Computer Vision

Computer vision is a discipline that studies how to reconstruct, interrupt and understand a 3d scene from its 2d images, in terms of the properties of the structure present in the scene.

Computer Vision Vs Image Processing

Image processing studies image to image transformation. The input and output of image processing are both images.

Computer vision is the construction of explicit, meaningful descriptions of physical objects from their image. The output of computer vision is a description or an interpretation of structures in 3D scene.

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.

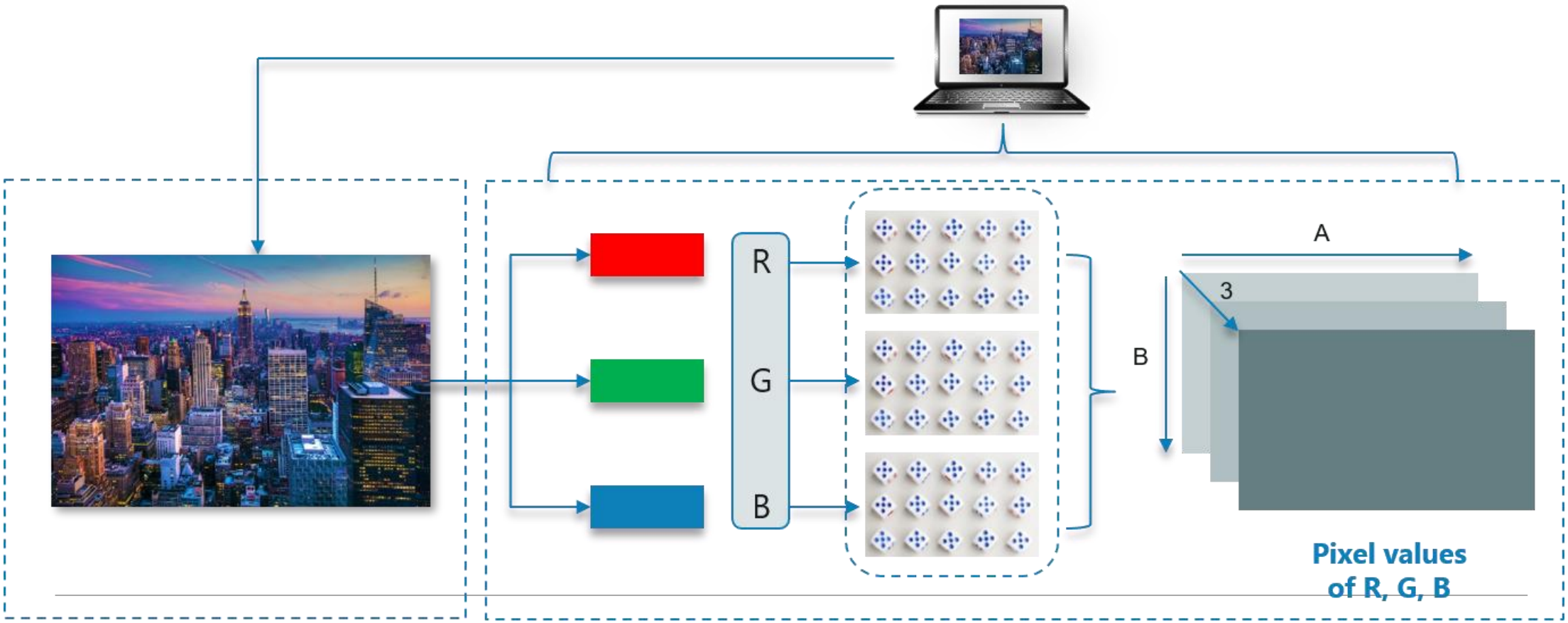
The idea here is to automate tasks that the human visual systems can do. So, a computer should be able to recognize objects such as that of a face of a human being or a lamppost or even a statue.

Features

1. OpenCV supports a wide variety of programming languages such as C++, Python, Java etc. Support for multiple platforms including Windows, Linux, and MacOS.
2. 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.
3. This makes it easier to integrate it with other libraries which use NumPy. For example, libraries such as SciPy and Matplotlib.

How computer read an Image

- The computer reads any image as a range of values between 0 and 255.
- For any colour image, there are 3 primary channels – Red, green and blue. How it works is pretty simple.
- A matrix is formed for every primary colour and later these matrices combine to provide a Pixel value for the individual R, G, B colours.



- As shown, the size of the image here can be calculated as B x A x 3.
- Note: For a black-white image, there is only one single channel.
- Next up on this OpenCV Python Tutorial blog, let us look at what OpenCV actually is.

- ❑ **Loading an image using OpenCV**
- ❑ **Image Shape/Resolution**
- ❑ **Displaying the image**
- ❑ **Resizing the image**

Face Detection Using OpenCV

This seems complex at first but it is very easy. Let me walk you through the entire process and you will feel the same.

Step 1: Considering our prerequisites, we will require an image, to begin with. Later we need to create a cascade classifier which will eventually give us the features of the face.
Step 2: This step involves making use of OpenCV which will read the image and the features file. So at this point, there are NumPy arrays at the primary data points.
All we need to do is to search for the row and column values of the face NumPy ndarray. This is the array with the face rectangle coordinates.
Step 3: This final step involves displaying the image with the rectangular face box.

Image

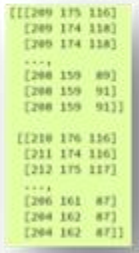
Create a cascade classifier. It will contain the features of the face

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)

Step - 3

Display the image with the rectangular face box