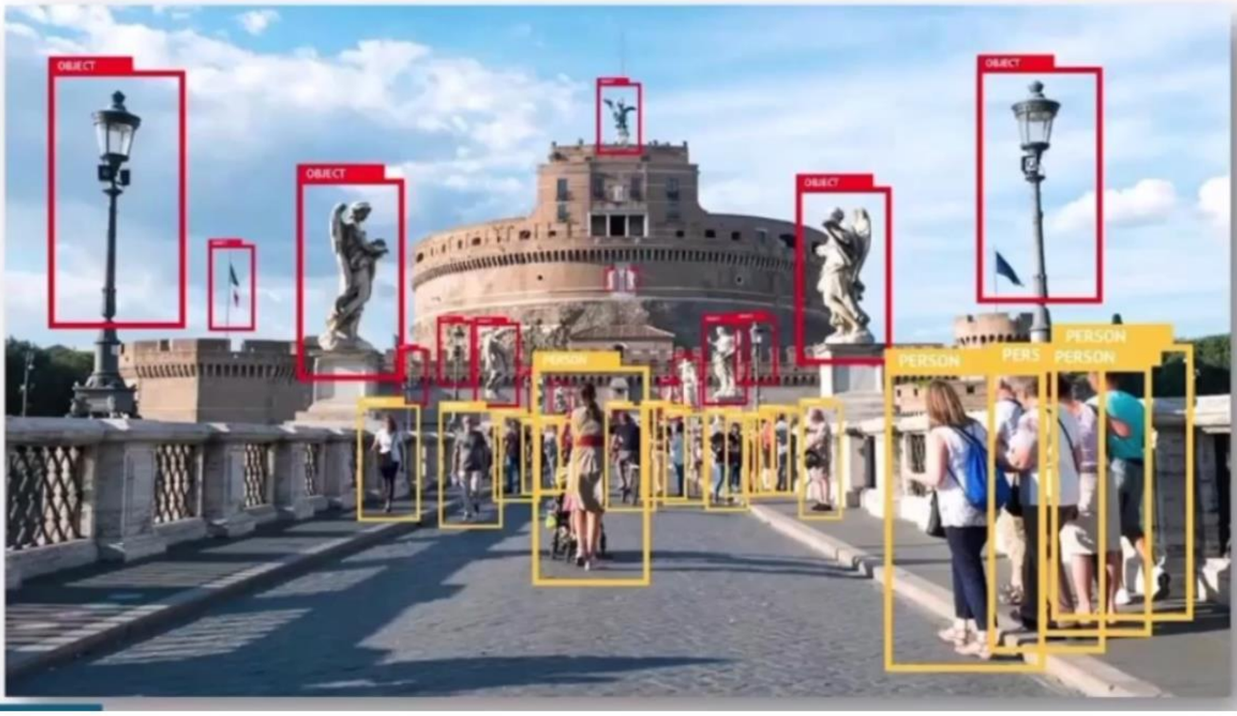
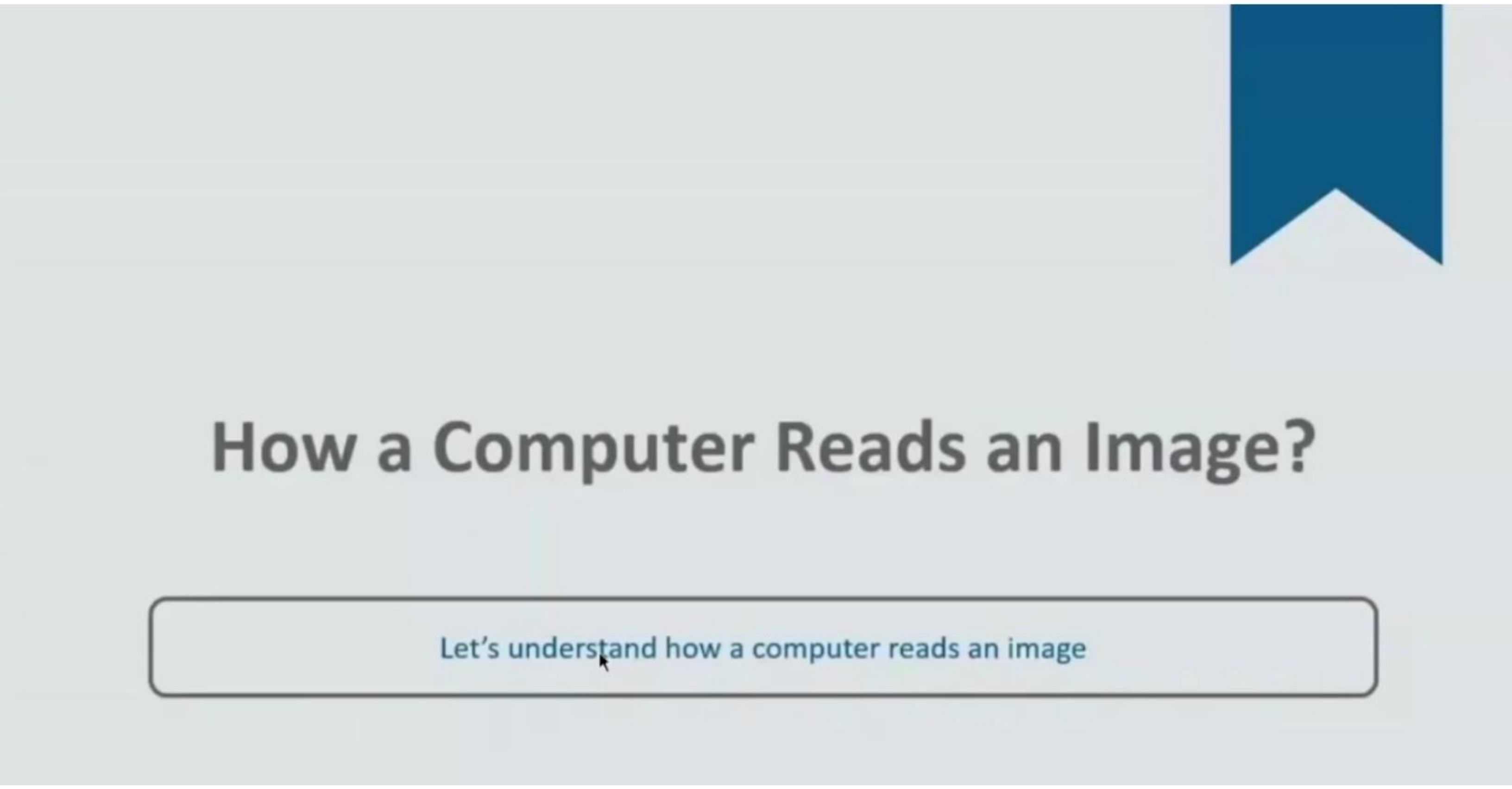
Agenda

