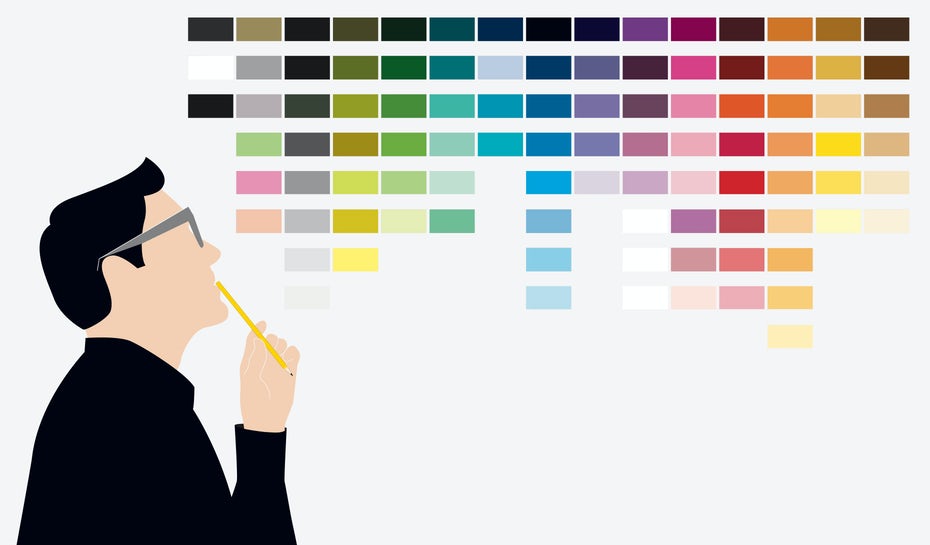


A Report on

**COLOUR DETECTION**



Under the Guidance (Mentor) of

Aishwarya Saxena.

**Submitted By:**

**Tanmay Pandey**

**Machine Learning – Python**

**College – Birla Institute of Applied Sciences.**

**Winter Internship – November-December 2020.**

**INDEX**

|  |  |  |
| --- | --- | --- |
| **SR. No.** | **TOPIC** | **PAGE No.** |
| 1 | INTRODUCTION | 3 |
| 2 | OBJECTIVE | 3 |
| 3 | BACKGROUND | 3 |
| 4 | HARDWARE AND SOFTWARE REQUIREMENTS | 4 |
| 5 | CODE | 5-11 |
| 6 | OUTPUT | 11-12 |
| 7 | FUTURE SCOPE | 13 |
| 8 | CONCLUSION | 13 |
| 9 | BIBLIOGRAPHY AND REFERENCES | 13 |

**COLOUR DETECTION**

**Introduction:**

Colour is one of the most important characteristics of an image, and Colour Detection is the process of detecting a colour form a range of different colours present in an image. Colour detection has its uses in various sectors like image processing, computer vision, gaming applications, thermal scanners, monitor screen calibration etc.

Therefore, in order to achieve this task of colour detection in computer also like humans, we use machine learning in python to make a computer understand the difference between different colours, so that the computer could recognize objects and colours very easily.

**Objectives:**

1. Colour detection can be effectively applied when we need to differentiate between different objects on the basis of their colour on a large scale.
2. It will reduce human error and time to filter objects based on their colour.
3. It is user friendly and easily accessible.
4. Colour based analysis like in scanners, monitors will also get benefitted with this project.

**Background**

In this colour detection Python project, I am going to build a user-friendly GUI based application through which we can get the colour name of any pixel in an image and additionally it will also have a separate option of filtering a primary colour present in the image. The various modules and libraries that will be used in the project are:

* **Open CV**: It will be used to process the image and convert it into RGB and HSV format.
* **Pandas**: It will be used to read and process the dataset imported for colour recognition.
* **Numpy**: It will be used to store the image array in a numpy array and also the range of colour for filtering.
* **Tkinter:** It is also used to make the GUI menu of the project.

The dataset for the project has been taken from **Kaggle**. It contains names of all 855 colours along with their RGB values which is used for colour detection.

