Following Line with Centroid Detection

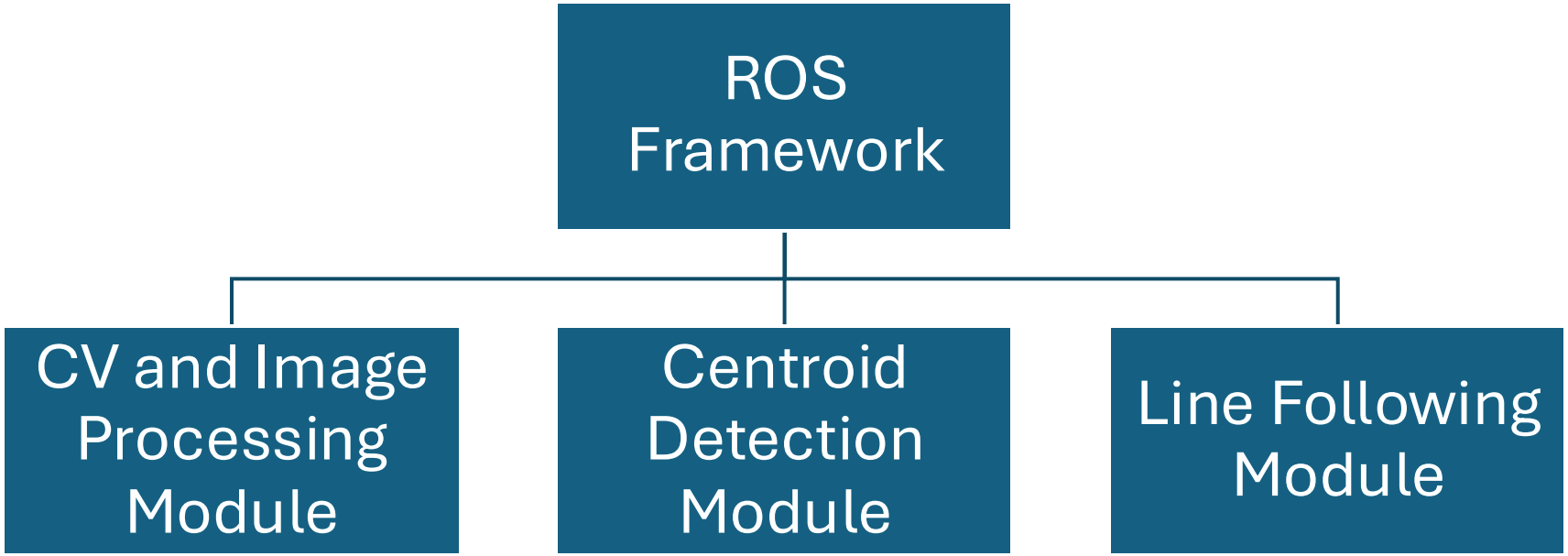
A Solution to Follow the Line based on turtlebot3

Pang (Jeff) Liu

Solution Introduction

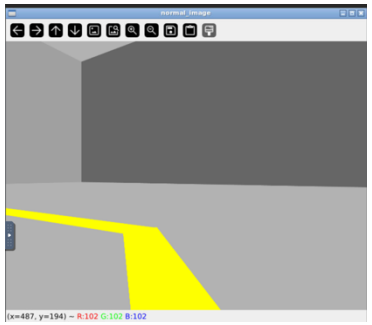
- This solution achieve the goal that let the robot follows the line.
- The core idea is finding the line and create a “ball” in the middle of the line, then let the robot follow that “ball”. This is called Centroid Detection.
- The detection is based on color and use OpenCV to process images.
- The solution also contain some special conditions, including avoid obstacles.

