COLOR DETECTION PROGRAM USING DEEP LEARNING

BY

AUGUSTINE OSADEBE EROMOSELE
PSC 1813816

DEPARTMENT OF COMPUTER SCIENCE

FACULTY OF COMPUTING

UNIVERSITY OF BENIN

BENIN CITY.

January, 2026

COLOR DETECTION PROGRAM USING DEEP LEARNING

BY

AUGUSTINE OSADEBE EROMOSELE
PSC 1813816

IN FULFILLMENT OF THE REQUIREMENT
FOR THE AWARD OF
BACHELOR OF SCIENCE (B.Sc.) DEGREE IN

DEPARTMENT OF COMPUTER SCIENCE

FACULTY OF COMPUTING

UNIVERSITY OF BENIN

BENIN CITY.

January, 2026

CERTIFICATION

This is to certify that this project work "COLOR DETECTION PROGRAM USING DEEP LEARNING" was carried out by Augustine Osadebe Eromosele and submitted to the department of Computer Science in fulfillment for the award of Bachelor of Science (B.Sc.) Degree, University of Benin, Benin City, Nigeria.

_____ _____

Dr. (Mrs.) R. O. Osaseri Date
Project Supervisor

_____ _____

Dr. (Mrs.) A.R Usiobaifo Date
Head of Department

DEDICATION

This project is dedicated to Almighty God, whose grace, protection and wisdom have brought me this far. It is also dedicated to my loving parents Mr. Benjamin Eromosele and Mrs. Bridget Ilobekemen Eromosele who are gone to be with the Lord and family, whose support, prayers and encouragement have been a source of strength throughout my academic journey.

ACKNOWLEDGEMENT

I wish to express my profound gratitude to the following persons who contributed immensely in various ways towards the success of this research work and during my period in school.

Firstly, special thanks and credit to the Almighty God for His guidance, protection, wisdom, grace, understanding, strength and good health to fulfill my academic dreams.

My sincere appreciation goes to my project supervisor, Dr. (Mrs.) R. O. Osaseri, for her patience, professional guidance and constant encouragement which greatly contributed to the success of this work. I am equally grateful to the Head of Department - Dr. (Mrs.) A. R. Usiobaifo and all lecturers of the Department of Computer Science, and indeed the entire staff and students of the University of Benin, Benin City, for the knowledge and academic support given to me during my programme.

I extend my heartfelt appreciation to my late parents Mr. Benjamin Eromosele and Mrs. Bridget Ilobekemen Eromosele, my amiable wife – Mrs. Gloria Eromosele, my loving children Osezele Desmond Eromosele, Obehi Favour Eromosele, Efua Faith Eromosele, Ofure Praise Eromosele and my beloved brothers and sisters who are Mr. Emmanuel, Sister Ayo, Mr. Ojo, Mr. Lucky, Mr. Ebhozele and Gloria for their moral, financial and spiritual support.

My appreciation also goes to my friends and colleagues, especially Chief Dan Ose Okoh, S.A.N and Hon. Justice Ehinon Okoh for their support, cooperation and understanding throughout the course of my studies and this project.

I will not fail to acknowledge the various authors of textbooks, writers, project and periodicals consulted during the course of writing this project.

Finally, my humble thanks goes to everyone who contributed in one way or another to the success of this research but whose names are not mentioned, I sincerely say thank you.

ABSTRACT

Color detection is a task that humans perform effortlessly; however, enabling computers to accurately identify colors remains a challenging problem. In many industries, traditional color recognition systems rely heavily on manual processes and paid labor for color-coding items or datasets, which are often time-consuming, repetitive, and prone to human error.

To address these limitations, this project presents a **deep learning–based color detection program** capable of recognizing multiple colors in real time. The system is implemented using **Python**, a high-level general-purpose programming language, in conjunction with the **Open Source Computer Vision Library (OpenCV)**. By leveraging deep learning techniques, the proposed solution enhances accuracy and efficiency in automated color recognition tasks.

The developed system enables computer devices to detect and classify multiple colors in real time, making it suitable for applications across various industries, including pharmaceutical manufacturing, autonomous vehicle development, and robotics. The adoption of this system can significantly reduce production time, minimize reliance on manual labor, and lower operational costs while improving overall productivity.

TABLE OF CONTENTS

CHAPTER ONE
INTRODUCTION

1.1 Background of Study

Color detection is the process of identifying the name of each color. It seems very simple but on the contrary, for humans it is a very easy task, but for computers it is not easy. The human eye and brain work together to convert light into color. Light receptors in our eyes transmit signals to the brain. Our brain recognizes color. We use the same strategy to determine the name of the color.

Color recognition is the process of separating colors and identifying colors. This is one of the interactions between humans and computers. Here, color plays the role of interface between people and computers. In the first color, the modal is used for the recognition process, it only recognizes the main colors in each color image, the main colors are separated, and the color is recognized to identify the name. They are like red, green and blue. It gives the color known as output using two methods, namely printing the text on the output screen and playing the audio file (.wav) with the name of the original color. The pixels of the three colors are then counted. Whenever it exceeds 300 pixels of these three colors (RGB), it must know that the given colors have been found. And we explain that under 300 pixels, there is nothing. It could be a change of light. Although colors below 300 pixels should not be considered color. In addition, other colors are also

ignored. These are the basic methods of color recognition. In real-time systems, time signals are processed by some algorithms. For this project, the actual input signal is a continuous motion image signal, that is, a video signal. It does not have any finite duration, algorithm monitoring for every frame and processing by the given algorithm (Hu and Zhou, 2013).

In this Python color detection project, we will create an application that will allow you to automatically retrieve the color name in real time. So, for that we have a data file that contains the name of the color and its values. Then we calculate the distance of each color and find the shortest.

1.2 Statement of Problem

Computer currently require the input of human effort to help detect colors. The current system require a great amount of time especially when dealing with hundreds of colors which as a result could be monotonous. People often make mistakes, and in some cases, they can misname colors if they are color blind. There are multiple shades of a given color on the color wheel, which can lead to inconsistencies in the stored data. Our proposed system will eliminate the errors in human input. It will accurately tell colors in real-time and can be stored for future reference. It will reduce the time usually taking to manually input names of colors into a computer that exist in the current system.

1.3 Aim and Objectives

The importance of colors and how it can be a great organizational tool has been widely underestimated. The aim of this proposed system is to develop a program that will

understand how colors work. Create an app where you can instantly get the name of any color and to examine real-life application of a color detection program.

The Objectives of this study are stated as follows:

1. Review the literature concerning the existing system of a color detection program and application
2. Design the architecture of a proposed system
3. Implement the proposed system

1.4 Research Methodology

The proposed system will use a deep learning approach with tools such as:

1. Unified Modeling Language was adopted to model view of the system using:
    - Sequence diagram
    - Activity diagram
    - Work flow diagram
2. Review existing system on color detection program and a new system was proposed.
3. A real-time color detection program was designed using:
    - Python (A high-level general-purpose programming Language)
    - Visual Studio Code
    - Open-Source Computer Library (OpenCV)
    - Numpy

1.5 Significance of the Study

As we know that colors are the most important attributes to recognize and evaluate the properties of an object material. This project will examine the development of a color detection system with numerous benefits. E.g., A system that can help in image processing, digital signal processing and object identification. Use to better distinguish different shades of color. Grading of colored products determines code mark; this system will detect the data code of a package. Textile industries, automotive industries, food producing industries, printing industries, and pharmaceutical industries all use color detection application to improve the quality control in visual inspection and monitoring of color changes. In a nutshell, color detection is an essential tool in modern technological world and it will be unwise not to constantly improve on an existing system.

1.6 Scope of the Study

The scope of this system is to focus on the design and implementation of a color detection program for performance improvement in the producing industries. Such as, Food, textile industries, printing industries and pharmaceutical industries.

We will develop a basic tool that will help us identify colors and images. This program will return the RGB values of the colors, which is really useful. Most photographers and web designers will understand how useful RGB can be. Creating a color recognition system is a big task to start with in computer vision.

1.7 Limitation of the Study

In the execution of any project work, especially the one of this magnitude one is always faced with one form of constraints or the other. Some of the constraints or limitations to my work were time as well as funds.

In view of my tight academic calendar there was the problem of attending classes and working on my project concurrently but due to the grace of God that granted me good health and strength I was able to forge ahead with this project work.

Another pressing constraint was finance because of daily downloading

of materials from the internet.

Lastly, is the problem of erratic power supply which hinder the speed of this project work.

1.8 Definition of Basic Terms

Grey Conversion:

Converting color images to grayscale images. Color images contain 24 bits per pixel; reduced to 8 bits per pixel. In general, grayscale represents the amount of interval quantization in grayscale image processing. Currently, the most common storage method is 8-bit storage. Grayscale image, and the intensity of each pixel can range from 0 to 255.

Binary Conversion:

It is a process of converting any type of image into binary image. The basic binary image is a 2-bit image, it contains only 0's and 1's. Here, 1 is represented as white and 0 is represented as black, so it is called a black and white image. The purpose of the

conversion is to count the black and white pixels of the image. Each color turns white and the other colors turn black.

