Session 5: Computer Vision

Objectives

- Learn the basic tools of OpenCV

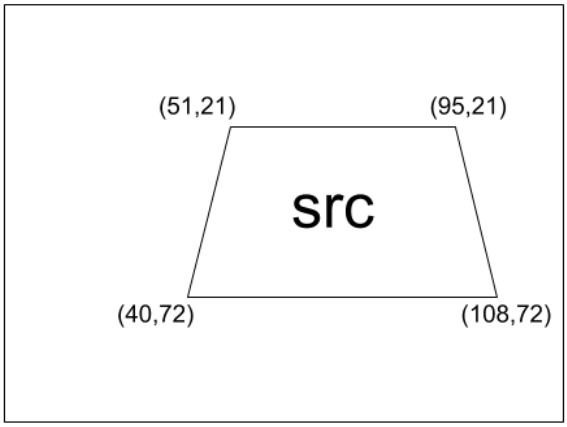
Exercise 5-1: Perspective Transform

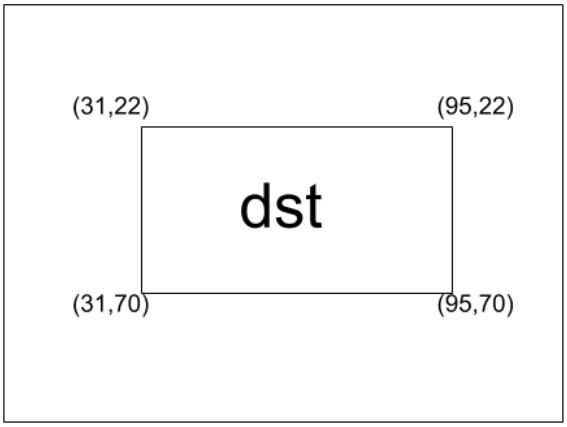
TO-DO:

- Print out the calibration sheet (perspectiveTransform.pdf)
- Place robocar at the first line.
- Adjust and tighten camera mount arm to place rectangle roughly in the middle of the field of vision.
- Run `python persptf.py`
- Adjust placement using the 10-sec preview. Once preview ends, program saves two pics - original.jpg and warped.jpg.

Inspect the saved picture files and extract coordinates.

- Find and record the coordinates of the vertices of the original polygon.
- Enter the coordinates as the source (`src`) matrix in the function warp(img).
- Adjust the destination (`dst`) matrices in the function warp(img) to map the original polygon to a rectangle in the middle of the warped image.





- Run again: `python persptf.py`

- Check that the rectangle in the transformed image warped.jpg looks ok and centered. If not repeat above steps.

Once you are satisfied with the transform coordinates,

- Record both `src` and `dst` matrices for later exercises.

Common color spaces:

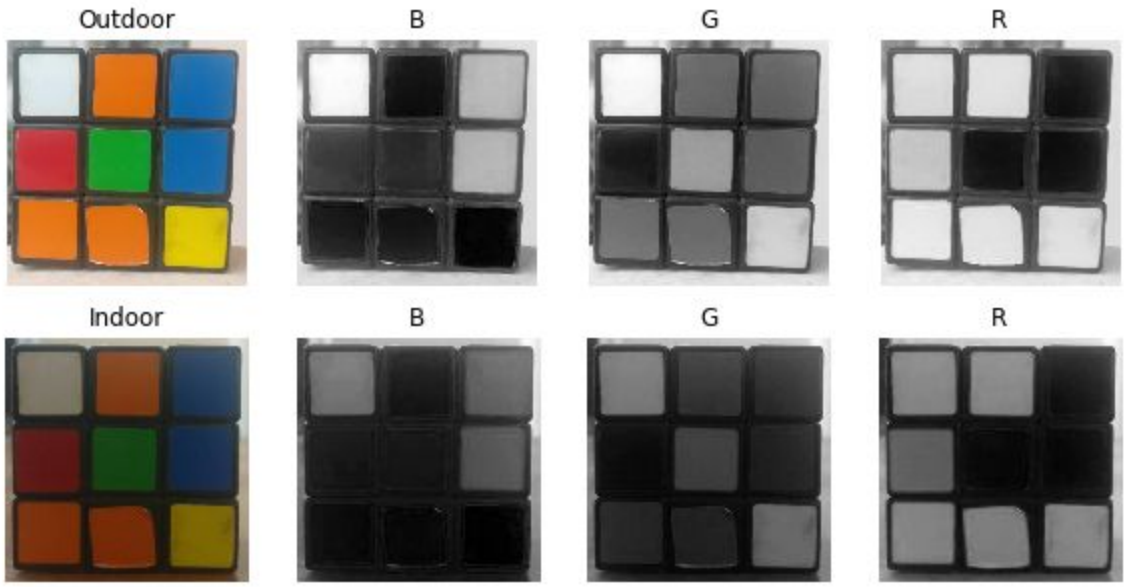
Color space, also known as the color model (or color system), is an abstract mathematical model which simply describes the range of colors as tuples of numbers, typically as 3 or 4 values or color components (e.g. RGB).

