AUTOMATED DISCONNECTED TOWING SYSTEM

by

Yaqin Wang

A Thesis

Submitted to the Faculty of Purdue University

In Partial Fulfillment of the Requirements for the Degree of

Master of Science

Department of Computer and Information Technology

West Lafayette, Indiana

May 2020

THE PURDUE UNIVERSITY GRADUATE SCHOOL
STATEMENT OF COMMITTEE APPROVAL

Dr. Eric T. Matson, Chair

Department of Computer and Information Technology

Prof. Anthony H. Smith

Department of Computer and Information Technology

Dr. John A Springer

Department of Computer and Information Technology

Approved by:

Dr. Eric T. Matson

Head of the Graduate Program

Dedicated to my grandmother.

ACKNOWLEDGMENTS

I wish to gratefully acknowledge my thesis committee for their insightful comments and guidance, and my family for their support and encouragement. I also wish to acknowledge LHP Engineering Solutions for providing guidance and research resources for my thesis.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

ADTS	automated disconnected towing system
ROI	region of interest
ROS	robotic operating system
CTE	cross trek error

ABSTRACT

Author: Wang, Yaqin. M.S.
Institution: Purdue University
Degree Received: May 2020
Title: Automated Disconnected Towing System
Major Professor: Eric T. Matson

Towing capacity affects a vehicle's towing ability and it is usually costly to buy or even rent a vehicle that can tow certain amount of weight. A widely swaying towing trailer is one of the main causes for accidents that involves towing trailers. This study propose an affordable automated disconnected towing system (ADTS) that does not require physical connection between leading vehicle and the trailer vehicle by only using a computer vision system. The ADTS contains two main parts: a leading vehicle which can perform lane detection and a trailer vehicle which can automatically follow the leading vehicle by detecting the license plate of the leading vehicle. The trailer vehicle can adjust its speed according to the distance from the leading vehicle.

CHAPTER 1. INTRODUCTION

1.1 Background

Towing capacity determines the maximum weight that a vehicle can pull while towing any kind of cargo, such as a trailer, another vehicle or a boat. Even though we often refer towing for transportation, towing capacity also involves in water-based transportation (Job, 1978). People need to own or rent pickup trucks or large SUVs when towing is needed. Table 1.1 shows the towing capacity of the popular vehicle models (Advisor, 2019 (accessed November 7, 2019)) in the market, as well as the manufacturer suggested retail price (MSRP).

From Table 1.1, we can see that it is very costly to buy or even rent a vehicle that can tow a certain amount of weight. According to Santander Consumer USA, the most popular vehicle model in USA in 2019 is Toyota Camry (Macesich, 2019 (accessed November 7, 2019)), which only has about 1000 pounds towing capacity. The other problem with towing is the high risk of vehicle accidents happening every year. According to NHTSA, there are over 50000 accidents every year that are related to towing, and over 400 people have died in those accidents (Koenigsberg, 2019 (accessed November 7, 2019)). Most of those accidents are caused by the widely swaying towing trailer. If a pickup truck turns or brakes too suddenly, the side forces will cause the trailer swaying from side to side behind the tow vehicle, which is also called ”fishtailing.” It is usually impossible to stop a swaying trailer and the tow vehicle will end up dragged by the trailer and cause a severe accident.

Table 1.1. Towing capacity for selected SUVs and pickup trucks in 2018.

Make	Model	Trim	Engine	Max Tow Capacity	MSRP
Jeep	Grand Cherokee	4WD	3.6L V-6	6200	$30895
Acura	MDX	AWD	3.5L V-6	3,500	$46200
Audi	Q7	Turbocharged	2.0L	4,400	$49900
BMW	X5	All	All	6,000	$57200
Ram	1500	Reg Std 4WD	3.6L V-6	7050	$46500
Ford	F-150	Reg Shortbed 4WD	3.3L V-6	7400	$37025
Chevrolet	Silverado	Reg Cab Bed 4WD	5.3L V-8	6500	$40230
Toyota	Tundra	Double Cab Bed 4WD	5.7L V-8	9900	$43635
Nissan	Titan	Reg Cab Bed 4WD	5.6L V-8	9560	$34000

1.2 Research Question

This study proposes an automated disconnected towing system (ADTS) that does not require a physical connection between a leading vehicle and a trailer vehicle by only using a computer vision system. Each of the leading and trailer vehicles has a depth camera mounted in the front. The cameras on the leading vehicle are able to perform lane detection. And the one on the trailer vehicle is able to perform license detection and to detect the distance from the leading vehicle and adjust the speed accordingly.

1.3 Significance

The contribution of this study are: Firstly, for those who own small vehicles, such as a Volkswagen beetle, can tow a few thousands of pounds of cargo. People will not need to worry about the towing capacity of their vehicles when towing. Secondly, ADTS can help reducing vehicle accidents that are related to towing. Also, it is possible to adopt ADTS in any type of small vehicle, which is a not large truck or a semi bus, or a non-commercial vehicle. In that way, any vehicle can turn into a trailer.

1.4 Assumptions

There are several assumptions for the automated disconnected towing system to work. Firstly, the ADTS requires both leading and trailer vehicles having own individual their power systems. In addition, ADTS requires no other vehicles or objects appearing in between the leading vehicle and the trailer vehicle. Also, the leading vehicle can not conduct sudden braking because the trailer vehicle needs a certain time to process the distance information and adjust the speed accordingly.

1.5 Limitations and Future Works

There are a few limitations to this project. Firstly, the prototype is only tested in the parking lot and the speed is hardcoded up to 10 MPH. Secondly, the computer vision system may not work in extreme weather, such as heavy rain or fog. Lastly, there is not object detection or pedestrian detection function added to the vision system. Also, this study mainly focuses on software-related system design. All of these can be considered as part of the future work.

1.6 Summary

Lane detection is one of the main tasks in an autonomous driving system. This project is to design an affordable automated disconnected towing system (ADTS) by using color filters in the computer vision system. ADTS contains two main parts: a leading vehicle which can perform lane detection and a trailer vehicle which can automatically follow the leading vehicle by detecting the license plate of the leading vehicle. The trailer vehicle can adjust its speed according to the distance from the leading vehicle. If the distance is too far away, the trailer vehicle will speed up; and if the distance is too close, it will slow down or come to a full stop.

The rest of the paper is structured as follows. In chapter II, a thorough discussion of related work is presented which builds the foundations of this research. Next, chapter III articulates the technical details of the system. In chapter IV, the results of experiments and tests of the proposed system are presented. Finally, chapter V concludes this study and addresses future work.

CHAPTER 2. REVIEW OF LITERATURE

2.1 Advanced Driver-Assistance System

ADAS (advanced driver-assistance system) assists human drivers when driving or parking. Current ADAS technology uses electronic systems, such as vehicle onboard computer system (Meier, Tanskanen, Fraundorfer, & Pollefeys, 2011), electronic control units(ECU), and microcontroller units (MCU). The purpose of ADAS is to improve driver's safety and comfort, and more generally, to improve road safety and traffic flow (Van Arem, Van Driel, & Visser, 2006).

According to ASIRT, there are about 1.25 million motor vehicle crash deaths each year, which is about 3287 per day (ASIRT, 2019). Among those, most of the vehicle accidents are caused by human error (Brookhuis, De Waard, & Janssen, 2019). ADAS is developed to support, assist, automate, and improve safety and more comfortable driving for human drivers. ADAS has been proven to help minimize human errors to decrease fatal road accidents (Hamid et al., 2017).

ADAS helps by reducing collisions and road accidents using its safety features. The technologies that ADAS provides can alert human drivers about potential risks and problems. Current ADAS technology can also detect objects and pedestrians, perform basic classification, and in some cases, take control of the vehicles when necessary. The conventional ADAS features in nowadays vehicles include automated lightning system, adaptive cruise control (ACC), lane keeping system, blind-spot monitoring, forward collision warning, surround-view cameras, lane departure warning, pedestrian detection system, road sign recognition, autonomous emergency braking, and parking assist (Biassoni, Ruscio, & Ciceri, 2016).

2.2 Levels of Automotive Autonomy

2.2.1 SAE Standard

In 2018, SAE International released a new standard, J3016 (Committee et al., n.d.), which is called "Levels of Driving Automation". The standard defines the six levels of driving automation, from no automation to full automation.

2.2.2 Levels of Automation

There is a total of six levels of automation defined by SAE J3016, from level 0 to level 5. For the first three levels, ADAS features will assist the human driver when driving or parking. Even when a human driver is not controlling the acceleration pedal or his or her hands are off the steering wheel by using adaptive cruise control, or lane keeping, the human driver is still in control of the vehicle, which means he or she has to supervise the supporting features. On the contrary, for the last three levels, the human driver is not in control of the vehicle when the autonomous features are engaged. Even when the human driver is seating in the driver's seat, he or she does not need to supervise the autonomous features. Human drivers may not even be required to seat in the driver's seat for the 5th level of automation.

Level 0 provides features that are limited to constantly support and even to warn the human driver. Automation features in level 0 include automatic emergency brake, blind-spot warning, and lane departure warning. Level 0 is considered as no automation at all levels.

Level 1 provides features that can assist steering or acceleration pedal (accelerate and brake) for human drivers. Features in Level 1 can be adaptive cruise control or lane centering. Level 1 provides more assistance to the drivers than level 0.

Level 2 provides both steering and acceleration assists to the human driver. Level 2 automation can perform both adaptive cruise control and lane centering. Level 2 is considered as partial automation.

Level 3 is considered as conditional automation. One of the features of level 3 is traffic jam assist. Traffic jam assist can control the vehicle by using lane markings, traffic signs, and other vehicles on the road as reference. Driver is still necessary but is not required to supervise the driving.

Level 4 is considered high automation, in which the vehicle is able to perform all functions in driving and parking under certain conditions. Level 4 automation can be applied to local driver-less taxi.

Level 5 automation is the highest level which is considered as full automation. A vehicle with level 5 automation is capable of performing all functions in the driving and parking under all conditions. Vehicles with level 5 automation are considered as self-driving cars.

2.3 Lane Detection

It seems that the issue of lane detection is not so difficult. Bear this in mind, the vehicle only needs to identify the host lane, and detect a short distance ahead of itself. A commonly used, simple hue transform-based algorithm can solve the problem in about 90% of the high way scenarios (Borkar, Hayes, Smith, & Pankanti, 2009). However, there is no easy answer for lane detection. It takes a lot of effort, resource, and time to build an efficient lane detection system, because of the obvious gaps in research, diversity scenarios, and high liability requirement (Hillel, Lerner, Levi, & Raz, 2014).

