Digital Image Processing: Assignment 3

Vaibhav Sharma: T23156

September 9, 2024

1 Interpolation

Appling decimation(downsampling),nearest neighbor interpolation, bilinear interpolation and bicubic interpolation and analyzing results.

1.1 Decimation (Downsampling)

Decimation, or downsampling, involves reducing the resolution of an image. If an image is represented as a matrix I of size $M \times N$, downsampling by a factor of k results in a matrix of size $\frac{M}{k} \times \frac{N}{k}$. The downsampling operation can be expressed as:

$$I_{\text{downsampled}}(i, j) = I(i \cdot k, j \cdot k)$$

where i and j are indices of the downsampled image.

(a) Original Image: Shape (256,256)

(b) Downsampled Image: Shape (128,128)

Figure 1: Origianl and Downsampled Image

1.2 Interpolation Techniques

Interpolation is used to estimate pixel values in an image after resizing. We examine three interpolation techniques:

1.2.1 Nearest-Neighbor Interpolation:

This technique assigns the value of the nearest pixel. If (x, y) is the coordinate in the resized image, the interpolated value is:

$$I_{\text{NN}}(x, y) = I_{\text{orig}}(\text{round}(x), \text{round}(y))$$

1.2.2 Bilinear Interpolation:

This technique uses the values of the four closest pixels. The interpolated value at (x, y) is given by:

$$I_{\text{bilinear}}(x, y) = (1 - a)(1 - b)I_{11} + a(1 - b)I_{21} + (1 - a)bI_{12} + abI_{22}$$

where I_{11}, I_{12}, I_{21}, and I_{22} are the pixel values at the surrounding coordinates, and a and b are the fractional parts of x and y coordinates.

1.2.3 Bicubic Interpolation:

This method considers 16 neighbouring pixels and uses cubic polynomials. The interpolated value is calculated using:

$$I_{\text{bicubic}}(x, y) = \sum_{i=0}^{3} \sum_{j=0}^{3} P(i, j) \cdot I_{i,j}$$

where $P(i, j)$ represents the cubic polynomial coefficients and $I_{i,j}$ are the pixel values in the 4x4 neighborhood.

1.3 Results

Below are the images showing the results of downsampling and various interpolation techniques.

