SIR C R REDDY COLLEGE OF ENGINEERING

ELURU

Certificate

This is to certify that this is the **BONAFIDE RECORD** *of the work done in* **SOC-IV- VIDEO ANALYTICS (R2032124)** *laboratory by Mr/Ms_____ bearing Regd. No,_____ of III/IVB.Tech – II Semester course during 2023-2024*

Total number of Experiments held…….

Total number of Experiments done……..

LAB – IN– CHARGE

HEAD OF THE DEPATRMENT

EXTERNAL EXAMINER

SOC-IV

VIDEO ANALYTICS LABORATORY

MANUAL

(R2032124)

III/IV B.Tech, Semester-II

Academic Year: 2023-2024

DEPARTMENT OF INFORMATION TECHNOLOGY

SIR C R REDDY COLLEGE OF ENGINEERING

Eluru-534007, West Godavari Dist, Andhra Pradesh, India

(Approved by AICTE, New Delhi & Permanently affiliated to JNTUK, Kakinada)

Accredited by NBA

Telephone No: 08812-230840, 230565, Fax: 08812-224193

Website: www.sircrrengg.ac.in

SIR C.R.REDDY COLLEGE OF ENGINEERING
ELURU-534007, WEST GODAVARI DIST, A P., INDIA
(Approved by AICTE, New Delhi, Accredited by NBA)
Phone no: 08812-230840, 2300656 Fax: 08812-224193
Visit us at http://www.sircrrengg.ac.in
DEPARTMENT OF INFORMATION TECHNOLOGY

VISION OF THE DEPARTMENT

To be a premier Department in the region in the field of Information Technology through academic excellence and research that enable graduates to meet the challenges of industry and society

MISSION OF THE DEPARTMENT

- To provide dynamic teaching-learning environment to make the students industry ready and advancement in career;
- To inculcate professional and leadership quality for better employability and entrepreneurship;
- To make high quality professional with moral and ethical values suitable for industry and society

SIR C.R.REDDY COLLEGE OF ENGINEERING
ELURU-534007, WEST GODAVARI DIST, A P., INDIA
(Approved by AICTE, New Delhi, Accredited by NBA)
Phone no: 08812-230840, 2300656 Fax: 08812-224193
Visit us at http://www.sircrrengg.ac.in
DEPARTMENT OF INFORMATION TECHNOLOGY

PROGRAM OUTCOMES (POs)

PO1: Engineering Knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

PO2: Problem Analysis: Identify, formulate, research literature, and analyze complex engineering problems reaching substantiated conclusions using the first principles of mathematics, natural sciences, and engineering sciences.

PO3: Design/Development of Solutions: Design solutions for complex engineering problems and system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, society, and environmental considerations.

PO4: Conduct Investigations of Complex Problems: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

PO5: Modern Tool Usage: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

PO6: The Engineer and Society: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

PO7: Environment and Sustainability: Understand the impact of the professional engineering solutions in society and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

PO8: Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

PO9: Individual and Team Work: Function effectively as an individual, and as a member or leader in diverse teams, and in multi-disciplinary settings.

PO10: Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

PO11: Project Management and Finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multi-disciplinary environments.

PO12: Life-long Learning: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

SIR C.R.REDDY COLLEGE OF ENGINEERING
ELURU-534007, WEST GODAVARI DIST, A P., INDIA
(Approved by AICTE, New Delhi, Accredited by NBA)
Phone no: 08812-230840, 2300656 Fax: 08812-224193
Visit us at http://www.sircrrengg.ac.in
DEPARTMENT OF INFORMATION TECHNOLOGY

PROGRAM SPECIFIC OUTCOMES (PSOs)

PSO1: Apply core information technology of system,architecture , information management , programming, networking for development of current technical concepts.

PSO2: Design and develop software by adopting emerging technologies for the need of IT industry.

PROGRAM EDUCATIONAL OBJECTIVES (PEOs)

PEO1: Solve real world problems through effective professional skills in Information Technology industry and academic research.

PEO2: Analyze and develop applications in Information Technology domain and adapt to changing technology trends with continuous learning.

PEO3: Practice the profession in society with ethical and moral values.

SIR C.R.REDDY COLLEGE OF ENGINEERING
ELURU-534007, WEST GODAVARI DIST, A P., INDIA
(Approved by AICTE, New Delhi, Accredited by NBA)
Phone no: 08812-230840, 2300656 Fax: 08812-224193
Visit us at http://www.sircrrengg.ac.in
DEPARTMENT OF INFORMATION TECHNOLOGY

GENERAL LABORATORY INSTRUCTIONS

1. Students are advised to come to the laboratory at least 5 minutes before (to the starting time), those who come after 5 minutes will not be allowed into the lab.
2. Plan your task properly much before to the commencement, come prepared to the lab with the synopsis / program / experiment details.
3. Student should enter into the laboratory with:
    a. Laboratory observation notes with all the details (Problem statement, Aim, Algorithm, Procedure, Program, Expected Output, etc.,) filled in for the lab session.
    b. Laboratory Record updated up to the last session experiments and other utensils (if any) needed in the lab.
    c. Proper Dress code and Identity card.
4. Sign in the laboratory login register, write the TIME-IN, and occupy the computer system allotted to you by the faculty.
5. Execute your task in the laboratory, and record the results / output in the lab observation note book, and get certified by the concerned faculty.
6. All the students should be polite and cooperative with the laboratory staff, must maintain the discipline and decency in the laboratory.
7. Computer labs are established with sophisticated and high end branded systems, which should be utilized properly.
8. Students / Faculty must keep their mobile phones in SWITCHED OFF mode during the lab sessions.Misuse of the equipment, misbehaviors with the staff and systems etc., will attract severe punishment.
9. Students must take the permission of the faculty in case of any urgency to go out ; if anybody found loitering outside the lab / class without permission during working hours will be treated seriously and punished appropriately.
10. Students should LOG OFF/ SHUT DOWN the computer system before he/she leaves the lab after completing the task (experiment) in all aspects. He/she must ensure the system / seat is kept properly.

