**EXPERIMENT – 01: PERFORMING OPERATION OF READING AND DISPLAYING IMAGE and video IN OPEN CV**

**AIM:** To read and display image and video file using open-cv in python.

**PROCEDURE/STEPS:**

**Step1:**Open new python file in jupyter notebook.

**Step2:**install Open cv using pip command

**Step3:** Write the code for reading and displaying the Image and video file

**Step 4:**Execute the code

**Program:**

pip install opencv**-**python

**import** cv2

**import** numpy **as** np

im1**=**cv2**.**imread("C:/Users/Administrator/Downloads/download.jfif")

cv2**.**imshow("image",im1)

cv2**.**waitKey(0)

cv2**.**destroyAllWindows()

view=cv2.VideoCapture("1.mp4")

while (view.isOpened()):

ret,frame=view.read()

cv2.imshow('video',frame)

if cv2.waitKey(1) & 0xFF == ord('q'):

break

view.release()

cv2.destroyAllWindows()

**RESULT:** The python program for reading the image and video file using open cv was executed and the output was displayed

**EXPERIMENT – 02: PERFORMING BASIC OPERATIONS ON AN IMAGE**

**AIM:** To perform basic operation such as Resize, Color space change, Blending of image, Blurring, subtraction, Pixel manipulation on an image using open-cv in python.

**PROCEDURE/STEPS:**

**Step1:**Open new python file in jupyter notebook.

**Step2:**install Open cv using pip command

**Step3:** Write the code for performing basic operation on an image

**Step 4:**Execute the code

**Program:**

**Import** numpy **as** np

**import** cv2

im1**=**cv2**.**imread("C:/Users/Administrator/Downloads/download.jfif")

im1\_resized**=**cv2**.**resize(im1,(400,300))

cv2**.**imshow("image",im1\_resized)

cv2**.**waitKey(0)

cv2**.**destroyAllWindows()

colors\_avl**=**[i **for** i **in**dir(cv2) **if**i**.**startswith("COLOR\_BGR")]

im3**=**cv2**.**cvtColor(im1,cv2**.**COLOR\_BGR2HSV)

cv2**.**imshow("color change",im3)

cv2**.**waitKey(0)

cv2**.**destroyAllWindows()

im\_blend=cv2.addWeighted(im3,0.2,im4,0.5,0)

imsub=cv2.subtract(im3,im4)

And=cv2.bitwise\_and(im3,im4)

Or=cv2.bitwise\_or(im3,im4)

Not=cv2.bitwise\_not(im3,im4)

Xor=cv2.bitwise\_xor(im3,im4)

Notor=cv2.bitwise\_not(Or)

cv2.imshow("blended",im\_blend)

cv2.waitKey(0)

cv2.imshow("sub",imsub)

cv2.waitKey(0)

cv2.imshow("and",And)

cv2.waitKey(0)

cv2.imshow("or",Or)

cv2.waitKey(0)

cv2.imshow("not",Not)

cv2.waitKey(0)

cv2.imshow("xor",Xor)

cv2.waitKey(0)

cv2.imshow("notor",Notor)

cv2.waitKey(0)

cv2.destroyAllWindows()

**RESULT:** The python program for performing basic operation on an image was executed and the output was displayed.

**EXPERIMENT – 03: PERFORMING GEOMETRIC OPERATIONS ON IMAGE**

**AIM:** To perform geomentric operation on image using open cv in python.

**PROCEDURE/STEPS:**

**Step1:**Open new python file in jupyter notebook.

**Step2:**install Open cv using pip command

**Step3:** Write the code for performing geometric operations on image

**Step 4:**Execute the code

**Program:**

import cv2

import numpy as np

im1=cv2.imread("13.jpg")

cv2.imshow("im1",im1)

cv2.waitKey(0)

cv2.destroyAllWindows()

shp=im1.shape

height,width=shp[:2]

cent=(height/2,width/2)

