part1

September 13, 2020

1 Book Flipping

[1]:
```
# User is required to install following librarie
# Numpy for array manuplations
# Opencv for image processing
# Matplotlib for Ploting
import sys
import os
import numpy as np
import cv2
from collections import namedtuple
import matplotlib.pyplot as plt
```

[4]:
```
def path_set(path) -> None:
    """ setting up path where all the videos are store"""
    try:
        os.chdir(path)
    except:
        print("Unexpected Error: ",sys.exc_info()[0])
```

[5]:
```
path_set("C:\\Users\\Anshul\\OneDrive\\Desktop\\Gaurav")
```

[9]:
```
def videoToframe(no_of_videos, list_of_path, frame_rate=0.3) -> None:
    """ With no of videos each has specific path. Defalut frame rate set to␣
↪3fps"""
    curr_path = os.getcwd()
    for i in range(no_of_videos):
        # for each of the input videos
        count = 0
        #for giving new names to each frame
        sec = 0
        new_path = 'frames_'+list_of_path[i][:-4]
        os.mkdir(new_path)
        video = cv2.VideoCapture(list_of_path[i])
        os.chdir(curr_path+"\\"+new_path)

        while(True):
            video.set(cv2.CAP_PROP_POS_MSEC, sec*1000)
            success, image = video.read()
            cv2.imwrite("frame_"+str(count)+".jpg", image)
            count +=1
            sec += frame_rate
            sec = round(sec, 2)

            if not success:
                break
        os.chdir(curr_path)
```

[10]:
```
no_of_videos = 1
list_of_path = ['video_1.mp4']
```

[11]:
```
# convert videos to frame on rate of 3fps
videoToframe(no_of_videos, list_of_path)
```

[12]:
```
def get_img_avg_brightness(path):
    image = cv2.imread(path)
    hsv = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)
    _,_,v = cv2.split(hsv)
    return int(np.average(v.flatten()))
```

[13]:
```
def brightness_graph(path):
        x,y = 0,0
        xaxis, yaxis = [],[]

        for img in os.listdir(path):
                img_path = os.path.join(path, img)

                y = get_img_avg_brightness(img_path)
                #print("the image brightness level of img "+img_path[-6:] +␣
 ↪str(((get_img_avg_brightness(img_path)))))

                yaxis.append(y)
                x = x+1
                xaxis.append(x)

        print("average brightness value: ",sum(yaxis)/len(yaxis))
        plt.plot(xaxis, yaxis)
        plt.plot(xaxis, yaxis,'dm')
        plt.xlabel('images')
        plt.ylabel('Brightness value of image')
        plt.title("Graph of brightness level of the frames")
```