Solution Framework

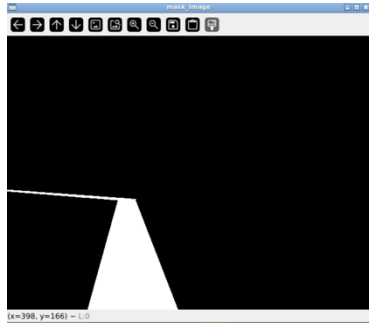


CV and Image Processing

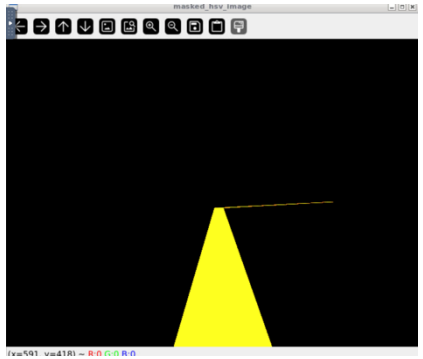
Use camera to receive the images



Receive the image from the camera and transform to HSV format



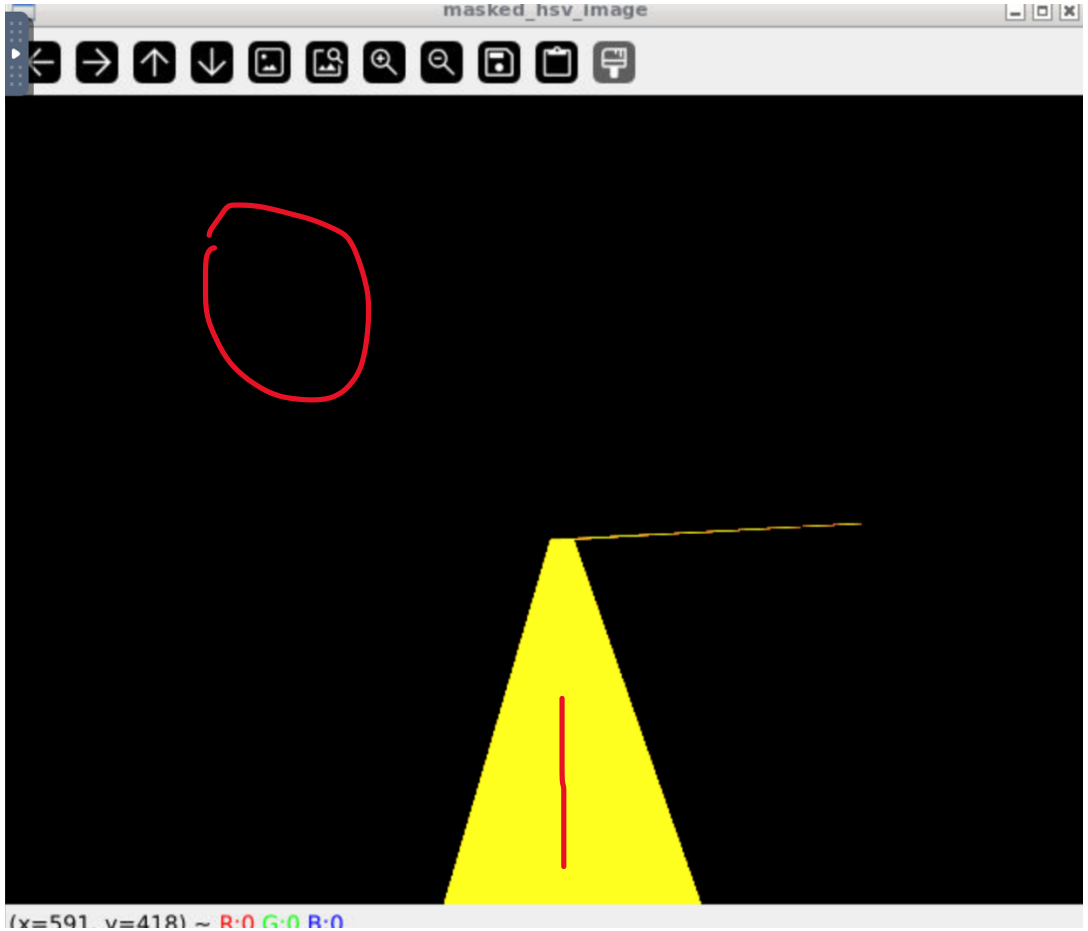
Use **Color Segmentation** to extract yellow color, then use color thresholding to generate a binary mask