Subtraction:

An RGB image has 24 bits and three colors each with 8 bits per pixel. Similarly, RGB is divided into 8-bit colors each. Color subtraction is the process of subtracting the color values between two colors. Here, the three colors are subtracted, converting the gray image from the original RGB image.

Multiplication Image:

This is the process of multiplying pixel values between two or more images; Here this process is used to multiply binary images with split-color images. At the end of this process, we can get any primary color larger than 300 pixels. The area containing fewer than 300 pixels is not considered a color. After multiplication, this process has three segmented colors of 300 pixels. By combining these colors, we can get the segmented color image containing only primary colors

1.9 A Short History of Color

When most people think of color, the first thing that comes to mind is the color palette that we were shown in elementary art class. What most people don't know is the great history behind the paint and the discoveries that were important in its production and processing.

Sir Isaac Newton introduced the first color wheel in the 17th century when he discovered the different forms of visible light. At that time, color was thought to be the product of a mixture of light and dark, with red being "lightest" and blue being "darkest". Newton found this theory wrong and, when he was isolated as the bubonic plague destroyed Europe, began to test the properties of white light and "using it to test the famous color display". In his classic prism experiment, he claimed that white light has different colors. He then mapped these colors to an octave scheme similar to the original color wheel on the original ROY G BIV. His experiments also led to the discovery that all two colors can be obtained by mixing primary colors. Mixing different colors and sizes resulted in different "hues" of new colors compared to the classic ROY G BIV series and produced the original hue wheel, which may be the most common color we see.

CHAPTER 2
LITERATURE REVIEW

2.1 Review of the Study

The modern world is a huge mass of information surrounded by new digital information. In order to monitor and organize these seas that destroy visual information, image analysis techniques are the main requirements. A particularly useful world are methods that can analyze the content of an image or video automatically. The content of the image determines the importance of the quantity that can be used. An important part of image content is the content of the image. So, there is a need for object confirmation process.

Color, as a remarkable and solid characteristic of vehicles, can function a beneficial and dependable cue in a lot of programs in brilliant transportation systems. Therefore, automobile color popularity in science scenes has come to be a critical study subject matter in this area. The Primary purpose of this review is to develop a functioning model that will detect and classify colors in real time. The OpenCV computer vision library is used in this study. The basis of color recognition is the extraction of color channels and the creation of color gradients from color images. Color channels have different representations, e.g., RGB and HSV. RGB values represent the proportions of red, green, and blue in each channel, while channel values in HSV relate to hue, saturation, and value. It represents the basic elements of color, saturation represents color purity, and value represents the Brightness of a color RGB can lead to different reception results with different light intensities; Color information cannot be easily separated from

luminance. On the other hand, HSV infection can lead to changes in the properties of the human eye.

There are numerous bodies of works concerning image classification, object recognition, and learning algorithm. In this review, we focus on those on color recognition and deep learning, which are more important to our work.

2.2 Related Works

Color and their size have been highlighted in conjunction with computer vision for years. Rubin et al (2012), stated that although color is one of the most interesting and important aspects of vision, most of the colorimetric models and methods available to describe and define color were developed outside of optometry. . His research provides a summary of some of the most popular types of color and a short history of the progress that has led to our current understanding of the complexities of color.

Behic (2020), developed a software that helps identify colors in images. The built-in program also returns the RGB values of the colors, which is really useful. Many graphic designers and web designers understand how useful RGB values can be. According to them, creating a color recognition system is a big task to start with in computer vision.

Stiles et al (1982), proposed a research project that describes the concepts and strategies of color science. The RGB indicator is used here to identify the shade in the image. An

RGB display can be a shade that combines blue light with red and green in many ways to create a variety of colors.

In their report, Berns et al (2002), explained that image segmentation divides an image into regions or components. The degree of division depends on the problem to be solved. Non-trivial image segmentation is one of the most difficult tasks in image processing. The accuracy of the domain determines the success or failure of the computer analysis program.

Gonzalez (2004) in his book explains that MATLAB takes any response as a network, making it into a picture where images are used in the usual way that will talk about grayscale, RGB, HSV and other shading models. It also explores the use of MATLAB to describe methods for determining the shape and color of previously generated objects.

Also, Abadpour et al (2010), talk about the relationship between the conversion of RGB images and grayscale images and therefore black and white (two numbers) images and many others, but the study of the cream of the species does not show that - represent. In a color image system that uses fundamental analysis, color recognition involves comparing each pixel in a metric and leading to the most color-matched color of a given object.

In their article on color recognition, William et al (2014), propose a new function of color recognition, namely the extraction of colors for the purpose of human-computer interaction based on vision. Vision-based human-computer interaction can be achieved

by analyzing color regions focused on color-based image segmentation and vision-based color recognition by addressing these difficulties.

However, complex backgrounds, unfamiliar light conditions, and many moving objects make this task difficult. Following all these, Bhanot (2018) also led the way in a completely different direction. Visualization in Python has been an area of interest for this author. When he met OpenCV which allows importing and manipulating images in Python, the author began to wonder if information on these images could be extracted using Machine Learning and used in some way or another. We have seen that online searches can be done based on certain filters, one of which is color. This brought about the writer to jot down code that could extract colors from images and filter images based on those colors.

Also, in their article, Ray et al (1999), explain how to understand the basics of OpenCV, how to extract images from images using the KMeans algorithm, and how to filter images from an image collection based on RGB values of color. This opens the door to many advanced applications such as color search and search results or finding clothing items of color.

Sande et al (2010) in their paper analyzed the invariants and variables of color descriptors in a systematic way. The change characteristics of the analysis of color descriptors have been determined, using a taxonomy based on change characteristics related to photometric changes, and tested experimentally using a dataset with known lighting conditions.

Bai et al (2015), proposed a deep-learning-based algorithm for automatic car color recognition. In contrast to the conventional methods, which usually use manual design, the proposed algorithm is able to learn a representation that works well for the car color recognition task, and lead to the highest level of identification and avoid pre-processing. Furthermore, they combined the commonly used spatial pyramid scheme with the original convolutional neural network architecture, which improves its performance.

Hu et al (2018), in their paper presented a framework of active contour-based visual tracking using level set. The main components of their framework include contour-based tracking initialization, color-based contour evolution, adaptive shape-based contour evolution for non-periodic motions, dynamic shape-based contour evolution for periodic motions, and the handling of abrupt motions. For the color-based contour evolution, Markov random field theory is used to measure correlations between values of neighboring pixels for posterior probability estimation.

Chen et al (2007), presented an approach to the identification of computer graphics using statistical moments of characteristics function of wavelet sub bands and their prediction errors. They also studied two commonly used color representations, namely HSV and RGB, and found that the appropriate selection of color images is essential for generating effective designs.

Bayou (2010), present a color detection method using artificial retina that can be applied in robotics and other color detection needs.

Shi et al (1994), proposed a model for feature selection, a tracking algorithm based on a model of affine image changes, and a technique for monitoring feature during tracking. Their model is cheap and robust and helps distinguish between good and bad features based on measurements that use affine motion as the underlying image modifier.

Color detection is an area with many consumers. There are a plethora way to detect color, from physical methods to the latest machine learning techniques and even web based methods. From the above papers, one can say that deep learning algorithms are best suited for real-time color detection and recognition. From the knowledge gathered, it is clear that most of the research are implemented on autonomous driving systems and less research is being carried out to identify the use of deep learning to create updated models that detect and recognize color in real-time.

Therefore, this project will be using deep learning approach to proposed an updated model to detect color in real-time that can benefit various industries and improve future innovation.

CHAPTER 3
SYSTEM ANALYSIS AND DESIGN METHODOLOGY

System analysis is a discipline in the software development industry that seeks to provide a system for operation and maintenance, security, change and dissemination to enable the economic development of computer systems for the purposes for which they were created. (Satya, 2011).

System analysis refers to a process through which an existing system is examined (usually a business system) with the intent of improving it, or creating a new system, through better procedures and methods (Satya, 2011).

3.1 Existing System

Currently, the process in which a computer recognizes a color and stored them is carried manually. It requires a huge amount of time for manual labor to categorically input the names of a color so that it can be sorted and stored for future use. This process can be exhaustive, monotonous and tiring to carry out. And when we factor the amount of human effort required to carry out this process you realize it can impact production cost in a tremendous way. Errors due to humans not being able to properly recognize and differentiate various shades of colors is without doubt unavoidable.

