MINI PROJECT REPORT

ON

“REAL TIME COLOR SEGREGATOR”

Submitted in

Partial Fulfillment of requirements for the Award of Degree

*of*

Bachelor of Technology

*In*

Computer Science and Engineering

By

**(Group Number: 40)**

# Shreya Chopra(1816410264)

# Shruti Vishwakarma (1816410269)

# Siddhartha Srivastava (1816410278)

Under the supervision of

**Kumar Saurabh**

**(Asst. Professor)**



**Pranveer Singh Institute of Technology**.

 Kanpur - Agra - Delhi National Highway - 2

Bhauti -Kanpur - 209305.

(Dr. A.P.J. Abdul Kalam Technical University)

1

**ACKNOWLEDGEMENT**

We are pleased to acknowledge “Asst. Professor Kumar Saurabh” for their valuable guidance during the course of this mini project work. We extend our sincere thanks to Mr. Pranveer Singh who gave us this opportunity to work on this project.

This project has been a guiding path for us as it has helped us in gaining knowlwdge about the topics, it gave us a clear view of machine leaning, its working and applications. We have made this project with a hope that it will be helpful to the people.

Thank You.

(15.January.2021)

2

**CONTENT**

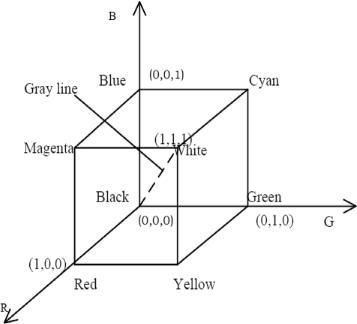
* Introduction
* Feasibility Study
* Technology Used
* Hardware Required
* Software Required
* Coding
* Conclusion
* Future Scope
* References

3

**INTRODUCTION**

Color information plays an important role in the color image segregation and real-time color sensor, which affects the result of image segregation and correct real-time temperature value. In this paper, a novel real-time color image segregation method is proposed, which is based on color similarity in RGB color space. According to the color and luminance information in RGB color space, the dominant color is determined at first, and then color similarity can be calculated with the proposed calculation method of color component, which creates a color-class map. Next, the information of the corresponding color-class map is utilized to classify the pixels. Due to the characteristic that thermal inks feature color values that change in real time as the temperature changes, the segregation results of thermal ink can be used as a real-time color sensor.

RGB space is represented by the three primary colors of red, green, blue; other colors are made up with the three primary colors. The RGB model is represented by the Cartesian coordinate system.



4

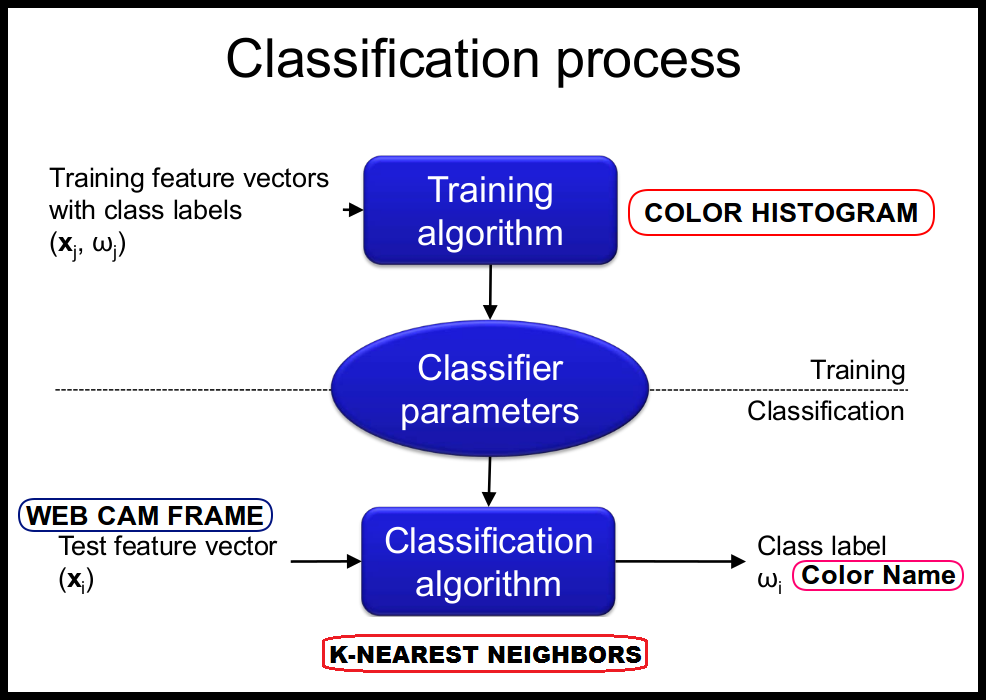
This is a color recognition application. The aim of this project is to make an application which will print the name of the coloRs just by clicking on them. It focuses on color classification by K-Nearest Neighbours Machine Learning Classifier which is trained by R, G, B Color Histogram.

