CS 663 : Digital Image Processing : Assignment 1

Instructor : Suyash P. Awate

Due Date : 11 Aug 2016, Thursday, 11:55 pm

Instructions for submission are at `www.cse.iitb.ac.in/~suyash/cs663/submissionStyle.pdf`

The folder structure is at `www.cse.iitb.ac.in/~suyash/cs663/assignment1_GrayTrans.zip`
Note: The input data / image(s) for a question is / are present in the corresponding data/ subfolder.

5 points are reserved for submission in the described format.

For all questions, display images using a colormap that uses at least 200 colors / intensities. Also, display the colormap alongside each image. Don't round the resulting images to integers; use a floating-point data type for the pixel value.

1. (15 points) Image Resizing.

   (a) (3 points) Image Shrinking.
   Input image: `1/data/circles_concentric.png`.
   Assume the pixel dimensions to be equal along both axes, i.e., assume an aspect ratio of 1:1 for the axes.
   Shrink the image size by a factor of d along each dimension using image subsampling by sampling / selecting every d-th pixel along the rows and columns.
   • Write a function `myShrinkImageByFactorD.m` to implement this.
   • Display the original and subsampled images, with the correct aspect ratio, for $d = 2$ and $d = 3$ appropriately to clearly show the Moire effects. Display the pixel units along each axis and the colorbar.

   (b) (6 points) Image Enlargement using Bilinear Interpolation.
   Input image: `1/data/barbaraSmall.png`.
   Assume the pixel dimensions to be equal along both axes, i.e., assume an aspect ratio of 1:1 for the axes. Consider this image as the data. Consider the number of rows as M and the number of columns as N.
   Resize the image to have the number of rows $= 3M - 2$ and the number of columns $= 2N - 1$, such that the first and last rows, and the first and last columns, in the original and resized images represent the same data.
   Use bilinear interpolation for resizing.
   • Write a function `myBilinearInterpolation.m` to implement this.
   • Display the original and resized images, without changing the aspect ratio of objects present in the image. Display the pixel units along each axis and the colorbar.

(c) (6 points) Image Enlargement using Nearest-Neighbor Interpolation.
Redo the previous problem using nearest-neighbor interpolation.
- Write a function `myNearestNeighborInterpolation.m` to implement this.
- Display the original and resized images.

2. (80 points)

Input images: (1) `2/data/barbara.png`, (2) `2/data/TEM.png`, and (3) `2/data/canyon.png`.

Assume the pixel dimensions to be equal along both axes, i.e., assume an aspect ratio of 1:1 for the axes.

For each of these three images, do the following. For the color image, apply the analysis independently to each channel (Note: this is a suboptimal way of processing color images, in general; some of the reasons for which will be evident from the results that you will get).

(a) (5 points) Linear Contrast Stretching.
Design a linear grayscale transformation function to enhance the intensity contrast such that the resulting intensity range is $[0, 255]$.
- Write a function `myLinearContrastStretching.m` to implement this.
- Show the formula (or the pseudo code) for the linear function in the report.
- Display the original image and the contrast-enhanced image, without changing the aspect ratio of objects present in the image. Display the colorbar.

(b) (20 points) Histogram Equalization (HE).
Perform (global) HE on the entire image.
- Write a function `myHE.m` to implement this.
- Display the original image and the contrast-enhanced image.

(c) (25 points) Adaptive Histogram Equalization (AHE).
Perform AHE on the entire image. Tune the window size for the local analysis to an appropriate value in order to perform contrast enhancement for objects in the image, while preventing excessive noise amplification. For pixels close to the boundary, where the window falls outside the image, use only the pixels that are within the window and within the image (i.e, effectively cropping the window to restrict it within the image boundaries).
- Write a function `myAHE.m` to implement this.
- Display the original image and the contrast-enhanced image.

Redo AHE with neighborhood sizes chosen to be (i) significantly larger and (ii) significantly smaller than that chosen for the previous result in order to clearly show the effects of (i) low contrast improvement and (ii) excessive noise amplification.
- Display the additional two contrast-enhanced images.

(d) (30 points) Contrast-Limited Adaptive Histogram Equalization (CLAHE).
Perform CLAHE on the entire image. Manually tune the window-size parameter and the histogram-threshold parameter $\in [0, 1]$ to achieve a result better than that obtainable using AHE.
- Write a function `myCLAHE.m` to implement this.
- Display the original image and the contrast-enhanced image.

Redo CLAHE with (i) the same window size as before and (ii) histogram-threshold parameter set to a value that is half of the value tuned before.

- Display the additional contrast-enhanced images.