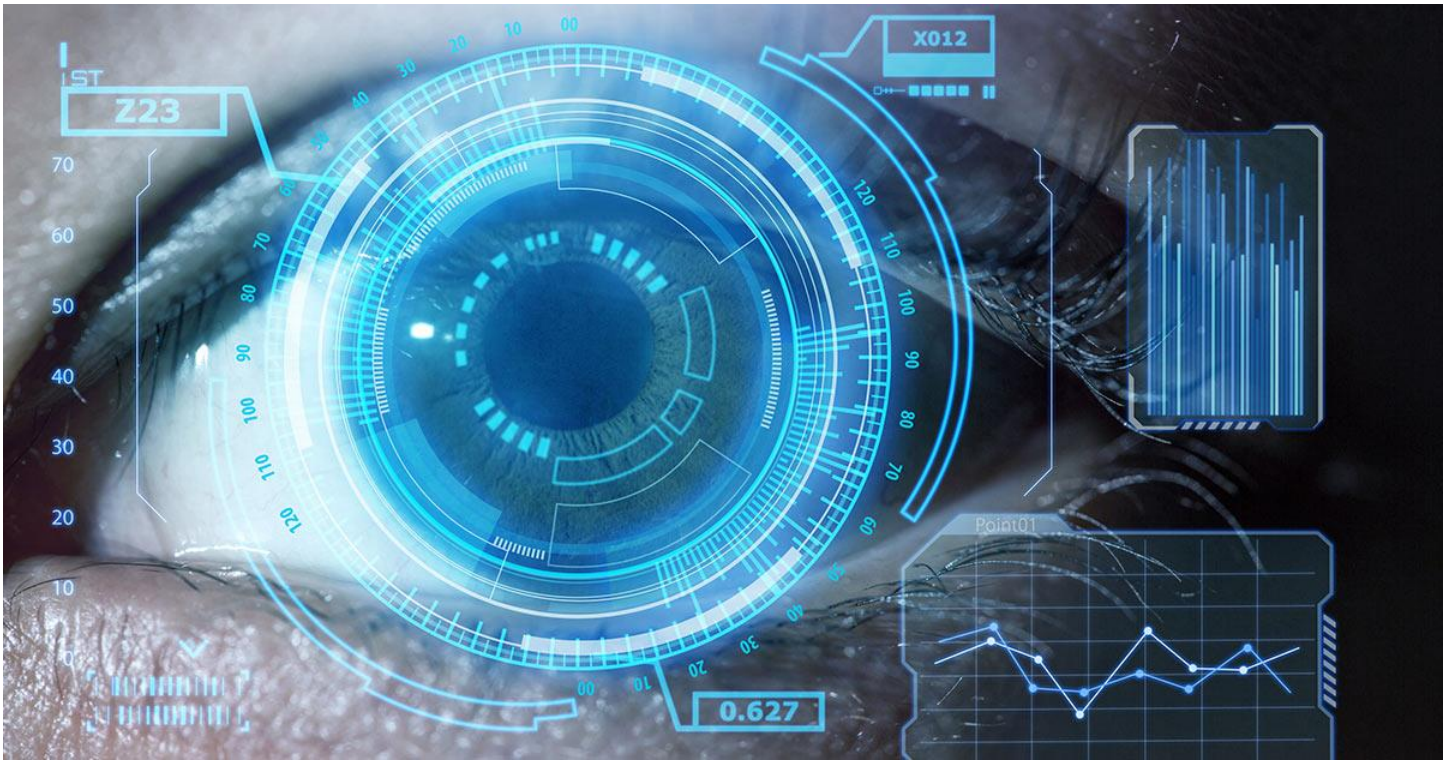
“COMPUTER VISION”

ENG: AHMED MUBARAK

01020451375



SESSION NO.“7”

- OPENCV LIBRARY
  1. Histogram
  2. Find similar images from video
  3. Blurring