SIR C.R.REDDY COLLEGE OF ENGINEERING
ELURU-534007, WEST GODAVARI DIST, A P., INDIA
(Approved by AICTE, New Delhi, Accredited by NBA)
Phone no: 08812-230840, 2300656 Fax: 08812-224193
Visit us at http://www.sircrrengg.ac.in
DEPARTMENT OF INFORMATION TECHNOLOGY

SYLLABUS

List of Experiments:

1. Image enhancement application
2. Object/image recognition applications based on digital image transforms.
3. Image analysis systems for object recognition.
4. Content Based Image Retrieval
5. Image compression, Image Fusion
6. Image Steganography, Watermarking
7. Video Enhancement and Noise Reduction
8. Motion Estimation and Segmentation,
9. Motion Tracking
10. Multi-target/Multi-camera tracking
11. Action Recognition

List of Augmented Experiments:

1. Implement Optical character recognition.
2. Implement Gesture Recognition.

SIR C.R.REDDY COLLEGE OF ENGINEERING
ELURU-534007, WEST GODAVARI DIST, A P., INDIA
(Approved by AICTE, New Delhi, Accredited by NBA)
Phone no: 08812-230840, 2300656 Fax: 08812-224193
Visit us at http://www.sircrrengg.ac.in
DEPARTMENT OF INFORMATION TECHNOLOGY

COURSE OUTCOMES

CO	OUTCOMES
CO1	Discuss the principles and techniques of digital image and the fundamentals of digital video processing
CO2	Apply the image recognition and motion recognition in videos
CO3	Apply the motion estimation, segmentation and modeling concepts in video applications.
CO4	Analyze video processing in applications

CO-PO MAPPING

	PO 1	PO 2	PO 3	PO 4	PO 5	PO 6	PO 7	PO 8	PO 9	PO 10	PO 11	PO 12	PSO 1	PSO 2
CO1			3		2									
CO2			3	2	2								2	
CO3			3	3	2								2	
CO4		2	3	2	2								2	
AVG		**2**	**3**	**2.33**	**2**								**2**	

SIR C.R.REDDY COLLEGE OF ENGINEERING
ELURU-534007, WEST GODAVARI DIST, A P., INDIA
(Approved by AICTE, New Delhi, Accredited by NBA)
Phone no: 08812-230840, 2300656 Fax: 08812-224193
Visit us at http://www.sircrrengg.ac.in
DEPARTMENT OF INFORMATION TECHNOLOGY

CO-PO MAPPING JUSTIFICATION

CO1	**PO3**	**3**	This knowledge of fundamentals of digital image processing and video processing helps students to model solutions for complex engineering problems.
CO1	**PO5**	**2**	The ability to take the aid of modern tools and apply appropriate techniques helps in various applications of computer vision.
CO2	**PO3**	**3**	This knowledge of image and motion recognition helps in designing solutions for engineering problems.
CO2	**PO4**	**2**	By gaining this ability the students will be able to solve complex engineering problems from user's perspective in the domain of computer vision with better effectiveness.
CO2	**PO5**	**2**	Students get the ability to perform image recognition and motion recognition in videos using modern engineering and IT tools.
CO3	**PO3**	**3**	This knowledge helps in finding solutions for motion analysis and segmentation of videos.
CO3	**PO4**	**3**	This knowledge helps in designing of experiments, analysis and interpretation the videos for video segmentation applications.
CO3	**PO5**	**2**	This knowledge helps to select, and apply appropriate modern tools for video analytics.
CO4	**PO2**	**2**	This knowledge helps to analyze and design models for complex engineering problems in video analytics.
CO4	**PO3**	**3**	This knowledge helps to analyze and interpret the images and videos to provide solutions.
CO4	**PO4**	**2**	This knowledge helps to apply IT tools for modeling to complex engineering activities.
CO4	**PO5**	**2**	This knowledge helps to use modern engineering tools like anaconda and write effective programs.

CO-PO MAPPING JUSTIFICATION

CO2	**PSO1**	**2**	Apply the image recognition and motion recognition in videos
CO3	**PSO1**	**2**	Apply core information technology of system ,architecture , information management , programming, networking for development of current technical concepts.
CO4	**PSO1**	**2**	This knowledge helps to apply and analyze image processing and video processing concepts and write programs using python language.

SIR C.R.REDDY COLLEGE OF ENGINEERING
ELURU-534007, WEST GODAVARI DIST, A P., INDIA
(Approved by AICTE, New Delhi, Accredited by NBA)
Phone no: 08812-230840, 2300656 Fax: 08812-224193
Visit us at http://www.sircrrengg.ac.in
DEPARTMENT OF INFORMATION TECHNOLOGY

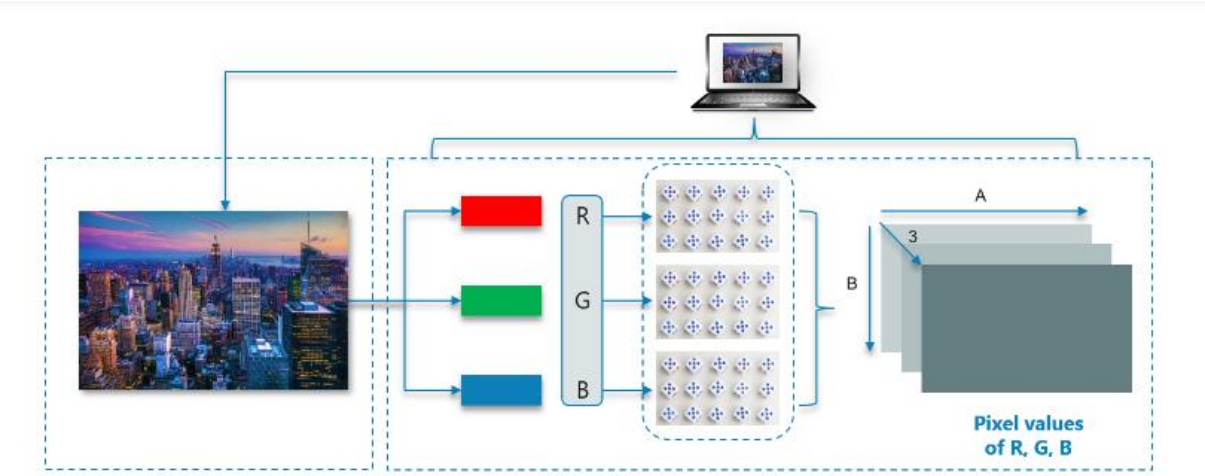
INDEX

Introduction

Computer Vision is an interdisciplinary field that deals with how computers can be made to gain a high-level understanding from digital images or videos.The idea here is to automate tasks that the human visual systems can do. So, a computer should be able to recognize objects such as that of a face of a human being or a lamppost or even a statue.