**Hardware and Software Requirements**

**Hardware Requirements:**

|  |  |
| --- | --- |
| Hardware Tools | Minimum Requirements |
| Processor | i5 or above |
| Hard Disk | 10 GB |
| RAM | 8 GB |
| Monitor | 15” Coloured |
| Mouse | Optical |
| Keyboard | 122 Keys |

**Software Requirements:**

|  |  |
| --- | --- |
| Software Tools | Minimum Requirements |
| Platform | Windows, Linux or MacOS |
| Operating System | Windows, Linux or MacOS |
| Technology | Machine Learning – Python |
| Scripting Language | Python |
| IDE | Pycharm (or Sublime) |

**Source Code of Program**

**# importing libraries**

import cv2

import numpy as np

import pandas as pd

from tkinter import \*

**# declaring global variables**

click = False

r = g = b = x\_pos = y\_pos = 0

**# function to calculate approximate RGB value of required color.**

def get\_color\_name(R, G, B):

minimum = 1000

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']

return cname

**# function to get x,y coordinates of pixel where mouse double clicked**

def draw\_function(event, x, y, flags, params):

if event == cv2.EVENT\_LBUTTONDBLCLK:

global b, g, r, x\_pos, y\_pos, click

click = True

x\_pos = x

y\_pos = y

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

b = int(b)

g = int(g)

r = int(r)

**# function to perform Pixel Color Detection**

def pixel\_det(img\_path):

ipath = str(img\_path)

csv\_path = 'colors.csv'

global img

**# reading csv file**

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

global df

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

**# reading image**

img = cv2.imread(ipath)

img = cv2.resize(img, (800, 600))

**# creating window**

cv2.namedWindow('image')

cv2.setMouseCallback('image', draw\_function)

while True:

cv2.imshow('image', img)

if click:

**# cv2.rectangle(image, startpoint, endpoint, color, thickness)-1 fills entire rectangle**

cv2.rectangle(img, (20, 20), (600, 60), (b, g, r), -1)

**# Creating text string to display (Color name and RGB values)**

text = get\_color\_name(r, g, b) + ' 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)

**# 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)

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

break

cv2.destroyAllWindows()

**# Function for Color Filtering**

def Filter(img\_path, rcolor):

**# getting hsv range of color from file**

index = ['color\_name', 'L1', 'L2', 'L3', 'U1', 'U2', 'U3']

fdf = pd.read\_csv("Colors\_HSV.csv", names=index, header=None)

c = fdf[fdf["color\_name"] == rcolor]

l1 = [int(c['L1']), int(c['L2']), int(c['L3'])]

u1 = [int(c['U1']), int(c['U2']), int(c['U3'])]

**# importing image in bgr format**

ipath = str(img\_path)

img = cv2.imread(ipath)

**# converting bgr to rbg**

grid\_RGB = cv2.cvtColor(img, cv2.COLOR\_BGR2RGB)

**# converting image to hsv format**

grid\_HSV = cv2.cvtColor(grid\_RGB, cv2.COLOR\_RGB2HSV)

**# lower color range and upper color range**

lower = np.array(l1)

upper = np.array(u1)

**# creating a mask of detected color**

mask = cv2.inRange(grid\_HSV, lower, upper)

**# removing mask and viewing image**

res = cv2.bitwise\_and(grid\_RGB, grid\_RGB, mask=mask)

res = cv2.resize(res, (1100, 700))

res = cv2.cvtColor(res, cv2.COLOR\_BGR2RGB)

cv2.imshow('filtered image', res)

cv2.waitKey(0)

cv2.destroyAllWindows()

**# Button Function of Main Menu**

def Pixel\_Button\_Function():

img\_path = TextPX.get()

win.destroy()

pixel\_det(img\_path)

def Filter\_Button\_Function():

img\_path = TextPX.get()

color = TextPY.get()

win.destroy()

Filter(img\_path, color)

**# Main of the Program**

**# Window Creation.**

win = Tk()

win.geometry("1165x800")

win.title("COLOUR DETECTION PROGRAM")

win.configure(background="cyan")

**# Frames**

TopFrame = Frame(win, bg="cyan", pady=2, width=1150, height=100, relief=RIDGE)

TopFrame.grid(row=0, column=0)

Gtitle = Label(TopFrame, font=("Times", 50, "bold"), text="COLOUR DETECTION PROGRAM", bd=21, bg="cyan", fg="black", justify=CENTER)

Gtitle.grid(row=0, column=0)

ImageFrame = Frame(win, bd=10, bg="lemon chiffon", pady=2, width=1150, height=150, padx=2, relief=RIDGE)

ImageFrame.grid(row=1, column=0)

MainFrame = Frame(win, bd=10, bg="lemon chiffon", pady=2, width=1155, height=300, padx=2, relief=RIDGE)

