Joe Paul

AUE-8240

Final Project Report

5/2/22

AUE 822 Final Project Report
Autonomous Lane Keeping and Sign Recognition

Introduction

The task provided in this project is to control a scale vehicle fully autonomously though a taped course. Within this course, two signs are also present: a stop sign and a school zone sign. These signs indicate whether the vehicle should slow down or stop. To navigate this course, the vehicle is outfitted with two cameras. These cameras interact with two separate PC's which communicate with each other and send commands to an Arduino to control the vehicle steering and speed. To address this task our group constructed a control architecture that uses the input from the cameras, processes it and then uses it to control the vehicle. In this report I will first outline the general control architecture, discuss the details of the submodules, discuss the effectiveness of our approach, and address any changes we would have made.

Theory

Given the provided sensing array, the approach utilized to control the vehicle requires visual sensing of the lines on the course and the signs alongside. To detect the lines on the course the camera first captures an image. This image, captured in the pixel frame, must first be preprocessed to identify the coordinates of the lines. To preprocess the image, a couple of different filtering steps were applied to allow for the maximum distinction between the lines and the rest of the environment. This captured image is then run through an edge detection algorithm to determine the edges of the lines in the pixel frame. Once these edges are established, they are then passed through a Hough transformation process to convert them into line segments. The parameters of the Hough transformation module can be varied to increase the sensitivity to the lines generated from the edge transform.

After separating the image into a series of line segments, it must be determined which of the two taped lines these segments belong to. There are several methods of doing so, each with tradeoffs between processing speed, reliability, and complexity. After this transformation is complete a center line must be determined between the two lines visible in the pixel frame. Because this line is in the pixel frame of the image, it must then be transformed into the camera frame. Because only one camera is used in this process, to determine the distance away from the camera of an object, the process must be informed by some knowledge of the camera relative to the pixel frame. In our case, we utilized information on the relative height of the camera to the ground and assumed that the ground was flat. This allows us to know the Y coordinate in the camera frame of any objects at ground level and allows us to determine the distance of objects from the camera. Coupled with the intrinsic parameters of the camera, this allows us to transform the lines seen in the pixel frame into the camera frame. We assume that the camera frame is

sufficiently close to the center of the front axle that a vertical line perpendicular to the front axle and centered is representative of the desired path of the vehicle relative to the tape lines.

With this assumption, and the lines now transformed into the camera frame, we can determine the vehicle's departure angle and off-center error relative to the course. This information can then be passed to the vehicle control module to center the vehicle.

A secondary program is necessary to evaluate the presence of signs along the vehicle's path. This program utilizes a convolutional neural network to determine whether a sign is present in the view of the secondary camera. We leveraged an existing image recognition neural network called cifar10Net, and utilized transfer learning, the process of teaching an existing neural network to recognize new items similar to those already within its data set. The convolution neural network passes the image through a set of convolution filters where the neural network can pick out characteristics of the image which help indicate what type of image it is. We were able to utilize transfer learning because the images in the already trained data set have similar characteristics to the new images that we needed to recognize. To train the CNN, a data set of images of the two sign types was collected. These datasets then had to be labeled and boxed to show what we wanted the CNN to be able to recognize. After the training process, the CNN can be utilized to examine images taken by the secondary camera in determining the presence of a sign. the presence of a sign can then be sent to the vehicle control module to tell the vehicle at which speed it should travel.

General Control Architecture

Before beginning the construction of the individual control programs our group first met to determine how the control programs would interact with each other. We initially determined that three separate sub control modules would be necessary. The first module is the vehicle control module. This module receives feedback from the other two modules in determines how to process that information into steering and speed controls that are then sent to the Arduino to control the vehicle. We decided that the inputs to this control module would be the departure angle and centering error determined from the lane recognition module. Additionally, we determined that the sign recognition module would output either a value of zero, one, or two depending on the presence of a sign in the secondary camera's frame. Finally, we determined when in the process the primary script would query the second computer to determine whether a sign was present.