rotation\_matrix=cv2.getRotationMatrix2D(center=cent,angle=95,scale=0.5)

rotated\_im=cv2.warpAffine(src=im1,M=rotation\_matrix,dsize=(width,height))

cv2.imshow('Original image',im1)

cv2.waitKey(0)

cv2.imshow('Rotated image',rotated\_im)

cv2.waitKey(0)

cv2.destroyAllWindows()

tx,ty=width/2,height/2

trans\_matrix=np.array([[0,2,tx],[1,0,ty]])

trans\_im=cv2.warpAffine(src=im1,M=trans\_matrix,dsize=(width,height))

cv2.imshow('Original image',im1)

cv2.waitKey(0)

cv2.imshow('Rotated image',trans\_im)

cv2.waitKey(0)

cv2.destroyAllWindows()

**RESULT:** The python program for performing geometric operations on image was executed and the output was displayed.

**EXPERIMENT – 04: PERFORMING DIFFERENT IMAGE THRESHOLDING TECHNIQUE IN OPEN-CV**

**AIM:** To perform different image thresholding technique in open-cv in python.

**PROCEDURE/STEPS:**

**Step1:**Open new python file in jupyter notebook.

**Step2:**install Open cv using pip command

**Step3:** Write the code for performing different image thresholding technique in open-cv

**Step 4:**Execute the code

**Program:**

**import** cv2

**import** numpy **as** np

im1**=**cv2**.**imread("10.jpg",0)

im2**=**cv2**.**imread("9.jpg")

im3**=**cv2**.**resize(im1,(400,400))

im4**=**cv2**.**resize(im2,(400,400))

cv2**.**imshow("w1",im4)

cv2**.**imshow("w2",im3)

cv2**.**waitKey(0)

cv2**.**destroyAllWindows()

ret,thres**=**cv2**.**threshold(im1,209,240,cv2**.**THRESH\_BINARY)

cv2**.**imshow("thres",thres)

cv2**.**waitKey(0)

cv2**.**destroyAllWindows()

ret,thres**=**cv2**.**threshold(im1,209,240,cv2**.**THRESH\_BINARY)

ret,thres1**=**cv2**.**threshold(im1,200,240,cv2**.**THRESH\_BINARY\_INV)

ret,thres2**=**cv2**.**threshold(im1,209,240,cv2**.**THRESH\_TRUNC)

ret,thres3**=**cv2**.**threshold(im1,209,240,cv2**.**THRESH\_TOZERO)

ret,thres4**=**cv2**.**threshold(im1,210,240,cv2**.**THRESH\_TOZERO\_INV)

cv2**.**imshow("thres",thres)

cv2**.**waitKey(0)

cv2**.**imshow("thres\_1",thres1)

cv2**.**waitKey(0)

cv2**.**imshow("thres\_2",thres2)

cv2**.**waitKey(0)

cv2**.**imshow("thres\_3",thres3)

cv2**.**waitKey(0)

cv2**.**imshow("thres\_4",thres4)

cv2**.**waitKey(0)

cv2**.**destroyAllWindows()

**RESULT:**The python program forperforming different image thresholding technique in open-cvwas executed and the output was displayed.

**EXPERIMENT – 05: APPLICATION OF IMAGE FILTERS IN OPEN CV**

**AIM:**To perform image filters in open cvin python.

**PROCEDURE/STEPS:**

**Step1:**Open new python file in jupyter notebook.

**Step2:**install Open cv using pip command

**Step3:** Write the code for performing image filters in open cv

**Step 4:**Execute the code

**Program:**

**import** cv2

**import**numpy**as**np

**import**matplotlib.pyplot**as**plt

img**=**cv2**.**imread("11.jpg")

im1**=**cv2**.**imread("18.jpeg")

im2**=**cv2**.**resize(cv2**.**imread("23.jpeg"),(400,400))

im3**=**cv2**.**resize(cv2**.**imread("salt and pepper noise.jpeg"),(400,400))

