Local Histogram Equalization

Introduction

Histogram Equalization is a method in image processing of contrast manipulation using an image's histogram. A histogram is nothing but the probability distribution of its intensity values. This can be represented as a probability density function, pdf. Using this idea, we can manipulate the intensity value of an image using its corresponding histogram value.

There are two types of histogram equalization techniques, local and global. A global histogram equalization deals with manipulation of pixel values without the consultation of any neighbors. What this means is that the output intensity is a function of just the input value and the corresponding transformation function. On the other hand, a local histogram equalization considers neighboring pixels and creates a CDF (cumulative density function) based approximating for the resulting intensity. This repository deals with Local Histogram Equalization.

Approach

Local Histogram Equalization can be described in the following steps:

1. Replicate padding.
2. Get slices (m x n) based on values for m and n.
3. Compute pdf for ith slice
4. Compute CDF for PDF(ith slice) with value z
5. Find the middle element and replace with the round(CDF) value z.
6. Return new matrix without the replicate padding.
7. Repeat for all slices.
8. Reshape matrix to appropriate kernel formula if all values appended singularly.

To discuss more into details, we perform padding on an image to preserve information about the height and width of an image so when we perform the transformations, no information is lost. Moreover, padding an image actually improves the performance by keeping information at the

borders. This is especially true for localized approaches since it uses a neighbor of $m \times n$ pixels to compute the transformation. There are different types of padding that can be implement. The two most common are zero and edge padding. Zero padding refers to padding an image with width = k of value zero. On the other hand, edge/replicate padding pads the border with the edge pixel so you'll have a coated layer replicate the original image.

To illustrate this, here's an example of Zero and replicate padded grid:

0	0	0	0	0	0	0
0	1	2	3	4	5	0
0	6	7	8	9	10	0
0	11	12	13	14	15	0
0	16	17	18	19	20	0
0	0	0	0	0	0	0

Zero padding

1	1	2	3	4	5	5
1	1	2	3	4	5	5
6	6	7	8	9	10	10
11	11	12	13	14	15	15
16	16	17	18	19	20	20
16	16	17	18	19	20	20

Replicate Padding

After padding an image, we can perform slicing it based on a neighborhood of size $m \times n$. To perform this, we'll have to sequentially break up the image into equal $m \times n$ slices with overlapping values to prevent from "blocky" effect experienced from global equalization.
This looks something like this:

$$B = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 1 & 0 & 0 \\ 0 & 0 & 1 & 3 & 0 \\ 0 & 4 & 1 & 4 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

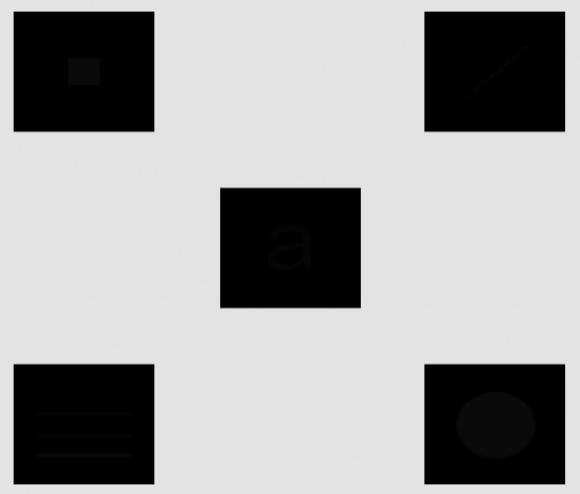
The matrix B is the padded image and the red rectangular box denotes the first slice. We'll perform this until all pixels of the image have been mapped in an $m \times n$ slice. Each slice represents a list of neighbors and the middle pixel will be the one being updates based on various histogram techniques. After this, the process is the same as global histogram equalization. But now, we've broken the initial image to smaller slices, each of size $m \times n$. This is a powerful technique as the neighboring pixels can indicate the value for the nearest pixel rather than having all pixels behave independently.

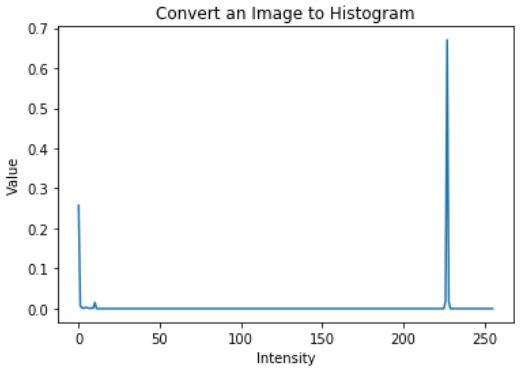
To perform equalization, we notice the property of a histogram based on probability which can be represented as a probability density function. The pdf of a single pixel is the proportion of its frequency to the size of the slice. In the red rectangular box highlighted above, the probability of a selected pixel being 0 equals 5/9. Similarly, we can calculate this proportion for all the values in a slice.

Upon calculation of individual densities, we go on to calculate the cumulative density of the intended pixel. This is the critical step that establishes mapping from the input pixel to the output pixel. A CDF (Cumulative Density Function) is a summation of PDF's from 0 – X. Note that the CDF of X=x is 0. Since the middle pixel in the rectangular box is 2, we perform the cumulative density function by $\sum_{i=0}^{n} pdf(slice_i)$. This will give us the raw CDF, but this needs to be scaled up to represent the true output intensity. This can be done by multiplying the value from summation by the range of possible values, [0 – L – 1]. In the case of images, this would be 255, because [0, 256 – 1] = 255. After calculating this value, we round off to represent a pretty integer and replace the existing middle value with the calculated scale up CDF value. Once this is done, the padded image is discarded and is automatically replaced with the new calculated grid. This is how local histogram equalization works.

