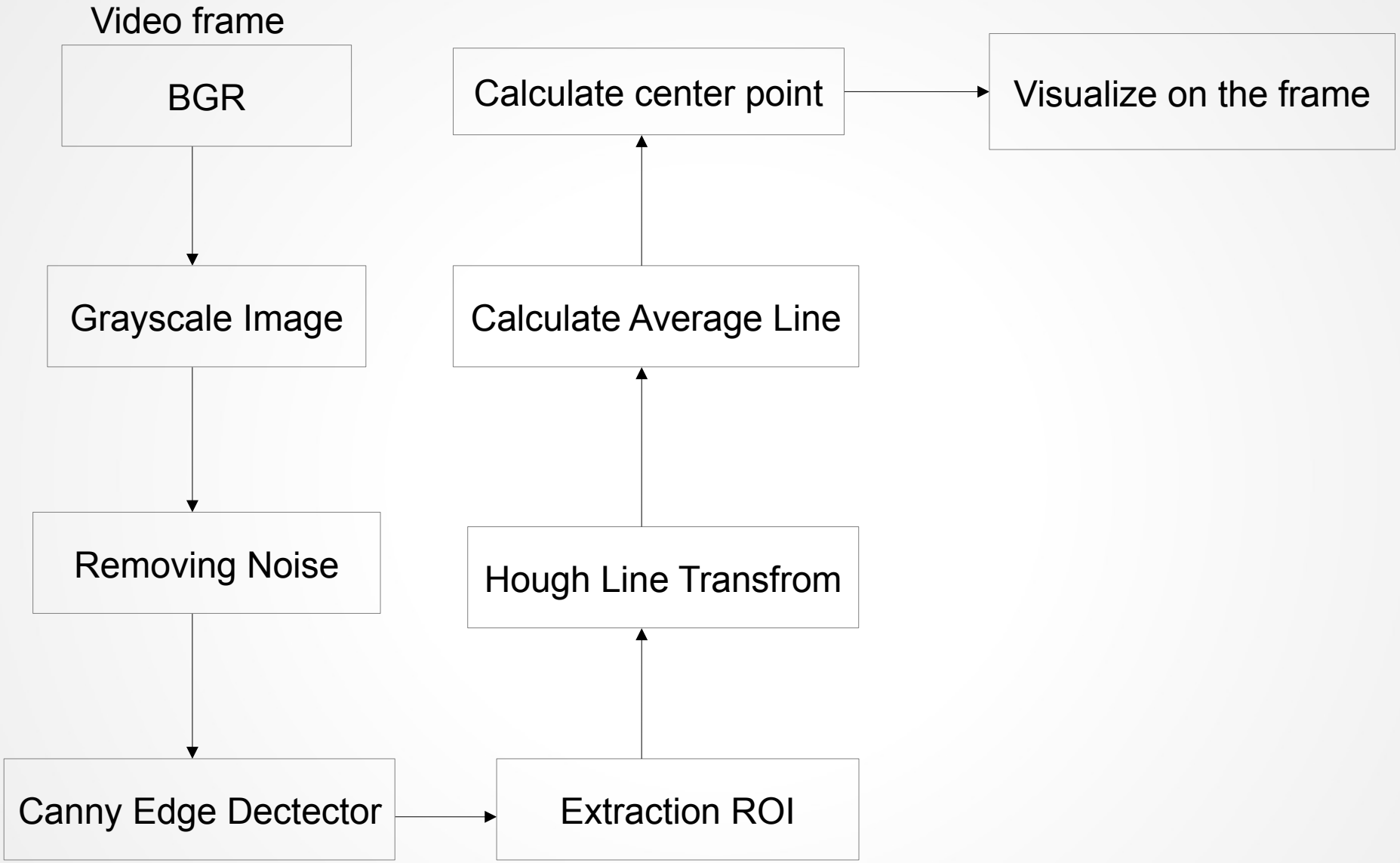
Middle Lane Detection

Le Tanh Sang

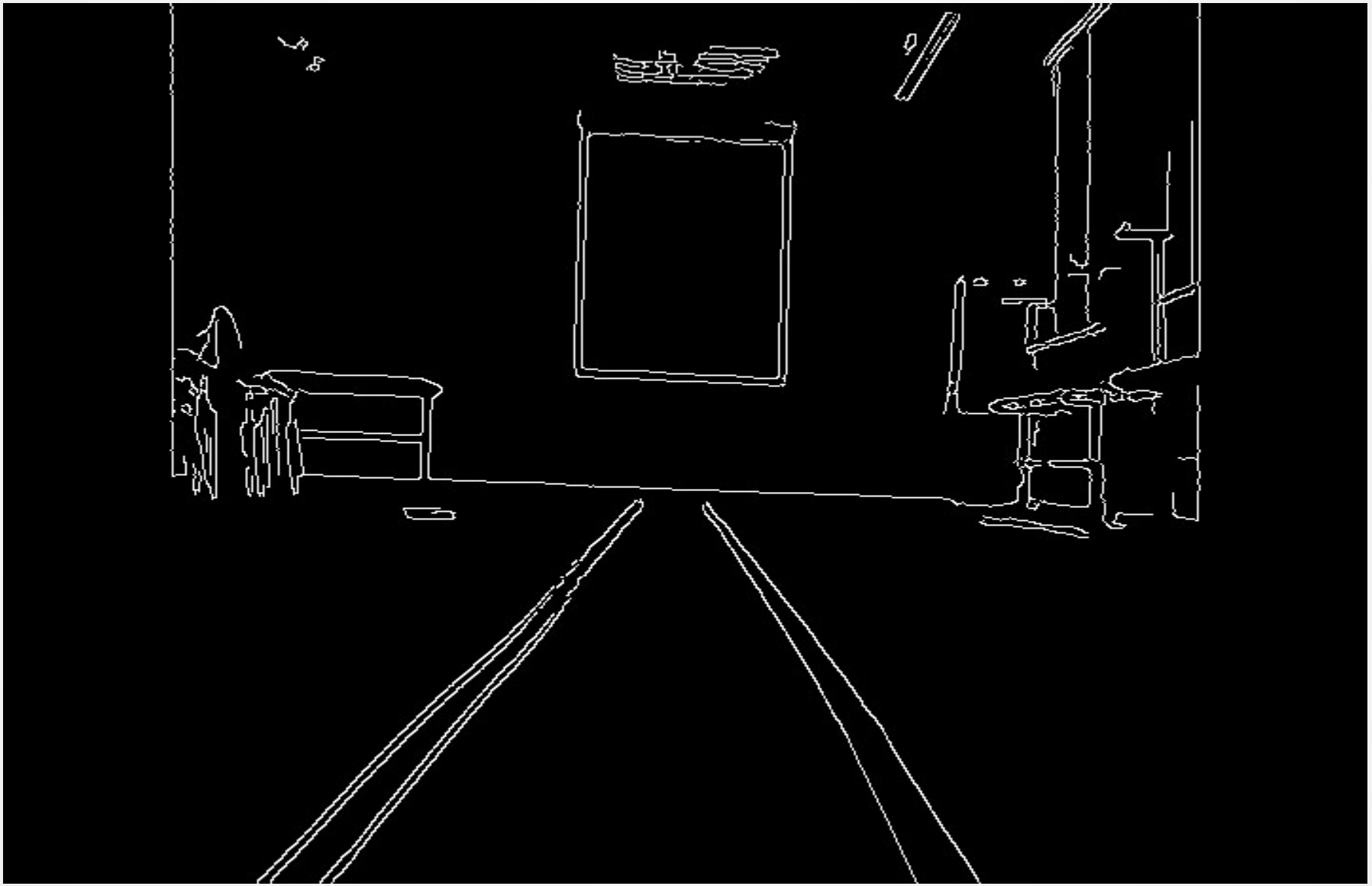
Image progressing Method