kernal**=**np**.**ones((4,4),np**.**float32)**/**16

filter**=**cv2**.**filter2D(img,**-**1,kernal)

cv2**.**imshow("img",img)

cv2**.**imshow("filter",filter)

cv2**.**waitKey(0)

cv2**.**destroyAllWindows()

blur**=**cv2**.**blur(im1,(10,10))

cv2**.**imshow("img",im1)

cv2**.**imshow("blur",blur)

cv2**.**waitKey(0)

cv2**.**destroyAllWindows()

gblur**=**cv2**.**GaussianBlur(im2,(5,5),0)

cv2**.**imshow("img",im2)

cv2**.**imshow("gblur",gblur)

cv2**.**waitKey(0)

cv2**.**destroyAllWindows()

bilateral**=**cv2**.**bilateralFilter(im2,9,60,50)

cv2**.**imshow("img",im2)

cv2**.**imshow("bilateral",bilateral)

cv2**.**imshow("gblur",gblur)

cv2**.**waitKey(0)

cv2**.**destroyAllWindows()

medianFilter**=**cv2**.**medianBlur(im3,5)

cv2**.**imshow("img",im3)

cv2**.**imshow("medianFilter",medianFilter)

cv2**.**waitKey(0)

cv2**.**destroyAllWindows()

titles**=**['img','2d','im1','blur','im2','gaussian','bilateral','im3','median']

images**=**[img,filter,im1,blur,im2,gblur,bilateral,im3,medianFilter]

**for** i **in** range (len(images)):

plt**.**subplot(3,4,i**+**1),plt**.**imshow(images[i])

plt**.**title(titles[i])

im4**=**cv2**.**imread('24.jpeg',0)

im4**=**cv2**.**cvtColor(im4,cv2**.**COLOR\_BGR2RGB)

lap**=**cv2**.**Laplacian(im4,cv2**.**CV\_64F,ksize**=**1)

lap1**=**np**.**uint8(np**.**absolute(lap))

cv2**.**imshow("image",im4)

cv2**.**imshow("laplacian",lap)

cv2**.**imshow("Laplacian",lap1)

cv2**.**waitKey(0)

cv2**.**destroyAllWindows()

im5**=**cv2**.**imread('25.jpeg',0)

im5**=**cv2**.**cvtColor(im5,cv2**.**COLOR\_BGR2RGB)

sobelx**=**cv2**.**Sobel(im5,cv2**.**CV\_64F,1,0,ksize**=**3)

sobelX**=**np**.**uint8(np**.**absolute(sobelx))

cv2**.**imshow("img",im5)

cv2**.**imshow("sobelx",sobelx)

cv2**.**imshow("sobelX",sobelX)

cv2**.**waitKey(0)

cv2**.**destroyAllWindows()

sobely**=**cv2**.**Sobel(im5,cv2**.**CV\_64F,0,1,ksize**=**3)

sobelY**=**np**.**uint8(np**.**absolute(sobely))

cv2**.**imshow("img",im5)

cv2**.**imshow("sobely",sobely)

cv2**.**imshow("sobelY",sobelY)

cv2**.**waitKey(0)

cv2**.**destroyAllWindows()

titles**=**['im4','lap','lap1','sobelx','sobelX','sobely','sobely']

images**=**[im4,lap,lap1,sobelx,sobelX,sobely,sobelY]

**for** i **in** range (len(images)):

plt**.**subplot(2,4,i**+**1),plt**.**imshow(images[i])

plt**.**title(titles[i])

plt**.**xticks([]),plt**.**yticks([])

plt**.**show()

