ECE558 Project 01

Due 09/24/2019

How to submit your solutions: put source code folder [your_unityid_code], your report (word or pdf) and results images (.png) if had in a folder named [your_unityid_project01] (e.g., twu19_project01), and then compress it as a zip file (e.g., twu19_project01.zip). Submit the zip file through **moodle**.

Problem 1 (40 points): Visualizing and checking the smoothness prior.

Given an image I, consider all valid pairs of neighboring pixels, compute the difference between their intensity or color values, and plot the histogram.

- **Neighbors**: e.g., for a 100x100 image, consider all valid pairs of (x, y) and $(x+1, y)$ where $x, y \in [0, 100)$. Your code should be flexible to handle different types of neighbors, e.g., (x, y) and $(x, y+1)$, or (x, y) and $(x+1, y)$; In implementation, you can define the set of different types, and provide an input argument to select one type from the set.
- **Difference**: using the squared of difference for intensity, RGB, HSV and Lab. E.g., consider gray image, the difference is calculated by

  $\big(I(x,y) - I(x+1,y)\big)^2$ or for 3-channel color images

  $$\sum_{c=0}^{2}\big(I_c(x,y) - I_c(x+1,y)\big)^2$$

  You can use built-in functions for color conversion. You can also implement those by yourself (but, at no extra credit).

- **Histogram**: you can use built-in functions to visualize the histogram.

Your code needs to self-contained. Evaluating your code on the provided wolves.png (used in HW01) is required. Record the run time of your code. You can also report results on other images (but, at no extra credit).

Remarks: for grading, we need at least 2 histograms for each of the 4 types of differences (intensity, RGB, HSV, and Lab). The 2 histograms can be those plotted for (x, y) and $(x, y+1)$, and (x, y) and $(x+1, y)$ respectively. The code needs to be able to handle other types of pairs, e.g., (x, y) and $(x+1, y+1)$. You are encouraged to show more results.

Problem 2 (60 points): Finding the shortest path.

Consider the image segment shown.

$$\begin{array}{rcccl} & 3 & 1 & 2 & 1\,(q) \\ & 2 & 2 & 0 & 2 \\ & 1 & 2 & 1 & 1 \\ (p) & 1 & 0 & 1 & 2 \end{array}$$

(a) (8 points) Let $V = \{0, 1\}$ and compute the lengths of the shortest 4-, 8-, and m-path between p and q. Show the corresponding paths. If a particular path does not exist between these two points, explain why.

(b) (2 points) Repeat for $V = \{1,2\}$.

(c) (50 points) Write python / matlab code to implement the function.
Input arguments: an image, a predefined set V, two pixel locations p and q inside the image, the path type (4-, 8-, or m-path)
Outputs: the length of the shortest path, and the path (i.e., the sequence of pixels)
Test your code: you should test your code thoroughly to make sure it can handle different situations, e.g., a particular path does not exist between the given p and q, as well as invalid input arguments; create at least 3 more different testing examples beside the given one for (a) and (b); record the run time of your code.
Requirement: Built-in functions in Matlab or Python for finding the shortest path cannot be used. You need to implement a self-contained code from scratch. Submit your code together with your results.