MainFrame.grid(row=2, column=0)

LeftFrame = Frame(MainFrame, bd=10, width=562, height=300, pady=2, padx=2, bg="lemon chiffon", relief=RIDGE)

LeftFrame.pack(side=LEFT)

RightFrame = Frame(MainFrame, bd=10, width=562, height=300, pady=2, padx=2, bg="lemon chiffon", relief=RIDGE)

RightFrame.pack(side=RIGHT)

BottomFrame = Frame(win, bg="cyan", pady=2, width=1150, height=50, relief=RIDGE)

BottomFrame.grid(row=3, column=0)

Btitle = Label(BottomFrame, font=("Times", 20, "bold"), text="Created By: Tanmay Pandey", bd=21, bg="cyan", fg="black", justify=CENTER)

Btitle.grid(row=0, column=0)

**# text area for getting link of image and color name in main menu**

imgtitle1 = Label(ImageFrame, font=("Times", 16, "bold"), text="Enter the source of image you want for color detection.", bg="lemon chiffon", bd=21, fg="black", justify=LEFT)

imgtitle1.grid(row=0, column=1)

imgtitle2 = Label(ImageFrame, font=("Times", 16, "bold"), text="Image Source:", bg="lemon chiffon", bd=21, fg="black", justify=RIGHT)

imgtitle2.grid(row=1, column=0)

src = str()

TextPX = Entry(ImageFrame, font=("arial", 16, "bold"), bd=2, fg="blue", textvariable=src, width=80, justify=LEFT)

TextPX.grid(row=1, column=1, sticky=W)

csrc = str()

TextPY = Entry(RightFrame, font=("arial", 16, "bold"), bd=2, fg="blue", textvariable=csrc, width=26, justify=LEFT)

TextPY.grid(row=1, column=0, sticky=E)

**# Labels**

LeftLabel1 = Label(LeftFrame, font=("Times", 16, "bold"),text="This Option is for color detection of a pixel in a image.\n Click to perform pixel detection\n\n\n\n", bg="lemon chiffon", bd=21, fg="black", justify=CENTER)

LeftLabel1.grid(row=0, column=0)

RightLabel1 = Label(RightFrame, font=("Times", 16, "bold"), text="This Option is for filtering a colour from image.\n (80% Accuracy)\n Fill the details and click the button to perform color filter", bg="lemon chiffon", bd=21, fg="black", justify=CENTER)

RightLabel1.grid(row=0, column=0)

RightLabel2 = Label(RightFrame, font=("Times", 16, "bold"), text="Enter Colour Name: ", bg="lemon chiffon", bd=2, fg="black")

RightLabel2.grid(row=1, column=0, sticky=W)

**# Buttons**

button1 = Button(LeftFrame, text="PIXEL COLOR DETECTION", font=("Times 16 bold"), height=2, width=44, bg="OliveDrab2", command=Pixel\_Button\_Function)

button1.grid(row=1, column=0)

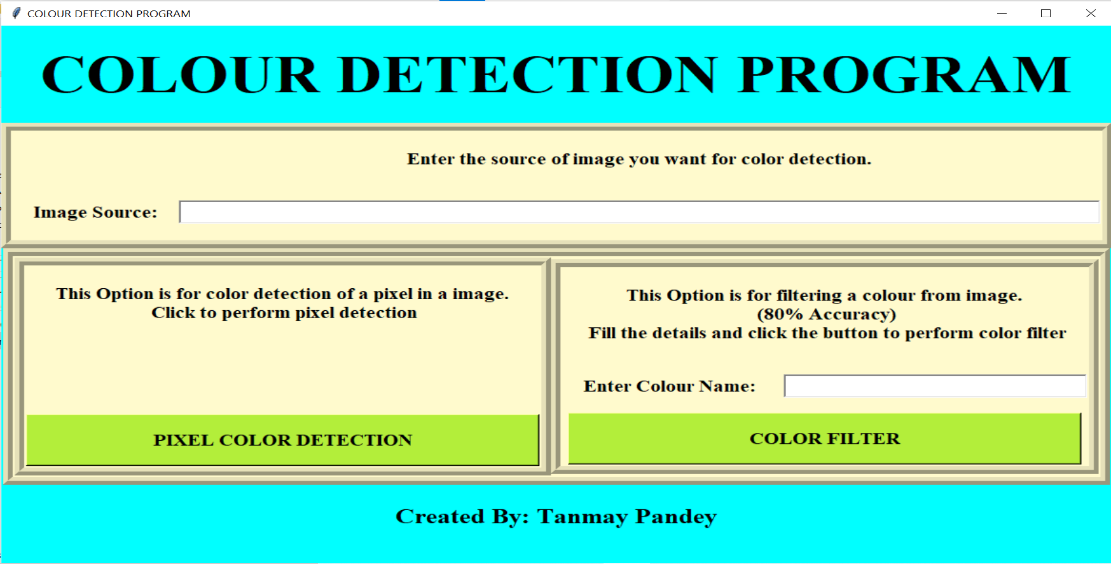
button2 = Button(RightFrame, text="COLOR FILTER", font=("Times 16 bold"), height=2, width=44, bg="OliveDrab2", command=Filter\_Button\_Function)