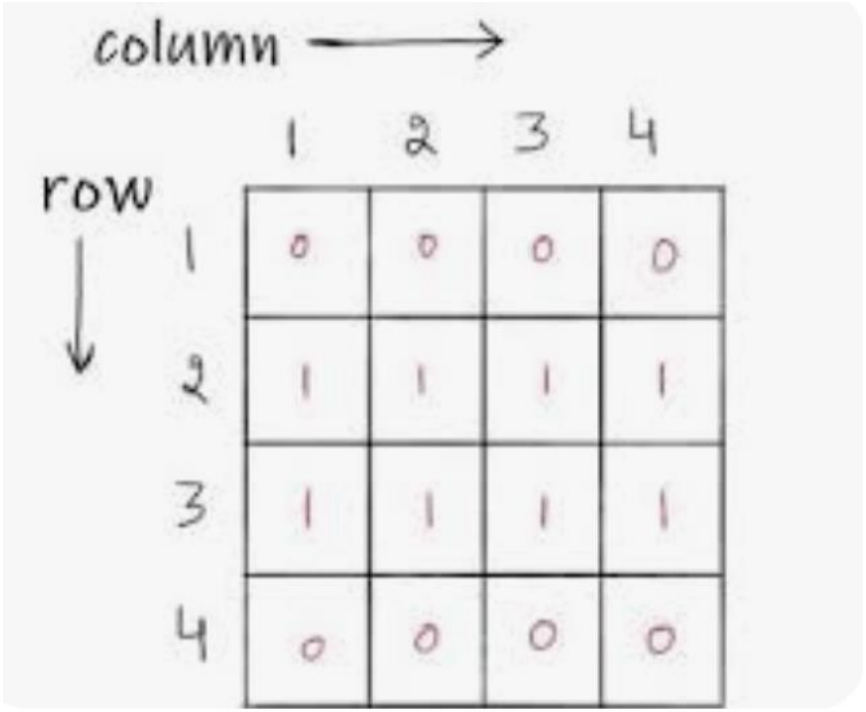


Apply the mask to get an image with only yellow and black

Centroid Detection (1)

Now the image contains only two colors: yellow and Black, we can assume that black is “0” and yellow Is “1”. This is why we need a binary image, because We need a binary matrix





Centroid Detection (2)

