ENPM673_Project2

Problem 1: Histogram Equalization

Histogram equalization

Gray scale

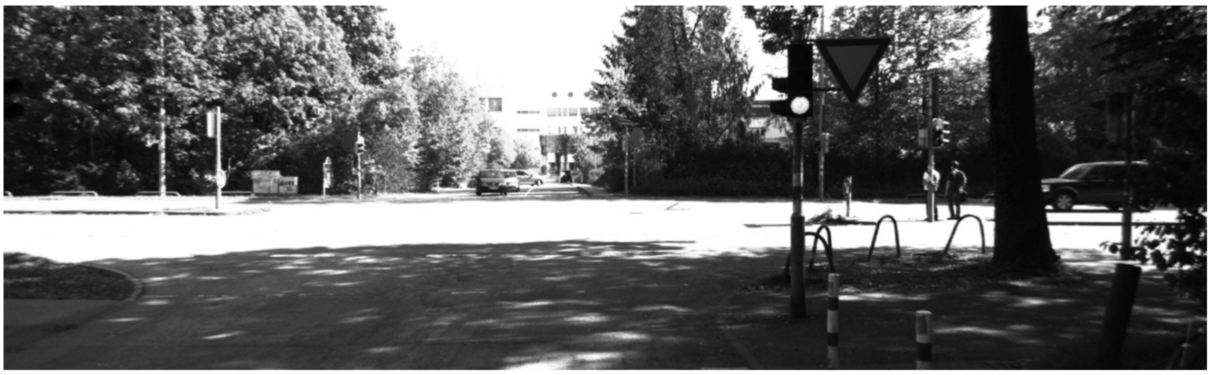
Step 1. Calculate the histogram of pixels intensity

Step 2. Calculate the cummulative distribution function through the histogram from step 1

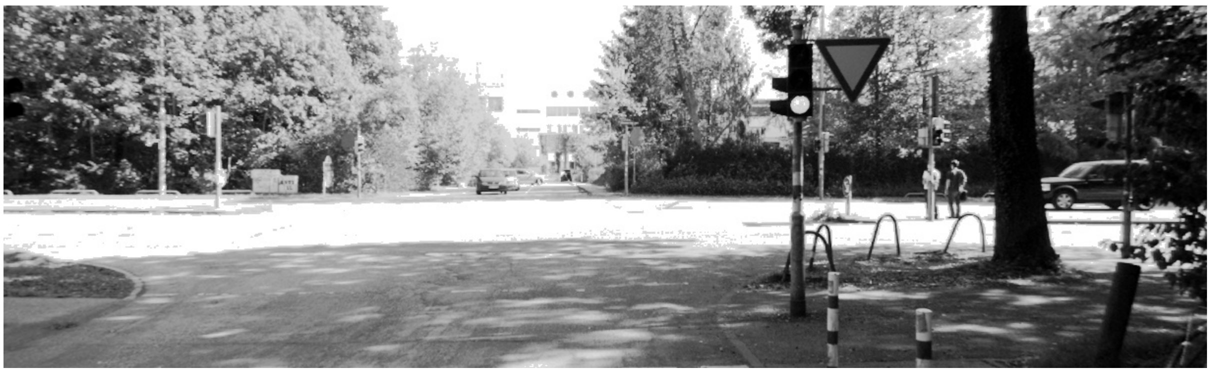
Step 3. Calculate the new intensity value based on the distribution of the intensity

> For example: if there are 50% of the pixels are less or equal to 100, then pixels with intensity of 100 will become $50\% \times 255$

Before historgram equalization

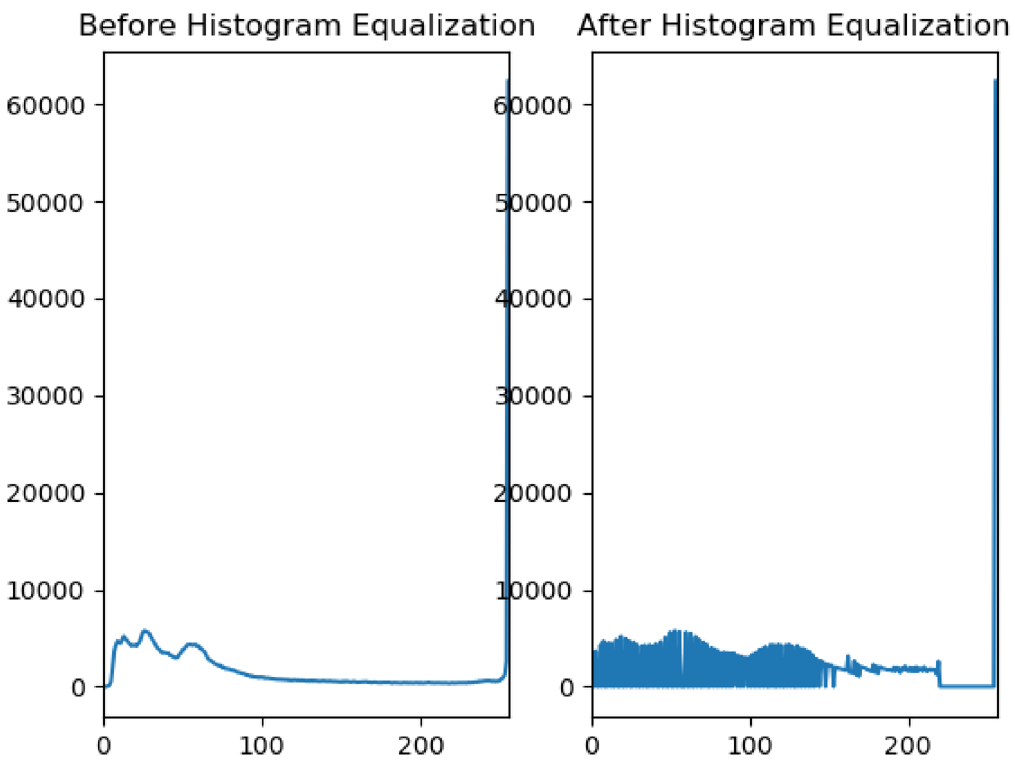


After historgram equalization



The graph below shows the difference of histogram before and after histogram equalization.

Because many pixels have intensity of max intensity 255 and $100\% \times 255 = 255$, the historgram after histogram equalization still has a very high amount of 255 intensity pixels.

