Classic Computer Vision using OpenCV

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 from a file in 'jpg' format and save it in 'bmp' format.

c) Read a color image from a file, show the mouse cursor over the image, and the coordinates and RGB components of the pixel under the cursor. When the user clicks on the mouse, let him modify the RGB components of the selected pixel.

d) 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 100(lines)x200(columns) pixels with yellow color; 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.

d) Implement a simple tracking algorithm for colored objects, using the following steps: 1) take each frame of the video; 2) convert from BGR to HSV color-space; 3) threshold the HSV image for a range of color values (creating a binary image); 4) extract the objects of the selected range (use a bitwise AND operation, using as operands the original and the binary image) .

SOME USEFUL LINKS:

1. https://docs.opencv.org/4.x/d0/de3/tutorial_py_intro.html
2. https://docs.opencv.org/4.x/d3/df2/tutorial_py_basic_ops.html
3. https://matplotlib.org/stable/tutorials/introductory/images.html
4. https://numpy.org/devdocs/user/quickstart.html
5. https://opencv-tutorial.readthedocs.io/en/latest/intro/intro.html
6. https://pythonexamples.org/python-opencv/
7. https://learnopencv.com/getting-started-with-opencv/
8. https://learnopencv.com/how-to-select-a-bounding-box-roi-in-opencv-cpp-python/
9. https://learnopencv.com/mouse-and-trackbar-in-opencv-qui/#mouse-annotation
10. https://docs.opencv.org/4.x/d7/dfc/group__highgui.html
11. https://learnopencv.com/color-spaces-in-opencv-cpp-python/
12. https://docs.opencv.org/4.x/df/d9d/tutorial_py_colorspaces.html
13. https://docs.opencv.org/4.x/dd/d43/tutorial_py_video_display.html
14. https://learnopencv.com/reading-and-writing-videos-using-opencv/
15. https://learnopencv.com/opencv-threshold-python-cpp/
16. https://pyimagesearch.com/category/opencv/

SOME USEFUL FUNCTIONS / ATTRIBUTES:

OpenCV
cv2.imread()
cv2.imwrite()
cv2.cvtColor()
cv2.namedWindow()
cv2.imshow()
cv2.waitKey()
cv2.destroyAllWindows()
cv2.selectROI()
cv2.rectangle()
cv2.line()
cv2.randu()
cv2.threshold()
blue,green,red = cv2.split(imgBGR) #note: BGR
hue,sat,value = cv2.split(imgHSV)
img_merged= cv2.merge((hue,sat,value)) #note: convert to BGR before display
cap = cv2.VideoCapture(0); while(…): ret, frame = cap.read(); … ; cap.release();
NumPy
nLins, nCols, nChannels = img.shape
imgZeros = np.zeros(img.shape,np.uint8)
imgGrayConstValue = np.full((nLins,nCols), value, dtype= np.uint8)
imgColorConstValue = np.full((nLins,nCols,3), (Rvalue,Gvalue,Bvalue), dtype=np.uint8) # note: RGB
Matplotlib
plt.imshow(img); plt.title('Image XXX'); plt.xticks([]), plt.yticks([]); plt.show()
img1 = cv2.cvtColor(img, cv2.COLOR_BGR2RGB); plt.imshow(img1); plt.show(); # Matplotlib expects RGB format