DIGITAL_IMAGE_PROCESSING_LAB_WORKSHEET_2

Reading and displaying the Image and Video using OpenCV

1.Read and Display Image

In [4]:

```
# Importing required modules
import numpy as np
import matplotlib.pyplot as plt
import cv2 as cv
```

In [5]:

```
# Defining and reading an Image usinig opencv module

img1 = cv.imread('./photoofme.jpg', cv.IMREAD_GRAYSCALE) # by using graycale module we can
```

In [9]:

```
# RESIZE IMAGE TO FIT IN SCREEN
scale_percent = 20 # percent of original size
width = int(img1.shape[1] * scale_percent / 100)
height = int(img1.shape[0] * scale_percent / 100)
dim = (width, height)
# resize image
img_res = cv.resize(img1, dim, interpolation = cv.INTER_AREA)
```

In [10]:

```
# Displaying the image
cv.imshow('Image window', img_res)

cv.waitKey(0)
cv.destroyAllWindows
```

Out[10]:

```
<function destroyAllWindows>
```

1.2 Reading , displaying and ploting the image using matplotlib in cv2.

In [ ]:

```
# Importing required modules
import numpy as np
import matplotlib.pyplot as plt
import cv2 as cv
```

In [11]:

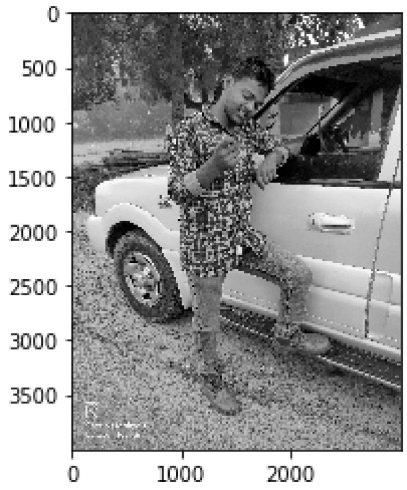
```
# Defining and reading an Image usinig opencv module

img2 = cv.imread('./photoofme.jpg', cv.IMREAD_GRAYSCALE)
```

In [12]:

```
# Displaying the image using matplotlib module

plt.imshow(img2, cmap='gray', interpolation='nearest')
plt.show()
```



1.3 READING AND SAVING AN IMAGE BY OPENCV2

In [13]:

```
# Defining and reading an Image usinig opencv module
img3 = cv.imread('./photoofme.jpg', cv.IMREAD_GRAYSCALE)
img3= np.full((512,512,3), 12, np.uint8)
```

In [14]:

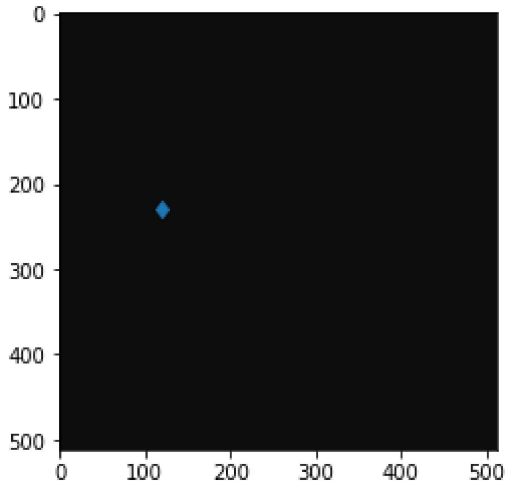
```
# Convert GBR colour mode to RGB colour mode
RGBimg = cv.cvtColor(img3, cv.COLOR_BGR2RGB)

# To plot on the image
plt.plot([120,700] , [230,500] , 'd' , linewidth=20)

# using matplotlib to display the image
plt.imshow(RGBimg)
```

Out[14]:

```
<matplotlib.image.AxesImage at 0x1d031145548>
```



In [15]:

```
# Save the image
cv.imwrite('./sample.JPEG' , RGBimg)  # THIS WILL SAVE A NEW IMAGE IN YOUR FOLDER
```

Out[15]:

```
True
```

2.WORKING WITH VEDIO

2.1 LOADING AND READING OF VEDIO FILE

In [16]:

```
# importing the necessary libraries
import cv2
import numpy as np

# Creating a VideoCapture object to read the video
cap = cv2.VideoCapture(0)
 # Define the codec and create VideoWriter object
fourcc = cv2.VideoWriter_fourcc(*'XVID')
out = cv2.VideoWriter('AMIT.avi',fourcc, 20.0, (640,480))

# Loop untill the end of the video
while (cap.isOpened()):

    # Capture frame-by-frame
    ret, frame = cap.read()
    if ret==True:
        frame = cv2.flip(frame,0)

        # write the flipped frame
        out.write(frame)
        cv2.imshow('frame',frame)
        if cv2.waitKey(1) & 0xFF == ord('q'):
            break
    else:
        break


# release the video capture object
cap.release()
# Closes all the windows currently opened.
cv2.destroyAllWindows()
```