2.3.1 Issue in Current Research

Soon, the trend for the automotive industry will switch from an increasing amount of semi-automatic functions to full automation. Table 2.1 summarized some of the major automatic functions on vehicles in recent five years. It is obvious that lane departure warning (LDW) has been a popular research area in both academic and industrial research. The complicated scenarios on the road require LDW to be able to detect and identify the host lane and the nearby area in front of the human driver. Significant research effort was also devoted to full autonomy, mainly due to the DARPA challenges (Bacha et al., 2008; Borkar et al., 2009; Broggi & Cattani, 2006; Kammel & Pitzer, 2008; Kong, Audibert, & Ponce, 2009; Kornhauser et al., 2007; Levinson et al., 2011; Lipski et al., 2008; Montemerlo et al., 2008; Rasmussen & Korah, 2005; Urmson et al., 2008; Ying & Li, 2016). However, there are a limited number of people have sufficient understanding about the complicity of road and lane as shown in Table 2.1 and 2.2.

In fact, full automation is the most complicated task in the autonomous driving system because it needs to deal with all the subsystems and overall structure. Undoubtedly, people may believe that the functions listed in Table 1 have been fully explored by researchers on fully autonomous vehicles. However, this might not be true under certain circumstances, such as non-highway scenarios. Those features in Table 1 need to be combined with highly accurate map information and on-board the localization system to perform a relatively good road and lane perception.

According to the discussion above, studies on autonomous driving did not pay much attention to perception systems, which is expected to be a popular research field in the automotive industry in the near future(Hillel et al., 2014). The unsolved problems include how to deal with unexpected complexity of the road and lane conditions, and how to extend the range of the area in front of the vehicle.

Table 2.1. Current autonomous features I.

Lane Keeping Assist System	
Function Requirements	Actively assist the driver to remain in the marked lane 1. Focus on host lane only 2. Identify small range of the in front of the car 3. Moderate reliability
Lane Departure Warning System	
Function Requirements	Warn drivers of straying from the lane 1. Focus on host lane only 2. Identify small range of the area in front of the car 3. Low reliability
Lane Centering System	
Function Requirements	Keep a car centered in the lane 1. Focus on host lane only 2. Identify medium range of the area in front of the car 3. High reliability 4. Split lane identification
Lane Change Assist System	
Function Requirements	Assist steering around an imminent crash 1. Focus on multi lanes 2. Identify large area in the front and back of the car 3. Moderate reliability
Steering Assist System	
Function Requirements	Assist steering around an imminent crash 1. Focus on host lane only 2. Identify small range of the area in front of the car 3. Moderate reliability
Left Turn Assist System	
Function Requirements	Warn the driver of opposing traffic and brake automatically 1. All lanes 2. Identify medium range of the area in front of the car 3. Moderate reliability

2.3.2 Requirement of High Accuracy and Reliability

Driver assistance system should fulfill the requirement of an extremely low error rate in order to serve the great public. In such alarming systems like LDW, a false alarm rate must have a lower limit because its high frequency would disturb the drivers and bring about the public to against it. The exact acceptable rate of the false alarm is still under

Table 2.2. Current autonomous features II.

Adaptive Cruise Control System	
Function	Automatically adjusts the vehicle speed to maintain a safe distance from vehicles ahead
Requirements	1. Focus on host lane only 2. Identify medium range of the area in front 3.High reliability 4. Adjust speed and auto break
Full-Autonomous in Paved Road	
Function	Autonomous driving in city and highway
Requirements	1. All lanes 2. Identify large area including junctions, round-about, and road under construction
Full-Autonomous in Non-Paved Road	
Function	Autonomous driving in all road conditions, including unpaved road
Requirements	1. All lanes 2. Identify large area including junctions, round-about, and road under construction 3. Identify the road without lane lines”

discussion (Barickman, Smith, & Jones, 2007; Burzio et al., 2010). Some existing systems will have a few false alarms per hour, in which, one false alarm per hour equals with one error in 54000 frames (Batavia, 1999). For those features in closed-loop automatic driving, errors should be even lower(Hillel et al., 2014). This kind of lower error rate is very difficult to achieve in vision-oriented systems. For other kinds of complicated computer vision system, such as a web-based searching application or surveillance system, people tend to be more forgiven about the error rate in such systems (Hillel et al., 2014).

Complexity in Road and Lane Conditions As mentioned above, the complexity of road and lane conditions are the main challenge for a sufficient lane detection system. To be able to detect in different lane and road conditions, different algorithms and subsystems need to be developed and fully tested. The complexity of the road and lane conditions include:

First of all, most lane marks are white or yellow with a width of 0.1m, but there are also many other exceptions like circular reflectors, cat's-eyes, lane marks with special colors and with changing the width.

In addition, in the US, lanes are usually 3.05 to 3.66 meters wide((US), 2007), with the overall exception range of 12%. Also, the number of lanes may be different in different areas and different parts of the road. For the most part, the road is straight, and the curvature is usually under certain limits (Hillel et al., 2014). Usually in urban roads, the curve radius is larger than 80 meters for 50 KPH (von Reyher, Joos, & Winner, 2005). Again, there are also exceptions.

In most cases, the road is open and human drivers can see things clearly in front of a vehicle. However, we need to consider uncommon circumstances, such as other vehicles, shadow on the surface of the road or bad weather. Sometimes, the vehicles in the other lane may block our vision. Trees and buildings along the road may create shadows interfering with our vision while driving. Sometimes, when our vehicle comes out of a tunnel, there would be a sudden change of lightning causing the overexposure to the light of images.

No matter what kind of weather it is, the lane detection system is expected to function normally. At least, the system should be able to identify the change of road and lightning conditions and adjust corresponding variables accordingly. Those situations discussed above requires a more advanced design of algorithms and sub-systems. There are still a lot of challenges even the most asic autonomous feature. Current technology and algorithms still need improvements to achieve higher accuracy in various and challenging scenarios.

2.4 Computer Vision

Lane detection is one of the most important topics in current autonomous driving. Researchers in both academics and industries have achieved much success in recent years (Hillel et al., 2014). Among many techniques approaching the lane detection function, computer vision is the most popular method. There are a few reasons why computer

vision is the most frequent-used method when approaching lane detection. Firstly, when we human drive, we also mostly rely on vision input as a reference. And it is the same for a computer vision system to conduct the lane detection function. Secondly, comparing to other autonomous driving techniques, like LiDAR or radar system, cameras that used in the computer vision system is one of the cheapest sensor equipment. Besides, the image input from the camera is relatively reliable, and we don't need to process them much to implement dependent settings.

In lane detection technology in computer vision system, figure-based method is one of the two frequent-used methods (Ozgunalp & Dahnoun, 2014; Su, Zhang, Lu, Yang, & Kong, 2017; Sun, Tsai, & Chan, 2006; Ying, Li, Wen, & Tan, 2017). When applying the figure-based method, the computer vision system separates the lane lines from other markings by features in the image input, which could be shape, dimension, color, the texture of the road in the image. Generally speaking, in order to detect the lane line, the figure-based method is based on the algorithm that recognizes various unstable features in an image input of the road. In most cases, the figure-based method requires the road to be clean and the borderlines of the lane to be obvious with distinct colors from other objects in the image. If the requirements are not met, the computer vision system is easily interfered with by noise, which could be a shadow or different colors of the pavement on the road. The vision system is unlikely to have good performance when those requirements are not met.

The other method that is frequently used in lane detection is model-based method (Su et al., 2017; C.-K. Wang, Huang, & Shieh, 2009; J. Wang & An, 2010, ?). The most significant difference of the model-based method is that it solves the lane detection problem as it solves the parameters solution of a mathematical problem (Tan, Zhou, Zhu, Yao, & Li, 2014). Because this method relies on a mathematical model in which the model is built based on the shape of the road (Deng & Han, 2013; Kim, 2006; Ruyi, Reinhard, Tobi, & Shigang, 2011). The model-based method collects the parameters of the mathematical model through the feature points of lane marks in the image input. In this way, comparing to the figure-based method, the model-based method is less likely to be interfered with by the noise. Even when the lane lines are not so clear on the road, the

model-based method can detect the lanes better than the figure-based method. However the model-based method usually requires higher computational power (Chen & Wang, 2006; Jung & Kelber, 2005), than figure-based method. If we simplify the model to fit with smaller computers, the accuracy of the result will be lower.

Based on the discussion in the previous paragraphs, the figured-based method is too sensitive to the noise, while the model-based method requires higher computational power to conduct complicated algorithms. In this research, the goal is to explore an alternative way in lane detection technology by employing a computer vision system, which is more tolerant of the noise and requires less computing power in processing the image input.

2.5 Color Spaces

2.5.1 Definition of Noise

When employing a computer vision system in lane detection, interfering noise is one of the biggest challenges (Srivastava, Singal, & Lumba, 2014). The noise usually refers to shadows caused by different lighting conditions. But when detecting the lane lines, the noise could be a lot of other things, such as skid marks caused by a sudden stop, pavement stains, heavy rains or snow, fog, and more. Figure 2.1 shows the noise created by the shadow and skid marks on the road that might interfere with the lane detection result. Image processing for lane detection is crucial in the autonomous/assistive driving system, and there are still problems remaining unsolved. Whenever there is interfering noise in the input image, the detection result will be compromised. To find a way to reduce the noise in image processing is a crucial step in lane detection using a computer vision system.

Figure 2.1. Example of noises in the road and lane detection.

2.5.2 Color Spaces

One of the common tools used in image processing in lane detection is color filters (Chiu & Lin, 2005; Crisman & Thorpe, 1993; He, Wang, & Zhang, 2004; Kong, Audibert, & Ponce, 2010; Srivastava et al., 2014; C.-K. Wang et al., 2009). Crisman and Thorpe developed a SCARF system which includes two-color cameras to process image segmentation by color. The different regions that are separated by color are classified. At the same time, they also use the hough-like transform to vote for various binary road models (Crisman & Thorpe, 1993). Turk developed the VITS system that also uses a two-color camera (Turk, Morgenthaler, Gremban, & Marra, 1988). In VITS, the color red and blue is used to decrease the interfering noise caused by shadows. Another two-model algorithm developed by He can detect the right and left edges of the borderline of the road and can recognize and enhance the borderlines of the road (He et al., 2004). Instead of a two-color camera, this two-model algorithm uses a full-color camera to process the input image. The other color-based method is approached by Chui and Lin, which can be applied in more challenging scenarios (Chiu & Lin, 2005). Their system can distinguish the borderlines of the lane by using color-based segmentation in image processing.

There have been many efforts made to reduce the interfering noise in image processing in lane detection. One of the common techniques used is color filters (Liu, Wang, & Chen, 2019). In this research, we have considered a few in the image processing, which are BGR, LAB, HSV, HSL, and LUV. By utilizing various color filters and setting different thresholds of each individual channel of the color filter, color filters could improve the accuracy of the computer vision system in the lane detection.

