EE 610 Image Processing (July-Nov 2018)

Assignment 1 : Basic Image Editor

- Build a form (GUI) in with the following elements [6]:
    - Image display area
    - Image load button that opens a file selector.
    - It should be able to handle color as well as grayscale images. Color images should be converted to HSI/HSV or Lab, and only the I/V/L channel should be manipulated.
    - Several image manipulation buttons that do the following to the image being displayed:
        - Equalize histogram
        - Gamma correct (ask for input gamma upon pressing the button)
        - Log transform
        - Blur with a mechanism to control the extent of blurring
        - Sharpening with a mechanism to control the extent of sharpening
    - Undo last change
    - Undo all changes (revert to original image)
    - Save current image button
- Add one more feature that is not listed above. You can be as creative as you want. [1]
- FOR TWO-PERSON TEAMS ONLY:
    - Compute 2-D DFT and display magnitude and phase
    - Load a frequency magnitude mask
    - Compute and display modified image using the mask
    - Add two more surprise features not listed above
- Prepare a report and submit pdf [3]:
    - It should have the following sections:
        - Abstract (this should be a summary of the report)
        - Introduction (this should be an introduction to the problem being solved and a brief overview of the approach)
        - Background read (this should include description of other tools or techniques that you read)
        - Approach (this should contain the main design philosophy that you followed and description of features implemented and why)
        - Selection of test images (this should tell which aspect of your app was tested using which image and why)
        - Results on test images (this should original image, transformation steps, and final output for each test image)
        - Discussion (this should talk about main challenges faced, and what more you would do if you had more time)
        - Reference (this should mention all sources from which you learned, including other code URLs on the net that you referred. Also, don't forget to cite these sources in the previous sections where they are relevant)
        - Appendix with code
            - Using coding best practices, e.g.:
                - Proper indentation
                - Informative variable names
                - Comment about every single line of the code explaining the role of each variable, e.g.

```
## Initialize Co-occurrence matrices

poolFetSize = poolImSize*poolImSize*poolImDepth # Pooled feature size is the
product of pooled image squared and pool image number of channels (depth)

fets2Keep = min(fets2Keep,poolFetSize) # Just in case we chose too many
features

coOcPos = np.zeros((fets2Keep,fets2Keep)) # Initialize positive co-occurrence
matrix
coOcNeg = np.zeros((fets2Keep,fets2Keep)) # Initialize negative co-oc matrix


## Iterate through image pairs

for nIter in range(0,nBatches): # For all batches
    for nSample in range(0,nTrainBatchSize): # for all samples in a batch
        img1Class = 100 # some large number
        while img1Class > 5: # We only want to train for the first five
            img1Num = np.random.randint(nImages) # pick a random image
            img1Class = clas[img1Num,0] # Find its class

        img2Class = 100 # Pick class 100
        while img2Class != img1Class: # Loop till the classes of the two images
are the same
            img2Num = np.random.randint(nImages) # Pick a random second image
            img2Class = clas[img2Num,0] # Find its class
```

- Notes:
    - You can look at code available on the Internet, but do not directly copy. Customize.
    - Comments in the code should be your own.
    - F grade for copying from another team. Discussing with other teams is fine.
    - All image processing functions e.g. histogram computations, power-law transformations, FFT computations etc. have to be programmed from scratch instead of using in-built functions (from libraries).
    - Bonus marks will be given for writing good and efficient code. See vectorization example for code efficiency: http://ufldl.stanford.edu/wiki/index.php/Vectorization
    - More credit will be given for making common sense assumptions about things left unsaid in the assignment statement. For example, you should automatically handle display of negative pixels or images with pixels out of the display range.
    - Reports should look professional. Check out IEEE paper format: https://www.ieee.org/content/dam/ieee-org/ieee/web/org/conferences/Conference-template-A4.doc or https://paginas.fe.up.pt/~jca/wrsc/templates/IEEE-Conference-A4-format.pdf . You can use LaTeX by opening a free account on overleaf.com or sharelatex.com