Stop Sign Recognition Using Hough Transform

Wenxi Chen
Faculty of Science
University of Western Ontario
London, Canada
wchen466@uwo.ca

Yuchen Chen
Faculty of Science
University of Western Ontario
London, Canada
mchen.hba2023@ivey.ca

Wenhan Sun
Faculty of Science
University of Western Ontario
London, Canada
wsun228@uwo.ca

Zhengzheng Yu
Faculty of Engineering
University of Western Ontario
London, Canada
zyu322@uwo.ca

***Abstract*—With the rise of self-driving technologies, the problem of road sign detection and recognition is becoming more and more important in the development of a new system. This paper describes a program for stop sign detection using Hough transform to process the image and recognize the stop sign. We have selected a wide range of images containing the stop sign to test the robustness of our system. Noise were artificially added to raise the difficulty and to simulate a real-world scenario. Our system is capable of recognizing all stop signs in our test images.**

I. Introduction

Traffic sign detection and recognition have been experimented with increasing research interest in the last times. This is due to the importance of improving safety in road vehicles. Drivers sometimes miss signs because of distractions or lack of concentration. Driver error is the most common cause of car accidents in Canada. Driver errors that result in accidents include (but are not limited to): following too close; driving while distracted; failing to come to a complete stop at stop signs; etc [1]. In this project, we are going to attempt to recognize stop signs by obtaining the distance-transform from an edge image and doing a further matching with pre-selected templates, corresponding to those signals searched. We are going to use Hough transform to achieve the goal above.

By implementing the Hough transform, we will have it recognize octagon shapes (stop signs) and we aim to assist the driver's situational awareness to minimize potential accidents from happening. In the future, there is potential to modify and expand this method allowing it to recognize other road signs other than stop signs, such as yield, speed limit, and school zone signs.

II. Background and Related Work

The Hough transform is a technique that can be used to isolate features of a particular shape within an image. Because it requires that the desired features be specified in some parametric form, the classical Hough transform is most commonly used for the detection of regular curves such as lines, circles, ellipses, etc. The Hough technique is particularly useful for computing a global description of a feature(s) (where the number of solution classes need not be known a priori), given (possibly noisy) local measurements. The motivating idea behind the Hough technique for line detection is that each input measurement (e.g. coordinate point) indicates its contribution to a globally consistent solution (e.g. the physical line which gave rise to that image point) [2].

III. Methods

A. Research Objectives

The end goal of this paper is to analyze whether it is feasible to use Hough transformations for shape and stop sign recognition. Moreover, there are two more specific research objectives that this paper concerns:

1) Whether or not the Hough transform could accurately and effectively identify octagonal shapes such as a stop sign.
2) To what extent would the background noise and scenery of an image detract from the recognition of the stop sign.

B. Research Methodology

The overall research methodology consists of 4 steps: data gathering, creating edge maps, applying the Hough transformation and an algorithm to identify octagons from the transformed results.

1) Data Gathering:

The data used consists of stop sign images taken in real life with a phone camera, as well as photos found online with Google Images. Because the end goal is to identify stop signs, the images used were taken in natural settings such as roads, parks, etc., instead of standalone pictures of stop signs by themselves with a white background.

Realistically, every image would have noise and other disturbing elements if it was taken in real-life scenarios. These noises and disturbing elements are embraced in this paper to determine the extent to which they affect the results of the Hough Transformations as stated in the research objectives, as shown in Fig. 1.

Fig. 1: Sample image used for processing

2) Creating Edge Maps:

In order to perform the Hough transformation effectively, edge maps were created as a preprocessing step to recognize the edges in the image. This is a significant step, because the Hough transformation will need to specify the distinct edges, as the stop signs have eight straight edges. The details of the Hough transformations will be explained in later sections. Consider the zoomed part of the original image (Fig. 2):

Fig. 2: Zoomed stop sign of the original image

There are two ways to create the edge map, which are the Canny and the Sobel edge detection methods:

(a) The Sobel Edge Detection. This method uses two 3×3 matrices to estimate the gradient in the x and the y direction. The Sobel method calculates the gradient of the intensity of the image at each pixel, and gives the direction to either increase or decrease the image's intensity from lighter or darker. Edges are represented by strong intensity contrasts [4].

