CS663 Assignment 1 Question 1

Shreya Laddha, Archishman Biswas, Shreyan Jabade, Rishabh Arya

September 1, 2020

This document describes the findings regarding question 1 of assignment 1. For each section in the question, we have added the relevant comments, graphs and images.

1 Question 1a

This function reduces the size of the original image by selecting every dth pixel in the image. The Moire effects can be seen in Figure 2 and 3. True sizes of images have been maintained.

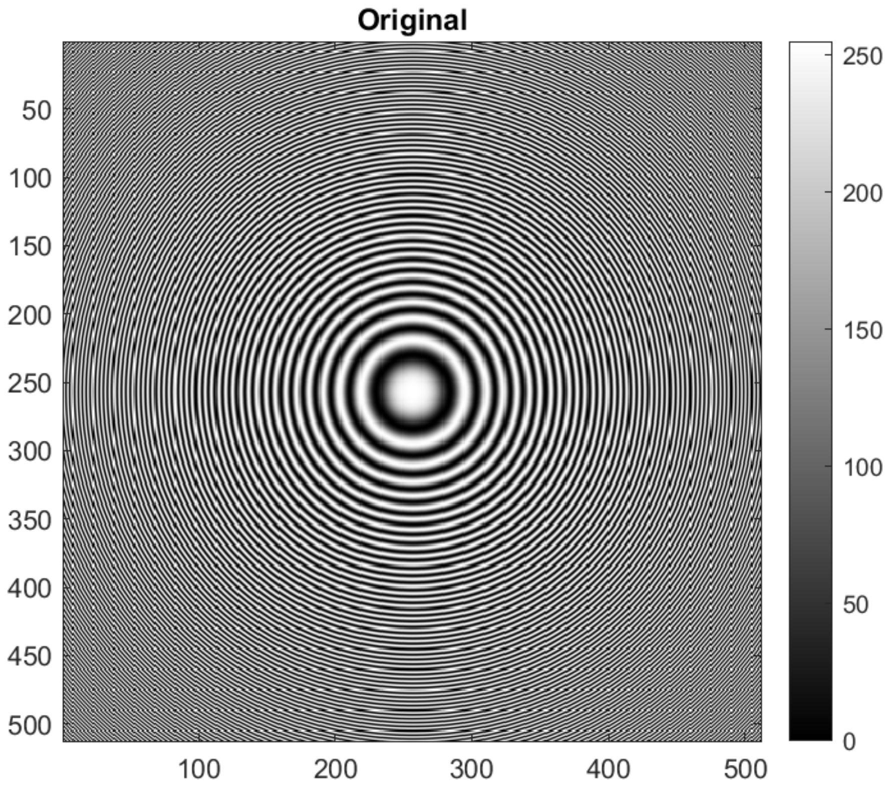


Figure 1: Original

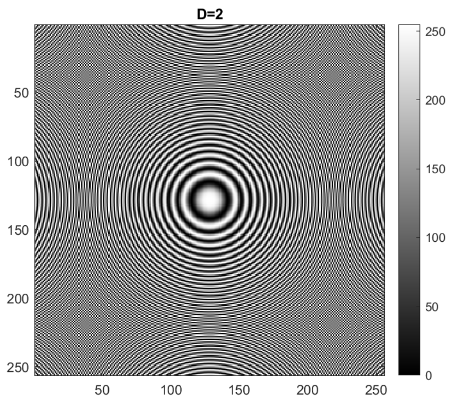


Figure 2: D=2

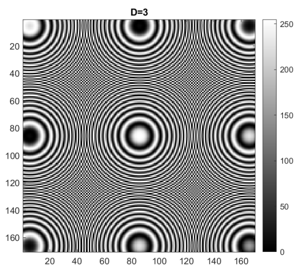


Figure 3: D=3

2 Question 1b

In the function myBilinearInterpolation.m, we have implemented bi-linear interpolation to enlarge a (M×N) pixels image to a (3M-2×2N-1) pixels image. This enlargement operation can be seen as inserting 2 rows between each pair of rows and inserting 1 column between each pair of columns.

Thus, we can first bi-linearly interpolate to fill the rows which are indexed by 1,4,7.... Then to fill the row which are not indexed by 1,4,7..., we can apply bi-linear interpolation using vector values of the previous row and the next row.

