

MODULE 2 – Using Numpy and OpenCV and Introduction to Image Processing Applications

A. PURPOSE

1. Students are able to open image files from private Google
2. Students are able to understand the basics of the OpenCV library in Python
3. Students are able to understand color channels in OpenCV and their conversions

B. TOOLS AND MATERIALS

C. THEORETICAL BACKGROUND

NumPy (Numerical Python) is a Python library for data scientists to construct N-dimensional array objects. NumPy is part of the Python ecosystem of the open-source SciPy tool. Therefore, this ability has similarities with Python. However, memory consumption and NumPy arrays are more negligible, and runtime is faster than lists in Python. NumPy also makes it easy to execute operations related to 1-d and Matrix (2-d array).

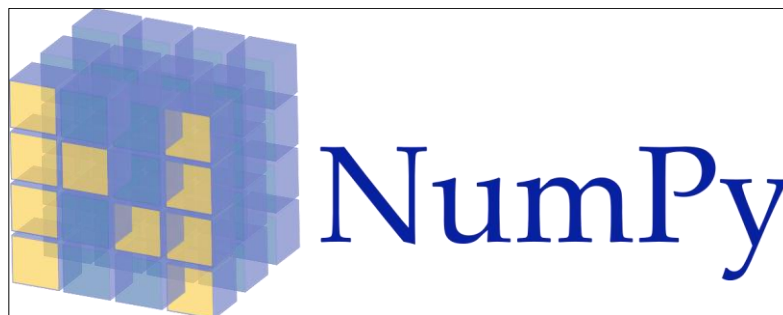


Figure 1. NumPy Logo

Before using NumPy, one must first import the Numpy package using **pip install NumPy**. However, if Numpy is already installed or used in the Google Colab facility, NumPy can be imported directly using `import NumPy as np`.

Array creation with NumPy is implemented by adding a data type with the **dtype** parameter. For example, the following are some of the types found in NumPy:

Table 1. NumPy data type (Source : <https://vsvaibhav2016.medium.com/basics-of-numpy-python-for-data-analysis-45b0c43f591b>)

bool_	Boolean (True or False) stored as a byte
int_	Default integer type
intc	Identical to C int e.g int32 in64
intp	Integer used for indexing
int8	Byte (-128 to 127)
int16	Integer (-32768 to 32767)
int32	Integer (-2147483648 to 2147483647)
int64	Integer (-9223372036854775808 to 9223372036854775807)
uint8	Unsigned integer (0 to 255)
uint16	Unsigned integer (0 to 65535)
uint32	Unsigned integer (0 to 4294967295)
uint64	Unsigned integer (0 to 18446744073709551615)
float16	Half precision float: sign bit, 5 bits exponent, 10 bits mantissa
float32	Single precision float: sign bit, 8 bits exponent, 23 bits mantissa
float64	Double precision float: sign bit, 11 bits exponent, 52 bits mantissa
complex64	Complex number, represented by two 32-bit floats (real and imaginary components)
complex128	Complex number, represented by two 64-bit floats (real and imaginary components)

NumPy arrays are referred to as **ndarrays**, also known as arrays. To solve problems with arrays, especially multidimensional arrays with NumPy, the NumPy dimensions are called the axis, and the number of axes is called the rank. For example, for 3D space, the coordinates of a point in 1 axis 3D space with length three can be expressed as [2,1,1]. At the same time, the length of the axis is 3. Apart from this example, an array like `[[1 , 1. , 1.], [2. , 2. , 2.]]` is an example of a variety with rank 2, which has two axes where the first axis is length 2, and the second axis has a size of 3. The following are some of the essential attributes of ndarrays:

Table 2. ndarray attribute usage

Attribute	Usage
<code>ndarray.ndim</code>	Number of array dimensions. In Python, it is called rank.
<code>ndarray.shape</code>	The size of the array in each dimension. A matrix with n rows and m columns or a matrix (n,m).
<code>ndarray.size</code>	The number of elements of an array or is the multiplication of elements of shape
<code>ndarray.dtype</code>	Is the standard Python data type for an element in an array
<code>ndarray.itemsize</code>	The size in bytes of each array element. Example: An array that has elements of type float64 has itemsize 8 (= 64/8)
<code>ndarray.data</code>	Buffer contains array elements

There are several main methods of NumPy arrays used to manipulate data as follows:

1. Indexing

Method for accessing an array element. For example: `a[5]`. `a` is a NumPy array, while 5 is the 5th element in the array.

2. Slicing

Method for accessing arrays especially subarrays (multiple elements at once). Subarray access is done with a special character (`:`). For example: `a[2:6]`, meaning that access to array elements is done from index 2 to 5 (6-1). `a[:5]`, means access the first 5 elements of an array. `a[::2]`, meaning that array access is performed starting at index 0, up to the end index, with a distance of 2 elements for each access.

3. Iterating

Interactive array access method. For example :

```
for rows in b:  
    print(rows)
```

This means that each array is accessed and displayed per row

```
for elements in b. flats :  
    print ( elements )
```

This means that each array is accessed and displayed per element in each row

4. Reshaping

Method for changing the shape of an array. For example :

```
a=np.array([ 3. , 6. , 1. , 5. , 3. , 2. , 0. , 0. , 2. , 5. , 1. ,  
4.])
```

```
a. reshape (6 ,2)
```

This means that array `a` will be converted into 6 rows 2 columns to

```
array([[3., 6.],  
       [1., 5.],  
       [3, 2.],  
       [0, 0.],  
       [2., 5.],  
       [1., 4.]])
```

OpenCV (Open Source Computer Vision Library) was developed by Intel Corporation and is an open source library provided for programming related to digital images and many AI (Artificial Intelligence) methods. OpenCV has many features related to digital imagery such as face recognition, face tracking, face detection, Kalman filtering, simple algorithm on Computer Vision for low level API, etc. OpenCV was originally a library for the C/C++ programming language and is now being developed into other languages such as Python, Java, and Matlab.

