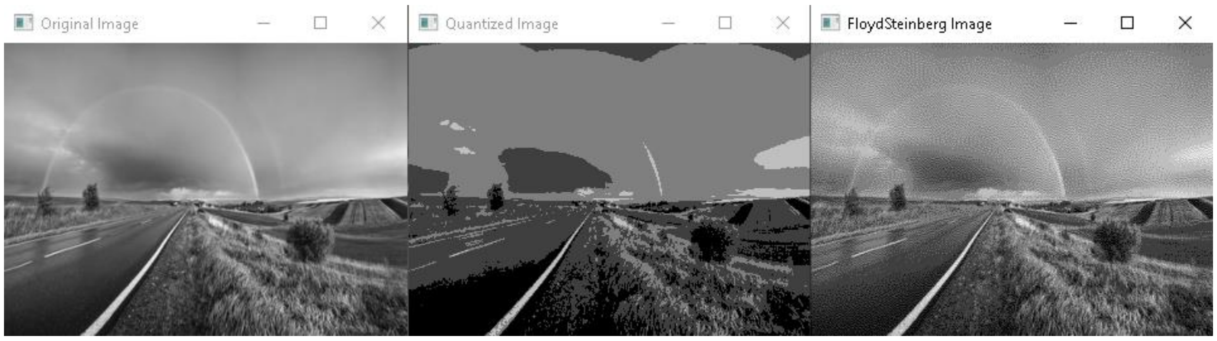
In the second example, we can see the same thing like in the first example, too. When the q value is increased, the bit level of quantized image increases, and also the quality of image is better than small q value in the bigger q value.

Another example is below:

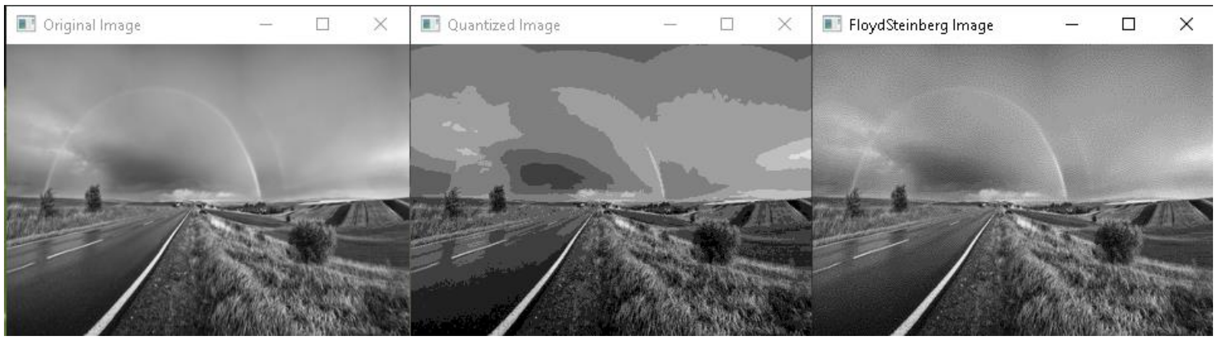
Q = 2


Original Image
Quantized Image
FloydSteinberg Image

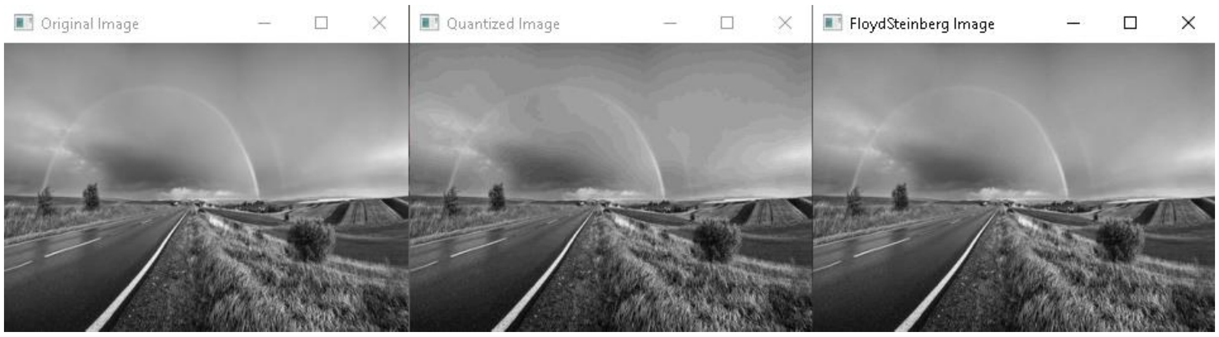
Q = 4


Original Image
Quantized Image
FloydSteinberg Image

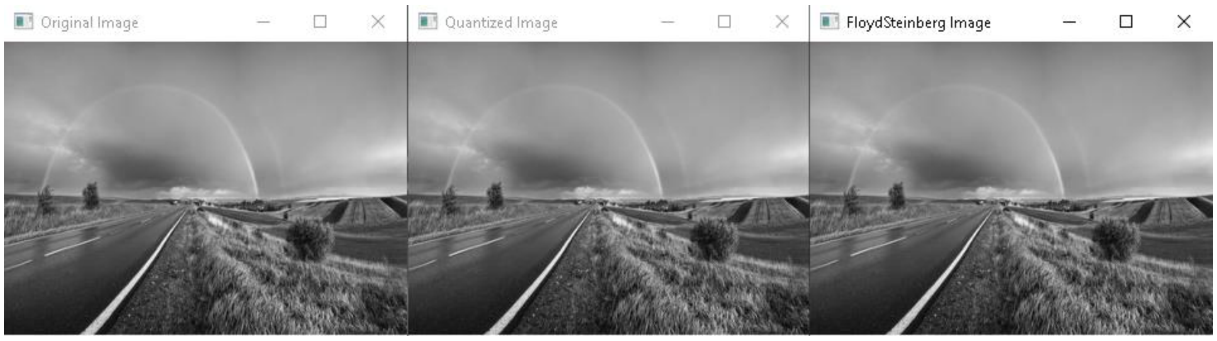
Q = 8


Original Image
Quantized Image
FloydSteinberg Image

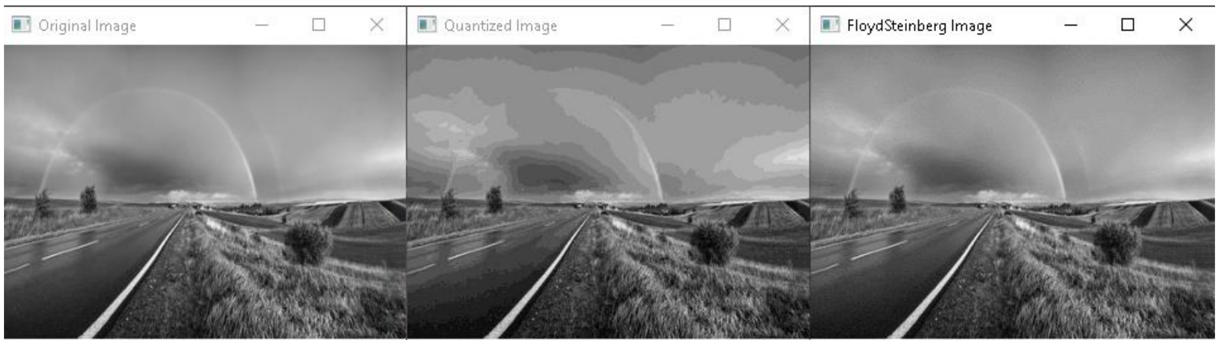
Q = 16


Original Image
Quantized Image
FloydSteinberg Image

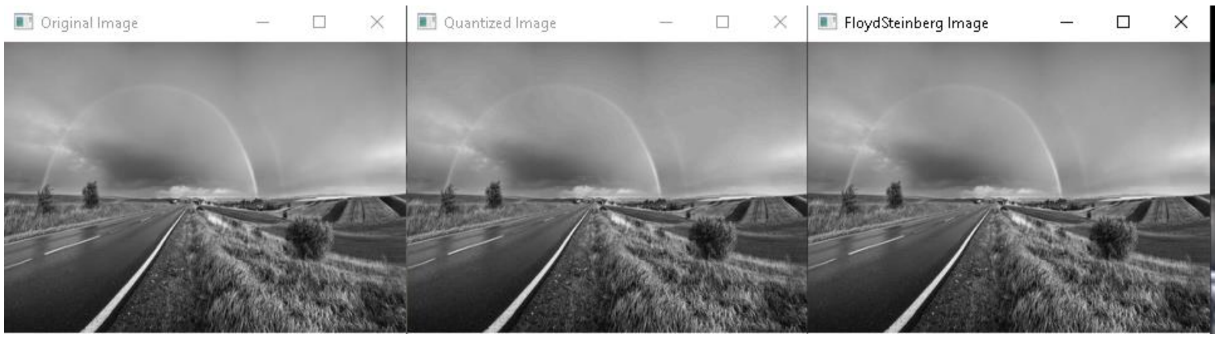
Q = 32



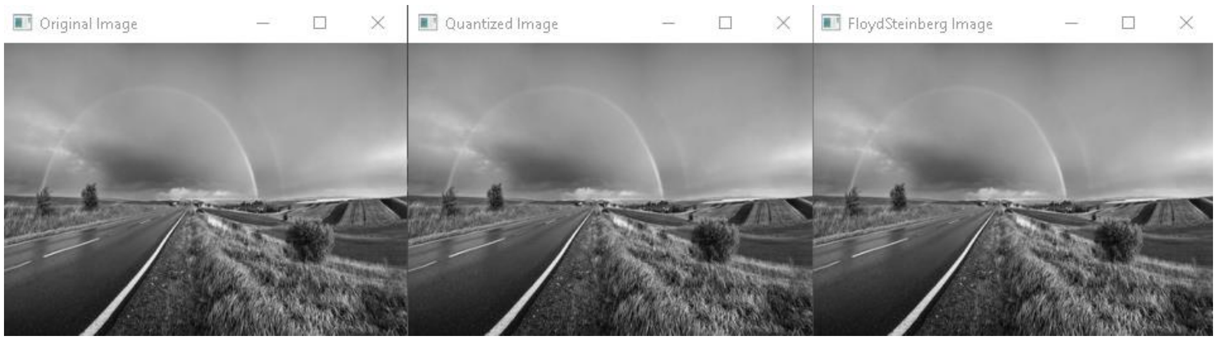
Q = 64



Q = 128



Q = 256



If we look the last example, we can see that the quantized image is black and white when the q value is 2. The details in the pictures is not clear enought to understand what there is. For example, in the original image, there is a rainbow in the sky and also there are dark clouds and light clouds. When