Traffic Sign Recognition for Intelligent Speed Assist

Conor Power
BSc in Applied Computing
Waterford Institute of Technology
conorgpower@gmail.com

***Abstract*— This document outlines the design and modelling of the Traffic Sign Recognition (TSR) ADAS feature for the purpose of implementing the Intelligent Speed Assist (ISA) ADAS feature within an automated vehicle. Vision based speed assistance systems are common amongst automated vehicles due to their ability to reduce both the number of road accidents and the severity of road accidents.**

Keywords*—*detection, speed, circles, Matlab, OCR

I. Introduction

Intelligent speed assist systems using traffic sign recognition are an important element to an autonomous vehicle to regulate the speed of the vehicle depending on the speed limit of the road the vehicle is currently on. This project attempts to use image processing and optical character recognition (ORC) to accurately read speed signs so that the vehicles speed may be adjusted accordingly. The main goal will be to accurately read a series of signs on a road and output their values.

II. Image Preprocessing

A. Video Input

For the input to the system an MP4 video was used for development purposes. This was found on YouTube [1], then edited down to the relevant section and finally transformed into frames in image (.jpg) format. Initial testing was done on still images, however, I felt that the ability for the system to perform in live video situations is crucial to the functionality of the system. The three instances of signs in the video are at frames 11, 27 and 44. An example frame from the video is displayed below.

Figure 1 - SIngle Frame from Video Input.

B. Cropping the Region of Interest

The OCR program for the system, which will read the signs values, will only be executed on regions where a circle is detected. This is to reduce the workload of the intensive computational algorithms used to perform these functions. This is done by using the 'imcrop' MATLAB function to select a region of interest in a frame when a circle is detected. The function accepts the original image, seen is Figure 2, and a matrix array with the location of the area being selected and outputs the cropped image, shown in Figure 3 below.

Figure 2 - Original Frame

Figure 3 - Cropped Frame

C. Image Smoothing

In order to achieve the maximum accuracy during the optical character recognition stage, the cropped image must be filtered in order to reduce the noise that negatively affects the OCR performance. Initially, the 'rgb2gray' filter is applied to the cropped image. This removes the noise caused by all variants of rgb values in an image by changing the colour map to grayscale. Next, the 'imsharpen' filter is applied which

sharpens the contrast in the image using the unsharp masking feature. Following that, the 'imbinarize' filter is used. This filter uses a threshold value to classify each pixel as either a 0 or a 1, turning the image into black and white pixels only. After experimentation, I found that setting the foreground polarity to dark and sensitivity to 0.55 was most effective for the maximal removal of noise while maintaining the bulk of the numeric values in the image. Finally, 'bwmorph' was applied twice, once to clean and once to fill. This removes the isolated pixels causing noise and then fills in the gaps between pixels that are beside one another.

Figure 3 - Original Crop of Frame 44

Figure 4 - Post rgb2gray

Figure 5 - Post sharpen

Figure 6 - Post binarization

Figure 7 - Post bwmorph

III. Circle detection

Each frame of the video must be examined in an attempt to identify potential circles for further processing. During this process, the 'imfindcircles' function is used. This function finds circles using circular Hough transforms. The process entails an accumulation array computation process, which involves foreground pixels of high gradient voting in the accumulation array to form full circles, followed by center estimation and radius estimation [2].

During my experimentation with the function, I found the radius range of 50-110 pixels and a sensitivity of 0.9 were best for identifying speed signs without identifying other circular patterns. The object polarity parameter was not as straightforward, with some signs being brighter than their backgrounds and others being darker, depending on the lighting in the frame. For this reason the third sign in the video is best detected using bright object polarity, where as the first two are best detected using dark. A smarter solution for determining object polarity may be employed in future iterations.

Once 'imfindcircles has successfully identified a potential sign, the 'viscircles' method can be used to draw a circle around it. This is done by passing the centers and radii parameters, the result of 'imfindcircles' into the 'viscircles' method. In this case a bright red line is drawn around the circles found.

Figure 8 - No Circles Drawn at 0.8 Sensitivity

Figure 9 - Correct Circle Drawn at 0.9 Sensitivity

Figure 10 - Incorrect Circles Drawn at 0.95 Sensitivity

IV. Optical Character Recognition

Optical character recognition (OCR) is used in MATLAB to recognize text. The characters on the signs proved to be the most difficult element of the system to correctly predict, far more difficult than circle detection.. For the purposes of this project only number characters needed to be read, therefore, the character set parameter was filled with the following string '0123456789' to increase the accuracy of the OCR function. Once the OCR function ran on the image the text found was shown to the left of the cropped image.

For the three signs detected in the video, after the preprocessing had taken place, none could be correctly interpreted by the OCR. After experimentation with all OCR parameters and adjusting preprocessing parameters, I found that further cropping the signs allowed it to be interpreted correctly. Further cropping the signs to reduce the noise surrounding the images had to be done manually, however, a more sophisticated algorithm to isolate the digits may be possible in future versions.

