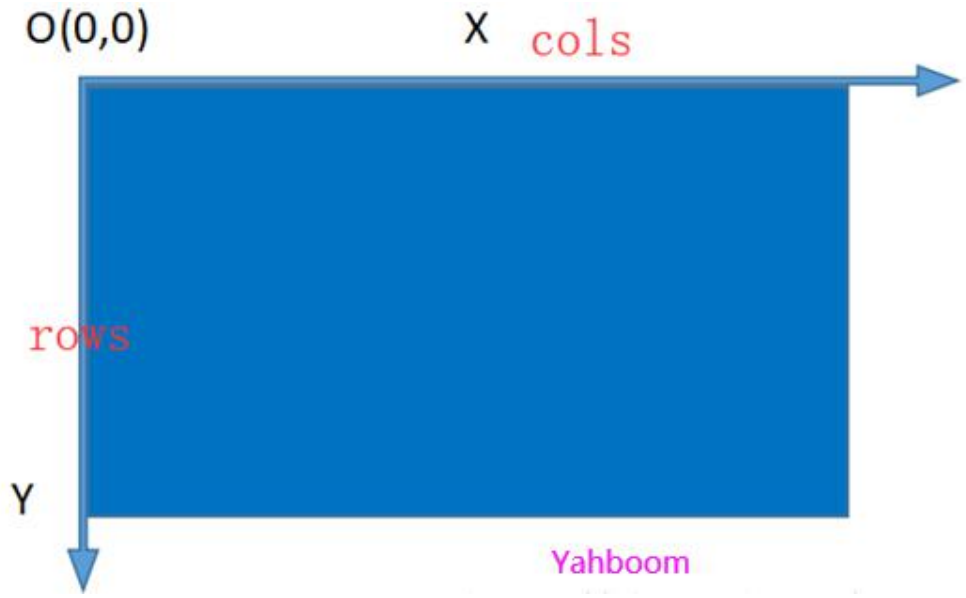
Path: /home/pi/Yahboom_Project/Raspbot/3.AI Vision course/12.Autopilot/Autopilot.ipynb

1. OpenCV coordinate system

row == heigh == Point.y

col == width == Point.x

Mat::at(Point(x, y)) == Mat::at(y,x)



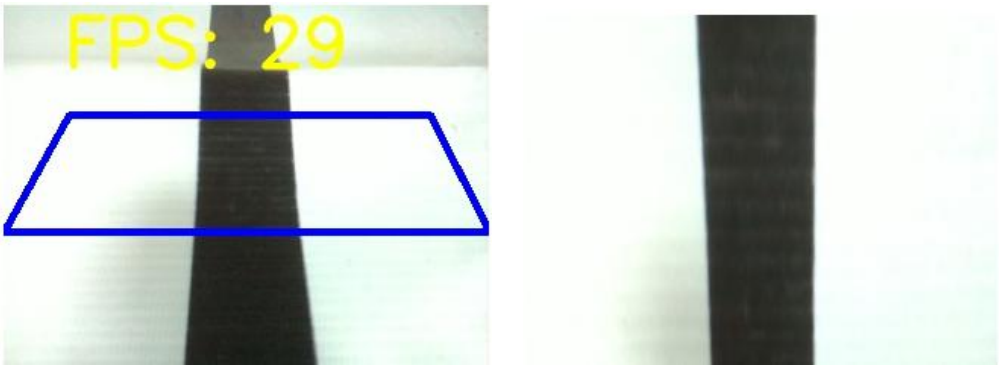
Obtain image: ret, frame = image.read()

Image resize: frame = cv2.resize(frame,(320,240))

2.Perspective transformation

Because the picture seen by the camera is distorted, we need to transform the area in front of the body into a top view through perspective transformation. Then zoom in to the entire picture area.

As shown in the figure below, the left is the original camera picture, and the right is the picture after perspective transformation.



【original camera picture】 【picture after perspective transformation】

Function:

matSrc = np.float32([[0, 149], [310, 149], [271, 72], [43, 72]])

```
matDst = np.float32([[0,240], [310,240], [310,0], [0,0]])
matAffine = cv2.getPerspectiveTransform(matSrc,matDst)# mat 1 src 2 dst
dst = cv2.warpPerspective(frame,matAffine,(320,240))
```

Then, we can draw the rectangle based on the perspective coordinates.

```
# Draw the rectangle box
pts = np.array([[0, 149],    [310, 149], [271, 72], [43, 72]], np.int32)
pts = pts.reshape((-1, 1, 2))
cv2.polylines(frame, [pts],True, (255, 0, 0), 3)
```

3.Binary image

The core idea of binarization is to set a threshold, and the value greater than the threshold is 0 (black) or 255 (white), making the image called black and white. The threshold can be fixed or adaptive.