the q value is increased, these details are more clear than the previous q values. Because, the bit level of the image increases. Also, the dithered image when the q value is 2, there is no clear pattern of dots is formed in places of uniform greyness. One of the disadvantages of the FS Dithering Algorithm is this situtation. When the dots are large, the problem is big because the original image and the dithered image look like different images. The other disadvantage of FS is shadowing. A sharp line dividing two even regions of very different greyness might re-appear faintly several rows below, due to weird destructive interference of the error terms. Also, I think that the FS algorithm is very slower than the quantize algorithm.

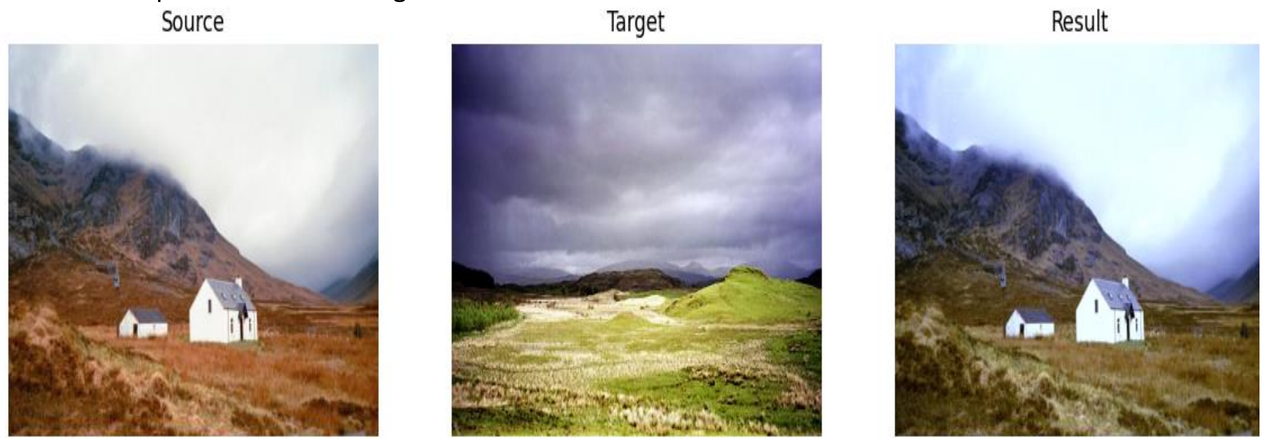
For runnig the pa1_1 , it needs command line arguments.