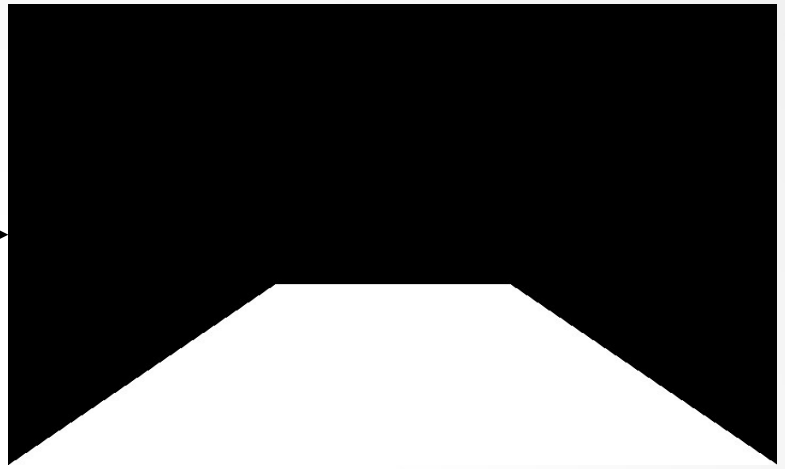
Grayscale

Removing noise

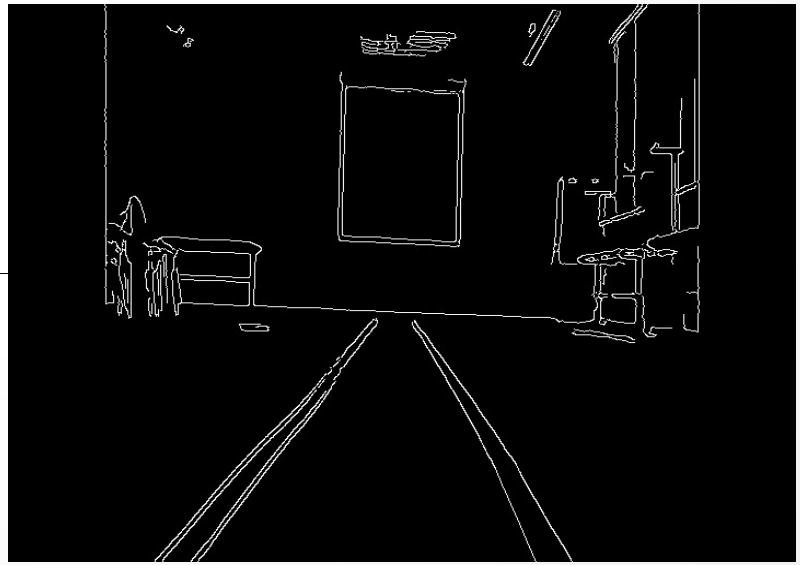
Canny edge detector

