Ceng 466 - Take Home Exam 1

Eda Özyılmaz
2171882

Hilal Ünal
2172112

I. Introduction

This document is designed as a report for CENG466, Fundamentals of Image Processing, Take Home Exam 1. This report includes explanations of our methodology and analysis of the results we received while developing our code. The report mainly explains the methodology and rationale behind our algorithm design. Comments and explanations can be found in the following section.

II. Questions

In this homework, there are three questions which are Question 1 - Image Interpolation, Question 2 - Histogram Processing, and Question 3 - Noise Elimination. For some questions, we obtained expected results; however, for some we didn't. The questions and our answers, analysis, and methods for these questions are explained in detail in the following sub-sections.

A. Question 1 - Image Interpolation

Interpolation methods are used when stretching, zooming, shrinking, rotating, and resizing images. We need to make interpolation to fill up the color of moved pixel when we are enlarging images. Interpolation is estimating values at each unknown location with using the known pixels. There are three approaches, bilinear, bicubic, and nearest neighbor interpolation methods, to estimate the unknown values and we used two of those methods in our homework.

For enlarging images we used both bilinear and bicubic interpolation methods as stated in the homework specification file.

- Bilinear interpolation means applying a linear interpolation in two directions. Thus, it uses 4-neighbors, takes weighted average of their intensity values to produce the output and to estimate the intensity level of the unknown pixel. The following equation can be use to compute bilinear interpolation:

$$v(x, y) = ax + by + cxy + d \tag{1}$$

  There are four equations for each 4-neighbors and there are four unknowns to find which are a, b, c and d. (x,y) denotes to the coordinates of the location we want to assign an intensity value. v(x,y) denotes to this intensity value.
- Bicubic interpolation uses 16-neighbors to estimate the intensity of an unknown pixel. It averages the intensity values of sixteen neighbors and uses it to estimate the unknown pixel's intensity level. In general, bicubic interpolation is smoother than bilinear interpolation. The following equation can be use to compute bicubic interpolation:

$$v(x, y) = \sum_{i=0}^{3} \sum_{j=0}^{3} a_{ij} x^i y^j \tag{2}$$

  where (x,y) denotes the coordinates and v(x,y) denotes to the intensity value. There are sixteen equations which we can obtain from the sixteen nearest neighbors and sixteen unknowns to find.

In general, bicubic interpolation is better to preserve fine details than bilinear interpolation, because bicubic interpolation averages the values of the sixteen neighbors of the pixel so it takes more data and produces smoother images. However, bilinear interpolation averages the values of four neighbors of the pixel, obtained image is less smoother than the bicubic interpolation. After obtaining the results, we checked whether this general statement is similar with what we found or not. For this homework to resize and zoom in the given shrink images, we used existing Matlab function which is *imresize(..)* rather than creating our own interpolation methods.

For the *imresize(I,[numrows numcols],method)* function, we used three parameters which are the image we want to resize, the image size as [number of rows, number of columns], and interpolation method 'bilinear' or 'bicubic'. The arguments of the *imresize(I,[numrows numcols],method)* function are explained in detail:

- We get the matrix of the image we want to resize by using *imread(path_of_the_image)* function. This matrix is the first argument of the *imresize(..)* function.
- Second argument of the *imresize(..)* function is the row and column dimensions of the output image. For this homework, this means the size of the original image.
- The third argument of *imresize(..)* function is the interpolation method. When we chose this argument as 'bilinear' interpolation method, the output pixel value is weighted average of pixels in the nearest 2-by-2 neighborhood. When we chose the third argument as 'bicubic' interpolation method, the output pixel value is weighted average of pixels in the nearest 4-by-4 neighborhood.

Comparison between bicubic and bilinear interpolation is not visible to the human eye; therefore, we computed the Euclidean distances between bilinear-original image and bicubic-original image. The euclidean distance is calculated from the center of the original image's pixel to the center of resized image's pixel. For each pixel, the Euclidean distance is the

hypotenuse, with the x-maximum and y-maximum. Small deformation of the original image means small Euclidean distance. The stronger the deformation, the larger the Euclidean distance.

In our homework, for all of the given images, we obtained larger Euclidean distance for the bicubic interpolation and smaller Euclidean distance for the bilinear interpolation. This means that for the given images in our homework the bilinear interpolation is more similar to the original image than the bicubic interpolation.