**RESULT:**The python program forperforming geometric operations on image was executed and the output was displayed.

**EXPERIMENT – 06: PERFORMING DIFFERENT MORPHOLOGICAL OPERATION USING OPEN-CV**

**AIM:**To perform different morphological operation using open-cv in python.

**PROCEDURE/STEPS:**

**Step1:**Open new python file in jupyter notebook.

**Step2:**install Open cv using pip command

**Step3:** Write the code for performing different morphological operation using open-cv

**Step 4:**Execute the code

**Program:**

**import** cv2

**import**numpy**as**np

**import**matplotlib.pyplot**as**plt

im1**=**cv2**.**imread("26.jpeg",0)

im1**=**cv2**.**cvtColor(im1,cv2**.**COLOR\_BGR2RGB)

\_,mask**=**cv2**.**threshold(im1,220,255,cv2**.**THRESH\_BINARY)

kernal**=**np**.**ones((29,29),np**.**uint8)

dilation**=**cv2**.**dilate(mask,kernal,iterations**=**1)

erode**=**cv2**.**erode(mask,kernal,iterations**=**1)

titles**=**['img','mask','dilation','erode']

images**=**[im1,mask,dilation,erode]

**for** i **in** range (len(images)):

plt**.**subplot(1,4,i**+**1),plt**.**imshow(images[i])

plt**.**title(titles[i])

plt**.**xticks([]),plt**.**yticks([])

plt**.**show()

**RESULT:**The python program forperforming different morphological operation using open-cv was executed and the output was displayed.

**EXPERIMENT – 07: IMAGE SEGMENTATION USING WATER SHED METHOD**

**AIM:**To perform image segmentation using water shed methodin python.

**PROCEDURE/STEPS:**

**Step1:**Open new python file in jupyter notebook.

**Step2:**install Open cv using pip command

**Step3:** Write the code for performing image segmentation using water shed method

**Step 4:**Execute the code

**Program:**

import cv2

import numpy as np

image=cv2.imread("10.jpg")

r=cv2.selectROI("select the area",image)

cropped\_image=image[int(r[1]):int(r[1]+r[3]),int(r[0]):int(r[0]+r[2])]

cv2.imshow("cropped image",cropped\_image)

cv2.waitKey(0)

**RESULT:**The python program forperforming image segmentation using water shed methodwas executed and the output was displayed.

**EXPERIMENT – 08: IMAGE SEGMENTATION USING MEAN SHIFT METHOD**

**AIM:**To perform image segmentation using mean shift method in python.

**PROCEDURE/STEPS:**

**Step1:**Open new python file in jupyter notebook.

**Step2:**install Open cv using pip command

**Step3:** Write the code for performing image segmentation using mean shift method

**Step 4:**Execute the code

**Program:**

import cv2

import numpy as np

image=cv2.imread("11.jpg")

segmented\_image=cv2.pyrMeanShiftFiltering(image,40,40)

cv2.imshow('original\_Image',image)

cv2.imshow('segmented\_image',segmented\_image)

cv2.waitKey(0)

cv2.destroyAllWindows()

**RESULT:**The python program forperforming image segmentation using mean shift method was executed and the output was displayed.

**EXPERIMENT – 09: IMAGE SEGMENTATION USING CLUSTERING METHOD**

**AIM:**To perform image segmentation using clustering method in python.

**PROCEDURE/STEPS:**

**Step1:**Open new python file in jupyter notebook.

**Step2:**install Open cv using pip command

**Step3:** Write the code for performing image segmentation using clustering method

**Step 4:**Execute the code

**Program:**

import cv2

import numpy as np

img=cv2.imread("17.jpeg")

imgrgb=cv2.cvtColor(img,cv2.COLOR\_BGR2RGB)

vec=imgrgb.reshape((-1,3))

vec=np.float32(vec)

criteria=(cv2.TERM\_CRITERIA\_EPS + cv2.TERM\_CRITERIA\_MAX\_ITER,100,1)

k = 3

attempts=100

compactness,label,center=cv2.kmeans(vec,k,None,criteria,attempts,cv2.KMEANS\_PP\_CENTERS)

center=np.uint8(center)

res=center[label.flatten()]

res\_img=res.reshape((imgrgb.shape))

cv2.imshow('original',img)

cv2.imshow('segmented',res\_img)

cv2.waitKey(0)

cv2.destroyAllWindows()

**RESULT:**The python program forperforming image segmentation using mean shift method was executed and the output was displayed.