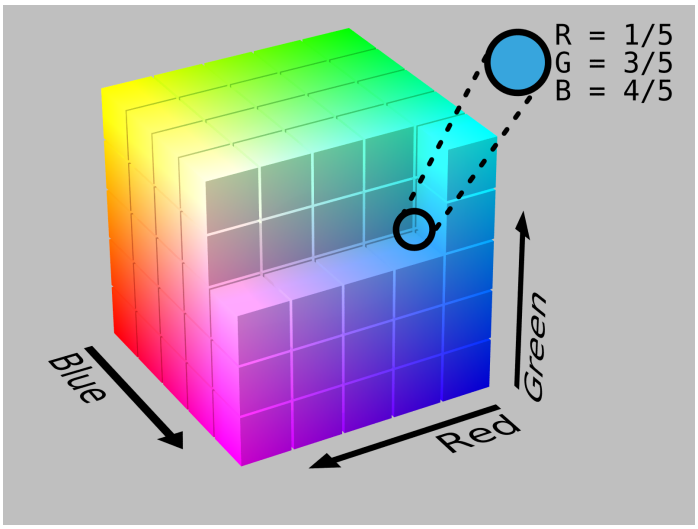

Figure 2.2. RGB color space.

The RGB color space uses the three primary colors, which are red, green, and blue, to present all the other colors in the images on the digital displays, for example, the digital screens for phones, televisions, and laptop computers. In order to create all possible colors in the visible spectrum, we can combine the three primary colors and they can be combined in different proportions (Liu et al., 2019). Each primary color in the RGB color space is also called the channel, ranges from 0 to 255. The numbers in the range of the channel represent the percentage of full intensity from 00000000 to 11111111 in binary or 00 to FF in hexadecimal (Xiao & Ma, 2006). The total possible combinations of color is about $256 \times 256 \times 256 = 16777216$. Figure 2.2 shows the color wheel of the RGB color space.

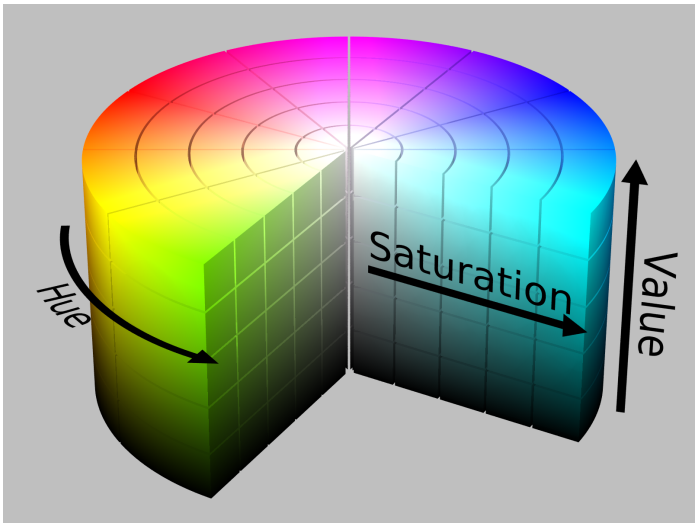

Figure 2.3. HSV color space.

The HSV color space stands for hue, saturation, and value, as shown in Figure 2.3. This color space can better explain the concept of light than RGB. Also, many complicated digital color pickers are based on HSV color space, such as Abobe software (Bear, 2019 (accessed November 2, 2019)). In HSV, hue represents the color portion of the space, ranging from 0 to 360 degrees. Within the scope, the color of red ranges from 0 to 60 degrees; yellow ranges from 61 to 120; green ranges from 121 to 180; cyan ranges from 181 to 240; blue ranges from 241 to 300; and magenta ranges from 301 to 360 degrees. The HSV color space is commonly used in computer graphics.

The HSV color wheel also contributes to high-quality graphics (Liu et al., 2019). Figure 2.3 shows the HSV color space.

Along with HSV, HSL are the other alternative color spaces for the RGB color space (Liu et al., 2019), as shown in Figure 2.4. The HSL represents hue, saturation, and lightness, and the HSV represents hue, saturation, and value. Both of HSL and HSV are intended to perform more closely as human eyes process colors. In HSL color space, the center axis from bottom to the top represents color black and white, respectively. In HSL, channel H represents color from red to green, and blue. The red is represented by percentage 0 or 360. The green is 120, and the blue is 240. The number between the three

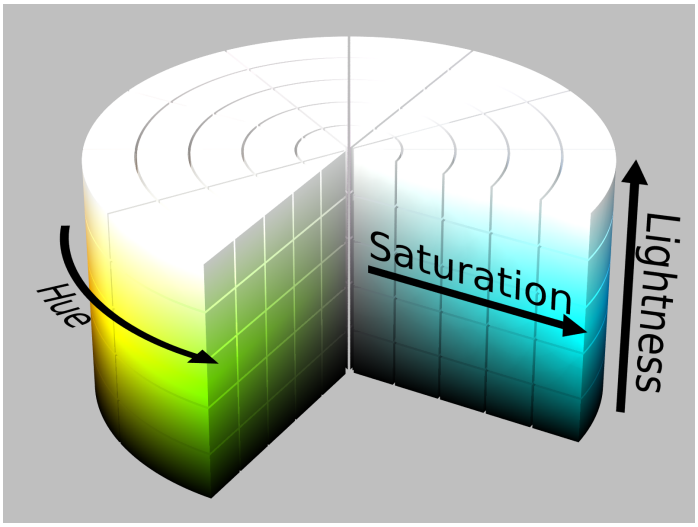

Figure 2.4. HSL color space.

colors are different shades. The channel S ranges from 0 to 100, which is a percentage value. And 100 in channel S represents a full color. Channel L is also a percentage number, which also ranges from 0 to 100. 0 percentage means dark (black), and 100 is light (white).

The other frequently used color space is LAB color space. Comparing to the three color spaces discussed previously, LAB is regarded as the most authentic method to represent the color. LAB also contains three channels, in which each channel represents color in a real number. Channel L represents lightness, and it ranges from 0 to 100, which indicates the color black and white, respectively. Channel A ranges from -128 to 0, and to 127, which indicates the color green, gray, and red, respectively. Channel B ranges from -128 to 0, and to 127, which indicates the color blue, gray, and yellow.

The other three-dimensional color space is LUV, and it is easier to compute than the other ones (Liu et al., 2019). Channel L represents illuminance and brightness. The channel U changes from the color green to red, while the parameter number of U increases. Channel V changes between the color blue and purple. When both channels U and V have the value equal to 0, the L channel represents a gradient change of the gray-scale.

2.6 Summary

In this chapter, a detailed discussion of related work is presented which builds the foundations of this project, including the current features of ADAS technologies, levels of autonomy by SAE standard, issues in current lane detection technology, and some commonly used color spaces in computer vision systems.

CHAPTER 3. METHODOLOGY

The goal of this study is to design an automated disconnected towing system (ADTS) that does not require a hard connection between the leading vehicle and trailer vehicle. ADTS contains two subsystems: a lane following system for the leading vehicle and license plate following system for the trailer vehicle. The lane following system allows the leading vehicle to follow the lane line and drive by itself. And the automated disconnected towing system allows the trailer to detect the license plate that is attached to the back of the leading vehicle. The other important function of ADTS is to adjust the trailer car's current speed according to the distance from the leading vehicle. In the rest of this chapter, the requirements of the hardware and software will be discussed in the rest of this chapter.

3.1 Line Following System

The goal of the line following system is to design an image processing system that enables the leading vehicle to drive itself autonomously by following a lane line. Overall, this line following system takes video images as input and produces cross track error (CTE) as output. The CTE is used by Roboteq, which is a type of DC motor controller, to calculate the steering angle command. The overall structure is demonstrated in Figure 3.1.

Figure 3.1. Line following system overview.

In order for the leading vehicle to follow the yellow lane line, it has to be able to recognize it first. A standard yellow tape is used to do indoor testing on the vision system and later, a pavement marking tape is used for outdoor testing. The camera will filter out the colors based on the RGB range, and it will only look for the color yellow. Then it will find the contour of the yellow area. If all the conditions are met and the lane line is found, the system will draw a blue cross in the middle of the yellow contour, as shown in Figure 3.2.

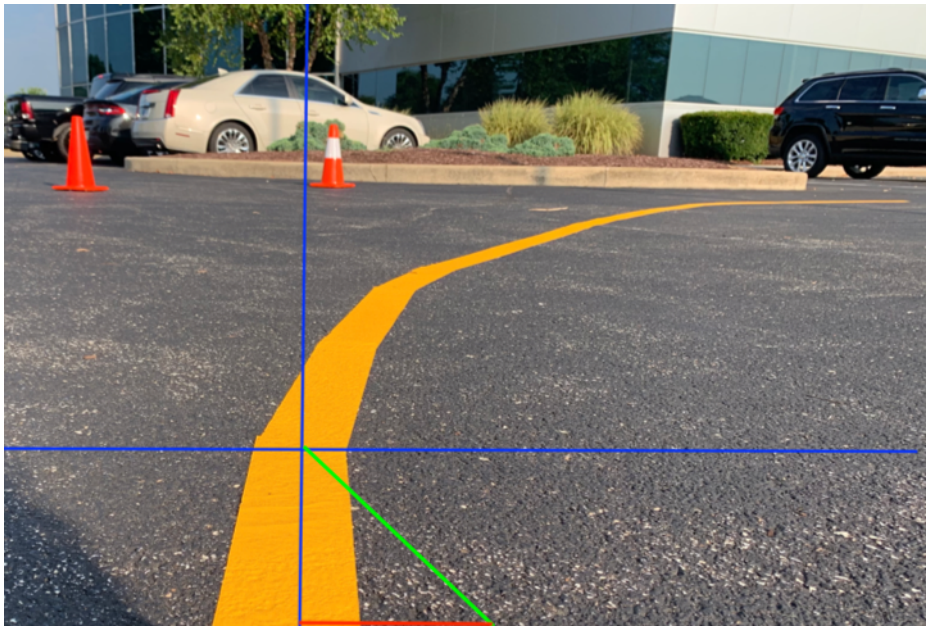

Figure 3.2. The line following system detects the lane and draw a blue cross in the middle of the yellow contour.

3.1.1 Hardware Requirement

The hardware requirements for lane following system include a camera and an NVIDIA Jetson TX2. The camera used in this project is a ZED stereo camera. However, for line following purpose, almost any webcam can perform the task.

3.1.2 Software Requirement

All the computer programs in this project are written in Python. The Robotic Operating System (ROS) is used as the platform in this project to run all the computer programs.

In order for the leading vehicle to follow the yellow lane line, it has to be able to recognize it first. To do so, firstly, the camera needs to find the region of interest in the input image. The region of interest (ROI) is set to be about lower one-third of the height of the original image, and about center one-third of the image. And then, the program turns the ROI into an array of points. An image filled with zero intensities with the same dimensions of the trimmed image is created as a mask. Then, the program fills the mask with values of 1 when the area overlaps with the frame (RIO) and fills with values of zero outside of RIO. A bitwise operation is conducted between the mask and the frame to only keep the triangular area of the frame.

