Electrical Engineering 332: Computer Vision

Final Project Report

Matthew Schilling

Personal Views on the Field of Computer Vision and its Applications

The field of computer vision is very interesting, and working in it is a lot more fun than I had thought it would be. Initially I had wondered whether I would find this class interesting, however as we learned about the clever ways that people can think about sequences of numbers stored in a matrix/tensor (depending on whether an image is grayscale or color) and relate it back to things that happen in the real world way extremely interesting and fun. For example, the idea of using a gradient in order to find edges, or a histogram in order to smooth colors, was extremely satisfying to learn about. It was great to begin to think about how I would approach the same problem during class time when you would stop and let us think about it. During homework, I would find myself ecstatic every time I managed to get one of the algorithms to work.

In general, the field of computer vision is extremely interesting to me. As I have already said it is extremely cool to think of images as mathematical puzzles, but I also think that what we could potentially do in this field is incredible. A lot of the time, I did not even necessarily feel like I was interacting with the world, it felt almost as though I could apply the same skills to graphics work.

One of my favorite hobbies is anime, and at times I wondered if similar techniques are not used in the animation industry to process their images and allow for better animation. An obvious case is when we learned about paneling, but the same ideas could be applied in other ways. For example, to create a second image in a sequence, you might first identify the edges in an image, then slightly shift the edges that are in motion, before recoloring the portion of the image in the neighborhood of where the edges moved. Alternatively, you could also attempt to do something like overlay the animation on top of a sequence that you have stunt doubles do. This would require image tracking similar to what was implemented in the seventh machine problem, with mapping of the animated pieces on top of the templates, in order to produce more realistic fight scenes for example. Of course, I could be entirely off-base, but it was certainly an interesting thought I had had during the period where I was implementing those particular algorithms.

Some of the other incredible applications I can see for computer vision include Nvidia's robotic sous-chef, self-driving cars, and part of the processing for artificial intelligence. These are all applications that are in the process of being developed right now that could potentially change the world, and I would look forward to each coming to fruition. In some of these, I even hope that I am able to help bring them about.

The first application I had mentioned, Nvidia's robotic sous-chef, is a robotic arm that is designed to help automate the kitchen. The automation of the kitchen is one of my personal life goals, simply because I believe no one should have to work an eight hour day and then come home and have to do even more work in order to prepare food. As for how computer vision comes into play, the computer has to identify things like ingredients to be sliced, and differentiate it from, say, a pet that got on the counter, in order to prevent butchering the wrong thing. At present, the "kitchen manipulator" is capable of identifying ingredients, track objects, and even cook some meals. At present however, it seems that the robot mostly takes advantage of machine learning, which is a shame because some things like edge detection could probably be done far more efficiently using say, Canny Edge Detection.

Self-driving cars are one of the most obvious applications of computer vision, however even those are not terribly advanced. Currently, out of the six levels of automation for cars, the highest commercially available level is three, and only in Germany at the moment (the Audi A8L). At this level, the car is able to do a number of tasks in specific conditions without any assistance, but not many. In Audi’s case, it is mostly navigation of traffic jams. Though some level four vehicles do exist, they exist primarily within limited areas and are only available for things like ridesharing. Of course, this is largely due to regulatory and infrastructural barriers, as well as cybersecurity being an issue, not necessarily that the algorithms do not exist to parse real-world traffic and react to it.

The last application I would like to talk about is the use of computer vision in artificial intelligence. At present, humans are hard at work to create intelligent entities that are capable of interacting with both us and the world at large. One of the things we would like, when we create those entities, is for them to be able to see the way we do. By that I mean we would like them to be able to do things like recognize our faces to call us by name, or tell that there are dogs present, or be able to talk about the beauty of a painting. Though that last one is definitely very subjective and far-off, the first two are tasks that involve computer vision, the eigenvector approach to such recognition probably being more efficient that simply training the artificial intelligence with machine learning.Allowing our artificial friends to see and process the world will be an important step towards creating true artificial life, a goal that I think is very much admirable as a way to see to it that we are not lonely. We are already seeing this sort of thing with Google Nest’s Home Hub or Gatebox (an anime version), that sort of development will only continue in the coming decades.

To recap, I think that overall the field of computer vision is extremely fun to delve into, and its applications are incredible. I look forward to seeing where the field will end up in the next few decades.

Project Description

The project I chose to do involves sorting through photos in a directory to find photos that match a particular color that the user provides. At that point, the images are both displayed and have their filename printed. This project was inspired by both a project to identify colors and the project idea for a searchable image database. Theoretically, the same engine could be applied to different directories of images, although for this particular project I did supply the directory.