This outlining allowed our group to delegate work clearly between the three areas and understand what each area was expecting to receive from the others. We made very few changes to this general architecture, with the primary one being the control of the vehicle speed within the sign recognition module. This was necessary to disallow the robot from stopping repeatedly after it initially stopped for a sign within its frame of view. It also allowed the vehicle to travel at half speed until it saw a stop sign (approximately halfway around the course). We believe that doing this processing within the sign recognition module allowed the system to be more reliable.

Subsystem Architecture

Lane Recognition

Our group attempted two separate ways of determining whether the lines shown in the Hough transformation belonged to the left or right tapeline relative to the vehicle. The first approach, and the approach we used in our final test, determined whether these lines belong to the left or right side based on their position in the camera frame and their angles. This separation assumed that lines appearing in the left half of the image belonged to the left line and lines in the right half of the image belonged to the right lane. Additionally, we attempted a secondary approach to better capture the distinction between these two lines especially when only one of the two lines was clearly visible in the camera frame. This secondary approach relied on the fact that when the vehicle was only able to see one line this was typically the outer line of the course. With that knowledge and based on the angle of the line in the camera frame this approach was able to designate whether the single line viewed was the left or right line. The outputs of the two separate strategies are shown in figure 1.

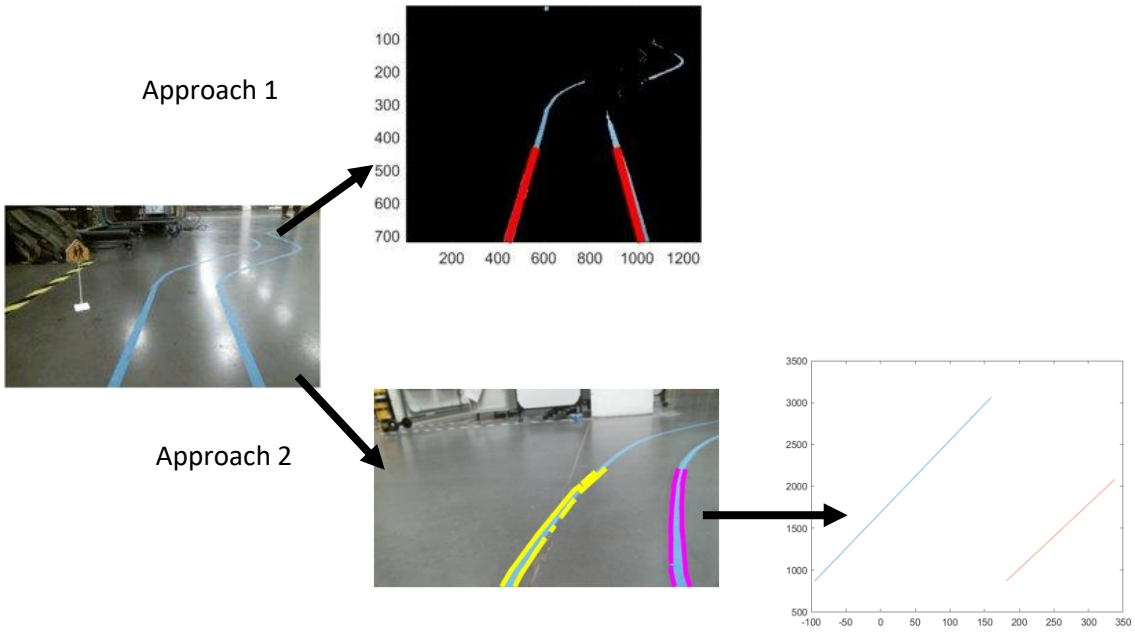

Figure 1. An illustration of the difference between the two approaches.

One clear distinction between these two methods is when in the process the polyline function is used to approximate the lines. In approach one the lines are approximated in the pixel frame, where in approach two the lines are only approximated in the camera frame. One problem with both these approaches is that they assumed that the camera is mounted flat relative to the ground, when in reality the camera is mounted at a 21-degree decline. The consideration of this decline did not appear to be critical in our testing, so this fact was ignored. This issue is clearly visible in the plot of the lines in the camera frame of approach 2, where the lines appear to be going further apart from each other the further they get from the camera, when they should be parallel.