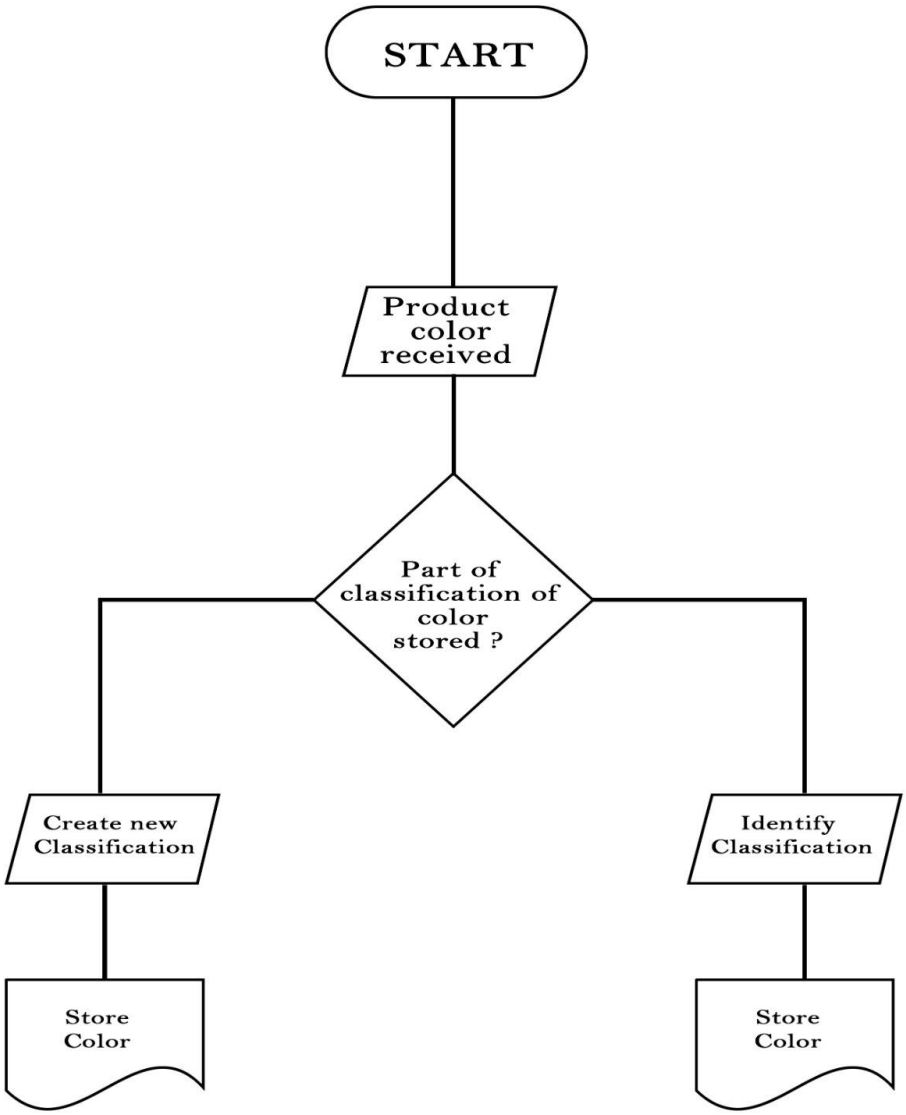


Fig. 3.1: Flowchart Showing the Operation Mode of the Existing system

3.2 Proposed System:

Color consists of 3 primary colors; red, green and blue. In computers we can set any color value between 0 to 255. So how do you define color? The answer is 256 * 256 * 256 = 16,581,375, there are about 16.5 million different colors to display. The Color detection program is design for most industries to replace their existing manual-based system which will control the following, color name, RGB color space, HSV color space all of which will contribute to enable the real-time detection of color effortless for a computer.

The proposed system provides services that cannot be easily embark on by the current manual system. Some of the reasons for the proposed system are:

- Faster production time due to faster color sorting
- Minimal errors encountered during color blindness by human effort
- Reduced cost of production reason being that the color detection program required little to no human intervention
- Vehicles can come at a halt when a traffic light shows red which is a major technological advancement for the automotive industries.
- Increase in production revenue due to cost of production has been cut down

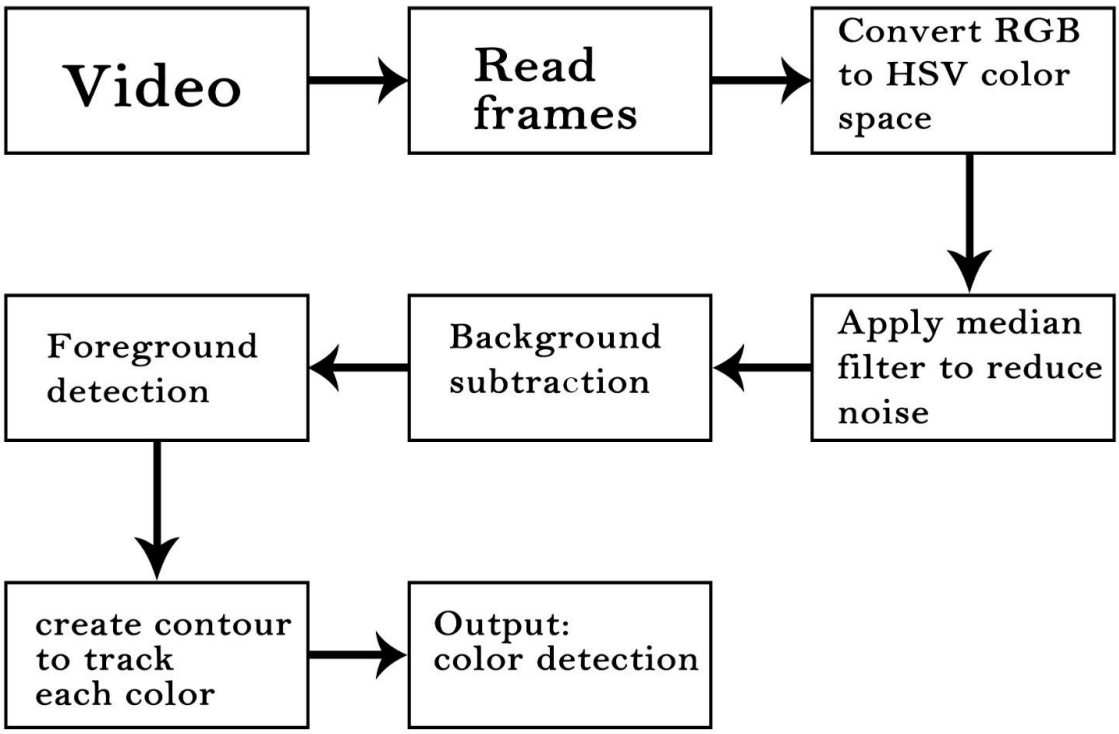


Fig. 3.2: System Architecture of the proposed color detection system

3.3 System Design

The process of establishing the elements of a system such as architecture, modules and components is known as system design. Also knowing the various interfaces of these elements, and the data that passes through the system. It aims to fulfill the needs and requirements of the company or organization by creating a system that works together and efficiently.

After analyzing and defining the software requirements, software design includes three technical activities, namely design, coding, implementation and testing necessary to develop and evaluate the software. Design is where quality and development are

encouraged. Software design is the process of translating requirements into software representations.

3.4 Sequence Diagram

Sequence diagram and an interaction diagram are called INTERACTION DIAGRAM. Interaction diagrams show conversation, which are composed of objects and their interactions, including the messages that can be sent between them.

A sequence diagram is an introduction that shows the flow of information over time. Graphically, a series chart is a table that displays data organized along the X axis and information organized by increasing time along the Y axis.

