DIP Assignment 1 Report

Name: Meetakshi Setiya
Roll no: 2019253

Q2

The same code has been used for both part b and c. Just run the matlab file and outputs for both parts (matrix in the command window for part b and images for part c) would be displayed.

A brief overview of the code and the assumptions I used is as follows:

1. If the input image I has dimensions MxN, the output image would have dimensions = max(M, floor(M*c)) x max(N, floor(N*c)), where c is the interpolation ratio. Thus, when c<0, the image would have a black outline around its two sides which can give a better and visible idea about the size reduction.
2. If the input mapping of any output coordinate coincides exactly with any input image coordinate (integer), that output pixel would take the value of the corresponding input pixel.
3. If a mapped output pixel falls entirely outside the input image boundary, zero padding comes to play and 0 value is assigned to that output pixel.
4. To calculate the value that an eligible output pixel (x,y) takes, I considered 4 of its nearest neighbors- (floor(x), floor(y)), (floor(x), floor(y)+1), (floor(x)+1, floor(y)), (floor(x)+1, floor(y)+1). V is constituted by the pixel values of these nearest neighbors.
5. If any of these neighbors fall out of the input image boundary, they are given 0 pixel value (zero padding). Thus, if a pixel lies on the boundary or just outside the boundary such that it has at least two neighbors that belong to the image, bilinear interpolation is done.
6. In case of corner pixels (total 4 in number), since these fall outside the image boundary and have just one neighbor coordinate that belongs to the image, they are assigned 0 value.
7. After creating V and X, A is calculated by finding inverse(X).V
8. The pixel value at the output coordinate is then calculated by substituting values in the equation ax+by+cxy+d (where a, b, c, d are from A).
9. Origin is top left and 0 indexing is done.

Part b.

Here is the output from my code:

```
Q2 b.
Input Matrix:
     2     0     0     0
     0     1     3     1
     3     0     2     0

Matrix after interpolation:
    2.0000         0         0         0
    1.0000    1.3333    1.3333         0
    1.0000    0.2222    0.2222         0
```

Yes, the interpolation result at (1,1) and (2,2) matches with the computed result (1.33 and 0.22, found after rounding off) in a.

Part c.

Here are the output images from my code:

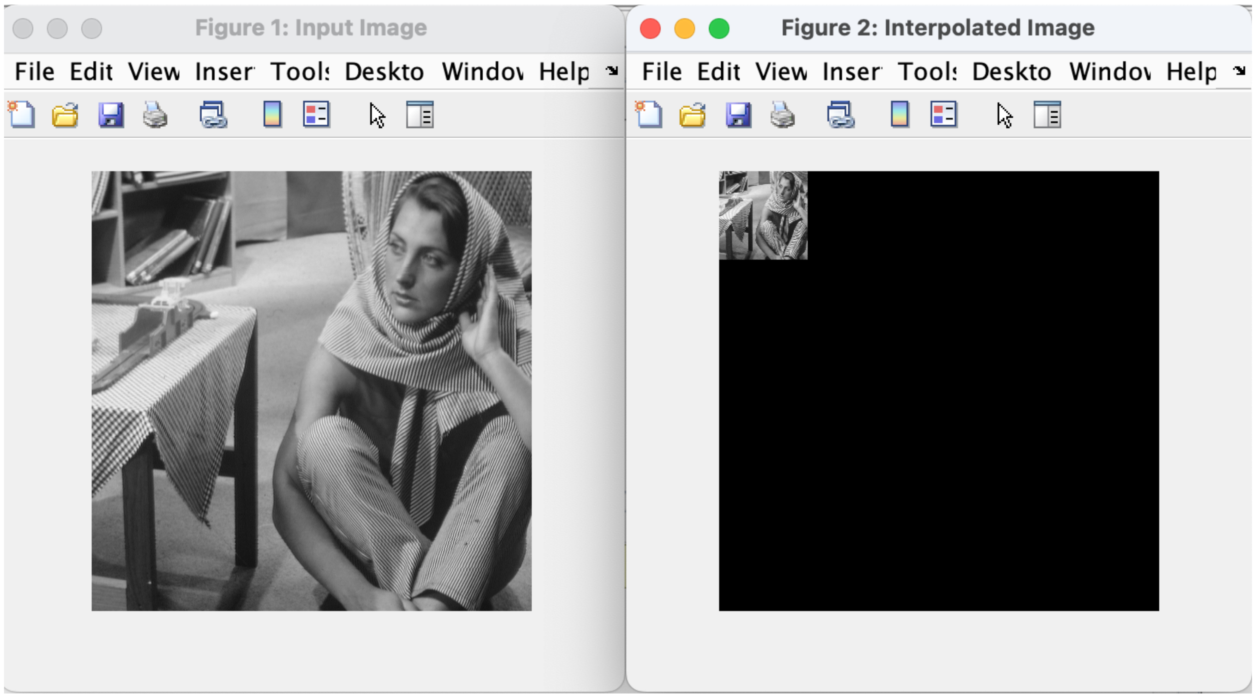
Figure 1: Input Image
File Edit View Inser Tool: Deskto Windov Help
Figure 2: Interpolated Image
File Edit View Inser Tool: Deskto Windov Help

The interpolated image itself has dimensions = 103×103 (not including the black border), whereas the input image has dimensions = 512×512.