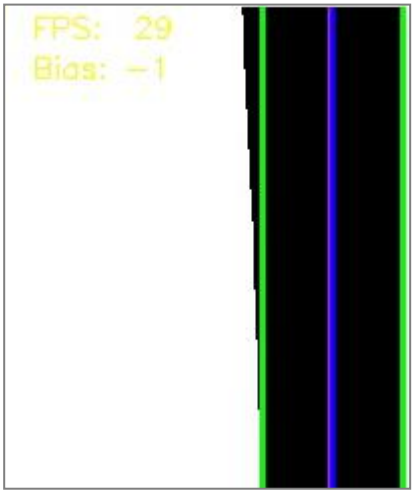
Threshold function is provided in Python-OpenCV:

cv2.threshold（src, threshold, maxValue, method）

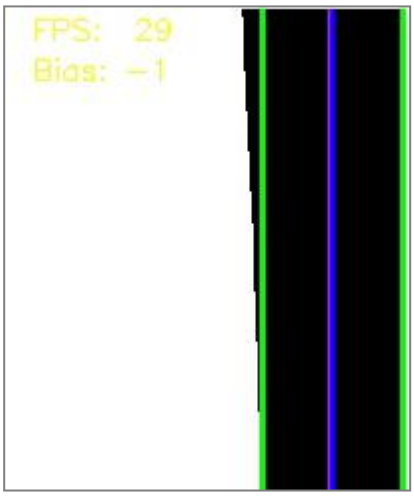
eg:

```
dst_gray = cv2.cvtColor(dst, cv2.COLOR_RGB2GRAY)
dst_retval, dst_binaryzation = cv2.threshold(dst_gray, 110, 255,
cv2.THRESH_BINARY)
```

After the perspective transformation of the picture, the converted image is shown below.



4. Get the edge of the black line and draw the line



The purpose of drawing the line is to facilitate the user to check whether there is a problem in the above process.
Draw the position of the two edges of the black line, and the position of the center line of the image and the center of the black line.

- The image size after perspective is still 320*240, so the centerline of the image is: midpoint: 320/2-1=159 Or: midpoint = np.int(histogram.shape[0]/2)

- Search on the left side of the black line: take the minimum index of the left image, and implement the following function: leftx_base = np.argmax(histogram[:midpoint], axis = 0)

- Search to find the right line of the black line: Invert the image and take the index of the minimum value on the left of the image, and then subtract the index from the image width:
- rightx_base = np.argmin(histogram[::-1][:rightpoint], axis = 0) #Reverse the histogram and take the rightmost value
  rightx_base = 319-rightx_base

- Calculation of actual midline position: (left line + right line)/2
- lane_center = int((leftx_base + rightx_base)/2) #The middle position of the left and right lines

- Image centerline: cv2.line(dst_binaryzation, (159,0), (159,240),(255,0,255),2) #purple
- Left line: cv2.line(dst_binaryzation,(leftx_base,0),(leftx_base,240),(0,255,0),2) #green
- Righht line: cv2.line(dst_binaryzation,(rightx_base,0),(rightx_base,240),(0,255,0),2) #green
- Actual midline:

cv2.line(dst_binaryzation,(lane_center,0),(lane_center,240),(255,0,0),2) #blue

5. Calculate the offset

After finding the white line of Autopilot in step 5, we can know the size of the body deviation according to the difference between the actual center line and the center line of the image, so that the car can be controlled to correct the deviation to achieve the purpose of Autopilot.

Calculation method:

Bias = midpoint-lane_center

According to the sign and magnitude of the offset value, we limit it to -20 ~ 20, which means that the maximum speed of the steering is 20, to prevent the steering from losing the white line of the track too quickly.

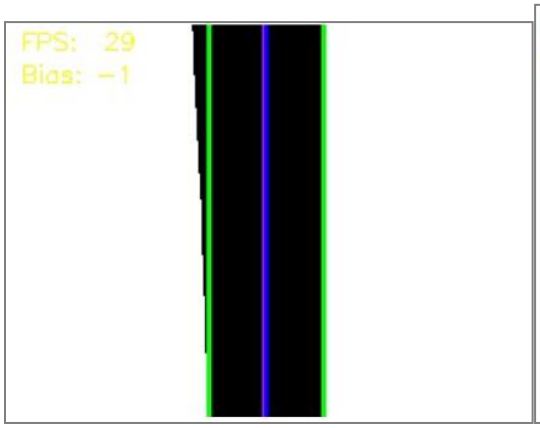
Steering angle PID adjustment:

```
Z_axis_pid.SystemOutput = Bias
Z_axis_pid.SetStepSignal(0)
Z_axis_pid.SetInertiaTime(0.5, 0.2)
```