After getting the frame of interest, the camera needs to detect the contour of the yellow lane line. OpenCV is used here to do image processing. The program filters out the colors based on the settings of the RGB range and all non-yellow colors are filtered out from the image. Then the program converts the frame to grayscale because the computer vision only needs the luminance channel for detecting edges. Also, changing the image to grayscale helps to save some computational power. A 5×5 Gaussian blur is applied to make the process easier and simpler. Then canny edge detector is applied with a minimum value of 50 and a maximum value of 150 to get the contour of the image. At the last, the program draws a blue cross in the middle of the contour, which will be the center of the lane line.

3.1.3 Cross Track Error in Line Following

Cross track error (CTE) is the key element to calculate the steering angle. The current CTE in Figure 3.3 is the angle between the red line and the orange line. To calculate the current CTE, the program will calculate the absolute value between the center of the image and the center of the detected lane line, as shown in the green line in Figure 3.3. The red line in the figure is the shortest absolute value between the center of the license plate to the bottom of the image. And then the arctangent function in the math library can calculate the degree of the angle between the red and the green line, which is the CTE.

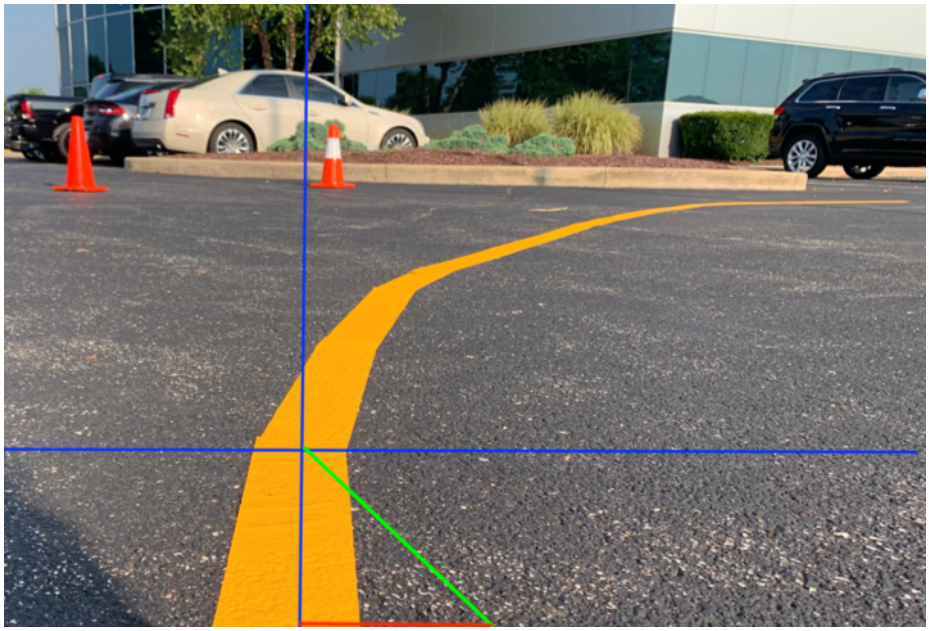

Figure 3.3. CTE in line following system.

3.2 License Plate Following System

The goal of the automated disconnected towing system (ADTS) is to design a computer vision system that can detect the license plate of the leading vehicle and follow it by keeping a certain distance. There are two functions that the license plate following system has to fulfill: the first one is the plate detection function, which is to be able to detect the license plate by using similar technology that is used in the line following system; the second one is the speed control function, which is to control the speed according to the distance information from the stereo camera. The overview structure of the license plate following system is showing in Figure 3.4.

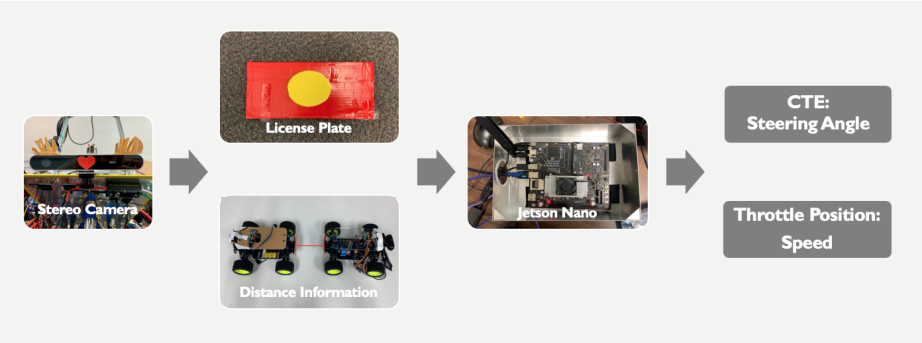

Figure 3.4. License plate following system overview.

The license plate attached to the leading vehicle is made with two colors, red and yellow. The plate detection function allows the camera to detect the license plate by its colors and shapes. The input of the plate detection function is the image input, and the output is cross track error (CTE), which is used in calculating the steering angle. At the same time, the speed control takes the distance information from the stereo camera, and the program will adjust the speed of the trailer vehicle according to the distance. The input of the speed control system is the distance, and the output is the speed.

3.2.1 Hardware Requirement

The hardware requirements for the license plate following system include a stereo camera and an NVIDIA Jetson Nano. The camera used in this project is a ZED stereo camera. However, in order to measure the distance between the leading vehicle and the trailer vehicle, a stereo camera is needed. Jetson Nano is used for ATDS because of its costs lower. One thing to keep in mind is that the main object of this project is to build an affordable, efficient automated disconnected towing system. However, if the budget allows, an NVIDIA Jetson TX2 would be better for its higher computational power.

3.2.2 Software Requirement

The computer programs for the license plate following system are also written in Python, and the platform to run the programs is the same as the line following sytem, which is ROS. In order for the trailer vehicle to follow the leading vehicle, it has to be able to recognize the license plate attached to the leading vehicle first, which is the license detection function. To do so, firstly, the camera needs to find the region of interest in the input image. The region of interest (ROI) is set to be about lower one-third of the height of the original image, and about center of the one-third of the image. And then, the program turns the ROI into an array of points. An image filled with zero intensities with the same dimensions of the trimmed image is created as a mask. Then, the program fills the mask with values of 1 when the area overlaps with the frame (RIO) and fills with values of zero outside of RIO. A bitwise operation is conducted between the mas and the frame to only keep the triangular area of the frame.

After getting the frame of interest, the camera needs to detect the contour of the license plate, which includes shapes: a big rectangle and a relatively smaller circular. Again, OpenCV is used here to do image processing. The program filters out the colors based on the settings of the RGB range and all non-yellow and non-red colors are filtered out from the image. After the program finds the two combinations of colors, it will try to find a rectangle shape and a circular shape. The program converts the frame to grayscale

by using it because we only need the luminance channel for detecting the edges of the shapes. Also, changing the image to grayscale helps to save some computational power. A 5×5 Gaussian blur is applied to make the process easier and simpler. Then Canny edge detector is applied with a minimum value of 50 and a maximum value of 150 to get the contour of the target shapes. After both shapes are located, the program draws a blue cross in the middle of the circular shape, which will be the center of the license plate.

For the speed control system in the license plate following system, the distance (depth) information is needed from the stereo camera. Figure 3.5 shows the depth view of captured objects in the image. From the figure, it is obvious to see the edges of objects. Because on those edges, the camera is not sure about the distance from the object, and it is showing as the default color. In general, the closer the object is, the darker it is showing in the depth view.

Figure 3.5. A example of a depth view from a stereo camera.

Table 3.1. The logic for speed control system in ADTS

if $min_{depth} <= 1.3$:
speed = 0
elif $1.3 < min_{depth} <= 1.4$:
speed = speed - 2
elif $1.4 < min_{depth} <= 1.6$:
speed = speed
elif $1.6 < mmin_{depth} <= 2.2$:
speed = speed + 2
elif $min_{depth} > 2.2$:
speed = speed + 4

In the speed control system, the logic is showing in Table 3.1. If the minimum distance between the leading vehicle and the trailer vehicle is less than 1.3 meters, the trailer vehicle comes to a full stop to avoid collision. If the minimum distance is between 1.4 and 1.6 meters, the trailer vehicle remains in a constant speed which is set to be 25 meters per hour. If the minimum distance is between 1.6 and 2.2 meters, the trailer vehicle increases its speed by 2 miles per hour to catch up with the leading vehicle. If the minimum distance is greater than 2.2 meters, the trailer vehicle increases its speed by 4 meters per hour to catch up with the leading vehicle.

3.2.3 Cross Track Error in License Plate Following System

Cross track error is the key element to calculate the steering angle. The current CTE in Figure 3.3 is the angle between the red line and the orange line. To calculate the current CTE, the program will calculate the absolute value between the center of the image and the center of the detected lane line, as shown in the red line in the Figure 3.3. The yellow line in the figure is the shortest absolute value between the center of the license plate to the bottom of the image. And then the arctangent function in the math library can calculate the degree of the angle between the red and the orange lines, which is the CTE.

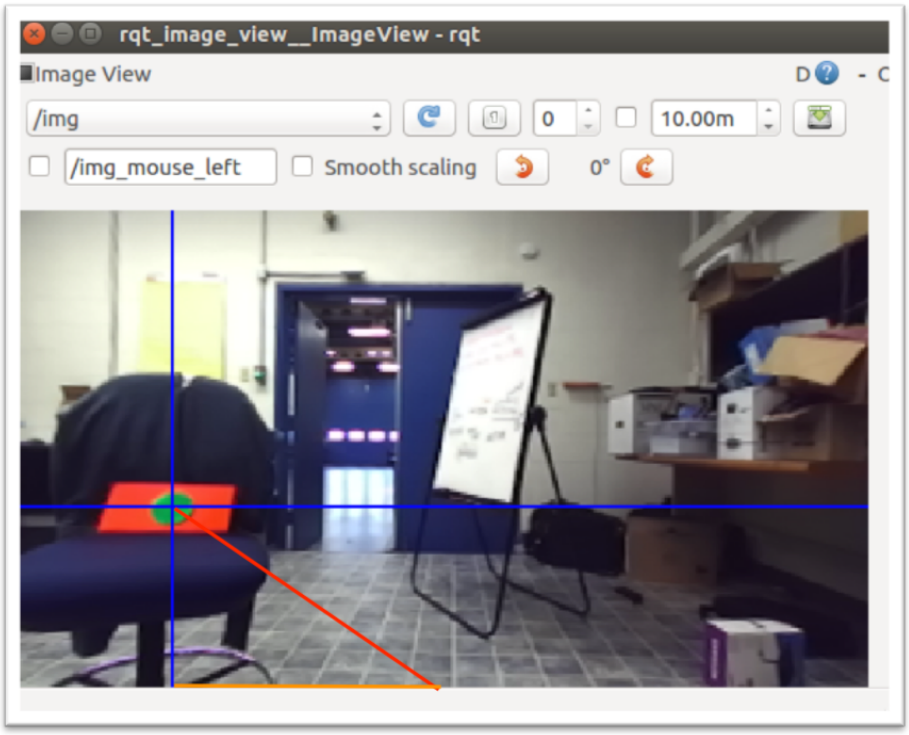

Figure 3.6. CTE in license plate following system.

