# Project Report

# Image Processing Application

**S.W.Ediriweera**

**130147J**

## Step 1

* Five full colour images of natural scenes have been downloaded from ImageNet (<http://www.image-net.org/>) and saved in ‘step1’ folder.



## Step 2

* Converted the images from 24bpp RGB format to 8bpp Gray scale format using Luminosity method.
* The **luminosity** method averages the RGB values, but it forms a weighted average to account for human perception. We’re more sensitive to green than other colors, so green is weighted most heavily. This method preserves luminance.
* Algorithm –

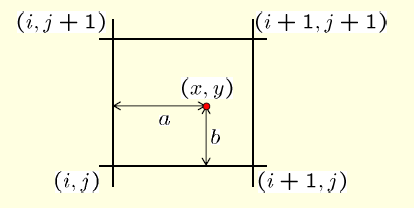
greyImage = 0.2989\*image(:,:,1) + 0.5870\*image(:,:,2) + 0.1140\*image(:,:,3);





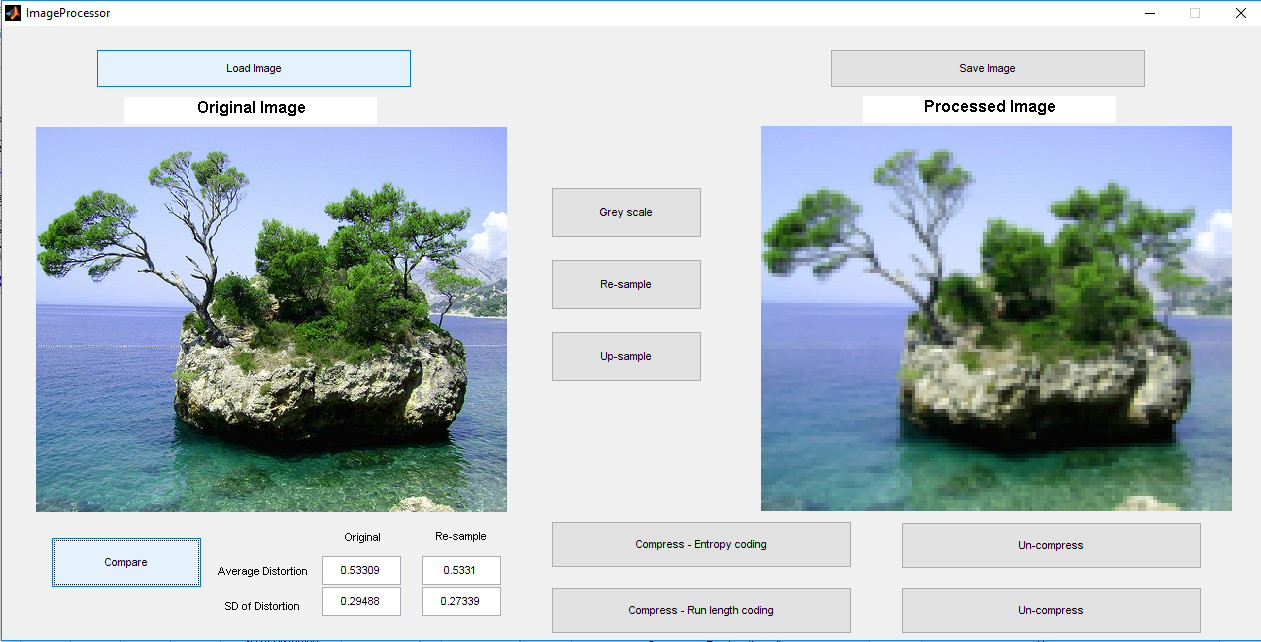
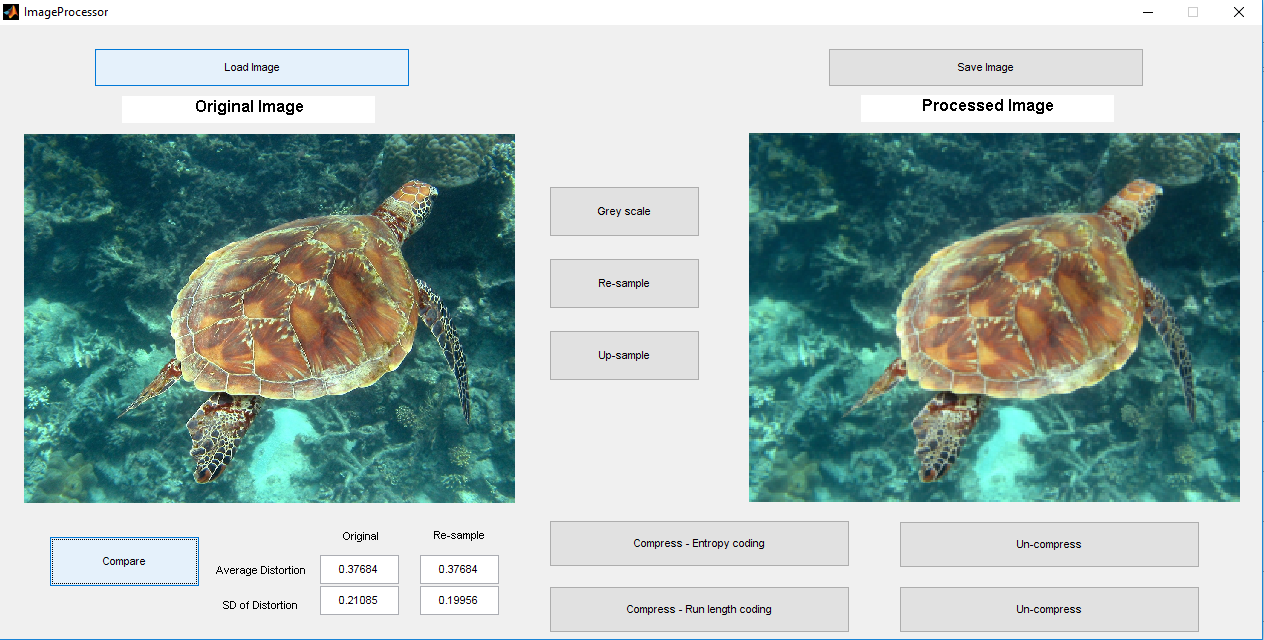