How Does A Computer Read An Image:

The computer reads any image as a range of values between 0 and 255. For any color image, there are 3 primary channels – Red, green and blue. A matrix is formed for every primary color and later these matrices combine to provide a Pixel value for the individual R, G, Bcolors. Each element of the matrices provides data pertaining to the intensity of brightness of the pixel.



the size of the image here can be calculated as B x A x 3. For a black-white image, there is only one single channel.

OpenCV:

OpenCV is a Python library which is designed to solve computer vision problems. OpenCV was originally developed in 1999 by Intel but later it was supported by Willow Garage.OpenCV supports a wide variety of programming languages such as C++, Python, Java etc. Support for multiple platforms including Windows, Linux, and MacOS.

OpenCV Python is nothing but a wrapper class for the original C++ library to be used with Python. Using this, all of the OpenCV array structures gets converted to/from NumPy arrays.

This makes it easier to integrate it with other libraries which use NumPy. For example, libraries such as SciPy and Matplotlib.

Sample Image:

Basic Operations WithOpenCV

```
Import cv2

# colored Image

img = cv2.imread ('C:\\Users\sircrrengg\Pictures\Penguins.jpg',1)

# Black and White (gray scale)

img = cv2.imread ('C:\\Users\sircrrengg\Pictures\Penguins.jpg',0)
```

The first requirement is to import the OpenCVmodule.Later we can read the image using **imread** module. The 1 in the parameters denotes that it is a color image. If the parameter was 0 instead of 1, it would mean that the image being imported is a black and white image. The name of the image here is 'Penguins'.

Displaying the image:

```
import cv2

# Black and White (gray scale)

Img = cv2.imread ('C:\\Users\sircrrengg\Pictures\Penguins.jpg',0)

cv2.imshow('image', img)

cv2.waitKey(0)

# cv2.waitKey(2000)

cv2.destroyAllWindows()
```

we first import the image using imread. We require a window output to display the images. We use the imshow function to display the image by opening a window. There are 2 parameters to the imshow function which is the name of the window and the image object to be displayed. Later, we wait for a user event**.** waitKey makes the window static until the user presses a key. The parameter passed to it is the time in milliseconds. And lastly, we use destroyAllWindows to close the window based on the waitForKey parameter.

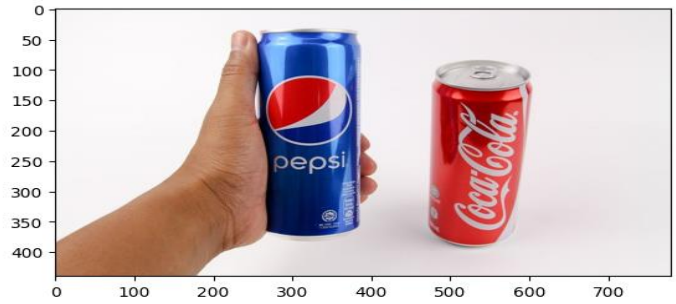
Experiment 1

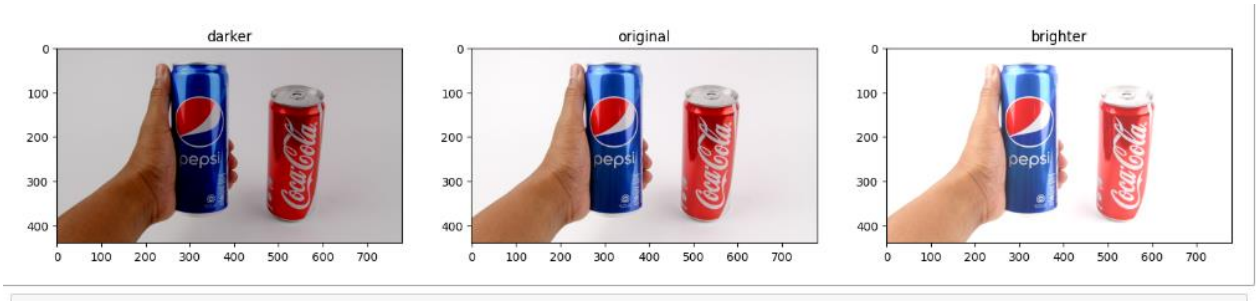
Aim: To perform Image enhancement

Image enhancement is the process of digitally manipulating a stored image.

```
import cv2
importnumpy as np
importmatplotlib.pyplot as plt
img_bgr = cv2.imread("image.jpg", cv2.IMREAD_COLOR)
img_rgb = cv2.cvtColor(img_bgr,cv2.COLOR_BGR2RGB)
plt.imshow(img_rgb)
matrix = np.ones(img_rgb.shape, dtype = "uint8") * 50
img_rgb_brighter = cv2.add(img_rgb,matrix)
img_rgb_darker = cv2.subtract(img_rgb,matrix)
plt.figure(figsize = [18,5])
plt.subplot(131);plt.imshow(img_rgb_darker); plt.title("darker");
plt.subplot(132);plt.imshow(img_rgb); plt.title("original");
plt.subplot(133);plt.imshow(img_rgb_brighter); plt.title("brighter");
```

Output:





Experiment -2

Aim: To perform object/image recognition based on digital image transform

Image Reflection:

Image reflection is used to flip the image vertically or horizontally. For reflection along the X-axis, we set the value of Y to -1 , X to 1,and vice-versa for the Y-axis reflection.

```
importnumpy as np
import cv2 as cv
importmatplotlib.pyplot as plt
img = cv.imread('dog.jpg',0)
rows, cols = img.shape
M = np.float32([[1, 0, 0],[0, -1, rows],[0, 0, 1]])
reflected_img = cv.warpPerspective(img, M,(int(cols),int(rows)))
cv.imshow("img",reflected_img)
cv.waitKey(0)
cv.destroyAllWindows()
```

OUTPUT:

Image Rotation:

In image rotation the image is rotated by a definite angle.By using this warpAffine function to make a matrix that can give a required rotation angle(by 30 degrees)

```
importnumpy as np
import cv2 as cv
img = cv.imread('dog.jpg', 0)
rows, cols = img.shape
M = np.float32([[1,  0, 0], [0, -1, rows], [0,  0, 1]])
img_rotation = cv.warpAffine(img,cv.getRotationMatrix2D((cols/2, rows/2),30, 0.6),(cols, rows))
cv.imshow('img', img_rotation)
cv.imwrite('rotation_out.jpg', img_rotation)
cv.waitKey(0)
cv.destroyAllWindows()
```

