5-2-2020

Autonomous Vehicle Lateral Control System

Mercer A. Mack

Dylan L. Drake

Nora E. Finegan

Jake S. Lloyd

FINAL REPORT

Autonomous Vehicle Lateral Control System

TEAM 46

Dylan Drake, Nora Finegan, Jake Lloyd, Mercer Mack

Project for: Faculty Sponsor

Faculty Contact: Dr. Hasan Ferdowsi, Electrical Engineering, Northern Illinois University, 590 Garden Rd. Dekalb 60115, 815-753-3963

Table of Contents

ABSTRACT

Autonomous vehicles used to be the talk of some distant future. Today, however, technology has advanced to the point where autonomous vehicles are now feasible and are starting to make a presence in the market. Since autonomy is a new feature for vehicles, it tends to be expensive to put into effect. Furthermore, the level of accuracy and efficiency of the autonomy is in its infancy and must be refined and perfected in order to make a reliable and safe product for consumer use. The Autonomous Vehicle Lateral Control System provides a lateral control system implemented in an open source autonomous vehicle simulator called CARLA that allows for total lateral autonomy when keeping and changing lanes. The project furthers research at Northern Illinois University and offers a cost-effective solution for fully autonomous lateral control. In the simulation, the system takes data from virtual sensors representing an on-board tracking camera, depth camera, and LIDAR (Light Detection and Ranging). It uses the data to execute a variety of algorithms that tell the vehicle to perform a lane change if it is safe to do so and if an object is obtruding the vehicle's path of travel or to keep a lane. Provided successful operation of the autonomous vehicle, the vehicle moves at a safe speed within a lane on a track. Once an object is detected in the path of the vehicle, the vehicle decides if the lane next to it is available and if it is permissible to perform a safe and steady lane change. Upon a successful lane change, the vehicle controls operation as before. The lateral control system used in CARLA can be applied to a 1:10 scale remote-controlled (RC) car. The outcome of the Autonomous Lateral Control System provided here would impact the market for autonomous vehicles by bringing the cost of autonomy down for full size vehicles while increasing the level of reliability and efficiency of autonomy for full size vehicles.

1 INTRODUCTION

1.1 Background

The ambition of self-driving vehicles has been driving many initiatives in the past twenty years, such as the Defense Advanced Research Projects Agency (DARPA) Challenges in the early 2000s skyrocketed the development of autonomous vehicles [9]. Self-driving cars are the future of the automotive industry and have progressed enough to break into the consumer market, as illustrated in [7]. Features have been developed and are active in select vehicles today, such as Cadillac's Super Cruise system and lane centering [7]. Furthermore, companies are continually researching and testing highly autonomous vehicles [7].

The technology, however, is limited. [8] explains that even the most premiere autonomous vehicles still require a degree of user input, especially in the area of lateral control, or lane changing. Lane changing algorithms that currently exist are either far too complex or conservative [8]. More complex solutions rely on a significant amount of data to be processed by complicated statistical methods; in the end, the car almost never changes lanes, as the decision cannot be made in time [8]. More conservative solutions have the same outcome, simply because the algorithm never meets its safety requirements for a lane change [8]. A method that permits autonomous vehicles to safely and effectively change lanes must be developed to continue the advancement in autonomous vehicle technology.

The process, however, has and continues to pose significant challenges. Autonomous vehicles must be able to correctly react in all situations. It is, however, very difficult to establish algorithms that mimic, understand, and perceive human behavior [8]. Regarding lane changing, an algorithm is needed that permits a vehicle to effectively change lanes in a responsible manner.

In addition to industry demand for a solution, the College of Engineering at NIU needs one as well. Research in the area of self-driving cars must be pushed to the next level so that the university can have an impact in the automotive industry. Furthermore, prospective students who are passionate about autonomous vehicle development may then build off the research and development obtained from the project. Lateral control ought to be developed at Northern Illinois University as a stepping-stone in its engineering program's autonomous vehicle research.

1.2 Purpose of the Project

The purpose of the project is to create a lateral control system that allows a self-driving car to safely change lanes with full autonomy. The original plan was to achieve the system using a lane changing algorithm in conjunction with sensors mounted on a remote controlled (RC) car. Due to outside circumstances mid-way through the project, the new plan was to implement an autonomous longitudinal and lateral controller with lane changing capabilities in CARLA, an autonomous vehicle simulation tool. The uniqueness of the project stems from its cost-effectiveness. This characteristic is sought after as the project progresses with industry application in mind and expands the research and development within the field of autonomous vehicles at Northern Illinois University.