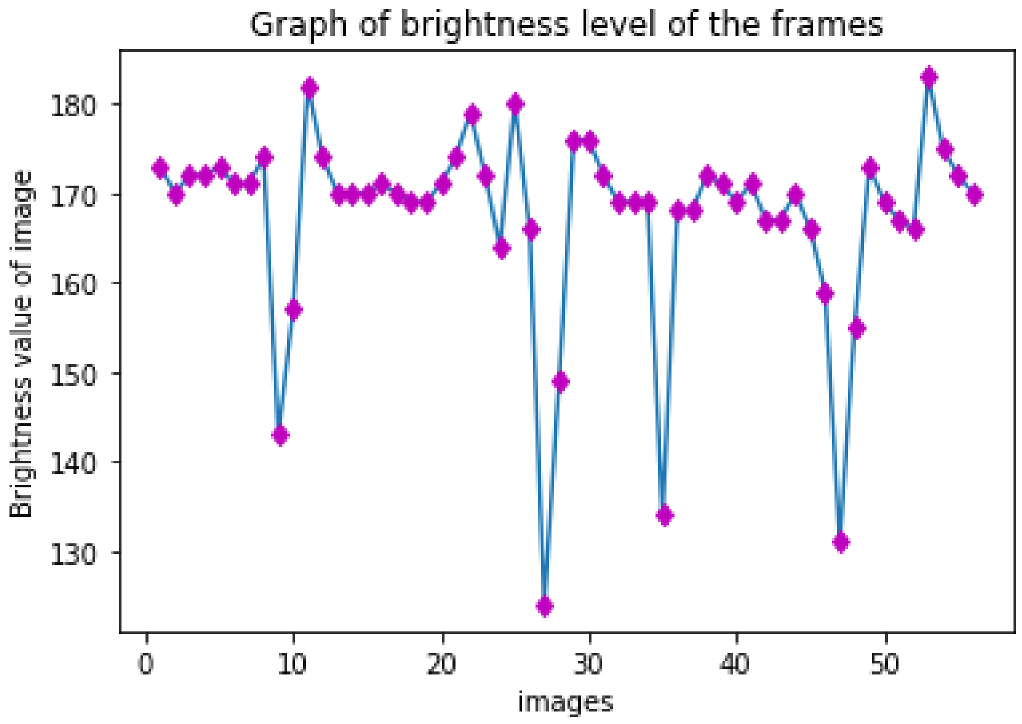
[14]:
```
path = os.getcwd()
work_path = path+"\\"+'frames_'+list_of_path[0][:-4]
```

```
work_path
```

```
[14]: 'C:\\Users\\Anshul\\OneDrive\\Desktop\\Gaurav\\frames_video_1'
```

```
[15]: brightness_graph(work_path)
```

```
average brightness value:  167.39285714285714
```



```
[16]: def skin_pixels_graph(path):
        x,y = 0,0
        xaxis, yaxis = [],[]

        for img in os.listdir(path):
                img_path = os.path.join(path, img)
                image = cv2.imread(img_path)
                image = image.astype('uint8')
                imgYCC = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)

                # Count the pixels having RGB values in defined range
                lower = np.array([0, 48, 80], dtype="uint8")
                upper = np.array([20, 255, 255], dtype="uint8")
                dst = cv2.inRange(imgYCC, lower, upper)
                count = cv2.countNonZero(dst)
                yaxis.append(count)
```

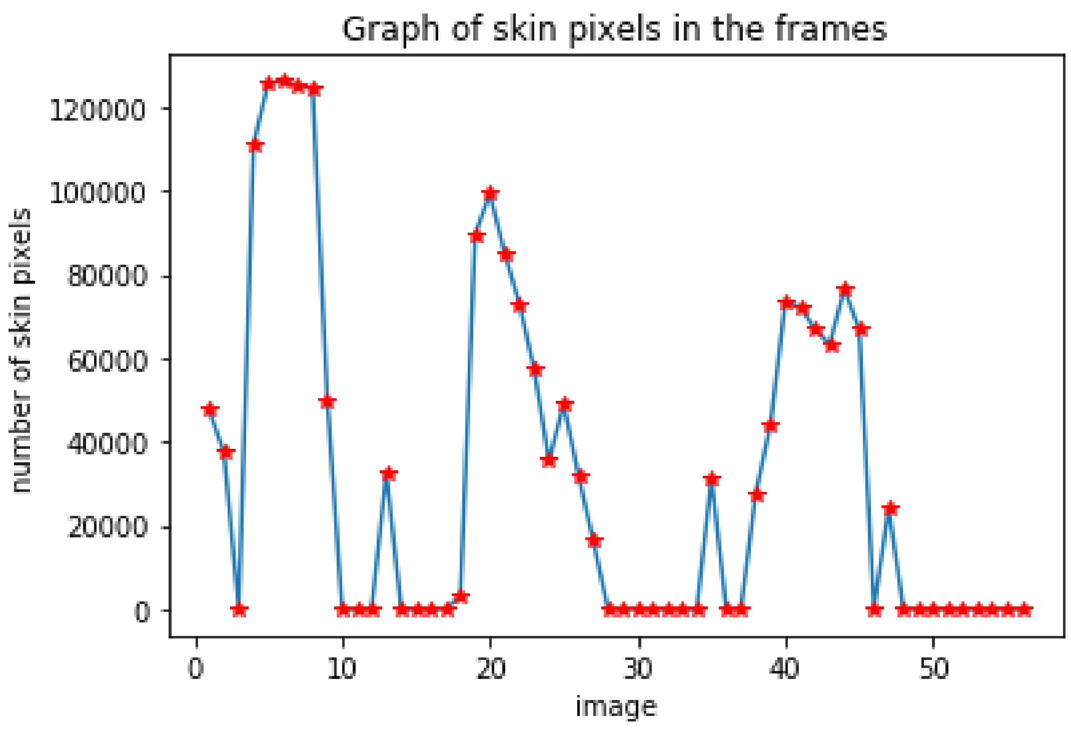
```
            x = x + 1
            xaxis.append(x)
    print("Avg skin pixels is: ",sum(yaxis)/len(yaxis))

    plt.plot(xaxis,yaxis)
    plt.plot(xaxis, yaxis,'*r')
    plt.xlabel('image')
    plt.ylabel('number of skin pixels')
    plt.title('Graph of skin pixels in the frames')
    plt.show()
```

