

Project Report
Artificial Intelligence (UCS411)

Name	Roll No
Khushi Bansal	102103624
Nishtha Kumari	102103604
Chhavi Dhankar	102103605
Aarchi	102103599

Submitted To:
Ms. Paluck



PROJECT ON COLOUR RECOGNITION SYSTEM

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

**THAPAR INSTITUTE OF ENGINEERING AND TECHNOLOGY,
(DEEMED TO BE UNIVERSITY), PATIALA, PUNJAB**

Jan-May 2023

INTRODUCTION

Artificial Intelligence (AI) has been revolutionizing the way we interact with technology, and one of its applications is color detection. Color detection is used in a wide range of applications, from detecting traffic lights in autonomous vehicles to sorting objects in industrial settings. In this project, we aim to develop a color detection system using Python, OpenCV, CSV, NumPy, and K-Nearest Neighbors (KNN) algorithm.

OpenCV is a popular computer vision library that provides tools for image and video processing. It provides a comprehensive set of functions for image processing, such as image filtering, edge detection, and feature detection. NumPy is a powerful numerical computing library for Python, which allows us to perform complex calculations on large arrays and matrices efficiently. The KNN algorithm is a machine learning algorithm that can be used for classification or regression problems.

It is a non-parametric algorithm that works by finding the K closest points to a given data point and classifying it based on the majority class of those K points.

We will be using a CSV file containing the RGB values of different colors, which will be used to train our KNN model. Then, we will use our model to detect the color of an image by finding the K closest colors to the given pixel and assigning the majority color to it.

In summary, this project aims to demonstrate the use of AI and machine learning in color detection, which can be useful in various applications. The use of Python, OpenCV, CSV, NumPy, and KNN algorithm makes the implementation of this project efficient and straightforward.