Colors are made up of 3 primary colors: Red, Green and Blue. It is easy for us to recognize any color because our brain and eyes functions orderly but it is not very straightforward for the computers. In computers, colors are defined within the range 0-255. It would be very amazing to know that we can define colors in 16,581,375 ways.

**What does this program do?**

1. **Feature Extraction:** Perform feature extraction for getting the R, G, B Color Histogram values of [training images](https://github.com/ahmetozlu/color_classifier/tree/master/src/training_dataset)
2. **Training K-Nearest Neighbors Classifier:** Train KNN classifier by R, G, B Color Histogram values
3. **Classifying by Trained KNN:** Read Web Cam frame by frame, perform feature extraction on each frame and then classify the mean color of it by trained KNN classifier.

Colors are classified by using K-Nearest Neighbour Machine Learning classifier algorithm. This classifier is trained by image R, G, B Color Histogram values. The general work flow is given at the below.



5

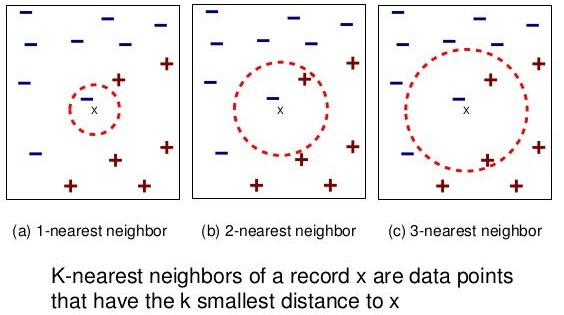
We should basically know 2 main pheomenon to understand basic Recognition Systems of Computer Vision and Machine Learning.

**1. Feature Extraction** = Color Histogram

Color Histogram is a representation of the distribution of colors in an image. For digital images, a color histogram represents the number of pixels that have colors in each of a fixed list of color ranges, that span the image's color space, the set of all possible colors.

**2. Classification** = K-Nearest Neighbors Algorithm

K nearest neighbors is a simple algorithm that stores all available cases and classifies new cases based on a similarity measure (e.g., distance functions). KNN has been used in statistical estimation and pattern recognition already in the beginning of 1970’s as a non-parametric technique.



6

**FEASIBILITY STUDY**

This project will be useful in following ways:-

In self-driving car, to detect the traffic signals.

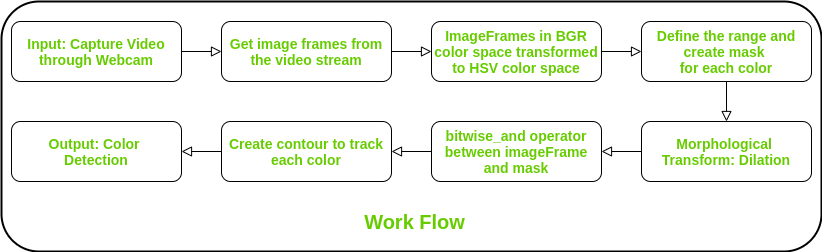
Multiple color detection is used in some industrial robots, to performing pick-and-place task in separating different colored objects.

This is an implementation of detecting multiple colors (here, only **red***,***green**and**blue** colors have been considered) in real-time using Python programming language.

This project focuses on the needs of the society. People who face problem in identifying colors will be able to do it in a much easier way as this project is based on the idea of color segregator.

It will find its greator application in “Color Blindness”. People who face such things will be able to differentiate between the colors. And they will be able to identify colors easily.

Real time color segregator works according to its name, as the name suggests color segregation, it will calculate the value of the color on the basis of RGB values. And suppose we are given a colorful picture, then wherever we place the pointer, it will show the name of the certain color.