(b) The Canny Edge Detection. This method has a series of steps [5]. First, smooth the image with Gaussian filters. Then compute the gradient magnitude and orientation using finite-difference approximations for the partial derivatives. After that, apply non-maxima suppression to the gradient magnitude. Finally, use the double threshold algorithm to detect and link edges. The equation used for Gaussian smoothing is as follows:

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{1}$$

Where σ is the spread of the gaussian that controls the degree of the smoothing.

Both the Canny and Sobel edge detections methods were used in order to determine the most suitable edge map for feeding into the Hough transformations. Fig. 3(a) shows the results of applying edge detection using the Canny method, while Fig. 3(b) shows the results of applying edge detection using the Sobel method.

(a) Sobel edge detection 0.12 threshold

(b) Canny edge detection 0.12 threshold, $\sigma = \sqrt{2}$

Fig. 3: Sample image used for processing

The differences between Canny and Sobel lies mainly in the background, which is a tree. The Canny detection detects the tree very precisely. The Sobel however, is less sensitive to the structure of the tree than Canny, as it only partially and roughly outlines the structure of the tree. Overall, it seems that the Canny is detecting more edges. Therefore, the Hough transformation might be affected because there are more edges. Ultimately, the Sobel detection was chosen because of its simpler outputs and less edges that the Hough Transformation needed to go through.

3) Octagon Pattern in the Hough Space:

Consider a regular octagon with vertices $P_1(x_1, y_1)$ to $P_8(x_8, y_8)$, the side length is a and the central angle is β. Assume the octagon is centered at the origin of the coordinate system, as shown in Fig. 4.

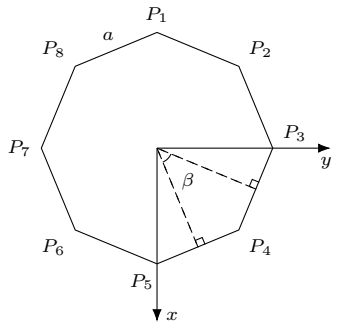

Fig. 4: A regular octagon centered at the origin of the coordinate system

To transform the octagon into the Hough space, each point $P_i(x_i, y_i)$ on the sides of the octagon are to be converted into the Hesse normal form $\rho = x_i \cos(\theta) + y_i \sin(\theta)$ [6], as shown in Fig. 5. Therefore, each of the points of the line segments is represented by the normal angle θ and the distance ρ from the origin.

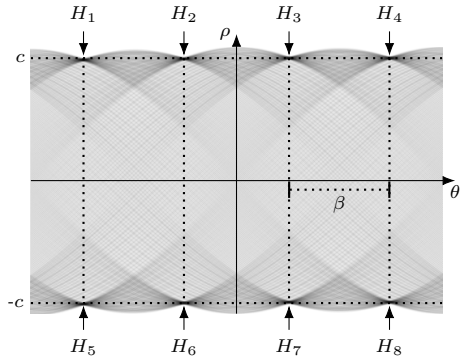

Fig. 5: The Hough transformation of the original octagon

Since each single dots in the Cartesian space are transformed into curves in the Hough space, each of the intersection peak indicates a side in the original image. Since the octagon is composed of 8 sides, there are 8 intersection peaks presented on the Hough space in the range $[-\frac{\pi}{2}, \frac{\pi}{2}]$. These intersection peaks are labelled from $H_1(\theta_1, \rho_1)$ to $H_8(\theta_8, \rho_8)$, which represent the sides $P_6P_7, P_5P_6, P_4P_5, P_3P_4, P_2P_3, P_1P_2, P_8P_1$, and P_7P_8 of the original octagon respectively.

As demonstrated in the Hough space, these attributes of the pattern are observable [3]:

(a) The intersection peaks are vertically in pairs (e.g. H_1 and H_5) with a distance of $2c$. These pairs represent the parallel sides of the original octagon (e.g. H_1 represents P_6P_7, H_5 represents P_2P_3, $P_6P_7 \parallel P_2P_3$).
(b) The sum of ρ of each pair is 0 (i.e. the distance from the origin to the two sides are both c). Since the side length of the octagon is a, $c = \frac{1+\sqrt{2}}{2}a$.
(c) The intervals of each vertical pairs are the same, which is $\beta = \frac{\pi}{4}$ (i.e. the central angle of the regular octagon).