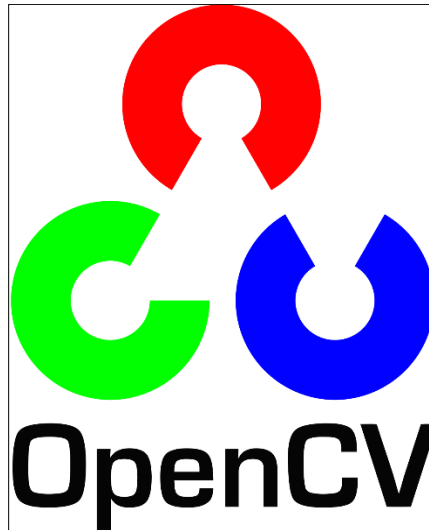


Figure 2. OpenCV Logo

The first version of OpenCV was introduced in 1999 by Gary Bradsky and started to be released in 2000. This first version of OpenCV requires the Intel Image Processing library in its use. However, the dependency on the library has been removed. It now stands as a standalone and multiplatform library so that it can be used on various operating systems, including Windows, Linux, macOSX, and Android.

OpenCV has 5 (five) basic libraries as follows:

1. **CV**
Supports image processing algorithms as well as computer vision
2. **ML**
Support machine learning algorithm
3. **Highgui**
Supports GUI, image, video I/O
4. **CXCore**
Supports data structures, XML,
5. **CvAux**
Special computer vision module

Based on the main library owned by OpenCV, there are several main features as follows:

1. Image and video I/O
2. General Computer Vision and digital image processing (for low and mid level APIs)
3. High level computer vision module
4. Methods for AI and machine learning
5. Image sampling and transformation
6. Methods for creating and analyzing binary images
7. Methods to account for 3D modeling
8. Motion detector
9. Structural Analysis
10. Camera Calibration
11. Image Labeling

Applications related to image processing are applied in various problem solving solutions in various fields such as the following:

1. Military

Some examples of applications in the military field such as night vision, identification of enemy aircraft with radar, and missile targets with virtual sensors.

2. Health

Some examples of applications in the health sector such as fracture detection, fetal photo reconstruction, cancer detection, and so on.

3. Biology

Examples of applications in the field of biology such as the recognition of chromosomes with microscopic images.

4. Education

Examples of applications in the field of education such as processing student registration with a scanner.

5. Remote Sensing

Some examples of applications in the health sector such as mapping regional boundaries, recognizing the types of rock layers below the earth's surface

6. Police/Law

Some examples of applications in the health sector such as fingerprint pattern recognition, facial reconstruction of criminals,

7. Trading

Some examples of applications in the field of commerce such as barcode readers in supermarkets, automatic letter or number recognition

There are several digital image processors that can produce various useful applications as previously described as follows:

1. Corel Painter

Corel Painter is a digital image processing tool that is quite unique with its main feature that it has 131 types of brushes with advanced technology such as Dynamic Speckles (a feature that combines brush thickness control with the laws of particle physics so that the brush becomes soft and dynamic), besides that there are also brushes that are connected to the brush. a sound that reacts to every brush stroke. The following is an example of what Corel Painter 2016 looks like:

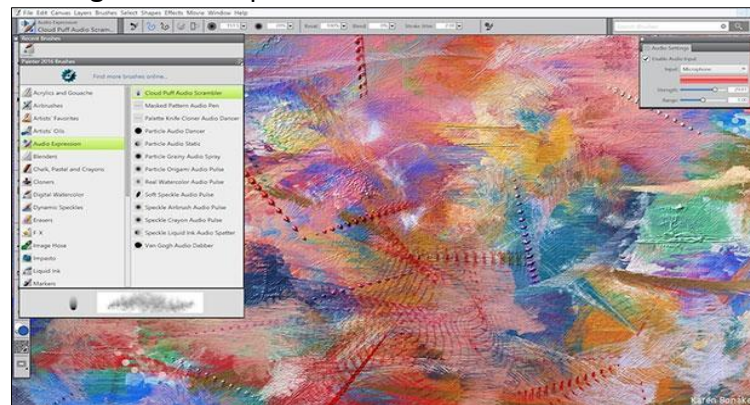


Figure 3. Corel Painter (source: <https://dailysocial.id/post/corel-painter-2016-hadirkan-fitur-melukis-dengan-suara>)

2. Adobe Photoshop

Adobe Photoshop is a software tool for processing digital images. Adobe Photoshop was developed in 1987 and licensed by Adobe Systems Incorporated in 1988. Industry standard raster applications also use this application as an editing medium. The following is an example of how the Adobe Photoshop main page looks:

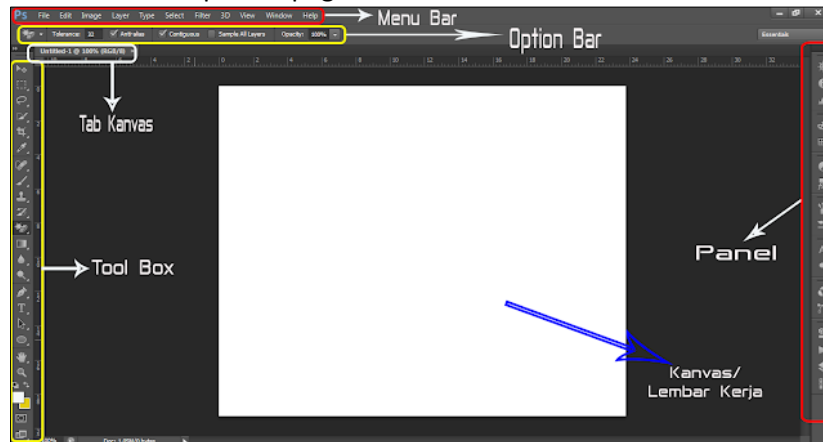


Figure 4. Adobe Photoshop (Source :

<https://www.infopubliknews.com/2020/06/bagian-bagian-adobe-photoshop.html>)

3. Microsoft Paint

Microsoft paint is a software owned by Microsoft that provides complete digital image processing facilities such as making straight lines, creating curved lines, editing text, giving colors, etc. Microsoft Paint display can be seen as follows:



Figure 5. MS-Paint (Sumber : <https://pt.slideshare.net/Et3nK/mengenal-microsoft-paint>)

4. GIMP

GNU Image Manipulation Program (GIMP) is an image processing software. GIMP runs on the GNOME desktop. GIMP has similarities with functions similar to Photoshop. GIMP is a free software that has capabilities such as image decomposition, image conversion, creating large-scale images, etc. Here is a view on GIMP.