button2.grid(row=3, column=0)

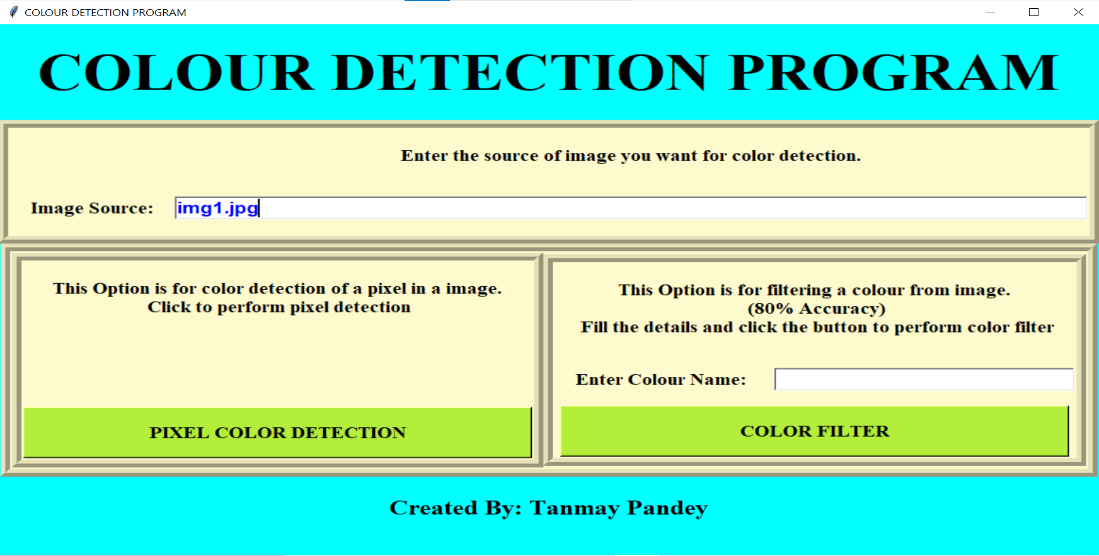
win.mainloop()

**Output**

1. **Main Menu of the Program.**



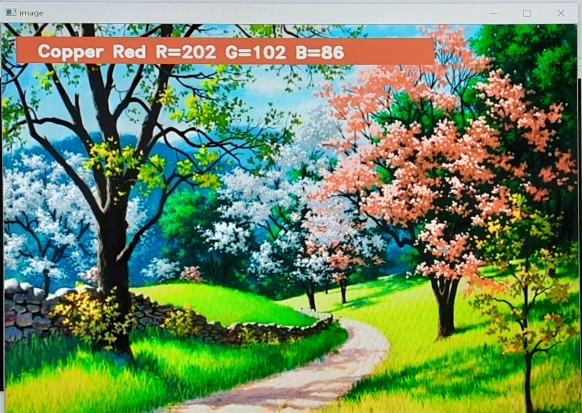
1. **Entering Values for Pixel Colour detection and clicking button.**