## Step 3

* Images are Re-sampled at 25% scale using bi-linear interpolation method.
* Then the down-sampled images are up-sampled back to their original size using nearest-neighbor method.
* Bi linear Algorithm –
  + RGB values for pixels in the scaled image are calculated.
  + The value for (x,y) pixel in the scaled image can be calculated using weights of its diagonal neighbors.
  + “a” is the horizontal distance between (x,y) and (i,j).
  + “b” is the vertical distance between (x,y) and (i,j).
  + Value of (x,y) pixel is F(x,y).
  + 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 Algorithm –
  + This is implemented by duplicating every pixel rows and columns.
  + Since the width and height was reduced to 25% when resampling, when up-sampling the height and width should be scaled to 400%.
  + Therefore each pixel row and column is duplicated 3 timed.

## Step 4

* Mean of the original image and the resampled image is calculated.
* Standard deviation of the original image and the resampled image is calculated.

**

## 

## 

## 

## Step 5

* The original gray-scale image is taken and compressed using
* Bit-plane based run-length coding
* Entropy coding
* Bit-plane based run-length coding
  + Each bit plane is sliced using –
    - bitPlane = bitget(image2,n);
  + Then for each plane each row is encoded starting from no. of ‘0’s
    - [no. of ‘0’s no. of ‘1’s no. of ‘0’s no. of ‘1’s ……]
  + Then saved as a ‘.data’file.

## Step 6

* The compressed gray-scale image is taken and uncompressed.
* The ‘.data’file is loaded.
* ‘0’s and ‘1’s are added to the image plane.
* The 8 planes are combined.
  + uncompressedImage = uncompressedImage+decodedPlanes(:,:,n).\*(2^(n-1));