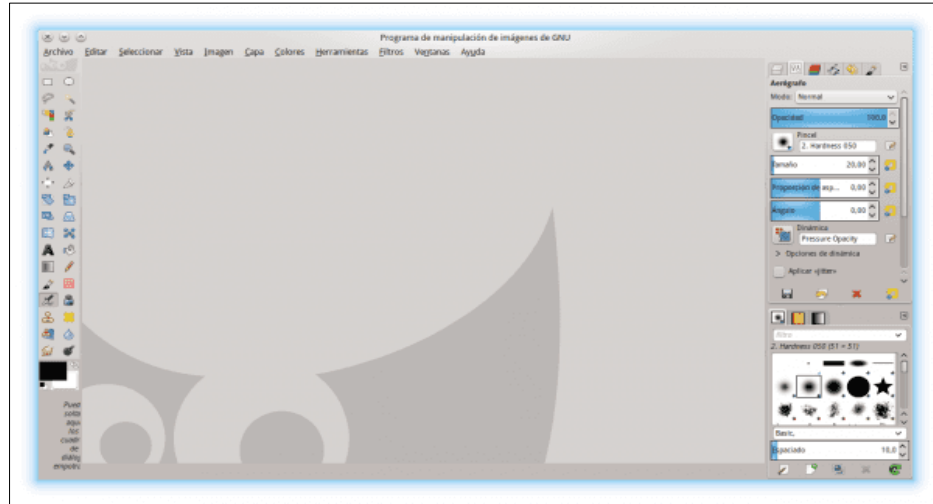


Figure 6. GIMP (Source : <https://blog.desdelinux.net/id/beri-gimp-tampilan-photoshop-cs6/>)

D. PRACTICUM

1. Go to <https://colab.research.google.com/>. After confirming that Google Colab is connected to your Github, continue by selecting the repository that was used in last week's practicum, rename the file to "Week2.ipynb".

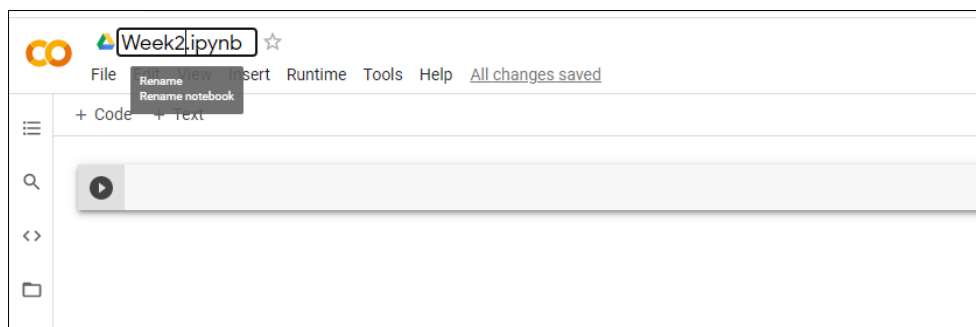
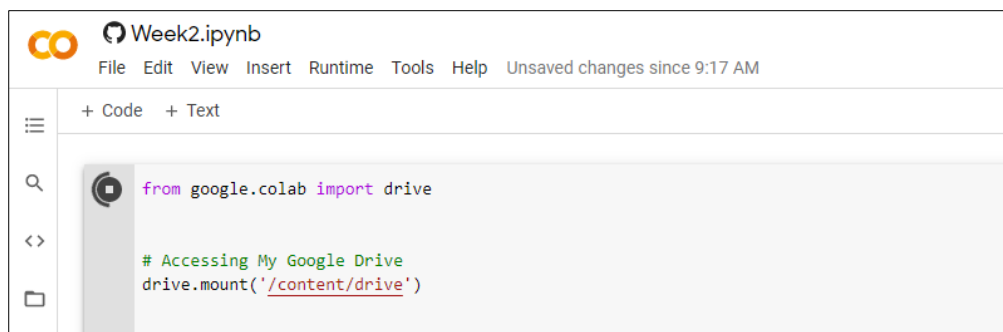


Figure 7. Renaming File in Google Collab

Import the folder in your Drive in the following way.



```
from google.colab import drive

# Accessing My Google Drive
drive.mount('/content/drive')
```

Figure 8. Import File from Google Drive

The Google Colab synchronization process to connect with GDrive requires a slight change in notification settings as shown below. Then a URL will appear that will lead to a new tab to login to a google account. At this stage, an authorization code is needed that appears after the login process is carried out.

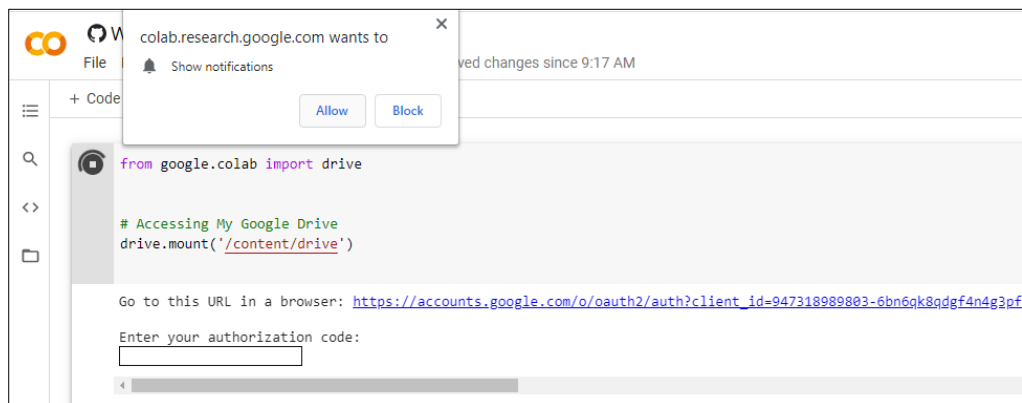


Figure 9. Google Drive authorization function

After the login process is done, then copy the authorization code and paste it in the field provided on Google Colab.

