Aim:- To perform Handling Files, Cameras and GUIs

Objective: -

To perform Basic I/O Scripts, Reading/Writing an Image File, Converting Between an Image and raw bytes, Accessing image data with numpy.array,Reading /writing a video file, Capturing camera, Displaying images in a window ,Displaying camera frames in a window

Theory:-

Basic I/O script:-

Most CV applications need to get images as input. Most also produce images as output. An interactive CV application might require a camera as an input source and a window as an output destination, However, other possible sources and destinations include image files, video files, and raw bytes. For example, raw bytes might be transmitted via a network connection, or they might be generated by an algorithm if we incorporate procedural graphics into our application. Let's look at each of these possibilities.

Reading/Writing an Image File:-

1.Using ImageIO : Imageio is a Python library that provides an easy interface to read and write a wide range of image data, including animated images, video, volumetric data, and scientific formats.

```
import imageio as iio

# read an image

img = iio.imread("g4g.png")

 # write it in a new format

iio.imwrite("g4g.jpg", img)
```

2.Using MatplotLib : Matplotlib is an amazing visualization library in Python for 2D plots of arrays. Matplotlib is a multi-platform data visualization library built on NumPy arrays and designed to work with the broader SciPy stack.

```
import matplotlib.image as mpimg

import matplotlib.pyplot as plt

# Read Images

img = mpimg.imread('g4g.png')

# Output Images

plt.imshow(img)
```

Writing an Image:-

Let's discuss how to write/save an image into the file directory, using the imwrite() function. Check out its syntax:

```
imwrite(filename, image).
```

1. The first argument is the filename, which must include the filename extension (for example .png, .jpg etc). OpenCV uses this filename extension to specify the format of the file.
2. The second argument is the image you want to save. The function returns True if the image is saved successfully.

Take a look at the code below. See how simple it is to write images to disk. Just specify the filename with its proper extension (with any desired path prepended). Include the variable name that contains the image data, and you're done.

```
cv2.imwrite('grayscale.jpg',img_grayscale)
```

Converting Between an Image and raw bytes:-

Conceptually, a byte is an integer ranging from 0 to 255. Throughout real-time graphic applications today, a pixel is typically represented by one byte per channel, though other representations are also possible.

An OpenCV image is a 2D or 3D array of the numpy.array type. An 8-bit grayscale image is a 2D array containing byte values. A 24-bit BGR image is a 3D array, which also contains byte values. We may access these values by using an expression such as image[0, 0] or image[0, 0, 0]. The first index is the pixel's *y* coordinate or row, 0 being the top. The second index is the pixel's *x* coordinate or column, 0 being the leftmost. The third index (if applicable) represents a color channel.

Accessing image data with numpy. Array:-

Python provides many modules and API's for converting an image into a NumPy array. Let's discuss to Convert images to NumPy array in Python.

Using NumPy module to Convert images to NumPy array:-

Numpy module in itself provides various methods to do the same. These methods are –

asarray() function is used to convert PIL images into NumPy arrays. This function converts the input to an array

```
# Import the necessary libraries

from PIL import Imagefrom numpy import asarra

# load the image and convert into

# numpy array

img = Image.open('Sample.png')

 # asarray() class is used to convert

# PIL images into NumPy arrays

numpydata = asarray(img)

# <class 'numpy.ndarray'>

print(type(numpydata))

#  shape

print(numpydata.shape)
```

Reading/Writing a video file :-

OpenCV provides the VideoCapture and Video Writer classes that support various video file formats. The supported formats vary by system but should always include an AVI. Via itsread() method, a VideoCapture class may be polled for new frames until it reaches the end of its video file. Each frame is an image in a BGR

format, Conversely, an image may be passed to the write() method of the VideoWriter class, which appends the image to a file in VideoWriter.

Capturing camera frames:-

A stream of camera frames is represented by the VideoCapture class too however, for a camera, we construct a VideoCapture class by passing the camera's device index instead of a video's filename.

Displaying images in a window :-

One of the most basic operations in OpenCV is displaying an image. This can be done with the imshow() function. If you come from any other GUI framework background, you would think it sufficient to call imshow() to display an image. This is only partially true: the image will be displayed, and will disappear immediately.

Displaying camera frames in a window :-

OpenCV allows named windows to be created, redrawn, and destroyed using thenamedWindow(), imshow(), and destroyWindow() functions. Also, any window may capture keyboard input via the waitKey() function and mouse input via the set MouseCallback() function.

Conclusion:-

We have finally grasped how a computer vision application can be developed opencv consist of mat class which is used for pixel manipulation and it has a class relationship to application buffered Image class.