The input and the corresponding output image is shown in the figures 4(showing gray color mapping) and 5(showing RGB color mapping). To see the details, the RGB color mapping is also shown. The resulting image are of good quality and this algorithm can also be extended to any arbitrary enlargement ratio.

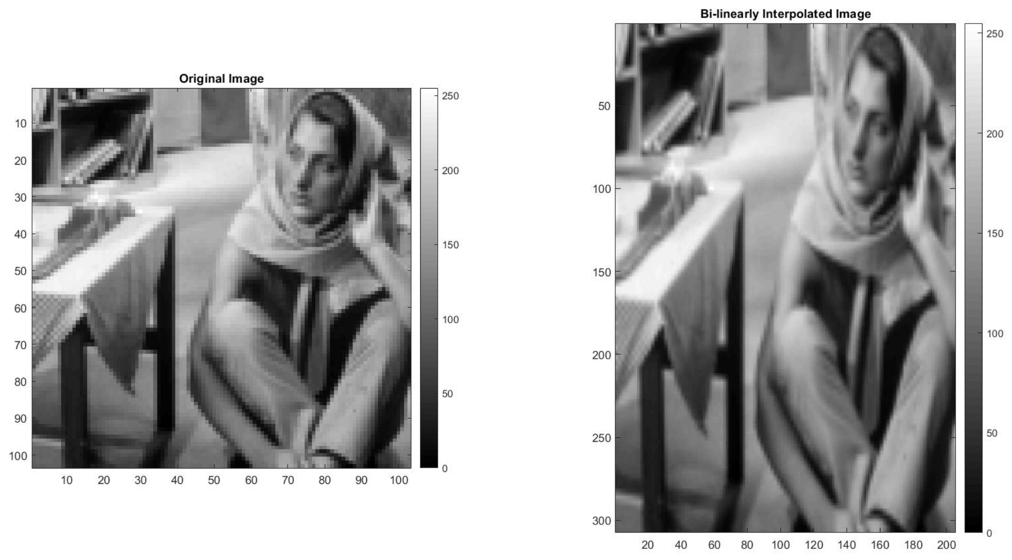


Figure 4: Bi-linearly interpolated image with gray colormap

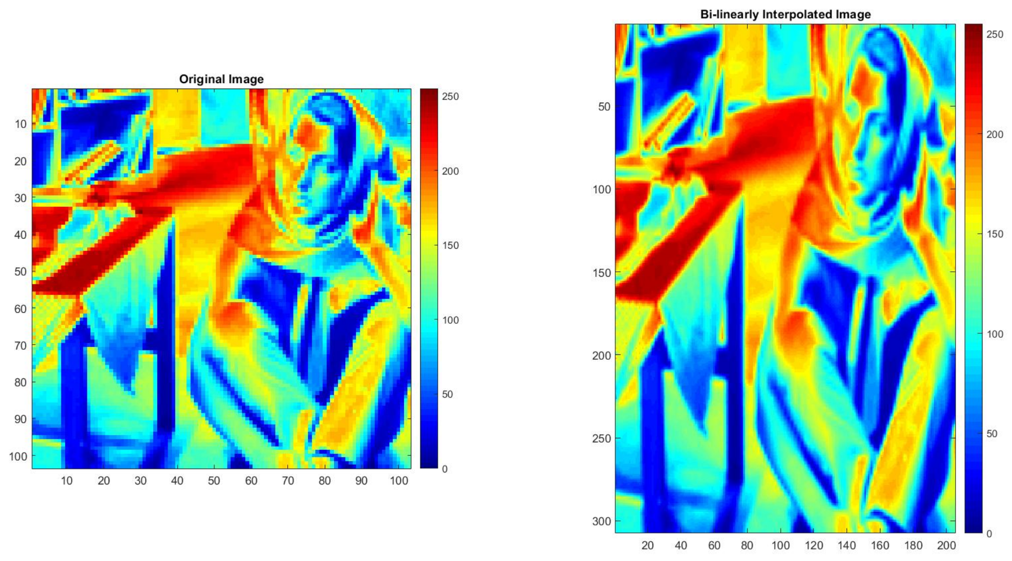


Figure 5: Bi-linearly interpolated image with RGB colormap

3 Question 1c

In the function myNearestNeighbour.m, we have implemented Nearest Neighbour interpolation to enlarge a (M x N) pixels image to a (3M-2 x 2N-1) pixels image.