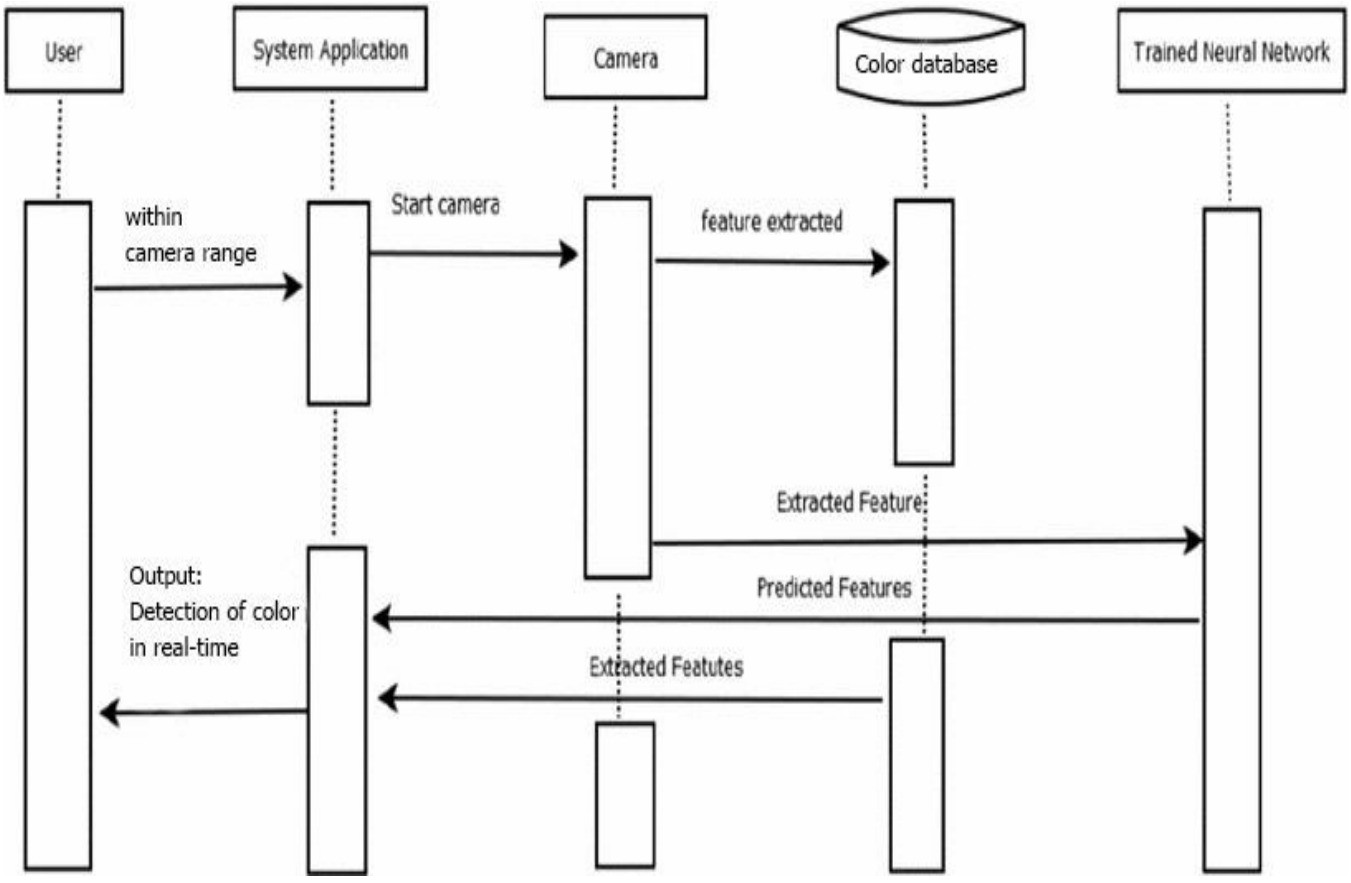


Fig. 3.3: Sequence Diagram of the proposed color detection system

3.5 Activity Diagram

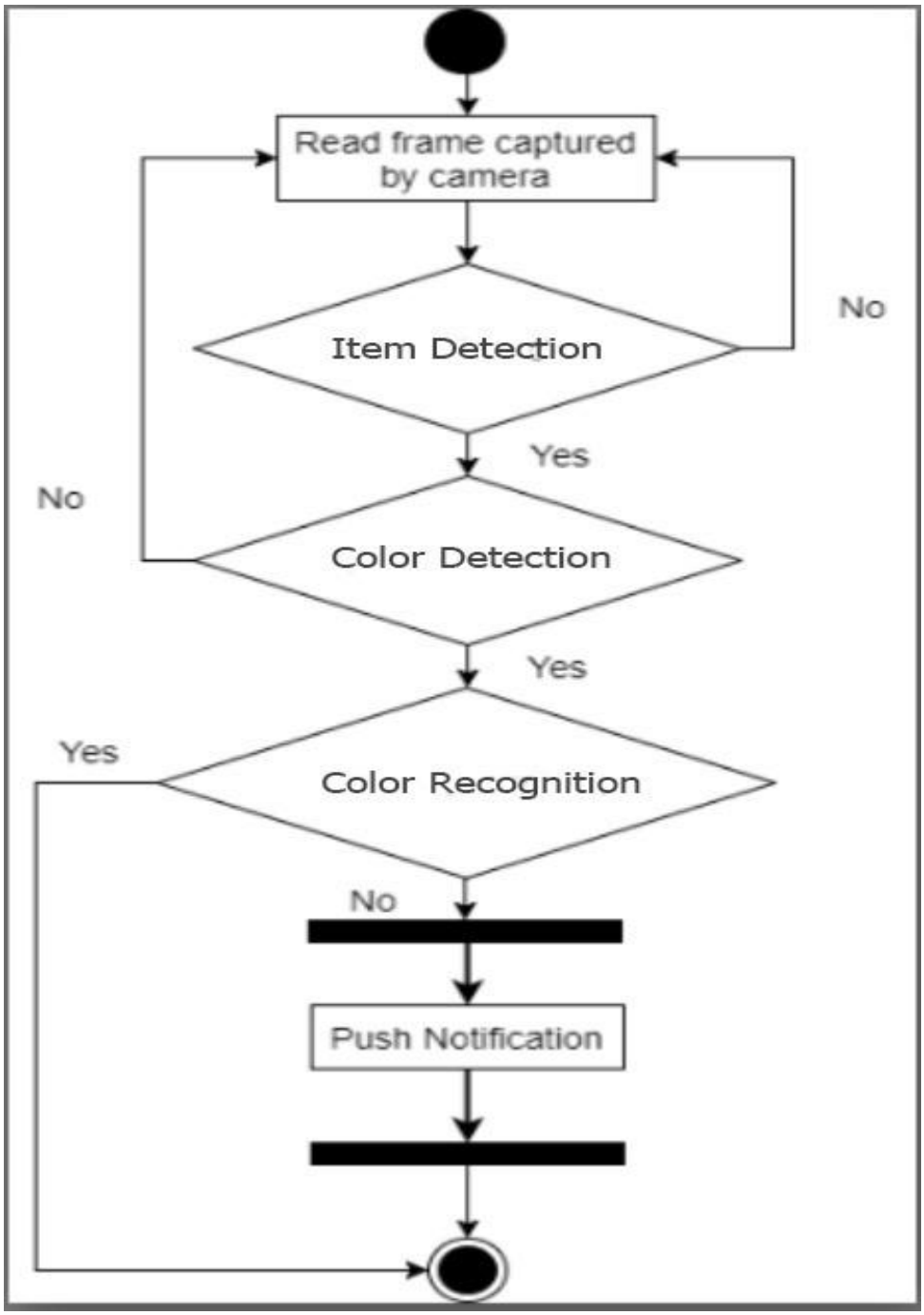


Fig. 3.4: Activity Diagram of the proposed system

3.6 Work Flow Diagram

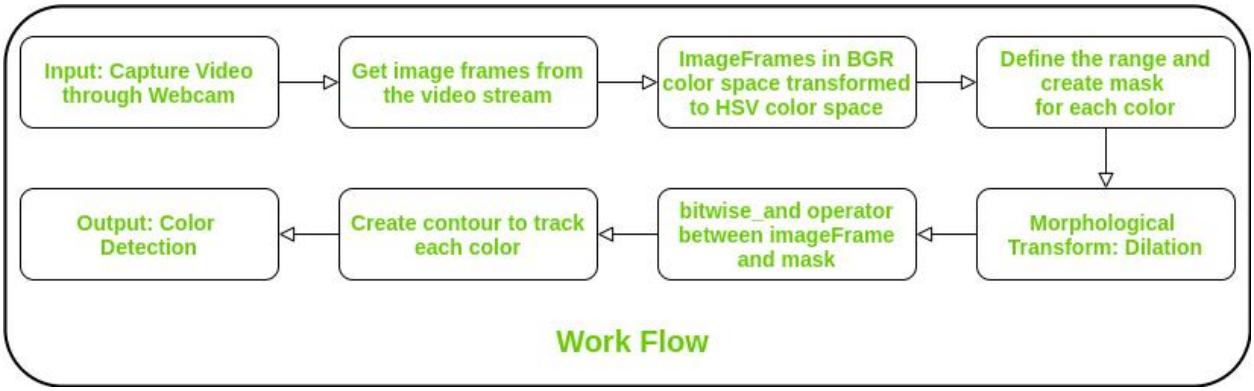


Fig. 3.5: Work Flow Diagram of the proposed system

3.7 Refined Dataset

Color	**#Hex code**	**R**	**G**	**B**
Anti-Flash White	#f2f3f4	242	243	244
Amber	#ffbf00	255	191	0
Android Green	#00308f	0	48	143
Air Force Blue	#f2f3f4	242	243	244
Baby Blue	#89cff0	137	207	240
Baby Pink	#f4c2c2	244	194	194
Bronze	#cd7f32	205	127	50
Black	#000	0	0	0
Blue	#00f	0	0	255
Ball Blue	#21abcd	33	171	205
Burgundy	#800020	128	0	32
Champagne	#fad6a5	250	214	165

Cherry	#de3163	222	49	99
Cocoa Brown	#d2691e	210	105	30
Cinnamon	#d2691e	210	105	30
Cobalt	#0047ab	0	71	171
Chocolate	#7b3f00	123	63	0
Coffee	#6f4e37	111	78	55
Cream	#fffdd0	255	253	208
Copper	#b87333	184	115	51
Corn	#fbec5d	251	236	93
Cyan	#00ffff	0	255	255
Coral	#ff7f50	255	127	80
Dark Blue	#00008b	0	0	139
Deep Green	#056608	5	102	8
Dark Cyan	#008b8b	0	139	139
Dark Gray	#a9a9a9	169	169	169
Dark Brown	#654321	101	67	33
Dark Khaki	#bdb76b	189	183	107
Deep Coffee	#704241	112	66	65
Denim	#1560bd	21	96	189
Dark Scarlet	#560319	86	3	25
Dark Lava	#483c32	72	60	50
Dark Red	#8b0000	139	0	0