However, in real testing cases, the stop signs are not completely regular octagons, which results in inevitable errors (e.g. different side lengths and central angles). To handle the errors is a challange for the identification algorithm.

4) Octagon Identification Algorithm:

For the input of the algorithm, the image would be processed using Sobel edge detection as discussed previously. Once the edge map is obtained, perform the Hough transformation to get an accumulator for all $\Delta\theta$ in the Hough space.

The θ of the target edge pairs is indefinite, since the rotation of the octagon may vary. However, it is definite that the edge pairs are distributed in 4 separated chunk slices of the Hough space, each with range β.

Hence, the sum of the normalized square of each column of the accumulator is first to be computed. With the sum, sum up the sum of the last 3 chunk slices respectively onto the first chunk slice. Then, $\hat{\theta}$, which is the $\Delta\theta$ of the local maximum in the first chunk slice, would be the most possible rotation angle of the octagon.

Now, for each chunk slice, specify the range of the portion of the possible intersection peaks, which is $[\hat{\theta}-\epsilon, \hat{\theta}+\epsilon]$, where ϵ indicates the bias of the tolerance. Then, find all 4 pairs of the intersection peaks in the range, and this is the naïve algorithm to identify the octagon from the edge map (Algorithm 1).

Algorithm 1 Naïve identification algorithm

```
read img
edgeMap ← Sobel(img)
accumulator ← Hough(edgeMap)
sum ← [0] for Δθ in the Hough space
for Δθ in the Hough space do:
    sum[Δθ] ← normalize(sum + accumulator(Δθ)^2)
end for
for Δθ in the first quater of the Hough space do:
    for k in range(4) do:
        sum[Δθ] ← sum[Δθ] + sum[Δθ + k · β]
    end for
end for
θ̂ = indexOf(max(sum))
peak ← [(0,0),(0,0)] for k in range(4)
for k in range(4)
    L = k · β + θ̂ − ε
    R = k · β + θ̂ + ε
    peak[k] ← HoughPeaks(range(L,R), accumulator, 2)
end for
```

However, this naïve algorithm cannot handle noises effectively. Such noises might be formed from various factors (e.g. the effection of "STOP", background, pole, etc.). Therefore, an adaption to handle the noise is to be further executed to regularize one pair of intersection peaks (Algorithm 2).

Algorithm 2 Adaptation for handling the noises

```
distance ← [0] for k in range(4)
for k in range(4) do:
    distance[k] ← |peak[k][0] - peak[k][1]|
end for
anomaly ← indexOf(max(|distance−μ(distance)|))
mean ← μ(distance / valueAt(anomaly))
regL ← anomaly·β + θ̂ − ε
regR ← anomaly·β + θ̂ + ε
peak' ← HoughPeaks(range(regL,regR), accumulator, 5)
diffMin ← mean
for i in range(4) do:
    for j in range(i+1, 5) do:
        diff ← ||peak'[i]−peak'[j]|−mean|
        if diff < diffMin do:
            diffMin ← diff
            peak[anomaly][0] ← peak'[i]
            peak[anomaly][1] ← peak'[j]
        end if
    end for
end for
```

As described, the index of the anomaly value is first to be found. Using the mean of the anomaly excluded, find 5 intersection peak candidates in the anomaly range. Then, iterate through the candidates to find the 2 peaks with the absolute difference the closest to the excluded mean, and replace the anomaly eventually.

IV. Results

A. The Original Test Cases

The data were passed into the algorithm. Fig. 6 shows the original image, Fig. 7 shows its edge map, Fig. 8 shows its Hough transform output, and lastly, Fig. 9 shows its detected edges.