3.3 Experiment Setup for Vision System

The vision system for ADST is divided into two parts: indoor testing and outdoor testing, for different lightning conditions. The camera used in lane following is Logitech Webcam C270, and the one used for license plate detection is a depth camera, ZED 1.

For the first stage of testing the vision system, the webcam and ZED camera is mounted to the robot cars. On one of the robot car, we have attached a real licence plate in the back, which functions as the leading vehicle.

3.4 Experiment Setup for ADTS

Experiments to test the system will be conducted in two parts: indoor and outdoor environments. The indoor experiment takes place in the company garage area with a concrete floor with a smooth finish on the ground, as shown in Figure 3.7. The outdoor experiments take place in the parking lot outside of the company building with uneven asphalt surface, as shwon in Figure 3.8.

Figure 3.7. Indoor experiment environment setup.

Figure 3.8. Outdoor experiment environment setup.

3.4.1 Equipment setup

First, the experiment will use two robot cars to experiment, as shown in Figure 3.9. Each of the robot car will use NVIDIA Jetson Nano as microprocessor and a webcam for the vision system.

After testing the system on the robot cars, the next step is to test the system on two go-karts, and Figure 3.10 is showing one of the go-karts. The NVIDIA Jetson TX2 will be used on the leading vehicle, for its availability, and the Jetson Nano will be used in the trailer vehicle for the low-cost purpose. The experiments on the course tracks will only use the go-karts.

3.4.2 Course tracks design and setup

There are three different course tracks to test the system, in different shapes, different lengths, and different floor surfaces. Course track I consists of 10-meter straight lines in both indoor and outdoor environments, as shown in Figure 3.11.

Course track II is a combination of straight lines and curves, about 20-meter long and 8-meter wide, as shown in Figure 3.12.

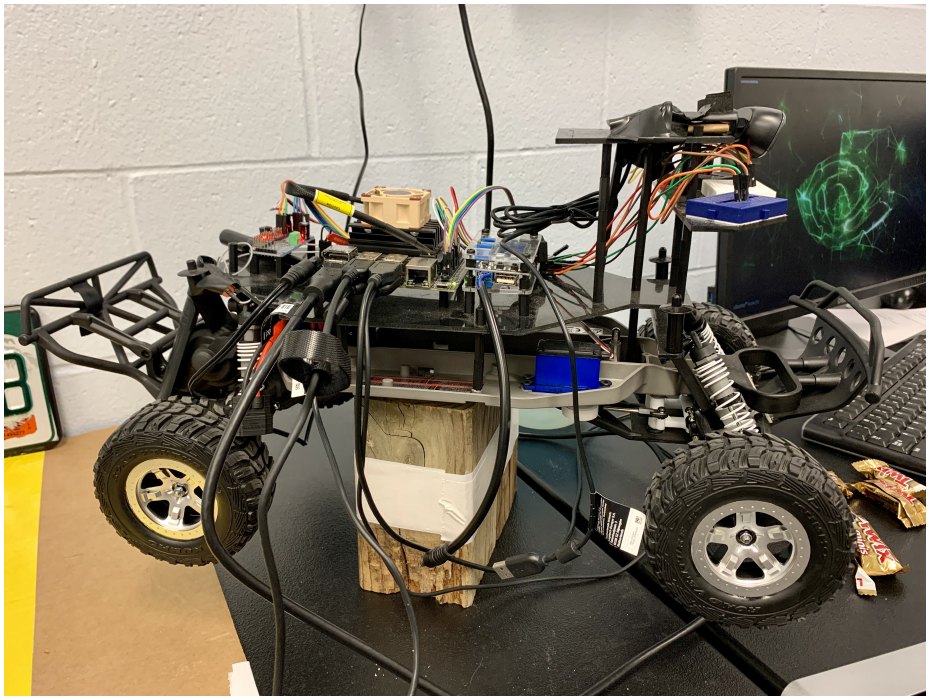

Figure 3.9. RC car to be used in the experiment.

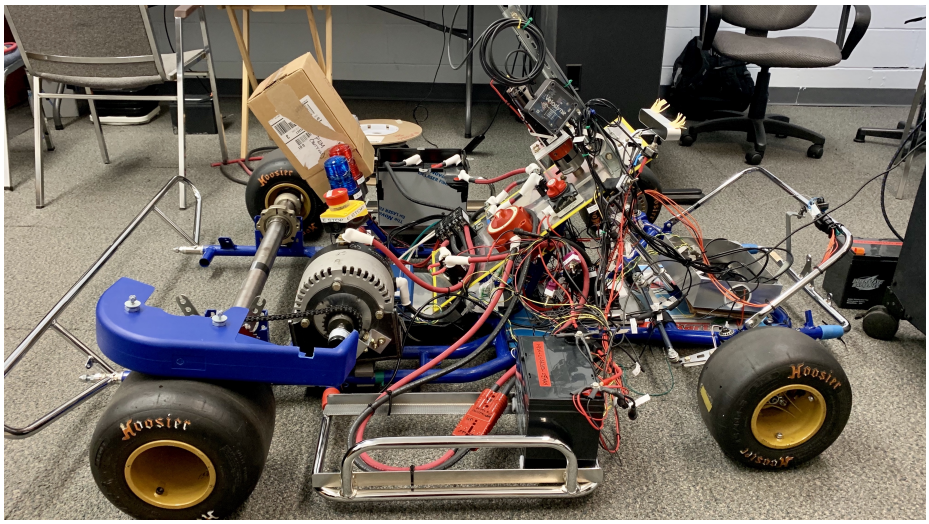

Figure 3.10. Go-kart to be used in the experiment.

Course track III is an Figure-8 course with about 10-meter straight lines and 2.5-meter radius curves.

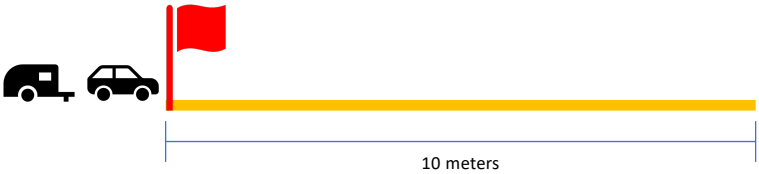

Figure 3.11. Course track I.

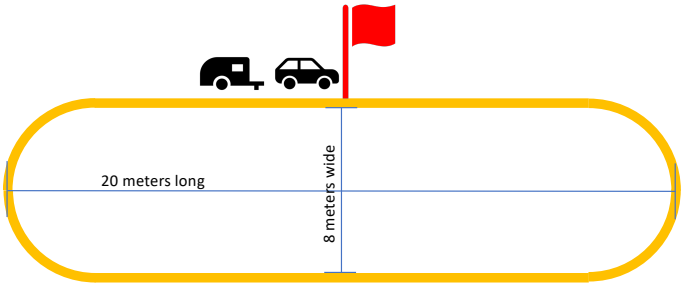

Figure 3.12. Course track II.

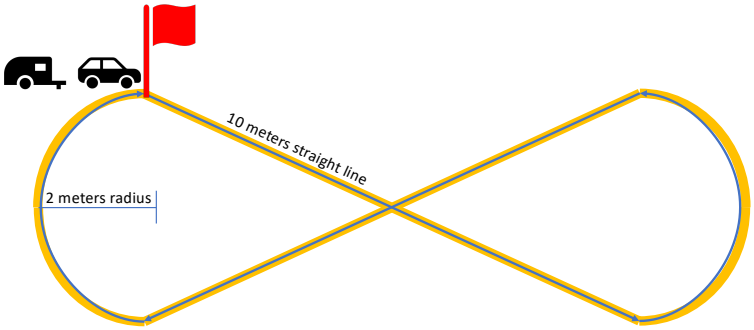

Figure 3.13. Course Track III.

Table 3.2. Experiment plan for testing ADST on course track 1.

Course Track 1	1st Run	2nd Run	3rd Run	4th Run	...
Test Environment					
Time of Experiment					
Running Time					
Error Type					
Error Location					
Estimate Running Distance					

3.4.3 Experiment plans

An example of a detailed experiment plan is showing in Table 3.2. ADTS will run 30 times on each course track. And for each run, the variables of each run include the current time of the experiment, the running period, possible errors, possible stopping point will be recorded.

3.5 Summary

In this chapter, the line following system and the license plate following system are discussed in detail for hardware and software requirements. Indoor and outdoor experiments and testing will be conducted to verify the performance and effectiveness. The results of the experiments will be discussed in the next chapter.

CHAPTER 4. EVALUATION AND RESULTS

4.1 Computer Vision System in ADST

ADST is an automated disconnected towing system that does not require a hard connection between the leading vehicle and the trailer vehicle. ADST contains two computer vision subsystems: a line following system for the leading vehicle and an automated towing system for the trailer vehicle. The line following system allows the leading vehicle to follow the lane line and drive by itself. And the automated disconnected towing system allows the trailer to detect the license plate that is attached to the back of the leading vehicle. The other important feature of ADTS is to adjust the trailer car's current speed according to the distance from the leading vehicle.

For line following and license plate following, each subsystem requires a stereo camera and an NVIDIA Jetson Nano. The camera used in this project is a ZED stereo camera. For line following and license plate detection, a regular webcam is able to perform with any problem. However, in order to measure the distance between the leading vehicle and the trailer vehicle, a stereo camera is needed. The Jetson Nano is used for ATDS because of its affordable price. The main propose of this project is to build an affordable, efficient automated disconnected towing system. However, if the budget allows, an NVIDIA Jetson TX2 would be better for its higher computational power.

The computer programs for the computer vision system are written in python, and the platform to run the programs is ROS. In order for the trailer vehicle to follow the leading vehicle, it has to be able to recognize the license plate that is attached to the leading vehicle first, which is the license plate detection function in ADTS. To do so, firstly, the camera needs to find the region of interest in the input image. The region of interest (ROI) is set to be about lower one-third of the height of the original image, and about the center of the one-third of the image. And then, the program turns the ROI into an array of points. An image filled with zero intensities with the same dimensions of the

trimmed image is created as a mask. Then, the program fills the mask with values of 1 when the area overlaps with the frame (RIO) and fills with values of zero outside of RIO. A bitwise operation is conducted between the mask and the frame to only keep the triangular area of the frame.

4.2 Line Following

The goal of the line following system is to design an image processing system that enables the leading vehicle to drive itself autonomously by following a lane line. Overall, this line following system takes video images as input and produces cross track error (CTE) as output. The CTE is used by Roboteq, which is a type of DC motor controller, to calculate the steering angle command. The overall structure is demonstrated in Figure 4.1.

Figure 4.1. Line following system overview.