**TECHNOLOGY USED**

Two main phenomena used are basics of Computer Vision and Machine Learning. OpenCV is used for color histogram calculations and KNN Classifier. NumPy is used for matrix or n-dimensional array calculations.

OpenCV:

**OpenCV** (*Open Source Computer Vision Library*) is a [library of programming functions](https://en.wikipedia.org/wiki/Library_(computing)) mainly aimed at real-time [computer vision](https://en.wikipedia.org/wiki/Computer_vision).[[1]](https://en.wikipedia.org/wiki/OpenCV#cite_note-1) Originally developed by [Intel](https://en.wikipedia.org/wiki/Intel_Corporation), it was later supported by [Willow Garage](https://en.wikipedia.org/wiki/Willow_Garage) then Itseez (which was later acquired by Intel). The library is [cross-platform](https://en.wikipedia.org/wiki/Cross-platform) and free for use under the [open-source](https://en.wikipedia.org/wiki/Open-source_software) [Apache 2 License](https://en.wikipedia.org/wiki/Apache_License). Starting with 2011, OpenCV features GPU acceleration for real-time operations.

Applications of OpenCV are:

* 2D and 3D feature toolkits
* [Egomotion](https://en.wikipedia.org/wiki/Egomotion) estimation
* [Facial recognition system](https://en.wikipedia.org/wiki/Facial_recognition_system)
* [Gesture recognition](https://en.wikipedia.org/wiki/Gesture_recognition)
* [Human–computer interaction](https://en.wikipedia.org/wiki/Human%E2%80%93computer_interaction) (HCI)
* [Mobile robotics](https://en.wikipedia.org/wiki/Mobile_robotics)
* Motion understanding
* [Object detection](https://en.wikipedia.org/wiki/Object_detection)
* [Segmentation](https://en.wikipedia.org/wiki/Segmentation_(image_processing)) and recognition
* [Stereopsis](https://en.wikipedia.org/wiki/Stereopsis) stereo vision: depth perception from 2 cameras
* [Structure from motion](https://en.wikipedia.org/wiki/Structure_from_motion) (SFM)
* [Motion tracking](https://en.wikipedia.org/wiki/Video_tracking)
* [Augmented reality](https://en.wikipedia.org/wiki/Augmented_reality)

KNN Classifier:

K-Nearest Neighbors (**KNN**) is one of the simplest algorithms used in **Machine Learning** for regression and **classification** problem. **KNN** algorithms use data and classify new data points based on similarity measures (e.g. distance function). **Classification** is done by a majority vote to its neighbors.

8

**KNN works** by finding the distances between a query and all the examples in the data, selecting the specified number examples (K) closest to the query, then votes for the most frequent label (in the case of **classification**) or averages the labels (in the case of regression).

Numpy:

**NumPy** is a **Python** library used for working with arrays. It also has functions for working in domain of linear algebra, fourier transform, and matrices. **NumPy** was created in 2005 by Travis Oliphant. It is an open source project and you can use it freely. **NumPy** stands for Numerical **Python**.

**NumPy** arrays facilitate advanced mathematical and other types of operations on large numbers of data. Typically, such operations are executed more efficiently and with less code than is possible using Python's built-in sequences.

9

**HARDWARE REQUIRED**

**1. i3 Processor:**

It is one of three types of processors in the "i" series (also called the Intel Core family of processors).The Core i3 processor is available in multiple speeds, ranging from 1.30 GHz up to 3.50 GHz, and features either 3 MB or 4 MB of [cache](https://www.computerhope.com/jargon/c/cache.htm). It utilizes either the LGA 1150 or LGA 1155 socket on a [motherboard](https://www.computerhope.com/jargon/m/mothboar.htm).

**2. Memory: 1GB RAM**

RAM (1GB) will be required for the proper functioning of the model.

**3. Monitor:**

Monitor will be required for the display of the picture. Picture along with its color name will be displayed on the monitor.

**4.Webcam**:

Webcam will be used for live detection of color that the person will be holding or showing in the camera.

10

**SOFTWARE REQUIRED**

**Python (OpenCV and Panda Module):**

**Python** is a general-purpose coding language—which means that, unlike HTML, CSS, and JavaScript, it can be **used for** other types of programming and software development besides web development. That includes back end development, software development, data science and writing system scripts among other things.