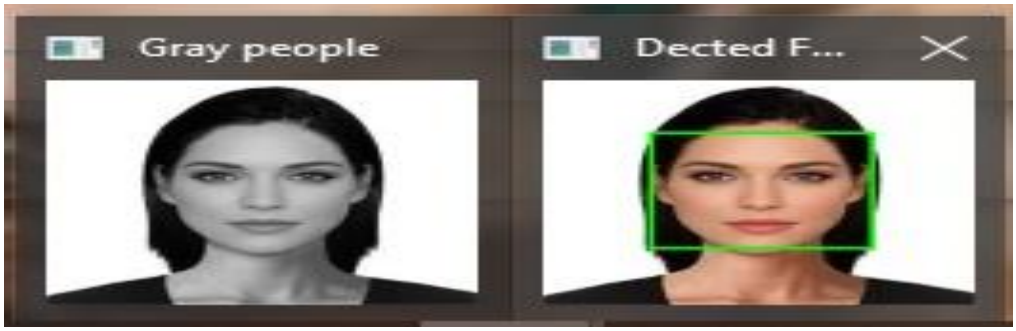
OUTPUT:

Experiment -3

Aim: To perform image analysis system for object recognition.

```
importnumpy as np
import cv2 as cv
img = cv.imread('C:/Users/sircrrengg/Desktop/face1.jpg')
gray = cv.cvtColor(img,cv.COLOR_BGR2GRAY)
cv.imshow('Gray people',gray)
haar_cascade = cv.CascadeClassifier('frontface.xml')
faces_rect = haar_cascade.detectMultiScale(gray,scaleFactor = 1.1, minNeighbors = 3)
for(x,y,w,h) in faces_rect:
cv.rectangle(img,(x,y),(x+w,y+h),(0,255,0),thickness = 2)
cv.imshow('Dected Faces',img)
cv.waitKey(0)
```

OUTPUT:



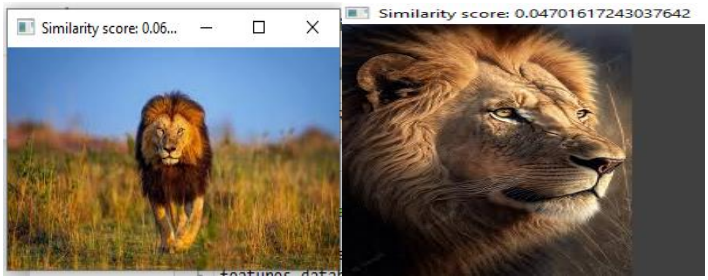
Experiment -4

Aim: To perform content based image retrieval.

```
import cv2
importnumpy as np
# Define the query image
query_image = cv2.imread('C:/Users/sircrrengg/Desktop/testlion.jpg')
# Define the database of images
database_images=['C:/Users/sircrrengg/Desktop/lion1.jpg',
'C:/Users/sircrrengg/Desktop/lion2.jpg',
'C:/Users/sircrrengg/Desktop/monkey.jpg','C:/Users/sircrrengg/Desktop/lion3.jpg']
# Create an empty list to store the features of the images
features_database = []
# Define the feature extractor function (in this case, we use color histogram)
defcolor_histogram(image):
hist = cv2.calcHist([image], [0, 1, 2], None, [8, 8, 8], [0, 256, 0, 256, 0, 256])
hist = cv2.normalize(hist, hist).flatten()
returnhist
# Extract features from the database images and store them in the list
forimage_name in database_images:
image = cv2.imread(image_name)
features = color_histogram(image)
features_database.append(features)
# Extract features from the query image
query_features = color_histogram(query_image)
# Find the similarity scores between the query image and the database images
similarity_scores = []
for features in features_database:
score = cv2.compareHist(query_features, features, cv2.HISTCMP_CORREL)
similarity_scores.append(score)
# Sort the similarity scores in descending order
sorted_indexes = np.argsort(similarity_scores)[::-1]
# Display the top 3 most similar images
for i in range(3):
```

```
index = sorted_indexes[i]
image_name = database_images[index]
similarity_score = similarity_scores[index]
image = cv2.imread(image_name)
cv2.imshow(f"Similarity score: {similarity_score}", image)
cv2.waitKey(0)
# Release resources
cv2.destroyAllWindows()
```

Output:



Experiment-5

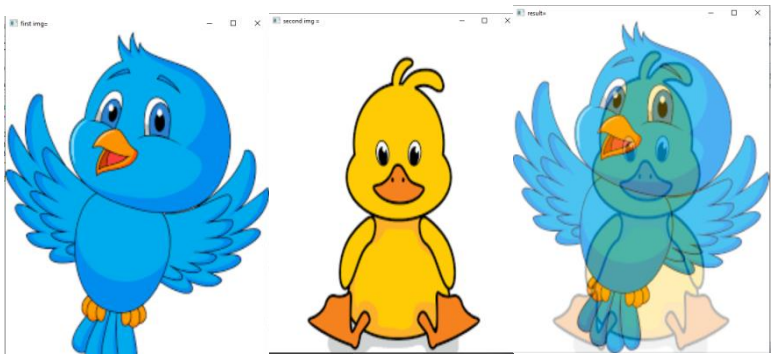
Aim: To perform Image compression/Image Fusion

Image Compression:

```
import cv2
importnumpy as np
def save(path, image, jpg_quality=None, png_compression=None):
ifjpg_quality:
cv2.imwrite(path, image, [int(cv2.IMWRITE_JPEG_QUALITY), jpg_quality])
elifpng_compression:
cv2.imwrite(path, image, [int(cv2.IMWRITE_PNG_COMPRESSION), png_compression])
else:
cv2.imwrite(path, image)
imgpath = "face1.jpg"
img = cv2.imread(imgpath)
cv2.imshow('face', img)        #display the image
outpath_jpeg = "C:/Users/CPNM LAB/Desktop/compresedjpg.jpg"          # save the image in JPEG format with 85% quality
save(outpath_jpeg,img,jpg_quality=85)
outpath_png = "C:/Users/CPNM LAB/Desktop/compresedPNG.png"
save(outpath_png, img,png_compression=4)          # save the image in PNG format with 4 Compression
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Image Fusion:

```
Importcv2
import numpy as np
img1= cv2.imread("Desktop/im.jpg")
img1 = cv2.resize(img1, (500,700))
cv2.imshow("first img=",img1)
img2= cv2.imread("Desktop/im2.jpg")
img2 = cv2.resize(img2, (500,700))
cv2.imshow("second img =",img2)
result = cv2.addWeighted(img1,0.7,img2,0.3,0)
cv2.imshow("result=",result)
cv2.waitKey(0)
```



Experiment-6

Aim: To perform Image watermarking