The goal of the first stage of the experiment for line following is to simply follow a single line. After successfully performing the single line following, the system is modified and redesigned to perform double line detection, which is to detect two lines and keep the vehicle in the middle of the two lines. The line following system takes video frames as input, and process with HSV color space, which filters out unnecessary information for the line detection purpose. The HSV color space also helps to locate the yellow line in the image, by the giving specific color channel parameters. After using color space, Canny edge detection is used to find all the edges in the input frame. And then, we locate the region of interest to the lower third of the whole frame, which is usually where the lane appears when driving. Cross track error is used to calculate the angle from the center of the line to the center of the image, as shown in Figure 3.3. After successfully performing line following, we modified the system and tried to have the vehicle following two lines,

which is indicated as lane following. In detecting the lanes, Hough Transform is used, which is a commonly used technique to identify and detect features of a particular shape in an image. Here, we only used the classical Hough transform, which is most commonly used to detect regular curves like circles, lines, ellipses, etc.

4.2.1 Results for Single Line Following System

During the experiments, we use different error types and error locations to measure the results when driving in the course, as shown in Table 4.1. For the error types, there are software error and hardware error; and for error locations, there are location I: at the beginning of the course; location II: at the middle part of the course; and location III: at the end of the course, as shown in Table 4.2.

Table 4.1. Error types

Type of Errors	Error Names	Example Errors
Type I	Software Error	camera failure
Type II	Hardware Error	low battery

Table 4.2. Error locations

Location of Errors	Location Names
Location I	at the beginning of the course
Location II	in the middle of the course
Location II	towards the end of the course

Table 4.3. Results for indoor testing single line following on course track I.

July 25th 2019								
Test Platform	Go-kart							
Test Environment	Indoor							
Course Track 1	1st Run	2nd	3rd	4th	5th	6th	7th	8th
Running Time (second)	50	0	35	35	30	20	25	22
Error Type	I	II	I	I	I		I	
Error Location	I	I	II	I	III		I	
Running Distance (meters)	5	0	8	5	8	10	10	10

Table 4.3 demonstrates the results for testing single line following in the indoor environment. During the first few runs, the camera initialization latency was high and it was having trouble finding the line. Sometimes, when it finally found it, the vehicle drove for about 5 meters and then stopped. The reason for stopping is that the camera lost the line. The color thresholds were adjusted, and also the angle of the camera. On the 6th run, the vehicle was able to finish the course without stopping.

Table 4.4 shows the results for testing a single line following in the outdoor environment. The biggest difference from indoor testing is the change in lighting conditions. The natural light changes along with the position of the sun. When driving towards the sun, which means the license plate is in the shadow, the camera was having a difficult time detecting. The vehicle drove on the course in both directions. Table 4.4 shows that half of the runs fail because of shadow interference. More details will be discussed later in this chapter. Table 4.5 shows the results for the indoor test on course track III, which is the most difficult course we designed for the experiments. Figure 4.2 shows course track III in the garage area.

Table 4.4. Results for outdoor testing single line following on course track I.

August 9th 2019								
Test Platform	Go-kart							
Test Environment	Outdoor							
Course Track 1	1st Run	2nd	3rd	4th	5th	6th	7th	8th
Running Time (second)	55	0	50	0	0	45	40	40
Error Type	I	I	I	I	I		I	
Error Location	II	I	II	I	II		I	
Running Distance (meters)	5	0	10	5	0	10	5	10

Table 4.5. Results for indoor testing single line following on course track III.

Feburary 27th 2020								
Test Platform	RC car							
Test Environment	Indoor							
Course Track II	1st Run	2nd	3rd	4th	5th	6th	7th	8th
Running Time (second)	20	25	0	25	10	15	12	10
Error Type	I	I	II	I	I		II	
Error Location	II	I	II	I&II	II		I	
Running Distance (meters)	10	5	0	20	15	10	15	10

4.2.2 Results for Lane Following System

When testing the lane following system in the outdoor environment, we apply the same error types and error locations as in the indoor environment, as shown in Table 4.1 and 4.2. The biggest difference for lane following from line following is implementing the Hough Transform algorithm to detect and calculate the slopes of the two lines in the frame. Table 4.6 shows the results for the lane following in the indoor environment. Overall, the system performance is more stable in the indoor environment.

Figure 4.2. Course Track III for indoor testing environment.

Table 4.6. Results for indoor testing lane following on course track I.

January 7th 2020								
Test Platform	RC Car							
Test Environment	Indoor							
Course Track 1	1st Run	2nd	3rd	4th	5th	6th	7th	8th
Running Time (second)	20	25	10	25	10	15	12	10
Error Type	I	I	II	I	I		II	
Error Location	II	I	II	I&II	II		I	
Running Distance (meters)	10	5	8	10	10	10	5	10

4.3 License Plate Following

The other computer vision system in ADST is to detect the license plate of the leading vehicle and follow it by keeping a certain distance. There are two functions that the ADTS system has to fulfill: the first one is the plate detection function, which is to be able to detect the license plate by using just one camera; the second one is the speed control function, which is to control the speed according to the distance information from the stereo camera.

The license plate attached to the leading vehicle was made with two colors, red and yellow; but later on, pink and green were used instead because they are on the opposite position of the HSV color wheel, as shown in Figure 4.3. The plate detection function allows the camera to detect the license plate by its colors and shapes. The input of the plate detection function is the image input, and the output is cross track error (CTE), which is used in calculating the steering angle. At the same time, the speed control takes the distance information from the stereo camera, and the program will adjust the speed of the trailer vehicle according to the distance. The input of the speed control system is the distance, and the output is the speed.

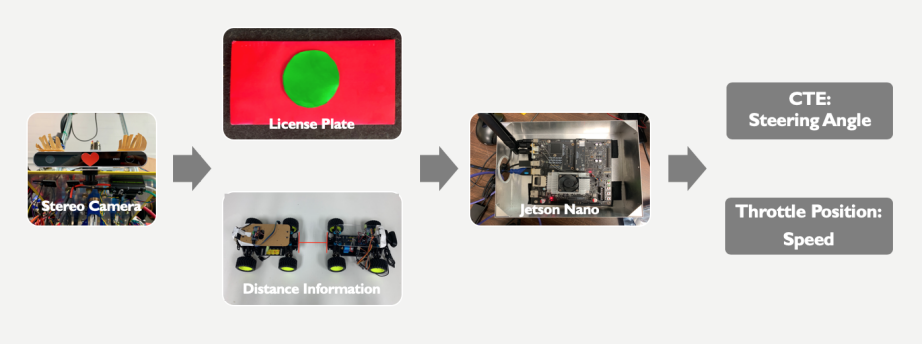

Figure 4.3. Automated disconnected towing system overview.

In the early stage of the original system design and testing, the plate is made with two colors in two shapes: the green circle in the center of a green rectangle. ADTS detects the license by using the color filter and edge detection. After ADTS successfully detects the license plate made with two colors, two real license plates are used to test the performance of ADTS. The Python library, pytesseract, is used to convert images to characters, such as numbers and letters. Using the real license plate is one step closer to the real-world scenarios, in the future.

The original image input is showing in Figure 4.4. The image input includes all the color information in RGB color space. The license plate used is a real Colorado plate, which contains letters and numbers, as well as some information related to the registration.

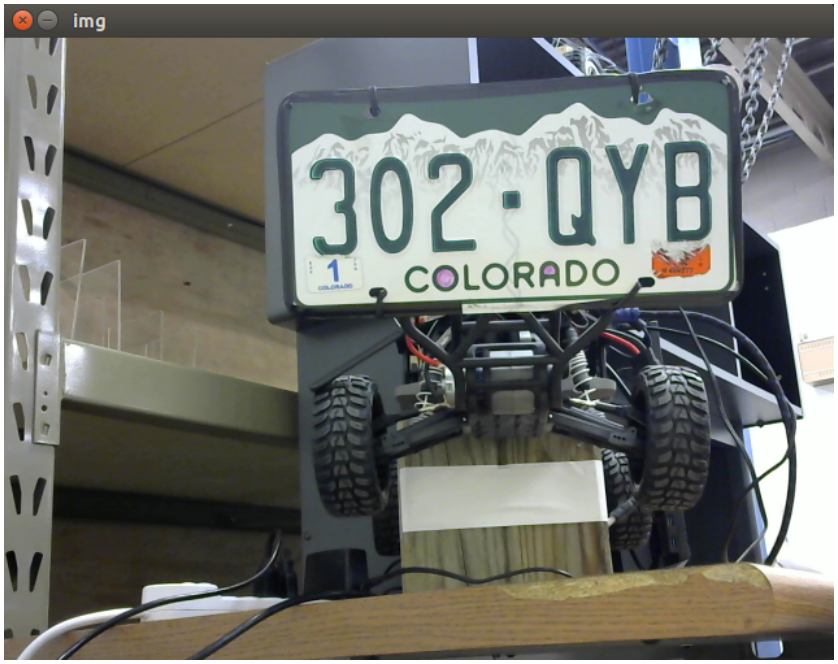

Figure 4.4. Grey scale image for license plate detection.

The first step in license plate detection is to take the input image and convert it into gray scale, as shown in Figure 4.5. It is a common technique in image processing to convert from RGB to grayscle, that is because sometimes, less information is needed for each pixel. When testing ADTS, the only thing needs to be specified is the single intensity value for each pixel. However, in RGB or BGR color space, the red, green and blue channels have equal intensity. and so it is only necessary to specify a single intensity value for each pixel, as opposed to the three intensities needed to specify each pixel in a full color image.

The next step is to use a thresholding technique to convert the image to black and white, and to only show the edges in of the objects in the image. Image thresholding is a type of image segmentation analysis technique, which is an effective method of converting an image from grayscale into binary image: foreground and background. When converting to black and white and an edges-only image, we filter out the unnecessary features, which is showing in Figure 4.6.

Figure 4.5. Gray scale image for license plate detection.

Figure 4.6. Black and white, and edges-only image for license plate detection.

After converting to edges-only image, the next step is to crop the image into the region of interest area and to mask the cropped image with the original image, as shown in Figure 4.7. The mask is a binary image consisting of zero- and non-zero values. The mask we are using is applied directly to the grayscale image of the same size. All pixels are zero in the mask are set to zero in the output image and the others remain unchanged, which is area of the license plate in our image input.

Figure 4.7. Masked image.

A small machine learning model is trained to perform the license plate recognition for letters and numbers. KNN, the K-nearest neighbor algorithm is used in ADTS, which is a simple, effective easy-to-implement supervised machine learning algorithm. KNN is one of the popular machine learning models to solve problems like classification or regression. The KNN model is trained on Jetson Nano and it detects and reads letters where it assigned floating values to each character and those are saved in a text file in 2D format.

We trained and tested the KNN model on two real license plates. The results are showing in Figure 4.8 and 4.9. The results were less than perfect. In detecting the Colorado plate, the model also captured other information that is not related to the license plate in the frame. There are many bounding boxes in one image, other than just one for the license plate. As for detecting the letters and numbers on the plate, the results are also not accurate. It only read half of the characters on the license plate correctly. However, the model filters out the state information, Colorado, at the bottom of the license plate successfully.

When detecting the Michigan plate, there is more interfering noise than the Colorado one, because of the College symbol, a capitalized m, is in the same line of the real characters in the plate. The detecting results are worse for this plate. Only one out of five characters on the plate are recognized.

