LAB1: Introduction OPENCV

Introduction to OpenCV

OpenCV is considered as the well-known library for computer vision. This library can be used by several programming languages including the python.
In this introduction, we will try to take a look at some of the functions in this library and then try to implement some others by ourselves.

This tutorial will include how to:

1. Read an image
2. Extract the RGB values of a pixel
3. Resize the Image
4. Draw a Rectangle
5. Display text
6. Open and play a video

This is the original image that we will manipulate throughout this tutorial. You can use your image.

Reading an image

```
# Importing the OpenCV library
import cv2

# Reading the image using imread() function
image = cv2.imread("road.jpg")

# Extracting the height and width of an image
h, w = image.shape[:2]
# Displaying the height and width
print("Height = {},  Width = {}".format(h, w))
```

Extracting the RGB values of a pixel

```
# Extracting RGB values.
# Here we have randomly chosen a pixel
# by passing in 100, 100 for height and width.
(B, G, R) = image[100, 100]

# Displaying the pixel values
print("R = {}, G = {}, B = {}".format(R, G, B))

# We can also pass the channel to extract
# the value for a specific channel
B = image[100, 100, 0]
print("B = {}".format(B))
```

Pay attention!!!! In OpenCV, the order of color channel is BGR and not RGB while some libraries consider them as RGB such as matplotlib that allows us also to show the image. In such cases, you should convert from BGR to RGB.

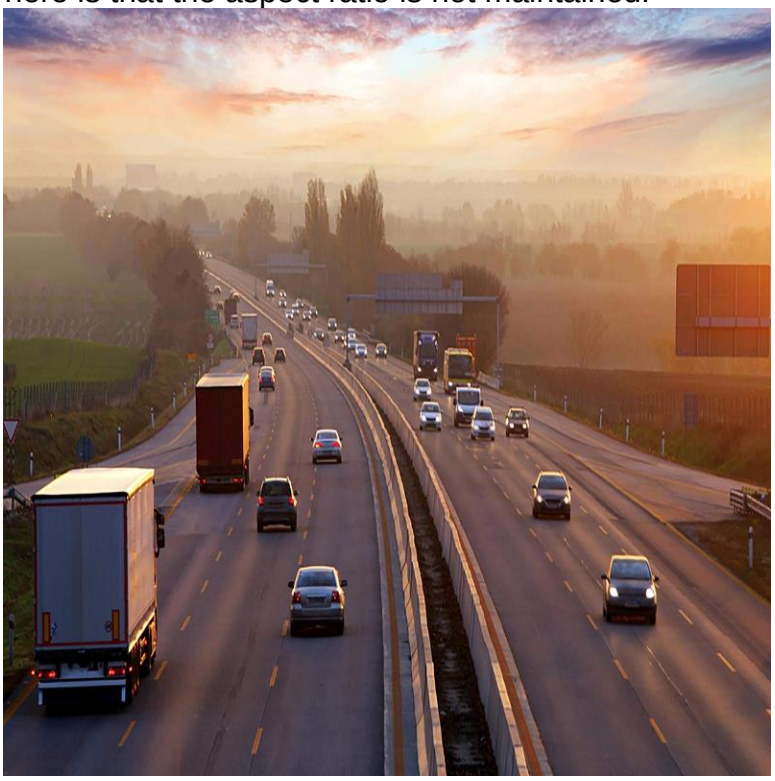
Extracting a sub-region of the image

```
# We will calculate the region of interest
# by slicing the pixels of the image then show the image
roi = image[100 : 500, 200 : 700]
cv2.imshow('image',roi)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Resizing the Image

```
32    # resize() function takes 2 parameters,
33    # the image and the dimensions
34    resize = cv2.resize(image, (800, 800))
35    cv2.imshow('image',resize)
36    cv2.waitKey(0)
37    cv2.destroyAllWindows()
```

The main problem here is that the aspect ratio is not maintained.



Drawing a Rectangle

```
52    # We are copying the original image,
53    # as it is an in-place operation.
54    output = image.copy()
55
56    # Using the rectangle() function to create a rectangle.
57    rectangle = cv2.rectangle(output, (1500, 900),
58                              (600, 400), (255, 0, 0), 2)
59    cv2.imshow('image',output)
60    cv2.waitKey(0)
61    cv2.destroyAllWindows()
```

Displaying text

```
# Adding the text using putText() function
text = cv2.putText(output, 'OpenCV Demo', (500, 550),
                   cv2.FONT_HERSHEY_SIMPLEX, 4, (255, 0, 0), 2)
cv2.imshow('image',text)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Play a video in OpenCV

```
def play_videoFile(filePath, mirror=False):
    cap = cv2.VideoCapture(filePath)
    cv2.namedWindow('Video Life2Coding', cv2.WINDOW_AUTOSIZE)
    while True:
        ret_val, frame = cap.read()

        if mirror:
            frame = cv2.flip(frame, 1)

        cv2.imshow('Video Life2Coding', frame)

        if cv2.waitKey(1) == 27:
            break  # esc to quit

    cv2.destroyAllWindows()


def main():
    play_videoFile('test.mp4', mirror=False)


if __name__ == '__main__':
    main()
```

Exercise I: Image conversion
Given an RGB colored image, write a function that converts the image into gray scale one. The gray value of a pixel having the triplet (R, G, B) is the one value (R+G+B)/3. Compare your result to the command in OpenCV "`img_gray = cv2.cvtColor(img, cv2.COLOR_RGB2GRAY)`".

Exercise II: Image conversion
Given an image that may be an RGB colored image or a grayscale one, write a function that converts the image into binary one.
Compare your result to the command in OpenCV "`(thresh, blackAndWhiteImage) = cv2.threshold(img_gray, 127, 255, cv2.THRESH_BINARY)`".

Exercise III: Plotting each channel separately
In this exercise, we want to take a colored image and plot each channel separately. To do so, the first method is to plot each channel separately as if it is a gray scale image. In the second method, to show the amount of red in each pixel, we leave the red channel untouched and we set the two channels (green and blue) to zero and then we show the image.

Exercise IV: Histogram
We want to compute the histogram of gray scale images. An histogram of a gray scale image is a 256-array that counts how many times a gray value is present in the image. For example, the value at position 0 in the array will contain how many pixels in the image have the gray value 0 and so on. Write a function that compute the histogram of a gray scale image and plot the histogram using the matplotlib as follows:

`from matplotlib import pyplot as plt`
 	
`...`
`plt.plot(hist, 'b')`
`plt.show()`
	We import the pyplot from the matplotlib library and we use it under the name plt. To plot, we call plt.plot. and plt.show to show the plot