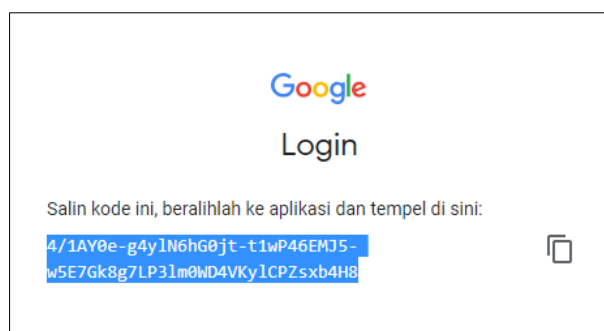
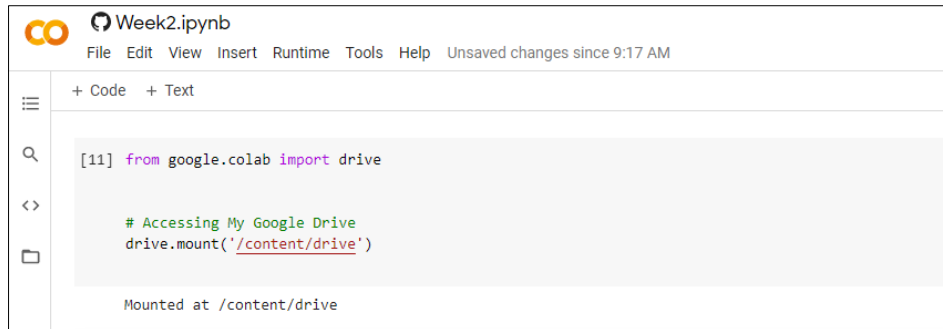


Figure 10. Code from Google Drive

If successful, then Google Colab can already access your gdrive folder with the program output description "Mounted at /content/drive".



```

Week2.ipynb
File Edit View Insert Runtime Tools Help Unsaved changes since 9:17 AM

+ Code + Text

[11]: from google.colab import drive

# Accessing My Google Drive
drive.mount('/content/drive')

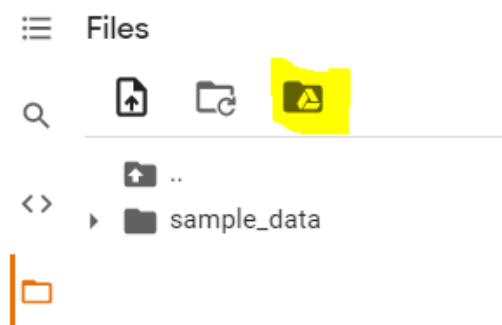
Mounted at /content/drive
    
```

Figure 11. Google Drive Succeed to mount

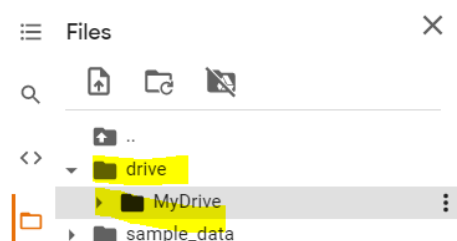
The second way is by clicking the folder button on the left



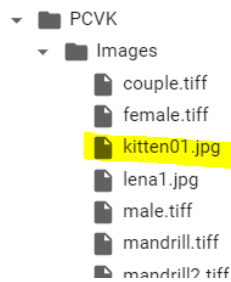
Select gdrive button -> Connect to Google Drive



The drive will appear in the folder list



Select the file and click the 3 dot icon to the right of the file. click copy path:



Then the program code can be continued by opening an existing Google Drive file. For example, in the program code below the image file with the name kitten01.jpg will be opened for further processing.

```
import cv2 as cv
from google.colab.patches import cv2_imshow
from skimage import io
import matplotlib.pyplot as plt
import numpy as np
img = cv.imread('/content/drive/MyDrive/PCVK_MM/Images/kitten01.jpg')
plt.imshow(img) #perhatikan hasilnya adalah citra dgn channel
```

The result is an image output that has been plotted with matplotlib to determine the length and width of the image.

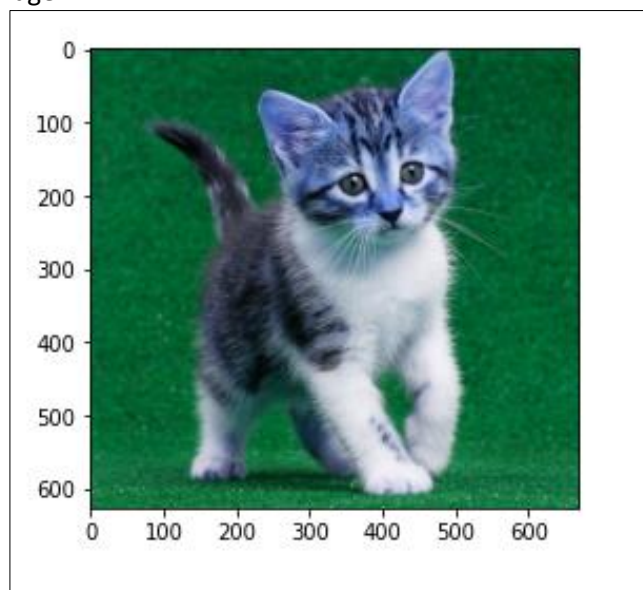


Figure 12. Image with BGR channel preview

2. OpenCV reads image and saves in color channel BGR (Blue Green Red) Display pixel value and image

```
img2 = img  
img3 = cv.cvtColor(img, cv.COLOR_BGR2RGB)    #konversi channel BGR -> RGB  
plt.imshow(img3)
```

