IP Homework Set #1 - Optimal Color Quantization

Homework Due: Thursday **25/10/19**

Task:

Write matlab routines that perform Optimal Quantization to color images using Lloyd-Max algorithm. Given a 24 bits image with color-values in [0..255] in each band (8 bits per pixel per color), represent the image using only K bits per pixel (for all colors), so that the image colors are represented by 2^K different index values.

In your program use and write two Matlab functions :

I) **function [indimg,lut] = cimg2ind (cimg,k)**

(file name is accordingly **cimg2ind.m**)
This routine quantizes a 24 bit color image into a K bit indexed image.

Input: cimg - a 24 bit color image
k – number of allocated bits per pixel.
Output: indimg – an indexed image in the range [0..2^K-1]
lut - a lookup table which maps 2^K index values to color RGB values.
(lut is a matrix of length 2^Kx3 where each entry i contains 3 color values in [0 255]).
Method: Use the Generalized Lloyd method as taught in class.
Apply your quantization in the CIELAB color space. However, the lut will be given in the RGB space.

II) **function cimg = ind2cimg(indimg,lut)**

(file name is accordingly **ind2cimg.m**)
Given an indexed image and a lookup table, this routine returns an 24 bit color image.

Input: indimg – an indexed image in the range [0..2^K-1]
lut - a lookup table which maps the index values to color values.
Output: cimg - a 24 bit color image in the range [0..255] x3

Method: Uses the lookup table lut to map back the indexed image.
Note that the resulting image is usually <u>NOT</u> equal to the original image used in cimg2ind.

In order to check your routines, run the following function:

Function **CompareImages**(imName,k)

```
  img = imread(imName);  % read the image into internal matrix img
  figure(1);  %open a figure window
  image(img); axis image;   % show the image in its actual size
  [indimg,lut] = cimg2ind (img);
  qimg = ind2cimg (indimg,lut);
  figure(2); %open a different figure window
  image(qimg); axis image;
```

Notes:
The algorithm is iterative. You must choose a good stopping criteria and good initial guess.
To reduce runtime, it is forbidden to loops over an image pixels if not necessary.
Try using global matrix operations when possible.

Conversion from RGB to CIELAB and back to RGB can be found in the internet, e.g.
http://ai.stanford.edu/~ruzon/software/rgblab.html

Image files can be downloaded from the course webpage.

Submission: Archive all your files (including m-files and results) into a single file and send to gmail account: **imagproc@gmail.com** before 08/04/10. Do not forget to add to each file with your name and i.d.

Do not forget to put Names and Student I.D. in your documentation.