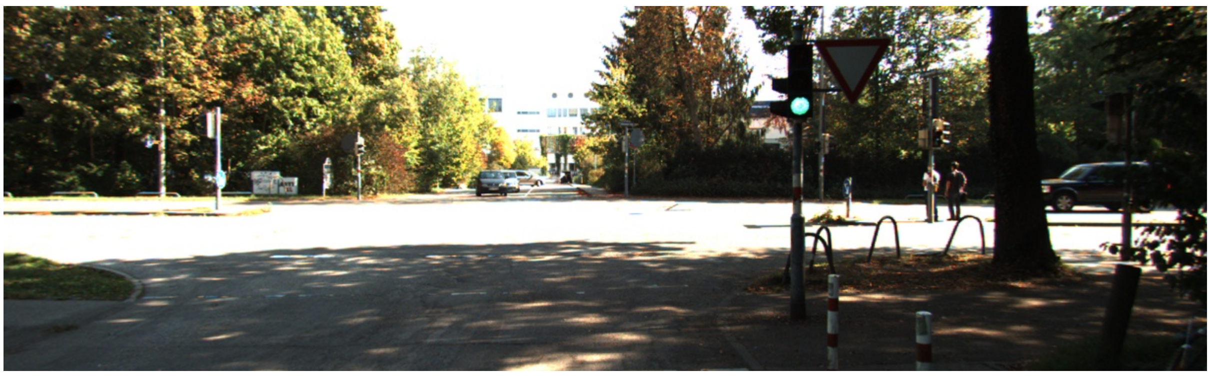


Colored histogram equalization

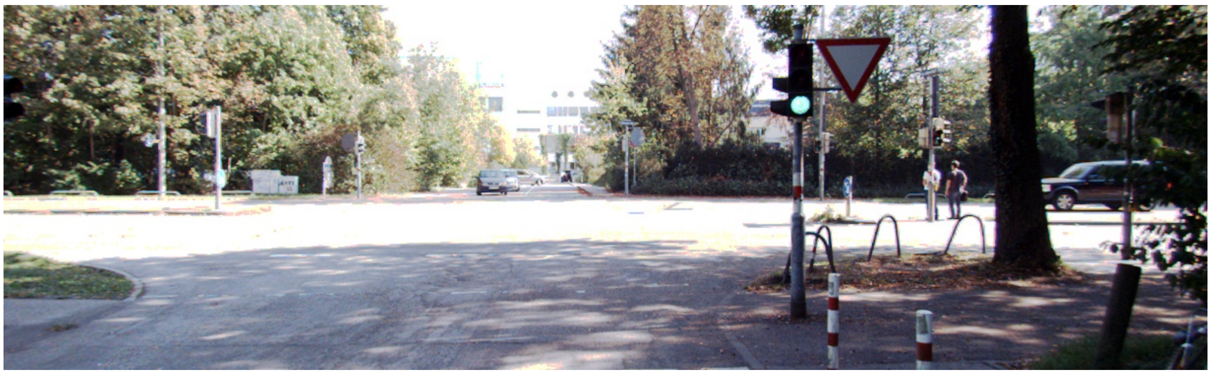
The colored histogram equalization can be done by first seperating the image into r, g, b channels, then apply gray scale historgram equalization to each channels and combine them back together.

However, some of the color will be distored and will not remain the same color property as the original image.

Before histogram equalization



After histogram equalization



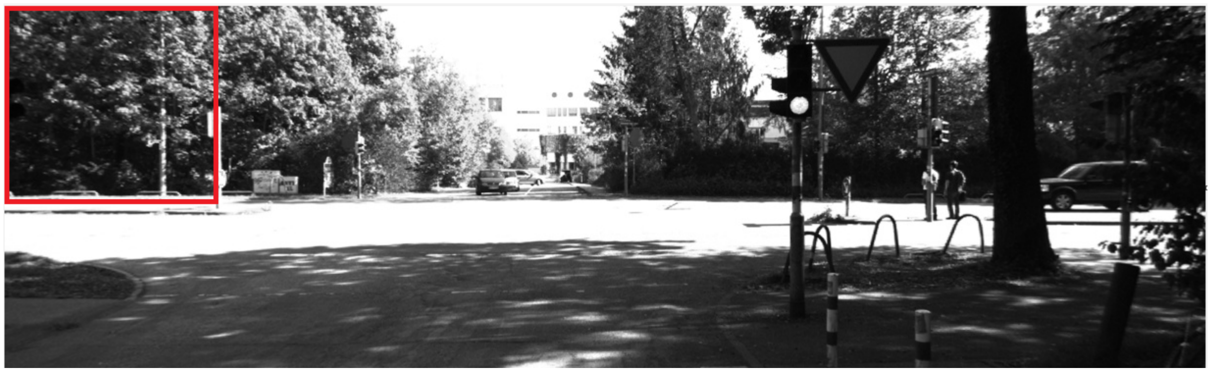
Video

Improved video using histogram equalization

https://drive.google.com/file/d/1Njzg7buXzMTDgAGx-P1LMzP8-VkkAdnA/view?usp=sharing

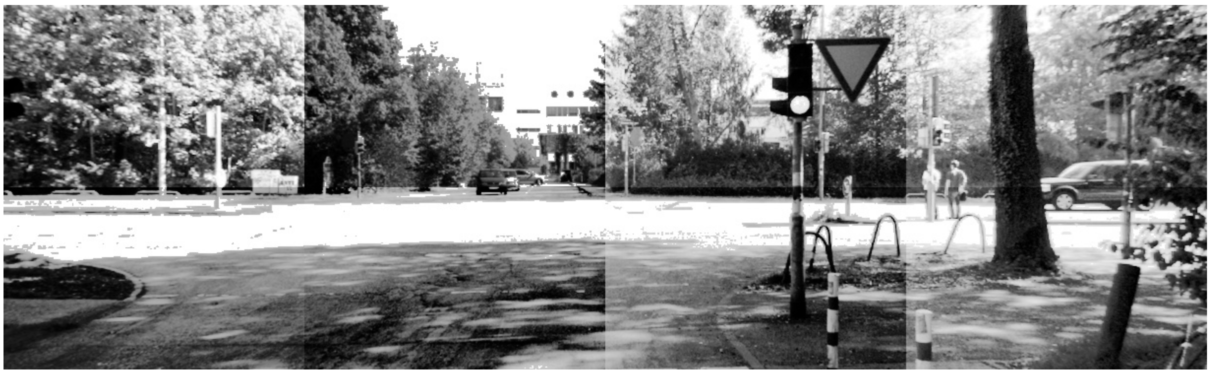
Adaptive historgram equalization

Adaptive histogram performs histogram equalization over a window such as the red region below.



The picture will be devided into different regions, then historgram equalization will be apply to each of them.