```
import cv2
img = cv2.imread("Desktop/crr.jpg")
wm = cv2.imread("Desktop/crr1.jpg")
h_wm, w_wm = wm.shape[:2]
h_img, w_img = img.shape[:2]
center_x = int(w_img/2)
center_y = int(h_img/2)
top_y = center_y - int(h_wm/2)
left_x = center_x - int(w_wm/2)
bottom_y = top_y + h_wm
right_x = left_x + w_wm
roi = img[top_y:bottom_y, left_x:right_x]
result = cv2.addWeighted(roi, 1, wm, 0.3, 0)
img[top_y:bottom_y, left_x:right_x] = result
cv2.imshow("Watermarked Image", img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

OUTPUT:

Experiment-7

Aim: To perform image enhancement and noise reduction

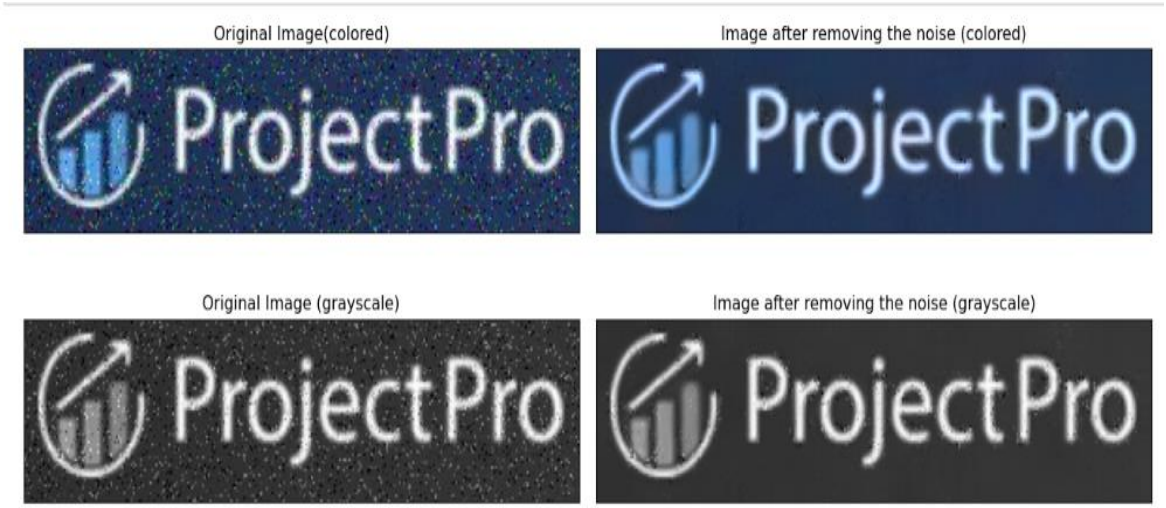
Image enhancement refers to the process of highlighting certain information of an image, as well as weakening or removing any unnecessary information according to specific needs. For example, eliminatingnoise, revealing blurred details, and adjusting levels to highlight features of an image.

Parameters for fastNlMeansDenoising function are given below:

- src: The input image
- dst: The output dimension
- h: The strength of the filter. 10 is the ideal value
- hForColorComponents: Same as h but for colored images
- templateWindowSize: The window size of the template. 7 is the ideal value.
- searchWindowSize: The window size of the search area. 21 is the ideal value.

```
importnumpy as np
import cv2
frommatplotlib import pyplot as plt
image = cv2.imread('C:/Users/CPNM LAB/Desktop/noiseimg.jpg',1)
image_bw = cv2.imread('C:/Users/CPNM LAB/Desktop/noiseimg.jpg',0)
noiseless_image_bw = cv2.fastNlMeansDenoising(image_bw, None, 20, 7, 21)
noiseless_image_colored = cv2.fastNlMeansDenoisingColored(image,None,20,20,7,21)
titles = ['Original Image(colored)','Image after removing the noise (colored)', 'Original Image (grayscale)','Image after removing the noise (grayscale)']
images = [image,noiseless_image_colored, image_bw, noiseless_image_bw]
plt.figure(figsize=(13,5))
for i in range(4):
plt.subplot(2,2,i+1)
plt.imshow(cv2.cvtColor(images[i],cv2.COLOR_BGR2RGB))
plt.title(titles[i])
plt.xticks([])
plt.yticks([])
```

```
plt.tight_layout()
plt.show()
```



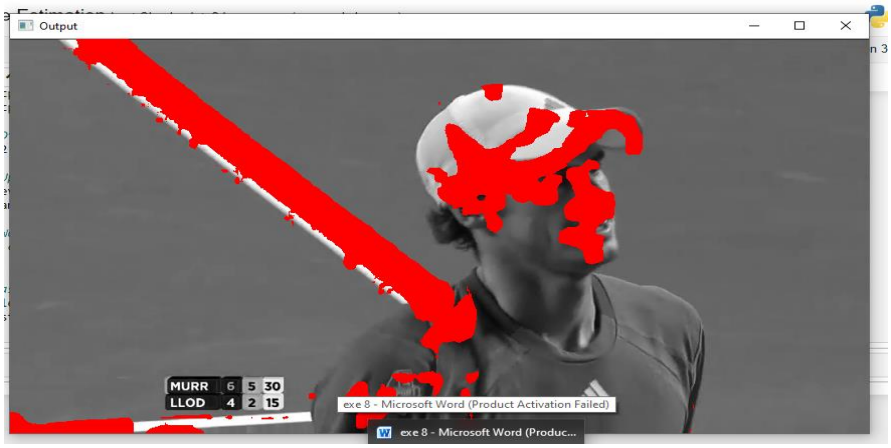
Experiment -8

Aim: To perform motion estimation and segmentation

```
import cv2
# Load video file
cap = cv2.VideoCapture("tennisgame.mp4")
# Initialize variables
frame_count = 0
prev_frame = None
# Loop through video frames
while True:
    # Read frame
ret, frame = cap.read()
    # Check if frame was successfully read
if not ret:
break
    # Convert frame to grayscale
gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    # Perform motion estimation and segmentation on first frame
iframe_count == 0:
prev_frame = gray
frame_count += 1
continue
    # Calculate optical flow using Lucas-Kanade method
flow = cv2.calcOpticalFlowFarneback(prev_frame, gray, None, 0.5, 3, 15, 3, 5, 1.2, 0)
    # Compute magnitude and direction of flow vectors
mag, ang = cv2.cartToPolar(flow[..., 0], flow[..., 1])
    # Threshold magnitude to identify moving pixels
threshold = 10
mask = mag > threshold
    # Create output image with moving pixels in red
output = cv2.cvtColor(gray, cv2.COLOR_GRAY2BGR)
output[mask] = [0, 0, 255]
    # Display output image
```

```
cv2.imshow("Output", output)
    # Update variables for next frame
prev_frame = gray.copy()
frame_count += 1
    # Wait for key press
if cv2.waitKey(1) == ord("q"):
break
# Release video file and close windows
cap.release()
cv2.destroyAllWindows()
```