1.3 Previous Work Done by Others

1.3.1 Existing Products

Common lane assist technologies are lane departure warning, lane keeping assist, lane centering assist, and blind spot detection. Lane departure warning uses a camera to monitor the lane markings and alerts the driver if he or she is leaving the lane. The camera is usually attached to the top of the windshield. An exception would be Nissan having their camera mounted on the rear of the car, stating lane departure warning is typically used on straight roads [3].

The next step up from the technology is the lane keeping assist. If a driver lets the car drift too much, the car itself steers away from the edge of the lane. Another advance in these systems is the lane centering assist. As the name suggests, it helps keep the car in the middle of the lane. While blind spot detection can use a camera, similar to the three technologies described above, it does not need to. The feature can use sonar or radar sensors to look back and to the side. It warns the driver if a car quickly comes into his or her blind spot [3].

Amazon entered the autonomous vehicle industry in January of 2017 when the company secured a patent for technology that assists vehicles performing lane changing tasks via a roadway management system. The system is an internet-based datacenter containing information on the roadway where the vehicle is located. When connected, the vehicle is provided with information on the best lane to drive in considering time of day and speed [4].

Tesla has a driver assistance system called Autopilot. It comes in two packages, Autopilot or Full Self-Driving Capability. Despite the names, a driver is still envisioned to be fully attentive behind the wheel. These systems use eight cameras, one radar, twelve ultrasonic sensors, and an onboard computer to safely provide their driver assistance features. Full Self-Driving Capability has the feature for automatic lane changes; however, it must have another feature called autosteer employed in order to use it [5].

1.3.2 Patent Search Results

There are plenty of patents that give good insight to a wide range of topics for autonomous vehicles. These patents can help aid with the process and methodology of the project. A patent for Uber Technologies, Inc. is a good source of information. They used onboard map data and real-time sensor data from the car to synthesize the best overall route for the vehicle to take and to help the car make more accurate decisions in real-time [1]. The idea of using current map data on board the vehicle is an interesting concept, but due to the small scale of the project and the lack of map data available, the source would not be used exactly. However, their methodology of sensor data fusion could be a good reference when it comes time to analyze the different data sets coming in from each sensor and using the data within an algorithm to control the vehicle safely.

Another patent found is from Google, Inc. where they solidified a method to help autonomous vehicles track object trajectories, such as other moving vehicles on the road [2]. The patent could be a reliable source of information when it comes to tracking other objects on the test course for the RC car. However, the complexity of the patent is beyond the scope of the project because the project does not call for recognition and tracking of moving

objects. It does not mean the patent cannot give insight to tracking stationary objects. The methodology and thought process for the patent could be an immense help for the project.

The patent from Amazon Technologies, Inc. uses a cloud-based road management system to help inform the autonomous vehicle about the lane and roadway conditions and helps the vehicle decide based on the information [6]. The method of using a cloud-based data to inform vehicles of where they are and what to expect on the given road they are on is an interesting concept. Such a way of controlling the vehicle is useful because in its pure form, it does not need any sensors. It uses the data from the cloud to make the vehicle's decision. The added benefit of having sensors on the vehicle means that the cloud data can constantly update in real-time based on the input and feedback from the sensors. The method can relate to the project by making a database full of information about the test track and using what is in the database along with sensor input to inform the car on what decision to make. The application could be useful, especially when it comes to the real road. However, since the project is restricted to the test track, the added benefit of cloud data of the road may not be beneficial for the scope of the project.

1.4 Brief Overview of the Report

Self-driving technology is an old field of research but is still incredibly active. Many autonomous vehicles have been prototyped and even put onto the streets. The vehicles, however, are limited in their autonomy; therefore, research into the design and creation of highly autonomous vehicles is a growing field. The project being presented continues innovation in the exact subject manner. With a modest budget in mind, the highest-quality hardware possible was selected in order to create an autonomous vehicle capable of higher degrees of autonomy than in commercially available self-driving cars. The higher degree of freedom is built upon a lateral control system that allows a self-driving car to safely change lanes with full autonomy. While the original objective was to use a lane changing algorithm in conjunction with sensors mounted on a remote controlled (RC) car, the new objective was to implement a lane-changing algorithm in the CARLA vehicle simulator.