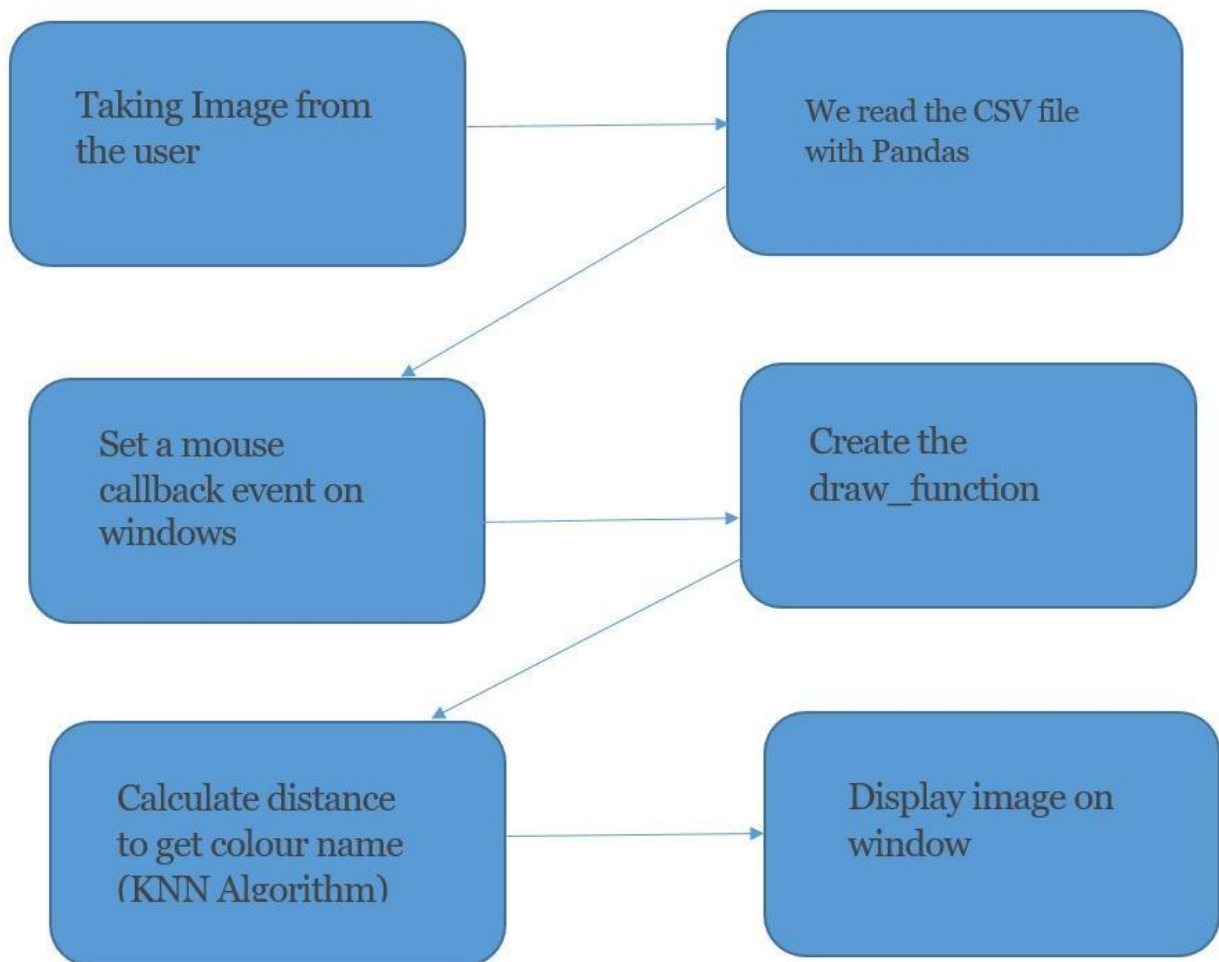
Literature Review

1. In 2020, Batur Alp Hakul et al. [1] proposed Colour recognition using color histogram feature extraction and Knearest neighbour classifier. The KNN classifier is used to distinguish twelve different colors. These colors are blue, black, brown, green, navy, orange, forest green, red, pink, violet, white and yellow. color histogram feature extraction method is used to extract features that distinguish the colors. Black and pink have the best accuracy (90%) with $k=5$. It can be seen from the result that training data and k value are very important in classification accuracy, and the accuracy is increased with proper training dataset and correct selected k values.
2. In 2018, Shima Ramesh maniyath [2] proposed soil colour detection using digital image processing. MATLAB coding is used for the process from Munsell soil chart images is used for creation of the database. HSV segmentation algorithm is used to segregate soil part from the background of given input image. Images are classified based on their RGB values using K-NN and images are labelled using Munsell soil notation. The output is obtained as per the Munsell soil notation.
3. In 2018, M. Mary Deepa et al.[3] proposed a method for using color threshold for identification of 2-D images using the RGB Color model to detect colors. The colors detected here are blue, red, green, magenta, cyan. The given 3-D color image is converted into grey-scale image, then the 2 images are subtracted and 2 dimensional black and white picture is obtained, unwanted noise from the image is removed by Median Filtering. Detecting with a linked component Digital images are marked in linked region and metric for every marking area is calculated using bounding box and its properties. The shade of each image element is detected by analyzing RGB value of each pixel.is detected by analyzing RGB value of each pixel.