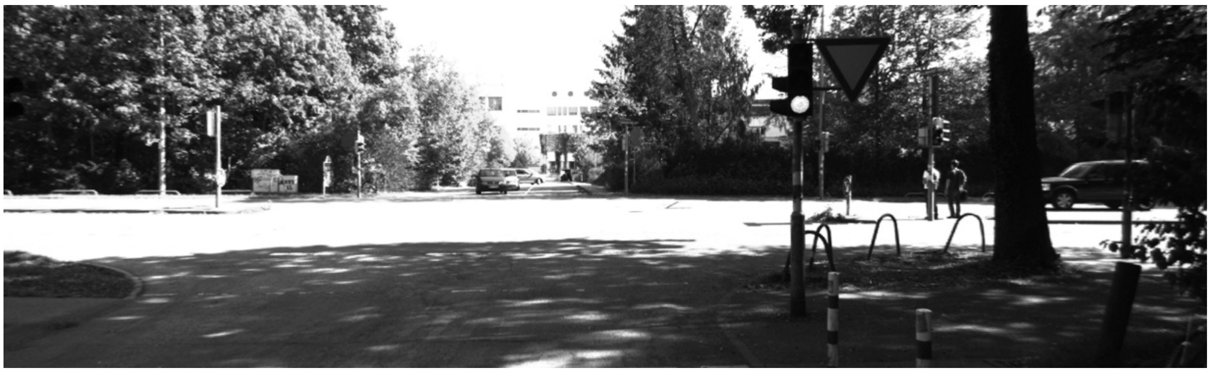
The picture below is devided into 8 pieces. And histogram equalization is applied to each of the sub-windows.



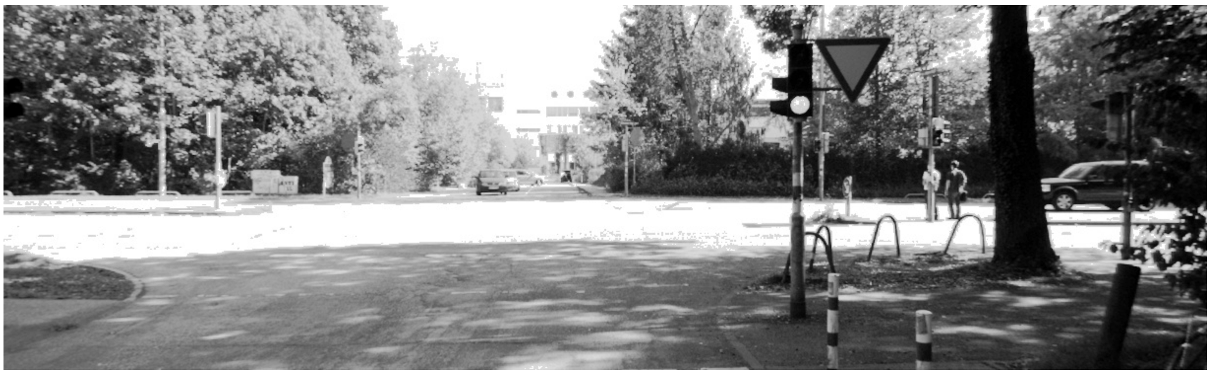
Because the windows are calculate seperately, the boundary of windows show discontinuity.

To alleviate the effect, I use a sliding window with smaller size to perform adaptive historgram equalization.

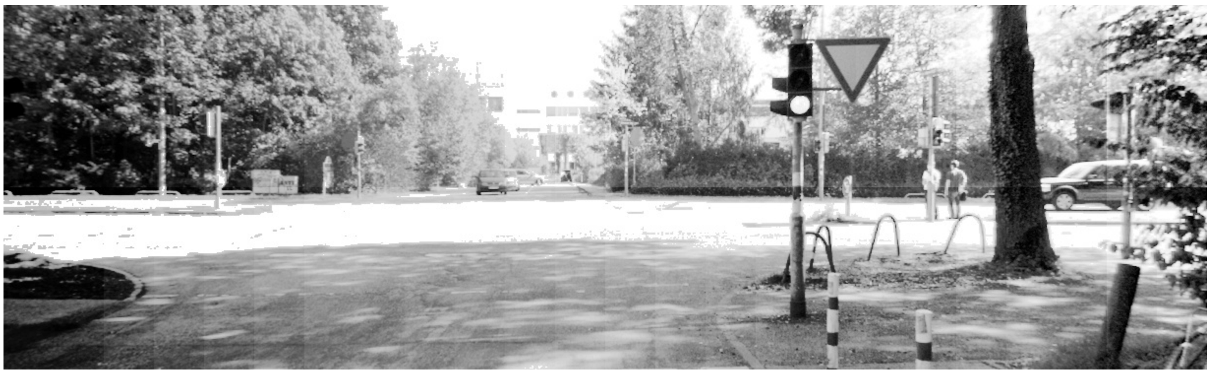
gray scale



historgram equalization

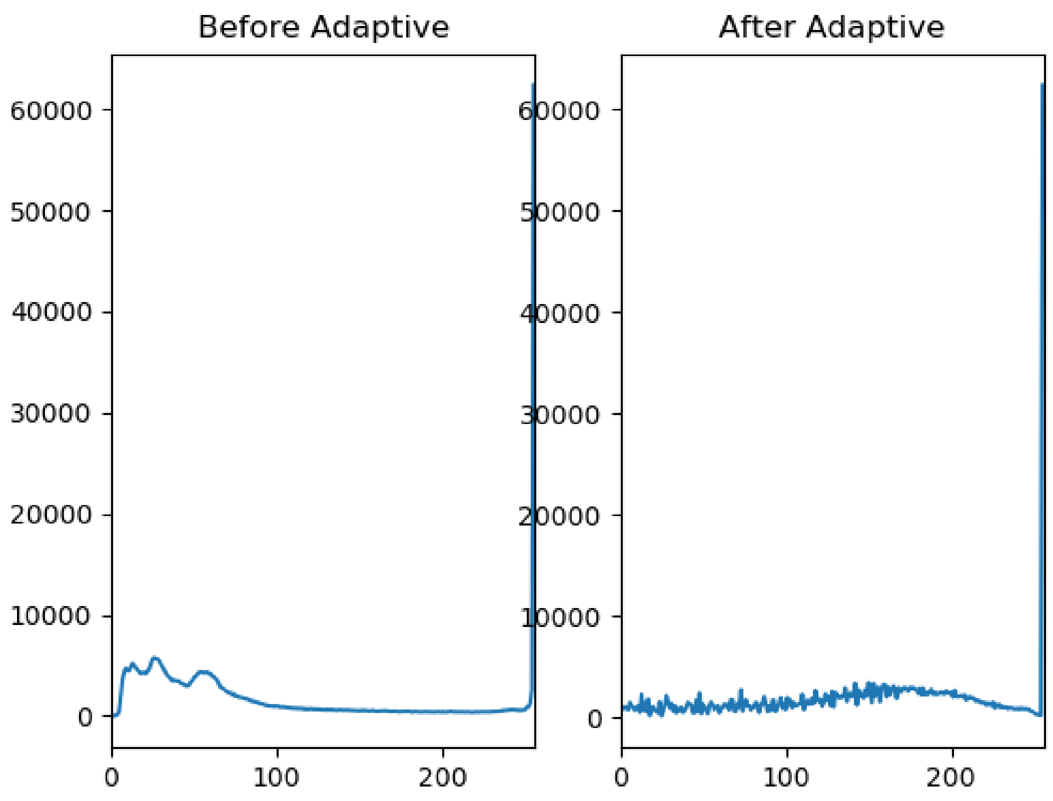


adaptive historgram equalization (using sliding window)



The adaptive historgram picture shows more details at the tree and the right half of the picture then using normal histogram equalization.