I did limit the scope of this project a little bit, in that the only searchable colors are ‘white’, ‘black’, ‘red’, ‘blue’, ‘green’, ‘teal’, ‘magenta’, and ‘yellow’. Further, at this moment, it seems the HEIC format is not able to be supported, however .jpg, .png, .bmp, and .jpeg are all valid image formats. Also worth noting is that the way I implemented the project, there need not be an overarching naming convention, just as long as the file fits into one of the supported formats, however I will discuss that a little more later on.

In the directory, there are twenty-five different images. Some of them were pulled from the class, most of them were pulled from my Google Photos account. Most of them are also memes stolen off of the website iFunny a long time ago, and do bear the watermark. As such, keep in mind that searching for ‘yellow’ in the directory will turn up most of the images due to the iFunny watermark at the bottom of each meme. As a bonus point, there is a small joke in the form of three images, the images being Billy Joel, Alexandria Ocasio-Cortez, and Elon Musk, which I challenge you to figure out. The answer will be the first sentence in the *Remarks and Future Work* section of this report.

Project Design

For this project, I made six separate functions. The functions are pixelColorID, colorID, searchForColor, showImages, findAndShow, and findAndShowV2. The core of the program is pixelColorID and colorID, the rest of the functions build on this functionality or provide extraneous functionality.

The first function, pixelColorID, is what actually identifies color. It takes in a pixel, then uses three if statements to parse whether each part of the RGB is 0 or not. The result of these three if statements is a single three-bit binary number, where 1 means that the RGB value corresponding to that bit is nonzero. For example, if a pixel has an RGB value of [0,255,155], then the three bit number will be 011. Next, the number is used as input to a switch statement that identifies the color that corresponds to that combination of RGB. Note that there are not any thresholds, I discuss this further later in the report, but any intensity will cause the number to trigger, and intensities are not compared by the switch statement. In the above example, the pixel would be identified as ‘teal’, which is what would be returned by pixelColorID. The reason that I chose to write the function this way is because other suggestions involved machine learning as a means to identify colors, but I thought that by instead just creating constraints I could avoid machine learning and thus provide a more elegant, efficient, and speedy solution.

The next function, colorID, takes an image. It iterates over the image pixel by pixel calling pixelColorID, and putting the result into a cell array. Then, once looping is finished, the cell array has unique (a MATLAB function) called on it to get rid of duplicates.

Up next is searchForColor. This is the main worker function, it takes a color as a string and a directory also as a string. It then iterates over all images in the directory, and calls colorID on each. Then, I use the ismember function provided by MATLAB with the input color and colors of the image cell array as input. If this returns true, then the image filename is added to a cell array.

An auxiliary function, showImages takes a cell array of image filenames. It iterates over each image, creates a new figure, and then calls imshow on each filename. This function was written for findAndShow, which takes the color and directory as input strings, then calls searchForColor to get the image filename list that is then given to showImages when that is called.

Lastly is findAndShowV2. This was written to run a little bit faster than findAndShow, since it takes quite a bit to run given the challenge of iterating pixel by pixel over a number of potentially large images. It is basically a copy of searchForColor, however in addition to adding image filenames to a cell array if the file is found, it also prints the name out and calls imshow on the image.

When using the program, my suggestion is that you simply call findAndShowV2. Things worth mentioning include the following: the directory you provide does not necessarily have to be the full directory, you can just specify a subfolder if the images are stored in the same directory as the code but in their own folder. Furthermore, you can only search for one color at a time, I did not have time to implement searching for multiple colors or for combinations of colors. Again, the colors you can search for are ‘white’, ‘black’, ‘red’, ‘green’, ‘blue’, ‘yellow’, ‘teal’, and ‘magenta’. The color names must be typed to exactly match, otherwise the program will fail, as I also did not have time to program in robustness.

Before I get on with the *Remarks and Future Work* section, I’d like to provide my results for each searchable color.

