The Hong Kong University of Science and Technology
Department of Computer Science and Engineering
CSIT 5410 (SPRING 2021)

Assignment 1

Total = 100 marks
Due: 11:55pm, March. 05, 2021
Assignments must be submitted via Canvas
Late Policy: 10% reduction; only one day late is allowed, i.e., 11:55pm, March. 06.

Overview

In this assignment, you will use Octave/MATLAB to program three functions *myquantizer* for image quantization, *myscaler* for image rescaling and *mysobeledge* for image edge. You need to complete the missing implementations in the corresponding M-files. From the course CANVAS website, you can download the M-files and related files.

Part1 Image Quantization

In this part, you need to complete the function *myquantizer.m* to perform image quantization. The function prototype is "*myquantizer(img, level) → output_img*", where "*level*" is an integer in [0, 255] defining the number of gray levels of output, "*img*" is an image with 256 gray levels. **The built-in function "*imquantize*" is not allowed to use.**

Note that, computers always represent "white" via the pixel value of 255(for uint8). For example, when "level" == 4, the resulting image should contain pixels of {0, 85, 170, 255}, instead of {0, 1, 2, 3}.

Part 2 Image Scaling

In this part, you are required to implement the image rescaling in *myscaler.m.* The function prototype is "*myscaler(img, output_size) → output_img*", where *'img'* and *'output_img'* are two-dimensional matrices storing images, and *'output_size'* is a tuple of (width, height) defining the spatial resolution of output. **The built-in function "*imresize*" is not allowed to use.**

Note that, the new size may not be strictly larger or smaller than the original size. That is, your scaler should be able to resize the input image into an arbitrary size.

Part 3 Edge Detection in the Spatial Domain

You need to complete the function in the *mysobeledge.m* with format *g=mysobeledge(img,T,direction)*, computes the binary edge image from the input image *img*. This function takes an intensity image *img* as its input, and returns a binary image g of the same size as *img* ($m \times n$), with 1's where the function finds edges in *img* and 0's elsewhere. This function finds edges using the Sobel approximation to the derivatives with the assumption that input image values outside the bounds are zero and all calculations are done using double-precision floating point. The function returns *g* with the size $m \times n$. The image *g* contains edges at those points where the absolute filter response is above or equal to the threshold *T*. **The built-in functions "*edge*", "*fspecial*", "*imfilter*", "*conv*" and "*conv2*" are not allowed to use.**

Descriptions of the function input parameters:
img = An intensity gray scale image
T = Threshold for generating the binary output image. If you do not specify *T*, or if *T* is empty ([ ]), *mysobeledge(img,[],direction)* chooses the value automatically according to **Algorithm 1** described below.
direction = A string for specifying whether to look for *'horizontal'* edges, *'vertical'* edges, positive 45 degree *'pos45'* edges, negative 45 degree *'neg45'* edges, or *'all'* edges (maximum of all responses).

Algorithm 1: Automatically determined threshold

1. Initialize T to be the mean between the minimum and maximum values in the response map (the map obtained by applying the edge detection algorithm).
2. Threshold the image using T, which produces two sub-regions G1 and G2:
   G1, consisting of all pixels with intensity values >= T; and
   G2, consisting of pixels with values < T.
3. Compute the average intensity values m1 and m2 for regions G1 and G2.
4. Update the threshold value: T=0.5*(m1+m2).
5. Repeat steps 2 through 4 ten times. (Optional: repeat steps 2 through 4 until the change of T in successive iterations is smaller than 5%.)

Sample run of the programming assignment
The main routine of the assignment including displaying the final results is well written in the csit5410_assign1.m file. After completing all the functions, you are supposed to get four figures on the screen when you run the command below in the MATLAB environment.

```
>> csit5410_assign1;
```