The chart below can also show that adaptive histogram has a smoother color intensity then histogram equalization.



Video

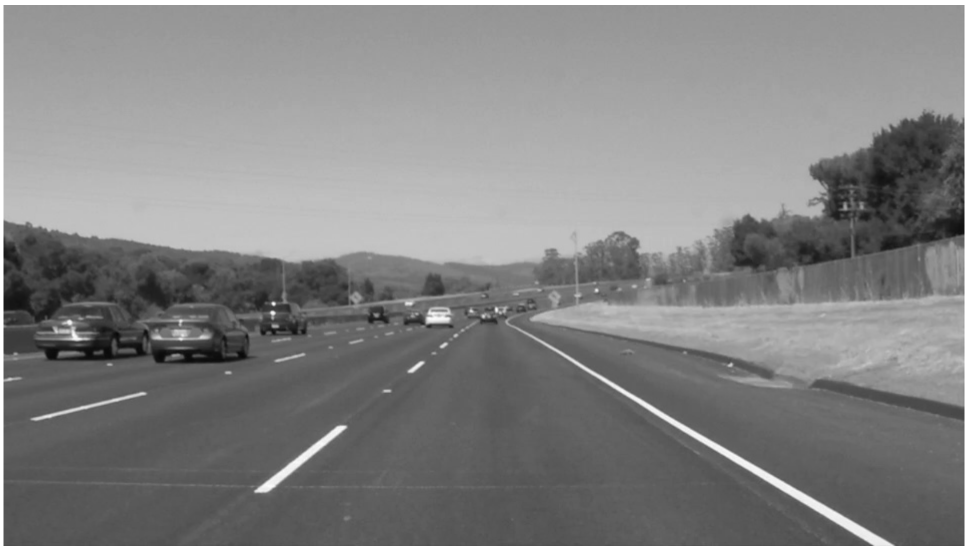
The video is in gray scale because adaptive equalization takes a lot of time.

https://drive.google.com/file/d/1sERaSEEDhurDwctUiRfrHoqK7quopKIc/view?usp=sharing

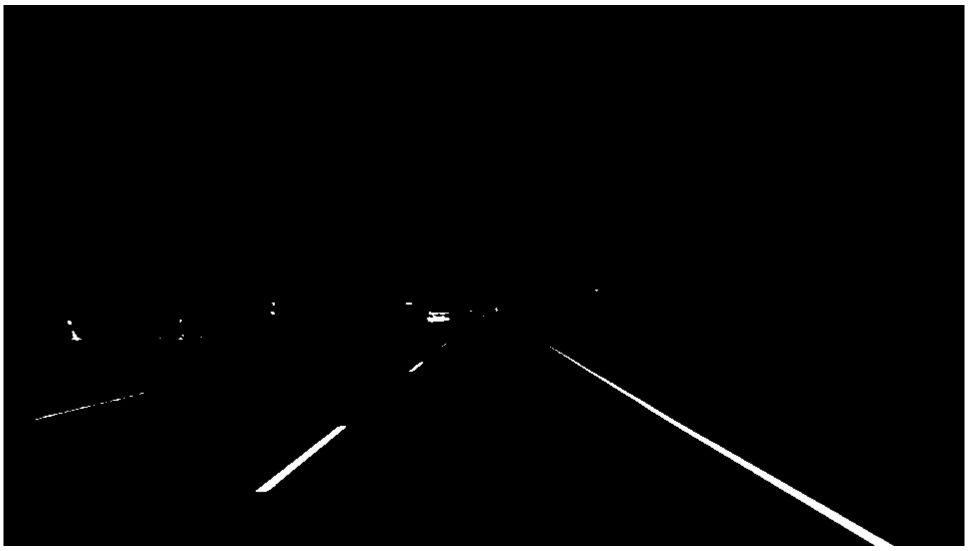
Problem 2: Straight Lane Detection

Detection Pipeline

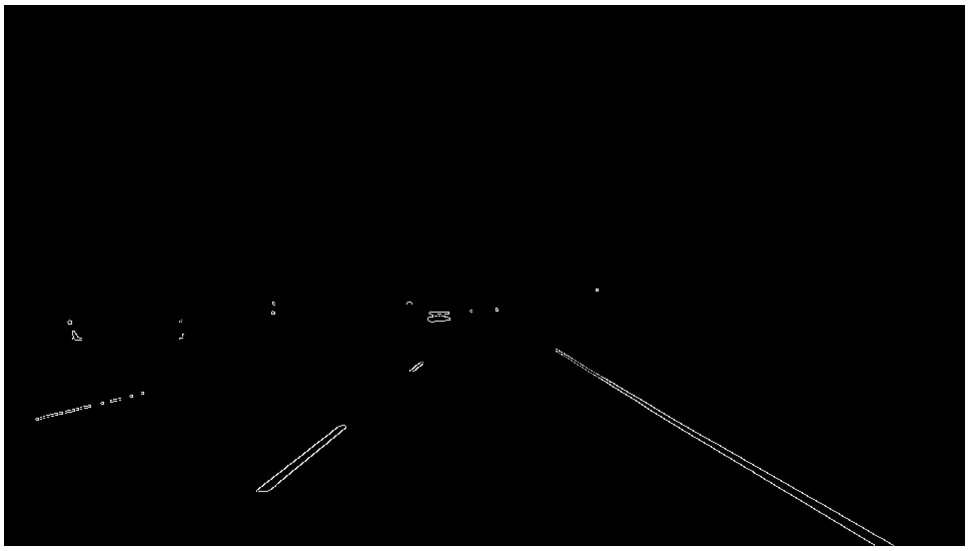
Step 1. Change image to gray scale



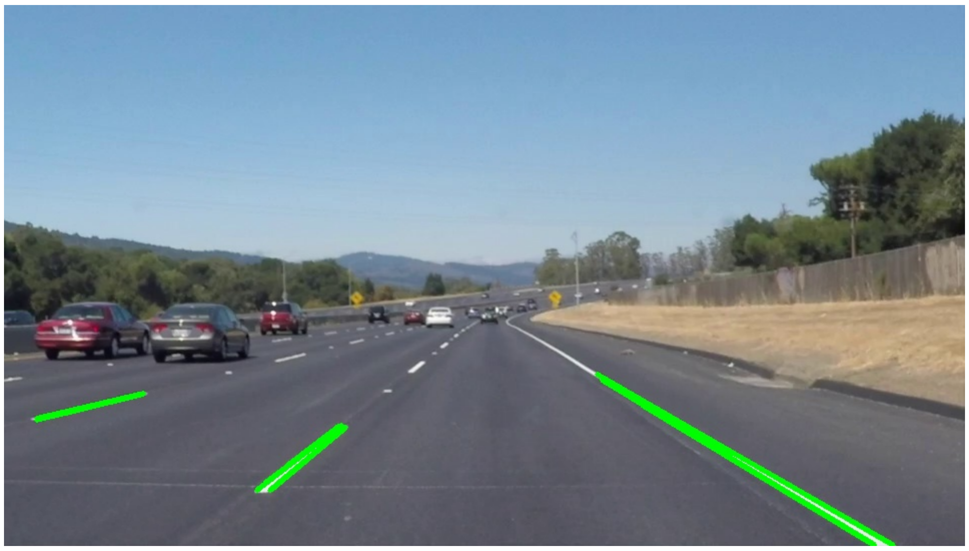
Step 2. Apply threshold to the image



Step 3. Canny edge detection



Step 4. Using hough transformation to find lines in the image



Step 5. Catagorize each points on line based on the distance to each line

After step 5, the line on the picture in step 4 will be seperated into 3 catagories.