RGB - Red, Green, Blue

https://www.learnopencv.com/color-spaces-in-opencv-cpp-python/

The RGB colorspace has the following properties

- It is an additive colorspace where colors are obtained by a linear combination of Red, Green, and Blue values.
- The three channels are correlated by the amount of light hitting the surface.

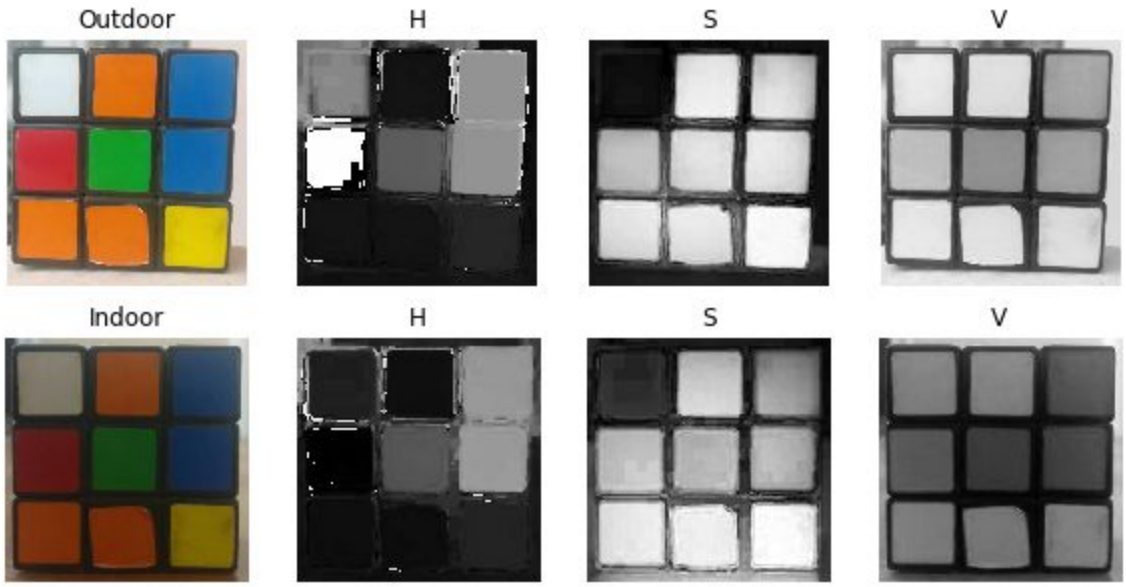


HSV - Hue, Saturation, Value

https://www.learnopencv.com/color-spaces-in-opencv-cpp-python/

The HSV color space has the following three components

H – Hue (Dominant Wavelength).
S – Saturation (Purity / shades of the color).
V – Value (Intensity).