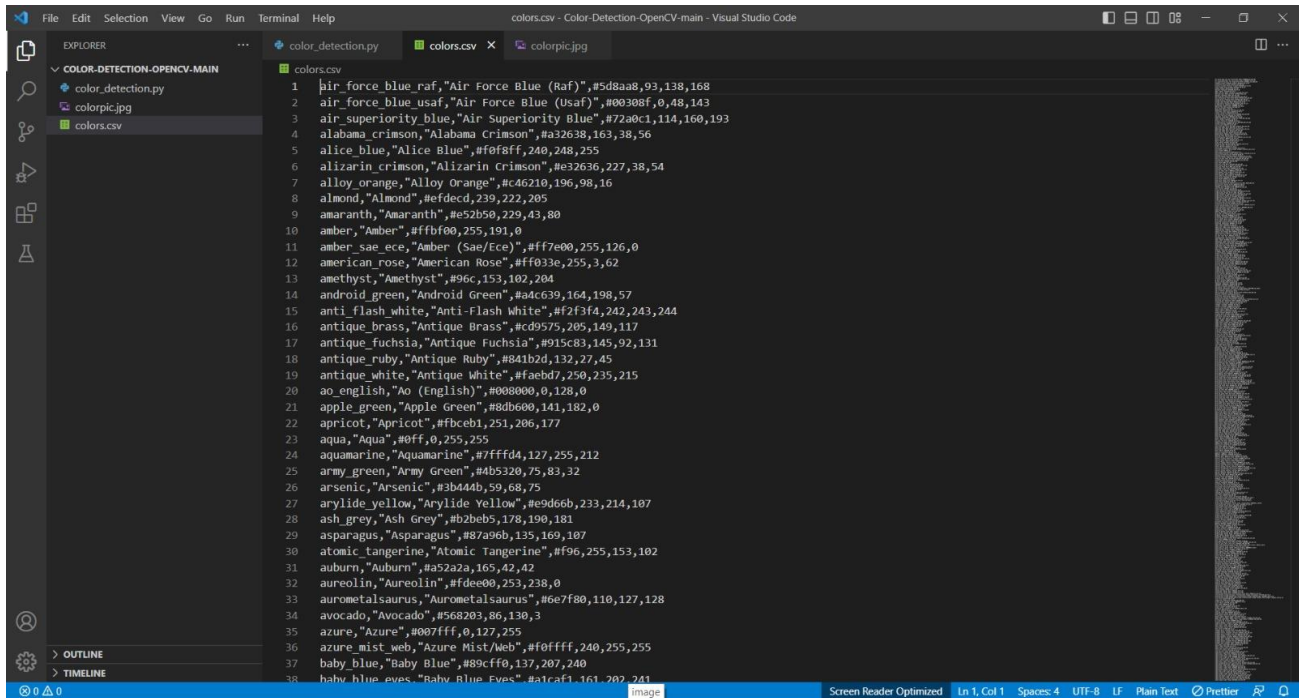
4. In 2016, can eyupogl[4] proposed color face recognition using K-NN classification algorithm and PCA. KNN is used for classification of color face images. Initially K-NN Classifier used to perform the classification. Later, principal component analysis (PCA) and K-NN classifier are used together to extract features of color face images and to simplify the image data. The applications are tested for different color space model and K values. The colour space models are HSV, RGB and YIQ. Finally, results are compared with each other. Based on the mentioned 2 tables, the classification accuracy of K-NN and classification accuracies of PCA and K-NN, the increase of k value decreases the classification accuracies. Besides, the change of k value does not affect the classification accuracy in some situations.
5. In 2015, SK Niranjan et al.[5] presented a method for classification of Raw Arecanut. Classification is based on colour attributes. Colour moments and colour histogram along with KNN algorithm is used for classification of raw arecanut. This model uses - classifier with 4 distance measures to examine the impact. Result of 98% was obtained using K nearest neighbor having K value as 3 and Euclidean distance measure for colour histogram features. In theoretical approach, accuracy around 20% was obtained.
6. In 2013, Mihir Narayan Mohanty et al.[6] proposed Statistical approach for color recognition. It begins with the image possession and boundary detection of the object is done to distinguish it from the background. The binary values of different layers Id obtained using iterative method. For processing pixel wise Region of Interest (ROI) was used. Statistical method is used to determine threshold that helps in colour detection of an object. Thresholding method is applied over the ROI obtained, the detection of the colour of the given object is performed.

Methodology

Flow Chart:



Dataset Used:



```
1  air_force_blue_raf,"Air Force Blue (Raf)","#5d8aa8,93,138,168
2  air_force_blue_usaf,"Air Force Blue (Usaf)","#00308f,0,48,143
3  air_superiority_blue,"Air Superiority Blue","#72a0c1,114,160,193
4  alabama_crimson,"Alabama Crimson","#a32638,163,38,56
5  alice_blue,"Alice Blue","#f0f8ff,240,248,255
6  alizarin_crimson,"Alizarin Crimson","#e32636,227,38,54
7  alloy_orange,"Alloy Orange","#c46210,196,98,16
8  almond,"Almond","#efdecf,239,222,205
9  amaranth,"Amaranth","#e52b50,229,43,80
10 amber,"Amber","#ffbf00,255,191,0
11 amber_sae_ece,"Amber (Sae/Ece)","#ff7e00,255,126,0
12 american_rose,"American Rose","#ff033e,255,3,62
13 amethyst,"Amethyst","#96c,153,102,204
14 android_green,"Android Green","#a4c639,164,198,57
15 anti_flash_white,"Anti-Flash White","#f2f3f4,242,243,244
16 antique_brass,"Antique Brass","#cd9575,205,149,117
17 antique_fuchsia,"Antique Fuchsia","#915c83,145,92,131
18 antique_ruby,"Antique Ruby","#841b2d,132,27,45
19 antique_white,"Antique White","#faebd7,250,235,215
20 ao_english,"Ao (English)","#008000,0,128,0
21 apple_green,"Apple Green","#8db600,141,182,0
22 apricot,"Apricot","#fbc02d,255,206,177
23 aqua,"Aqua","#00ffff,0,255,255
24 aquamarine,"Aquamarine","#7fffd4,127,255,212
25 army_green,"Army Green","#4b5320,75,83,32
26 arsenic,"Arsenic","#3b444b,59,68,75
27 arylide_yellow,"Arylide Yellow","#e9d6eb,233,214,107
28 ash_grey,"Ash Grey","#b2bebe,178,190,181
29 asparagus,"Asparagus","#87a96b,135,169,107
30 atomic_tangerine,"Atomic Tangerine","#f96,255,153,102
31 auburn,"Auburn","#a52a2a,165,42,42
32 aureolin,"Aureolin","#fdee00,253,238,0
33 aurometalsaurus,"Aurometalsaurus","#6e7f80,110,127,128
34 avocado,"Avocado","#56b203,86,130,3
35 azure,"Azure","#007fff,0,127,255
36 azure_mist_web,"Azure Mist/Web","#00ffff,240,255,255
37 baby_blue,"Baby Blue","#89cffe,137,207,240
38 baby_blue_eyes,"Baby Blue Eyes","#a1c4cf,161,202,241
```