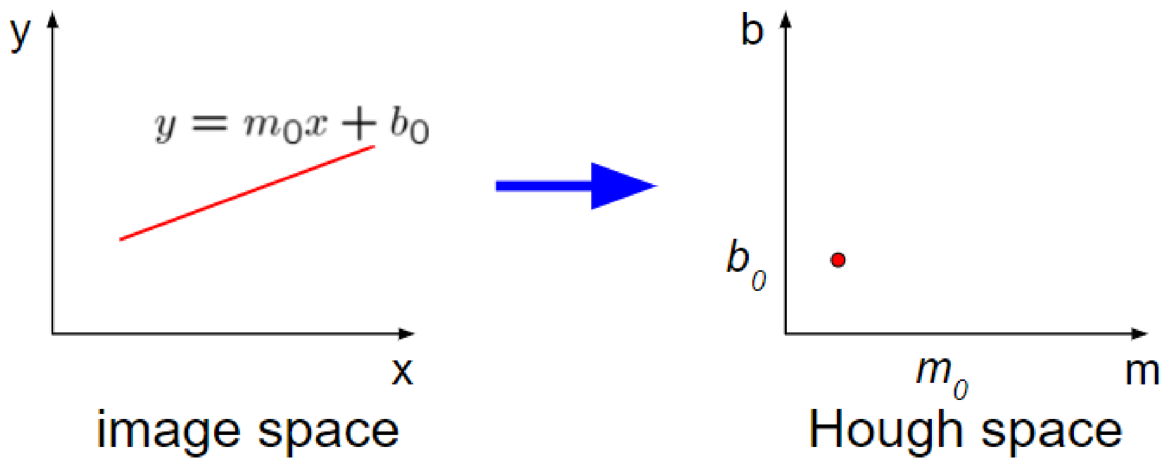
Step 6. Find the optimal line in each catagories by least square line fitting

Step 7. Draw the line with different colors based on its length

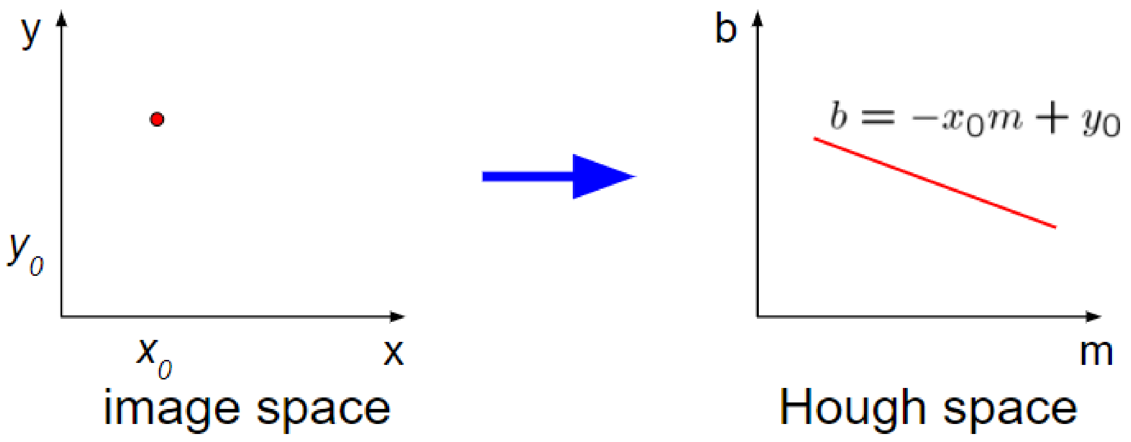
Hough Transformation

Hough Transformation transform points on a image to a parameter space. This parameter space can be the parameter of any function.

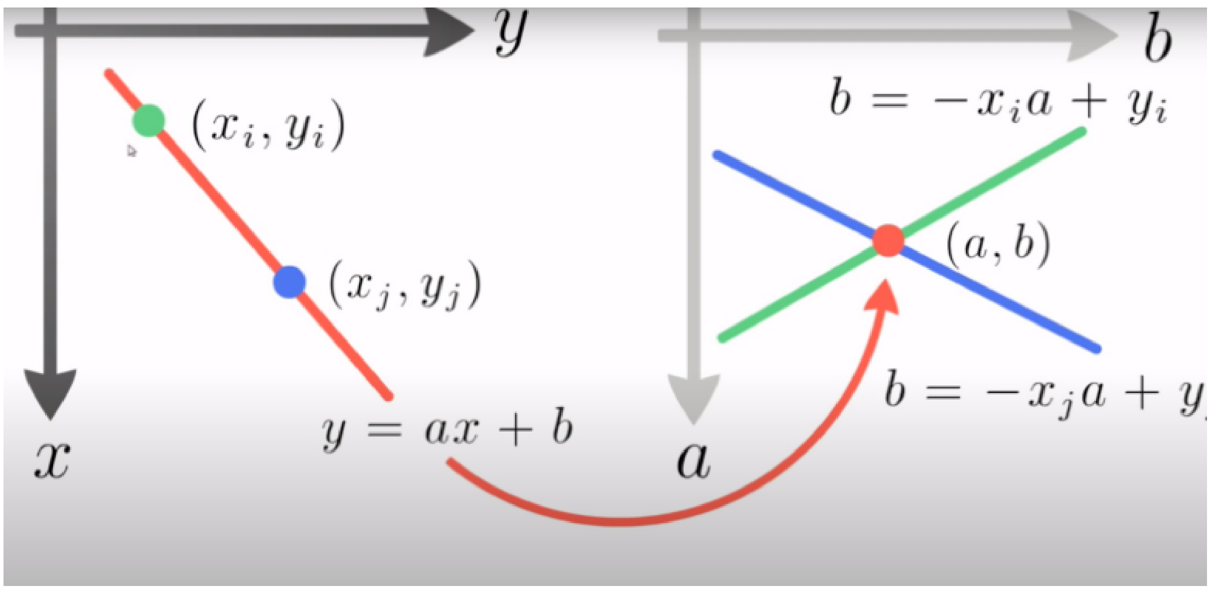
Take a 2d line equation ax+b=y for example, the line in the original graph is transformed into a point in the parameter space.



A point in the original graph is transformed into a line in the parameter space.