OUTPUT:



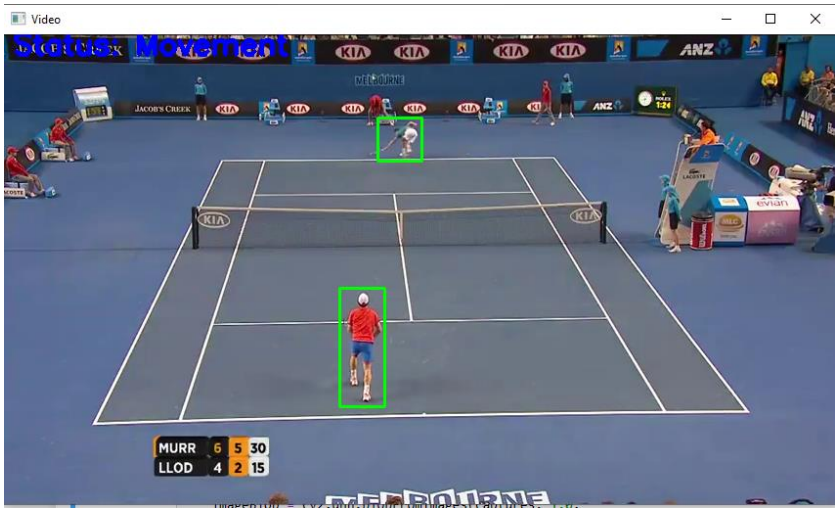
Experiment -9

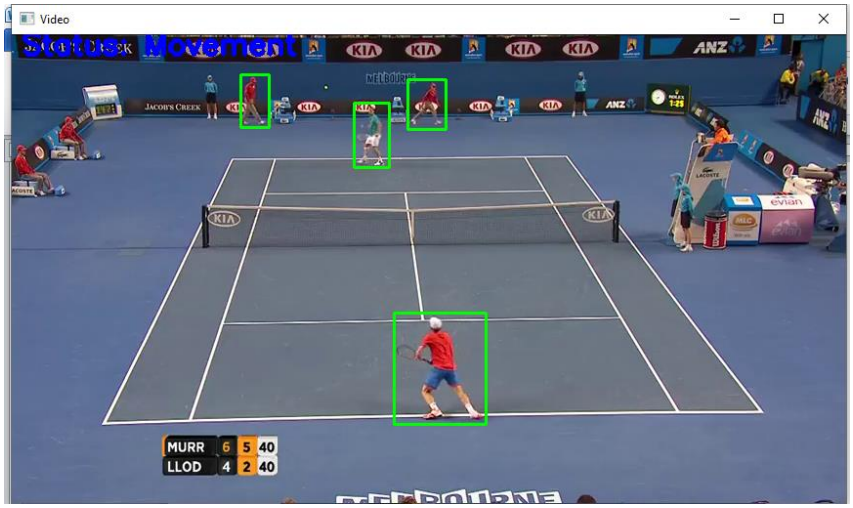
Aim: To perform Motion Tracking

```
import cv2 as cv
import numpy as np
from matplotlib import pyplot as plt
defmotionDetection():
    cap = cv.VideoCapture("Downloads/tennisgame.mp4")
    ret, frame1 = cap.read()
    ret, frame2 = cap.read()
    while cap.isOpened():
        diff = cv.absdiff(frame1, frame2)
        diff_gray = cv.cvtColor(diff, cv.COLOR_BGR2GRAY)
        blur = cv.GaussianBlur(diff_gray, (5, 5), 0)
        _, thresh = cv.threshold(blur, 20, 255, cv.THRESH_BINARY)
        dilated = cv.dilate(thresh, None, iterations=3)
        contours, _ = cv.findContours(
            dilated, cv.RETR_TREE, cv.CHAIN_APPROX_SIMPLE)
        for contour in contours:
            (x, y, w, h) = cv.boundingRect(contour)
            if cv.contourArea(contour) < 900:
                continue
            cv.rectangle(frame1, (x, y), (x+w, y+h), (0, 255, 0), 2)
            cv.putText(frame1, "Status: {}".format('Movement'), (10, 20),
cv.FONT_HERSHEY_SIMPLEX,1, (255, 0, 0), 3)
        # cv.drawContours(frame1, contours, -1, (0, 255, 0), 2)
        cv.imshow("Video", frame1)
        frame1 = frame2
        ret, frame2 = cap.read()
        if cv.waitKey(50) == 27:
            break
    cap.release()
    cv.destroyAllWindows()
```

```
if __name__ == "__main__":
    motionDetection()
```

OUTPUT:





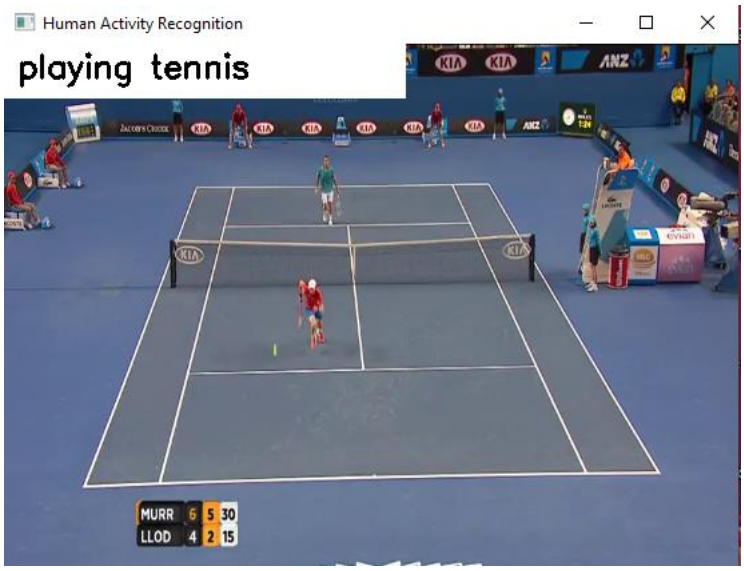
Experiment -10

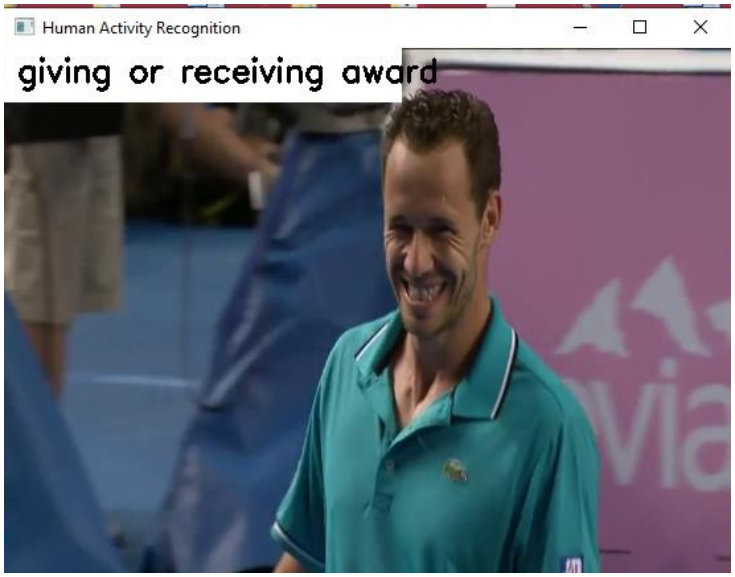
Aim: To perform Action Recognition

```
# Required imports
from collections import deque
importnumpy as np
import cv2
# Parameters class include important paths and constants
class Parameters:
def __init__(self):
self.CLASSES = open("human/model/action_recognition_kinetics.txt").read().strip().split("\n")
self.ACTION_RESNET = 'human/model/resnet-34_kinetics.onnx'
      #self.VIDEO_PATH = None
self.VIDEO_PATH = "Downloads/tennisgame.mp4"
       # SAMPLE_DURATION is maximum deque size
self.SAMPLE_DURATION = 16
self.SAMPLE_SIZE = 112
# Initialise instance of Class Parameter
param = Parameters()
# A Double ended queue to store our frames captured and with time old frames will pop out of the deque
captures = deque(maxlen=param.SAMPLE_DURATION)
# load the human activity recognition model
print("[INFO] loading human activity recognition model...")
net = cv2.dnn.readNet(model=param.ACTION_RESNET)
print("[INFO] accessing video stream...")
# Take video file as input if given else turn on web-cam So, the input should be mp4 file or live web-cam video
vs = cv2.VideoCapture(param.VIDEO_PATH if param.VIDEO_PATH else 0)
while True:
   # Loop over and read capture from the given video input
   (grabbed, capture) = vs.read()
   # break when no frame is grabbed (or end if the video)
if not grabbed:
print("[INFO] no capture read from stream - exiting")
```

```
break
    # resize frame and append it to our deque
capture = cv2.resize(capture, dsize=(550, 400))
captures.append(capture)
    # Process further only when the deque is filled
iflen(captures) <param.SAMPLE_DURATION:
continue
    #captures array is filled we can construct our image blobWe will use SAMPLE_SIZE as height and width for modifying the captured frame
imageBlob = cv2.dnn.blobFromImages(captures, 1.0,
                                   (param.SAMPLE_SIZE,  param.SAMPLE_SIZE),
                                   (114.7748, 107.7354, 99.4750), swapRB=True, crop=True)
    # Manipulate the image blob to make it fit as as inputfor the pre-trained OpenCV's Human Action Recognition Model
imageBlob = np.transpose(imageBlob, (1, 0, 2, 3))
imageBlob = np.expand_dims(imageBlob, axis=0)
    # Forward pass through model to make prediction
net.setInput(imageBlob)
outputs = net.forward()
    # Index the maximum probability
label = param.CLASSES[np.argmax(outputs)]
    # Show the predicted activity
cv2.rectangle(capture, (0, 0), (300, 40), (255, 255, 255), -1)
cv2.putText(capture, label, (10, 25), cv2.FONT_HERSHEY_SIMPLEX,  0.8, (0, 0, 0), 2)
    # Display it on the screen
cv2.imshow("Human Activity Recognition", capture)
key = cv2.waitKey(1) & 0xFF
    # Press key 'q' to break the loop
if key == ord("q"):
break
```

OUTPUT:


Human Activity Recognition
playing tennis
MURR 6 5 30
LLOD 4 2 15


Human Activity Recognition
giving or receiving award

List of Augmented Experiments

Experiment-1

AIM:To implement optical character recognition from an image

```
pip install pytesseract
import cv2
importpytesseract
importnumpy as np
importmatplotlib.pyplot as plt
 #Loading image using OpenCV
img = cv2.imread('C:/Users/CPNM LAB/Desktop/dataimg.jpg')
plt.imshow(img)
 #Preprocessing image
#Converting to grayscale
gray_image = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
 #creating Binary image by selecting proper threshold
binary_image = cv2.threshold(gray_image ,130,255,cv2.THRESH_BINARY +
v2.THRESH_OTSU)[1]
 #Inverting the image
inverted_bin = cv2.bitwise_not(binary_image)
#Some noise reduction
kernel = np.ones((2,2),np.uint8)
processed_img = cv2.erode(inverted_bin, kernel, iterations = 1)
processed_img = cv2.dilate(processed_img, kernel, iterations = 1)
pytesseract.pytesseract.tesseract_cmd = r'C:\Program Files\Tesseract-OCR\tesseract.exe'
#Applyingimage_to_string method
text = pytesseract.image_to_string(processed_img)
print(text)
```

OUTPUT:

```
WHERE CAN I BUY |
ARTIFICIAL GRASS?
```

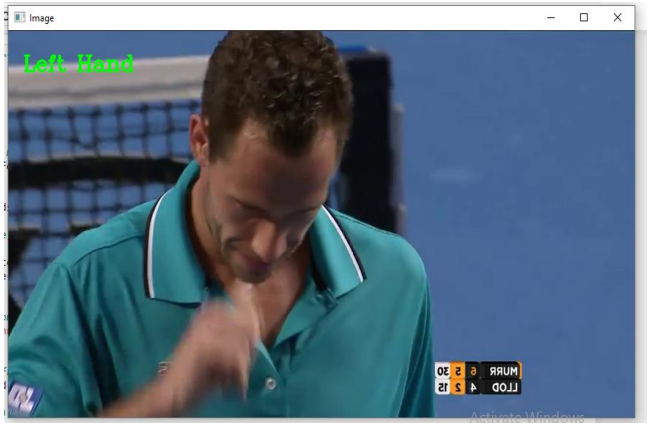
Experiment-2

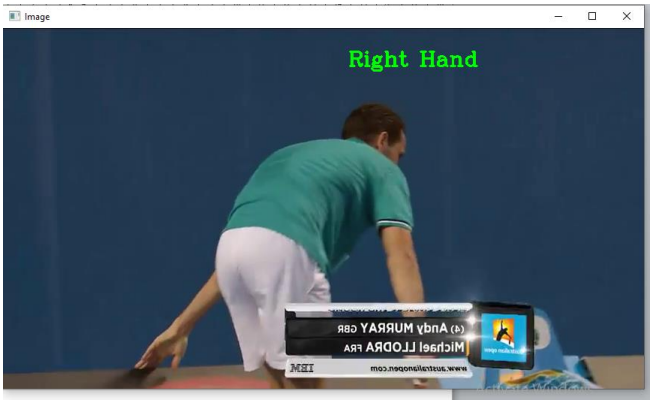
AIM: To implement gesture recognition

```
#Installing required libraries
#pip installmediapipe
#pip installopencv-python
# Importing Libraries
import cv2
importmediapipe as mp
# Used to convert protobuf message to a dictionary.
fromgoogle.protobuf.json_format import MessageToDict
# Initializing the Model
mpHands = mp.solutions.hands
hands = mpHands.Hands(static_image_mode=False,model_complexity=1,
                      min_detection_confidence=0.75,min_tracking_confidence=0.75,
                      max_num_hands=2)
# Start capturing video from webcam
cap = cv2.VideoCapture("tennisgame.mp4")
while True:     # Read video frame by frame
        success, img = cap.read()
        # Flip the image(frame)
        img = cv2.flip(img, 1)
        # Convert BGR image to RGB image
        imgRGB = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
        # Process the RGB image
        results = hands.process(imgRGB)
        # If hands are present in image(frame)
        ifresults.multi_hand_landmarks:
                # Both Hands are present in image(frame)
                iflen(results.multi_handedness) == 2:
                                # Display 'Both Hands' on the image
                        cv2.putText(img, 'Both Hands', (250,
                        50),cv2.FONT_HERSHEY_COMPLEX,0.9, (0, 255, 0), 2)
                # If any hand present
                else:
```

```
            for i in results.multi_handedness:
                    # Return whether it is Right or Left Hand
                    label = MessageToDict(i) ['classification'][0]['label']
                    if label == 'Left':
                            # Display 'Left Hand' on
                            # left side of window
                            cv2.putText(img, label+' Hand',(20, 50),
                            cv2.FONT_HERSHEY_COMPLEX,0.9, (0, 255, 0), 2)
                    if label == 'Right':
                            # Display 'Left Hand'
                            # on left side of window
                            cv2.putText(img, label+' Hand', (460, 50),
                            cv2.FONT_HERSHEY_COMPLEX,
                                                    0.9, (0, 255, 0), 2)
    # Display Video and when 'q'
    # is entered, destroy the window
    cv2.imshow('Image', img)
    if cv2.waitKey(1) & 0xff == ord('q'):
            break
```

Output:





CONTENT BEYOND SYLLABUS

Experiment-1

AIM: Write a program to annotate the image with text

To annotate image with text we use the putText() function in OpenCV.

Syntax: putText(image, text, org, font, fontScale, color)

- The first argument is the input image.
- The second argument is the actual text string that we want to annotate the image with.
- The third argument specifies the starting location for the top left corner of the text string.
- The next two arguments specify the font style and scale.
    - OpenCV supports several font-face styles from the Hershey font collection, and an italic font as well. Check out this list:
        - FONT_HERSHEY_SIMPLEX = 0,
        - FONT_HERSHEY_PLAIN = 1,
        - FONT_HERSHEY_DUPLEX = 2,
        - FONT_HERSHEY_COMPLEX = 3,
        - FONT_HERSHEY_TRIPLEX = 4,
        - FONT_HERSHEY_COMPLEX_SMALL = 5,
        - FONT_HERSHEY_SCRIPT_SIMPLEX = 6,
        - FONT_HERSHEY_SCRIPT_COMPLEX = 7,
        - FONT_ITALIC = 16

- The font scale is a floating-point value, used to scale the base size of the font up or down. Depending on the resolution of your image, select an appropriate font scale.

- The last required argument is the color, which is specified as a BGR triplet.

Program to annotate images with text:

```
# Import dependencies
import cv2
# Read Images
img = cv2.imread('C:/Users/CPNM LAB/Desktop/monkey.jpg')
# Display Image
cv2.imshow('Original Image',img)
```

```
cv2.waitKey(0)
# Print error message if image is null
ifimg is None:
print('Could not read image')
# make a copy of the original image
imageText = img.copy()
#let's write the text you want to put on the image
text = 'I am a Happy Monkey!'
#org: Where you want to put the text
org = (0,100)
# write the text on the input image
cv2.putText(imageText, text, org, fontFace = cv2.FONT_ITALIC, fontScale = 0.5, color = (250,225,100))
# display the output image with text over it
cv2.imshow("Image Text",imageText)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

OUTPUT:

Experiment-2

AIM: Write a program to count the number of frames in an video and display different color transformations of an video.

```
importnumpy as np
import cv2
cap = cv2.VideoCapture('C:/Users/sircrrengg/Desktop/tennisgame.mp4')
while(True):
   # Capture frame-by-frame
ret, frame = cap.read()
      # Our operations on the frame come here
   #img = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
img = cv2.cvtColor(frame,cv2.COLOR_BGR2GRAY)  # BGR color to gray level
   # Display the resulting image
cv2.imshow('Gray',img)
img = cv2.cvtColor(frame,cv2.COLOR_BGR2RGB) # BGR color to RGB
   # Display the resulting image
cv2.imshow('RGB',img)
img = cv2.cvtColor(frame,cv2.COLOR_BGR2HSV) # BGR color to HSV
cv2.imshow('HSV',img)
img = cv2.cvtColor(frame,cv2.COLOR_BGR2HLS) # BGR color to HlS
cv2.imshow('HLS',img)
if cv2.waitKey(1) & 0xFF == ord('q'):  # press q to quit
break
count=int(cap.get(cv2.CAP_PROP_FRAME_COUNT))
# When everything done, release the capture
cap.release()
print('Total no of frames: ',count)
```

OUTPUT: Total no of frames: 3890