After tuning and testing we were able to perform acceptably with approach one and therefore did not continue tuning approach two. Approach one was able to recognize the lines sufficiently and with basic one-line conditions was able to slow the vehicle when only one line was visible. Approach two is likely to be more robust however due to the time constraints of the project we were not able to analyze it further.

Both approaches utilize both gaussian and color filtering to accurately separate the lines from their environments. These filtering techniques were tuned from sample datasets taken from the vehicle in two separate lighting conditions, each composed of about 100 pictures.

Sign Recognition

The sign recognition portion of the task was relatively straightforward. After training the CNN we were able to adjust the sensitivity of the detector to reduce false positives. As mentioned in the overall system architecture section, we also decided to incorporate the stopping and slow down logic for the vehicle speed control into the sign recognition module on the second computer. This module sent the results for the speed of the vehicle to the second computer via a UDP connection. Additionally, we tuned this functionality by varying the camera angle such that the vehicle would not recognize the sign too far away. Additionally, we tuned the percent certainty acceptance value to lessen false positives.

Our control architecture only needed to see the sign in a single captured image to initialize the necessary speed change. This caused the vehicle to stop early at some points, including in our final test.

Vehicle Control

The vehicle control module utilizes the outputs of the lane recognition function and a Stanley controller to calculate the steering angle necessary to stay on path. To tune the system, we varied two parameters: the sensitivity to departure angle and the sensitivity to centering error. To increase the vehicle's ability to make sharper turns we increased the departure angle coefficient such that the vehicle would provide more input relative to its angular error. To increase the vehicle centering performance we increase the angular sensitivity to centering error.

The vehicle speed control utilizes a pulse width modulation type control to pulse the vehicle on and off at the minimum input speed. We tuned both the on and off time of this pulse to control our vehicle speed. When the vehicle was in a slow zone, we simply lowered the on time of the forward pulse. Because the processing time of the lanes varied, especially when only one lane or no lanes was visible, sometimes the vehicle would experience spurts in speed longer than expected. This method of speed control was sufficient to complete the task as desired.

The steering control module utilizes a Stanley controller to process the input departure angle and lane centering error into a steering output. To increase the systems responsiveness we tuned two parameters, k_1 for the departure angle sensitivity, and k_2 for the lane centering error sensitivity. We increased the departure angle sensitivity to increase responsiveness in bends and increased the centering error parameter to increase centering speed. This control worked relatively well although final inspection of the code revealed an issue that caused lane centering error to constantly report as 0 rather than the desired value. Correction of this error would likely

increase the vehicle's ability to center more quickly and alleviate issues we saw with the vehicle tending to center over a line rather than in the middle of the lane.

Results

During testing our vehicle was able to track the lines sufficiently and control its speed when necessary. Figure 2 illustrates the centering error experienced by the vehicle in a lap of the track. During testing, we identified an error in our steering control script. Our steering control script originally utilized speed input from the vehicle based on its current speed state, however rather than passing in the speed as an output value we had changed to only accept speed as the state value of zero one or two. This error is also mentioned in the vehicle control subsystem overview. This caused the centering error term to not be correct most of the time. This did not affect our overall performance, however, should be addressed if this system was to be improved.