Experimentation

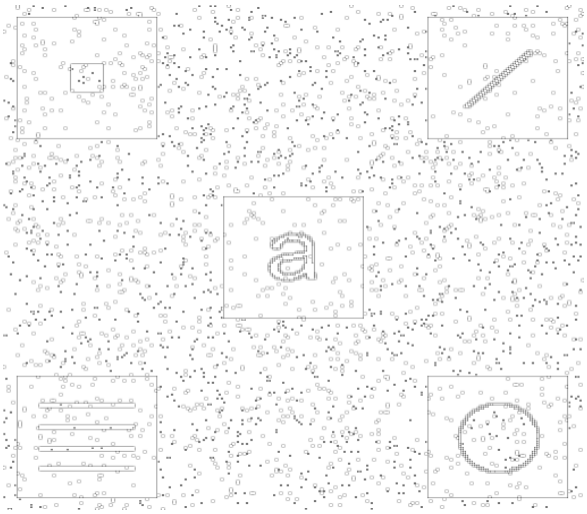
Local Histogram Equalization can be a powerful technique when applied to images of high contrast. Consider the following image with its histogram:

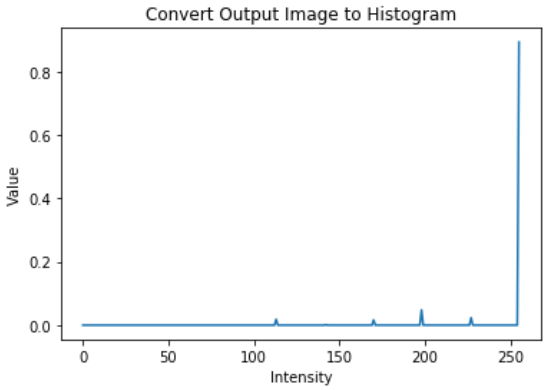




From the histogram, it's quite obvious that most of the pixel values are distributed to the grey/white side (about 70%), while as the remaining 30% are black. Using histogram equalization, we can manipulate the image to reduce the high contrast present.

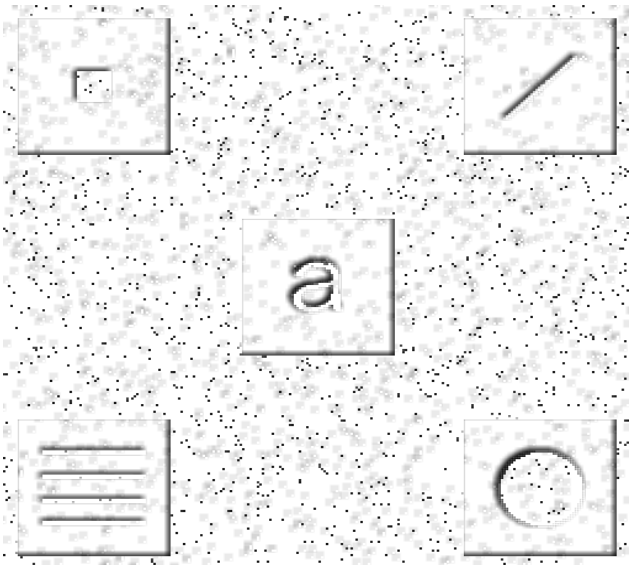
Let's demonstrate this by constructing a neighborhood of m = n = 3. The following result with the corresponding histogram is shown below:

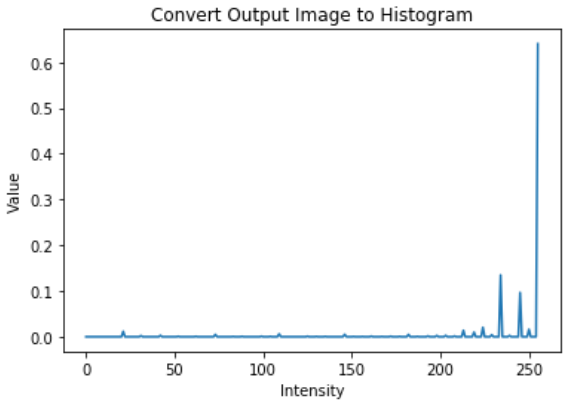




Using this technique, it's clearly obvious from the histogram that over 86% of all pixels have now been very bright, close to 250-255. This reduced the contrast by over 26%. In the process of doing so, it's clear to see that the output image has some interesting hidden properties consisting of various shapes.

When changing the Kernel size to 7, i.e. m = n = 7, things get interesting:





Although the shapes have become more apparent and bulkier, the contrast has increased from the previous transformation. Over 10% of pixels are in the realm of 230-240 while just 65% are white. This is because using larger neighborhood tend to increase noise and less accurate. This is true because using larger neighborhoods would often have inconsistent information since 3x3 flows more in terms of intensity than 7x7 which can different values. Therefore, there needs to be a trade-off between choosing neighborhood value and output contrast.

Conclusion:

Histogram Equalization is a powerful technique in representing the intensity distribution of an image which is used to improve image contrast. There are 2 kinds of HE (histogram equalization) techniques, local and global. A global histogram equalization deals with manipulation of pixel values without the consultation of any neighbors. What this means is that the output intensity is a function of just the input value and the corresponding transformation function. On the other hand, a local histogram equalization (LHE) considers neighboring pixels and creates a CDF (cumulative density function) based approximating for the resulting intensity. In this repository, we dealt with LHE by changing high contrast images to low contrast which proved to be a powerful transformation.

All the code implemented is found on my GitHub repository.

Reference:

Google Colab Link -

https://colab.research.google.com/drive/1UCgTdZMaFBSUlPakZFovCZ33bEMxJdo8?usp=sharing