Algorithm:

The K-Nearest Neighbors (KNN) algorithm is a popular machine learning algorithm used for classification tasks, including color recognition. In color recognition, KNN can be used to classify an input color into one of several predefined color categories based on the colors of its nearest neighbors in the training dataset. KNN can be sensitive to the choice of K and the distance metric. In general, larger values of K can improve the robustness of the algorithm but may also result in lower accuracy. Similarly, different distance metrics may work better for different color spaces. Therefore, it is important to experiment with different values of K and distance metrics to find the optimal settings for a particular color recognition task.

Here's a brief overview of how the KNN algorithm can be used for color recognition:

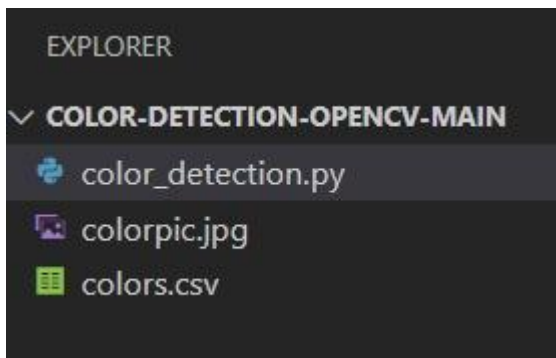
1. Collect a dataset of labeled color samples. Each sample should be represented as a feature vector that contains the RGB (red, green, blue) values of the color. Split the dataset into training and testing sets.
2. Choose a value of K (the number of nearest neighbors to consider) and a distance metric (e.g., Euclidean distance) to measure the similarity between the input color and the training set.
3. For each color in the testing set, find its K nearest neighbors in the training set based on the chosen distance metric.
4. Classify the input color by assigning it to the most common color category among its K nearest neighbors.
5. Evaluate the accuracy of the classification algorithm using metrics such as accuracy, precision, and recall.

STEP BY STEP WORKING OF ALGORITHM

1. Creating Files and Importing Libraries:

The project folder contains 3 files:

- Color_detection.py – main source code of our project.
- Colorpic.jpg – sample image for experimenting.
- Colors.csv – a file that contains our dataset.



```
import cv2
import pandas as pd
```

2. Taking an image from the user

```
img_path = r'colorpic.jpg'
img = cv2.imread(img_path)
```

3. Next, we read the CSV file with pandas:

The pandas library is very useful when we need to perform various operations on data files like CSV. `pd.read_csv()` reads the CSV file and loads it into the pandas DataFrame. We have assigned each column with a name for easy accessing.

```
# declaring global variables (are used later on)
clicked = False
r = g = b = x_pos = y_pos = 0

# Reading csv file with pandas and giving names to each column
index = ["color", "color_name", "hex", "R", "G", "B"]
csv = pd.read_csv('colors.csv', names=index, header=None)
```

4. Set a mouse callback event on a window:

First, we created a window in which the input image will display. Then, we set a callback function which will be called when a mouse event happens. With these lines, we named our window as 'image' and set a callback function which will call the `draw_function()` whenever a mouse event occurs.

```
# function to get x,y coordinates of mouse double click
def draw_function(event, x, y, flags, param):
    if event == cv2.EVENT_LBUTTONDBLCLK:
        global b, g, r, x_pos, y_pos, clicked
        clicked = True
        x_pos = x
        y_pos = y
        b, g, r = img[y, x]
        b = int(b)
        g = int(g)
        r = int(r)

cv2.namedWindow('image')
cv2.setMouseCallback('image', draw_function)
```

5. Create the draw_function:

It will calculate the rgb values of the pixel which we double click. The function parameters have the event name, (x,y) coordinates of the mouse position, etc. In the function, we check if the event is double-clicked, then we calculate and set the r,g,b values along with x,y positions of the mouse.