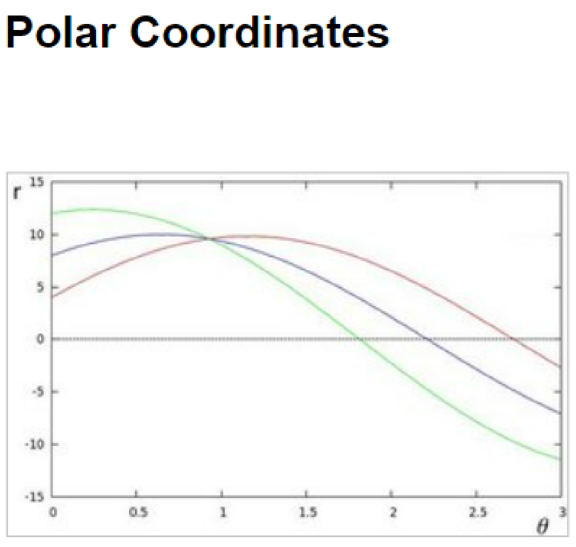
If there are two lines intersect with each other in the Hough space, it means that there are at lease two points that share the same line in the image space.

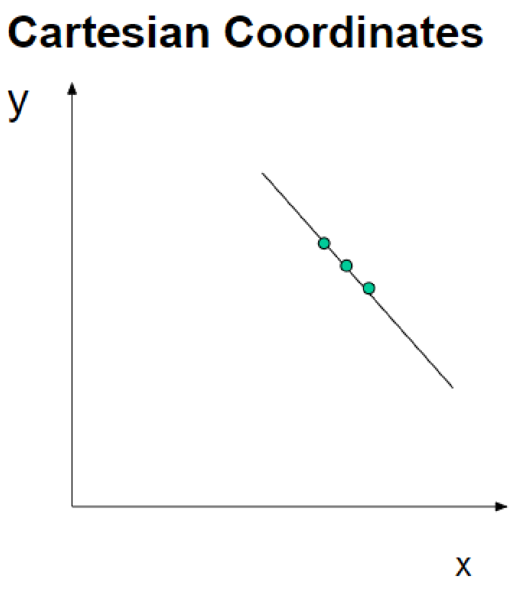


Therefore, we can setup a voting algorithm to calculate the amount of lines and their corresponding parameter in the original image space.

This line equation doesn't necessary has to be ax+b. Any function that can represent a line can perform Hough Transformation.

For example the polar representation





Algorithm:

Using the distance to origin representation for Hough Transformation

$d = x \cos\theta + y \sin\theta$

```
Initialize H[d, t] = 0
for each edge point[x, y] in the image:
  for t from 0 to 180:
    d = x*cost + y*sint
    H[d, t] += 1
Find d, t that H[d, t] is bigger then voting threshold.
The stronger line in the image is d = x*cost + y*sint
```

Pipeline Generalizability

Because I only use threshold and canny to pre-process the image. This pipline is fulnerable to other whilte objects such a white car. If a white car passes nearby, it will be detected as a long lane. And it will be detected as a dash lane if it has some distance with our camera.

Video

The video shows all the detected dash-lane and straight lane.

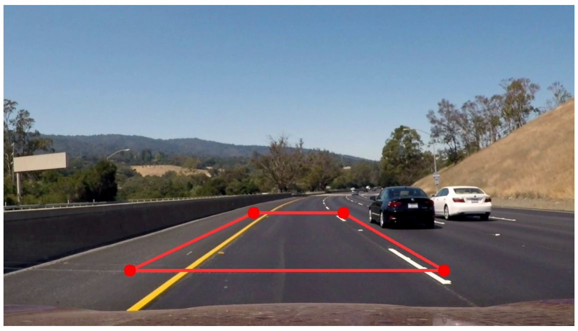
https://drive.google.com/file/d/1yuTZoD9jOfp5SHQNljtFGN87rhgB97u0/view?usp=sharing

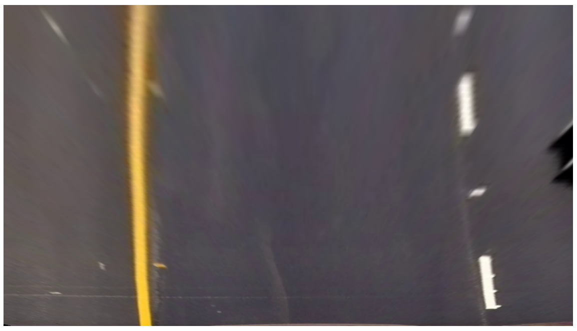
Problem 3: Curve detection

Detection Pipline

Step 1. Apply homography transformation to the region of interest into top view image

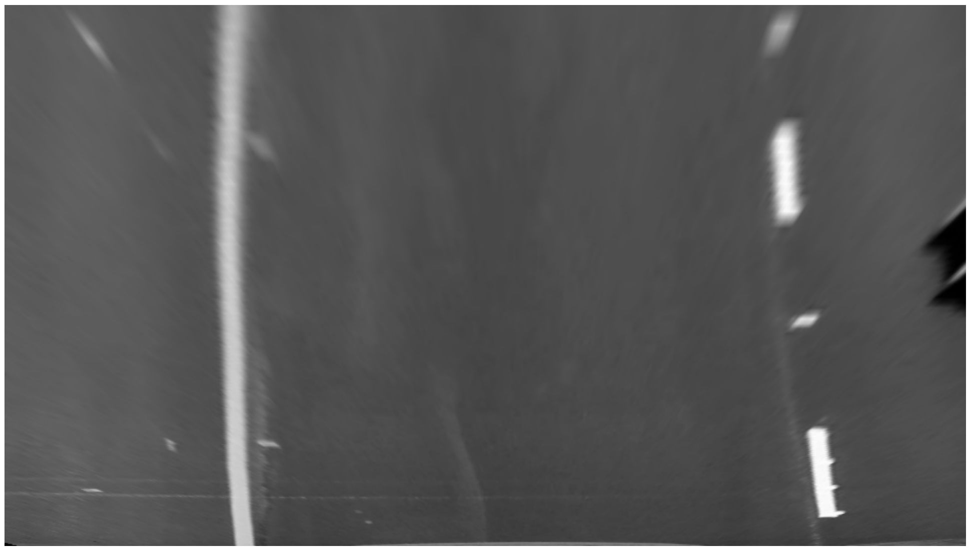
Find the region of interest by selecting a trapezoid at the bottom half of the image.