**OpenCV:**

**OpenCV** (*Open Source Computer Vision Library*) is a [library of programming functions](https://en.wikipedia.org/wiki/Library_(computing)) mainly aimed at real-time [computer vision](https://en.wikipedia.org/wiki/Computer_vision).[[1]](https://en.wikipedia.org/wiki/OpenCV#cite_note-1) Originally developed by [Intel](https://en.wikipedia.org/wiki/Intel_Corporation), it was later supported by [Willow Garage](https://en.wikipedia.org/wiki/Willow_Garage) then Itseez (which was later acquired by Intel). The library is [cross-platform](https://en.wikipedia.org/wiki/Cross-platform) and free for use under the [open-source](https://en.wikipedia.org/wiki/Open-source_software) [Apache 2 License](https://en.wikipedia.org/wiki/Apache_License). Starting with 2011, OpenCV features GPU acceleration for real-time operations.

Applications of OpenCV are:

* 2D and 3D feature toolkits
* [Egomotion](https://en.wikipedia.org/wiki/Egomotion) estimation
* [Facial recognition system](https://en.wikipedia.org/wiki/Facial_recognition_system)
* [Gesture recognition](https://en.wikipedia.org/wiki/Gesture_recognition)
* [Human–computer interaction](https://en.wikipedia.org/wiki/Human%E2%80%93computer_interaction) (HCI)
* [Mobile robotics](https://en.wikipedia.org/wiki/Mobile_robotics)
* Motion understanding
* [Object detection](https://en.wikipedia.org/wiki/Object_detection)
* [Segmentation](https://en.wikipedia.org/wiki/Segmentation_(image_processing)) and recognition
* [Stereopsis](https://en.wikipedia.org/wiki/Stereopsis) stereo vision: depth perception from 2 cameras
* [Structure from motion](https://en.wikipedia.org/wiki/Structure_from_motion) (SFM)
* [Motion tracking](https://en.wikipedia.org/wiki/Video_tracking)
* [Augmented reality](https://en.wikipedia.org/wiki/Augmented_reality)

11

**Pandas Module:**

In computer programming, **pandas** is a software **library** written for the **Python** programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series. It is free software released under the three-clause BSD license.

**Working with Pandas**

* Convert a Python's list, dictionary or Numpy array to a **Pandas** data frame.
* Open a local file using **Pandas**, usually a CSV file, but **could** also be a delimited text file (like TSV), Excel, etc.
* Open a remote file or database like a CSV or a JSONon a website through a URL or read from a SQL table/database.

12

**Coding**

import numpy as np

import cv2

webcam = cv2.VideoCapture(0)

while(1):

\_, imageFrame = webcam.read()

hsvFrame = cv2.cvtColor(imageFrame, cv2.COLOR\_BGR2HSV)

red\_lower = np.array([136, 87, 111], np.uint8)

red\_upper = np.array([180, 255, 255], np.uint8)

red\_mask = cv2.inRange(hsvFrame, red\_lower, red\_upper)

green\_lower = np.array([25, 52, 72], np.uint8)

green\_upper = np.array([102, 255, 255], np.uint8)

green\_mask = cv2.inRange(hsvFrame, green\_lower, green\_upper)

blue\_lower = np.array([94, 80, 2], np.uint8)

blue\_upper = np.array([120, 255, 255], np.uint8)

blue\_mask = cv2.inRange(hsvFrame, blue\_lower, blue\_upper)

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

red\_mask = cv2.dilate(red\_mask, kernal)

res\_red = cv2.bitwise\_and(imageFrame, imageFrame,

mask = red\_mask)

13

green\_mask = cv2.dilate(green\_mask, kernal)

res\_green = cv2.bitwise\_and(imageFrame, imageFrame,

mask = green\_mask)

blue\_mask = cv2.dilate(blue\_mask, kernal)

res\_blue = cv2.bitwise\_and(imageFrame, imageFrame,

mask = blue\_mask)

contours, hierarchy = cv2.findContours(red\_mask,

cv2.RETR\_TREE,

cv2.CHAIN\_APPROX\_SIMPLE)

for pic, contour in enumerate(contours):

area = cv2.contourArea(contour)

if(area > 300):

x, y, w, h = cv2.boundingRect(contour)

imageFrame = cv2.rectangle(imageFrame, (x, y),

(x + w, y + h),

(0, 0, 255), 2)