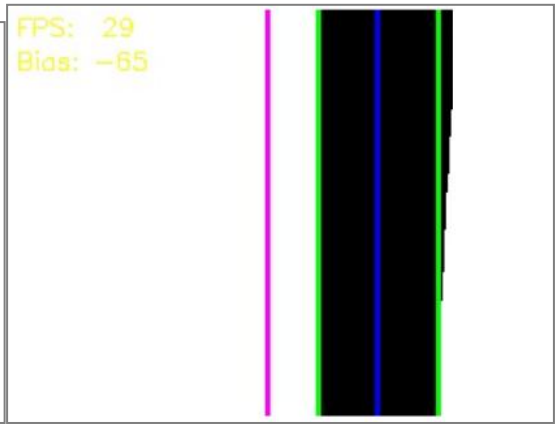
After the PID calculation, the PID output is also limited to a maximum of ± 20.

```
if Z_axis_pid.SystemOutput > 20:
    Z_axis_pid.SystemOutput = 20
elif Z_axis_pid.SystemOutput < -20:
    Z_axis_pid.SystemOutput = -20
```

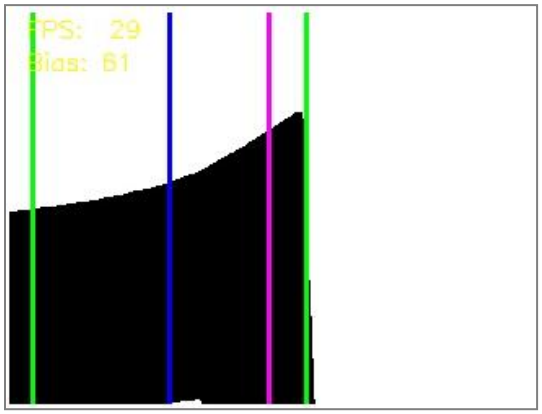
6. Control the robot to complete autopilot



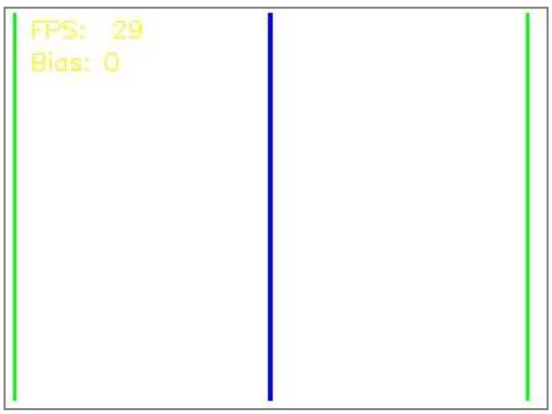
Pic-1



Pic-2



Pic-3



Pic-4

Pic-1: Slight deviation, go straight.
Pic-2: A large right deviation is corrected by turning right.
Figure 3: The deviation caused by the left right angle is calibrated by turning in place.
Figure 4: The picture is lost when passing a right angle, and the direction of rotation is judged according to the previous state before detection.

About code:

```
            if leftx_base == 0 and rightx_base == 319:
                if prev_left > prev_right:
                    car.Control_Car(-70, 60)
                elif prev_left < prev_right:
                    car.Control_Car(70, -70)
                prev_left = 0
                prev_right = 0
            else:
                if Bias > 3:
                    #prev_left = 1
                    #prev_right = 0
                    if Bias > 140:
                        car.Control_Car(-70, 60)
                        prev_left = 0
                        prev_right = 0
                    else:
                        car.Control_Car(45+int(Z_axis_pid.SystemOutput),
45-int(Z_axis_pid.SystemOutput))
                    time.sleep(0.001)
                elif Bias < -3:
                    #prev_right = 1
                    #prev_left = 0
                    if Bias < -140:
```

```
                    car.Control_Car(60, -70)
                    prev_left = 0
                    prev_right = 0
                else:
                    car.Control_Car(45+int(Z_axis_pid.SystemOutput),
45-int(Z_axis_pid.SystemOutput))
                time.sleep(0.001)
            else:
                car.Car_Run(45, 45)
        if left_sum != right_sum:
            if left_sum < right_sum:
                prev_left = prev_left + 1
            elif right_sum < left_sum:
                prev_right = prev_right + 1
```