Import the library

In [1]: **from** keras.preprocessing.image **import** ImageDataGenerator

Arguments for ImageDataGenerator class

In [2]: train\_datagen**=**ImageDataGenerator(rescale**=**1.**/**255,shear\_range**=**0.2,zoom\_range**=**0.2,ho test\_datagen**=**ImageDataGenerator(rescale**=**1.**/**255)

Applying ImageDataGenerator functionality to trainset and testset

In [3]: x\_train**=**train\_datagen**.**flow\_from\_directory('/content/drive/MyDrive/dataset/data\_se

Found 5750 images belonging to 9 classes.

In [4]: x\_test**=**test\_datagen**.**flow\_from\_directory('/content/drive/MyDrive/dataset/data\_set/

Found 2250 images belonging to 9 classes.

# !pip install opencv.python

In [5]: **import** cv2

# imread is used to read the image

In [6]: img**=**cv2**.**imread('/content/drive/MyDrive/dataset/data\_set/test\_set/A/10.png')

In [7]: img

Out[7]: array([[[0, 0, 0], [0, 0, 0],

[0, 0, 0], ...,

[0, 0, 0],

[0, 0, 0],

[0, 0, 0]],

[[0, 0, 0],

[0, 0, 0],

[0, 0, 0], ...,

[0, 0, 0],

[0, 0, 0],

[0, 0, 0]],

[[0, 0, 0],

[0, 0, 0],

[0, 0, 0],

...,

[0, 0, 0],

[0, 0, 0],

[0, 0, 0]],

...,

[[0, 0, 0],

[0, 0, 0],

[0, 0, 0], ...,

[0, 0, 0],

[0, 0, 0],

[0, 0, 0]],

[[0, 0, 0],

[0, 0, 0],

[0, 0, 0], ...,

[0, 0, 0],

[0, 0, 0],

[0, 0, 0]],

[[0, 0, 0],

[0, 0, 0],

[0, 0, 0], ...,

[0, 0, 0],

[0, 0, 0],

[0, 0, 0]]], dtype=uint8)

In [8]:

img**.**ndim

Out[8]: 3

In [9]: type(img)

Out[9]: numpy.ndarray

In [10]: img**.**shape

Out[10]: (64, 64, 3)

# flag 1 means color image

In [11]: img\_flag **=** cv2**.**imread('/content/drive/MyDrive/dataset/data\_set/test\_set/A/100.png

In [12]: img\_flag

Out[12]: array([[[0, 0, 0], [0, 0, 0],

[0, 0, 0], ...,

[0, 0, 0],

[0, 0, 0],

[0, 0, 0]],

[[0, 0, 0],

[0, 0, 0],

[0, 0, 0], ...,

[0, 0, 0],

[0, 0, 0],

[0, 0, 0]],

[[0, 0, 0],

[0, 0, 0],

[0, 0, 0], ...,

[0, 0, 0],

[0, 0, 0],

[0, 0, 0]],

...,

[[0, 0, 0],

[0, 0, 0],

[0, 0, 0], ...,

[0, 0, 0],

[0, 0, 0],

[0, 0, 0]],

[[0, 0, 0],

[0, 0, 0],

[0, 0, 0], ...,

[0, 0, 0],

[0, 0, 0],

[0, 0, 0]],

[[0, 0, 0],

[0, 0, 0],

[0, 0, 0], ...,

[0, 0, 0],

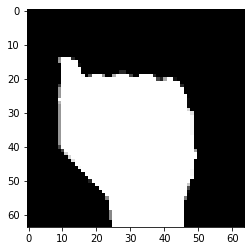
[0, 0, 0],

[0, 0, 0]]], dtype=uint8)

In [13]: **import** matplotlib.pyplot **as** plt

In [14]: plt**.**imshow(img)

Out[14]:



In [15]: plt**.**imshow(img\_flag)

Out[15]:



# resize the image

In [16]: resized\_img **=** cv2**.**resize(img,(100,100))

In [17]:

resized\_img**.**shape

Out[17]: (100, 100, 3)

In [18]: plt**.**imshow(resized\_img)

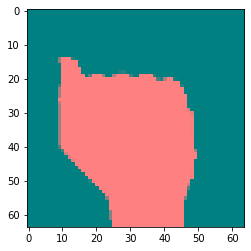
Out[18]:



In [19]: cv\_img **=** cv2**.**cvtColor(img,cv2**.**COLOR\_BGR2YCR\_CB)

In [20]: plt**.**imshow(cv\_img)

Out[20]:



In [21]: cv\_img **=** cv2**.**cvtColor(img,cv2**.**COLOR\_BGR2GRAY)

In [22]: plt**.**imshow(cv\_img)

