Image Acquisition

Antonio J. Sánchez Salmerón

March 2, 2023

- Requirements:
    1. The estimated time for performing this activity is approximately half an hour.
- Objectives of this notebook:
    1. Introduction to OpenCV: learning to acquire images.
- Summary of activities:
    1. The histogram of an image will be analyzed using Python code snippets for acquired images.

Contents

1 Introduction to OpenCV

In Python you always start by importing all the required modules:

[1]:
```
import numpy as np  # the numpy module is associated with the alias np
import cv2          # OpenCV module
import matplotlib.pyplot as plt  # the matplotlib.pyplot with the alias plt
import time                      # module to mesuare time cost
```

[2]:
```
# **** It is important to consult the help of the version you are using ****
!python --version
print('Numpy:',np.__version__)
print('OpenCv:',cv2.__version__)
```

```
Python 3.9.1
Numpy: 1.20.2
OpenCv: 4.5.2
```

1.1 Select video device

[3]:
```
#----------------------------------------------------------------------------
# If you have only one camera, try capturing from device 0
# Each webcam is assigned a numeric identifier
# If you have more cameras, try other devices 1, 2, ...
#----------------------------------------------------------------------------
#camara_dev = 0

#----------------------------------------------------------------------------
# If you have a camera on your cell phone, install an IP camera server.
# In the case of android, install the application "smart webcam".
#
# https://www.portalprogramas.com/smart-webcam/android/
#
# When you run this application on the cell phone, it creates an IP camera␣
 ↪handler
# with a specific ip address and port.
# In my case, "private ip": 192.168.1.200 (0) and "Port": 8082
# Modify the following device in accordance with your server
#----------------------------------------------------------------------------
#camara_dev = 'http://10.5.196.92:8082'
camara_dev = 'http://VxC:vxc2018@158.42.206.2/mjpeg/snap.cgi?chn=0'

print('Selected device:', camara_dev)
```

```
Selected device: http://VxC:vxc2018@158.42.206.2/mjpeg/snap.cgi?chn=0
```

1.2 Image acquisition

[4]:
```
# Open device
video = cv2.VideoCapture(camara_dev)

#video.set(cv2.CAP_PROP_FRAME_HEIGHT, 720)
#video.set(cv2.CAP_PROP_FRAME_WIDTH, 1280)

#video.set(cv2.CAP_PROP_EXPOSURE, -5.0) # 2^-5s = 1/32s
#video.set(cv2.CAP_PROP_EXPOSURE, -8.0)

# Image acquisition: a BGR image is captured
ret_ok, img = video.read()

# Close device
video.release()

if ret_ok:
    print('Image shape: ', img.shape)
    print('Image type:',type(img),'of', img.dtype)
else:
    print('Error: the selected device is not available.')
```

```
Image shape:  (720, 1280, 3)
Image type: <class 'numpy.ndarray'> of uint8
```

1.3 Shows image

[5]:
```
cv2.imshow('Image', img) # Opens a window to show the image
cv2.waitKey(0)           # Waits until key pressed on the window Image
cv2.destroyAllWindows()  # Closes the window
```

1.4 Shows image continously

[ ]:
```
video = cv2.VideoCapture(camara_dev)

cv2.namedWindow('Image') # Opens a window to show the image

key = -1 # to start key must be different to ord('q') = 113

while(video.isOpened() and (key != ord('q'))):

    ret_ok, img = video.read()

    if ret_ok:
        cv2.imshow('Image', img)
        key = cv2.waitKey(1)      # press 'q' key to fininsh the while loop
    else:
        print('End of video.')
        break

video.release()
cv2.destroyAllWindows()
```

1.5 Shows RGB histograms

[ ]:
```
# Split the BGR image into three planes
blue, green, red = cv2.split(img)

# Shows RGB histograms
hist_r = np.zeros((256,))
hist_g = np.zeros((256,))
hist_b = np.zeros((256,))

height, width, channels = img.shape
for x in range(height):
    for y in range(width):
        hist_b[img[x,y,0]] = hist_b[img[x,y,0]] + 1
        hist_g[img[x,y,1]] = hist_g[img[x,y,1]] + 1
        hist_r[img[x,y,2]] = hist_r[img[x,y,2]] + 1

plt.plot(hist_r, color='red')
plt.plot(hist_g, color='green')
plt.plot(hist_b, color = 'blue')
plt.show()
```

2 References

- Python documentation
- Numpy documentation
- Documentación de OpenCV 4.5
- Machine Learning for OpenCV 4 - Second Edition by Vishwesh Ravi Shrimali; Michael Beyeler; Aditya Sharma Published by Packt Publishing, 2019
- Machine Learning for OpenCV 4 : Intelligent Algorithms for Building Image Processing Apps Using OpenCV 4, Python, and Scikit-Learn. by Sharma, Aditya, et al. Second edition., 201

[ ]:
```
```