Desert	#c19a6b	193	154	107
Folly	#ff004f	255	0	79
Floral White	#fffaf0	255	250	240
Ebony	#555d50	85	93	80
Ginger	#b06500	176	101	0
Glitter	#e6e8fa	230	232	250
Gray	#808080	128	128	128
Green	#1cac78	28	172	120
Grullo	#a99a86	169	154	134
Han Blue	#446ccf	68	108	207
Honey Dew	#f0fff0	240	24	250
Hansa Yellow	#e9d66b	233	255	240
Harvard Crimson	#c90016	201	0	22
Han Purple	#5218fa	82	214	107
Indigo	#6f00ff	111	0	255
Iris	#5a4fcf	90	79	207
Jade	#00a86b	0	168	107
Jet	#f8de7e	52	52	52
Kenyan Copper	#7c1c05	124	28	5
Jasmine	#343434	248	222	126
Copper Jasper	#d73b3e	215	59	62
Khaki	#c3b091	195	176	145

Lemon	#fff700	255	247	0
Light Brown	#b5651d	181	101	29
Light Cyan	#e0ffff	224	255	255
Linen	#faf0e6	250	240	230
Lilac	#c8a2c8	200	162	200
Lion	#c19a6b	193	154	107
Magenta	#f0f	255	0	255
Mahogany	#c04000	192	64	0
Mustard	#ffdb58	255	219	88
Mint	#3eb489	62	180	137
Myrtle	#21421e	33	66	30
Navy Blue	#000080	0	0	128
Neon Green	#39ff14	57	255	20
Olive	#808000	128	128	0
Orchid	#da70db	218	112	214
Orange	#ff7f00	255	127	0
Peach	#ffe5b4	255	229	180
Pink	#ffc0cb	255	192	203
Pearl	#eae0c8	234	224	200
Pear	#d1e231	209	226	49
Purple	#800080	128	0	128
Quartz	#51484f	81	72	79

Raspberry	#e30b5d	227	11	93
Red	#f00	255	0	0
Rose	#ff007f	255	0	127
Rust	#b7410e	183	65	14
Ruby	#e0115f	224	17	95
Sand	#c2b280	194	178	128
Sapphire	#0f52ba	15	82	186
Seashell	#fff5ee	255	245	238
Scarlet	#ff2400	255	36	0
Snow	#fffafa	255	250	250
Tan	#d2b48c	210	180	140
Tangerine	#f28500	242	133	0
Teal	#008080	0	128	128
Umber	#635147	99	81	71
Vanilla	#f3e5ab	243	229	171
Violet	#8f00ff	143	0	255
White	#ffff	255	255	255
Wine	#722f37	114	47	55
Yellow	#ff0	255	255	0

CHAPTER 4
SYSTEM IMPLEMENTATION

4.1 Implementation

Implementation is the stage of the project where the knowledge is changed into a work process and gives confidence in the new process for the employees to work effectively and efficiently. This includes proper planning, analysis of the current system and its implementation barriers, design of methods to achieve failure, and the initiation of failure methods. In addition to the plan, the main activity that prepares for implementation is the education and training of the employee. The more complex the process, the more systematic analysis and planning efforts are necessary for its implementation alone.

This method can only be applied after thorough testing if it is found to be working as per specification. This includes proper planning, analysis of the current system and its implementation challenges, planning of ways to achieve change, and analysis of ways to change outside of the plan. The two main functions of implementation readiness are staff education and training and process analysis. The implementation process involves several activities. Required hardware and software are available. The system may require some software development. For this, programs are written and tested. The user then switches to a well-tested system and abandons the old system.

4.2 System Implementation Technologies

For this project the following language, libraries, and software were used:

1. Python

2. Open CV
3. Numpy
4. Visual Studio Code

4.2.1 Python

Python is a high-level interpreted general-purpose programming language. Its design philosophy emphasizes code readability with meaningful input. Its structured language and object-oriented approach help programmers write clear and logical code for small and large projects.

Python is a computer programming language commonly used to create websites and software, automate tasks, and analyze data. Python is a general-purpose language, which means that it can be used to create a variety of programs but is not specialized for certain problems. This diversity, along with its startup-friendliness, has made it one of the most widely used programming languages today.

4.2.2 Open CV

Open CV is a great open-source library for computer vision, machine learning and image processing and now works great on the real-time functionality that is so important in today's systems. Using it, one can process images and videos to identify objects, faces, or even a person's handwriting. When combined with various libraries, such as NumPy, python is able to program the Open CV configuration for estimation. To identify the image pattern and its various features, we use vector space and perform mathematical operations on these features.

4.2.3 Numpy

NumPy is the basic scientific computing package in Python. It's a Python library that provides a multidimensional array object, several derived objects (like arrays and masked arrays), and a variety of routines for quick operations on arrays, including math, logic, shape manipulation, sorting, selection, I/O, discrete Fourier transforms basic linear algebra, basic statistical operations, random simulation and much more.

NumPy completely helps an object-orientated approach, starting, as soon as again, with ndarray. For example, an ndarray is a class, owning several techniques and attributes. Many of its techniques are reflected via way of means of capabilities within side the outermost NumPy namespace, permitting the programmer to code in whichever paradigm they prefer.

4.2.4 Visual Studio Code (VS Code)

Visual Studio Code is a code editor in simple terms. Visual Studio Code is "a free editor that helps the programmer write code, helps with debugging, and fixes code using intelli-sense techniques." In simple terms, this allows users to write the code more easily.

4.3 Python Vs Other Programming Languages for Computer Vision

Color recognition is a domain specific variation of the machine learning prediction problem. OpenCV itself is implemented in C and C++. But it can be easily implemented with different programming languages and environments like Python, MATLAB, Java, R., etc.

Advantages of using Python as fundamental language for Object Detection are (Guo and Nguyen, 2012):

- Extensive Support Libraries
- Open Source
- Third Party Modules present
- User Friendly Data Structures
- More Choices in Graphic Packages and Toolsets
- Productivity & Speed

4.4 Steps to Identify colors in images

Steps to creating a python program that can detect colors:

Step 1:

Input: Take a video from a webcam.

Step 2:

Play video streams and photo frames.

Step 3:

Convert background image frame (RGB color space of three matrices of red, green and blue with integer values from 0 to 255) to HSV (hue-saturation-value) color space. Hue describes color in terms of saturation, representing the amount of gray in that color.

Saturation refers to the intensity of color in an image. The color appears purer as the saturation increases, the color appears purer. As saturation decreases, colors appear washed out or pale.

Value describes the brightness or depth of the color. This may be represented as 3 matrices with inside the variety of 0-179, 0-255, and 0-255 respectively.

Step 4:

Define the different type of each color and create the corresponding mask.

Step 5:

Morphological changes: Dilation, remove noise from the image

Step 6:

Bitwise and among the photo body and masks is achieved to specially detect that specific color and discard others.

Step 7:

Create an index for each color to show the contrast of the detected color region.

Step 8:

Output: Detection of the Color in real-time.