The approach here is that if we divide the index of the enlarged image with the enlargement factor of rows and columns, the number after rounding the result is the index of the nearest neighbour from the input image. Enlargement factor here for rows is (3M-2/M) and for columns is (2N-1/N). this algorithm can be extended to any arbitrary enlargement ratio.

On comparing the output image to the bilinear result, we would see that there are jump discontinuities of intensities in nearest neighbour output image.

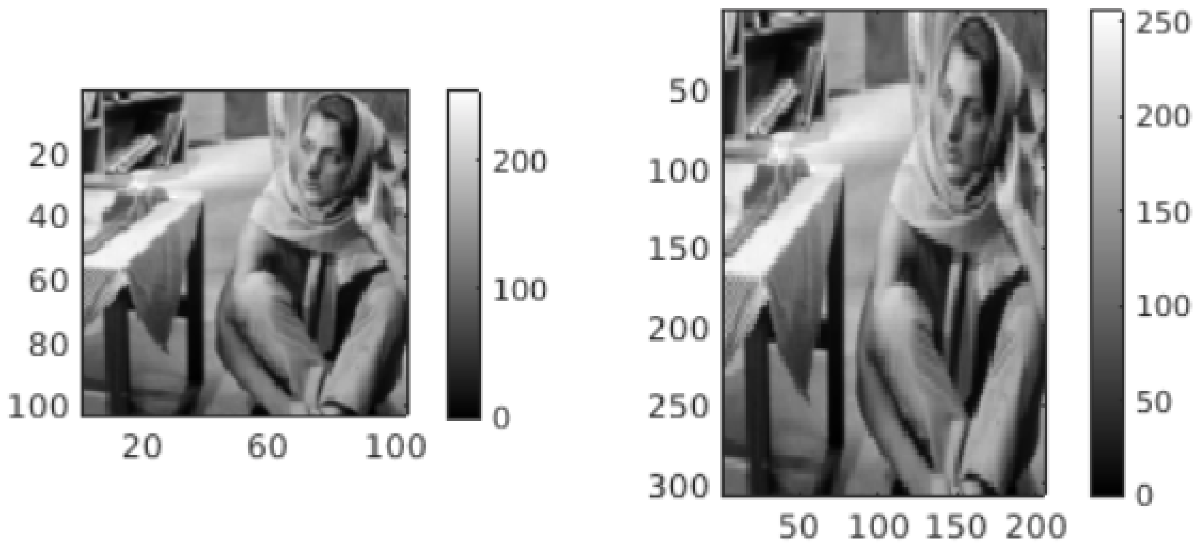
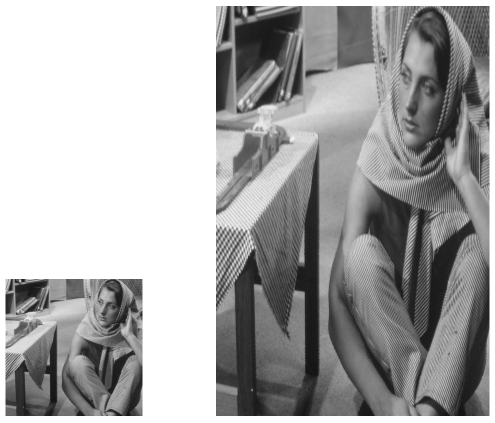


Figure 6: Nearest Neighbour interpolated image with gray colormap

4 Question 1d

In this part, we implement a function myBicubicInterpolation.m to obtain the enlarged version of original barbaraSmall.png image. The result can be seen in the figure below:



(a) Original Image (b) Bicubic Interpolated Image

Figure 7: Bi-Cubic interpolated image with gray colormap

It can be observed that the bicubic interpolation results in much ”smoother” image, more qualities in the image is visible as compared to the previous bilinear and nearest neighbour method.

5 Question 1e

Different interpolation methods are compared in the following figure. Nearest Neighbour interpolation gives the most ”pixelated” output as expected. Bilinear interpolation provides a relatively smooth enlarged image. Bicubic interpolation further improves the smoothness and clarity. Of the three methods, bicubic interpolation provides best results as is seen from the figure. But, bicubic method is computationally much more intensive.