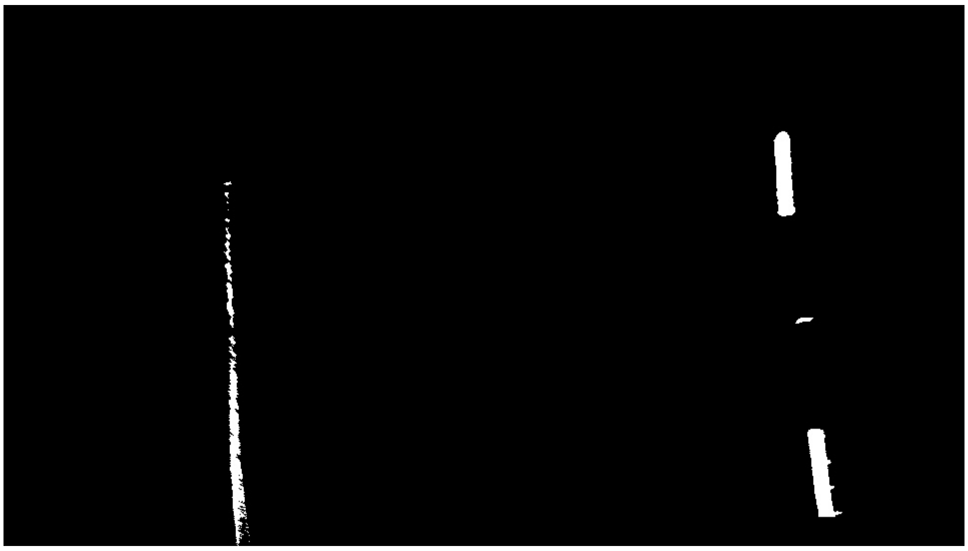


Step 2. White lane detection

a. Gray scale



b. Threshold



c. Remove white color that is not a lane by setting boundary

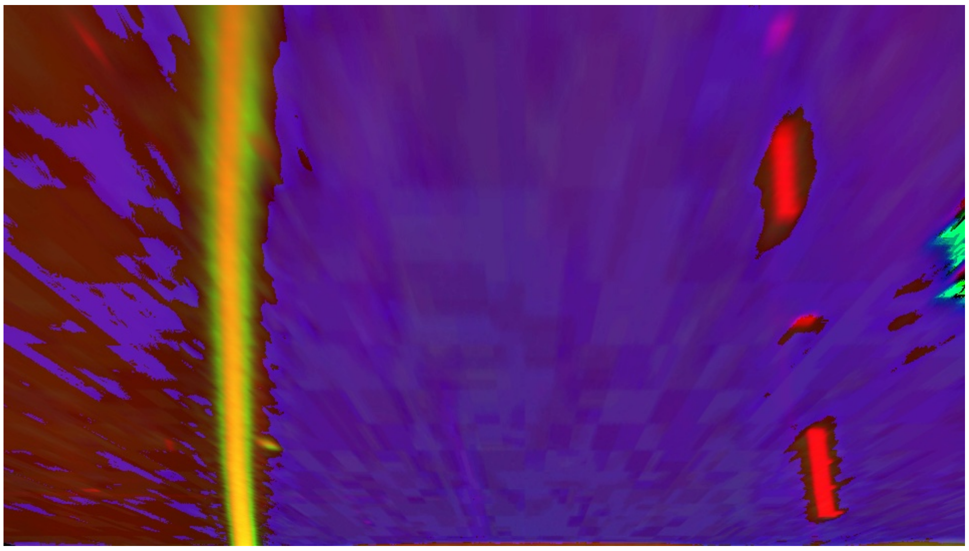
Step 3. Yello lane detection

a. Transform image to HSV Space

In HSV space each color is assigned by a particular number (The Hue).

Therefore it is more convenient to set a range for a certain color.

The rest of the parameters in HSV are the saturation which represent the amount of a color, and the Intensity which represent the brightness of the color.

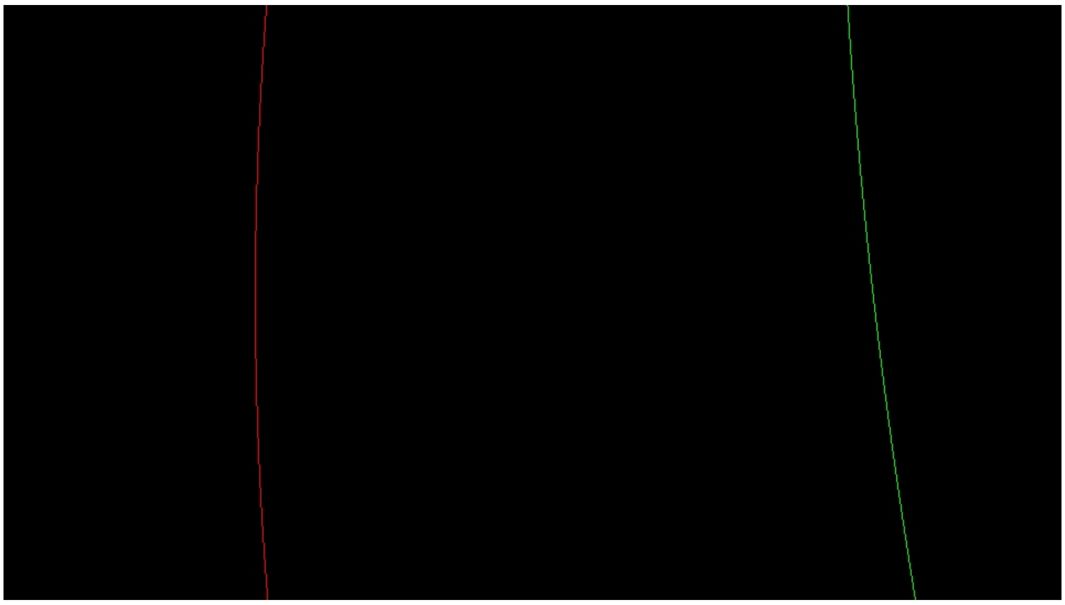


b. Find color between range that represent Yello

Step 4. Fit the white and yello lane with second order curve

a. Polynomial curve fitting

Fit all the points on the lane with a second order polynomial function.



b. Calculate curvature

Calculate the radius of curvature by the following equation.

$$R = \frac{\|(1 + \dot{f}^2(x))^{\frac{3}{2}}\|}{\|(\ddot{f}(t)\|}$$

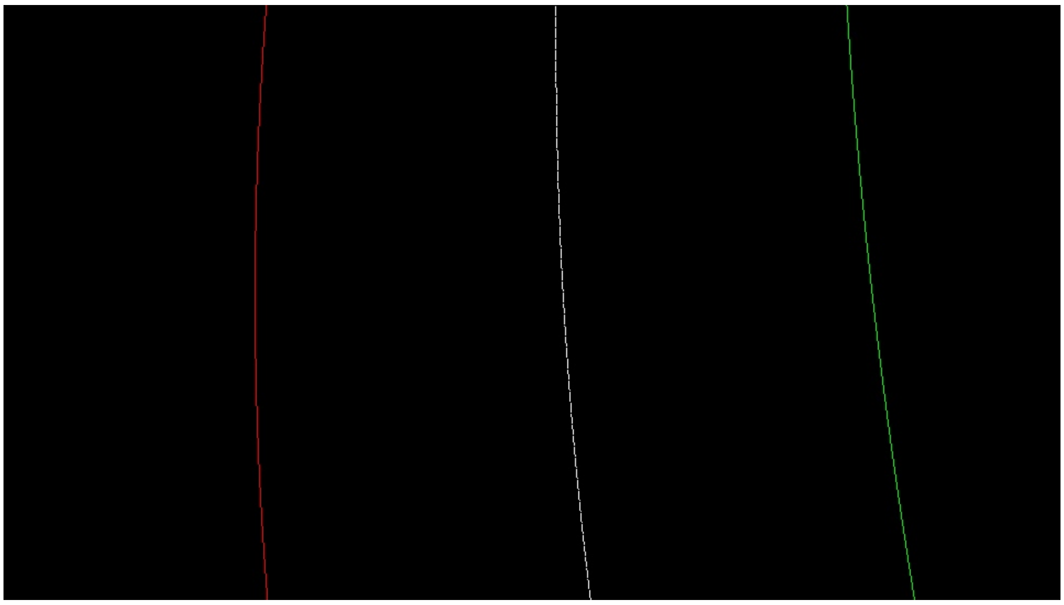
The radius of curvature is positive when center is at the right side of the curve.