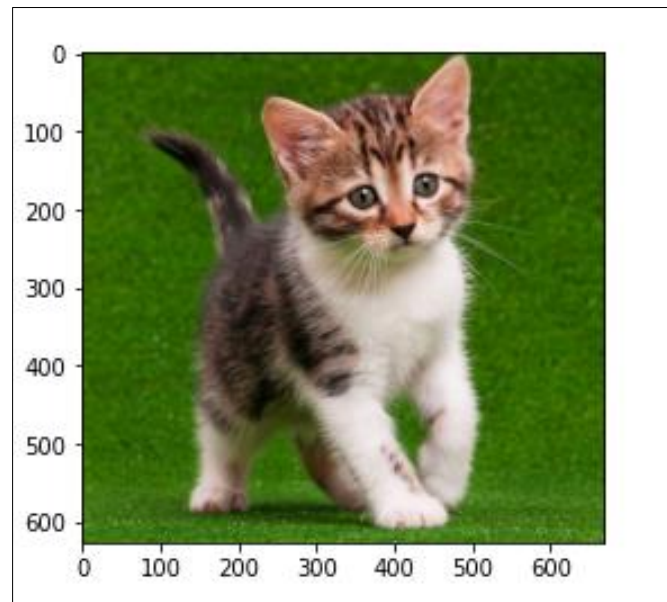


Figure 13. Image with RGB channel preview

3. Displaying Grayscale image, resizing, flipping, saving the resulted image

```
img_gray = cv.imread('/content/drive/MyDrive/PCVK_MM/Images/kitten01.jpg', cv.IMREAD_GRAYSCALE)  
plt.imshow(img_gray)
```

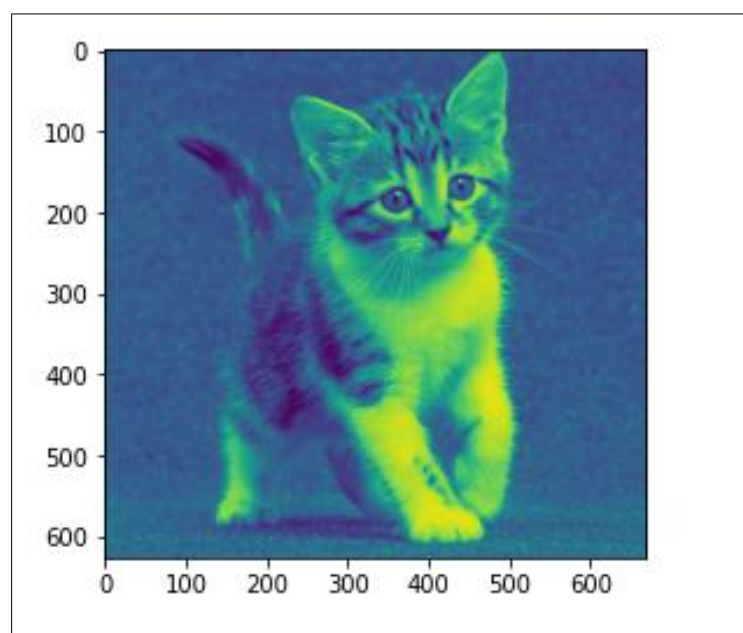


Figure 14. Image with grayscale channel

From the grayscale image, it is then selected to display a colormap with the color 'gray'

```
plt.imshow(img_gray, cmap='gray')
```

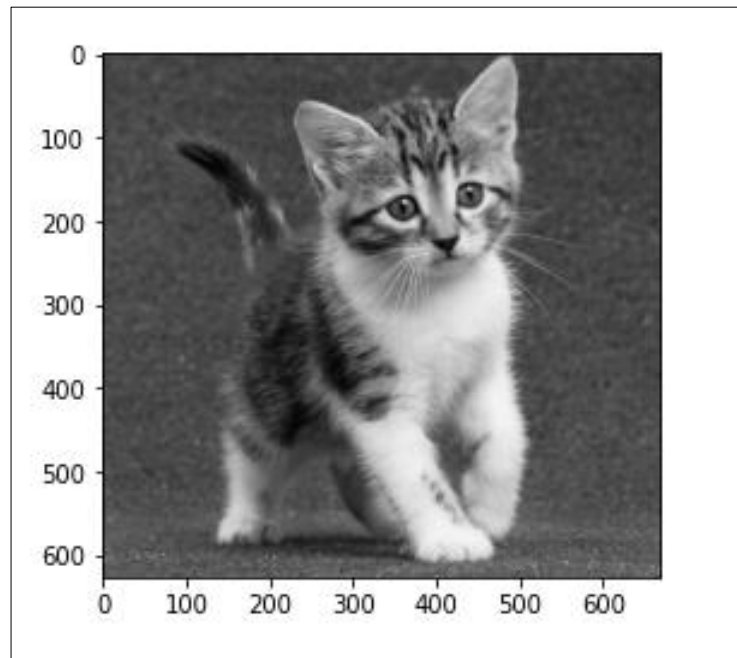


Figure 15. Image with grayscale mapping

From the grayscale image, it is then selected to display a colormap with the color 'magma'

```
plt.imshow(img_gray, cmap='magma')
```

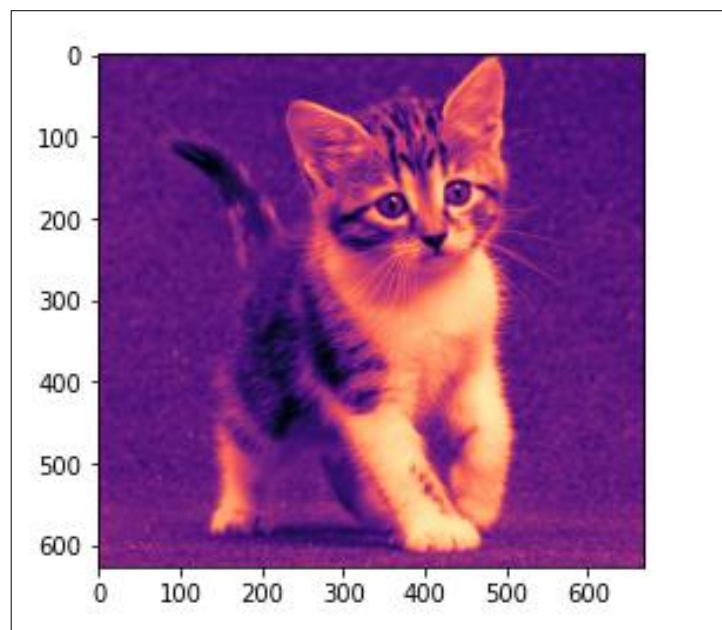


Figure 16. Image with magma mapping

RGB image resized to 512 x 1024

```
img4 = cv.resize(cv.cvtColor(img,cv.COLOR_BGR2RGB), (512,1024))
plt.imshow(img4)
```

