Computer Science Capstone: Computer Vision

and Color Identification

Craig Shaffer

April 25, 2023

**Program Outline: What does it do?**

Color blindness affects roughly 4.5% of the world’s population. To put that figure into perspective, the number of people affected - over 350 million - is greater than the United States population. Though many people have color blindness, the number of tools to assist those with a color vision deficiency is surprisingly low and the tools that do exist are often expensive. This project aims to make color identification more accessible via a python program that uses computer vision to accurately identify colors. The project does this in two ways.

The first function of the program is color segmentation. This is accomplished by accessing the device’s camera and converting the Red, Green, and Blue (RGB) values of the pixels on the screen to Hue, Saturation, and Value (HSV) values. Then, using the HSV values, the program can identify what basic color each pixel falls into based on the values it is receiving. After determining this, the program will label and draw an outline around objects of the same color. This would allow people with color vision deficiencies to identify the basic colors of objects. While this is useful, it doesn’t allow you to identify exact colors of objects seen in the camera.

The second function of the program is color identification and matching. The program will find the closest color match to the color of an object you click on. This is accomplished by getting the position of the mouse cursor when left clicked. It then takes the RGB values of the pixel at that position and looks for the closest match in a 800 color csv file. The closest match is found by searching for the minimum total difference between RGB values. The color with the minimum difference is then displayed for the user to view. Both functions can be used in tandem for the most accurate color identifying experience.

Overall, the program achieves the goal of being able to differentiate and identify colors. However, the functionality is not where it beats other tools. The area where it gains an advantage is cost. The tools required to build and run this project cost nothing. The only requirement is that the computer running the program has a camera. Through the use of this tool, people with limited color vision will be able to access fields that require perfect color vision. Ultimately, the program is functional, inexpensive, and positively impacts the lives of people with color blindness.

**Descriptions of Functions and Inputs**

color\_cont(frame, point)

The color\_cont() function takes the frame and point as inputs. The frame is the defined viewing area of the tool. The point input is a list with the start and end HSV values of a color (these value ranges are defined in the “colors” dictionary in the ColorTool.py file). The function takes these inputs and returns contour points and their positions around the shape of a colored object. However, it will only return the contours if the shape’s area is larger than 6000 pixels. These points are used to outline colored objects in the color segmentation portion of the code.

on\_mouse(event, x, y, flags, params)

The on\_mouse() function takes event, x, and y as inputs. The event input is defined as the mouse action taken. OpenCV allows you to use built in listener functions like EVENT\_RBUTTONDOWN and EVENT\_LBUTTONDOWN to check for mouse clicks that are used for the event input. The x and y inputs are the coordinates of the mouse click. The flags and params inputs are added because mouse clicks return these additional inputs and must be incorporated into the function. The on\_mouse() function then takes these inputs to globally alter the values of the l\_click, r\_click, and click\_pos. If l\_click is True and you left click, then it changes the value of l\_click to False and records the location of the click into the click\_pos variable. The opposite is also true. Right clicking the mouse works the same way. However, since the position of the mouse is not important for the procedure that right clicking the mouse triggers, it is not necessary to update the click position or click\_pos. This allows the user to toggle between the color identification and color segmentation functionality of the tool.

get\_match\_name(r, g, b)

The get\_match\_name() function takes r, g, and b as inputs. The r, g, and b inputs are the red, green, and blue values of a pixel. The function will then compute the distance between the RGB values of the pixel and the RGB value of each color in the colors.csv data file. The function returns the name of the closest color matched based on minimum distance.

get\_match\_rgb(r, g, b)

The get\_match\_name() function takes r, g, and b as inputs. The r, g, and b inputs are the red, green, and blue values of a pixel. The function will then compute the distance between the RGB values of the pixel and the RGB value of each color in the colors.csv data file. The function returns the RGB values of the closest color matched based on minimum distance.

get\_match\_group(r, g, b)

The get\_match\_name() function takes r, g, and b as inputs. The r, g, and b inputs are the red, green, and blue values of a pixel. The function will then convert the RGB values into HSV and find the color group it falls into based on the start and end HSV values of colors defined in the “colors” dictionary in the ColorTool.py file. The function will then return the name of the color group that the given color falls into.

**Folder/File Organization**

In the folder for this project, you will find two files. The file names and their descriptions are displayed below:

|  |  |
| --- | --- |
| File | Description |
| ColorTool.py | Main file containing the executable code. |
| colors.csv | A csv file containing ~800 colors and their hex codes. |

**Functionality and User Design**

The functionality of the code is toggled through clicks and button presses. After running the ColorTool.py file, the program will display the instructions for the program in the top left corner. These instructions tell the user how to use the tool. A left click toggles the color identification tool. This allows the user to find the closest match to the color at the location of their mouse click. If the user left clicks again, the tool will stop looking for matches and display the instructions again. A right click will toggle the color segmentation tool and allow the user to identify the colors of large objects seen by the web cam. If the user right clicks again, the color segmentation tool is toggled off and the instructions are displayed again. An additional feature is the ability to pause the recording with the space bar. This allows the user to freeze what the webcam is seeing and examine the output on the screen. The user can resume the program by pressing any button. To exit the program, the user will press the escape button. This will close any window opened by the program.