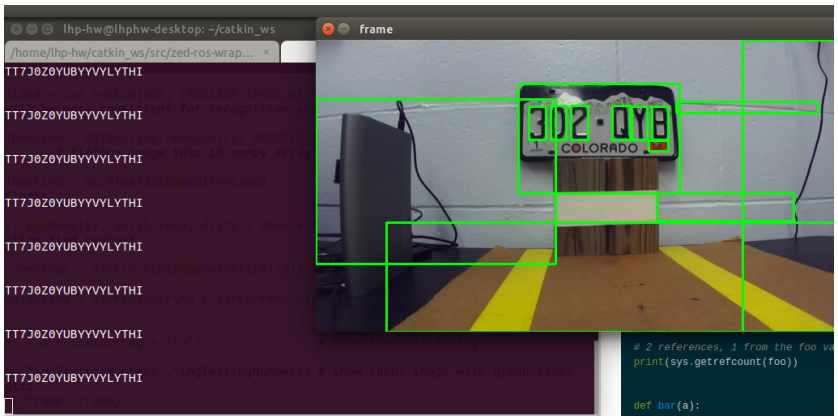

Figure 4.8. Results for license plate detection on a Colorado plate.

4.4 Results for ADTS

After trying the license plate detection by using a small KNN model, we decided to switch back to the previous plate made with two shapes in two contrasting colors, the pink rectangle with a green circle in the middle, as shown in Figure 4.3.

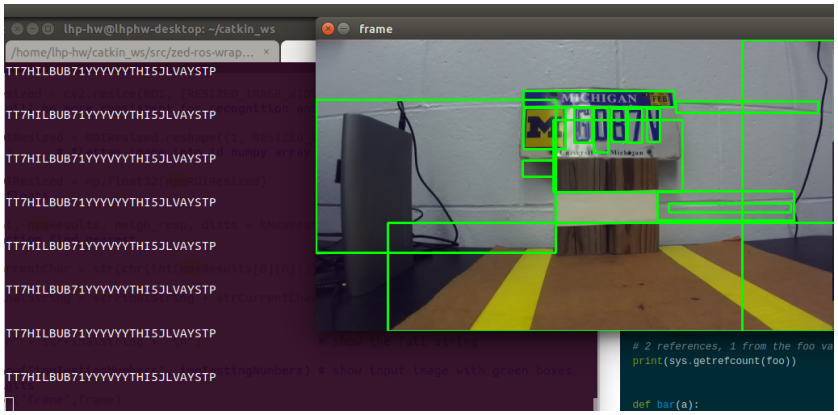

Figure 4.9. Results for license plate detection on a Michigan plate.

We have tested ADTS on both RC cars and go-karts. Table 4.7 shows the results on testing ATDS on go-karts on course track I. At first, a wagon was used and the license plate was attached on it as the leading vehicle. And then the wagon was pulled it manually to have the trailer vehicle follow the license plate on the cargo, as shown in Figure 4.10. After successfully perform ADTS on the cargo and the trailer go-kart, we have two go-karts driving at the same time: the leading vehicle performs the line following system and the trailer vehicle performs the license plate following.

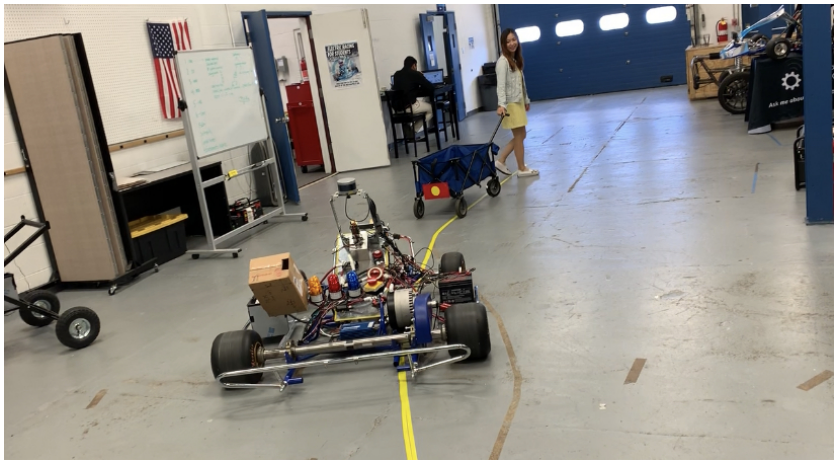

Figure 4.10. Using a cargo car as the leading vehicle to test ADTS.

After successfully tested ADTS with a wagon and a go-kart, we used two go-karts and tested on course track I and II, in both indoor and outdoor environments. Table 4.7 and table 4.8 show the results of the tests for course track I in both indoor and outdoor environment. And table 4.9 shows the results for the test on course track II in the outdoor environment. The results of the indoor tests again are better than the ones in the outdoor environment, because of the same reason for the line following tests: the change of the natural light according to the position of the sun. The shadow creates noises and interferes with the performance of the vision system.

Table 4.7. Results for indoor testing ADTS on course track I.

July 25th 2019								
Test Platform	Go-kart							
Test Environment	indoor							
Course Track 1	1st Run	2nd	3rd	4th	5th	6th	7th	8th
Running Time (second)	20	25	0	25	10	15	12	10
Error Type	I	I	II	I	I		II	
Error Location	II	I	II	I&II	II		I	
Running Distance (meters)	10	5	0	10	8	10	8	10

Table 4.8. Results for outdoor testing ADTS on course track I.

August 9th 2019								
Test Platform	Go-kart							
Test Environment	outdoor							
Course Track I	1st Run	2nd	3rd	4th	5th	6th	7th	8th
Running Time (second)	30	0	30	0	20	0	15	15
Error Type	I	I&II	II	I	I		II	
Error Location	II	I	II	I&II	II	I	I	
Running Distance (meters)	10	0	10	0	15	0	10	15

Table 4.9. Results for outdoor testing ADTS on course track II.

August 14th 2019								
Test Platform	Go-kart							
Test Environment	outdoor							
Course Track II	1st Run	2nd	3rd	4th	5th	6th	7th	8th
Running Time (second)	30	45	0	35	0	0	22	20
Error Type	I	I	II	I	I		II	
Error Location	II	I	II	I&II	II		I	
Running Distance (meters)	0	15	0	25	0	0	20	15

4.5 Discussion

During the tests for line/lane following and the ADTS systems, there are some problems discovered which are not expected before the tests. The biggest issue with all the vision systems is the shadow interference that is caused by the sun. When the sunlight is not directly on the license plate, the camera sometimes cannot detect the plate because of the shadow. Some times, with the impact of the shadow, the camera used is not able to recognize or detect anything, including lines or the license plates. This could also happen in cloudy days when there is not enough sunlight. But sunny days were chosen to do the tests on purpose because the clouds increase the possibilities of rains that might damage all the equipment used in the tests. Secondly, we were not able to test the ADTS on course track III, due to its availability and the weather conditions. However, we were able to build the track indoor in the garage area. Third, after the successful preformance of the license plate following by using the plate made with two colors, we implemented a small machine learning model, KNN, to recognize numbers and letters on the plate. The system is not accurate enough and is not stable during tests. It might be because the training set is not big enough, and the model needs longer training time. Lastly, we are only able to conduct tests on ADTS on go-karts on course track I and II, because one of the go-karts had mechanical issues later in the experiment.

4.6 Summary

We conducted a series of experiments to test the performance of the computer vision systems of ADTS. There are two vision subsystems, which are line/lane following and license plate following. The tests are conducted in both indoor and outdoor environments, and three different types of course tracks are designed and used in the experiments.

CHAPTER 5. CONCLUSION

5.1 Research Question

This study proposes an automated disconnected towing system (ADTS) that does not require a physical connection between a leading vehicle and a trailer vehicle by only using a computer vision system. Each of the leading and trailer vehicles has a depth camera mounted in the front. The cameras on the leading vehicle can perform lane detection. And the one on the trailer vehicle is able to perform license detection and to detect the distance from the leading vehicle and adjust the speed accordingly.

5.2 Summary of The Research

ADST contains two computer vision subsystems: a line following system for the leading vehicle and a license plate following system for the trailer vehicle. The line following system allows the leading vehicle to follow the lane line and drive by itself. And the license plate following system allows the trailer to detect the license plate that is attached to the back of the leading vehicle.

For line following and license plate following, each subsystem requires a stereo camera and an NVIDIA Jetson Nano. The camera used in this project is a ZED stereo camera. For line following and license plate detection, a regular webcam is able to perform with any problem. However, to measure the distance between the leading vehicle and the trailer vehicle, a stereo camera is needed. The other option for measuring the distance is to use radar. However, in this research, we are trying to only use one type of sensor, which is the camera, to perform all the tasks. Jetson Nano is used for ATDS because of its affordable price. The main propose of this project is to build an affordable, efficient automated disconnected towing system. The computer programs for the computer vision systems for ADTS are written in python, and the platform to run the programs is ROS.

5.3 Methodology

There are three different course tracks to test the ADTS, in different shapes, different lengths, and different floor surfaces. Course track I consists of 10-meter straight lines in both indoor and outdoor environments. Course track II is a combination of straight lines and curves, about 20-meter long and 8-meter wide. Course track III is an 8-shape course with about 10-meter straight lines and 2.5-meter radius curves.

ADTS is tested about 20-30 times on each course track, indoor and outdoor. And for each run, the current time of the experiment, error type, the running period, possible errors, possible stopping point are recorded.

5.4 Experiments

We conducted a series of experiments to test the performance of the computer vision systems of ADTS. There are two vision subsystems, which are line/lane following and license plate following. The tests are conducted in both indoor and outdoor environments, and three different types of course tracks are designed and used in the experiments.

5.5 Analysis

The purpose of this study is to only use a camera, computer vision technology, to perform lane following and license plate following in the automated disconnected towing system. By utilizing different tools, such as color filters and the Canny edge detector, the ADTS can perform the lane following and license plate following in the indoor environment with about 90% success rate. Once the tests were conducted in the outdoor environment, the shadow created by the sun is the main challenge for the vision system. For example, when the sunlight does not directly project on the license plate, the license

plate will be covered by the shadow. The ADTS will have a difficult time capturing the outlines and the colors of the plate. Although the outdoor testing for ADTS is less than perfect, there are so many possibilities and potential in the vision system. To solve the shadow interference could be one of the future research topics.

5.6 Problems and Future Studies

During the experiments for ADTS, there are some problems discovered which are not expected before the tests.

1. The biggest issue with all the vision systems is the shadow interference that is caused by the sun. When the sunlight is not directly on the license plate, the camera sometimes can't detect the plate because of the shadow. Some times, with the impact of the shadow, The camera we are using is not able to recognize or detect anything, including lines or the license plates. This could also happen in cloudy days when there is not enough sunlight. But we chose sunny days to do the tests on purpose because the clouds increase the possibilities of rains that might damage all the equipment used in the tests.