(a) Original Image

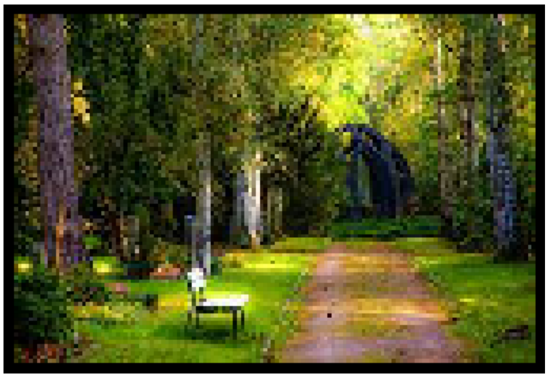
(b) Nearest-Neighbor Interpolation

(c) Bilinear Interpolation

(d) Bicubic Interpolation

Figure 2: Comparison of Interpolation Techniques for Image 1

(a) Original Image



(b) Nearest-Neighbor Interpolation

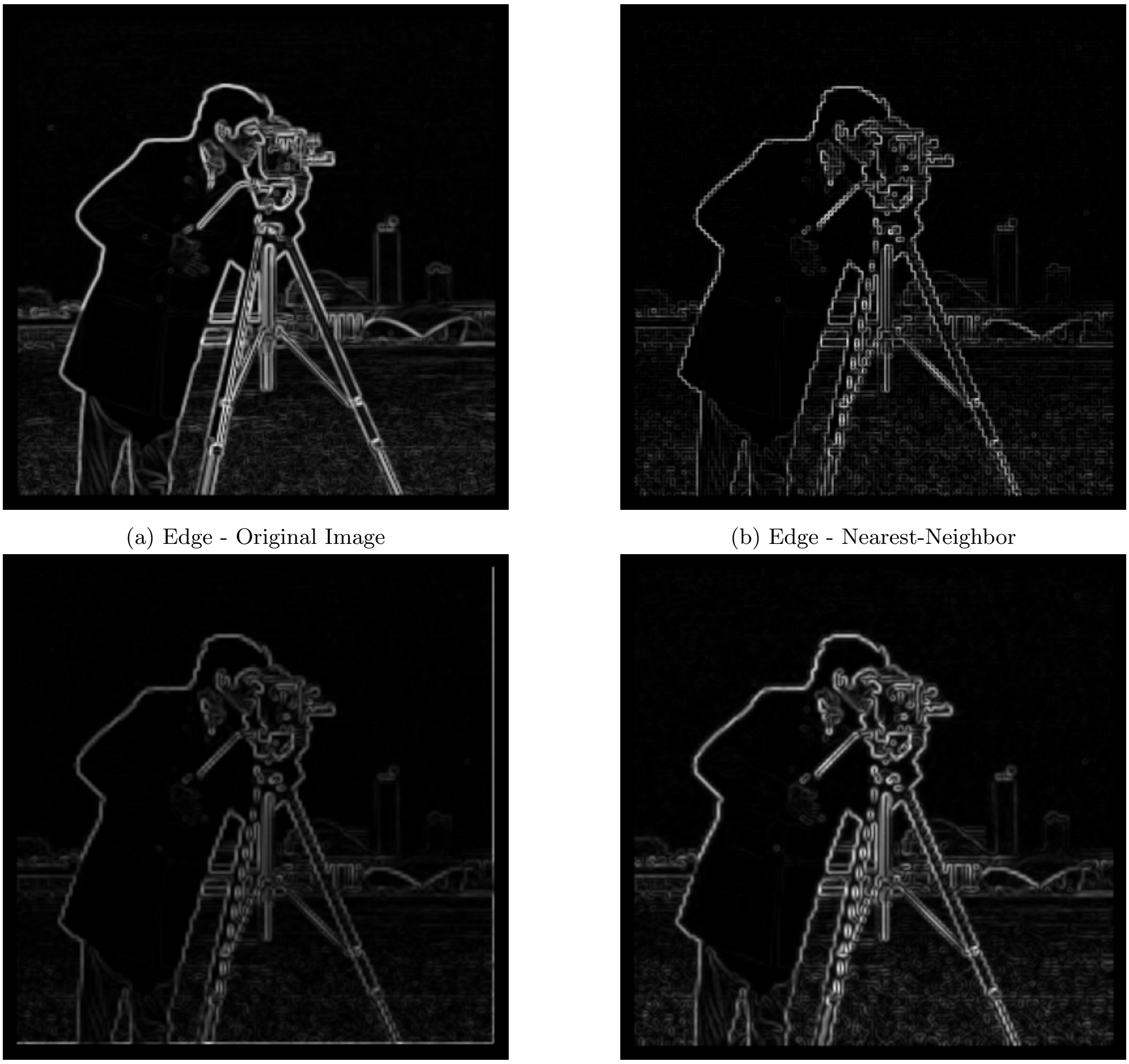
(c) Bilinear Interpolation

(d) Bicubic Interpolation

Figure 3: Comparison of Interpolation Techniques for Image 2

1.4 Edge Detection

Edge detection was applied to the interpolated images. The results are shown below.



(a) Edge - Original Image

(b) Edge - Nearest-Neighbor

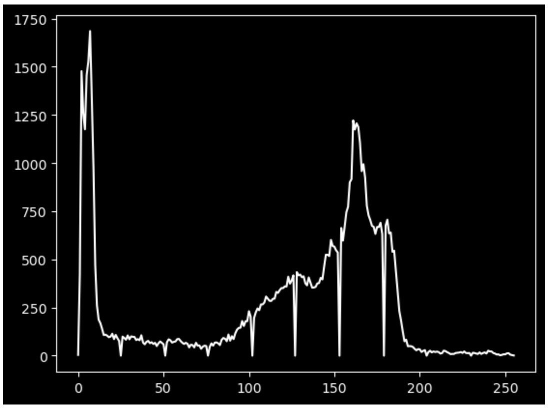
(c) Edge - Bilinear

(d) Edge - Bicubic

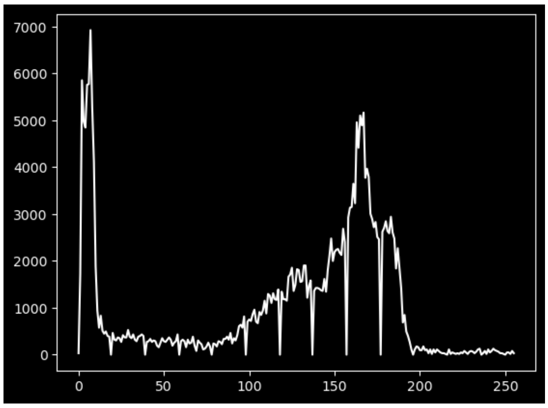
Figure 4: Comparison of Interpolation Techniques for Image 1

1.5 Histograms

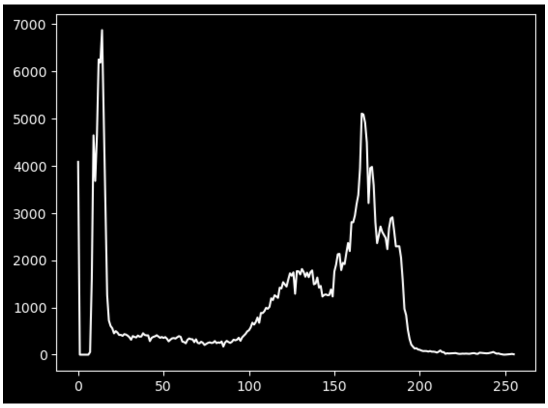
Histograms of pixel intensity values for the interpolated images are plotted below.