cv2.putText(imageFrame, "Red Colour", (x, y),

cv2.FONT\_HERSHEY\_SIMPLEX, 1.0,

(0, 0, 255))

contours, hierarchy = cv2.findContours(green\_mask,

cv2.RETR\_TREE,

14

cv2.CHAIN\_APPROX\_SIMPLE)

for pic, contour in enumerate(contours):

area = cv2.contourArea(contour)

if(area > 300):

x, y, w, h = cv2.boundingRect(contour)

imageFrame = cv2.rectangle(imageFrame, (x, y),

(x + w, y + h),

(0, 255, 0), 2)

cv2.putText(imageFrame, "Green Colour", (x, y),

cv2.FONT\_HERSHEY\_SIMPLEX,

1.0, (0, 255, 0))

contours, hierarchy = cv2.findContours(blue\_mask,

cv2.RETR\_TREE,

cv2.CHAIN\_APPROX\_SIMPLE)

for pic, contour in enumerate(contours):

area = cv2.contourArea(contour)

if(area > 300):

x, y, w, h = cv2.boundingRect(contour)

imageFrame = cv2.rectangle(imageFrame, (x, y),

(x + w, y + h),

(255, 0, 0), 2)

cv2.putText(imageFrame, "Blue Colour", (x, y),

cv2.FONT\_HERSHEY\_SIMPLEX,

15

1.0, (255, 0, 0))

cv2.imshow("Multiple Color Detection in Real-TIme", imageFrame)

if cv2.waitKey(10) & 0xFF == ord('q'):

cap.release()

cv2.destroyAllWindows()

break

16

**CONCLUSION**

From all the performed scenarios it is demonstrated that every color space presents its own issues that could affect the segregation done by clustering. But an important result observed from the behavior from each color space is regarding to the aspect of failure in different areas. It means that some color spaces were better performing the segregation of a given color, or a given illumination condition.There is some technique that could help to improve the robustness , like using other distance method based on the angle distance. It should be noted that despite fixing the discontinuity issue in the perceptual models, it is also necessary to fix the white and gray tones issue. This last issue is sometimes not possible or not a trivial task.. An approach to follow it is to use a given color space depending on the conditions of the scenario where it should be implementing. So the features of a given color space can really help to solve the required clustering. Unfortunately, this approach is not possible in most of the scenarios, where the machine learning algorithm must be prepared for unknown scenarios. From the previous can be stated one of the most common issues in machine learning algorithms, that most of the classifiers are weak classifiers. In the case scenarios presented in this chapter, it can also use the same solution as in any machine learning procedure; it is the boosting approach. Boosting can be coupled weak classifiers in order to create a stronger classifier. Each configuration presented in the chapter is really a weak classifier. Mixing some of them can improve the final clustering result.

The training data has a huge importance in classification accuracy. I created my training data carefully but maybe the accuracy can be higher with more suitable training data.

Another important thing is lightning and shadows. In my test images, the images which were taken under bad lighting conditions and with shadows are classified wrong (false positives), maybe some filtering algorithm should/can be implemented before the test images send to KNN classifier Thus, accuracy can be improved.

And on the basis of provided training data, the efficiency can be improved. The more accurate the data will be, the more accurate will be the result. Training is an important step in case of making an efficient working model. And training requires accurate training data values which could be used to train our model and make it work in an efficient way.

17

**FUTURE SCOPE**

The future scope of this project are very positive. As it will be a very powerful tool for the people who are facing color blindness and the people who are blind by birth. They can use this model to identify colors.

And new features can be added in the future which will be more helpful to the people who are willing to use it. This model can be used by people who are working in the fields where they need to recognize colors.

Real time color segregator calculates values on the basis of RGB values and displays the name of the color.

Its main focus will be according to the needs of the people. Because people do not really know the name of every color, they know only the name of the primary colors hence it sometimes becomes difficult for the people to identify the exact name of the color and at that point, this model will help in doing so. It will help to identify the name of each and every color.

18

**REFERENCES**

1. <https://docs.opencv.org/master/>

2. <https://docs.python.org/3/tutorial/modules.html>

3. <https://www.javatpoint.com/k-means-clustering-algorithm-in-machine-learning>

4. <https://en.wikipedia.org/wiki/Machine_learning>

19

20