The radius of curvature is negative when center is at the left side of the curve.

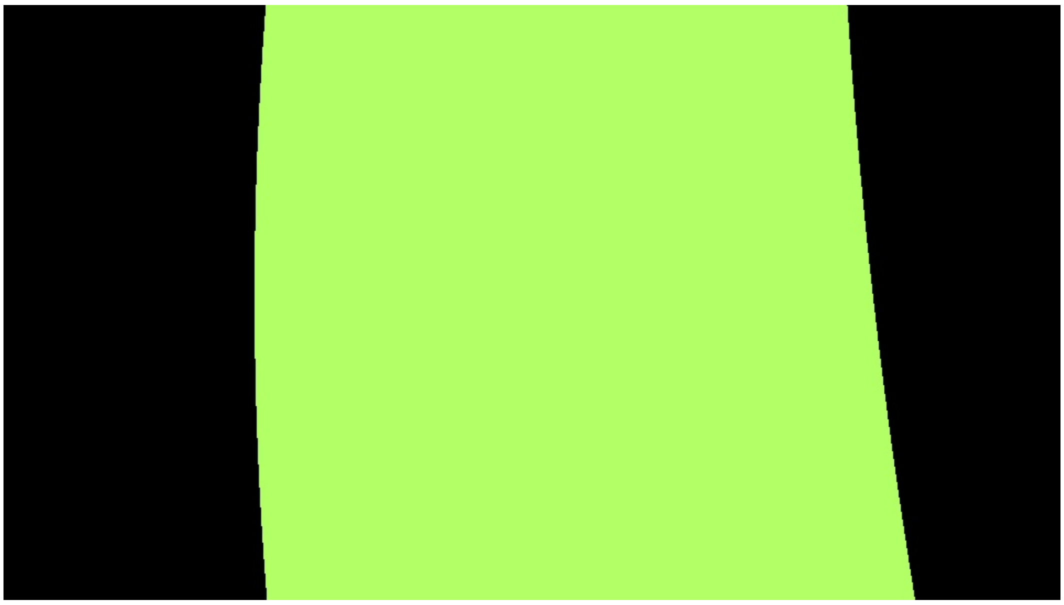
A positive radius curvature implies that the lane is turning right.

A negative radius curvature implies that the lane is turning left.

Step 5. Use the curves computed in step 4 to calculate an average curve



Step 6. Draw the polygon region formed by lane curves



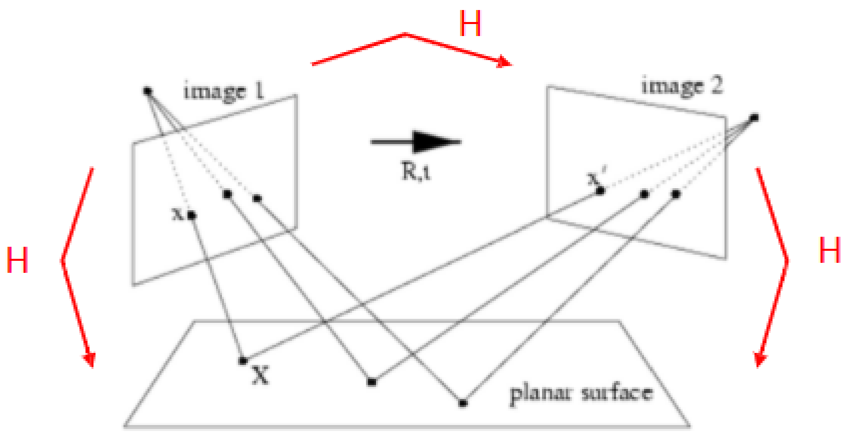
Step 7. Superimpose the polygon region back to the orignial image

Use the inverse of the homography matrix to transform the topview image back to the original view image.

Combine the transformed image with the original image by linear interpolating.

Homography Transformation

The homography transformation in the problem is the transformation between two camera view.



If the camera view is only transformed by rotation without translation, then the homography matrix is a 3 x 3 matrix.

The relationship between the coordinate of two camera view can be described as the follow equation.

$$scalar * \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} h_{00} & h_{01} & h_{02} \\ h_{10} & h_{11} & h_{12} \\ h_{20} & h_{21} & h_{22} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$scalar * x^{'} = \frac{h_{00}x + h_{01}y + h_{02}}{h_{20}x + h_{21}y + h_{22}}$$
$$scalar * y^{'} = \frac{h_{10}x + h_{11}y + h_{12}}{h_{20}x + h_{21}y + h_{22}}$$

Reorganize the equation to matrix format

$$\begin{bmatrix} x & y & 1 & 0 & 0 & 0 & -xx^{'} & -yx^{'} & -x^{'} \\ 0 & 0 & 0 & x & y & 1 & -xy^{'} & -yy^{'} & -y^{'} \end{bmatrix} \begin{bmatrix} h_{00} \\ h_{01} \\ h_{02} \\ h_{10} \\ h_{11} \\ h_{12} \\ h_{20} \\ h_{21} \\ h_{22} \end{bmatrix} = 0$$

By setting h22 to 1, this equation has 8 degree of freedom.
Therefore, it needs at least 4 pair of points in each camera view to solve the homography.

The matrix equation can be solved by using the SVD.

The h vector equals to the last column of V in SVD ($U\Sigma V^T$) equation of the matrix.

Pipeline Generalizability

Most of the steps in the pipeline is general enough.

The following steps still has spaces to improve

Step 1

The area of interest should be obtained by a more generalize way.
It can be set by choosing a trapzoid based on the detected lane.

Step 2.c

The boundary is not set automatically and optimally.

The surrounding area of the lane can be set by fisrt finding the histogram of the image then set boundary with a threshold to the area with highest histogram bar.

Video

The video shows the final superimpose image along with other lane detection images.

https://drive.google.com/file/d/1bAix04ZyhlGdIhr9xlc95Ps5ojPgHKt9/view?usp=sharing