**EXPERIMENT – 10: EDGE DETECTION IN OPEN-CV**

**AIM:**To perform edge detection in open-cvin python.

**PROCEDURE/STEPS:**

**Step1:**Open new python file in jupyter notebook.

**Step2:**install Open cv using pip command

**Step3:** Write the code for performing edge detection in open-cv

**Step 4:**Execute the code

**Program:**

import cv2

import numpy as np

im1=cv2.imread("13.jpg",0)

canny=cv2.Canny(im1,40,200)

cv2.imshow("im1",im1)

cv2.imshow("canny",canny)

cv2.waitKey(0)

cv2.destroyAllWindows()

**RESULT:**The python program forperforming edge detection in open-cvwas executed and the output was displayed.

**EXPERIMENT – 11: FEATURE EXTRACTION USING HOG**

**AIM:**To perform feature extraction usingHOGin python.

**PROCEDURE/STEPS:**

**Step1:**Open new python file in jupyter notebook.

**Step2:**install Open cv using pip command

**Step3:** Write the code for performing feature extraction usingHOG

**Step 4:**Execute the code

**Program:**

**RESULT:**The python program forperforming feature extraction usingHOGwas executed and the output was displayed.

**EXPERIMENT – 12: FEATURE DETECTION USING SIFT IN OPEN-CV**

**AIM:**To perform feature detection using sift in open-cvin python.

**PROCEDURE/STEPS:**

**Step1:**Open new python file in jupyter notebook.

**Step2:**install Open cv using pip command

**Step3:** Write the code for performing feature detection using sift in open-cv

**Step 4:**Execute the code

**Program:**

**RESULT:**The python program forperforming feature detection using sift in open-cvwas executed and the output was displayed.

**EXPERIMENT – 13: MOTION DETECTION USING OPTICAL FLOW**

**AIM:**To perform motion detection using optical flowin python.

**PROCEDURE/STEPS:**

**Step1:**Open new python file in jupyter notebook.

**Step2:**install Open cv using pip command

**Step3:** Write the code for performing motion detection using optical flow

**Step 4:**Execute the code

**Program:**

**RESULT:**The python program forperforming motion detection using optical flowwas executed and the output was displayed.

**EXPERIMENT – 14: FACE RECOGNITION USING HAARCASCADE METHOD**

**AIM:**To perform face recognition using haarcascade methodin python.

**PROCEDURE/STEPS:**

**Step1:**Open new python file in jupyter notebook.

**Step2:**install Open cv using pip command

**Step3:** Write the code for performing face recognition using haarcascade method

**Step 4:**Execute the code

**Program:**

**RESULT:**The python program forperforming face recognition using haarcascade methodwas executed and the output was displayed.

**EXPERIMENT – 15: OBJECT RECOGNITION USING TEMPLATE MATCHING**

**AIM:**To perform object recognition using template matchingin python.

**PROCEDURE/STEPS:**

**Step1:**Open new python file in jupyter notebook.

**Step2:**install Open cv using pip command

**Step3:** Write the code for performing object recognition using template matching

**Step 4:**Execute the code

**Program:**

**RESULT:**The python program forperforming object recognition using template matchingwas executed and the output was displayed.

**EXPERIMENT – 16: OBJECT TRACKING USING FEATURE DETECTION IN OPEN-CV**

**AIM:**To perform object tracking using feature detection in open-cv.

**PROCEDURE/STEPS:**

**Step1:**Open new python file in jupyter notebook.

**Step2:**install Open cv using pip command

**Step3:** Write the code for performing object tracking using feature detection in open-cv.

**Step 4:**Execute the code

**Program:**

**RESULT:**The python program forperforming object tracking using feature detection in open-cvwas executed and the output was displayed.