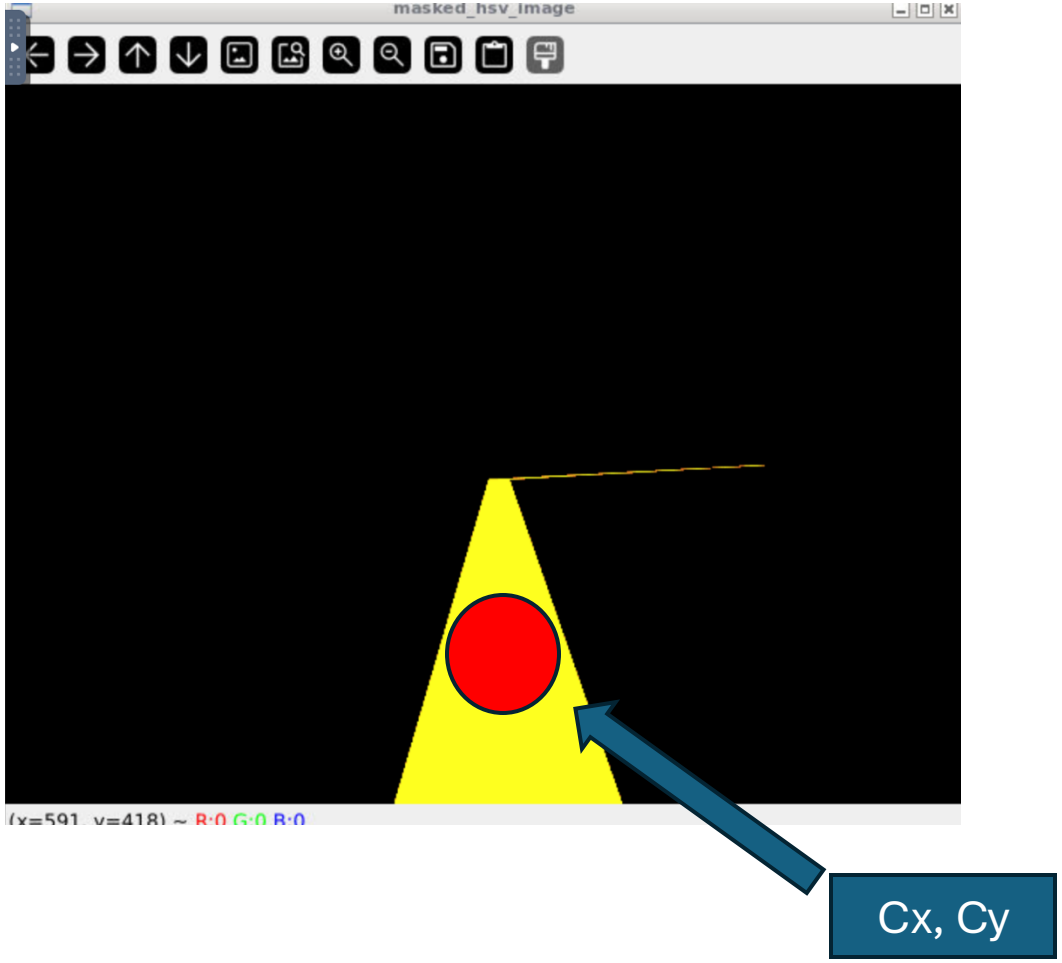
- We can assume that the binary image contains many 0 and 1
- This is also the distribution of the pixels, 0 is black and 1 is yellow
- The distribution can be recognized as **Image Moments**
- In the yellow area:
  - The sum is called **Zero-Order Moment (M00)**
  - The weighted sums along the x and y axes are **M10 and M01**

$$M_{00} = \sum_x \sum_y I(x, y)$$

$$M_{10} = \sum_x \sum_y x \cdot I(x, y)$$

$$M_{01} = \sum_x \sum_y y \cdot I(x, y)$$

Centroid Detection (3)



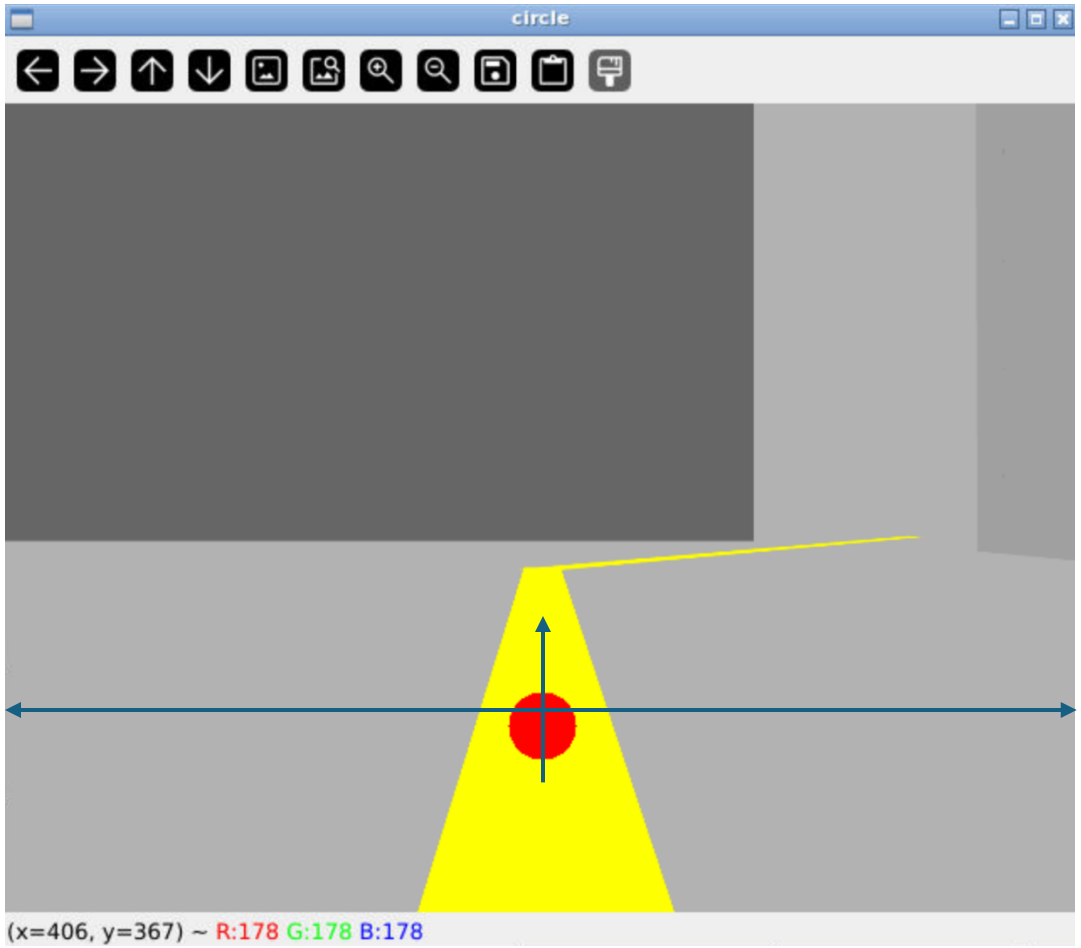
With M00, M10, and M01, we can calculate the place
Of the Centroid of the yellow area, which is (cx, cy):

