E9 241: Digital Image Processing (2020)
Assignment 1

Due Date: 1st November, 2020

Note: In the all functions file (.m/.py) write your own function according to the underlining questions.

1) **Histogram Computation:** Compute the histogram of the image coins.png. Verify your result using the MATLAB built-in function **hist** (or the corresponding function in python if you are using python). Submit a function named **compute_hist.m** (**compute_hist.py** for python).

   - Input to the Function : path to grayscale image and number of bins (N).
   - $\mathrm{Output\ of\ the\ Function}$: bin centres and corresponding frequencies (both from your code and built-in function).

2) **Otsu's Binarization Algorithm:**

   Part (a)

   In class, we showed that $\sigma_w^2(t) + \sigma_B^2(t) = \sigma_T^2$, where t is the threshold for binarization. Compute the threshold t for the image **coins.png** by:

   a. Minimizing the within class variance σ_w^2
   b. Maximizing the between class variance σ_B^2

   - Verify that (a) and (b) are equivalent.
   - Compare the time taken by them to compute t.

   Submit a function named **otsu_threshold.m(or .py)**.

   - Input to the function : path to grayscale image.
   - Output of the function : thr_w (threshold from part a.), thr_b (threshold from part b), time_w (time taken by part a), time_b (time taken by part b) and bin_image (binarized image).

3) Given the image **"quote.png**", separate the foreground text from the background using binarization. Thereafter display the foreground text in the background image given as '**background.png'**. Write a function named **change_background.m(/ .py)** for this task such that:

   Input to the function : paths to "quote.png" and " background.png".

   Output of the function : the foreground text in "quote.png" superimposed in "background.png".

4) Design a function in matlab/python **count_connected_components.m(/.py)** which counts the number of characters excluding punctuations in the image " **quote.png**".

   Input to the function : path to "quote.png".
   Output of the function : number of characters

5) Write a function **binary_morphology.m(/.py)** which takes the input the noisy image **"noisy.png",** performs binarization to separate the foreground text from noisy background and returns a clean binary image of the text without noise.

   Input to the function : path to "noisy.png".
   Output of the function : Cleaned image.

6) **Bonus Question** : Maximally stable extremal regions (MSER): Maximally stable extremal regions correspond to regions of connected components which when thresholded around a certain threshold are stable in terms of the size of the component. This allows an adaptive thresholding method where different regions can be thresholded using different thresholds automatically.

   Determine the binarized image for "mser.png" based on MSER using the following steps:

   1. Sweep over all thresholds.

   2. For each threshold, determine connected components in the image.

   3. A connected component is termed an MSER if the size of the component does not change much (within δ) for a small perturbation ϵ in the choice of the threshold. Note that both δ and ϵ are parameters that need to be chosen. Determine the stable threshold for each connected component.

   4. Ignore extremely large or extremely small connected components in the above analysis.

   Compare the performance of Otsu's binarization with the above method on "mser.png". Think about why Otsu will not be able to extract both the words in this case.

   Design a function named **count_mser_components.m(/.py)** which takes in the following input and gives the output in the mentioned order:

   Input to the function : path to "mser.png".
   Output of the function : mser_binary_image, otsu_binary_image, num_mser_components, num_otsu_components.