6. Calculate distance to get color name:

We have the r,g and b values. Now, we need another function which will return us the color name from RGB values. To get the color name, we calculate a distance(d) which tells us how close we are to color and choose the one having minimum distance.

Our distance is calculated by this formula:

$$d = \text{abs}(\text{Red} - \text{ithRedColor}) + (\text{Green} - \text{ithGreenColor}) + (\text{Blue} - \text{ithBlueColor})$$

```
# function to calculate minimum distance from all colors and get the most matching color
def get_color_name(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
```

7. Display image on the window:

Whenever a double click event occurs, it will update the color name and RGB values on the window.

Using the cv2.imshow() function, we draw the image on the window. When the user double clicks the window, we draw a rectangle and get the color name to draw text on the window using cv2.rectangle and cv2.putText() functions.

```
while True:

    cv2.imshow("image", img)
    if clicked:

        # cv2.rectangle(image, start point, 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 = get_color_name(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)

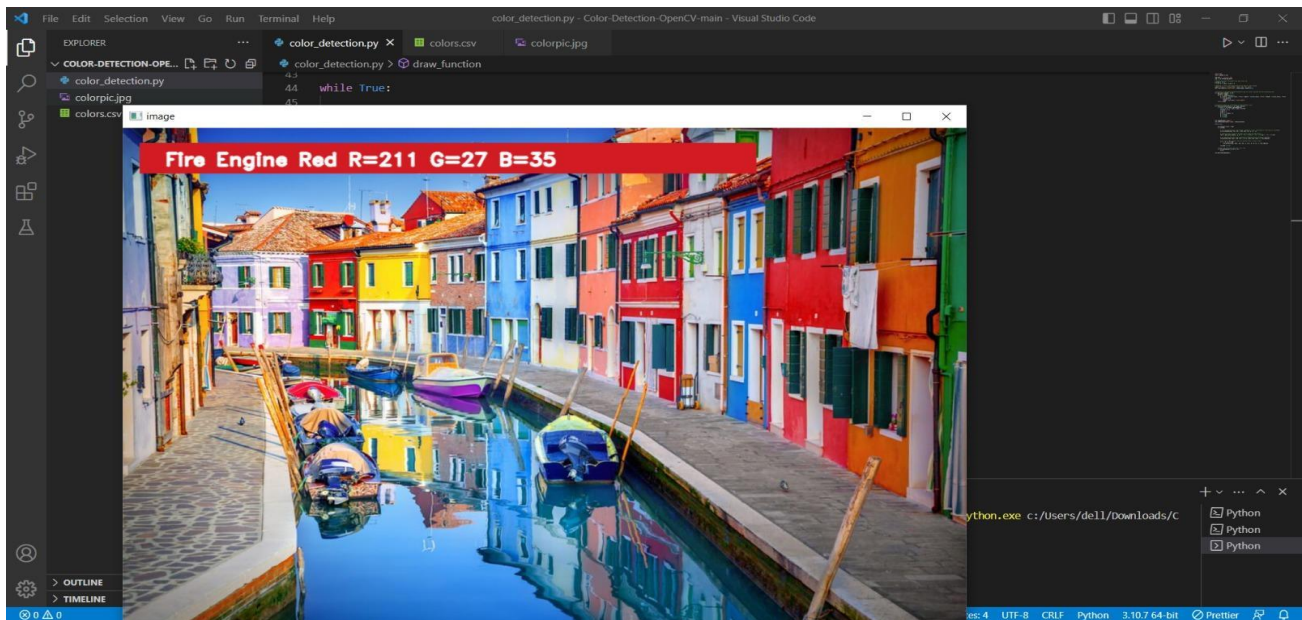
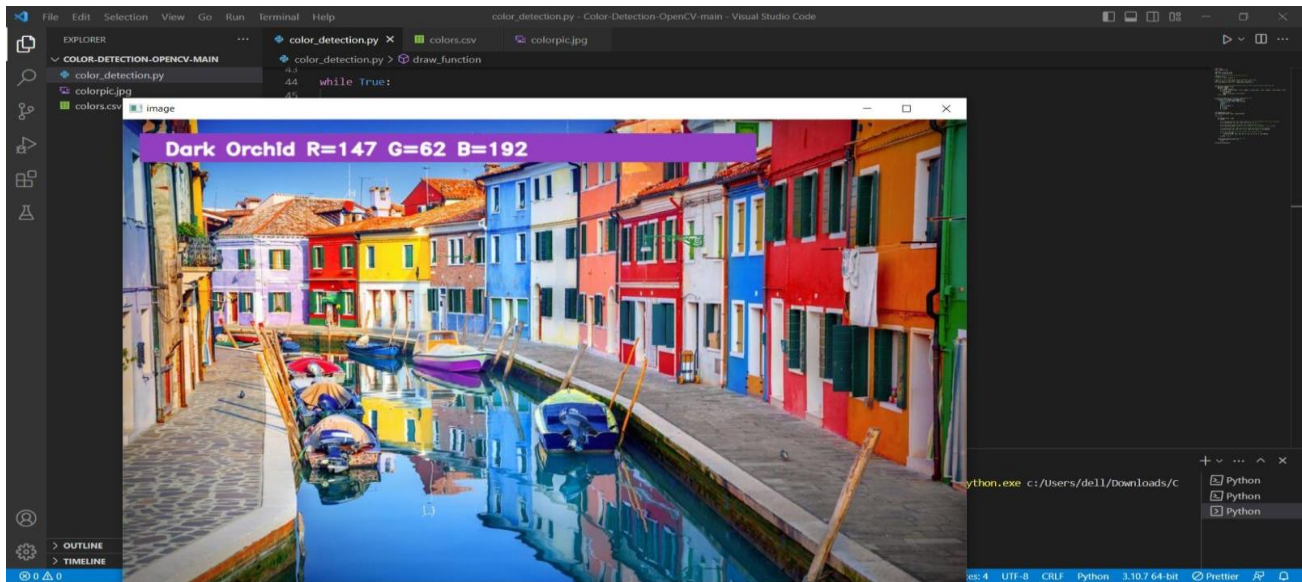
        clicked = False

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

cv2.destroyAllWindows()
```