$$cx = \frac{M_{10}}{M_{00}} = \frac{\sum_x \sum_y x \cdot I(x,y)}{\sum_x \sum_y I(x,y)}$$

$$cy = \frac{M_{01}}{M_{00}} = \frac{\sum_x \sum_y y \cdot I(x,y)}{\sum_x \sum_y I(x,y)}$$

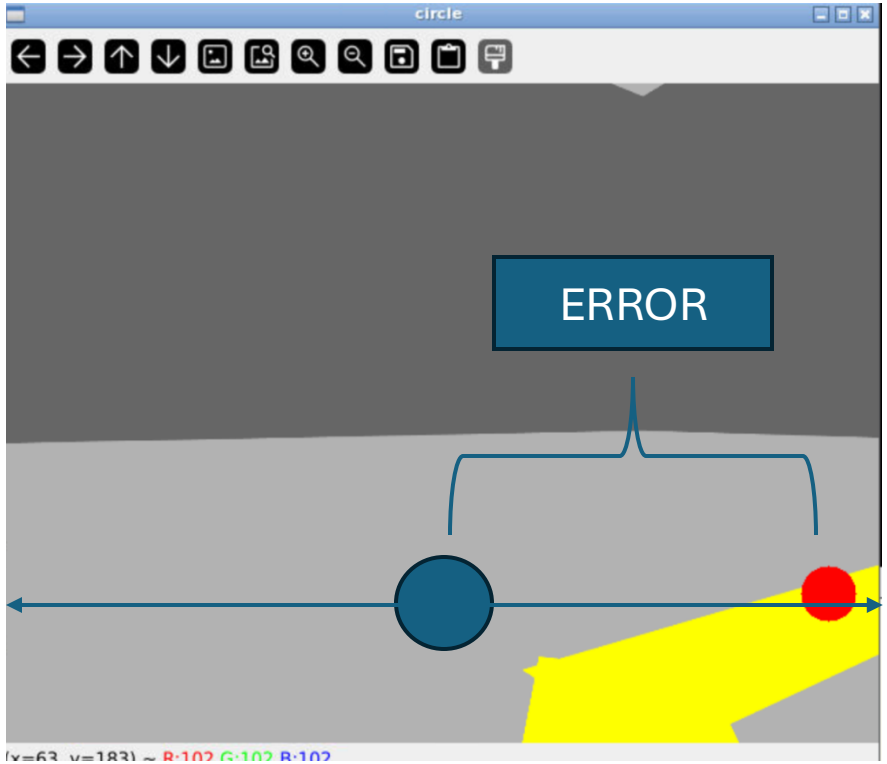
Line Following Module (1) – Make direction