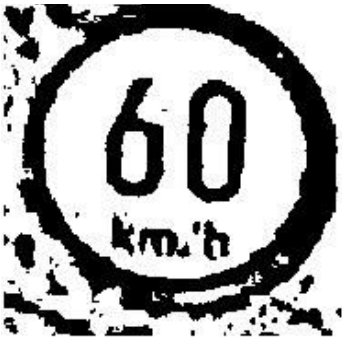

Figure 11 - Frame 27 Undetected

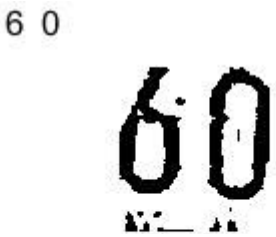

Figure 12 - Frame 27 Detected After Further Cropping

The main reason for the inaccuracy of the OCR engine before the additional cropping was due to image quality and the limitations of the OCR's performance in the presence of noise. "Accuracy of OCR systems is subject to or directly dependent on the quality of the input document" [3]. The prediction quality of the OCR was compromised, even after the image preprocessing stage, due to the low quality of the video input and the additional noise. To examine this hypothesis, a sign from a high quality image was also tested.

Figure 13 - High Quality Image with Preprocessing Not Detected

Figure 14 - High Quality Image with Preprocessing and Additional Cropping Detected Successfully

Through testing high quality images it is apparent that noise rather than image quality is the most limiting factor of the MATLAB OCR. For road worthy implementations of traffic sign recognition for intelligent speed assist, either a more accurate OCR must be used or a more complex method of isolating the numbers must be employed.

V. Benchmark testing

The following benchmarks were set out before implementing the system using the video input.

A. Detect Circles

This benchmark consists of successfully detecting any circular objects in a frame. The system passed this benchmark upon implementing the 'imfindcircles' method. Sensitivity of the function had to be raised to detect circles. Circles detected also had a thick red line drawn around them to highlight them in an image.

B. Detect Only Signs

This benchmark builds upon the previous benchmark of detecting circles. As mentioned in the previous benchmark, sensitivity had to be raised to identify circular patterns in the frames, however, to detect signs without detecting other circular patterns required some experimentation. This benchmark was achieved through a trial and error approach, eventually settling on a sensitivity of 0.9 for the 'imfindcircles' method.

C. Isolate a Circle

Following successful detection of signs only, this benchmark involves isolating the coordinates of the sign and using them to display that particular region in a new figure. This benchmark was successfully achieved by using the output of the 'imfindcircles method, which is the center coordinates and radii of the circles found, as inputs for the 'imcrop' coordinate array.

D. Identify Characters

This benchmark consists of successfully identifying characters in the cropped image using the OCR function. This benchmark was the most challenging to achieve but was achieved by attempting the OCR function under different image preprocessing conditions, although the characters recognized were not initially the correct characters.

E. Identify Correct Characters

This benchmark involved further work using the OCR method. In order to turn the readings from incorrect numbers to the correct ones, almost all noise had to be removed by cropping the image a second time. By cropping the image a second time only the numbers remained and were successfully read.

F. Display Characters

This benchmark involved representing the text detected by the OCR function and displaying it in a text box for the system user. This benchmark was achieved by assigning the text value of the OCR's output to a variable and displaying it to the left of the image using the 'text' function.

VI. Conclusions

Having completed this project, the traffic sign recognition for intelligent speed assistance has completed its main goal of accurately reading a series of signs on a road and displaying their values and has passed all the benchmarks set out. This goal however, has only been achieved in a narrow frame of view and should the system be applied to other roads and in more difficult weather and environmental conditions, the accuracy of its performance may be in doubt.

The existing system could improve it performance by making parameters of object polarity in 'imfindcircles' and sensitivity in 'imbinarize' more intelligent, along with the parameters for other image processing methods.

If I were to conduct a project with the same goal in the future, I would consider a machine learning approach. Given the relative consistency of road signs compared to more complex image classification problems, a comprehensive data set of Irish road signs trained using an effective image classification algorithm, such as Convolutional Neural Networks (CNN), may outperform this system.

VII. References

[1] BDofIRE (2019). 'Driving Test Ireland | The Full Route the Tester Took Me On | Raheny, Co.Dublin'. [online] Accessed on: 31/12/2019 Available at:https://www.youtube.com/watch?v=VrukTo4pV-k&t=186s

[2] H.K Yuen, .J. Princen, J. Illingworth, and J. Kittler. "Comparative study of Hough transform methods for circle finding." Image and Vision Computing. Volume 8, Number 1, 1990, pp. 71–77

[3] Singh A.K., Gupta A., Saxena A. (2016). "Optical Character Recognition: A Review". Accessed on: 02/01/2020. Available at: https://pdfs.semanticscholar.org/0fd3/a2b4c2cfc2c812243a8661993a21f17b3411