Image Processing Assignment

(Image processing application)

130281M – K. M. C. M. Karunarathne

Introduction

This is an image processing application implemented using Java. This has the ability to add some effects to images and save them using some compression algorithms. More details are given below.

Github - https://github.com/kmchmk1026/Image-Processing

Task 1.

Images are downloaded. They are inside the “images” folder.

Task 2.

8bpp Gray Scale format

This application can convert images from 24bit RGB format to 8bit gray scale format using two methods. First one is using the average level and second one is using luminance preserving method.

Average Gray Scale Conversion

- Take the red, green, blue values of each pixel.
- Calculate their average.
- Average = (red + green + blue) / 3
- Set that average value as all red, green and blue values for the new image.

Luminance Preserving Gray Scale Conversion

- Take the red, green, blue, values for each pixel.
- Calculate their weighted average using following formula.
- Weighted average = (0.2126 * red) + (0.7152 * green) + (0.0722 * blue)
- Set that average value as all red, green and blue values for the new image.

Images can be saved in both 24bit (RGB) and 8bit(Gray Scale) using this application.

- The name extension of saved image files is ".130281M".
- First four bytes of the image file contains the width of the image.
- The next four bytes of the image file contains the height of the image.
- Above three things are same for both 24bit and 8bit formats.

For 24bit format

- Next every 3 bytes packet contain the red, green, blue values of each pixel.

For 8bit format

- Next every byte contains the gray scale value for each pixel.

Also both formats can be opened using this application.

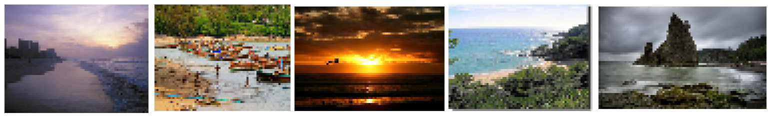
Task 3.

Re-sampling.

- Images can be both down sampled and up sampled.
- Down sampling is done using "Bi-linear interpolation" method and up sampling is done using "Nearest Neighbor" method.

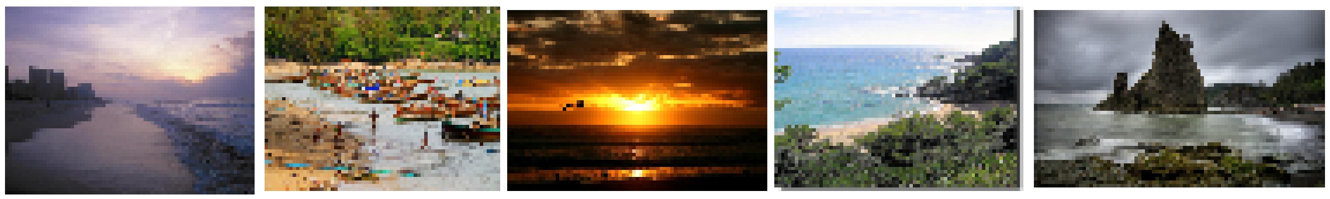
Bi-linear interpolation

- When scaling an image, values for points even in between pixels have to be calculated. It can be calculated using following formula.
- The value for (x,y) pixel in the scaled new image can be calculated using weights of its four neighbors.
- let “a” is the horizontal distance between (x,y) and (i,j).
- let “b” is the vertical distance between (x,y) and (i,j).
- Then the value for (x,y) point F(x,y) can be calculated using following formula.
- $F(x,y) = (1-a)(1-b)F(i,j) + a(1-b)F(i+1,j) + abF(i+1,j+1) + (1-a)bF(i,j+1)$



Nearest Neighbor

- duplicate each row and column and create a double sized image.



Task 4.

Calculate Deviations

Standard Deviation

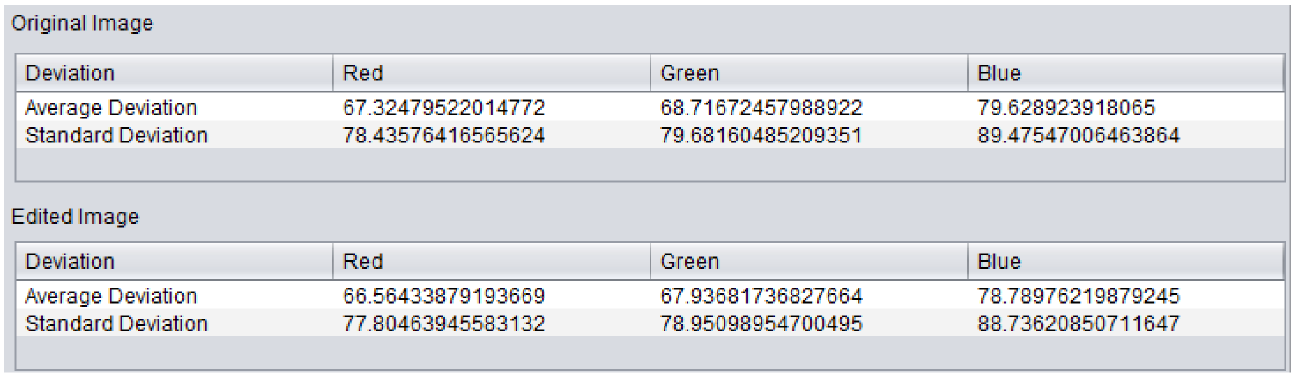
Below is the process of algorithm that has been used.

- First calculate the mean of the image. It is calculated using following formula.
  - Mean = sum of gray levels of a color of the whole image / number of pixels in the image
- Then for each pixel in the image, calculate the difference between mean and the particular value.
- Calculate the sum of all squares of those differences.
- The standard deviation is the square root of the sum.

Average Deviation

Below is the steps for calculating average deviations

- First calculate the mean of the image. It is calculated using the same formula that is given above.
- Calculate the absolute values of differences between the mean and the colour levels.
- Calculate the sum of those absolute values.
- Average deviation is the sum divided by the number of pixels.



Original Image

Deviation	Red	Green	Blue
Average Deviation	67.32479522014772	68.71672457988922	79.628923918065
Standard Deviation	78.43576416565624	79.68160485209351	89.47547006463864

Edited Image

Deviation	Red	Green	Blue
Average Deviation	66.56433879193669	67.93681736827664	78.78976219879245
Standard Deviation	77.80463945583132	78.95098954700495	88.73620850711647

Task 5.

Compression

Entropy Coding

- For the entropy coding, Huffman coding method is used.
- Algorithm is given below.
- Calculate the probabilities of each colour level.
- Build the Huffman tree.
- Generate the symbols.
- Create a bit stream for the image using Huffman symbols.
- Write it to the file.
- File additionally contains image width, image height and the Huffman symbol table

Bit Plane Based Run Length Coding

- The algorithms is as follows.
- Create 8 inverted image arrays.
- Group the 1's and 0's and get the counts as a sequence.
- Get the first bits.
- Write those details to file.
- The file may additionally contains the “number of entries for each bit position”, “number of bytes for an entry”, image width and image height.

Task 6.

Un - Compression

- Both Huffman coding compression and run length coding compression can be uncompressed using this application.