Lab 1

Fundamentals

General Goal

The goal of this project is to introduce the functionalities and a general workorder when using Image Processing and Toolboxes in Numpy and OpenCV.

Prerequisite

The following commands in OpenCV can be used. Study how the tools work. A good source of information is the help-sections for OpenCV.
https://docs.opencv.org/4.x/d6/d00/tutorial_py_root.html

From OpenCV you can find the following commands useful to be able to finish the exercises:
- Functions:
  - shape
  - imshow
  - imread
  - namedWindow
  - createTrackbar
  - getTrackbarPos
  - cvtColor
  - threshold
  - copy
  - waitKey
  - GaussianBlur
  - Dilate
  - findContours
  - resize
  - len
  - moments
  - drawContours
  - circle
  - putText
  - minAreaRect

From Numpy you can use the following commands to be able to finish the exercises:
- Functions:
  - Vstack
  - hstack

David Stjern

Hand in

Students should hand in a zipped file containing all completed exercises including answers to all questions. The zippefile needs to be named *firstName_lastName*.zip and then uploaded on canvas. There should be a separate section (indicated by a commentbox "' "') holding the answer to each exercise. **Each exercise needs to be its own script file, Excercis1.py** etc., **it is not allowed to have several exercises in one script**. Questions that require explanation should include comments as well. See the image below for an example of a comment box where you must write the answer. **Printouts in the terminal does not count as a valid answer.**

```python
import cv2
import os

#just a function to show image
def img_show(img):
    cv2.imshow("win", img)
    cv2.waitKey(0)
    cv2.destroyAllWindows()



img=cv2.imread(r'C:\Users\dasi0006\reps\VisionLabs\Project1\2023\Stude

print('Image Witdh is        : '        ,img.shape)
print('Image Height is       : '        ,)
print('Image color Channels are      :',)
print('CV Image Type is      : '        ,)
img_show(img)

######Answer#####
#
#
#Write answer here
#
################
```

Note: Submissions that fail to follow above mentioned pattern will not be assessed and will be reported as failed. No resubmission is allowed in this case.

Grading

Each exercise are pass/fail. To fully complete this assignment, you must pass exercise 1-7.

David Stjern

Exercise 1 - Reading image Types

1. Open the image *Original.png* in Python using using the CV2.imread function, list all the image properties in a commented section in your script.
2. There is a property in *Original.png* that has been automatically added by the CV2.imread function. Find the property using the CV2.imread function. What is the property and how does it affect the information in the image?

Exercise 2 – Finding Color values

In the previous exercise, you learned how to detect how many color channels an image has. Now it is time to extract specific RGB values in images.

Find the RGB values in image *colormap.jpg* for the four positions indicated below, you will need to know what pixel to look for in the image. For this it can be useful to use paint as help:
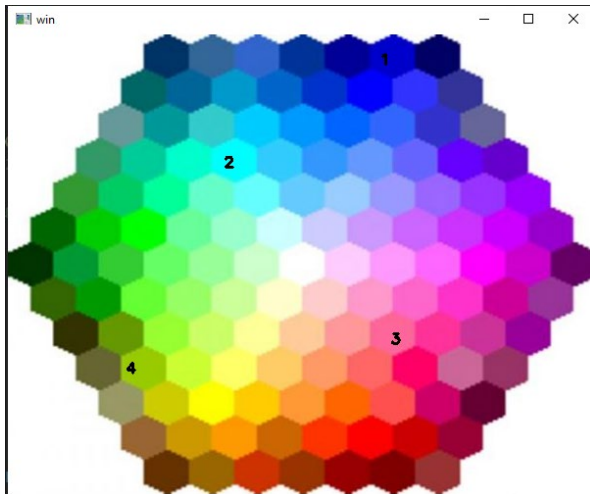
What are the RGB values for the four points? Leave all your answers in the comments box inside the script.

Exercise 3 – Binarization, Image stacking and trackbars

When working with images, displaying them side by side after making manipulations to them can be useful to follow what is going on when changing values in filters.

Create a copy of the image "piece03.png" using the five methods below
- cv.THRESH_BINARY
- cv.THRESH_BINARY_INV
- cv.THRESH_TRUNC
- cv.THRESH_TOZERO
- cv.THRESH_TOZERO_INV

Now you should have five different binarized versions of "piece03.png". You can view this as having five different and independent matrices containing image data for each method:

Now use a list to merge these images together and the NumPy stack function to display them vertically and horizontally. Think about how imshow reads the picture after this operation, what happens?