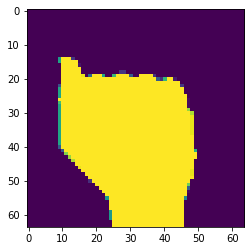
Out[22]:

In [23]:

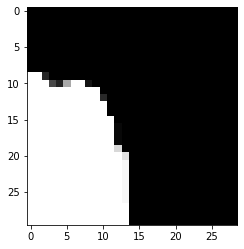
In [24]:

Out[24]:

In [25]:



|  |  |
| --- | --- |
| roi\_img **=** roi\_img **=** | img[50:280,35:150] img[10:40,35:150] |
|  |  |
| plt**.**imshow(roi\_img) | |



roi\_img **=** img[10:40,0:90]

In [26]:

plt**.**imshow(roi\_img)

Out[26]:

In [27]:



|  |
| --- |
| plt**.**imshow(roi\_img) |

roi\_img **=** img[0:90,10:40] In [28]:

Out[28]:



# averaging cv2.blur() or cv2.boxFilter

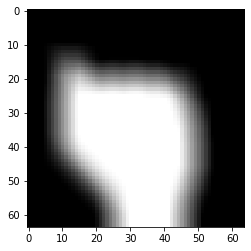
|  |
| --- |
| img\_bl **=** cv2**.**blur(img,(10,10)) |

In [29]:

|  |
| --- |
| plt**.**imshow(img\_bl) |

In [30]:

Out[30]:



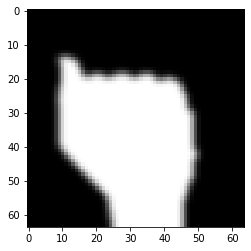
|  |
| --- |
| img\_gbl **=** cv2**.**GaussianBlur(img,(5,5),0) |

In [31]:

|  |
| --- |
| plt**.**imshow(img\_gbl) |

In [32]:

Out[32]:



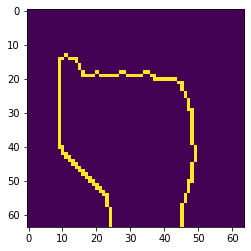
|  |
| --- |
| img\_edge **=** cv2**.**Canny(img,230,350) |

In [33]:

|  |
| --- |
| plt**.**imshow(img\_edge) |

In [34]:

Out[34]:



# binary

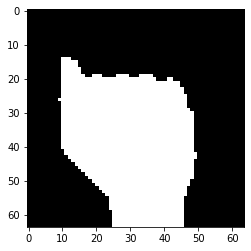
|  |
| --- |
| *#src -- image*  *#thresh*  *#max\_value*  *#type -- type of thersholding*  *#cv2.THRESH\_BINARY -- Binary Thersholding*  thresh, thresh\_img **=** cv2**.**threshold(img, 200, 255, cv2**.**THRESH\_BINARY) *#img* |

In [35]:

|  |
| --- |
| plt**.**imshow(thresh\_img) |

In [36]:

Out[36]:



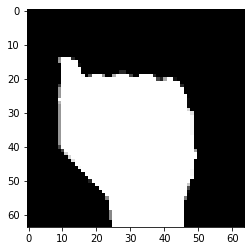
In [37]:

*#circle*

*#cv2.circle(image,(center coordinates),radius,(color), thickness)* circle **=** cv2**.**circle(img,(300,200),60,(255,0,0),5)

In [38]: plt**.**imshow(img)

Out[38]:



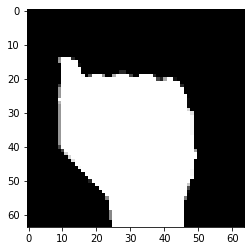
In [39]:

*#rectangle*

*#cv2.rectangle(img,(start coord),(end coord),color,thickness)* rectangle **=** cv2**.**rectangle(img,(200,100),(400,300),(0,0,255),10)

In [40]: plt**.**imshow(img)

Out[40]:



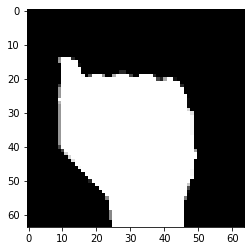
|  |
| --- |
| *#line*  *#cv2.line(img,(start coord),(end coord),color,thickness)*    line **=** cv2**.**line(img,(200,100),(400,300),(0,255,0),3) |

In [41]:

|  |
| --- |
| plt**.**imshow(img) |

In [42]:

Out[42]:



# creating or writing text an image

|  |
| --- |
| *#cv2.putText(img,text,(coord),fontstyle,fontscale,color,thickness)*  text **=** cv2**.**putText(img,"Opencv",(200,50),cv2**.**FONT\_HERSHEY\_SIMPLEX,2,(255,255,255 |

In [43]:

|  |
| --- |
| plt**.**imshow(img) |

) In [44]:

Out[44]:

