EE610 Image Processing

Assignment 2 – Image enhancement and restoration

Instructions:

1. Use python and libraries such as numpy, matplotlib, and OpenCV.

2. Submit your assignment, preferably a single .ipynb file, by September 4, 2022, 11:59pm on Moodle at: https://moodle.iitb.ac.in/mod/assign/view.php?id=90582

3. In your assignment, show not only the final operations that worked, but also the results of one or two other things that you tried that did not work.

4. Include copious comments about your thoughts in choosing the operations for a particular problem, and your observations about what worked and what did not work, including possible reasons.

5. Comment every line of code to demonstrate understanding of how the code works. Write as much code as you can on your own, but you can use functions from the aforementioned libraries.

6. Cite sources referred at the place where they were used, and include a reference section at the end.

Problems:

1. Take a night time photo with your phone camera (if your phone camera is already very good, then borrow someone else's phone) and try to enhance it by manipulating its histogram and reducing noise. It should still look natural and like a night time photo, but more details should be apparent than before. [2]

2. Remove the newspaper-ink-dot effect and try to make the image at https://momofilmfest.com/wp-content/uploads/2020/01/newspaper-dots.jpg more natural-looking. [2]

3. Triton is the largest of Neptune's satellites (moons), and is the most unusual in our solar system as it orbits its planet in the opposite direction to the planet's rotation. One of its images taken by Voyager 2 in 1989 is hosted at https://www.wired.com/images_blogs/thisdayintech/2009/08/triton_voyager2.jpg. This

picture seems to have some subtle horizontal (and possibly vertical) scan lines. Enhance this image by removing the scanning artifacts. Also try to reveal more details by manipulating the histogram. You can choose to work on a sub-image or a gray scale version, but for full marks maintain the color balance (ratio of R:G:B at each pixel) and work with all three channels and show the final result on the entire (or a large portion of the) image. [3]

4. A picture of a car blurred due to the relative motion of the camera is given at https://www.ee.iitb.ac.in/~asethi/Dump/MakeNumberPlateReadable.jpg. Restore and enhance the picture to reveal the numbers and letters on the number (license) plate. [3]

5. ML-based image restoration:

a. Select a few images from your personal collection, and divide them into good and bad images [1]

b. For bad images, try to figure out the degradation process (contrast, brightness, blurring, noise) [2]

c. Split the good images into training and validation sets (by image) [0]

d. Degrade the good images using the degradation process - contrast and brightness change, blurring, noise. [2]

e. Create a table of size NxW^2 or $Nx3W^2$ where N is the number of patches of size WxWx1 or WxWx3 mined from the degraded versions of the training images. [2] Correspondingly, mine their associated original central pixel of size Nx3 or Nx1. Tips:

i. You may only want to mine a few patches from random locations from each image, instead of mining all possible patches

ii. You may want to predict the residual of the difference between the original and degraded central pixel, and add it later to the degraded image

iii. You may want to first change the contrast and brightness of the original image, because it cannot be predicted very well from this machine learning process

f. Similarly, create tables of size MxW^2 or $Mx3W^2$ and Mx3 or Mx1 for the validation images [0]

g. Train a regression model, e.g. support vector machine regression with RBF kernel and vary the hyperparameters to check the performance on the validation dataset. You can modify and use the following commands: [3]

i. from sklearn.svm import SVR

ii. regressor = SVR(kernel='rbf', C=1, epsilon=0.1, gamma='scale') #change hyper parameters C, epsilon, and gamma as powers of 10 from 1e-2 to 1e2 to see which combo works best

iii. regressor.fit(X,y) # Here, X is your NxW^2 or $Nx3W^2$ array, and y is Nx1 or Nx3 array

iv. y_pred = regressor.predict(X_val) #X_val is MxW^2 or $Mx3W^2$ array

v. # Now compute MSE between y_pred and y_val, where y_val is Mx1 or Mx3 array

h. Now apply this model (regressor.predict) to a few of your bad images (every overlapping patch of size WxW) to see if these can be restored. [2]

i. Check which value of W gives good results. [2]