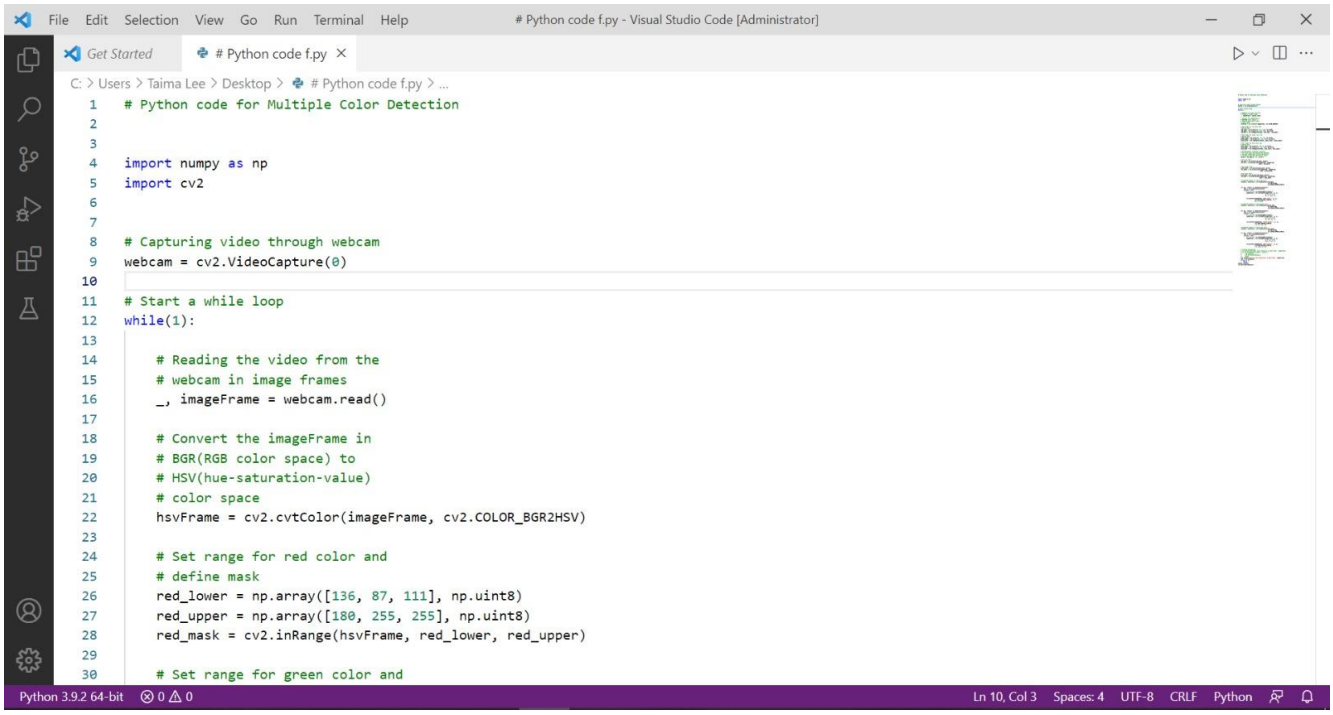


Fig. 4.1 Screenshot 1 of Visual Studio Code of the program code

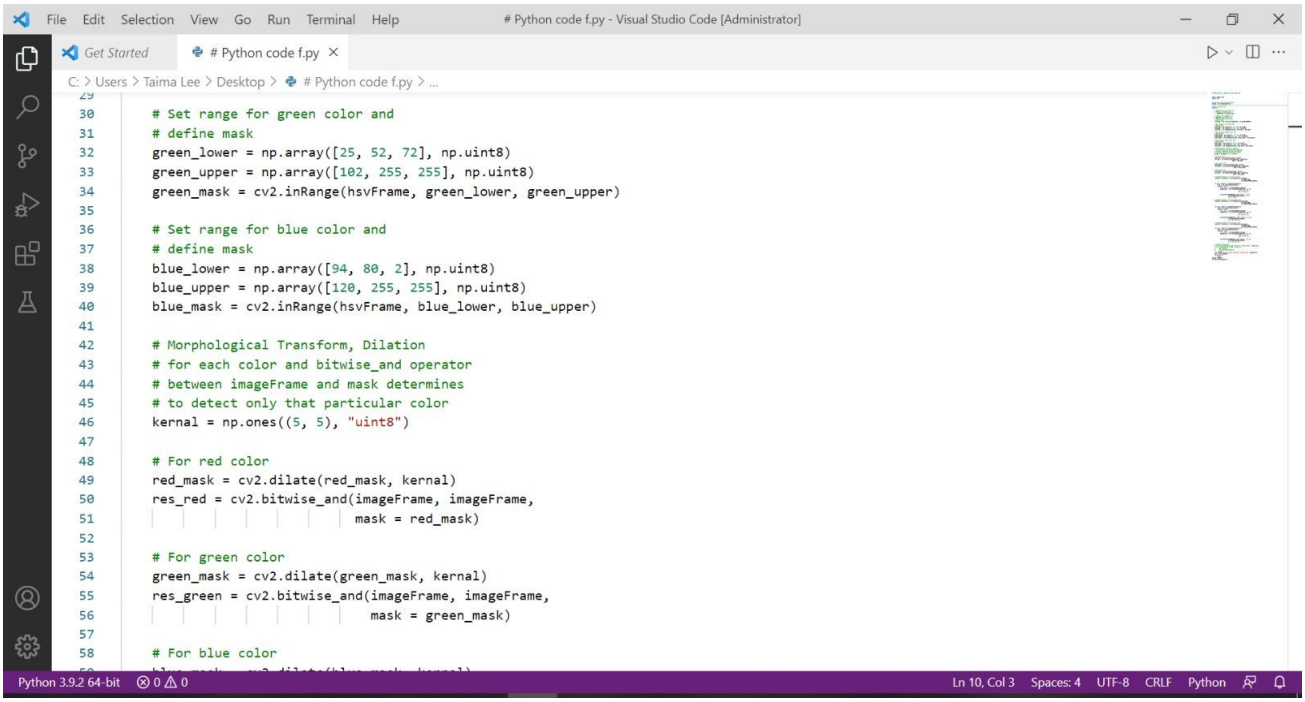


Fig. 4.2 Screenshot 2 of Visual Studio Code of the program code

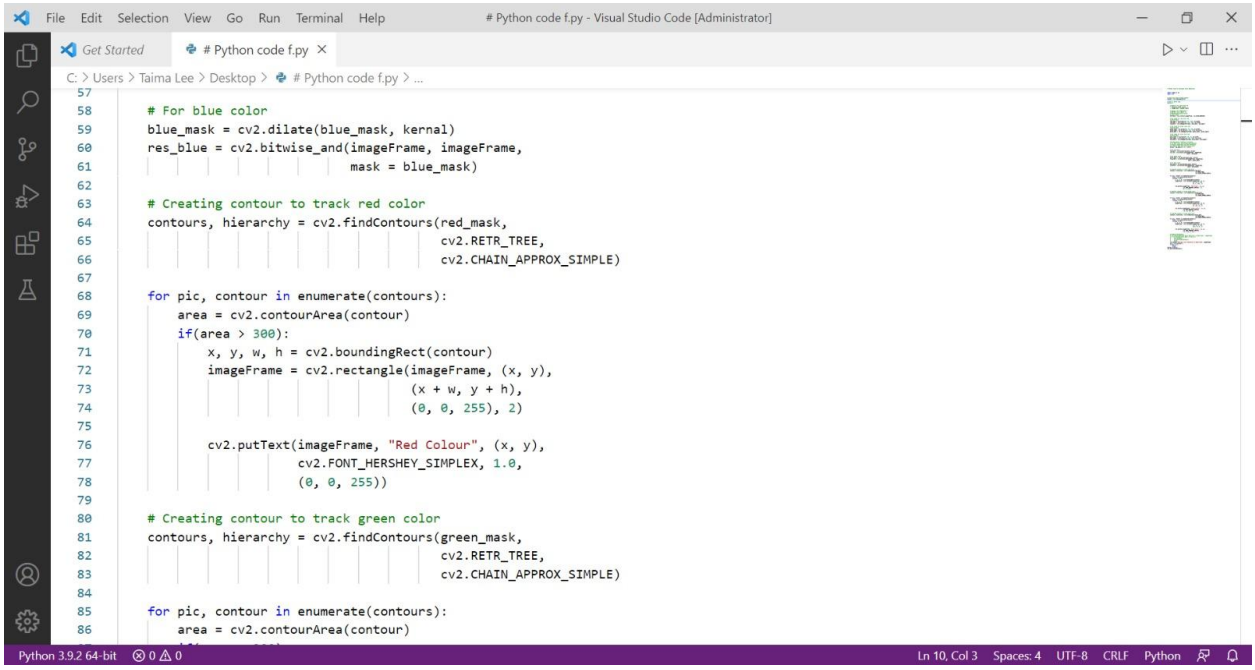


Fig. 4.3 Screenshot 3 of Visual Studio Code of the program code

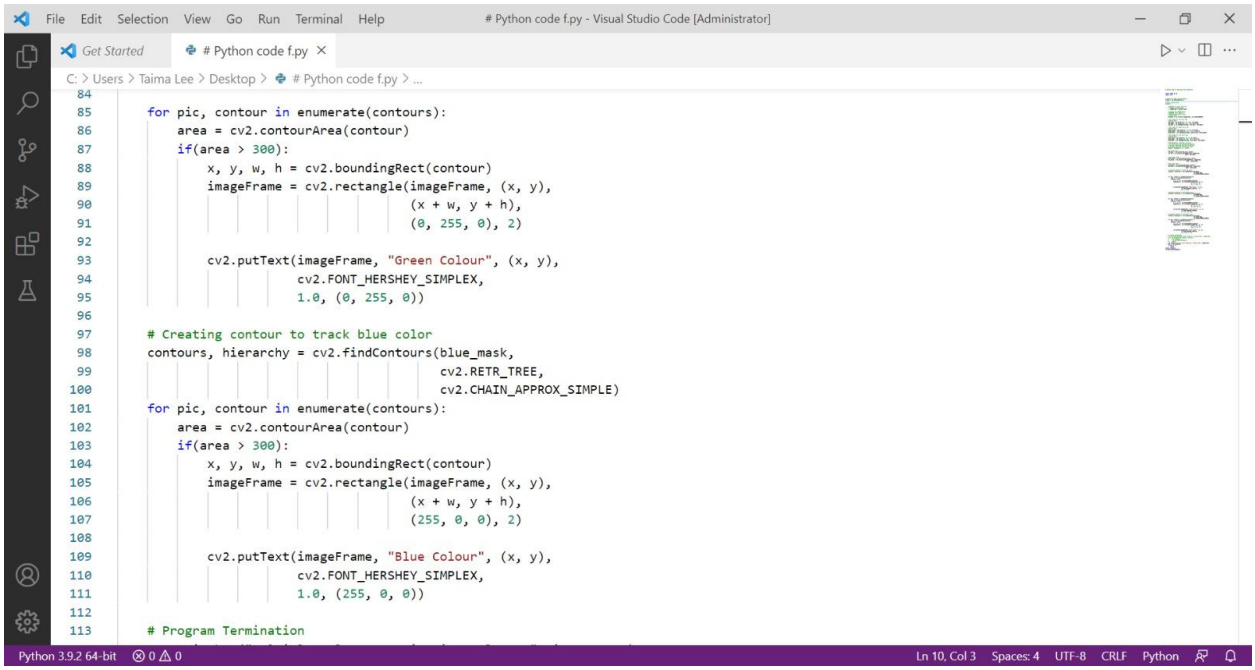


Fig. 4.4 Screenshot 4 of Visual Studio Code of the program code

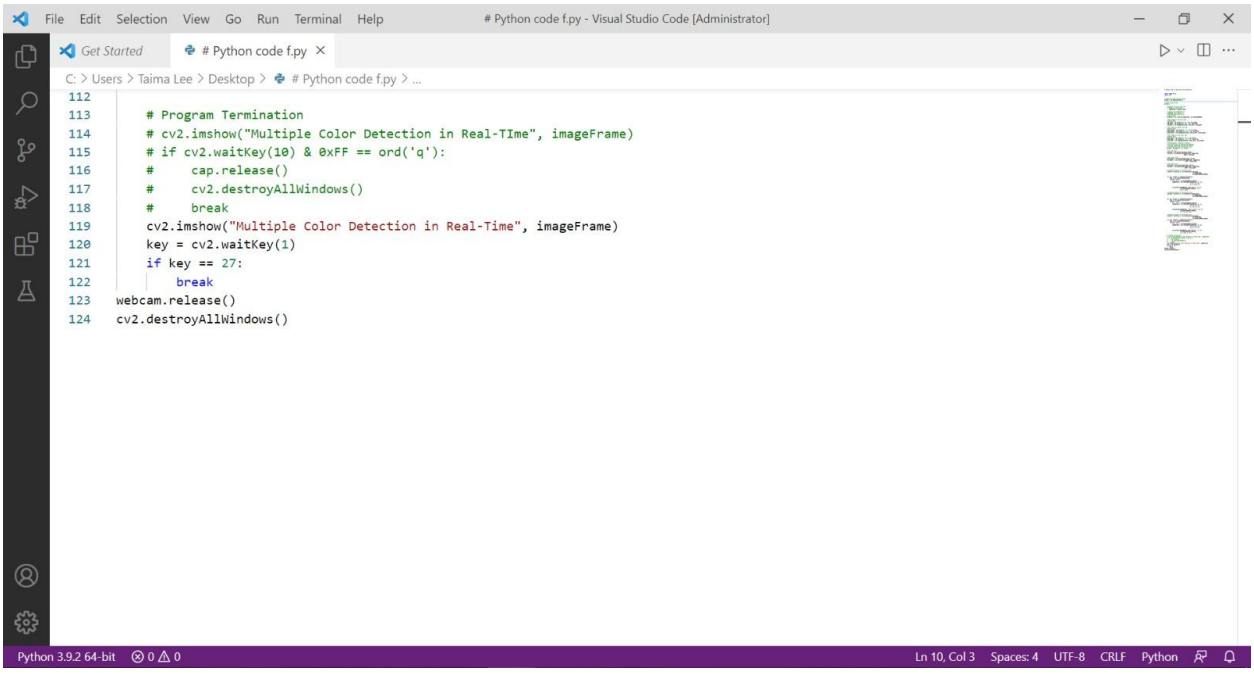


Fig. 4.5 Screenshot 5 of Visual Studio Code of the program code

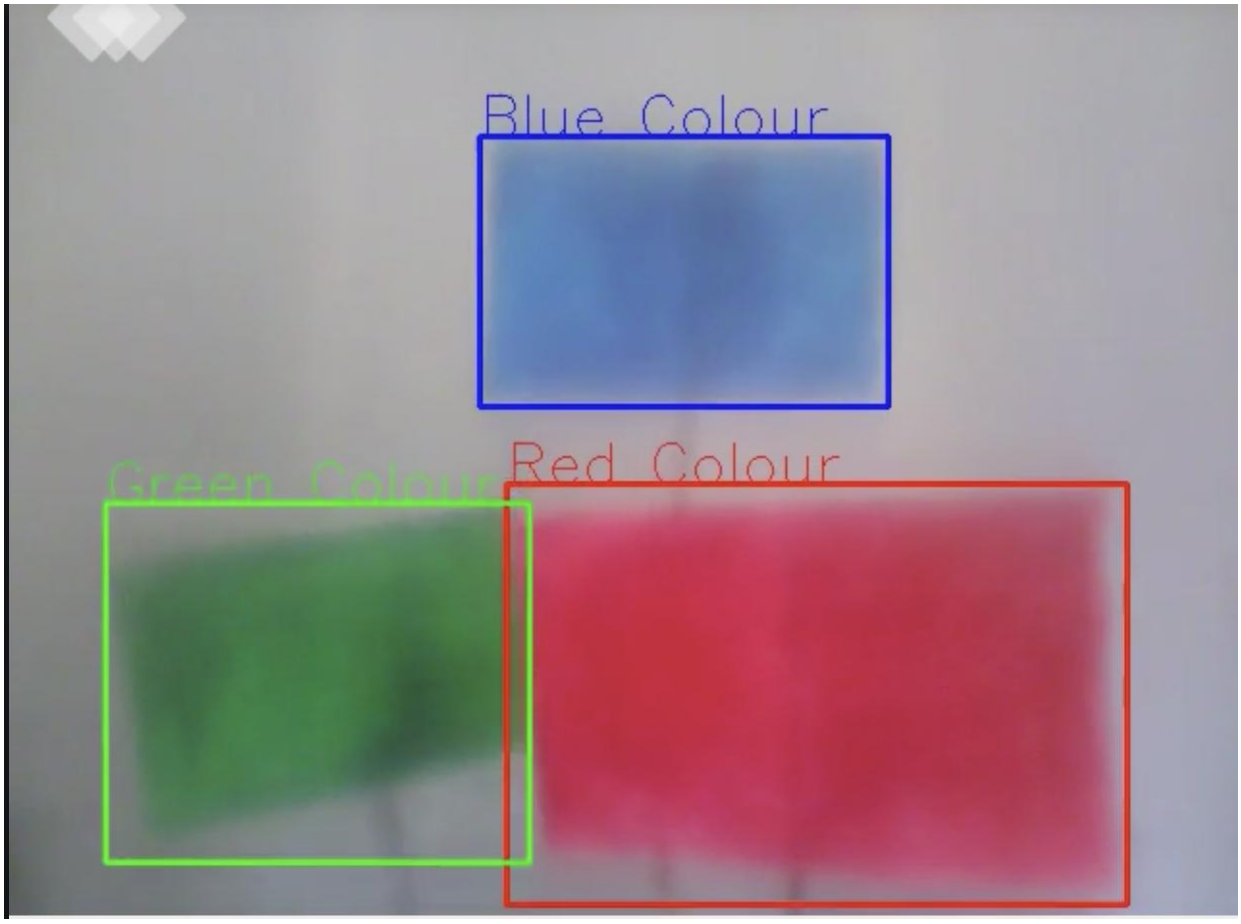


Fig. 4.6 A Running Color detection program

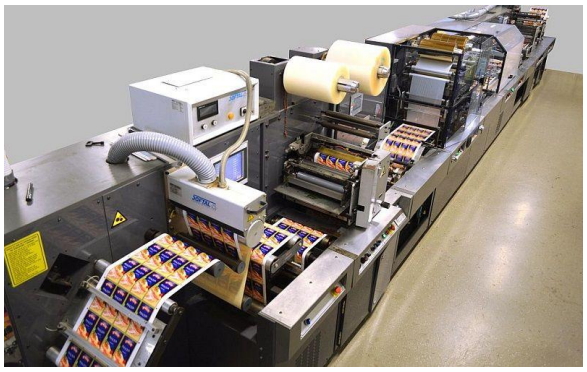
4.5 Application and Future Scope

Computer vision has not yet reached a stage where it can be used to solve life's problems, as it is still in its early stages of development. In the years that pass with the strong speed of research, computer vision or to be more precise, color detection will be completely everywhere. Computer vision is a part of machine learning. Some common and widely used application of color detection are (Guo, and Nguyen, 2012):

Printing and Packaging

One application of color detection program is in printing and packaging. In high-speed environments where tens of thousands of labels and packaging elements are running along lines, manufacturers and printers need to be able to verify the presence of registration and other color marks and ensure they meet a specific value. This supports not only quality control for printing but also overall process control – allowing production managers to rapidly identify potential issues with production and take action when needed.

Designed to support printers and packaging organizations, the color detection program is capable of differentiating between backgrounds and color marks during high-speed processes. Its lightning-fast 40 kHz switching speed and 25-microsecond discrete output response ensure that color marks are rapidly identified and outputs to PLCs or other control systems are equally rapid.



Small Part Color Verification

Another application of color detection program is small part color verification. Whether for automotive, consumer products, packaging, or electronics, there are infinite small

parts being produced every day that have to meet specific color values. This might be for identification purpose, branding, or simply matching parts to other components. In these applications, the ideal color mark sensor will verify that the parts have been produced according to the proper color specifications. If there are any issues, the fast discrete output can trigger an appropriate corrective action for a production team or another system.

Self-Driving Car

Self-driving vehicles have inbuilt color detection sensors to be able to detect when traffic light change color so they can respond accordingly. When the car detects a red light, the vehicle comes to a halt and when it goes green, it continues moving.

Face Recognition

Have you ever thought about how the iPhone X recognizes your face when you unlock it? It not only recognizes but also remembers the face for the next time the phone is unlocked. This is one of the simple face recognition applications that we see in our daily life.

Accounting Number of Objects

Computer vision is also used for counting purposes. For example, when you upload a photo to Facebook, it shows/counts the number of people in the photo.

