SAGIVTECH

Classical Image Processing – Hands On

Lecture 2: Histogram Equalization, Convolutions, Denoising

Task 1: Histogram Equalization

The aim of this task is to write a function that gets as input an image and as the output presents the original image and its histogram along the histogram equalized image and the new histogram.

You can use this image as an example:

https://commons.wikimedia.org/wiki/File:Unequalized_Hawkes_Bay_NZ.jpg

Task 2: Histogram Matching

While the goal of histogram equalization is to produce an output image that has a flattened histogram, the goal of histogram matching is to take an input image and generate an output image that is based upon the shape of a specific (or reference) histogram. Histogram matching is also known as histogram specification. You can consider histogram equalization as a special case of histogram matching in which we want to force an image to have a uniform histogram (rather than just any shape as is the case for histogram matching).

Let us suppose we have two images, an input image and a specified image. We want to use histogram matching to force the input image to have a histogram that is the shape of the histogram of the specified image. The first few steps are similar to histogram equalization, except we are performing histogram equalization on two images (original image and the specific image).

Step 1: Obtain the histogram for both the input image and the specified image.

Step 2: Obtain the cumulative distribution function for both the input image and the specified image.

$$CDF = H(j) = \sum_{i=0}^{j} h(i) \qquad where\ j = 0, 1, \ldots\ldots, 254, 255$$

Step 3: **Calculate the transformation T to map the old intensity values to new intensity values for both the input image and specified image.** Let K represent the total number of possible intensity values (e.g. 256). j is the old intensity value, and T(j) is the new intensity value.

$$T_{input}(j) = floor((K-1) * CDF_j)$$

$$T_{specified}(j) = floor((K-1) * CDF_j)$$

Step 4: **Use the transformed intensity values for both the input image and specified image to map the intensity values of the input image to new values**

We go through each available intensity value j one at a time, doing the following steps:

- See what the transformed intensity value is for the input image given the intensity value j. Let us call this $T_{input}(j)$.
- We then find the $T_{specified}(j)$ that is closest to $T_{input}(j)$ and make a note of what j is. For example, if j = 4:

  $$T_{input}(4) = 3$$

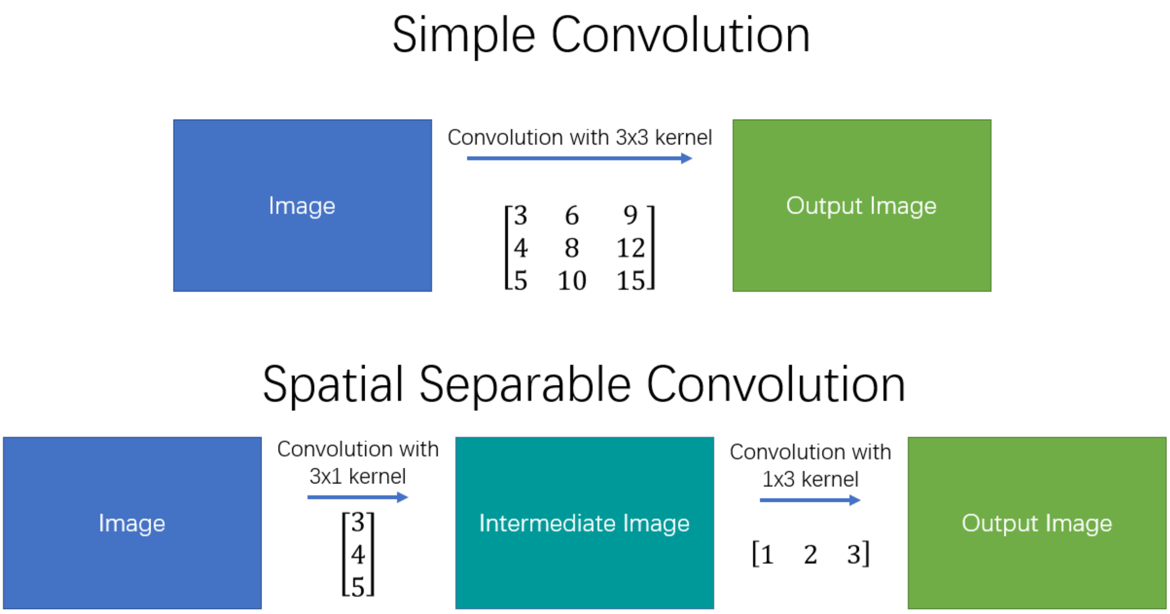
  $$T_{specified}(1) = 3$$

  we map all intensity values of 4 in the input image to 1.

After we have gone through all available intensity values and performed all the mappings, we have our output image which has a histogram that will approximately match the shape of the unequalized specified histogram.

Task 3: Separable Filter

Write a code that implements the following:



Prove that this is the same operation mathematically and by comparing the results.

Calculate the number of operations per case.

Task 4: Create Noisy Images and estimate SNR and noise distribution

Create an image with constant value, e.g. 128.

Use a ready function that adds noise, e.g. https://www.kaggle.com/code/chanduanilkumar/adding-and-removing-image-noise-in-python/notebook

Calculate the histogram of the image and compare it (visually) to the known distribution. Do they match?

Task 5: Simple Mean Filter with size as a parameter

Take the Barbara image from Hands On 1 and add Gaussian noise to it.

Write a code that performs simple mean filter with the size of the filter as a parameter.

Don’t forget to normalize.

Task 6: Median Filter

Take the Barbara image from Hands On 1 and add Salt and Pepper noise to it.

Write a code that performs median filter with the size of the filter as a parameter.

Task 7: Gaussian Filter

Take the Barbara image from Hands On 1.

Write a code that performs Gaussian filtering with the following parameters: size of the filter, value of sigma. Observe the relations between the actual size of the filter and sigma.

Don’t forget to normalize.