However, in general the bicubic interpolation obtains more similar results to the original image than the bilinear interpolation. For Question 1 in our homework, the results did not match with the general observations and our expectations about the result. We obtained more smoother results with bilinear interpolation unlike we expected.

B. Question 2 - Histogram Processing

Histogram matching, which is also known as histogram specification, is a process where an image is modified such that its histogram matches with another reference's specified histogram. Basically, histogram matching forces the intensity distribution of an image to match the intensity distribution of a reference image.

Actually, Matlab has a special function as *imhistmatch(..)* for histogram matching;however, in our homework we were expected to develop our own histogram matching function. To develop our own histogram function first, we found the histogram of the images individually by using equation

$$h(x) = h(x) + 1 \tag{3}$$

where x denotes to the intensity value for the image. Then we calculated the cumulative distribution function, which is CDF, with the equation

$$h_c(x) = h_c(x-1) + h(x) \tag{4}$$

After calculating the CDF's for both of the images, we computed a mapping that transforms one intensity from the first image so that it is in agreement with the intensity distribution of the second image. To do this, we used the following equation :

$$M(G_1) = min_{G_2 \in [0,255]} |F_1(G_1) - F_2(G_2)| \tag{5}$$
$$for\ \forall G_1 \in [0, 255]$$

where F_1 represents the CDF of the first image and the F_2 represents the second image's, the G_1 and G_2 are the intensity levels of images, and the function $M(G_1)$ represents mapping for each entry G_1. We computed a mapping for all three red, green, and blue pixels in both of the image, we applied histogram matching to each color panel independently.

- For the first example in our homework, we applied histogram matching on B1.jpg, when reference image is B2.jpg. Image B1 was lighter than B2. Therefore, when we took B2 as a reference image, we obtained B1_histmatch_output.jpg which is more darker version of original image B1.jpg. We plotted the histogram graph of this output as B1_histmatch.jpg. In this image, there are three sub-graphs which represents the histogram of red, green and blue pixels for the output image. In these sub-graphs, number of pixels for darker colors' which has the smaller intensity was higher than the number of pixels for the lighter colors'. This happened because the image is darker than its original version when we histogram matched it.
- Then we applied histogram matching on B2.jpg, when reference image is B1.jpg this time. Image B2 was darker than B1;therefore, when we made histogram matching we obtained more lighter version of B2.jpg, this output is called B2_histmatch_output.jpg. In the histogram graph of this output which is called B2_histmatch.jpg, there were more pixels with the higher intensity level. Because the image has more lighter pixels' in it after the histogram matching.
- We applied histogram matching on B3.jpg, when reference image is B4.jpg. B3 was more lighter than the image B4. Therefore, like the first example we tried, we obtained more darker version of B3, which is called B3_histmatch_output.jpg. The histogram of the output image is B3_histmatch.jpg, and it means that image B3 is became darker when we applied histogram matching because it has more darker pixels which have lower intensity level now.
- Finally, we applied histogram matching on B4.jpg, when the reference image is B3.jpg. B4 was more darker than the image B3 just like the second example. Therefore, we again obtained lighter version of image B4, B4_histmatch_output.jpg. The histogram of the output, B4_histmatch.jpg has more number of pixels with the higher intensity levels because the image is more lighter than its original version after applying histogram matching.

In this part of the homework, for all the input images, the histogram matched image and the histogram graph of the output turned out as we expected. When we matched an image with a darker image, original image became more darker. The histogram graph of the new version of image had more smaller pixel values than its original. When we matched an image with lighter image, as we expected it became lighter and its histogram graph had bigger pixel values than its original histogram graph.

C. Question 3 - Noise Elimination

Image noise is random variation of brightness or color information in the images. It is degradation in image signal caused by external sources. The principle sources of noise in digital images arise during image acquisition and/or transmission. The performance of imaging sensors is affected by a variety of factors, such as environmental conditions during image acquisition, and by the quality of the sensing elements themselves.

Images are corrupted during transmission principally due to interference in the channel used for transmission.

There are couple of types of images;

- Gaussian Noise
- Salt and Pepper Noise
- Speckle Noise

Noise in the images are generally are undesirable. To eliminate the noise in the images various filters can be used. This is what we tried to do in our homework.