For example:
```
python3 pa1_1.py 2.png 2_quantized.png 2_dithered.png 4
```

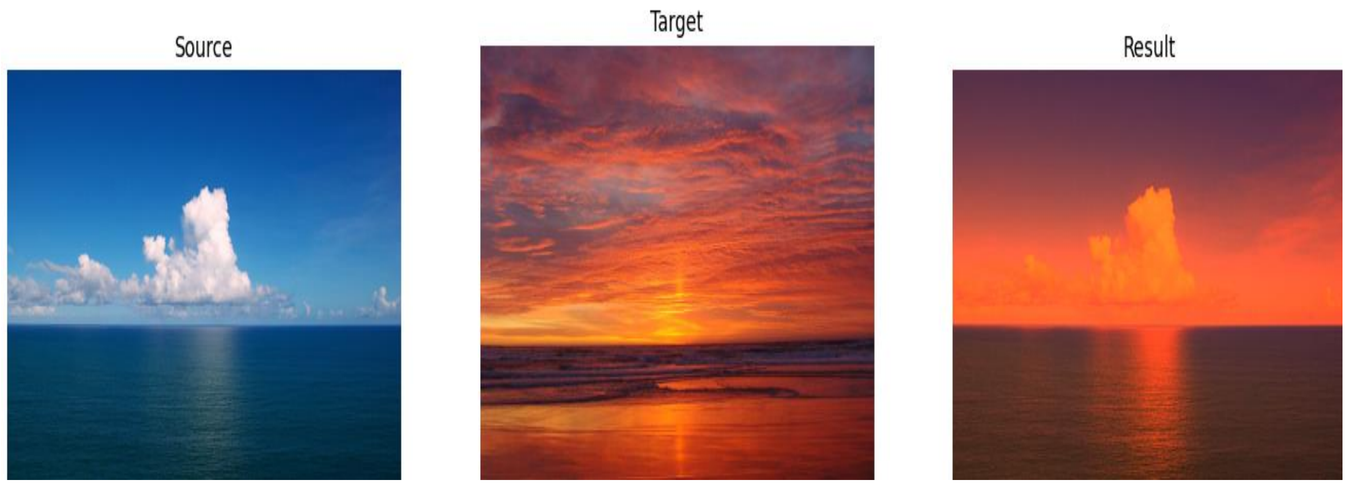
Part 2

In this part, we are expected to implement a color transfer algorithm. There are two python file named pa2_1.py and pa2_2.py. We are reading two image for source image which will be manipulated and the second image is target image which will be used for color characteristics in pa2_1.py. After reading these image, we call the function from pa2_2.py which is we implemented for color transfer. The color transfer algorithm has 10 steps. We put the pictures we read into the processes specified in the steps.

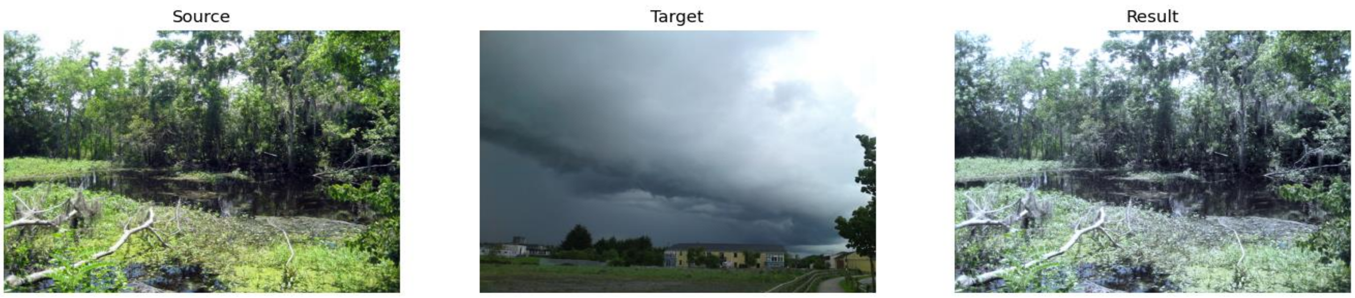
Example color transferrings are below:



First color transferring is above. In this example, we can see that the color characteristics of Target image is transferred to the Source image.

