

# **Color Detection Project**

## **Overview**

The **Color Detection Project** is designed to identify and display the names of colors from images or real-time video feeds based on their RGB (Red, Green, Blue) values. This project uses Python and computer vision techniques to provide an intuitive way to recognize and understand color compositions in various visual data.

## **Features**

* **Real-Time Color Detection**: Detect colors directly from live video feeds or webcam input.
* **Static Image Processing**: Analyze colors from uploaded or pre-existing images.
* **RGB to Color Mapping**: Match RGB values to color names using a predefined dataset.
* **Interactive Interface**: User-friendly interface for selecting and analyzing colors.

## **Technologies Used**

* **Programming Language**: Python
* **Libraries**:
  + OpenCV (for image and video processing)
  + NumPy (for mathematical operations)
  + Pandas (for managing the color dataset)
* **Dataset**: A CSV file containing color names mapped to their corresponding RGB value.

## **Project Structure**

* **color\_detection.py**: The main script to run the project.
* **colors.csv**: A dataset containing color names and their corresponding RGB values.
* **images/**: A folder containing sample images for testing.
* **requirements.txt**: A list of all required dependencies.

## **Project Code**

**Image Color Detection Code**

import cv2

import pandas as pd

import numpy as np

img = cv2.imread("colorpic.jpg")

imgWidth = img.shape[1] - 40

index = ['color', 'color\_name', 'hex', 'R', 'G', 'B']

df = pd.read\_csv("colors.csv", header=None, names=index)

r = g = b = xpos = ypos = 0

def getRGBvalue(event, x, y, flags, param):

global b, g, r, xpos, ypos, clicked

xpos = x

ypos = y

b,g,r = img[y,x]

b = int(b)

g = int(g)

r = int(r)

def colorname(B,G,R):

minimum = 10000

for i in range(len(df)):

d = abs(B-int(df.loc[i,"B"])) + abs(G-int(df.loc[i,"G"])) + abs(R-int(df.loc[i,"R"]))

if (d<=minimum):

minimum = d

cname = df.loc[i,"color\_name"] + "Hex" + df.loc[i, "hex"]

return cname

cv2.namedWindow("Image")

cv2.setMouseCallback("Image",getRGBvalue)

while True:

cv2.imshow("Image", img)

cv2.rectangle(img, (20,20), (imgWidth, 60),(b,g,r), -1)

text = colorname(b,g,r) + ' R=' + str(r) + ' G=' + str(g) + ' B=' + str(b)

cv2.putText(img,text, (50,50),2, 0.8, (255,255,255),2,cv2.LINE\_AA)

if(r+g+b >= 600):

cv2.putText(img,text,(50,50), 2, 0.8, (0,0,0),2,cv2.LINE\_AA)

if cv2.waitKey(20) & 0xFF == 27:

break

cv2.destroyAllWindows()

**Real Time Color Detection Code**

import numpy as np

import pandas as pd

import cv2

import imutils

camera = cv2.VideoCapture(0)

r = g = b = xpos = ypos = 0

index = ['color', 'color\_name', 'hex', 'R', 'G', 'B']

df = pd.read\_csv('colors.csv', names = index, header = None)

def getColorName(R,G,B):

minimum = 10000

for i in range(len(df)):

d = abs(R - int(df.loc[i,"R"])) + abs(G - int(df.loc[i,"G"])) + abs(B - int(df.loc[i,"B"]))

if (d <= minimum):

minimum = d

cname = df.loc[i, 'color\_name'] + ' Hex=' + df.loc[i, 'hex']

return cname

def identify\_color(event, x, y, flags, param):

global b, g, r, xpos, ypos, clicked

xpos = x

ypos = y

b, g, r = frame[y,x]

b = int(b)

g = int(g)

r = int(r)

cv2.namedWindow('image')

cv2.setMouseCallback('image', identify\_color)

while True:

(grabbed, frame) = camera.read()

frame = imutils.resize(frame, width=900)

kernal = np.ones((5, 5), "uint8")

cv2.rectangle(frame, (20,20), (800, 60),(b,g,r), -1)

text = getColorName(b,g,r) + ' R=' + str(r) + ' G=' + str(g) + ' B=' + str(b)

cv2.putText(frame,text, (50,50),2, 0.8, (255,255,255),2,cv2.LINE\_AA)

if(r+g+b >= 600):

cv2.putText(frame,text,(50,50), 2, 0.8, (0,0,0),2,cv2.LINE\_AA)

cv2.imshow('image',frame)

if cv2.waitKey(20) & 0xFF == 27:

break

camera.release()

cv2.destroyAllWindows()

## **Examples**

### **Static Image Mode**

Upload an image.



**Output**

Cursor will identify the dominant colors in the image. For example, in the above image ,the cursor hovers over a specific region in a colorful crowd, and the system identifies the color as Gray, displaying its **Hex Code (#bababe)** and **RGB values (R=194, G=180, B=177)** in the below image.



In the same image,the cursor hovers over a specific region in a colorful crowd, and the system identifies the color as **Brick Red**, displaying its **Hex Code (#CB4154)** and **RGB values (R=199, G=67, B=78)**.



### **Real-Time Detection**

The webcam feed will display the color name and its RGB value for the area under the cursor.

**Output**