ENG.AHMED MUBARAK

01020451375

1. Histogram

```python
import cv2
import numpy as np
import matplotlib.pyplot as plt

# Original #
image = cv2.imread("images/boat.jpg")
image_gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

#? Histogram #

#* Gray #
hist = cv2.calcHist(image_gray, [0], None, [256], [0, 256])

plt.title("Histogram for grayscale image".title())
plt.xlabel("pixel value".title())
plt.ylabel("number of pixels that have this value".title())

plt.plot(hist)
plt.xlim([0, 256])

cv2.imshow("Grayscale", image_gray)
plt.show()

* Colored #


def hist_colored(pic, use_mask=False):
    channels = cv2.split(pic)
    colors = list('bgr')  # <<<<==>>>> ['b','g','r']

    plt.title("Histogram for Colored image".title())
    plt.xlabel("pixel value".title())
    plt.ylabel("number of pixels that have this value".title())

    mask = None
    if use_mask:
        plt.title("Histogram for Colored image (with mask)".title())
        mask = np.zeros(image.shape[:2], dtype="uint8")
        cv2.rectangle(mask, (15, 15), (130, 100), 255, -1)
        masked = cv2.bitwise_and(pic, pic, mask=mask)
        cv2.imshow("Mask", mask)
        cv2.imshow("Applying the Mask", masked)

    for ch, c in zip(channels, colors):
        #! you must put the image in a list if you used a mask
        hist = cv2.calcHist([ch], [0], mask, [256], [0, 256])
        plt.plot(hist, color=c)
```

```
        plt.xlim([0, 256])

    cv2.imshow("The Image", pic)
    plt.show()


hist_colored(image)
# #* with mask #
hist_colored(image, use_mask=True)

# #? Histogram Equalization #
old = cv2.cvtColor(cv2.imread("images/old.jpg"), cv2.COLOR_BGR2GRAY)
eq = cv2.equalizeHist(old)
cv2.imshow('Histogram Equalization', np.hstack((old, eq)))
cv2.waitKey(0)


hist_eq = cv2.calcHist(eq, [0], None, [256], [0, 256])
plt.plot(hist)
plt.title("Histogram equalized image".title())
plt.xlim([0, 256])
plt.show()

hist_old = cv2.calcHist(old, [0], None, [256], [0, 256])
plt.plot(hist)
plt.title("Histogram for the original image".title())
plt.xlim([0, 256])
plt.show()
```

