IRL #7: Progress Review

Man-Ning Chen (Mandy)

Team G: EXCALIBR

Teammates: Yiqing Cai, Huan-Yang Chang,

Siddharth Raina, Sambuddha Sarkar

Individual Progress

Overview

My goal for this week is completing color checker detection and segmentation so that I can start developing color mapping function next week.

Color Calibration

Note: I am currently using my personal smart phone for this algorithm development. After the algorithm is completed, the testing will be done on EVT cameras.

Color calibration is to measure and adjust the color response of a device (input or output) to a known state. We use an X-Rite ColorChecker Classic Card [Fig. 1] as our ground truths (The manufacture gives us the true color values) and aim at mapping the colors recorded by the cameras these ground truths.

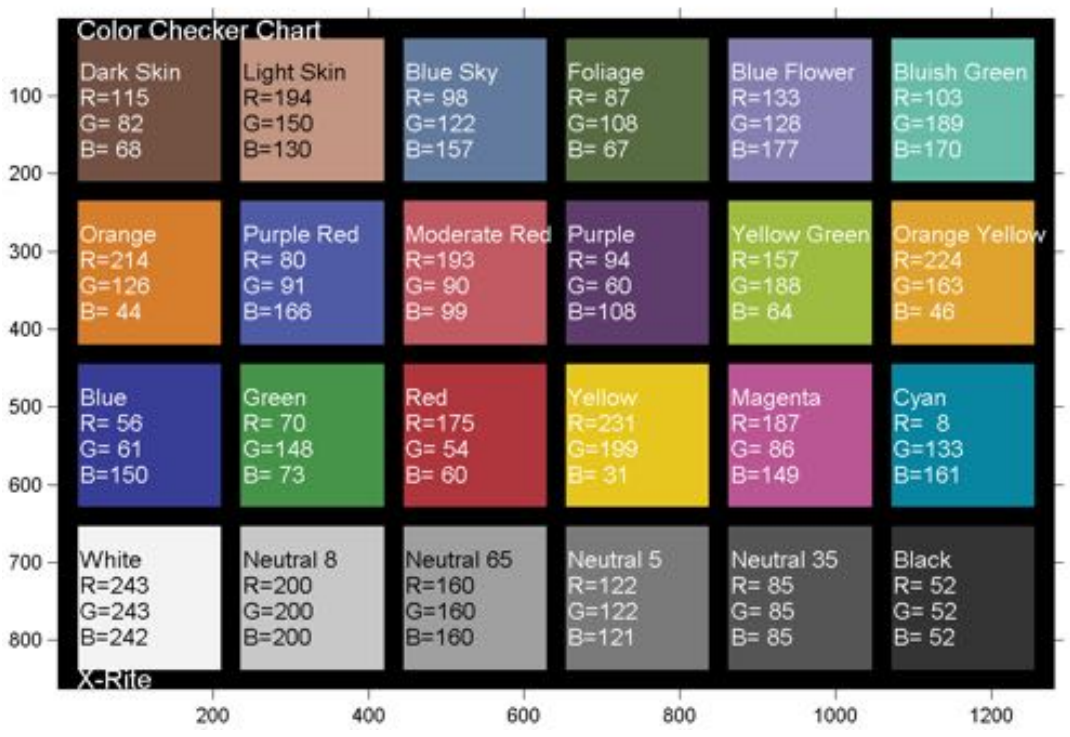


Figure 1. An X-Rite ColorChecker Classic Card

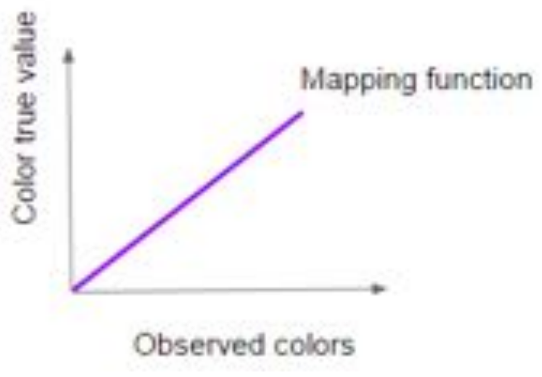


Figure 2. Mapping function (unreal): This graph is only for illustrate the concepts

Color Calibration Pipeline

My first plan was to divide and conquer the problem by doing color patch segmentation first and address the color board detection part later. I wanted to combine these two parts to get the algorithm which can detect each color on the colorchecker in the images.