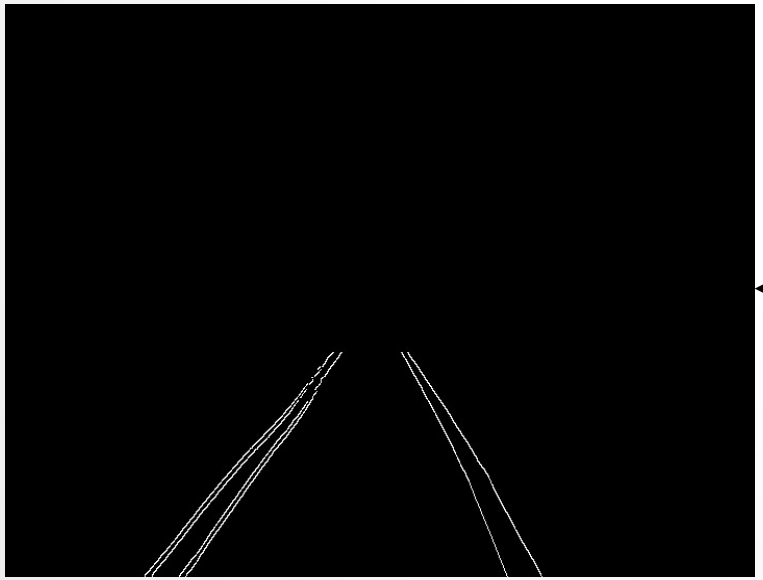


ROI – region of interest



X





Hough Line Transform

x1	y1	x2	y2

[[[141 573 234 454]]