YCrCb / YUV

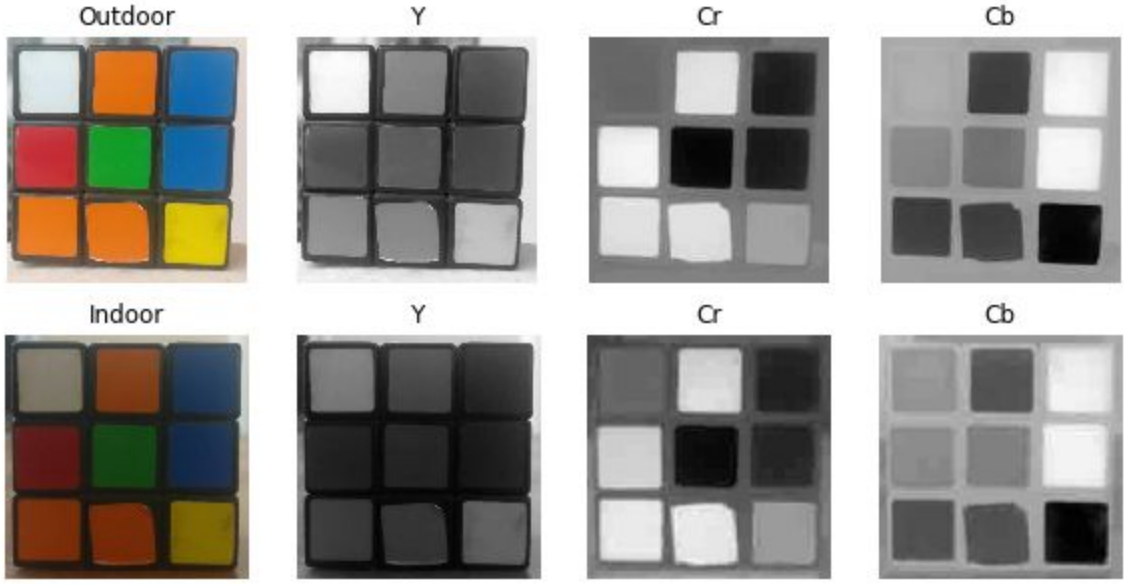
https://www.quora.com/What-is-the-difference-between-YUV-and-YCrCb

YUV and YCbCr are both difference encodings of RGB signals used in video.
YUV is found in legacy analog video.
YCbCr is found in digital video formats.

Y is a weighted sum of RGB, UV and CbCr are both weighted differences between RB and Y. The weights are designed such that UV and CbCr end up being zero or constant for neutral colors (black, gray and white colors where R=G=B).

This means that the luminance information is mostly carried in the Y channel, and the color information is carried in the other two. Video transmission and compression can take advantage

of this. The color information can be subsampled (transmitted at a lower resolution and using less bandwidth) with minimal loss in image quality.



Exercise 5-2: Detecting Color Pixels

In this exercise, we will use color space information to extract dark or vibrant colors. Edit the file line.py in ex-5-2-Extract_Pixels.

TO-DO:

- Update the perspective transform warp matrices coordinates.
- Use color space to:
    - select "dark" pixels and save as a jpg file called `03darkbin.jpg`.
    - select pixels with high saturation levels and save as a jpg file called `03saturationbin.jpg`.
- hint: You might find these openCV functions useful:
    - **cvtColor** (https://docs.opencv.org/2.4/modules/imgproc/doc/miscellaneous_transformations.html)
    - **threshold** (https://docs.opencv.org/3.4/d7/d4d/tutorial_py_thresholding.html)
- Open the two new image files and compare them with the originals. Do you think the extract pixels are correct?

Exercise 5-3: Detecting Edges and Corners

In this exercise, we will use openCV to extract more features from images. Edit the file line.py in ex-5-3-Edges_Corners.

TO-DO:

- Update the perspective transform warp matrices coordinates.
- Extract edges using openCV and save result as a jpg file called `04edge.jpg`.
- Extract corners using openCV and save result as a jpg file called `05corners.jpg`.
- Edit line 6 in line.py to use live images:

  ~~`TESTMODE = 0`~~ `TESTMODE = 1`
- Position your robocar to capture real images to process. For each frame, open the saved jpg files and make observations.