Project Report
On
“Color Detection”
By
Priyanshu Mangal

Overview of Project

The main objective of project is to detect the colors in an image. The program will also return as the RGB values of the colors, which is really helpful. Many graphic designers and web designers will understand how RGB values can be helpful. Building a color recognizer is a great project to get started with Computer Vision.

Table of Contents

Getting Started

We will use three main modules for this project. They are NumPy, Pandas and OpenCv.

Libraries

As mention earlier, there are three modules I will use for this project. To use these modules, we have to install the necessary libraries. Library installation is a very easy step using pip. Pip is a package management tool. We will do the installation using the command-line interface. Here is the line to install all 3 libraries at once:

```
pip install numpy pandas opencv-python
```

After the installation is completed, we have to import them to our program. Open a new file in any code editor. Here is the code on how to import the installed libraries:

```
import numpy as np
import pandas as pd
import cv2
```

OpenCv is imported as cv2. And for other libraries, we imported them "as" so that it is easier to call them in the program.

Define Image

I will save my image in the same folder as my program, which makes it easier to find and import.

```
img = cv2.imread("color_image.jpg")
```

Color Recognition

Teaching the Colors

First, we have to teach them the colors. To do that we need data that includes color names and some values to match with those colors. Since most of the colors can be defined using Red, Green, and Blue. That's why we will use the RGB format as our data points. I found a ready csv file with around 1000 color names and the RGB values. We will use this csv file in our program. The screenshot of the file is mentioned below:

alice_blue	Alice Blue	#f0f8ff	240	248	255
alizarin_crimson	Alizarin Crimson	#e32636	227	38	54
alloy_orange	Alloy Orange	#c46210	196	98	16
almond	Almond	#efdecd	239	222	205
amaranth	Amaranth	#e52b50	229	43	80
amber	Amber	#ffbf00	255	191	0

colors.csv

Let's import *colors.csv* file to program using *read_csv* method. Since the csv file we downloaded doesn't have column names, I will be defining them in the program. This process is known as data manipulation.

```

```

```
Index =["color", "color_name", "hex", "R", "G", "B"]csv = pd.
read_csv('colors.csv', names=index, header=None)
```

Global Variables

In the following steps, we will define two functions. To make the application work smoothly, we need some global variables. We will know how global variables can be helpful when working with functions.

```
clicked = False
r = g = b = xpos = ypos = 0
```

Color Recognition Function

This function will be called when we double click on an area of the image. It will return the color name and the RGB values of that color. This is where the magic happens!

```
def recognize_color(R,G,B):
    minimum = 10000
    for i in range(len(csv)):
        d = abs(R- int(csv.loc[i,"R"])) + abs(G-
int(csv.loc[i,"G"]))+ abs(B- int(csv.loc[i,"B"]))
        if(d<=minimum):
            minimum = d
            cname = csv.loc[i,"color_name"]
    return cname
```

Mouse Click Function

This function is used to define our double click process. We will need it when creating our application part.

```
def mouse_click(event, x, y, flags, param):
    if event == cv2.EVENT_LBUTTONDBLCLK:
        global b,g,r,xpos,ypos, clicked
        clicked = True
        xpos = x
        ypos = y
        b,g,r = img[y,x]
        b = int(b)
        g = int(g)
        r = int(r)
```

Application

In this step, we will open the image as a new window using OpenCV methods. And in that window, we will use the functions we defined earlier. The application is so simple, it returns the color name and color values when you double click on a certain area on the image.

Application Window

Firstly, open the image file as a new window using OpenCV.

```
cv2.namedWindow('Color Recognition App')
```

Secondly, let's call the mouse click function that we created. This gives more functionality to our application.

```
cv2.setMouseCallback('Color Recognition App', mouse_click)
```

The Application

Here is the while loop to start our application window working:

```
while(1):cv2.imshow("Color Recognition App",img)
    if (clicked):

        #cv2.rectangle(image, startpoint, endpoint, color,
thickness)-1 fills entire rectangle
        cv2.rectangle(img,(20,20), (750,60), (b,g,r), -
1)#Creating text string to display( Color name and RGB values )
        text = recognize_color(r,g,b) + ' R='+ str(r) +  ' G='+
str(g) +  ' B='+ str(b)

        #cv2.putText(img,text,start,font(0-
7),fontScale,color,thickness,lineType )
        cv2.putText(img,
text,(50,50),2,0.8,(255,255,255),2,cv2.LINE_AA)#For very light
colours we will display text in black colour
        if(r+g+b>=600):
            cv2.putText(img,
text,(50,50),2,0.8,(0,0,0),2,cv2.LINE_AA)

        clicked=False
```

Close the Application

If you worked with OpenCV projects, you may be familiar with this step. We have to define how to end and close the application window. Otherwise, it will run forever since we used *while (1)* to start the application. Adding the following lines is a good practice for future projects.

```
#Break the loop when user hits 'esc' key
    if cv2.waitKey(20) & 0xFF ==27:
        breakcv2.destroyAllWindows()
```

Results