[[194 535 277 417]]

[[174 572 228 494]]

[[183 510 250 430]]

[[178 569 253 455]]

[[387 350 464 474]]

[[246 436 307 361]]
……..]

https://docs.opencv.org/4.0.0/d6/d10/tutorial_py_houghlines.html

Calculate Average Line

Polynomial fit

$$y_1 = f(x_1) = a_0 + a_1 x_1$$

$$y_2 = f(x_2) = a_0 + a_1 x_2$$

$$y = f(x) = a_1 x + a_0$$

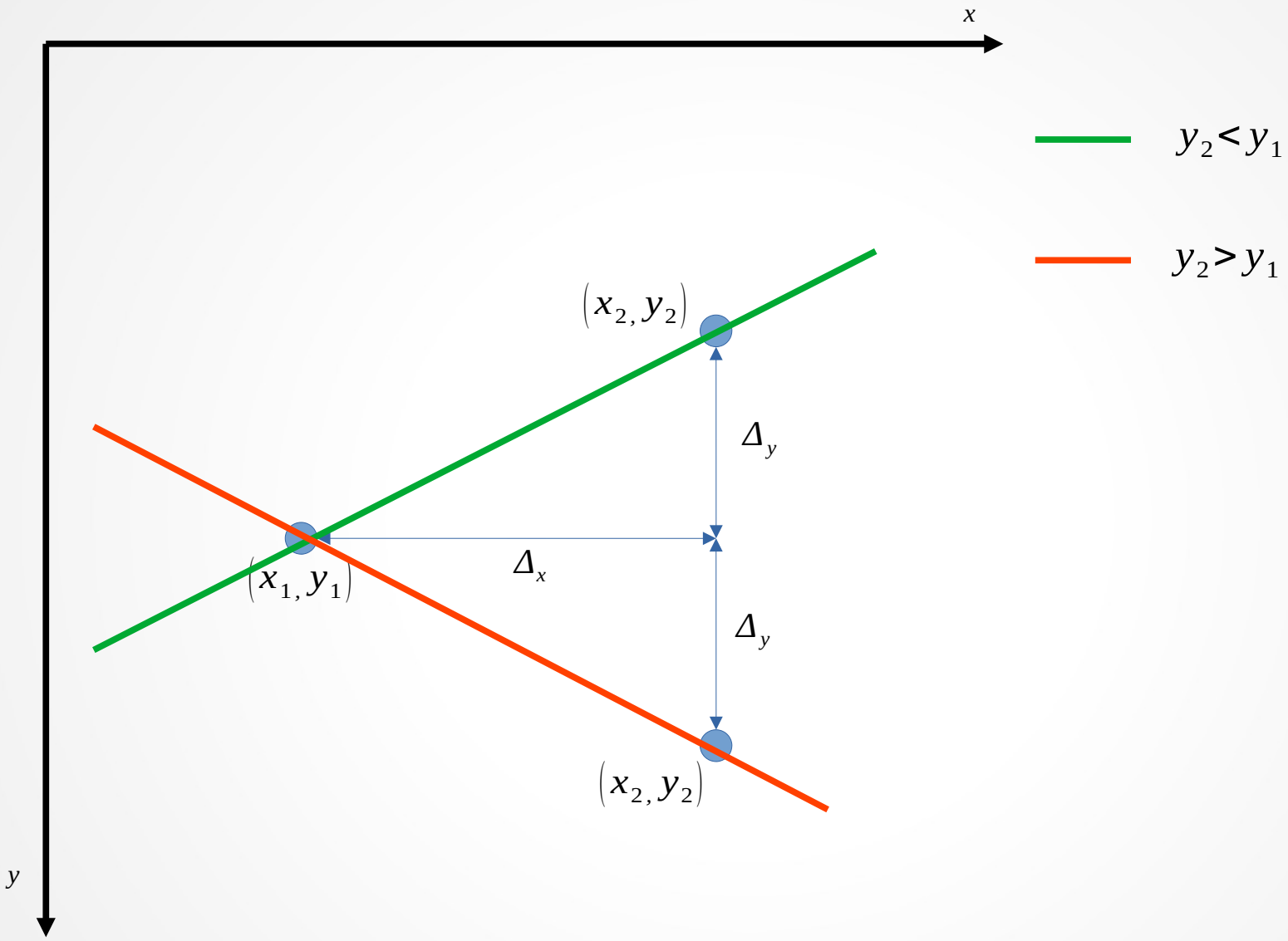
$$\begin{bmatrix} f(x_1) \\ f(x_2) \end{bmatrix} = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \end{bmatrix} \longrightarrow \begin{bmatrix} a_0 \\ a_1 \end{bmatrix} = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \end{bmatrix}^{-1} \begin{bmatrix} f(x_1) \\ f(x_2) \end{bmatrix}$$