2. Find similar images from video

```
import cv2
import numpy as np
import os
from tkinter import Tk, filedialog

LINE_SEP = '-'*50
COLORS = [
    # BGR
    (255, 0, 0),
    (0, 255, 0),
    (0, 0, 255)
]


def make_video(video_path='./images/', width=320, height=320, video_length_in_seconds=10):
    """
```

```
        أزرق|    أحمر|  أخضر
       ------|---------|---------
        1    |    2    |   3
        4    |    5    |   6
        7    |    8    |   9
        10   |    -    |   -
       ------|---------|---------
        4    |    3    |   3
    """
    codec = cv2.VideoWriter_fourcc(*'mp4v')
    fps = 30
    colored_video = True
    video = cv2.VideoWriter(
        os.path.join(video_path, 'video.mp4'), codec, fps, (width, height), colored_video)
    pixels = np.ones((width, height, 3), dtype=np.uint8)

    print('\nCreating Video...')
    for time in range(1, video_length_in_seconds+1):
        img = pixels*COLORS[time % 3]
        for _ in range(fps):
            video.write(img.astype(np.uint8))

    video.release()
    print('\nVideo Created.', LINE_SEP, sep='\n')


def calc_hist(image):
    gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    return cv2.calcHist(gray, [0], None, [128], [0, 256])


def are_the_same_images(h1, h2, acceptance_ratio=0.65):
    """
     ======================
     |    😁  الزيادة حلوة|
     ======================

        #! cv2.HISTCMP_CORREL: الترابط
        [-1, 1] ترجع قيمة بين
        بحيث
        1: تطابق تام
        -1: لا يوجد تطابق اطلاقا

        #! cv2.HISTCMP_INTERSECT: التقاطع
        [0, 1] ترجع قيمة بين
        بحيث
        1: تطابق تام
        0: لا يوجد تطابق اطلاقا


     ======================
```

```
     |    😣  الزيادة وحشة|
     ======================
        #! cv2.HISTCMP_CHISQR: chi-squared مسافة
        [0, unbounded] ترجع قيمة بين
        بحيث
        0: تطابق تام
        unbounded: لا يوجد تطابق اطلاقا
        unbounded:  فما فوق 10قيمة بعيدة عن الصفر مثلا

        #! cv2.HISTCMP_BHATTACHARYYA: Bhattacharyya مسافةbetween the two
        [0, 1] ترجع قيمة بين
        بحيث
        0: تطابق تام
        1: لا يوجد تطابق اطلاقا
    """

    ratio = cv2.compareHist(h1, h2, cv2.HISTCMP_INTERSECT)

    return ratio >= acceptance_ratio


def count_occurancec_of_frame(video_path, first_apperance_in_second=0.5):

    video = cv2.VideoCapture(video_path)
    fps = int(video.get(cv2.CAP_PROP_FPS))
    occurance = 0

    video.set(cv2.CAP_PROP_POS_FRAMES, int(fps * first_apperance_in_second))
    fetched, requested_frame = video.read()
    requested_hist = calc_hist(requested_frame)

    if not fetched:
        print('INVALID TIME OR VIDEO!')
        return

    print('\nWorking...')
    while True:
        ret, frame = video.read()
        if not ret:
            break

        hist = calc_hist(frame)
        if are_the_same_images(hist, requested_hist):
            occurance += 1


    print("The Frame Appeared {} Times".format((occurance // fps) + 1))
    print(LINE_SEP)
```

```
make_video()
count_occurancec_of_frame('./images/video.mp4', 1.5)
```

3. Blurring

```
import cv2
import numpy as np

image = cv2.imread("images/bug_noisy.jpg")


#? Average Blurring #
def avg_blur():
    #! The Kernel Side Length Must be Odd
    blured = np.hstack([image,
                        cv2.blur(image, (3, 3)),
                        cv2.blur(image, (5, 5)),
                        cv2.blur(image, (7, 7)),
                        cv2.blur(image, (13, 13))])
    cv2.imshow("Average Blur", blured)
    cv2.waitKey(0)


#? Gaussian Blurring #
def gauss_blur():
    # By setting the third parameter to 0, we are instructing OpenCV
    # to automatically compute the weights based on our kernel size
    g_blured = np.hstack([image,
                          cv2.GaussianBlur(image, (3, 3), 0),
                          cv2.GaussianBlur(image, (5, 5), 0),
                          cv2.GaussianBlur(image, (7, 7), 0),
                          cv2.GaussianBlur(image, (13, 13), 0)])
    cv2.imshow("Gaussian Blur", g_blured)
    cv2.waitKey(0)


#? Median Blurring #
def median_blur():
    m_blured = np.hstack([image,
                          cv2.medianBlur(image, 3),
                          cv2.medianBlur(image, 5),
                          cv2.medianBlur(image, 7),
                          cv2.medianBlur(image, 13)])
    cv2.imshow("Median Blur", m_blured)
    cv2.waitKey(0)


#? Bilateral Blurring #
```

```
def bilateral():
    filtered = np.hstack([image,
                          cv2.bilateralFilter(image, 5, 21, 21),
                          cv2.bilateralFilter(image, 7, 31, 31),
                          cv2.bilateralFilter(image, 9, 41, 41),
                          cv2.bilateralFilter(image, 12, 51, 51)])
    cv2.imshow("Bilateral", filtered)
    cv2.waitKey(0)


avg_blur()
gauss_blur()
median_blur()
bilateral()
```

WITH MY BEST WISHES

ENG/AHMED MUBARAK 😉