Automobile Spotting

When the object to be discovered is a car, it is a car defect. There is moving traffic tracking. It is used on highways and traffic lights to detect vehicle license plates that violate traffic laws.

CHAPTER 5
SUMMARY, CONCLUSION AND RECOMMENDATION

5.1 SUMMARY

The Summary of this project work can be outlined as follows:

In chapter one, the project work was introduced and a background study was carried out then we went ahead to look at the definition of some basic terms relating to color detection.

In chapter two, the review of the study was examined followed by a brief history of color and we looked at different types of color on the basis of composition and effect.

In chapter three of this project work, analysis of the color detection methodology was carried out. Program description, flow chart and steps for detecting color in an image were also discussed and analyzed in this chapter.

In chapter four, a picture that depicts the code in action was shown here and the actual program source code was also shown then the real-life application of color detection program was also discussed.

Chapter five provides the summary and conclusion.

5.2 CONCLUSION

Color detection software can be useful in the most interesting and sophisticated ways. All the fundamentals concerning the detection method together with special methods to attain

it had been profoundly discussed. During the path of programming, we are able to use each Python and MATLAB for color detection software, however, we decide on python as it takes much less simulation time than MATLAB. Someone having previous coding enjoy unearths it clean to implement. Contours, shape % colorings have been detected withinside the given pattern pics successfully.

5.3 RECOMMENDATIONS

After my research and my finalization of this project, I highly recommend that a color detection program is utilized properly in the industrial market, from a self-driving car to a printing and packaging company, the potentials is enormous especially in this computer age where humans are embracing diverse technologies and everything is becoming autonomous.

REFERENCES

1. Reetu Awasthiand Khusboo Khurana (2013): Techniques for Object Recognition in Images and Multi-Object Detection.
2. http://en.wikipedia.org/wiki/artificial_intelligence: Accessed; December 2021
3. http://en.wikipedia.org/wiki/computer_vision: Accessed; December 2021
4. http://en.wikipedia.org/wiki/opencv: Accessed; January 2022
5. Jing-Ming Guo and Hoang-Son (2012): Improved Hand Tracking System. Vol. 22, No. 5, May 2012.
6. Jianbo Shi and Carlo Tomasi (1994): Good Features to Track. IEEE Conference on Computer Vision and Pattern Recognition. June 1994.
7. Chen, W., Shi, Y.Q and Xuan, G (2007): Identifying Computer graphics using HSV color model and Statistical moment of Characteristics Functions. IEEE International Conference on Multimedia and Expo.
8. Weiming Hu 1, Xue Zhou, Wei Li, Wenhan Luo, Xiaoqin Zhang, Stephen Maybank (2013): Active contour-based visual tracking by integrating colors, shapes, and motions. IEEE Transactions on Image Processing, Vol. 22, NO. 5. May 2013.

9. Chuanping Hu; Xiang Bai; Li Qi; Pan Chen; Gengjian Xue; Lin Mei (2015): Vehicle Color Recognition with Spatial Pyramid Deep Learning. IEEE Transactions on Intelligent Transportation Systems. October 2015.
10. Koen van de Sande; Theo Gevers; Cees Snoek (2010): Evaluating Color Descriptors for Object and Scene Recognition. September 2010.
11. Israel Yohannes Bayou (2010): Color detection using Artificial Retina.
12. Pietro Zanuttigh, Carlo Dal Mutto, Guido M. Cortellazzo (2012): Fusion of Geometry and Color Information for Scene Segmentation. Journal of Selected Topics in Signal Processing. Vol. 6, No. 5. September 2012.
13. Claudia Nieuwenhuis, Daniel Cremers (2013): Spatially Varying Color Distributions for Interactive Multi Label Segmentation. Vol. 35, No. 5, May 2013.
14. Koen E.A Van de Sande, Theo Gevers, Cees G.M. Snoek (2011): Evaluating Color Descriptors for Object and Scene Recognition. Vol.32, No. 9, September 2011
15. https://www.youtube.com/watch?v=6Otgyyv--UU: Accessed; January 2022
16. https://data-flair.training/blogs/project-in-python-colour-detection: Accessed; January 2022
17. https://www.youtube.com/watch?v=nty0zSKB4_k: Accessed; January 2022
18. https://programmingdesignsystems.com/color/a-short-history-of-color-theory/: Accessed; January 2022

19. https://docs.opencv.org/4.x/d0/de3/tutorial_py_intro.html: Accessed January 2022

20. https://en.wikipedia.org/wiki/Python_(programming_language): Accessed January 2022

21. https://numpy.org/doc/stable/user/whatisnumpy.html: Accessed January 2022

22. https://www.educative.io/edpresso/what-is-visual-studio-code: Accessed January 2022

APPENDIX

```
# Python Program for Detecting Multiple Color

import cv2
import numpy as npy

# Take a video from a web camera
webcamera = cv2.VideoCapture(0)

# creating while loop
while(1):

 # Read video through webcameras in image frames
_, image_in_Frame = webcamera.read()

# Converting image_in_frame in BGR to HSV color space
# BGR (color space in RGB (RED GREEN AND BLUE))
# HSV (color space in HUE, SATURATION AND VALUE)

Frame_in_HSV = cv2.cvtColor(image_in_Frame, cv2.COLOR_BGR2HSV)

# Color red range setting and mask defining
lower_red = npy.array([136, 87, 111], npy.uint8)
upper_red = npy.array([180, 255, 255], npy.uint8)
mask_red = cv2.inRange(Frame_in_HSV, lower_red, upper_red)

# Color green range setting and mask defining
lower_green = npy.array([25, 52, 72], npy.uint8)
upper_green = npy.array([102, 255, 255], npy.uint8)
mask_green = cv2.inRange(Frame_in_HSV, lower_green, upper_green)

# Color blue range setting and mask defining
lower_blue = npy.array([94, 80, 2], npy.uint8)
upper_blue = npy.array([120, 255, 255], npy.uint8)
mask_blue = cv2.inRange(Frame_in_HSV, lower_blue, upper_blue)

# Morphological changes, bitwise_and operator and
# each color dilation between mask and imageframe
# to help detect individual colors
```

```
kernal = npy.ones((5, 5), "uint8")

# Color red
mask_red = cv2.dilate(mask_red, kernal)
reso_red = cv2.bitwise_and (image_in_Frame, image_in_Frame,
                    mask = mask_red)

# Color green
mask_green = cv2.dilate(mask_green, kernal)
reso_green = cv2.bitwise_and (image_in_Frame, image_in_Frame,
                     mask = mask_green)

# Color blue
mask_blue = cv2.dilate (mask_blue, kernal)
reso_blue = cv2.bitwise_and (image_in_Frame, image_in_Frame,
                    mask = mask_blue)

# Tracking red color with contour
contours, hierarchy = cv2.findContours(mask_red,
                                cv2.RETR_TREE,
                                cv2.CHAIN_APPROX_SIMPLE)

for pics, contour in enumerate(contours):
   area = cv2.contourArea(contour)
   if (area > 300):
      x, y, w, h = cv2.boundingRect(contour)
      image_in_Frame = cv2.rectangle(image_in_Frame, (x, y),
                           (x + w, y + h),
                           (0, 0, 255), 2)

      cv2.putText(image_in_Frame, "Red Colour", (x, y),
               cv2.FONT_HERSHEY_SIMPLEX, 1.0,
               (0, 0, 255))

# Tracking green color with contour
contours, hierarchy = cv2.findContours (mask_green,
                                cv2.RETR_TREE,
                                cv2.CHAIN_APPROX_SIMPLE)

for pics, contour in enumerate(contours):
```

```
        area = cv2.contourArea(contour)
        if (area > 300):
            x, y, w, h = cv2.boundingRect(contour)
            image_in_Frame = cv2.rectangle (image_in_Frame, (x, y),
                                (x + w, y + h),
                                (0, 255, 0), 2)

            cv2.putText (image_in_Frame, "Green Colour", (x, y),
                        cv2.FONT_HERSHEY_SIMPLEX,
                        1.0, (0, 255, 0))

    # Tracking blue color with contour
    contours, hierarchy = cv2.findContours (mask_blue,
                                    cv2.RETR_TREE,
                                    cv2.CHAIN_APPROX_SIMPLE)
    for pics, contour in enumerate (contours):
        area = cv2.contourArea (contour)
        if (area > 300):
            x, y, w, h = cv2.boundingRect(contour)
            image_in_Frame = cv2.rectangle (image_in_Frame, (x, y),
                                (x + w, y + h),
                                (255, 0, 0), 2)

            cv2.putText(image_in_Frame, "Blue Colour", (x, y),
                        cv2.FONT_HERSHEY_SIMPLEX,
                        1.0, (255, 0, 0))

    # Terminating Program
    cv2.imshow ("Real-Time Detection of Multiple Color", image_in_Frame)
    if cv2.waitKey(10) & 0xFF == ord('q'):
        webcamera.release()
        cv2.destroyAllWindows()
        break

    # Terminating Program
    cv2.imshow ("Real-Time Detection of Multiple Color", image_in_Frame)
    key = cv2.waitKey(1)
    if key == 27:
        break
webcamera.release()
cv2.destroyAllWindows()
```