**Project 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 PIL and OpenCV.

#### Prerequisite

The following commands in PIL and OpenCV can be used. Study how the tools work. A good source of information is the help-sections for PIL and OpenCV.

<https://pillow.readthedocs.io/en/stable/>

<https://docs.opencv.org/4.x/d6/d00/tutorial_py_root.html>

There can of course be other similar or same functionalities in the PIL and OpenCV, but then the students need to have a good knowledge of how the data is structured to avoid mistakes and errors (which the student might run into using only one of the libraries anyway).

From PIL you can use the following Modules, Functions and methods to be able to finish the exercises:

* Modules:
  + Image
* Functions:
  + .open
  + .getpixel()

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

#### 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. Questions that require explanation should include comments as well.

**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.

For extra credit exercise 8 and/or 9 needs to be completed.

#### Exercise 1 - Reading Types

Open the images *icon01.png* and *icon02.png* in Python using **PIL**. What is the type of each image? Why?

List RGB values of colors presented in *rgb01.png* and *rgb02.png* images.

#### Exercise 2 – Reading Images

Read the images *flower.png* and *eagle.png* and save them into variables *img\_1* and *img\_2*. Address the following properties for each image by looking into their variables (TIP: you might need functionalities both from PIL and OpenCV):

* resolution
* dimension
* class (data type)
* Number of channels
* image type (binary / gray scale / color). Elaborate on your answer.

Read the image *car.png* and find RGB values for the pixel located at .

#### 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. 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?

Lastly, create trackbars do be able to live adjust the boundary values to find the optimal lower and upper value for all pictures. In what threshold value range does the objects start do fade to black? Why?

#### Exercise 4 – Extracting Color Channels

Read the image *colorStripe.png* and extract its color channels as

* grayscale images
* color images

Then display the images to verify the operation has succeeded.

#### 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, to find these edges a number of pre-processing steps can be done to an image to get better edge detection. Do the following steps to the image piece05.png (Add trackbars for extra control and to find the best values for the tools)

1. Convert it to grayscale
2. Blur the now grayscale image using Gaussian Blur to reduce noise in the picture
3. Use canny to find the edges
4. Dilate the edges to fill in small gaps that might appear when using canny
5. Now use the findCountors on the diluted picture, how many objects can you find?

#### Exercise 6 – Finding Position

Find the centre position of the object inside the colour image *star.png* by

* using Edge detection
* using Binarization

(!Note! using scaledown img=cv2.resize(img, (960, 540) if other scale is used other cords will be the result)

#### Exercise 7 – Finding Orientation

Find the orientation of the object inside the color image *tree.png* by

* using Edge detection
* using Binarization

#### Exercise 8 (not mandatory)

Find the position and orientation of the object inside the color image *piece06.png*.

|  |  |
| --- | --- |
|  |  |
| *piece06.png* | position = (pixels)  orientation = (degree) |

#### Exercise 9 (not mandatory)

Find the position and orientation of three objects inside the image *piece03.png*.

|  |  |
| --- | --- |
|  |  |
| *piece03.png* | position\_black =  orientation\_black =  position\_blue =  orientation\_blue =  position\_red =  orientation\_red = |