- Now we get the Centroid (cx, cy)
- Our goal is that let the robot “follow the ball”
- But the robot cannot fly, it can only move on the ground
- So we only need to let the robot’s direction match the “ball”
- Which means the direction should be the middle of the image



Line Following Module(2) – P Control (2)

- Calculate the error between the middle, if the ball is not at the middle



$$\text{error} = cx - \frac{\text{width}}{2}$$

Then We can use P Control to control the angular speed,
To allow the robot move toward the “ball” (centroid)
This task is not complex, so I only use P to control, not all
Three PID. (I test PID, not good, only P is better)

$$\text{angular\_speed} = -K_p \cdot \text{error}$$

Line Following Module(3)
- Multi-State Decision Implementation(1)

- The remaining part is the algorithm of the line following.
- According to a normal test circumstance, there are several conditions we need to consider, and I summarize them as three states:
- State1 (line): detect the yellow line
- State2 (obstacle): detect the obstacle
- State3 (explore): didn’t find the yellow line or obstacle, start exploring mode

Line Following Module(3)
- Multi-State Decision Implementation(2)

Then the algorithm will be:

- If line:
  - Robot follows the centroid direction using P control
- Else if not line & not obstacle & not explore:
  - Robot rotates in place to find the line, if unsuccessful after 1-2 circles, change explore state into True
- Else if not line & not obstacle & explore:
  - Robot moves forward in a straight line until it finds the line or an obstacle
- Else: (not line & obstacle):
  - Use LeftWallFollower() to follow the left_side wall or avoid the obstacle (note: the LeftWallFollower() is the previous functional node script)

Demonstration

With the algorithm build above, the robot can:

- Follow the yellow line, whether it is straight, curve, or a turn
- When the robot encounter an obstacle, the robot will avoid it and continues following the yellow line
- If the yellow line disappears and there is no obstacle, the robot turns around and follows the path back
- If the robot starts in an area with no line nearby, it moves forward, searching for the line

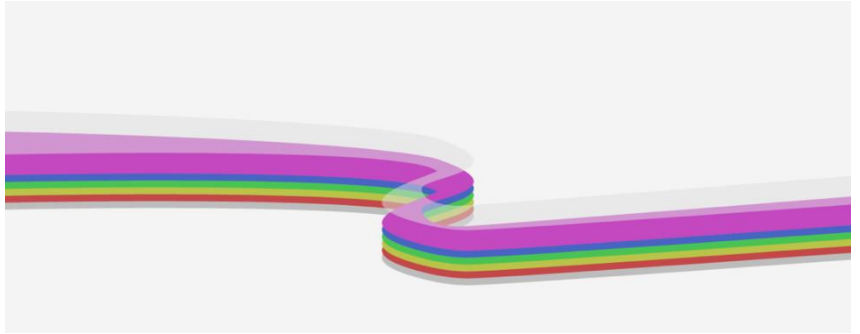
Test Result

- Test: 20 Times, with different obstacles and starting positions
- **Successful following line rate: 100%**
- **Successful turning back at the end of the line: 100%**
- **Successful avoid the obstacle and find other lines: 100%**
  - Perfectly avoid the obstacle and following the expect remain line: 70%
  - Avoid the obstacle but follow the coming line: 30%
- The result represents this algorithm is a suboptimal solution.

Challenges of this module

- The image processing method can solve every single kind of color, but what if the line is multiple colors or just a road?
- Also, the robot cannot always successfully move to another line when avoiding obstacles, is also because of this, because the robot can only detect the yellow line, but cannot distinguish or remember them!
- To solve this, we need a better and more reliable system to detect the line or the road, which could be into machine learning, deep learning and reinforcement learning.

  (my next focus!)



Thanks!