White:

```
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\FV1.jpg
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3698.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3767.jpg
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3768.jpg
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3769.jpg
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3792.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3843.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3846.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3847.JPG
```

```
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3852.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3857.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3862.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3863.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3868.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3871.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3874.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3876.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3881.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\gun1.bmp
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\joy1.bmp
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\pointer1.bmp
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3789.PNG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3793.PNG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3877.PNG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE
332\ColorID\Images\2E8600AC-CBD2-437D-8273-9098E2A8960B.jpeg
Finished.
```

Black:

```
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3698.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3767.jpg
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3769.jpg
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3792.JPG
```

```
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3843.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3846.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3847.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3852.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3857.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3863.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3874.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3876.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3881.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\joy1.bmp
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3793.PNG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3877.PNG
Finished.
```

Red:

```
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3698.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3768.jpg
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3769.jpg
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3792.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3843.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3846.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3847.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3857.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3863.JPG
```

```
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3871.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3874.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3881.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3789.PNG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3793.PNG
Finished.
```

Green:

```
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3698.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3769.jpg
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3792.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3843.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3846.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3852.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3857.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3863.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3876.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3881.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\joy1.bmp
Finished.
```

Blue:

```
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\FV1.jpg
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3698.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3768.jpg
```

```
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3769.jpg
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3792.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3843.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3846.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3847.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3852.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3857.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3862.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3863.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3868.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3871.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3874.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3876.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3881.JPG
Finished.
```

Yellow:

```
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\FV1.jpg
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3698.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3768.jpg
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3769.jpg
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3792.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3843.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3846.JPG
```

```
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3847.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3852.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3857.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3862.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3863.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3868.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3871.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3874.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3876.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3881.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\joy1.bmp
C:\Users\mschi\OneDrive\Documents\MATLAB\EE
332\ColorID\Images\2E8600AC-CBD2-437D-8273-9098E2A8960B.jpeg
Finished.
```

Teal:

```
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3698.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3768.jpg
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3769.jpg
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3792.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3843.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3846.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3857.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3863.JPG
```

```
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3868.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3871.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3874.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3881.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE
332\ColorID\Images\2E8600AC-CBD2-437D-8273-9098E2A8960B.jpeg
Finished.
```

Magenta:

```
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3698.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3768.jpg
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3769.jpg
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3792.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3843.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3846.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3852.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3857.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3863.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3868.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3871.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3874.JPG
C:\Users\mschi\OneDrive\Documents\MATLAB\EE 332\ColorID\Images\IMG_3881.JPG
Finished.
```

Note that it very much overdetects, which I suppose is better than not detecting enough. Further, it prints the entire filename, which is actually kind of annoying and I would definitely fix that if I had more time.

Remarks and Future Work

The joke from the *Project Description* portion of this report is that the photos of Alexandria Ocasio-Cortez and Elon Musk represent a lyric in a song by Billy Joel (who was in the first photo) called Piano Man. The lyric goes, “And the waitress is practicing politics, as the businessmen slowly get stoned,” in the joke the businessman is Elon Musk, the waitress practicing politics is Alexandria Ocasio-Cortez. All three people are some of my favorite people for different reasons, I just think the sequence of images is pretty funny.

My main remark is that color detection is somewhat inaccurate. Often, colors will be detected that do not actually exist in a given picture (for example try ‘green’, a picture with only black, white, and yellow will pop up). However, I am also unsure as to whether this might not be because some of the colors are actually present in error, but it is only in a few scattered pixels that the human eye cannot pick up on. It could also be that some of the black pixels are not perfectly black, as my constraints are a bit loose as to what qualifies as different colors. However, I decided that I did not want to use machine learning for color identification as was suggested by the website that inspired this project, as I felt like that is a bit of a cop out and by far not the most elegant solution. Instead, I sought to solve the problem using constraints and a switch statement instead.

I think that if I had an extra month to work, I would work on a few main things. The first would be working on is improving the color constraints. The second thing I would work on is adding more identifiable colors. The last thing that I would work on is the creation a better UI since at the moment you basically call a function in MATLAB and have a bunch of figures randomly pop up.

Firstly, here is how I would improve the color constraints. At the moment, the color constraints basically work using a switch statement, wherein I check each RGB value for each pixel, and denote whether that pixel's RGB value is zero or not. Then, based on that I have a switch statement with eight cases which basically translates a binary number where each bit denotes whether the intensity of say red is zero or not. Based on the combination, a color is assigned. However, this means that if R = 5, G = 255, B = 1, the color of that pixel is denoted as white despite being more or less green. A better analysis would take into consideration the comparison in magnitude between each color to better make the determination. The primary reason I did not here is that I thought dealing with the directory combing would take a lot longer than it actually did, though not by much thankfully. As such, I had wanted to just create a simple color detection method, the simplest of which was a switch statement. With more time, I would be able to set up a more complex conditional structure that would ultimately result in more accurate color detection.

Another way I would improve color detection is by looking at the neighborhood around each pixel. Then, I would have a threshold for how many neighboring pixels would have to have the same color as the pixel in question, in order to determine whether the color identified for the pixel is accurate or not. This would be done using a simple pixel-by-pixel detector, however it is my hope that this larger perspective would allow for noise to be smoothed out. Of course, this may lead to incorrect results from something akin to smearing, but I have no way to know at this point as I did not have time to attempt implementing this.