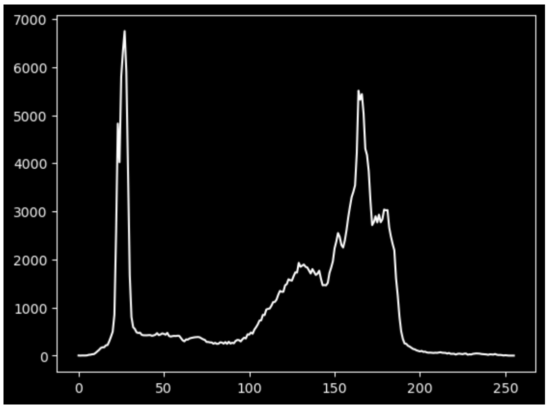
(a) Histogram - Original Image



(b) Histogram - Nearest-Neighbor

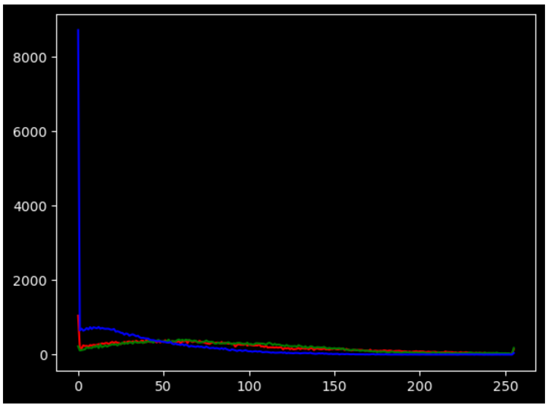


(c) Histogram - Bilinear

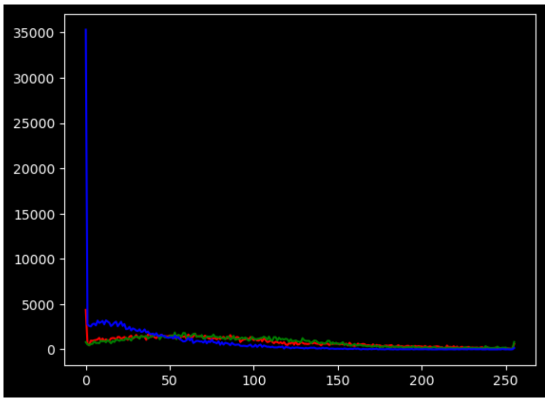


(d) Histogram - Bicubic

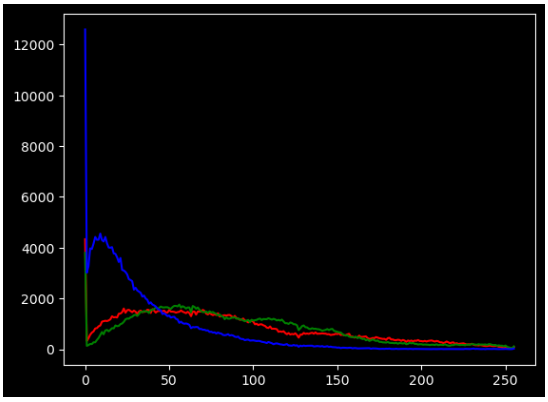
Figure 5: Comparison of Interpolation Techniques for Image 1



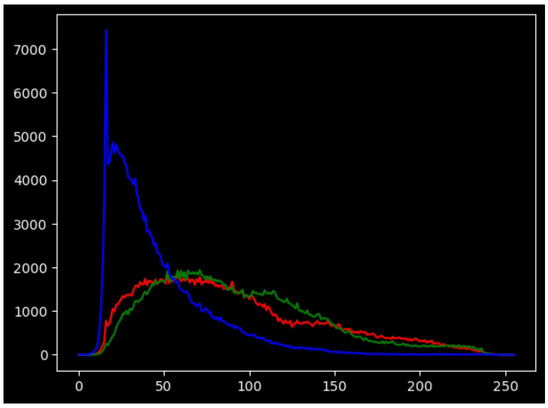
(a) Histogram - Original Image



(b) Histogram - Nearest-Neighbor



(c) Histogram - Bilinear



(d) Histogram - Bicubic

Figure 6: Comparison of Interpolation Techniques for Image 2

1.6 Conclusion

The results of the interpolation methods show varying levels of image quality and edge preservation. Nearest-neighbour interpolation tends to produce blocky images with visible artefacts, while bilinear interpolation provides smoother results but may still introduce some blurring. Bicubic interpolation generally yields the highest quality with more natural-looking results, preserving edges better. In conclusion, bicubic interpolation is often the preferred choice for high-quality image resizing because it produces smoother and more visually appealing results compared to nearest-neighbour and bilinear interpolation methods. The edge detection results and histograms further confirm these observations.

Code Availability

All the code used in this project is available in a public GitHub repository. You can access it at: https://github.com/Computer-Science-Practicum/DIP-Lab-Assignment.