The uniqueness of the project stems from its cost-effectiveness and its ability to be tested on a college campus. These characteristics are sought after as the project has the ambitions of scalability to full size vehicles and the expansion of research and development at Northern Illinois University. The success of the project is dependent upon working within the constraints stated below and the established optimal design. Research, design, and production completed within the constraints of a modest budget given the optimal design specifications is going to yield an autonomous vehicle capable of high degrees of autonomy, which is going to serve as a foundation for future researchers.

2 PROJECT DESIGN

The design for an Autonomous Vehicle Lateral Control System is a complex design to think about; there are many different designs that pose a possible solution to vehicle autonomy. In order to make an optimal design, three possible solutions were taken into consideration, and the optimal design is a synthesis of all three. The following paragraphs go into detail about each alternative design and concludes with the final design of the project and why the design was decided as so.

The first alternative design uses the Structure Core Imaging Processor, the Slamtec RPLIDAR (Light Detection and Ranging), a NVIDIA Jetson TX2, and an Arduino UNO R3 in conjunction with the autonomous vehicle (AV). The camera is positioned in the front of the car and its data is used to handle lane keeping and lane detection. The LIDAR is mounted onto the chassis of the vehicle and its data, along with the camera's data is used to aid in lane changing. Additionally, the LIDAR is the primary sensor for object detection. The TX2 and Arduino are the main hardware components that control the vehicle.

The camera handles the lane keeping task because it is a sensor that can successfully detect the lines of the lane and the type of lines of the lane [1]. It is an important task that keeps the vehicle in the center of the lane and informs the vehicle if a lane change is permissible. Furthermore, the camera and LIDAR work in unison to automate the lane changing process. First, the camera gathers data to make sure the vehicle is in a lane. Then, the vehicle decides on whether to continue its current path or adjust its course based on the camera data to ensure lane keeping and lane centering are achieved. Simultaneously, the LIDAR is constantly scanning for obtrusive objects in the path of the vehicle. When an object is detected in the path of the vehicle, the camera checks the type of lane line to confirm if a lane change is permitted. The LIDAR then detects if there are any obstructions in the next lane that inhibit the vehicle from making a safe lane change.

The hardware that is physically controlling the vehicle is the Arduino microcontroller and the motor, and the main computing hardware is the NVIDIA Jetson TX2. These are powered by an onboard rechargeable battery. When the sensors take data, they send it to the TX2, and the TX2 then performs the necessary calculations based on the algorithms for lane changing/keeping and object detection. The TX2 decides several things simultaneously in order to keep the car autonomous. The TX2 decides if the car is in the center of the lane and if there are any objects obstructing its path; if there are objects in its path, then the TX2 decides if the next lane is available. When the decisions are made, the TX2 outputs a signal to the Arduino. The Arduino then drives the motor that controls the wheels and the steering mechanism to whatever the TX2 has decided the vehicle to do.

One of the benefits of the first alternative design is the high computing capacity of the NVIDIA Jetson TX2. With the processor, it is possible to hook up multiple high-end sensors and perform rigorous calculations and algorithms based on the data taken by these sensors. The TX2 also makes it possible to host multiple neural networks that can be used for machine learning with the algorithms for lane detection and object detection. Another benefit of the design is the extension of the Arduino microcontroller to control the motor for the vehicle. Having a separate controller for the vehicle makes the system more modular and relieves some of the stress on the TX2, allowing it to solely be used for computing the algorithms of the design. Also, it is beneficial to have a LIDAR on board the vehicle because LIDARs are very accurate at mapping their environment and have a 360° range. Therefore, the vehicle would be aware of its entire surroundings and would be able to detect an object within 360°. The computing power of the TX2, modularity of adding the Arduino UNO R3 for control and the accuracy and range of having an on-board LIDAR for object detection give the design possible benefits that prove effective for the goal of the project.

The first alternative design does come with some drawbacks. One of the drawbacks is that most of the budget is allocated to the NVIDIA Jetson TX2 and the Structure Core imaging

sensor. Combined, these cost about $700; that does not leave much room to purchase additional sensors. It makes the only viable option for a LIDAR sensor the Slamtec RPLIDAR A1, which is at the lower end of the scale when it comes to quality LIDAR sensors. There is also a lack of sensors in the design. Having the computing power of the TX2 and only connecting one high-end and one low-end sensor to it is under-using the capabilities of the TX2 and could possibly make the autonomy of the vehicle less accurate and reliable. Another drawback is the communication problems that could ensue between the Arduino and the TX2. The TX2 must be programmed in Python and the Arduino must be programmed in the C language, so communicating between these devices could be troublesome due to the lack of programming language uniformity. The expensive microprocessor and imaging sensor lead to a low-end LIDAR, under-using the capabilities of the hardware available in the design hinders the autonomy, and the difficulty of communicating between different devices in different languages give the design possible drawbacks that prove ineffective for the goal of the project.