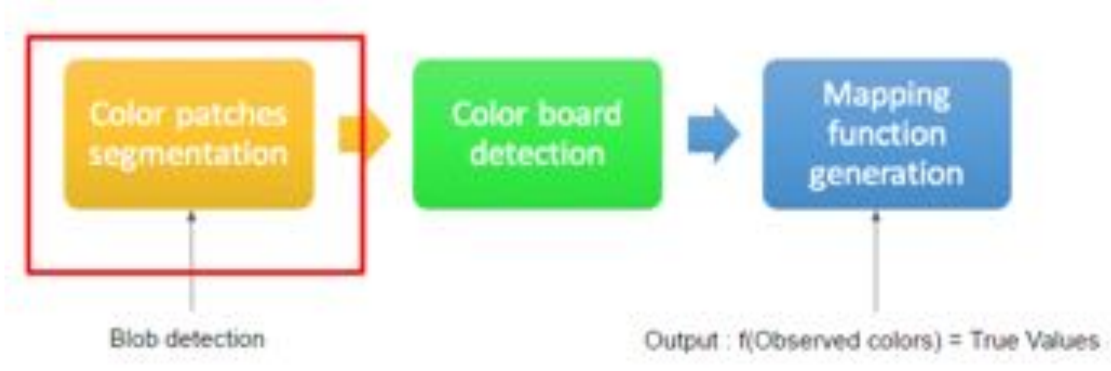

Color patches segmentation
Color board detection
Mapping function generation
Blob detection
Output : f(Observed colors) = True Values

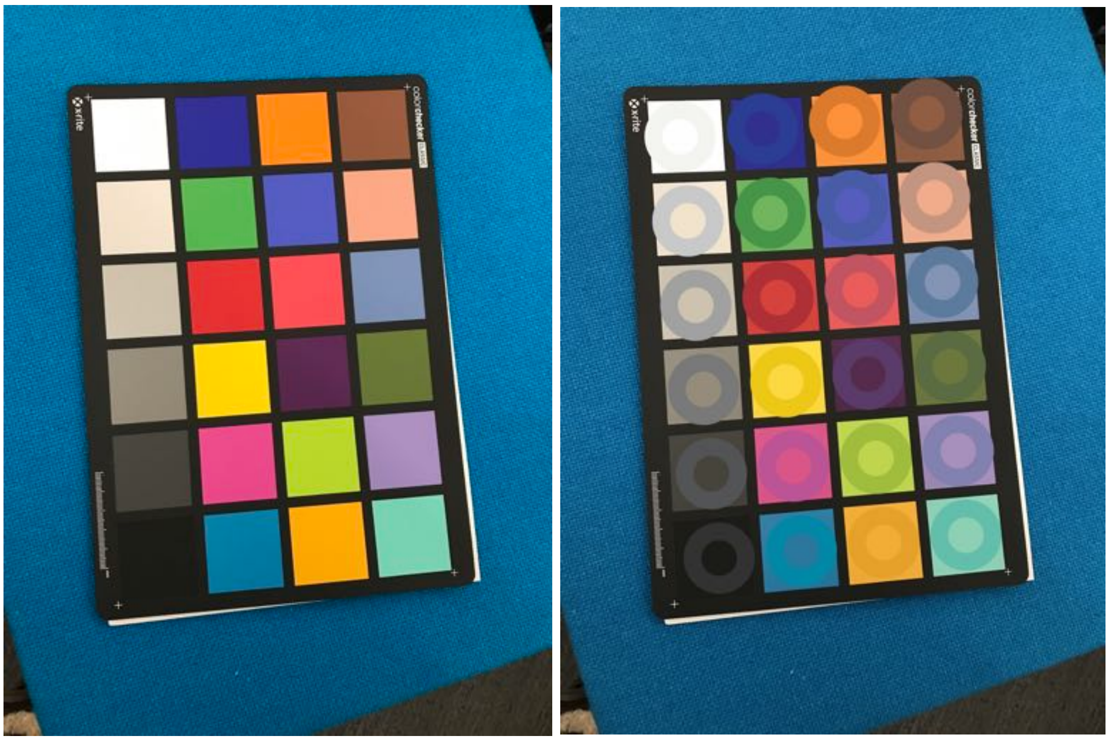
Figure 3. Previous Color Calibration Pipeline

However, I found this pipeline complicated the problem. Since I have the contour of the color checker which is a very strong information, I should use this information for color checker detection as well as segmentation.

Therefore, I did a lot of researches on how to use colorchecker properties to find it in an image. I found several researchers had similar need of an automatic colorchecker finder and their developed algorithms. I tried them all and found two of them work well. One of them is Macduff, another is CCFind.