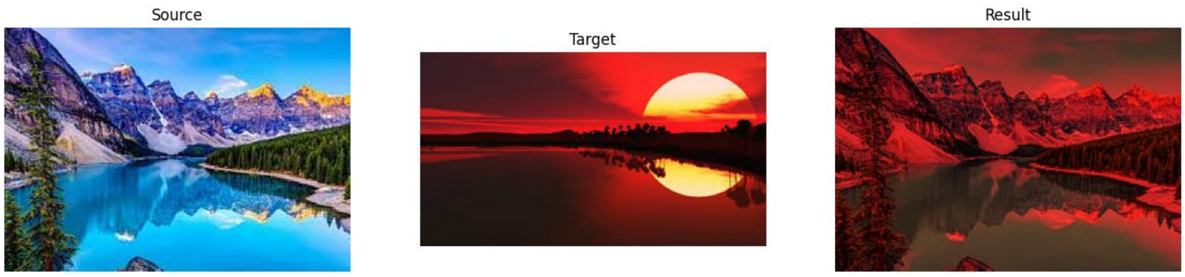


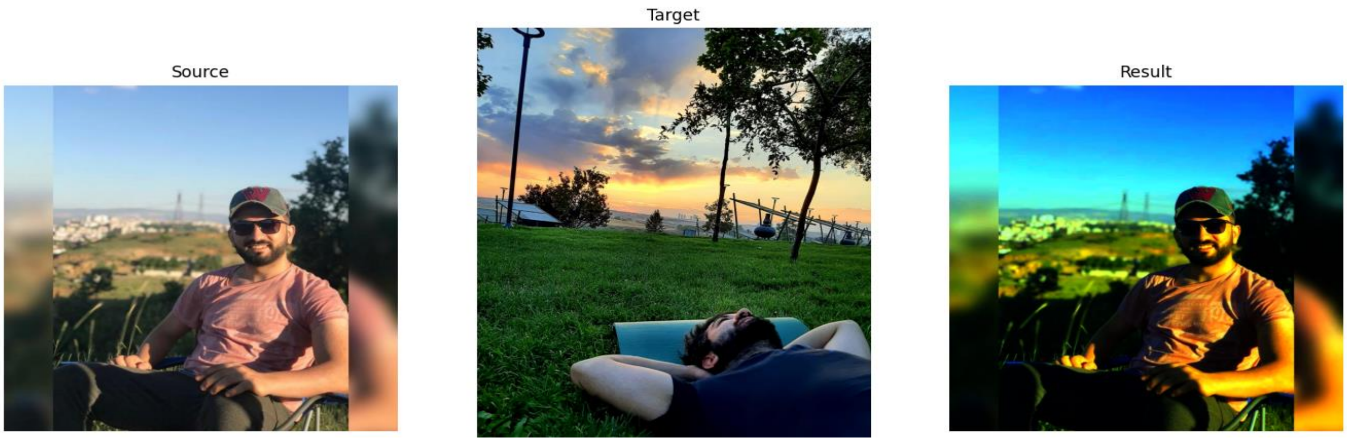
Again, in this example we can see that the color characteristics of the Target image is transferred to the Source image.Because of the algorithm and the differences of the RGB channels of the images, the color of Source image is too different from original state. We can see that the result image RGB channels contains the properties of the target image.



In this example, we can see that the colors of source , like the green of plants and tress, changed. When we look at the result image, the gray and white colors of target image changed the source image colors. According to me , this color transferring is like a filtering we used in real life, like instagram filters, snapchat filters.

The Lab color space is efficient for using color in all fields in the life. Because It is designed for visual media. The evaluation of color is related to the human eye's perception of color. Equal distances in the color space are represented as nearly equal perceived degrees. LAB color space is suitable for measuring color differences.Also, the Lab color space is usefol for boosting colors and definition in images due to the way it handles colors when compared to RGB. Lab is designed to approximate human vision. That’s why the LAB color space is used in this algorithm.





These two example is the bad example for this algorithm. If the the RGB channels too different from each other in two image, the result is bad according to the vice versa.

For running pa2_1.py, it needs to command line arguments. (python3 pa2_1.py source.jpg target.jpg)