In this part of the homework, we have three parts that we need to do. First is to write the convolution function to be able to apply filters to the images. Second part is to detect and eliminate the noise in the images, and lastly to apply edge detection filters to the images.

- First of all we started with writing a convolution function *the1_convolution.m* which takes an image and a filter, convolve the image with with filter and return the convolved image.
  We start with taking the transpose of the filter matrix and pad the image matrix according to size of the filter matrix. Then we apply the following equation to the results.

$$C(j,k) = \sum_{p} \sum_{q} A(p,q)B(j-p+1, k-q+1) \quad (6)$$

  Where C is the result image of the convolution and has the same size as the image matrix. B is the padded image matrix, padded according to the size of filter matrix , and A is the transposed filter matrix. p and q represent the height and width of the filter matrix while j and k represent the height and width of the image matrix.
- In the second part, to detect the noise types of the images, we used some of the filters we thought would be able to eliminate the noise. After trying several different filters from our lecture notes and from internet, we decided that the Gaussian filter and the median filter work the best for the inputs provided for our homework.
  For C1.jpg, giving the Gaussian filter as a filter parameter in the convolution function, we received the most satisfying output. The output image became more clear than any other filter we try for C1.jpg. As a conclusion we think that the noise in the C1.jpg image was the Gaussian noise.
  For C2.jpg, we receive the most clear output when we give median filter as a filter parameter in the convolution function. After applying median filter, on the output image we can clearly saw the details of the pattern in the C2.jpg. Even though the output image became a little blurrier, we still be able to see the texture better than before.
- For the last part, edge detection, we decided to use Prewitt filter. Applying Prewitt filter for horizantal edges, vertical edges and diagonal edges to images that provided for the homework using convolution function. To achieve a better result, we add the outputs we get from convolution functions when given the Prewitt filters.
  We try edge detection function on the images provided for first and second part of the homework, and we receive satisfying outputs. However applying the edge detection filter to C1.jpg and C2.jpg we get rather unexpected outputs.
  For C1.jpg, the output we receive looks like as if the objects are detected on the image and the background is black. We thought that it may be because of the noises in the image.
  The output we get for C2.jpg is different than the output for C1.jpg. On the output we can clearly see the edges of the patterns in the image.

While we get good results from edge detection function for images provided in first and second part, outputs we get for the images provided for third part do not exactly look like a edge detection. We conclude that noise in the images effects the edge detection filters, and blocks us from getting better solutions for our function.

III. Conclusion

In this homework, for the first part we bilinear and bicubic interpolation methods to resize an image, the results are not the results we excepted. In theory, most of the time bicubic interpolation is more realistic than bilinear interpolation. However, we obtained more realistic results with bilinear interpolation not with bicubic. In second question, we histogram matched given two images and obtained its result and the histogram of the output. ın this part, the results emerged as we expected. For third question, we first eliminated the noise from given images. To eliminate noise, we convoluted images with necessary filters. We picked these filters by deciding the noise type. Then, we detected the edges of the given images by convoluting image with edge detection filters. For some images, we obtained expected result; however, for some we did not.

References

[1] Gonzalez, R. C., & Woods, R. E. (2018). Digital image processing. New York, NY: Pearson.

[2] Paul Bourke. (2011, January). Histogram Matching [Online]. Retrieved November 2019, from http://paulbourke.net/miscellaneous/equalisation/.

[3] Ross Moore. (1999). Histogram Specification [Online]. Retrieved November 2019, from http://fourier.eng.hmc.edu/e161/lectures/.

[4] Saket Bhardwaj, & Ajay Mittal. (2012). A Survey on Various Edge Detector Techniques [Online]. Retrieved November 2019, from https://www.sciencedirect.com/science/article/pii/S221201731200312X.

[5] Pawan Patidar, & Manoj Gupta, & Sumit Srivastava, & Ashok Kumar Nagawat. (2010, November). Image De-noising by Various Filters for Different Noise [Online]. Retrieved November 2019, from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.206.3947 rep=rep1type=pdf .

[6] Bilal Charmouti, & Ahmad Kadri Junoh. (2017). Extended Median Filter For Salt and Pepper Noise In Image. Retrieved November 2019, from https://www.ripublication.com/ijaer17/ijaerv12n22$_1$56.*pdf*.