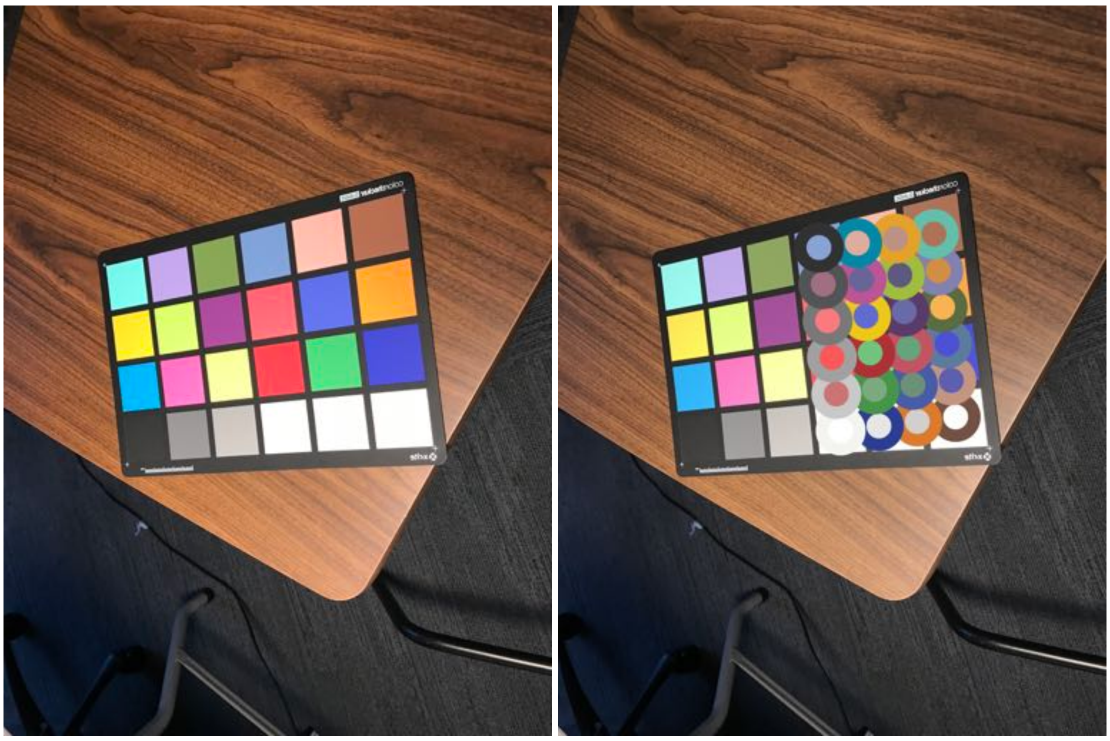
Macduff

Macduff could find and segment the colorchecker successfully. This algorithm is also quite efficient. Nonetheless, when I tested it with complex background, I found that this algorithm cannot work when the colorchecker does not occupy the majority part of the image. See Fig.4 and Fig. 5.



Input Image Output Image

Figure 4. Macduff Successful Result



Input Image Output Image

Figure 5. Macduff Failure Result

CCFind

In fact, I found this method before Macduff. Yet, when I take the image with complex background I thought that this method could not address images with complex background. It failed to find out the color patches and when I traced the code I found in succeeded cases, it cut off all background edges cleanly while in failure cases the background edges remained. I thought this prevented it from getting the repeated squares.

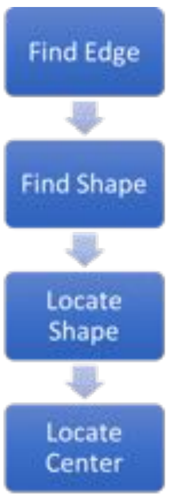
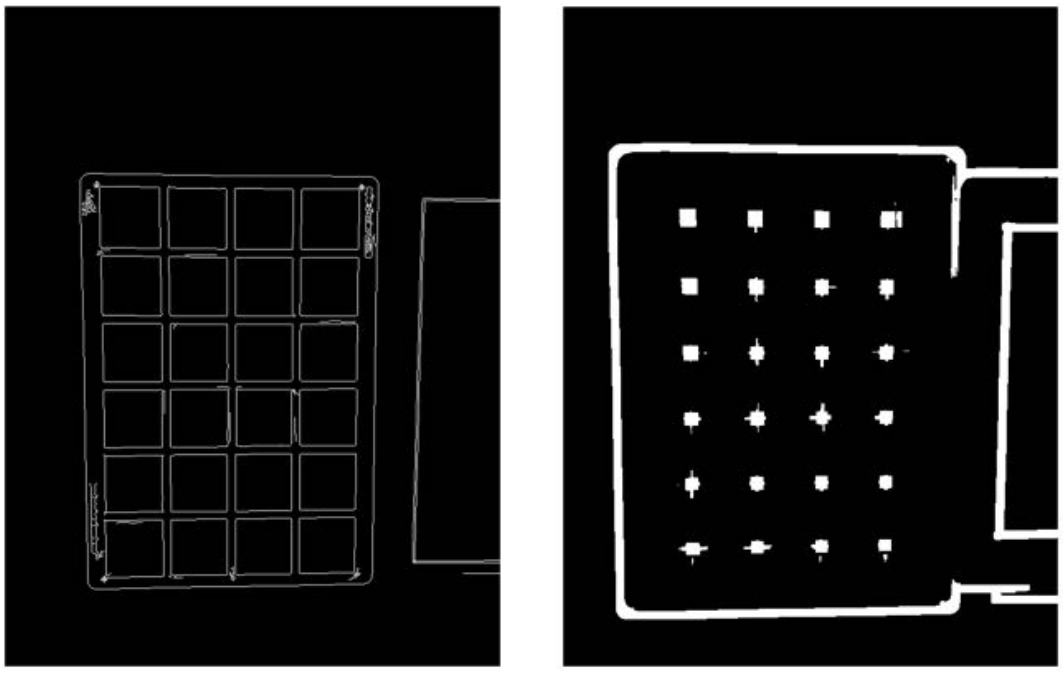