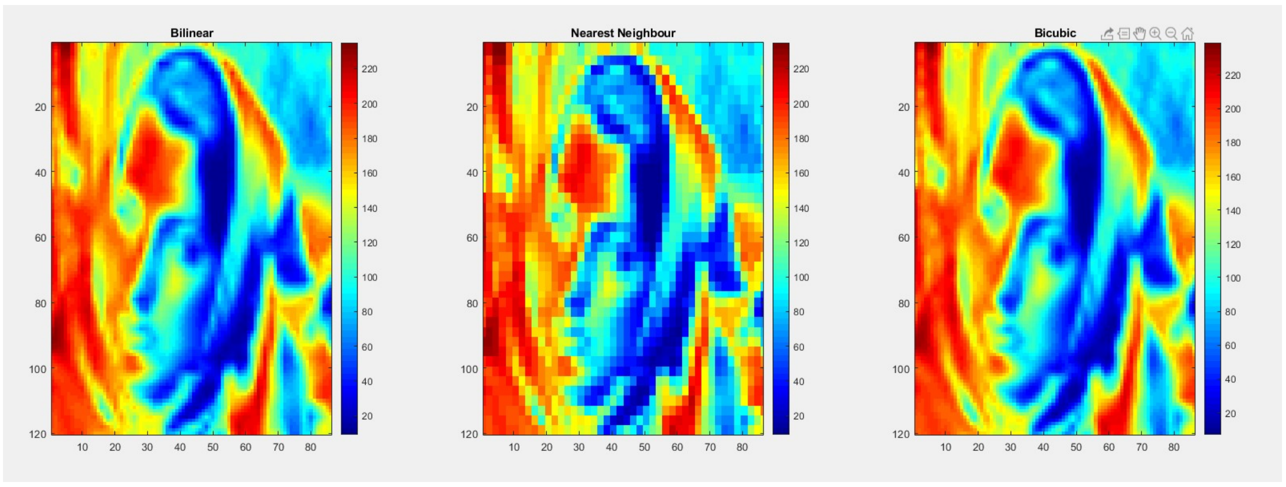


Figure 8: Comparison between nearest bilinear, nearest neighbour and bicubic interpolation method

6 Question 1f

In this part, we aim to develop an algorithm that can be used to rotate image using bilinear interpolation.

We first start with assigning a zero matrix of size double of that of the original image on both the axes to output image. An image rotation of 30^o in clockwise direction is equivalent to the underlying grid rotation by 30^o in anti-clockwise direction. Suppose we denote the anti-clockwise rotation matrix by 'R'.

For every (j,k) pixel in the output image, we can calculate a corresponding (x,y) = R*(j,k). Note that this (x,y) represent the pixel in original input image that got mapped to (i,j) during clockwise image rotation. This (x,y) may not be integer valued, thus we will use bilinear interpolation to obtain the pixel intensity at (x,y) by using the intensities of the 4 points that surrounds this point. For the values of (x,y) lying outside input image size, keep the corresponding output image pixel at (j,k) unchanged, i.e. to be zero.

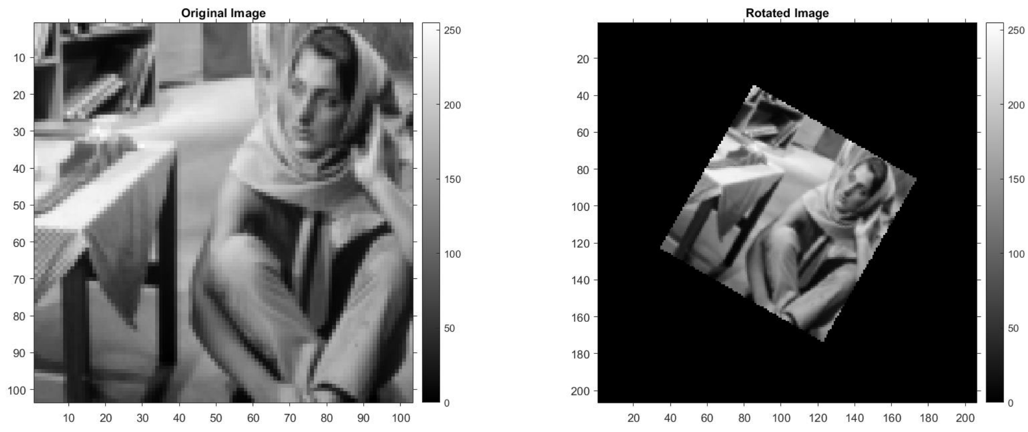


Figure 9

The rotated image can be seen in the figure 9. In the output image rotated by 30^o, the edges are not smooth after rotation. The output image size is chosen double in order to retain the corners after rotation. The algorithm can be modified for any angle rotation by modifying rotation matrix ”R”.