- What Is Computer Vision?
- How A Computer Reads An Image
- What Is OpenCV?
- Basics Of OpenCV
- Image Detection Using OpenCV
- Motion Detector Using OpenCV

What Is Computer Vision?

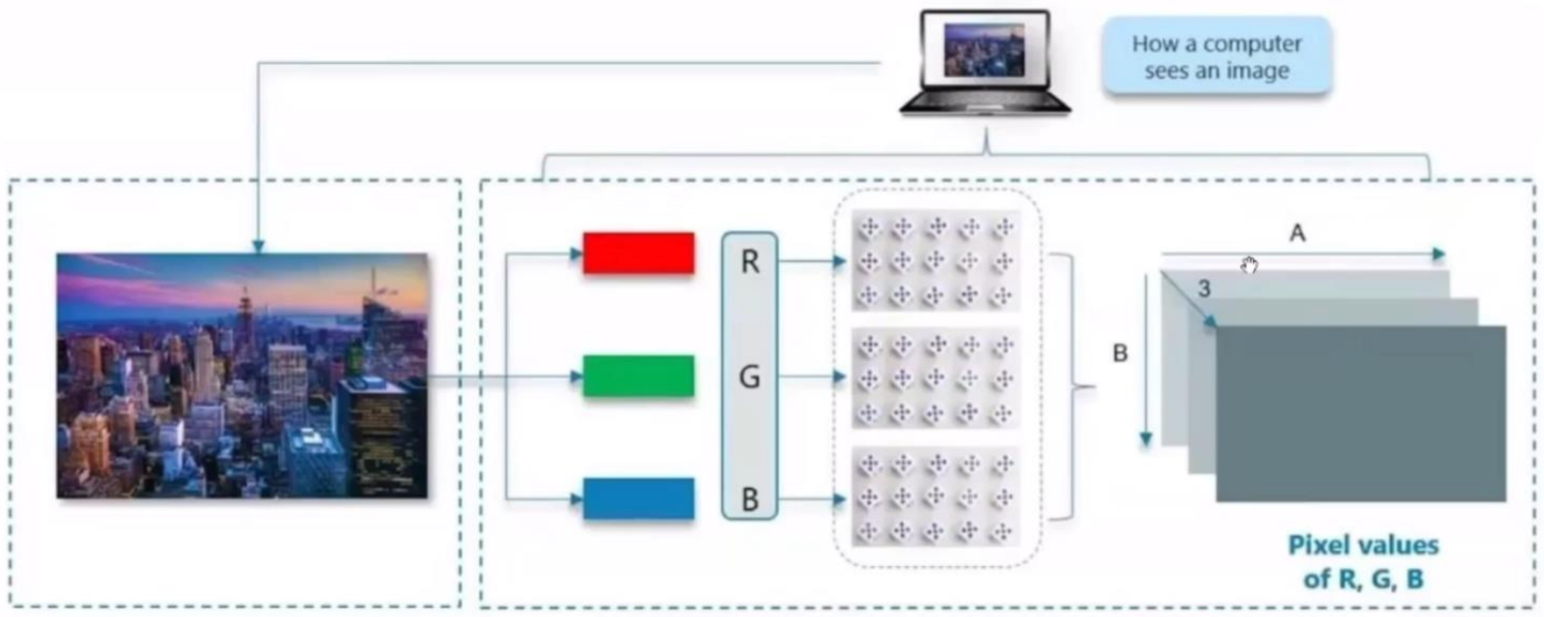




How a Computer Reads an Image?

Let's understand how a computer reads an image

How a Computer Reads an Image



Size of the image will be – B x A x 3

What is OpenCV?

Let's understand what is OpenCV

What is OpenCV?

OpenCV-Python is a library of Python designed to solve computer vision problems

OpenCV

NumPy

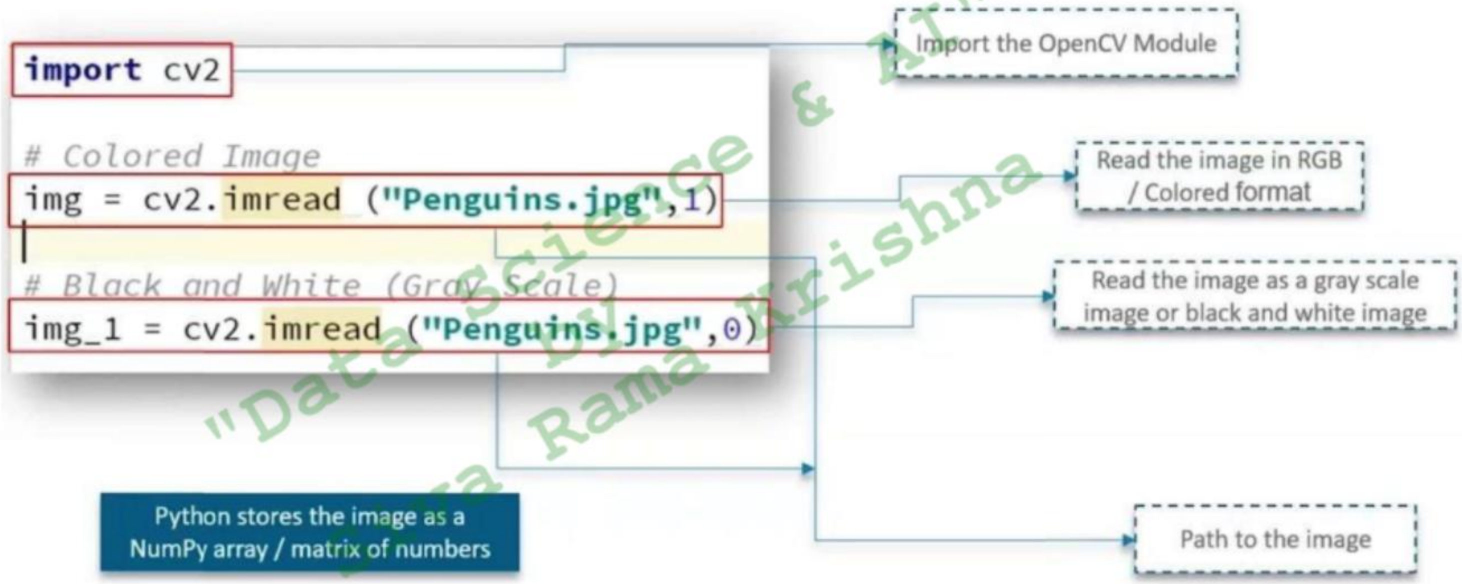
All the OpenCV array structures are converted to and from NumPy arrays

SciPy2015

matplotlib

This makes it easier to integrate it with other libraries that uses NumPy

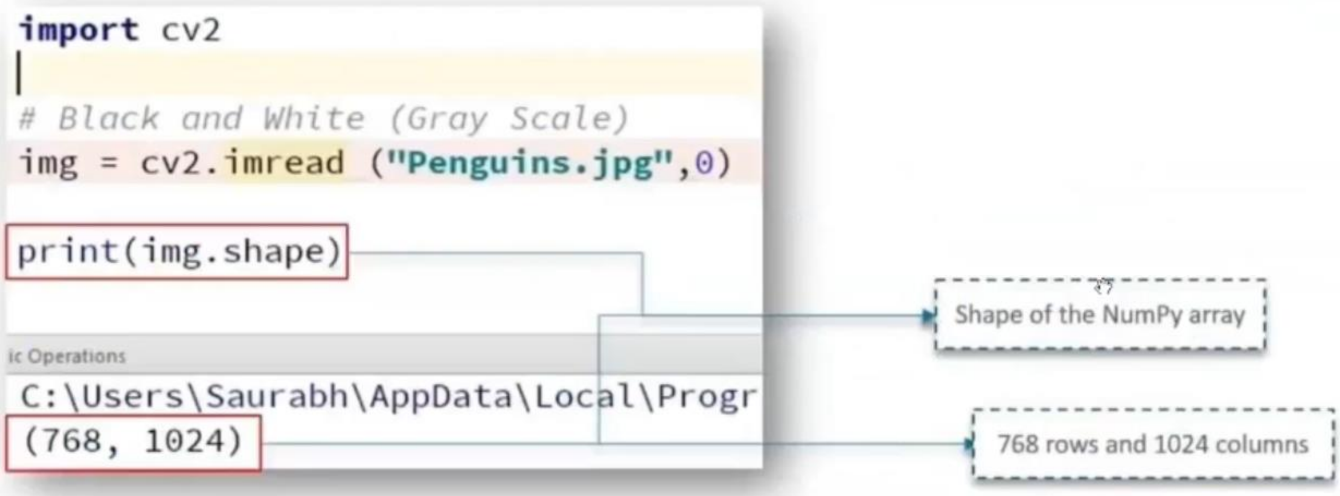
Load Images Using OpenCV



```
import cv2

# Colored Image
img = cv2.imread ("Penguins.jpg",1)

# Black and White (Gray Scale)
img_1 = cv2.imread ("Penguins.jpg",0)
```

Import the OpenCV Module
Read the image in RGB / Colored format
Read the image as a gray scale image or black and white image
Path to the image
Python stores the image as a NumPy array / matrix of numbers

Image Shape / Resolution



```
import cv2

# Black and White (Gray Scale)
img = cv2.imread ("Penguins.jpg",0)

print(img.shape)
```

```
C:\Users\Saurabh\AppData\Local\Progr
(768, 1024)
```

Displaying The Image

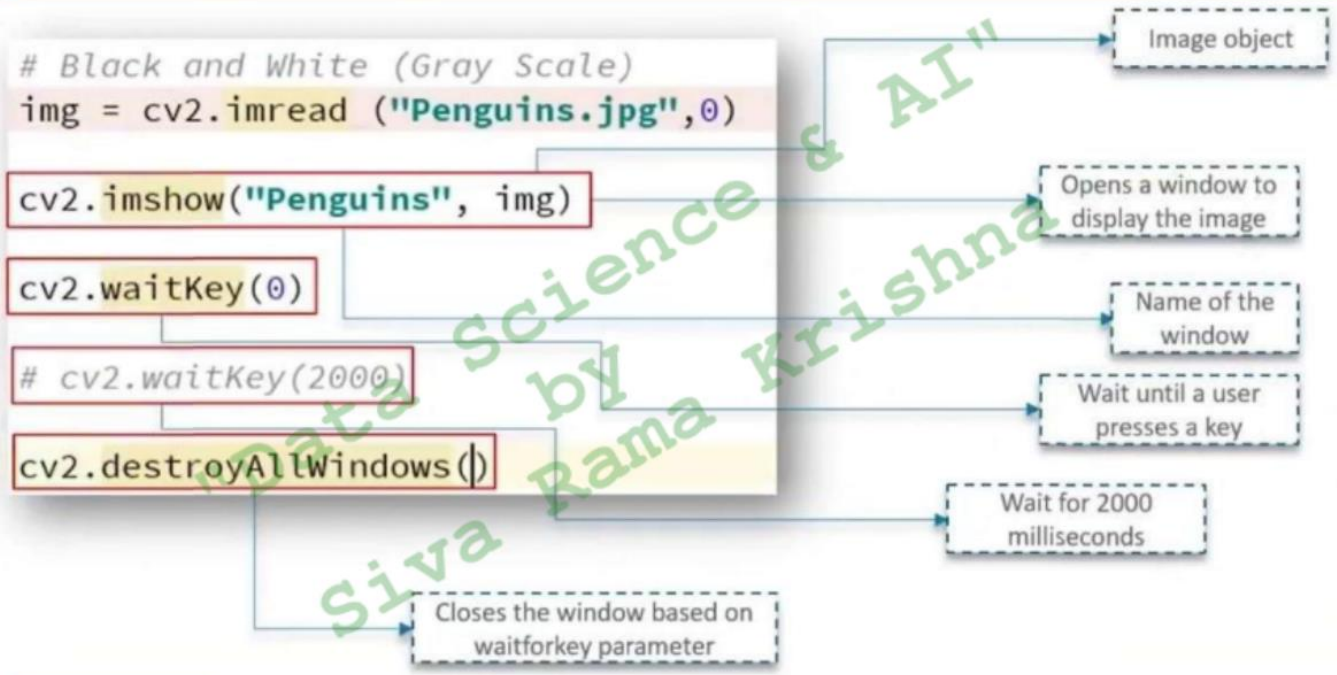
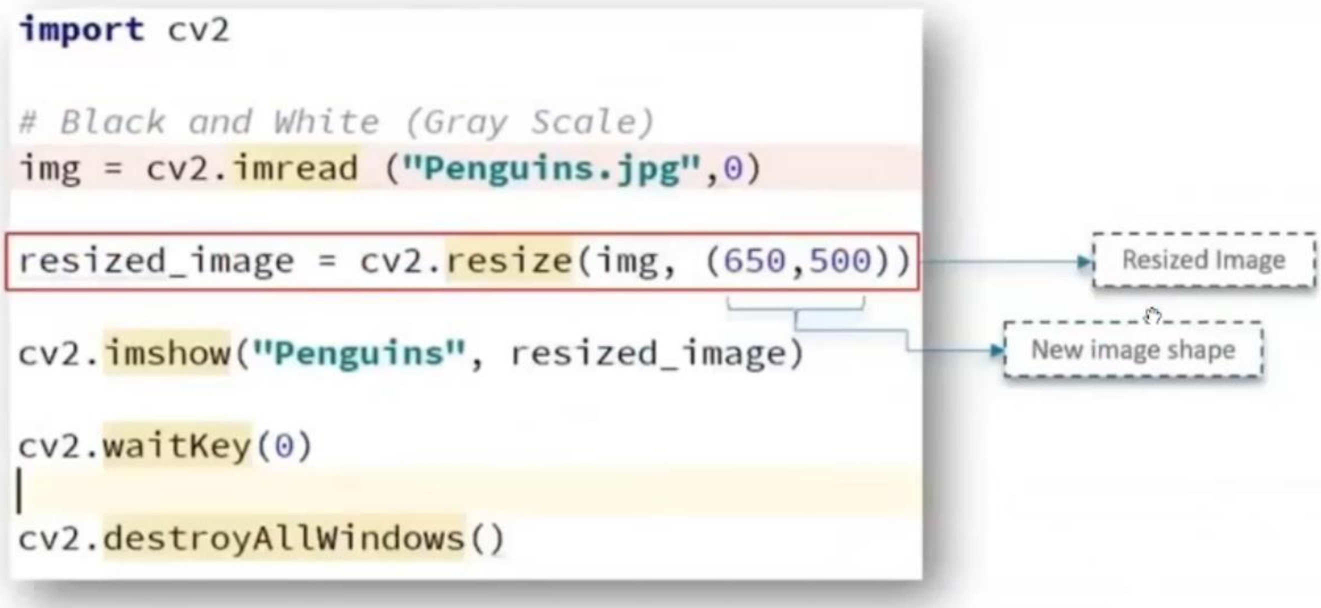


Image object
Opens a window to display the image
Name of the window
Wait until a user presses a key
Wait for 2000 milliseconds
Closes the window based on waitforkey parameter

```
# Black and White (Gray Scale)
img = cv2.imread ("Penguins.jpg",0)

cv2.imshow("Penguins", img)

cv2.waitKey(0)

# cv2.waitKey(2000)

cv2.destroyAllWindows()
```

Resizing The Image



```
import cv2

# Black and White (Gray Scale)
img = cv2.imread ("Penguins.jpg",0)

resized_image = cv2.resize(img, (650,500))

cv2.imshow("Penguins", resized_image)

cv2.waitKey(0)

cv2.destroyAllWindows()
```

Resizing The Image

```
resized_image = cv2.resize(img, (int(img.shape[1]/2),int(img.shape[0]/2)))
```

New image shape = Old image shape/2

Face Detection

Let’s see how to detect faces with OpenCV

Face Detection Using OpenCV

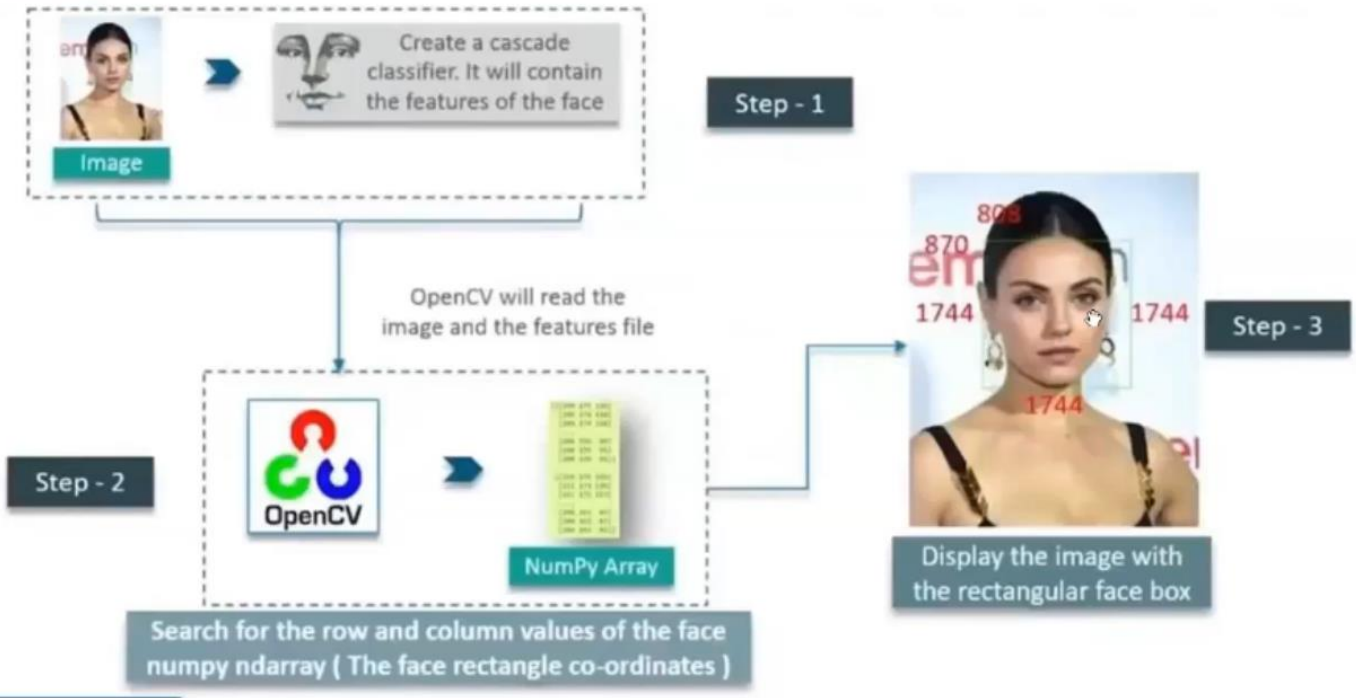
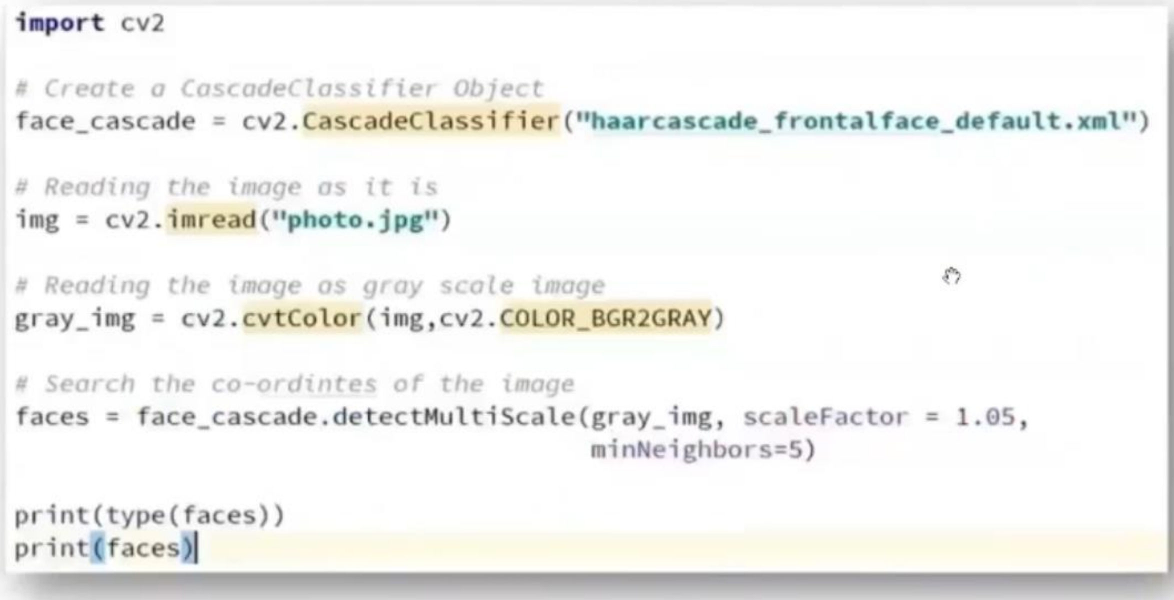


Image
Create a cascade classifier. It will contain the features of the face
Step - 1
OpenCV will read the image and the features file
Step - 2
OpenCV
NumPy Array
Search for the row and column values of the face numpy ndarray (The face rectangle co-ordinates)
808
870
1744
1744
1744
Step - 3
Display the image with the rectangular face box

Face Detection Using OpenCV



```
import cv2

# Create a CascadeClassifier Object
face_cascade = cv2.CascadeClassifier("haarcascade_frontalface_default.xml")

# Reading the image as it is
img = cv2.imread("photo.jpg")

# Reading the image as gray scale image
gray_img = cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)

# Search the co-ordintes of the image
faces = face_cascade.detectMultiScale(gray_img, scaleFactor = 1.05,
                                      minNeighbors=5)

print(type(faces))
print(faces)
```

Face Detection Using OpenCV

```python
# Create a CascadeClassifier Object
face_cascade = cv2.CascadeClassifier("haarcascade_frontalface_default.xml")
```

Create a CascadeClassifier Object

Path to the xml file which contains the face features

```python
# Reading the image as gray scale image
gray_img = cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
```

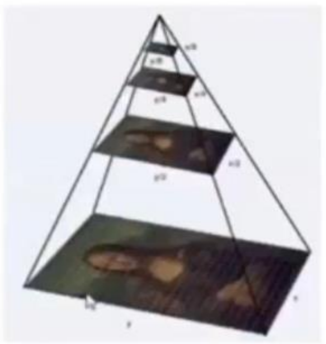
Converting colored image to gray scale

Face Detection Using OpenCV

```python
# Search the co-ordintes of the image
faces = face_cascade.detectMultiScale(gray_img, scaleFactor = 1.05,
                                      minNeighbors=5)
```

Method to search for the face rectangle co-ordinates

Decreases the shape value by 5%, until the face is found.
Smaller this value, the greater is the accuracy



Face Detection Using OpenCV - Output

```
print(type(faces))
print(faces)
```

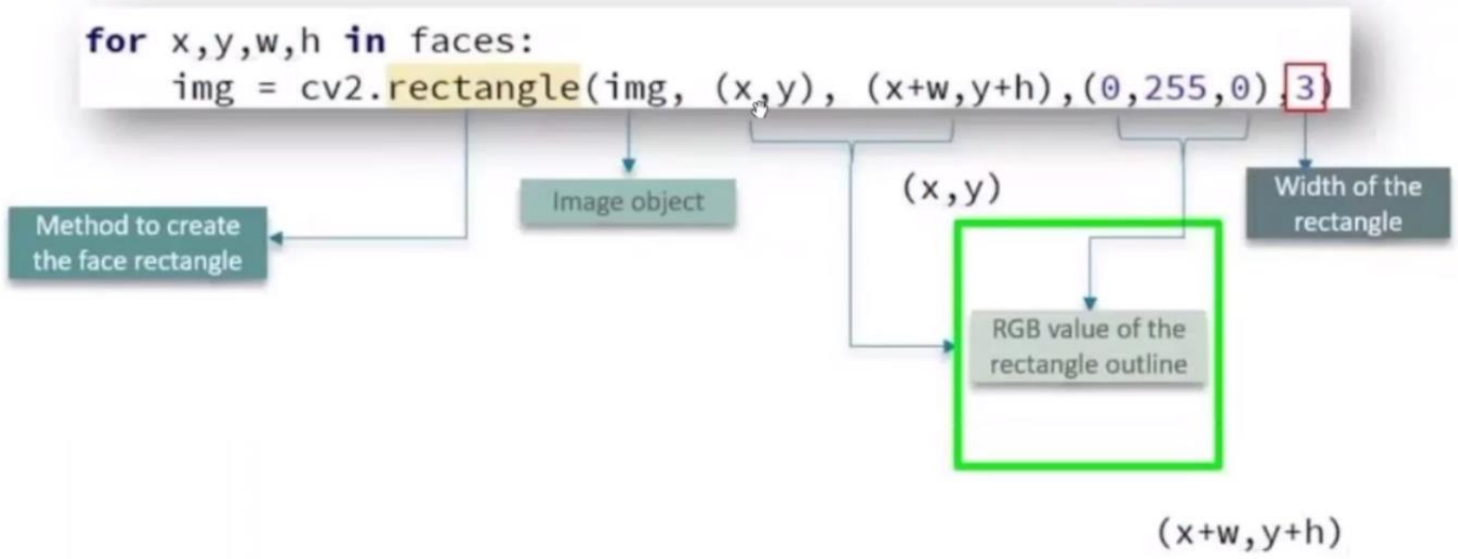
Output

```
<class 'numpy.ndarray'>
[[ 870  808 1744 1744]]
```

Adding The Rectangular Face Box

Let’s add the rectangular face box

Face Detection Using OpenCV



```
for x,y,w,h in faces:
    img = cv2.rectangle(img, (x,y), (x+w,y+h),(0,255,0),3)
```

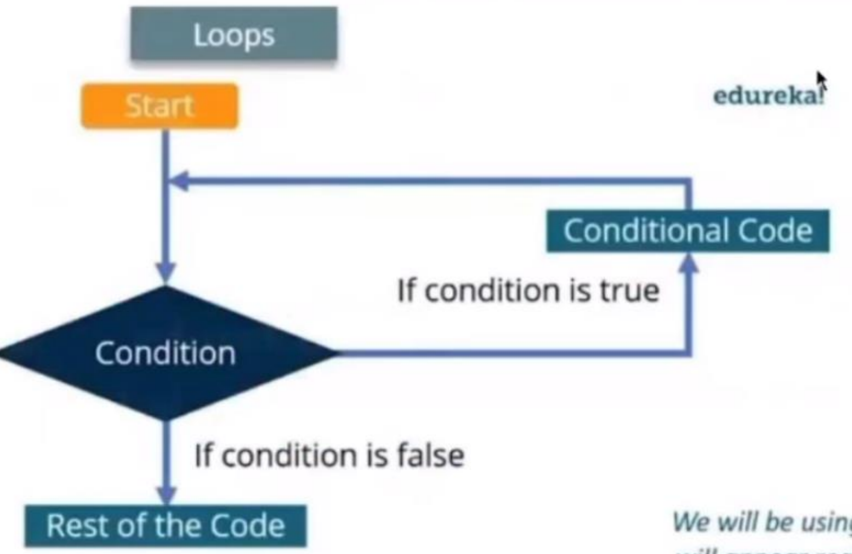
Method to create the face rectangle
Image object
(x,y)
Width of the rectangle
RGB value of the rectangle outline
(x+w,y+h)

Capturing Video

Let’s see how to use OpenCV to capture video with computer web cam

Capturing Video

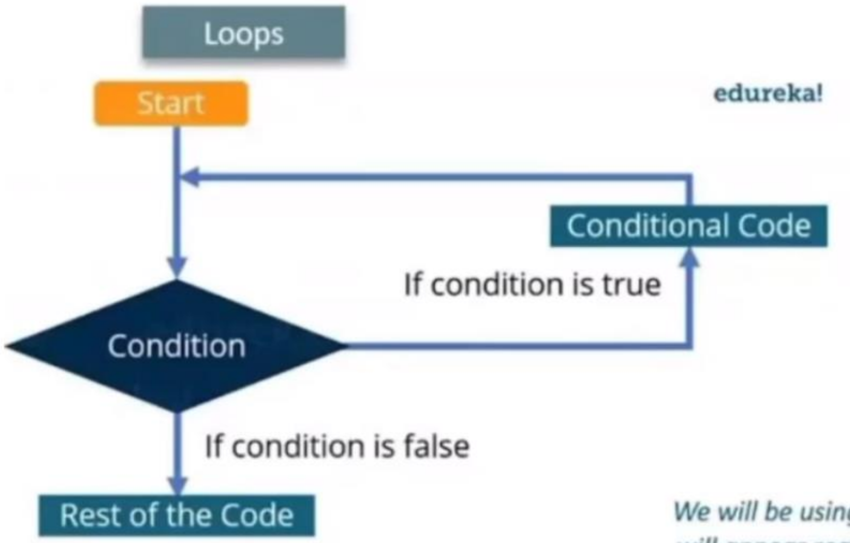
We will be using OpenCV for reading frames/images one-by-one.



We will be using loops to build a window where images will appear really fast, so that you can see it as a video

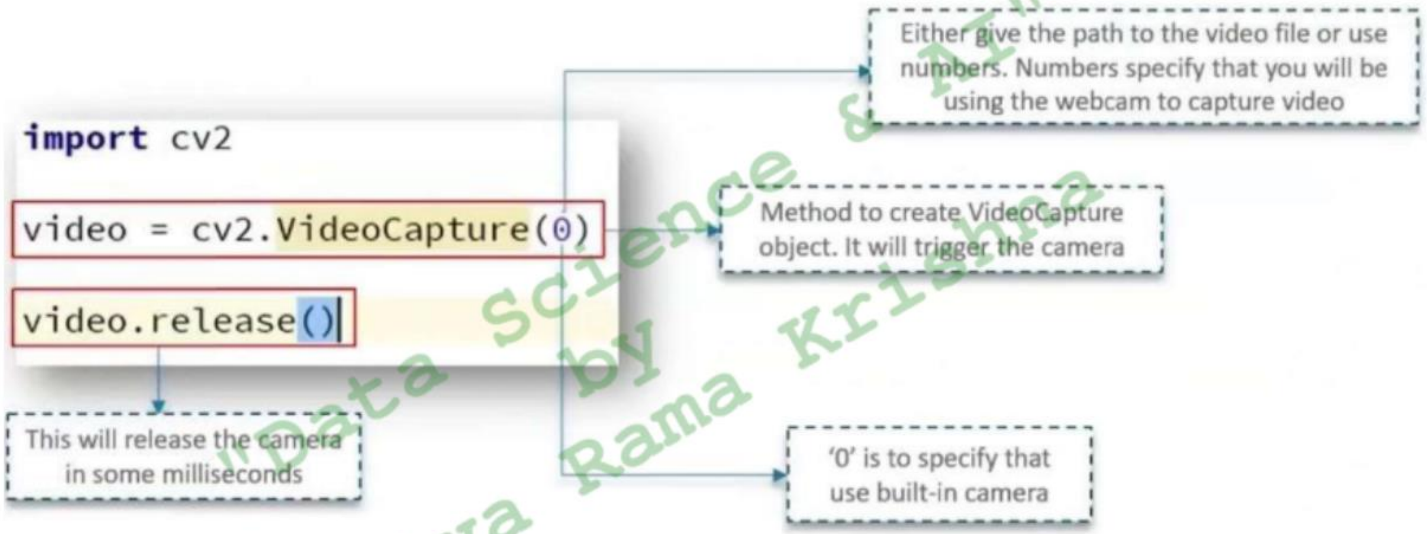
Capturing Video

We will be using OpenCV for reading frames/images one-by-one.



We will be using loops to build a window where images will appear really fast, so that you can see it as a video

Capturing Video



```
import cv2

video = cv2.VideoCapture(0)

video.release()
```

Either give the path to the video file or use numbers. Numbers specify that you will be using the webcam to capture video

Method to create VideoCapture object. It will trigger the camera

This will release the camera in some milliseconds

'0' is to specify that use built-in camera

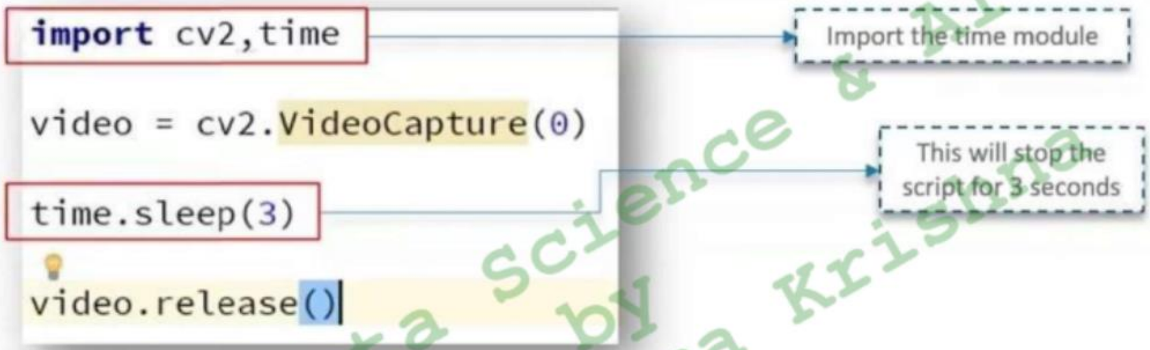
Note: If I have an external cam, I can put '1' to use that. Similarly, if I have two external cams and I want to use the third cam I can put '3'.

Capturing Video

When you execute the code, you will notice that your cam light switches on for split seconds, and then it turns off

Let's go ahead and add time delay, using Time module

Capturing Video



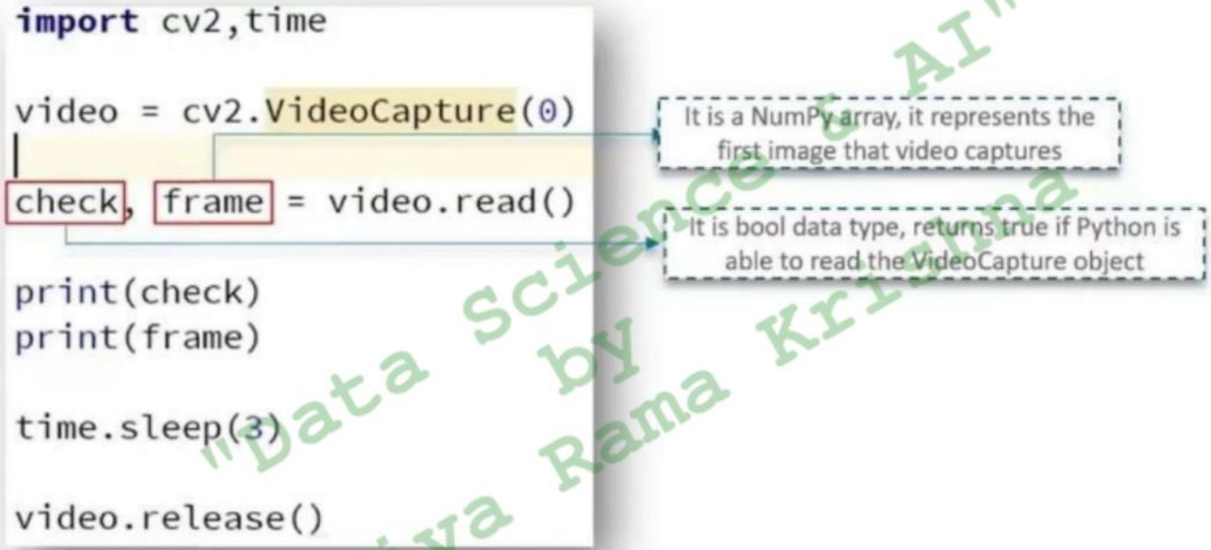
```python
import cv2,time

video = cv2.VideoCapture(0)

time.sleep(3)

video.release()
```

Import the time module

This will stop the script for 3 seconds

When you execute the above code, the web cam will be on for 3 seconds

Capturing Video



```python
import cv2,time

video = cv2.VideoCapture(0)

check, frame = video.read()

print(check)
print(frame)

time.sleep(3)

video.release()
```

It is a NumPy array, it represents the first image that video captures

It is bool data type, returns true if Python is able to read the VideoCapture object

Capturing Video

In order to capture the video, we will be using 'while' loop. While condition will be such that, until unless 'check' is True, Python will display the frames.

```
import cv2,time

video = cv2.VideoCapture(0)

a = 1

while True:
    a = a + 1
    check, frame = video.read()
    print (frame)

    gray = cv2.cvtColor(frame,cv2.COLOR_BGR2GRAY)

    cv2.imshow('Capturing',gray)

    key = cv2.waitKey(1)

    if key == ord('q'):
        break

print(a) # This will print the number of frames
video.release()
cv2.destroyAllWindows
```

Convert each frame into a gray scale image

This will iterate through the frames and display the window

This will generate a new frame after every 1 miliseconds

Once you enter 'q' the window will be destroyed