Ilan Sapoznikov

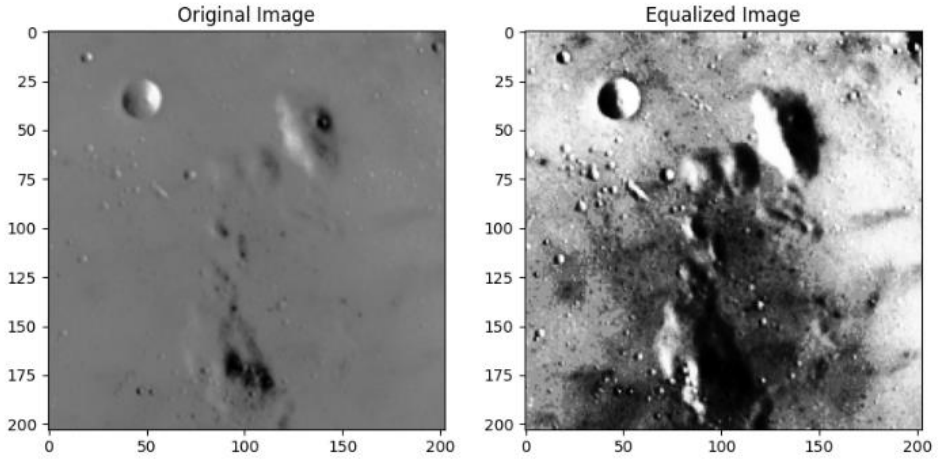
Liad Arvatz

Image Processing: Assignment #2

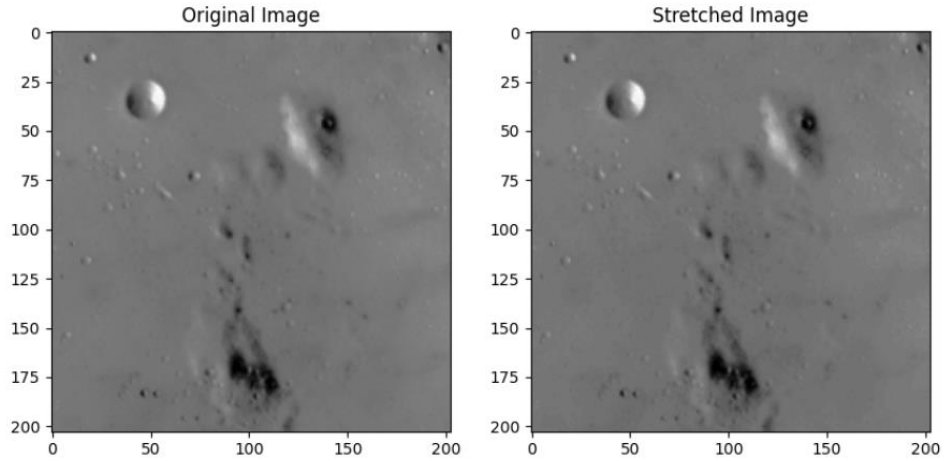
Problem 1- Point operations:

First picture:

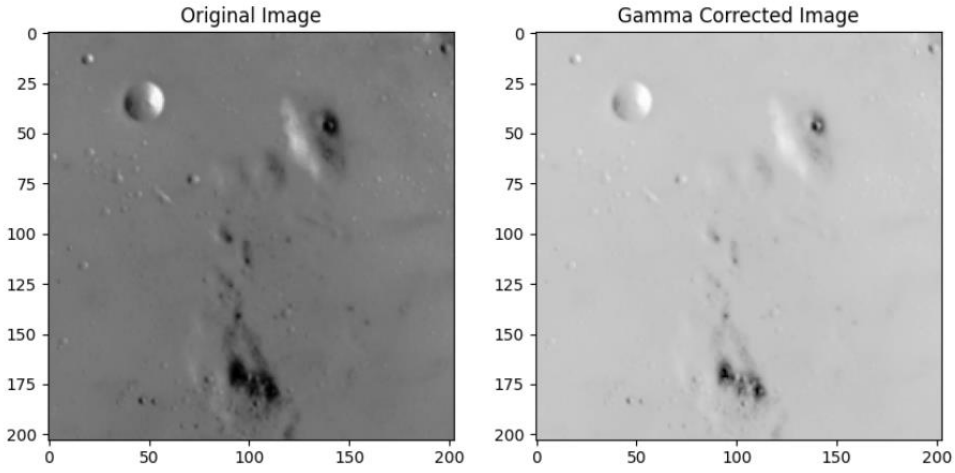
a) Correction with histogram equalization:



b) Correction with brightness and contrast stretching:



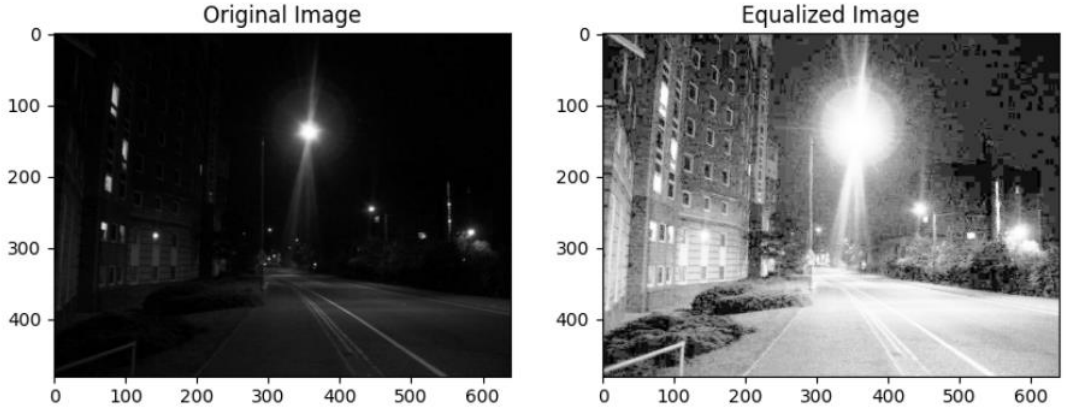
c) Correction with Gamma correction:



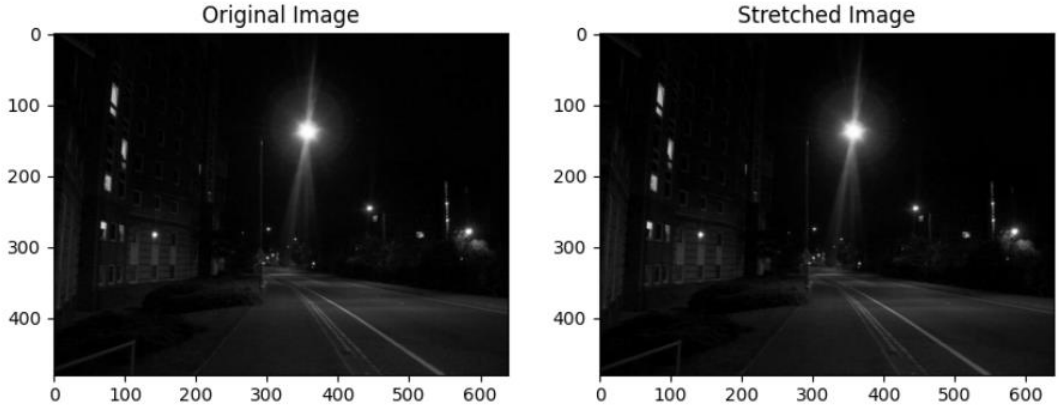
In this picture, we observe that the gray levels are centered around a specific value group and are not distributed across all levels. This scenario is a classic case for using histogram equalization.

<u>Second picture</u>:

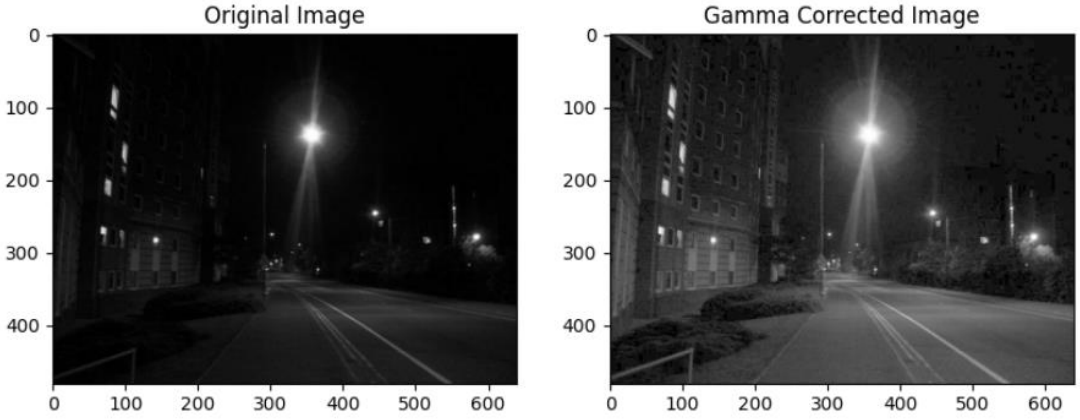
a) Correction with histogram equalization:



b) Correction with brightness and contrast stretching:

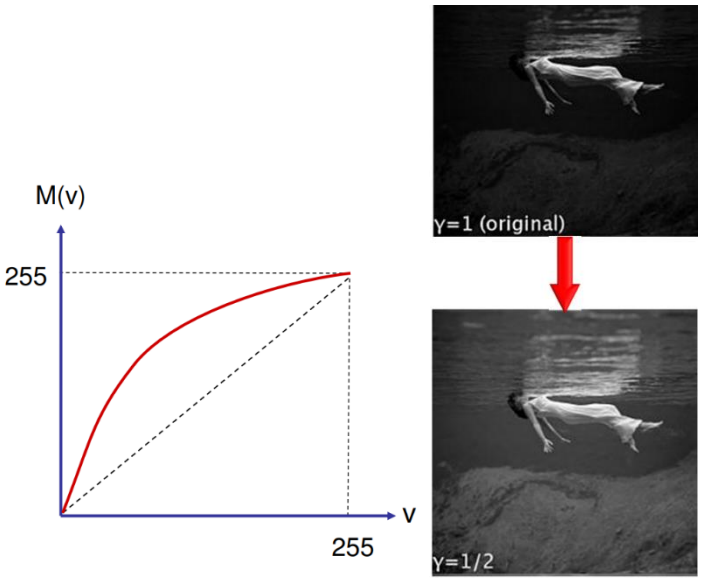


c) Correction with gamma correction:
when gamma=0.5



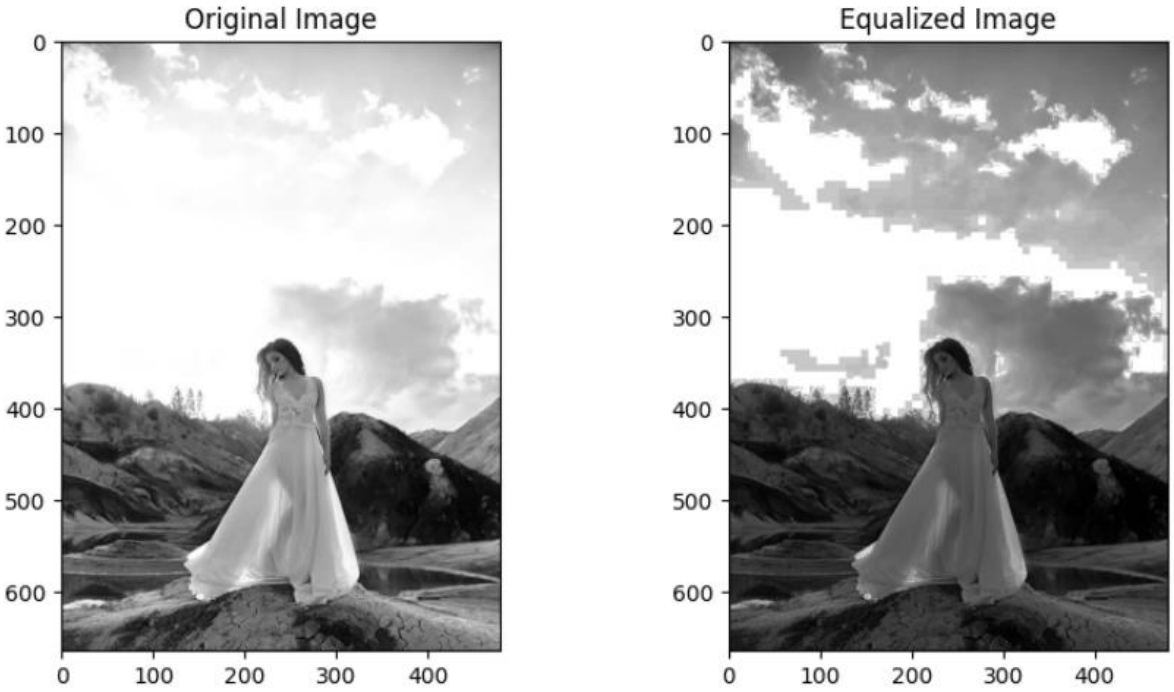
In this picture, we notice that the image is dark, and the elements are not clear. In this situation, gamma correction can adjust the gray levels in the picture. To achieve this, the gamma value must be less than 1, causing the pixel values to decrease and resulting in the image becoming brighter. (pixels with higher values represent brighter gray levels).

