Shanghai Jiaotong University

Advanced ICT Project 1

Traffic Sign Detection

Author:
WANG Lei
118260910044

Supervisor:
LI Hao

November 24, 2019

1 Preparation

The code was written in python. During this project, the library of **opencv, numpy, matplotlib, tqdm, math and random** are necessary. If some of these libraries do not exist, you need to use the **pip** command to install them.
How to run the code ? In the terminal,
python TrafficSignDetection.py –image [Name of the image]
You can observe the result of detection by an image saved as ans.jpg.

2 Image Processing

The object of our project is the detection of circle traffic sign. The basic idea of our project is to simplify the image and detect the circle with a proper radius.
First we perform image denoising, although gradient operators can be used to enhance the image, essentially by enhancing the edge contours, but they are greatly affected by noise. The image denoising is necessary, because noise is a place where the gray level changes greatly, so it is easy to be recognized as a pseudo edge. As shown in Figure 2 and Figure 3, after denoising the results of the edge detection (by Canny) significantly remove some irrelevant information.

Figure 1: Original Image

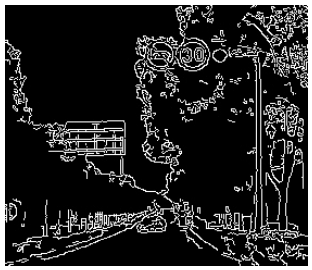

Figure 2: Edge detection without disnoising

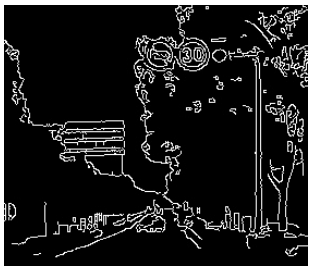

Figure 3: Edge detection with disnoising

The opencv has a useful function, Canny, Which can detect the edge easily. All of edge points are remarked as 255. By using the function np.where(), we can find the coordinate of all points on the edge.

3 Circle Detection

Circle detection is the main work in our project. By using the circle hough transform, the space of circle center and radius can be constructed. For each point on the edge, we take this point as the center, and calculate a vote value for each point on the graph between the minimum radius and the maximum radius.

The vote value is:

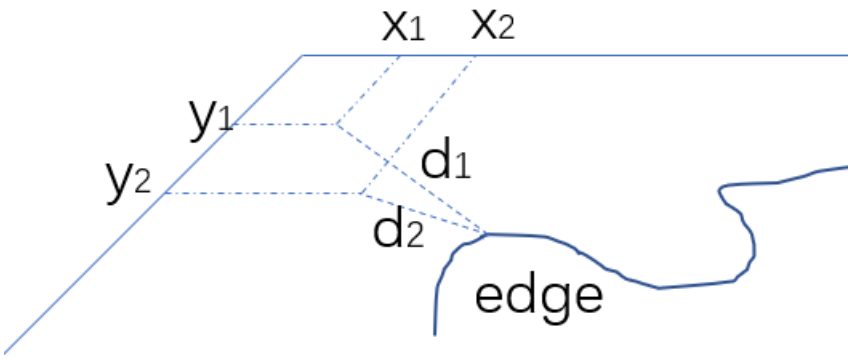

Figure 4: How to construct the map

$$value = \frac{1}{distance}^{1}$$

The number of points on the circle is proportional to the circumference. If we don't divide the distance, the algorithm will return a circle with a larger radius. As shown in Figure 5 and Figure 6[2], value divided by distance can better construct the space of the center of the circle and radius.

For each point on the edge, we do the above to get a space of center, radius and their corresponding value.

[1]Euclidean distance between the center of the circle and the point

[2]The parameters of Canny : 90, 180, the minimum and maximum radius : 15, 20

Figure 5: Divide by distance

Figure 6: Without operation

4 Result

There are four test images to detect the circle traffic sign. After testing many times, we find that the suitable parameters of Canny are 103, 180 for all cases. However, each picture has its own maximum radius and minimum radius. Figure 7 are 25, 15 ; Figure 9 are 25, 15 ; Figure 11 are 25, 15 ; Figure 13 are 25, 15.

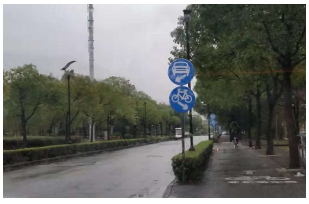

Figure 7: Original Image

Figure 8: Traffic Sign Detection

Figure 9: Original Image

Figure 10: Traffic Sign Detection

Figure 11: Original Image

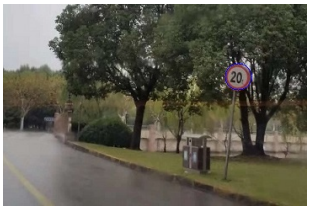

Figure 12: Traffic Sign Detection

Figure 13: Original Image

Figure 14: Traffic Sign Detection

5 Future

In my code, I didn't consider the distance between two circle traffic sign. Because of the case of concentric circles, it might cause some problems. This situation could be reduced by changing the maximum radius and minimum radius. However, we will add a parameter to describe it in the next version. Besides, during this project the standard hough transform was used to detect the center of the circle and radius. The accumulator required a lot of space and the amount of operation was huge. We will use the random hough transform to achieve our goal.

6 Appendix : Code Explication

All functions used in my code are presented here:

def image_process :

- input : Image name

- process : cv2.GaussianBlur (denoising), cv2.Canny (edge detection), cv2.threshold and .shape()
- output : Original image, image after processing and image shape

def find_edge_point :

- input : Image after processing
- process : By using the function np.where(), the coordinate of the edge point can be noted in a list.
- output : List of the edge points' coordinate

def find_circle :

- input : List of the edge points' coordinate, maximum radius, minimum radius, shape of image
- process : Calculate the distance of each point on the edge from other points within its specified range
- output : Vote map of the image

def plot_image :

- input : Center of circle and radius of circle
- process : cv2.circle (Mark traffic sign)
- output : Target image

def process :

- input : Image name
- process : Image process, find edge point, find circle, find the maximum value in the space of circle center and radius, plot image
- output : Target image