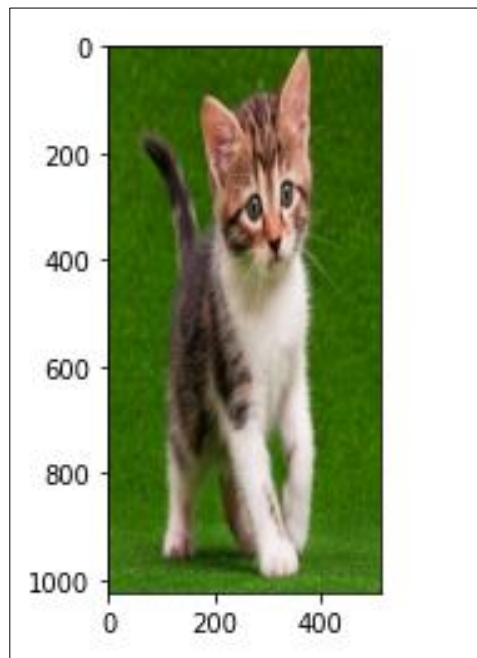


Figure 17. Image resized in 512 x 1024 pixel resolution

RGB image is displayed in a larger size with the image position upside down

```
img5 = cv.flip(cv.cvtColor(img,cv.COLOR_BGR2RGB),0)

#simpan File image
#cv.imwrite('/content/drive/MyDrive/PCVK_MM/Images/mandrill2.tiff',img5)

#tampilkan plot dengan ukuran canvas yg lebih besar
fig = plt.figure(figsize=(10,10))      #ubah-ubah ukuran (10,10) sesuai kebutuhan
ax = fig.add_subplot(111)
ax.imshow(img5)
```

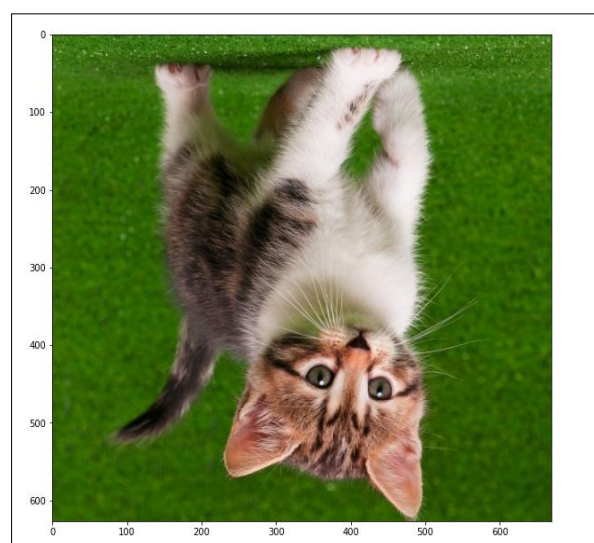


Figure 18. Image is mirrored upside down

4. Creating 2D Geometry image using OpenCV. Start with creating Black image

```
black_img = np.zeros(shape=(512,512,3),dtype=np.int16)
plt.imshow(black_img)
```

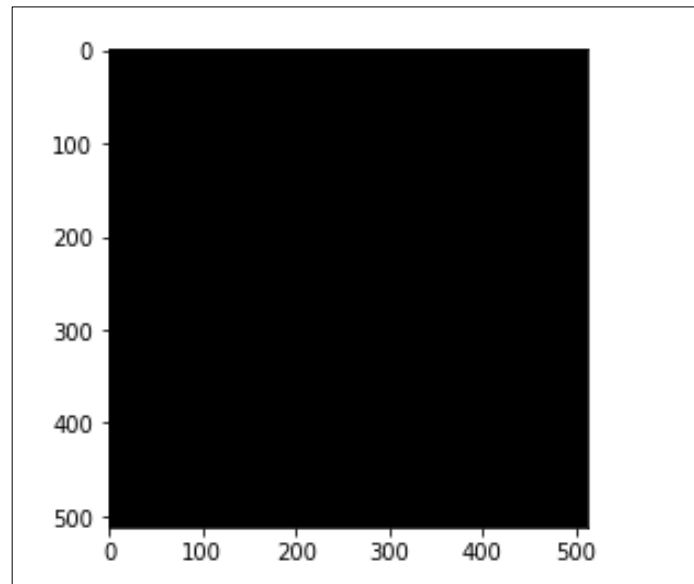


Figure 19. Black image

Add rectangle defining coordinat pt1 and pt2

```
#perhatikan koordinat titik2 pt1 dan pt2
cv.rectangle(black_img,pt1=(384,0),pt2=(510,150),color=(0,255,0),thickness=10)
plt.imshow(black_img)
```

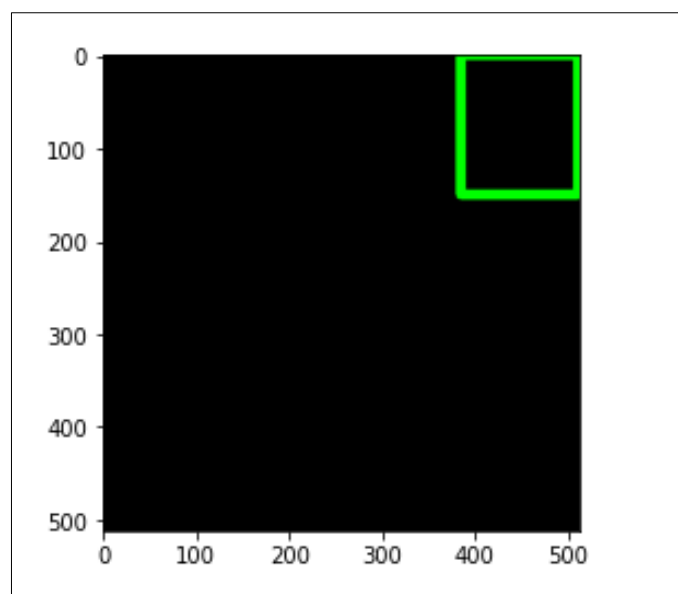


Figure 20. Image with green rectangle

Next, add a square shape according to the pt1 and pt2 coordinates written in the program code.

```
cv.rectangle(black_img,pt1=(200,200),pt2=(300,300),color=(0,0,255),thickness=15)
plt.imshow(black_img)
```