For example: $(x_1 = 141, y_1 = 573), (x_2 = 234, y_2 = 454)$

$$\begin{bmatrix} a_0 \\ a_1 \end{bmatrix} = \begin{bmatrix} 1 & 141 \\ 1 & 234 \end{bmatrix}^{-1} \begin{bmatrix} 573 \\ 454 \end{bmatrix} = \begin{bmatrix} 753.4193548387096 \\ -1.279569892473119 \end{bmatrix}$$

$$\longrightarrow \quad y = -1.279569892473119\, x + 753.4193548387096$$

Calculate Average Line



Calculate Average Line

We are going to store these (a_1, a_0) of all lines which have a slope $a_1 < 0$ into left list , while $a_1 > 0$ into right one

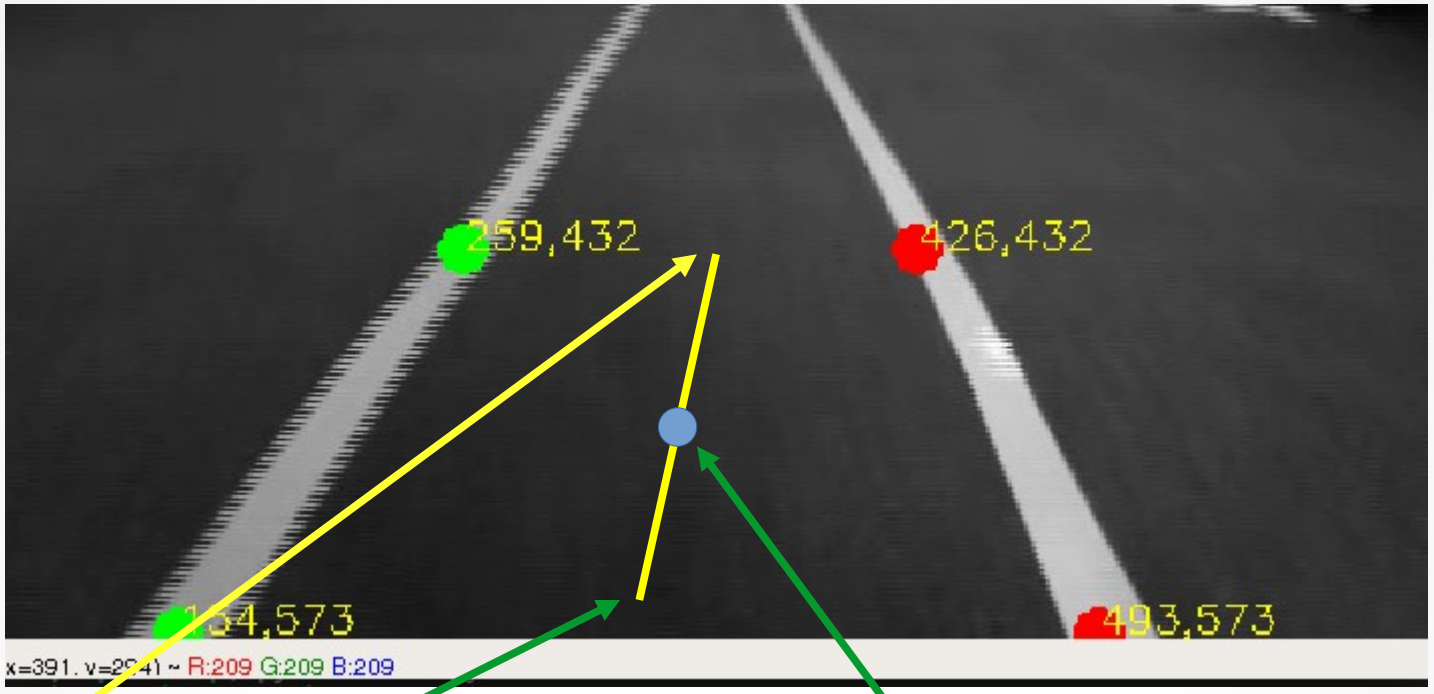
```
In [1]: left = [
   ...:     (1,2),
   ...:     (3,4),
   ...:     (5,6)
   ...: ]

In [2]: left
Out[2]: [(1, 2), (3, 4), (5, 6)]

In [3]: import numpy as np

In [4]: np.average(left, axis=0)
Out[4]: array([3., 4.])
```

