OpenCV exercises

1. Images –read, write and display; ROIs

a) Read the name of a file containing an image in 'jpg' format and show it in a window, whose name is the name of the file. Test whether the image was successfully read. Display the height and width of the image, on the console.

b) Read a color image in 'jpg' format and save it in 'bmp' format.

c) Read an image from a file, allow the user to select a region of interest (ROI) in the image, by clicking on two points that identify two opposite corners of the selected ROI, and save the ROI into another file.

2. Images – representation, grayscale and color, color spaces

a) Create a grayscale image, having 100(lines)x200(columns) pixels with constant intensity, 100; draw the two diagonals of the image with intensity 255. Display the image.

b) Create a color image, having 50(lines)x200(columns) pixels with constant intensity, 100; draw the two diagonals of the image, one in red color, the other in blue color. Display the image.

c) Read a color image, display it in one window, convert it to grayscale, display the grayscale image in another window and save the grayscale image to a different file.

d) Read an image (color or grayscale) and add "salt and pepper" noise to it. The number of noisy points must be 10% of the total number of image points. Suggestion: start by determining the number of image channels.

e) Read a color image (in RGB format), split the 3 channels and show each channel in a separate window. Add a constant value to one of the channels, merge the channels into a new color image and show the resulting image.

f) Read a color image (in RGB format), convert it to HSV, split the 3 HSV channels and show each channel in a separate window. Add a constant value to saturation channel, merge the channels into a new color image and show the resulting image.

3. Video – acquisition and simple processing

a) Display a video acquired from the webcam (in color) in one window and acquire and save a frame when the user presses the keyboard. Show the acquired frame in another window.

b) Display the video acquired from the webcam (in color) in one window and the result of the conversion of each frame to grayscale in another window.

c) Modify the program developed in b) so that the resulting frames are in binary format (intensity of each pixel is 0 or 255); use a threshold value of 128.

4. Image enhancement – histogram equalization

a) Take a low contrast image and plot its histogram.

b) Enhance the image constrast using:
b1) simple histogram equalization, or
b2) CLAHE,
and show the resulting enhanced images and their histograms.

5. Image enhancement – filtering

Take a noisy image and filter it (try different filter sizes), using:

a) a mean filter;

b) a Gaussian filter;

c) a median filter;

d) a bilateral filter.

6. Edge detection

Detect the edges of an image using:

a) the Sobel filter; try different thresholds;

b) the Canny filter; try different thresholds;

c) compare the outputs of the two filters when the same thresholds are used;

d) the Laplacian filter; try different apertures;
notes:1) in order to visualize the result it may be necessary to rescale the resulting values;
2) to isolate the edges it is necessary to detect the zero crossings in the result.

7. Hough transform – line and circle detection

a) Compare the functionality of HoughLines() and HoughLinesP() OpenCV functions for line detection.

b) Use HoughLines() to detect lines in a binary image; try different parameter values; draw the detected lines on the image, using line().

c) Use HoughLinesP() to detect line segments in a binary image; try different parameter values; draw the detected line segments on the image.

d) Take an image containing coins and use HoughCircles() to detect the coins in the image.