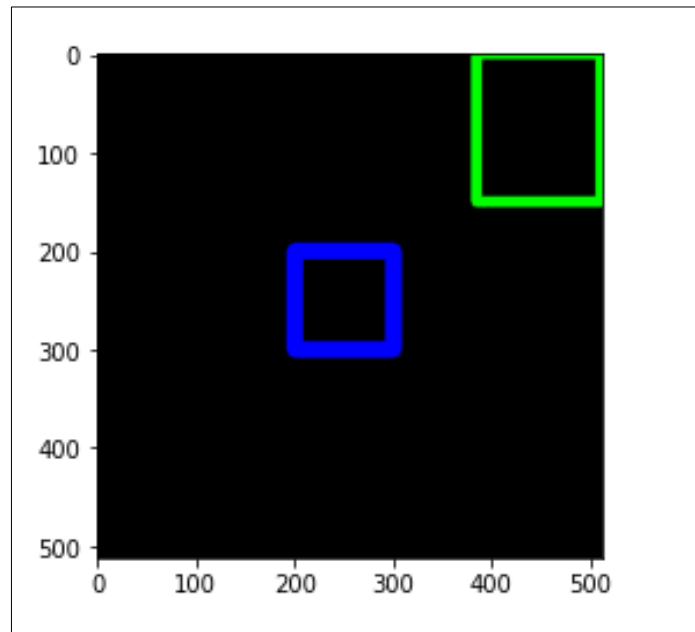


Figure 21. Adding blue square in image

The next step is to add a circle shape according to the radius written in the program code.

```
cv.circle(black_img,center=(100,100),radius=50,color=(255,0,0),thickness=8)
plt.imshow(black_img)
```

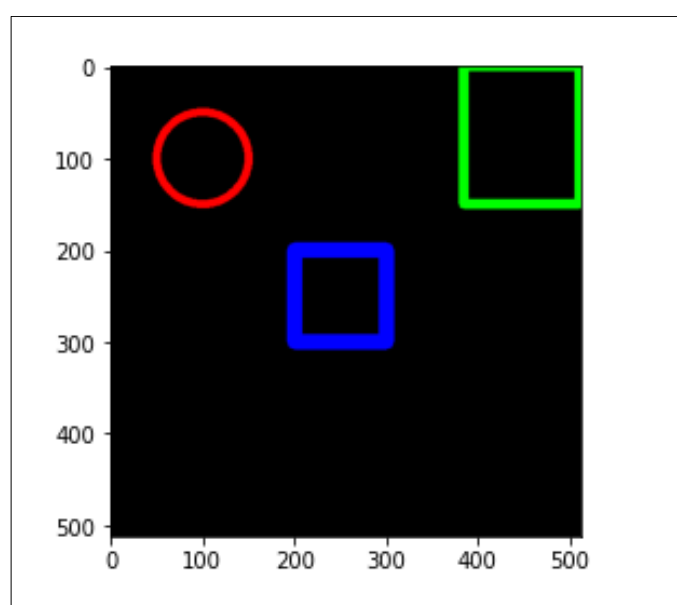


Figure 22. Adding circle to the image

Then do the addition of lines according to the coordinates of pt1 and pt2 as follows.

```
cv.line(black_img,pt1=(0,0),pt2=(512,512),color=(255,255,255),thickness=5)
plt.imshow(black_img)
```

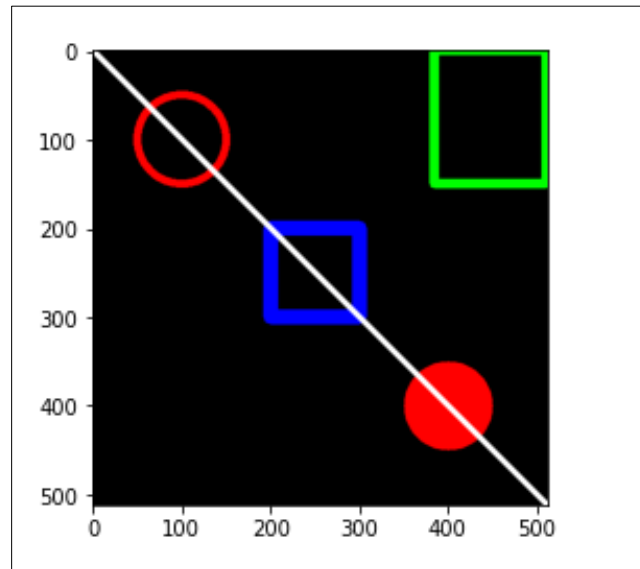


Figure 23. Adding diagonal line in image

The addition of text with a font that has been written with a predetermined size.

```
font = cv.FONT_HERSHEY_SIMPLEX
cv.putText(black_img,text='Hello',org=(10,500),fontFace=font,fontScale=4,color=(255,255,0),thickness=2,lineType=cv.LINE_AA)
plt.imshow(black_img)
```

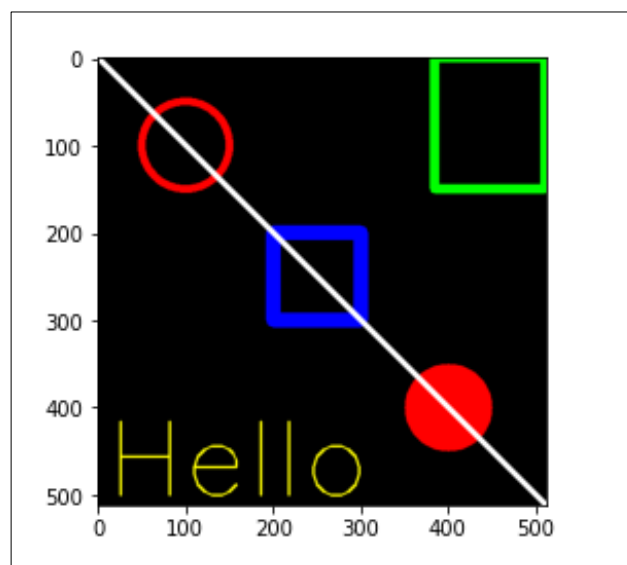


Figure 24. Adding text in image

```
black_img2=np.zeros(shape=(512,512,3),dtype=np.int32)
plt.imshow(black_img2)
```