$$y = \bar{a}_1 x + \bar{a}_0$$

Calculate Average Line

Now we just replace y in the formula for calculating value for x

Calculate Center Point



center_top.x = (426-259)/2 + 259
center_bottom.x = (493-254)/2 + 254

The lane's middle point

center_average_x = (center_top.x – center_bottom.x)/2 + center_bottom.x
center_average_y = (center_bottom.y - center_top)/2 + center_top.y

Visualize on the frame

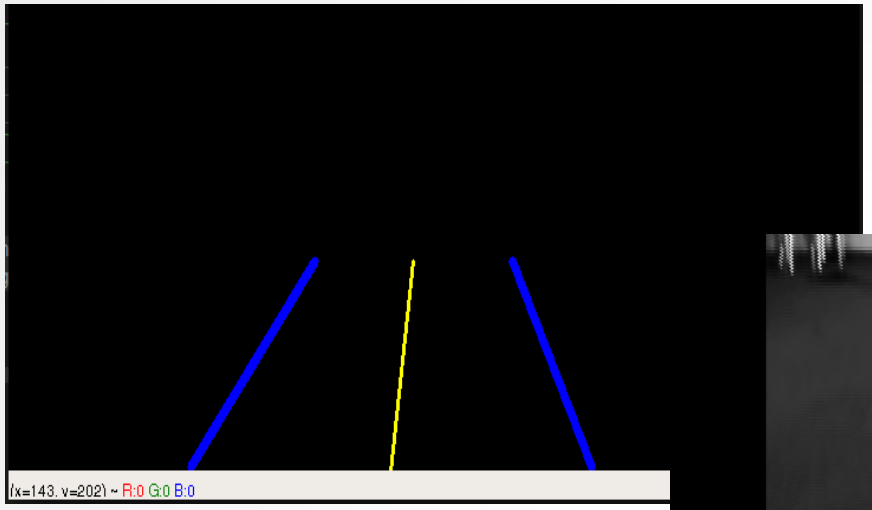


Image Blending

$$dst = \alpha \,.\, image_1 + \beta \,.\, image_2 + \gamma$$

https://docs.opencv.org/4.0.0/d0/d86/tutorial_py_image_arithmetics.html