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:

1. Basic I/O script:

The Python basic input-output involves the use of various built-in functions and its associated objects to handle the files. For example, the open(path,mode) function is given a path name of an file relative to the system working directory, and returns an object whose methods allow for the reading of and writing to the given file. The path attribute specifies a string containing the path_name to the file we wish to open.

2. Reading/Writing an Image File:

A. 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.

Syntax:

#Reading Image

imageio.imread(‘Hare_Krishna.jpg’)

#Writing Image

imageio.imwrite(‘Sita_Ram.jpg’)

B. **Using OpenCV :**

OpenCV (Open Source Computer Vision) is a computer vision library that contains various functions to perform operations on pictures or videos.

Syntax:

Reading Image

cv2.imread(‘Vitthal_Rukmini.jpg’)

C. **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.

Syntax:

Reading Image

matplotlib.image.imread(‘Shri_Jagganath.jpg’)

D. **Using PIL:**

PIL is the Python Imaging Library which provides the python interpreter with image editing capabilities.

Syntax:

Reading Image

image.open(‘Shri_Radhe.jpg’)

3. Converting Between an Image and Raw Bytes:

Sometimes, we may want an in-memory jpg or png image that is represented as binary data. In python this can be done is multiple ways as follows:

A. Using OpenCV:

Using Opencv2, encoded image in a one-dimension Numpy array can be converted to real bytes either with the tobytes() method or io.BytesIO().

Syntax:

```
im = cv2.imread('Vaman_Deva.jpg')
im_buf_arr = cv2.imencode(".jpg", im)
byte_im = im_buf_arr.tobytes()
```

B. PIL:

Using PIL it image can be converted into raw bytes as follows:

```
from PIL import Image
im = Image.open('Narsinha_Deva.jpg')
buf = io.BytesIO()
im_resize.save(buf, format='JPEG')
byte_im = buf.getvalue()
```

4. Accessing image data with numpy Array:

In Python, Pillow is the most popular and standard library when it comes to working with image data.NumPy uses the asarray() class to convert PIL images into NumPy arrays. The np.array function also produce the same result. The type function displays the class of an image.The process can be reversed using the Image.fromarray() function. This function comes in handy when the manipulation is performed on numpy.ndarray image data, that we laterwant to save as a PNG or JPEG file.

Syntax:

```
# Converting Image to Numpy Array
data = numpy.asarray(image)
```

5. Reading/Writing a video file:

Reading and writing videos is very similar to reading and writing images. A video is nothing but a series of images that are often referred to as frames. So, all you need to do is loop over all the frames in a video sequence, and then process one frame at a time. In python it can be easily done using OpenCV.

Syntax:

```
vid_capture = cv2.VideoCapture('Mahadev.mp4')
```

6. Capturing camera frames:

Python provides various libraries for image and video processing. With OpenCV, we can capture a video from the camera and can also display camera frames. It lets you create a video capture object which is helpful to capture videos through webcam and then you may perform desired operations on that video.

Syntax:

```
# Define a video capture object
vid = cv2.VideoCapture(0)
while(True):
# Capture the video frame by frame
ret, frame = vid.read()
```

7. Displaying images in a window:

Displaying the require image on window can be easily done by using:

A. Matplotlib

This package is mainly for data visualization. But, through the plotting techniques, we can view the image in a graphical format where each pixel lies on 2D x-y axes.

B. Pillow

This library often offers simple methods for Image manipulations. It uses the functions such as open() and show() from PILLOW's Image module to display image.

C. Scikit-Image

Scikit-Image is a sub-module of Scikit-Learn. It is built upon Python and supportive library Matplotlib thus it derives some of its functionalities. Methods are similar to that of the previous packages we saw before.

8. Displaying camera frames in a window

Python provides various libraries for image and video processing. With OpenCV, we can capture a video from the camera and can also display camera frames.

Syntax:

```
# Define a video capture object
vid = cv2.VideoCapture(0)
while(True):
# Capture the video frame by frame
ret, frame = vid.read()
```

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

Conclusion:

In this experiment, we successfully traversed basic I/O file handling methods in python. We also explored various fundamental libraries required for implementing computer vision which includes operations like 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.