Figure 6. CCFind Algorithm



Find Edge

Find Shape

Figure 7. How CCFind works

After Macduff algorithm failed in robust tests, I studied all the algorithms trying to build my own colorchecker detector. When I once again traced CCFind's code. I found that it did several downsampling in the code itself, which made me guess the reason it failed before is because of the image size. Therefore, I downsample my images first before applying CCFind on them. The results are good. Furthermore, since CCFind only use the contour information of the colorchecker, it would not be affected by bad lighting conditions.

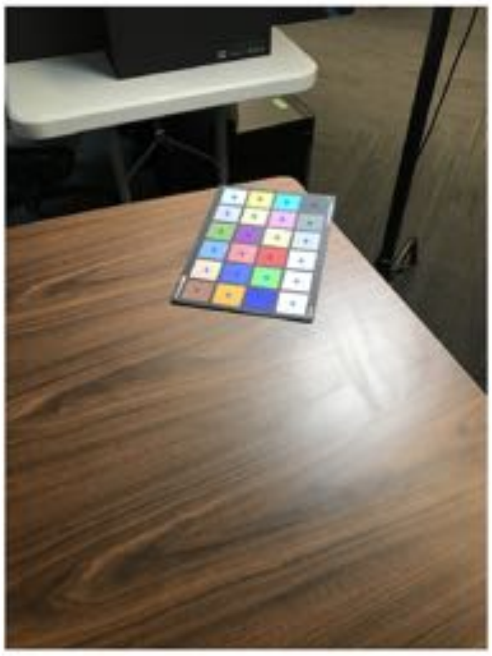


Image1

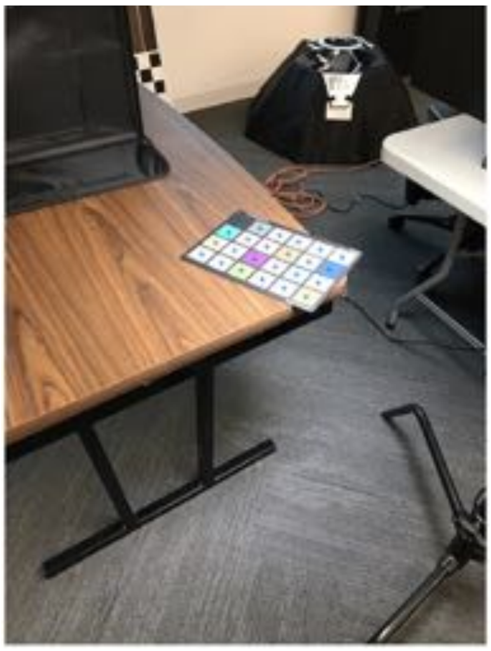


Image2

Figure 8. CCFind is robust while
1. the colorchecker only occupies small portion of the images
2. images are taken in special light conditions

Challenge

1. While the Intensity threshold used in our image cuts off the background, it also cuts off the black patch. We need other ways to find the center of the black patch. I am considering using the intersection of lines of other patch centers or extrapolating other patch center lines to get the position of the black patch.
2. The algorithm is easily affected by the lighting condition of the experiment condition. I'll have to conquer this problem.

These two were my last ILR challenge. They have been resolved in these two weeks by the new method. For this ILR, my new challenges are:

1. Keep conducting robust tests to see what situations CCFind cannot handle.
2. Start developing mapping function.

Teamwork

Yiqing Cai	Generate one of the best path for robot arm (go through as least positions as possible to cover FOVs of all cameras)
Huan-Yang Chang	Simulation system setup, Geometry calibration
Siddharth Raina	Sensor noise calibration plan development
Sambuddha Sarkar	Works on Blender. Generate virtual calibration target and render the virtual images of the target.

References

[1] https://en.wikipedia.org/wiki/Color_calibration

[2] https://en.wikipedia.org/wiki/Flood_fill