Creating black image again with int32 data type

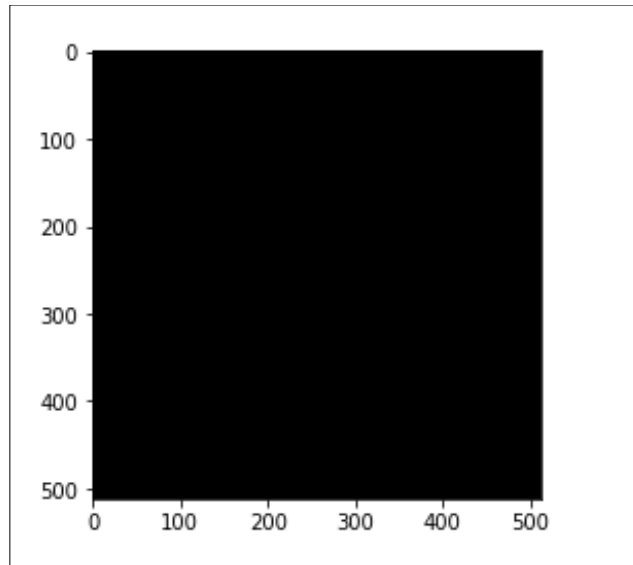


Figure 25. Create new black image

The following is the program code to initialize a NumPy array with the data type int32

```
vertices = np.array([[100,300],[200,200],[400,300],[200,400]],dtype=np.int32)
vertices
```

```
array([[100, 300],
       [200, 200],
       [400, 300],
       [200, 400]], dtype=int32)
```

The array is then reshaped as follows:

```
pts = vertices.reshape((-1,1,2)) # nilai 2 untuk menunjukkan bahwa tiap titik dibuat 3 channel yg mewakili
pts
```

```
array([[[100, 300]],
       [[200, 200]],
       [[400, 300]],
       [[200, 400]]], dtype=int32)
```

Add a polyline to the second black image that has been created.

```
cv.polylines(black_img2,[pts],isClosed=True,color=(255,0,0),thickness=5)
plt.imshow(black_img2)
```

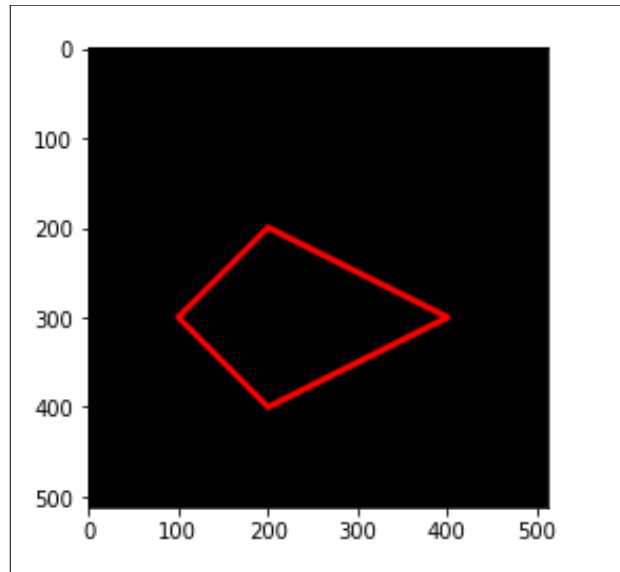


Figure 26. Adding polyline in image

- After all the code is finished save “Week2.ipynb” on your GitHub by selecting File then “Save a copy in GitHub”.

Question

- What is the difference between the images displayed without and with matplotlib?
- What is the difference and effect of creating a black image between int16 and int32 data types?
- What is the use of "google.colab.patches import cv2_imshow" in the following code snippet

```
from google.colab.patches import cv2_imshow
from skimage import io
```
- What is the use of "skimage import io" in the code snippet for question number 3

E. TASK

Based on practicum parts 1 and 2, do the following tasks:

- By using figsize, notice whether the image pixel size also changes?
- Show images in Red-Green and Green-Blue channels only!
- Show image row 10-100, column 10-100!
- Show image rows 5-30, all columns, channel Green only!
- Create 5 boxes of different sizes and colors in one image. it is recommended to use random numbers!
- Show the image upside down!

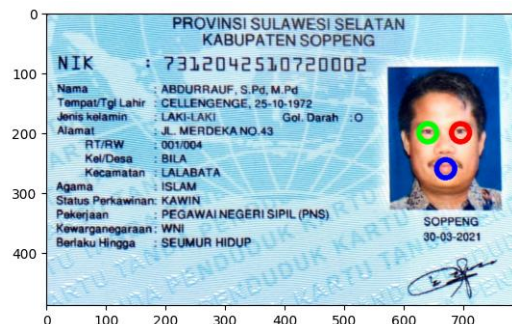
Based on practicum sections 3 and 4, do the following tasks:

- Make rectangles and circles on the face of your photo image when you are active (not a passport photo).

8. Create a rectangle in the lower left corner of channel B in the RGB color space from the kitten / lena / mandrill / male / female / couple / sailboat / peppers image !
9. Complete the writing of the file name on the image file from question no.8. use the font, font size, and font color according to your wishes.
10. Show your program code to the lecturers

TASK

- Close each specific part of the ID card using the functions you have learned. Get creative with the colors and sizes of the shapes.
- - Group 1 closes the NIK number and name section.
- - Group 2 close the TTL and Gender sections.
- - Group 3 closes the Goals section. Blood and TTD.
- - Group 4 closes the Address and District sections.
- - Group 5 closes the Religion and Marital Status section. Group 6 close Employment and citizenship.
- - Group 7 closes the section valid up to and photos.
- - Group 8 closes the district section and the date the KTP was issued is at the bottom of the photo.
- - Group 9 closes the name of the province and district at the very top of the KTP.
- - Group 10 closes RT/RW and Kel/village sections. Cover using 2 different color boxes.



--- GOOD LUCK ---