Result

```
PS C:\Users\dell\Downloads\Color-Detection-OpenCV-main> & C:/Users/dell/AppData/Local/Programs/Python/Python310/python.exe c:/Users/dell/Downloads/Color-Detection-OpenCV-main/color_detection.py
```



Applications:

- Color detection has a wide range of applications in various industries, including but not limited to:
 - Traffic control: In traffic management systems, color detection can be used to identify the colors of traffic lights and control the flow of vehicles.
 - Industrial automation: In manufacturing and industrial settings, color detection can be used to sort products based on their color, ensuring quality control and efficient production.
 - Robotics: Color detection can be used in robotics for object recognition and manipulation. For example, a robot can be programmed to pick up an object of a specific color using color detection.
 - Medical diagnosis: Color detection can be used in medical diagnosis to detect changes in the color of tissues or fluids, which can indicate certain health conditions.
 - Agriculture: In agriculture, color detection can be used to detect plant diseases by identifying changes in the color of the leaves or other parts of the plant.
 - Art and design: In the field of art and design, color detection can be used to identify and match colors in images, allowing for accurate color reproduction in print and digital media.
- These are just a few examples of the many applications of color detection. The use of artificial intelligence and machine learning in color detection can greatly enhance the accuracy and efficiency of these applications.

References :

- [1] Rabia Bayraktar, Batur Alp Akgul and Kadir Sercan Bayram, "Colour recognition using colour histogram feature extraction and Knearest neighbor classifier", 2020.
- [2] Shima Ramesh maniyath,Akshatha K N, Dr. S Rama Subramoniam, Architha L and S,Ramachandra hebbbar "Soil Color Detection Using Knn Classifier", 2018.
- [3] P.Sudharshan duth and M. Mary Deepa, "Color detection in RGB modeled images using MAT LAB", 2018.
- [4] Can Eyupoglu, "Implementation of Color Face Detection Using PCA and K-NN Classifier ", 2016.
- [5] S Siddesha, SK Niranjana and VN Manjunath Aradhya, "Color Features and KNN in Classification of Raw Arecanut images", 2015.
- [6] Mihir Narayan Mohanty and Sidhanta Kumar Kar , "Statistical approach for color image detection", 2013.