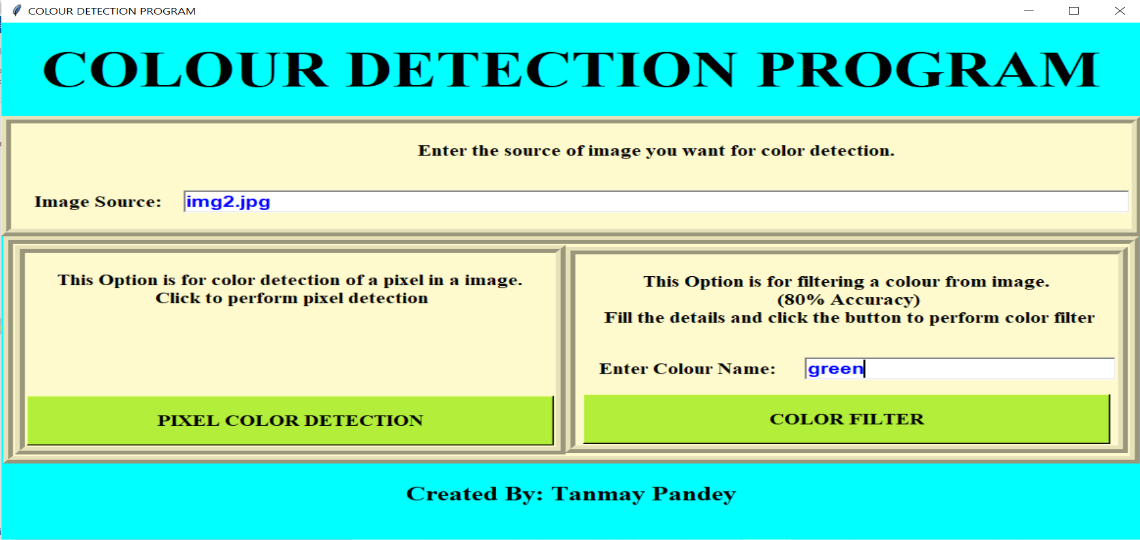
1. **Image Window for Pixel Colour detection.**



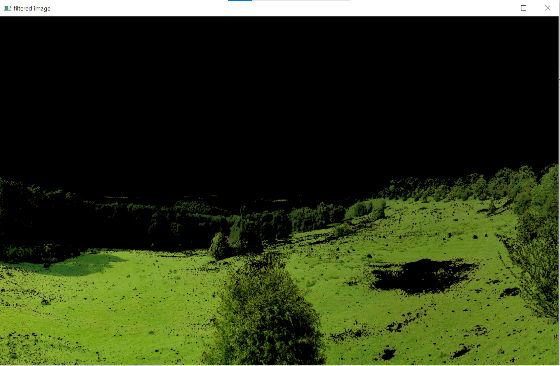
1. **Double Clicking anywhere in the Image Window To get Colour Name.**



1. **Entering Values for Colour Filter (e.g. By Green Colour) and clicking button.**



1. **Filtered Image by Colour Filter.**



Filtered Image

1. **More Examples of Colour Filter.**



Skin Color Filter

**Future Scope:**

In future, colour detection systems will be highly useful not only in the IT sector but also in the field of medical sciences, like in order to detect certain blood circulatory problems, thermal image processing is used and there, colour detection systems will play a vital role.

With the help of colour sensors programmed with this project, robots can achieve run time colour detection and can easily identify objects, thereby contributing a major role in computer vision and robotics.

In Remote sensing techniques, colour detection is the only way to recognize the land cover pattern in any particular area, like the colour red denotes forest cover, blue denotes water bodies while pink denotes barren land and white denotes snow cover etc.

Hence the future applications of this project are huge.

**Conclusion:**

With the help of Machine Learning in Python, the program had been made to detect almost all colours like red, green, blue, yellow etc by using pixel colour detection method. Also, it can filter a particular colour from the image and display the coloured region only using open cv libraries and can prove an effective program in future development of image processing.

**Bibliography and Reference:**

* Datasets ([www.kaggle.com](http://www.kaggle.com))
* ColourDetectionReference([https://www.mathworks.com/help/supportpkg/android/ref/colour-detection.html](https://www.mathworks.com/help/supportpkg/android/ref/color-detection.html))
* Open CV Libraries (<https://opencv.org/>)
* Pandas Libraries (<https://pandas.pydata.org/>)
* Numpy Library (<https://numpy.org/>)
* Propagate knowledge (<https://youtu.be/kwX2wTML-6A>)
* Images(<https://expresstricks.com/what-are-some-beautiful-lines-to-comment-on-beautiful-pictures-best-comments-for-beautiful-pics/>)
* Images(<https://www.shutterstock.com/search/scenery>)
* Images(<https://www.uniregensburg.de/Fakultaeten/phil_Fak_II/Psychologie/Psy_II/beautycheck/english/prototypen/prototypen.html>)