Here is an example to a similar situation:

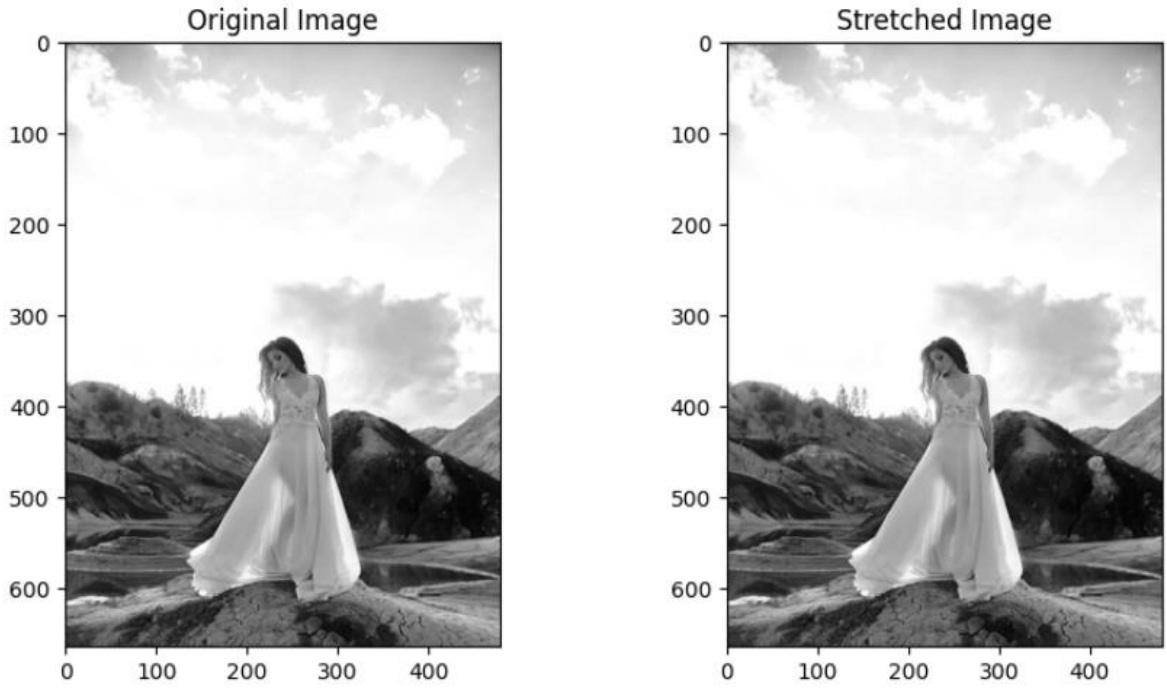


<u>Third picture</u>:

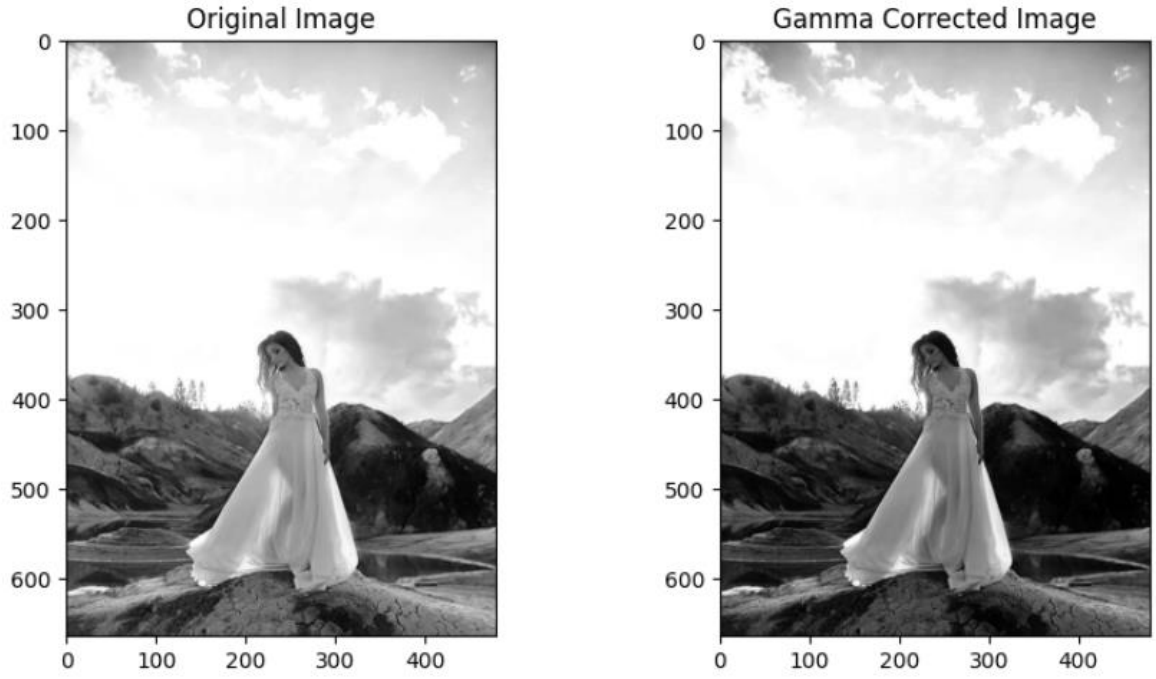
a) Correction with histogram equalization:



b) Correction with brightness and contrast stretching:



c) Correction with gamma correction:
when gamma=1.5



We observe that the image is excessively bright, particularly in specific areas, making simple adjustments to brightness and contrast ineffective. Furthermore, we notice that correction may not be necessarily required here. However, we can correct the picture using gamma correction with multiple gamma values, as the picture is only partly bright.

Problem 2- Puzzle solving with geometric operations:

Algorithm explanation:

First of all, we saved the files of the puzzles in a list.

For each puzzle out of the three puzzles that we received, we did the following:

1. Created a folder named 'abs_pieces', where all puzzle pieces, after transformation on the canvas, would be placed.

2. Accessed the first piece in the 'pieces' file and saved it separately.

3. Saved the dimensions of the canvas (the first piece).

4. For each piece in the puzzle, starting from the second piece:

1) Applied transformation based on corresponding points between that piece and the first piece (canvas). There are two types of transformations - Affine_transform and projective_transform.
If the puzzle is of type affine, the transformation will be performed using cv2.getAffineTransform; otherwise, using cv2.getPerspectiveTransform.
Padding with a row [0,0,1] will be performed on the matrices as needed to ensure squareness (occurs in Affine transformation).

2) Using inverse_transform_image, the piece will be fixed to its position on the canvas.

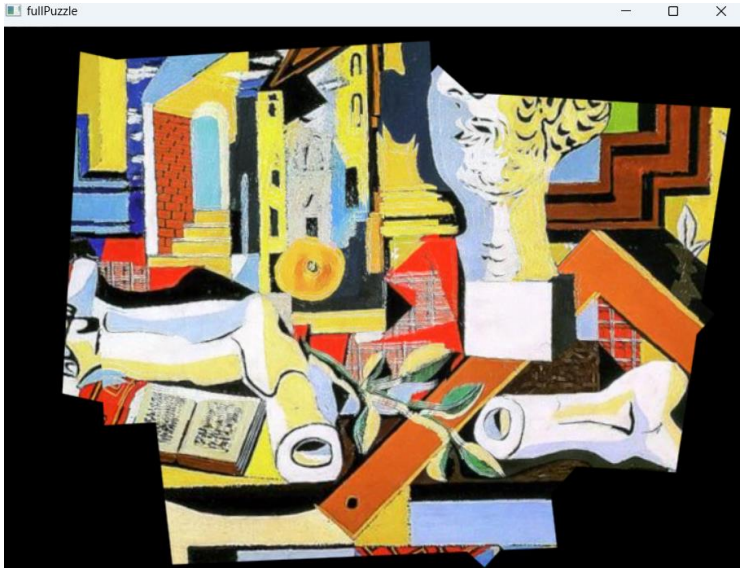
3) Inserted the image created from inverse_transform_image into the folder 'abs_pieces' we created earlier.

4) Stuck the piece to the canvas (the first image) using cv2.max between the two images, as we can do this since the area outside the relevant piece is black.

Final Result:

puzzle 1:

puzzle 2:



puzzle 3: