**OpenCV**

OpenCV is the huge open-source library for computer vision, machine learning, and image processing and now it plays a major role in real-time operation which is very important in today’s systems. By using it, one can process images and videos to identify objects, faces, or even the handwriting of a human. This article focuses on detecting object.

We will detect color in HSV format means Hue Saturation Value. Hue is color itself and Saturation is a color fillness of color and Value is how light or dark the color is . The HSV color space represents images in terms of Hue, Saturation, and Value (brightness). This representation makes it easier to manipulate specific aspects of an image, such as adjusting its brightness without affecting its color information.

Why do we need HSV?

R, G, B in RGB are all co-related to the color luminance( what we loosely call intensity),i.e., We cannot separate color information from luminance. HSV or Hue Saturation Value is used to separate image luminance from color information. This makes it easier when we are working on or need luminance of the image/frame.1

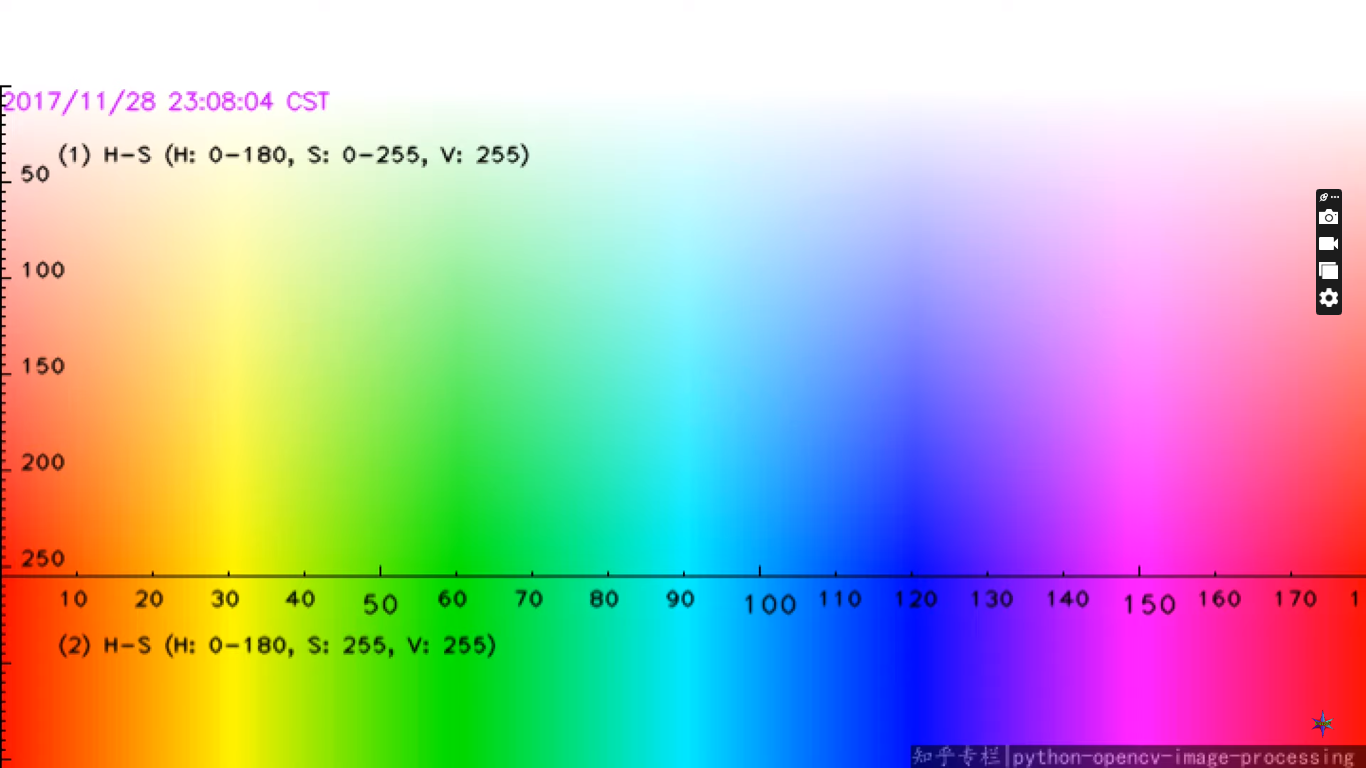
In OpenCV the

Hue ranges between {0-180}

Saturation ranges between {0-255}

Value is ranges between {0-255}

OpenCV Python Color Detection Example



The x axis represent the hue

The y axis represent the saturation

1.Visual Studio

2.Install opencv and numpy ,matplotlibs

pip install opencv-python

pip install numpy

3.import opencv and numpy ,matplotlibs module

4.read the image with help of imread function

5.Then convert the image from VTR format to RGB format using cv2 convert color function because by default opencv reads the image as a BG color image format so we converted to RGB format in order to look at the way humans can see it properly in proper color

5.Then display the image with help of imshow function

6.Then convert the RGB image into HSB image using cv2 color function that because opencv uses HSV format image in order to detect the color .

7.Then mention the lower and upper limits of the image so in this, the lower limit is for color orange and the upper limit is also for color orange so initially we are detecting only one color after we will move on to detecting more than one color.

8.The bitwise and operator it will remove the mask and black all the unnecessary unrequired colors and it will display only the color that will belong to particular axis .

9.Then detect the more than one color

Below are a few more examples of colors in RGB:

| **Color** | **RGB value** |
| --- | --- |
| Red | 255, 0, 0 |
| Orange | 255, 128, 0 |
| Pink | 255, 153, 255 |

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26 | import cv2  import numpy as np            img = cv2.imread('img.png')    hsv = cv2.cvtColor(img, cv2.COLOR\_BGR2HSV)      #Red color rangle  169, 100, 100 , 189, 255, 255      lower\_range = np.array([110,50,50])  upper\_range = np.array([130,255,255])    mask = cv2.inRange(hsv, lower\_range, upper\_range)    cv2.imshow('image', img)  cv2.imshow('mask', mask)      cv2.waitKey(0)  cv2.destroyAllWindows() |