Another thing I would have liked to have time to do is add more colors. For example, ‘orange’, or ‘brown’, are not searchable colors. However, in order to add more colors, I would have to first improve the color detection since I would need to be able to analyze color combinations far more than just pointing out that there is a combination of colors as I do now. Worth noting is that I do not think this would mesh entirely well with the neighborhood detection idea I would have. A particular case where this would be true is a picture of a sunrise, where you have a gradient between the yellow of the sun to the oranges and pinks of the surrounding sky, to the blue of the sky further away from the sun. In this case, some of the oranges might be ‘smeared’ during detection and not be detected in favor of calling those pixels pink or yellow.

In a similar vein, I would want to introduce some robustness on search entries. At the moment, you have to type exactly ‘yellow’, you cannot type ‘Yellow’ or ‘YELLOW’ or ‘yelllow’ (the last one is a spelling error on purpose). This is not very user friendly, as you have to know and type exactly what the name of a searchable color is. It is also impossible to search for multiple colors at the same time, or to search for only pictures with a specific combination of colors, both of which are things I would like to add if given the time.

The last addition I would like to mention is that is that the current UI is kind of ugly and not at all user friendly. Right now, to search for pictures of a certain color, you have to be in MATLAB, and then you type a command like:

```
findAndShowV2('green','Images');
```

which then prints out a list of filenames and at the same time creates a bunch of pop-up figures, making a mess of the user's screen and meaning they cannot really do anything else until the program finishes running, which can also take quite some time (exhaustively searching every pixel of every image of varying sizes is slow work).

To this end, I would very much like to revamp the UI. In conjunction with the above changes, a better UI would look something like: you run the program, and a window with all the pictures in the directory in scrollable format with a search bar at the top pops up. You then type the color(s) you are searching for, and any images without this color present disappear. In other words, I would like to have designed a UI that would actually be seen in something production-level, and I very much regret that I did not have the time to do as such.

I definitely see myself using computer vision in the future. Some projects I can foresee myself doing involve gesture recognition (I would have wanted to do that this quarter actually, however illness got in the way as I had to settle for doing a project that would be more rapidly implementable since I lost multiple weeks), facial recognition, and object recognition, given that these are the most likely to come up in the field of smart home technology, which is what I plan to do.

I would also be interested in seeing how some of these applications I have developed this quarter might run faster if we were to rewrite them to use threads and concurrency instead of just regular loops. For example, there are many times where we iterate over an entire image pixel by pixel, or group of pixels. This generally contributes heavily to runtime increasing. However, a lot of the time, this pixel-by-pixel analysis does not care about surrounding or previous pixels or results. For example, the creation of a histogram does not require that pixels be iterated over in any particular order. As such, histogram creation could be done far more efficiently by creating different threads to analyze each pixel, and then combining the results when threads are combined. Of course, I kind of doubt that MATLAB has very good support for multiple threads, so this may have to be done in another language that does.

Course Feedback and Suggestions

Overall I had a lot of fun with this course. It very much challenged me intellectually, and exposed me to a new way of thinking that I hope to carry forward into the rest of my life. I feel like I have also grown as a programmer, although not to the extent that I would have liked to. From working out how to properly implement sets in MATLAB during the first machine problem to sighing in relief when the tracking actually worked for the last, this class has provided me with plenty of opportunities to create tools that are used all the time in computer vision. I was actually disappointed at one point when it turned out that we did not have to implement paneling in its own machine problem, instead it was just covered in class.

My singular point of contention is actually that I am wondering why this report was fifteen pages long, while previous reports have been less than a page. While yes, we are also supposed to talk about things such as our thoughts on the field of computer vision as well as review this class, and our current report has to include things like a project description and remarks about how we would make our project better as well as future projects we would like to work on, I do not think that it is quite fourteen pages worth without adding some filler. This is especially true for people who are more concise with their words, of which type I am fortunate enough to not be as I also love to write. However, perhaps consider reducing the report to ten pages in length, or even eight.

Perhaps my favorite part about this class is that it was a more mathematical approach to computer vision. Instead of just focusing on what functions already exist and how to use them we got to learn how each function works. It was also entertaining to have machine learning as a black box railed against, I appreciate the perspective that any solution to a problem should be replicable and relatively deterministic, and understood by anyone who knows about the solution. I look forward to taking the advanced version of this course in the future, should I have the opportunity to.