Fig. 6: Original test image

Fig. 7: Edge map (Sobel)

From the edge map, it is evident that there is noise in the edges such as the pole of the stop sign, and the word "STOP" itself. This was expected, because realistically, to the human eye, the boundaries of the edges of the pole of the stop sign and the stop sign itself should be very distinct. Whereas, it would be a problem when the edge map does not contain the stop sign's edges, because it may be the case that the stop sign is very hard to see to the human eye as well.

From the Hough transformation results, it seems that the algorithm did well in recognizing the nature of the octagon. One could see that in the Hough transformation, the strongest edges are the strongest intersection peaks, which show up as very bright portion of curves. There are red box markers showing the edges that were picked by the algorithm. The algorithm is supposed to pick the intersection peaks that are strong enough while satisfying 4 pairs of parallel lines/edges to fit the shape of the stop sign; this seemed to work given the result. It is obvious that the red markers correspond to the 4

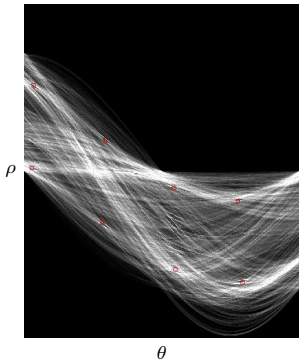

Fig. 8: Hough transform result

pairs of parallel lines. For each pair indeed, it shows that the value of θ are the same, which is expected.

Fig. 9: The detection of the stop sign

The final result shows the detected edge in yellow. While the result is not a perfect octagon as it has spaces in the edges, this would still be a correct recognition. This result was somewhat expected because the misrecognized edges were part of the stop sign itself. One could see that the stop sign has a white border, which represents two octagons. In this case, the noise is coming from the stop sign itself, which is acceptable.

Further tests were done with the same image, with manually added noise as shown in Fig. 10(a) and Fig. 10(b), and their edge maps are shown in Fig. 11(a) and Fig 11(b). The results are shown in Figure 12(a) and Figure 12(b). Where *image 2* is used for testing the algrithm's performance against angular interference, while *image 3* is used for testing if the algorithm can pick the set of edges that has similar distance with each other.

As illustrated in *image 2* and *image 3*, it is evident that the manually added noise was a significant edge on the edge map. The algorithm's performance on the light noise addition case is similar to the base case.

Since the sum of each columns of the Hough space for all three images (i.e. the original image, *image 2*, and *image 3*) shown in Fig. 13, observable anomaly impulses are presented. For *image 2*, the impulse at θ around 21° shows the location of the noise. While for *image 3*, the impulse at θ around 9° is more protruding than the other two images, as the θ of the edge and the noise are the same.

(a) *image 2* (b) *image 3*

Fig. 10: Original test image (noise added)

(a) *image 2* (b) *image 3*

Fig. 11: Edge map (Sobel, noise added)

(a) *image 2* (b) *image 3*

Fig. 12: The detection of the stop sign (noise added)

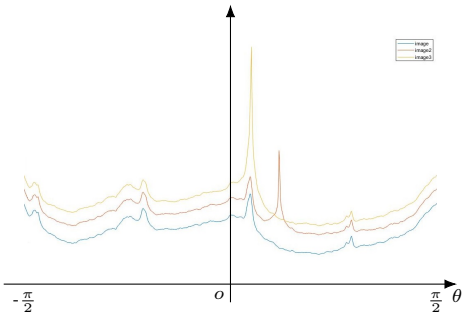

Fig. 13: The sum of squares comparison of the three images

B. Other Test Cases

There are vaious other test cases used to test the algorithm. Except for the stop sign, other signs with octagon shapes are to be considered as well, see Fig. 14 to Fig. 17.