[17]:
```
os.getcwd()
```

[17]:
```
'C:\\Users\\Anshul\\OneDrive\\Desktop\\Gaurav'
```

[18]:
```
skin_pixels_graph(work_path)
```

```
Avg skin pixels is:  33327.57142857143
```



[24]:
```
def remove_frames(path):
    count = 0
    for img in os.listdir(path):
        img_path = os.path.join(path,img)
        if get_img_avg_brightness(img_path)<135:
```

```
                        os.remove(img_path)
                        count += 1
        print(count," images removed")
```

[25]:
```
remove_frames(work_path)
```

```
3  images removed
```

[26]:
```
def image_transformations(path):
        os.mkdir("transform_images")
        i = 0

        for img in os.listdir(path):
                imgpath = os.path.join(path, img)
                image = cv2.imread(imgpath)
                image = image.astype('uint8')
                gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)  # grayscale
                gray = cv2.medianBlur(gray, 3) #smoothing
                _, thresh = cv2.threshold(gray, 150, 255, cv2.
 ↪THRESH_BINARY_INV)  # threshold
                kernel = cv2.getStructuringElement(cv2.MORPH_CROSS, (3, 3))␣
 ↪#morphological transformation
                dilated = cv2.dilate(thresh, kernel, iterations=1)  # dilate
                _,contours, hierarchy = cv2.findContours(dilated, cv2.
 ↪RETR_EXTERNAL, cv2.CHAIN_APPROX_NONE)  # get contours

                for c in contours:
                        [x, y, w, h] = cv2.boundingRect(c)
                        # discard areas that are too large
                        # discard areas that are too small
                        if h < 15 or w < 15:
                                continue
                        if h > 1500 or w > 1500:
                                continue

                        # draw rectangle around contour on original image
                        cv2.rectangle(image, (x, y), (x + w, y + h), (255, 0,␣
 ↪255), 2)

                cv2.drawContours(image, contours, -1, (255, 255, 0), 3)
                image = cv2.resize(image, (1020, 720))
                #cv2.imshow('boundary',image)
                cv2.waitKey(0)
                cv2.imwrite(path+"\\"+"transform_images"+"\\"+str(i)+"contours.
 ↪png",image)
                i = i+ 1
```

[28]:
```
work_path, os.getcwd()
```

[28]:
```
('C:\\Users\\Anshul\\OneDrive\\Desktop\\Gaurav\\frames_video_1',
 'C:\\Users\\Anshul\\OneDrive\\Desktop\\Gaurav')
```

[ ]:
```
image_transformations(work_path)
```

[ ]:
```
work_path = "C:
 ↪\\Users\\Anshul\\OneDrive\\Desktop\\Gaurav\\frames_video_1\\transform_images"

import random
images = []

for i in os.listdir(work_path):
    images.append(os.path.join(work_path,i))

def visualize_transformed_images(path, images):
    fig, ax = plt.subplots()
    img_names = random.sample(images, 1)
    print(img_names)
    img1 = cv2.imread(img_names[0])
    #img1 = cv2.resize(img1, (img_w, img_h))
    #img2 = cv2.resize(img2, (img_w, img_h))
    ax.imshow(img1)
    ax.set_aspect('auto')
    plt.show()

visualize_transformed_images(work_path, images)
```