2. We were not able to test the ADTS on go-karts on course track III, due to its availability and the weather conditions. However, we were able to build the track III in the indoor environment and test on the RC cars.

3. After successfully preformed the license plate following by using the plate made with two colors, we implemented a small machine learning model, KNN, to recognize numbers and letters on the plate. The system is not accurate enough and is not stable during tests. It might be because the training set is not big enough, and the model needs longer training time.

4. Lastly, we are only able to conduct tests on ADTS on go-karts on course track I and II, because one of the go-karts had mechanical issues later in the experiment.

The problems mentioned above can be future studies. Using QR code, barcode, or PDF417 for license plate following is another possibility, instead of the plate we made with two colors. We can also look into other networks for license plate pattern recognition.

5.7 Summary

This chapter has concluded this research by reviewing the research question, the methodology used in this research, problems discovered during experiments, and future works.

REFERENCES

Advisor, A. (2019 (accessed November 7, 2019)). Complete towing capacity database 2018 [Computer software manual]. Retrieved from https://axleadvisor.com/towing-capacity/

ASIRT. (2019). Annual global road crash statistics [Computer software manual]. Retrieved from https://www.asirt.org/safe-travel/road-safety-facts/

Bacha, A., Bauman, C., Faruque, R., Fleming, M., Terwelp, C., Reinholtz, C., ... others (2008). Odin: Team victortango's entry in the darpa urban challenge. *Journal of Field Robotics*, *25*(8), 467–492.

Barickman, F. S., Smith, L., & Jones, R. (2007). Lane departure warning system research and test development. In *20th international technical conference on the enhanced safety of vehicles (esv) national highway traffic safety administration.*

Batavia, P. H. (1999). *Driver-adaptive lane departure warning systems*. Carnegie Mellon University Pittsburgh,, USA.

Bear, J. H. (2019 (accessed November 2, 2019)). The hsv color model in graphic design [Computer software manual]. Retrieved from http://www.lifewire.com/what-is-hsv-in-design-1078068

Biassoni, F., Ruscio, D., & Ciceri, R. (2016). Limitations and automation. the role of information about device-specific features in adas acceptability. *Safety Science*, *85*, 179–186.

Borkar, A., Hayes, M., Smith, M. T., & Pankanti, S. (2009). A layered approach to robust lane detection at night. In *2009 ieee workshop on computational intelligence in vehicles and vehicular systems* (pp. 51–57).

Broggi, A., & Cattani, S. (2006). An agent based evolutionary approach to path detection for off-road vehicle guidance. *Pattern Recognition Letters*, *27*(11), 1164–1173.

Brookhuis, K. A., De Waard, D., & Janssen, W. H. (2019). Behavioural impacts of advanced driver assistance systems–an overview. *European Journal of Transport and Infrastructure Research*, *1*(3).

Burzio, G., Guidotti, L., Perboli, G., Settanni, M., Tadei, R., & Tesauri, F. (2010). Investigating the impact of a lane departure warning system in real driving conditions: a subjectivefield operational test. In *European conference on human centred design for intelligent transport systems.*

Chen, Q., & Wang, H. (2006). A real-time lane detection algorithm based on a hyperbola-pair model. In *2006 ieee intelligent vehicles symposium* (pp. 510–515).

Chiu, K.-Y., & Lin, S.-F. (2005). Lane detection using color-based segmentation. In (pp. 706–711).

Committee, S. O.-R. A. D., et al. (n.d.). *Sae j3016. taxonomy and definitions for terms related to driving automation systems for on-road motor vehicles* (Tech. Rep.). tech. rep., SAE International, 2016. Cited on.

Crisman, J. D., & Thorpe, C. E. (1993). Scarf: A color vision system that tracks roads and intersections. *IEEE Transactions on Robotics and automation*, *9*(1), 49–58.

Deng, J., & Han, Y. (2013). A real-time system of lane detection and tracking based on optimized ransac b-spline fitting. In *Proceedings of the 2013 research in adaptive and convergent systems* (pp. 157–164).

Hamid, U. Z. A., Zakuan, F. R. A., Zulkepli, K. A., Azmi, M. Z., Zamzuri, H., Rahman, M. A. A., & Zakaria, M. A. (2017). Autonomous emergency braking system with potential field risk assessment for frontal collision mitigation. In *2017 ieee conference on systems, process and control (icspc)* (pp. 71–76).

He, Y., Wang, H., & Zhang, B. (2004). Color-based road detection in urban traffic scenes. *IEEE Transactions on intelligent transportation systems*, *5*(4), 309–318.

Hillel, A. B., Lerner, R., Levi, D., & Raz, G. (2014). Recent progress in road and lane detection: a survey. *Machine Vision and Applications*, *25*(3), 727–745.

Job, J. (1978). Numerical modelling of iceberg towing for water supplies—a case study. *Journal of Glaciology*, *20*(84), 533–542.

Jung, C. R., & Kelber, C. R. (2005). An improved linear-parabolic model for lane following and curve detection. In *Xviii brazilian symposium on computer graphics and image processing (sibgrapi'05)* (pp. 131–138).

Kammel, S., & Pitzer, B. (2008). Lidar-based lane marker detection and mapping. In *2008 ieee intelligent vehicles symposium* (pp. 1137–1142).

Kim, Z. (2006). Realtime lane tracking of curved local road. In *2006 ieee intelligent transportation systems conference* (pp. 1149–1155).

Koenigsberg, S. (2019 (accessed November 7, 2019)). Trailer accident statistics [Computer software manual]. Retrieved from `http://www.joshts.com/National_Trailer_Accident_Statistics.pdf`

Kong, H., Audibert, J.-Y., & Ponce, J. (2009). Vanishing point detection for road detection. In *2009 ieee conference on computer vision and pattern recognition* (pp. 96–103).

Kong, H., Audibert, J.-Y., & Ponce, J. (2010). General road detection from a single image. *IEEE Transactions on Image Processing*, *19*(8), 2211–2220.

Kornhauser, A., Atreya, A., Cattle, B., Momen, S., Collins, B., Downey, A., ... others (2007). Darpa urban challenge princeton university technical paper. *Tech. Rep.*.

Levinson, J., Askeland, J., Becker, J., Dolson, J., Held, D., Kammel, S., ... others (2011). Towards fully autonomous driving: Systems and algorithms. In *2011 ieee intelligent vehicles symposium (iv)* (pp. 163–168).

Lipski, C., Scholz, B., Berger, K., Linz, C., Stich, T., & Magnor, M. (2008). A fast and robust approach to lane marking detection and lane tracking. In *2008 ieee southwest symposium on image analysis and interpretation* (pp. 57–60).

Liu, D., Wang, Y., & Chen, T. (2019, 02). Application of color filter adjustment and k-means clustering method in lane detection for self-driving cars. In (p. 153-158). doi: 10.1109/IRC.2019.00030

Macesich, M. (2019 (accessed November 7, 2019)). Most-popular suvs, trucks, cars in america right now [Computer software manual]. Retrieved from `https://santanderconsumerusa.com/blog/most-popular-suvs-trucks-cars-in-america-right-now`

Meier, L., Tanskanen, P., Fraundorfer, F., & Pollefeys, M. (2011). Pixhawk: A system for autonomous flight using onboard computer vision. In *2011 ieee international conference on robotics and automation* (pp. 2992–2997).

Montemerlo, M., Becker, J., Bhat, S., Dahlkamp, H., Dolgov, D., Ettinger, S., ... others (2008). Junior: The stanford entry in the urban challenge. *Journal of field Robotics*, *25*(9), 569–597.

Ozgunalp, U., & Dahnoun, N. (2014). Robust lane detection & tracking based on novel feature extraction and lane categorization. In *2014 ieee international conference on acoustics, speech and signal processing (icassp)* (pp. 8129–8133).

Rasmussen, C., & Korah, T. (2005). On-vehicle and aerial texture analysis for vision-based desert road following. In *2005 ieee computer society conference on computer vision and pattern recognition (cvpr'05)-workshops* (pp. 66–66).

Ruyi, J., Reinhard, K., Tobi, V., & Shigang, W. (2011). Lane detection and tracking using a new lane model and distance transform. *Machine Vision and Applications*, *22*(4), 721–737.

Srivastava, S., Singal, R., & Lumba, M. (2014). Efficient lane detection algorithm using different filtering techniques. *International Journal of Computer Applications*, *88*(3).

Su, Y., Zhang, Y., Lu, T., Yang, J., & Kong, H. (2017). Vanishing point constrained lane detection with a stereo camera. *IEEE Transactions on Intelligent Transportation Systems*, *19*(8), 2739–2744.

Sun, T.-Y., Tsai, S.-J., & Chan, V. (2006). Hsi color model based lane-marking detection. In *2006 ieee intelligent transportation systems conference* (pp. 1168–1172).

Tan, H., Zhou, Y., Zhu, Y., Yao, D., & Li, K. (2014). A novel curve lane detection based on improved river flow and ransa. In *17th international ieee conference on intelligent transportation systems (itsc)* (pp. 133–138).

Turk, M. A., Morgenthaler, D. G., Gremban, K. D., & Marra, M. (1988). Vits-a vision system for autonomous land vehicle navigation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *10*(3), 342–361.

Urmson, C., Anhalt, J., Bagnell, D., Baker, C., Bittner, R., Clark, M., ... others (2008). Autonomous driving in urban environments: Boss and the urban challenge. *Journal of Field Robotics*, *25*(8), 425–466.

(US), F. H. A. (2007). *Highway statistics, 2005*. Federal Highway Administration.

Van Arem, B., Van Driel, C. J., & Visser, R. (2006). The impact of cooperative adaptive cruise control on traffic-flow characteristics. *IEEE Transactions on Intelligent Transportation Systems*, *7*(4), 429–436.

von Reyher, A., Joos, A., & Winner, H. (2005). A lidar-based approach for near range lane detection. In *Ieee proceedings. intelligent vehicles symposium, 2005.* (pp. 147–152).

Wang, C.-K., Huang, H.-P., & Shieh, C.-H. (2009). Dynamic analysis of the hybrid recharging system with super-capacitors on the armed cleaner robot. In *2009 ieee/asme international conference on advanced intelligent mechatronics* (pp. 1533–1538).

Wang, J., & An, X. (2010). A multi-step curved lane detection algorithm based on hyperbola-pair model. In *2010 ieee international conference on automation and logistics* (pp. 132–137).

Xiao, X., & Ma, L. (2006). Color transfer in correlated color space. In *Proceedings of the 2006 acm international conference on virtual reality continuum and its applications* (pp. 305–309).

Ying, Z., & Li, G. (2016). Robust lane marking detection using boundary-based inverse perspective mapping. In *2016 ieee international conference on acoustics, speech and signal processing (icassp)* (pp. 1921–1925).

Ying, Z., Li, G., Wen, S., & Tan, G. (2017). Orgb: Offset correction in rgb color space for illumination-robust image processing. In *2017 ieee international conference on acoustics, speech and signal processing (icassp)* (pp. 1557–1561).