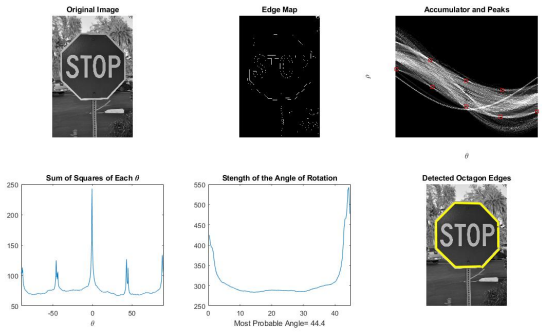

Fig. 14: Another stop sign

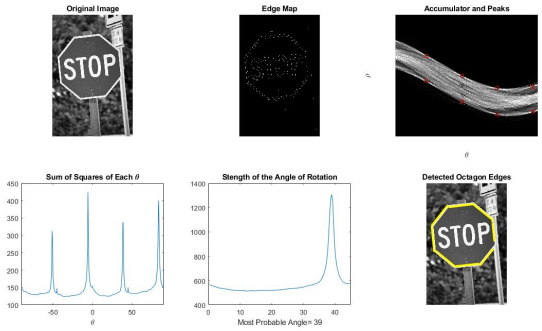

Fig. 15: Another stop sign

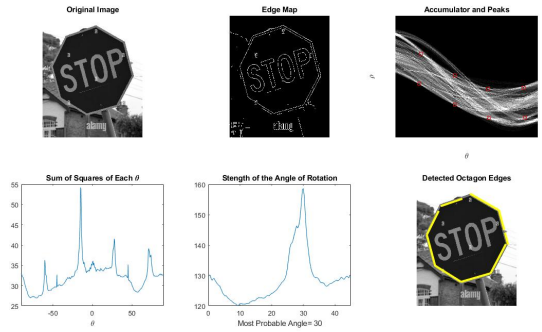

Fig. 16: Another stop sign

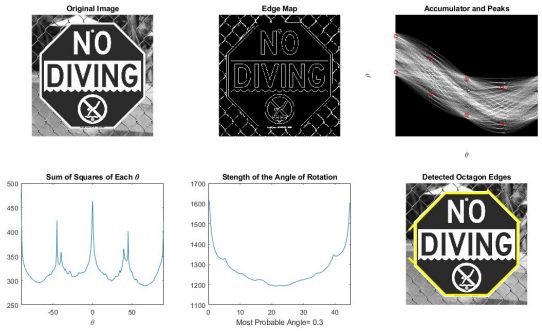

Fig. 17: ”No Diving” sign

V. Conclusions and Future Work

From the results, it should be concluded that the first research objective was met as the Hough Transformation could in fact be used to recognize the octagonal shapes. The second research objective was partially met, as there were still deficiencies in recognizing the stop sign shape with a significant amount of noise.

It was learned that the Hough Transformation result could robustly identify shapes that have some known properties. All it needs is an algorithm that somewhat describes the properties of the shape to search for in the Hough results while setting some threshold such as edge significance.

It is evident that this detection is not perfect and would not fit the safety standards when used in real life. One future step is setting a clear decision requirement to identify whether the identified result should be considered as a stop sign. This would require a lot more data from real life as well as safety considerations and would be a hard decision to make. An example of decision criteria would be whether the shape must be a complete octagon, or how far should the edges of the octagon be apart from each other for it to be considered as a stop sign.

References

[1] M. Lawyers, "The most common causes of car accidents in Canada," The Most Common Causes of Car Accidents in Canada, 08-Sep-2021. [Online]. Available: https://www.mannlawyers.com/resources/the-most-common-causes-of-car-accidents-in-canada/.

[2] R. Fisher, S. Perkins, A. Walker, and E. Wolfart, "Hough Transform," Image Transforms - Hough Transform, 2003. [Online]. Available: https://homepages.inf.ed.ac.uk/rbf/HIPR2/hough.htm.

[3] C. R. Jung and R. Schramm, "Rectangle detection based on a windowed Hough transform," Proceedings. 17th Brazilian Symposium on Computer Graphics and Image Processing, 2004, pp. 113-120, doi: 10.1109/SIBGRA.2004.1352951.

[4] Rafael C. Gonzalez and Richard E. Woods, Digital Image Processing. 3nd. ed. New Jersey: Pearson Prentice Hall. 2008.

[5] Mohamed Ali and David Clausi (2001), Using The Canny Edge Detector for Feature Extraction and Enhancement of Remote Sensing Images, IEEE, pp 2298-2300.

[6] R. Duda and P. Hart. Use of the hough transform to detect lines and curves in pictures. Communications of the ACM, 15(1):11–15, January 1972.