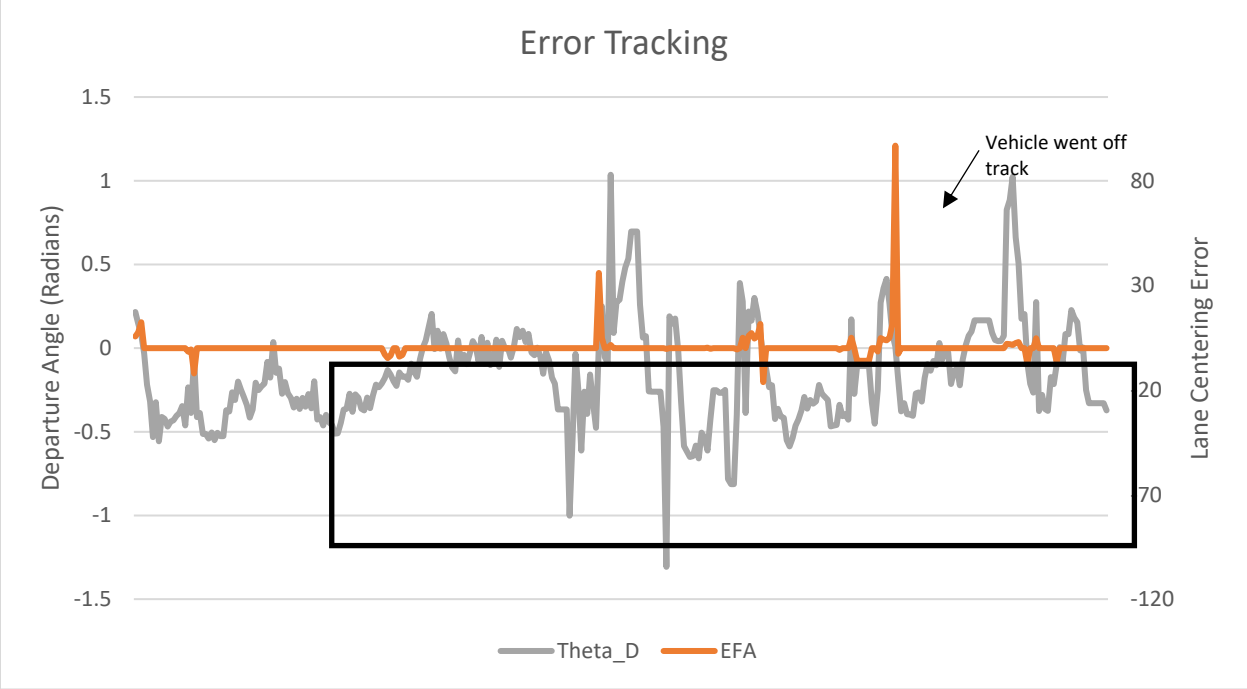

Figure 2. Error tracking data from a control lap.

The figure clearly demonstrates when the vehicle goes off track, and the black box illustrates that since the course is mostly right turns (clockwise track following) the vehicles angular error is almost always negative, indicating that it needs to turn right to stay within the bounds.

Our vehicle was also able to traverse the track quite quickly due to our relatively simple control logic that made the program run quite quickly and update the steering values very often. This did however cause the vehicle to lose control if it experienced a hiccup and processing delay. We found MATLAB to be relatively unstable and it would often crash on our computers without warning or clear reason.

The sign recognition module ran relatively quickly on a PC equipped with a graphics care and yielded the following results for a simulated run. We encoded 1 as a stop, 2 as a slowdown, and 0 as normal speed. It is clearly visible in this figure that the program outputs a stop and pauses for an extended period before processing the next command. Additionally, the slow down command is fixed on until a stop is detected as visible.

Figure 3. Resultant sign module output.

Changes

To improve our system, we would make a couple of changes to improve its lane tracking abilities and overall stability. One of these changes would be to utilize the second approach for lane tracking. This approach tends to yield a much more accurate picture especially of the centering error and could increase the system's ability to deal with compromised lane visibility immensely. Additionally, we would correct the vehicle steering control such that it incorporates the centering error term correctly. Finally, we would like to increase the complexity of our speed control logic such that it is better able to identify whether it has already seen a sign. Some possible ways to do this would be by predicting the change in location of the sign in an image from one frame to the next so that the vehicle knows it is seeing the same sign.

Conclusion

Overall, the implementation of the control systems learned in the class revealed the difficulty associated with translating class knowledge to hardware. Most of our time was spent debugging and tuning the system, with comparatively less time spent adjusting the portions of the project generated directly from the principles of the class.

This project also revealed the importance of segmenting portions of an overall control program such that it is easy to debug small portions at a time in a controlled environment. It would be very interesting to attempt a similar project on a full-scale vehicle that may be less

susceptible to some of the low-level control problems we experienced due to the relative simplicity of the vehicles and hardware used. This project was also great at improving my understanding of the basic functionality and implementation of certain key automotive control principles that I feel are often grouped into libraries and assumed to work properly. I greatly enjoyed this project and would recommend its continued use for students is future classes.