David Stjern

Lastly, create trackbars do be able to live adjust the boundary values to find the optimal lower and upper value for so that you can see the three pieces in the image. In what threshold value range does the objects show up in all images/filters? Why is it not useful to have a higher resolution than 0-255 in the trackbar? Leave all your answers in the comments box inside the script.
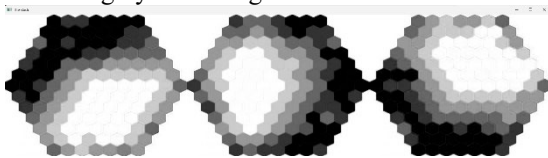

Figure 1: Example of all objects visible using all the binary filters
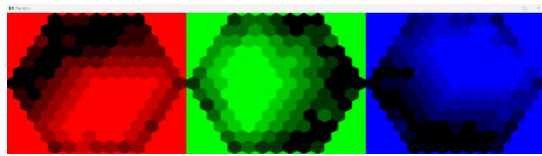
Exercise 4 – Extracting Color Channels

Read the image *colormap.jpg* and extract its color channels separately in:
- grayscale images

- color images

This means that the images that are displayed must only use one color channel each.

Display the images to verify the operation has succeeded. No need for an answer here but the code needs to work and giv the same output as the images above.

Exercise 5 – Preparing image for edge detection

To detect objects in an image, a common approach is to find the contours of the objects against the image background.
The contours are referred to as edges in image processing, and edges are registered when there is a sudden shift in pixel value between neighbouring pixels.

The objective for exercise 5 is to find get only 4 objects outputted from cv2.findContours when analysing *piece05.png*.

In https://www.geeksforgeeks.org/find-and-draw-contours-using-opencv-python/ you can see an example of this operation as well as some sample code. However, the images used in that example are not representative to a real world image which usually contains colour information in RGB or grayscale format, so *piece05.png* needs further processing before the function cv2.findContours can be used. **In essence the result needs to be 4 when running the line** *print("Number of Contours found = " + str(len(contours)))*

To get there, several pre-processing steps needs to be performed to an image to get better perform edge detection, you can see examples of this in the link as well.
Perform the following steps to the image piece05.png by prompting a LLM of your selection.
The preprocessing steps follow a standard structure that any LLM generates.
- Explain the different steps and why they are necessary inside of your code by writing comments where the filters are used.

In a certain step, it uses canny edge detection and sometimes binarization.
- What filter is the best in what circumstance, is there something you can do when acquiring the image that will increase the accuracy of the image pre-processing? Write your answer in the bottom of your script.

Tip: Use trackbars and a while-loop in order to live-adjust the different filter values in the pre-processing steps.

Exercise 6 – Finding Position

Using OpenCV, find the centre position in pixels of the object inside the colour image *star.png* by
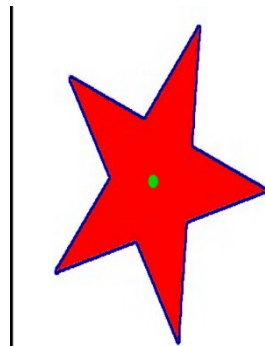- using a canny filter
- using Binarization filter

Figure 2 find the center coordinates of the green dot

Leave all your answers in the comments box inside the script clearly diversifying which coordinates you get when using the different filters.

David Stjern

Exercise 7 – Finding Orientation

Using OpenCV, Find the orientation in degrees of the object inside the color image *tree.png* by
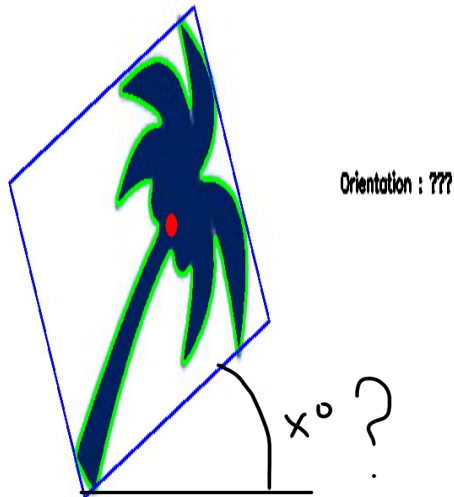- using a canny filter
- using binarization filter

Figure 3 find the orientation of the shape

Leave all your answers in the comments box inside the script clearly diversifying what angle x has in degrees you get when using the different filters.

Good luck!

David Stjern