2.2 Reading an vedio from local file

In [35]:

```
# importing libraries
import cv2
import numpy as np

# Create a VideoCapture object and read from input file
cap = cv2.VideoCapture('MVI_7813.MOV')

# Check if camera opened successfully
if (cap.isOpened()== False):
  print("Error opening video  file")

# Read until video is completed
while(cap.isOpened()):

  # Capture frame-by-frame
  ret, frame = cap.read()
  if ret == True:

    # Display the resulting frame
    cv2.imshow('Frame', frame)

    # Press Q on keyboard to  exit
    if cv2.waitKey(25) & 0xFF == ord('q'):
      break

  # Break the loop
  else:
    break

# When everything done, release
# the video capture object
cap.release()

# Closes all the frames
cv2.destroyAllWindows()
```

2.3 Capture an vedio in GBLUR form

In [30]:

```
# importing the necessary libraries
import cv2
import numpy as np

# Creating a VideoCapture object to read the video
cap = cv2.VideoCapture(0)

# Define the codec and create VideoWriter object
fourcc = cv2.VideoWriter_fourcc(*'XVID')
out = cv2.VideoWriter('AMIT2.mp4',fourcc, 20.0, (640,480))


# Loop untill the end of the video
while (cap.isOpened()):

    # Capture frame-by-frame
    ret, frame = cap.read()
    frame = cv2.resize(frame, (540, 380), fx = 0, fy = 0,
                        interpolation = cv2.INTER_CUBIC)

    # Display the resulting frame
    cv2.imshow('Frame', frame)

    # using cv2.Gaussianblur() method to blur the video

    # (5, 5) is the kernel size for blurring.
    gaussianblur = cv2.GaussianBlur(frame, (5, 5), 0)
    cv2.imshow('gblur', gaussianblur)

    # define q as the exit button
    if cv2.waitKey(25) & 0xFF == ord('q'):
        break
# release the video capture object
cap.release()
# Closes all the windows currently opened.
cv2.destroyAllWindows()
```

2.4 Capture an vedio in GrayScale form

In [28]:

```
# importing the necessary libraries
import cv2
import numpy as np

# Creating a VideoCapture object to read the video
cap = cv2.VideoCapture(0)

# Define the codec and create VideoWriter object
fourcc = cv2.VideoWriter_fourcc(*'XVID')
out = cv2.VideoWriter('output2.mp4',fourcc, 20.0, (640,480))


# Loop untill the end of the video
while (cap.isOpened()):

    # Capture frame-by-frame
    ret, frame = cap.read()
    frame = cv2.resize(frame, (540, 380), fx = 0, fy = 0,
                        interpolation = cv2.INTER_CUBIC)

    # Display the resulting frame
    cv2.imshow('Frame', frame)

    # conversion of BGR to grayscale is necessary to apply this operation
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

    # adaptive thresholding to use different threshold
    # values on different regions of the frame.
    Thresh = cv2.adaptiveThreshold(gray, 255, cv2.ADAPTIVE_THRESH_MEAN_C,
                                           cv2.THRESH_BINARY_INV, 11, 2)

    cv2.imshow('Thresh', Thresh)
    # define q as the exit button
    if cv2.waitKey(25) & 0xFF == ord('q'):
        break
  # release the video capture object
cap.release()
# Closes all the windows currently opened.
cv2.destroyAllWindows()
```

Capture a vedio in Flipped mode

In [34]:

```
# importing the necessary libraries
import cv2
import numpy as np

# Creating a VideoCapture object to read the video
cap = cv2.VideoCapture(0)
 # Define the codec and create VideoWriter object
fourcc = cv2.VideoWriter_fourcc(*'XVID')
out = cv2.VideoWriter('AMIT3.mp4',fourcc, 20.0, (640,480))

# Loop untill the end of the video
while (cap.isOpened()):

    # Capture frame-by-frame
    ret, frame = cap.read()
    if ret==True:
        frame = cv2.flip(frame,0)

        # write the flipped frame
        out.write(frame)
        cv2.imshow('frame',frame)
        if cv2.waitKey(25) & 0xFF == ord('q'):
            break
    else:
        break
# release the video capture object
cap.release()
# Closes all the windows currently opened.
cv2.destroyAllWindows()
```

THANK YOU FOR READING