The second alternative design uses a different combination of sensors than the first alternative design. The sensors used are ultrasonic, radar (Radio Detection and Ranging), and a camera. The camera, the Structure Core Imaging Processor, is placed in the front of the car for lane keeping and assisting in lane changing. Two Ultrasonic Sensor's HC-SR04 are used, one on each side of the vehicle. Ultrasonic sensors work well in close ranges and are optimal for furthering a vehicle's sensing capabilities [3]. Therefore, the two ultrasonic sensors aid in lane changing. They serve to rid of any blind spots, detecting objects that the radar is unable to pick up to ensure safe lane changes. The FMCW Distance radar sensor is the main sensor that detects objects. It is placed on top of the vehicle, and since it scans 360 degrees, it is used to detect any obstructions in the vehicle's current path of travel and in the adjacent lane should a lane change need to occur.

The design uses the NVIDIA Jetson TX2 and the Raspberry Pi 3- Model A+. Both pieces of hardware are powered by a rechargeable onboard battery. The Jetson TX2 serves as the principle computing hardware, while the Raspberry Pi handles a portion of the computing and the motor control of the vehicle. The input data from the camera and the radar are sent to the Jetson TX2. It performs calculations corresponding to the lane detection, lane keeping, and object detection. The HC-SR04 ultrasonic sensors send their input to the Raspberry Pi. The object detection calculations from the Raspberry Pi and the computations from the Jetson TX2 ultimately lead to a decision being made regarding the vehicle's subsequent path, and the Raspberry Pi implements the decision with the RC car.

There are advantages to the second alternative design. Since three different sensors are used, there are multiple sources of data used in lane keeping, lane changing, and object detection. It minimizes error and thus improves the reliability of the prototype. An ultrasonic sensor, for example, may detect an object in the adjacent lane that the radar is unable to pick up. Additionally, ultrasonic sensors are inexpensive yet are effective in short range applications [3]. It is important because one of the specifications for the project is to develop a cost-effective solution. radars are also more reasonably priced and are cheaper than LIDAR's [2]. Furthermore, radars work well in long ranges and in many different weather conditions [2]. Even though the project is developing an RC car in an indoor environment, these radar benefits are important to consider in producing a scalable prototype. The use of Raspberry Pi is beneficial due to both the Jetson TX2 and the Raspberry Pi operating with Python coding

language. Using one coding language for the project makes the process easier when writing and modifying code. Additionally, the Raspberry Pi is user-friendly, as its operating system comes with available software for education, programming, and general use.

There also disadvantages to the second alternative design. While radar is cheaper than LIDAR, it is not as accurate as its counterpart; radar may falsely detect objects due to reflection [6]. A significant aspect of the project is its ability to accurately detect any obstructions, and thus a less accurate sensor decreases the prototype's reliability. Furthermore, the use of three different sensors complicates sensor fusion. Given the time constraint of the project, it may be difficult to ensure all three sensors properly function together and may take time away from developing a lane-changing algorithm, which is the goal of the project. Using two different computing hardware also presents a challenge; one can run into issues with cross-device communication. The type of issues that have the potential to be present are along the lines of what to do if the TX2 receives conflicting information about if an object has been detected from the radar and ultrasonic sensors.

For the third alternative design, the focus is on downgrading the NVIDIA Jetson TX2 to the NVIDIA Jetson Nano in order to make room in the budget for more high-quality sensors. The design also does not include an Arduino UNO R3 and changes the combination of sensors. Instead of having a variety of different sensors, for example, a LIDAR, radar and camera, the design uses the Structure Core Depth Camera by Occipital and two Intel RealSense D435i Depth Cameras to handle all the autonomous needs. The design was inspired by an article published by Cornell University [4]. The article is a cumulation of the research done by Cornell's students and professors comparing the data accuracy of LIDARs to cameras for mapping a self-driving car's environment. The results show that using two cameras on opposite sides of the windshield, and at a birds-eye view, give nearly identical results to that of a LIDAR [4]. Using the information, the third design only uses the cameras and ultrasonic sensors mentioned above. The Structure Core Depth camera is mounted on the front of the vehicle with a birds-eye view angle; the camera is responsible for lane keeping and lane detection, as well as detecting objects directly in front of it. The two Intel RealSense Depth cameras are responsible for object detection in other lanes. The cameras are mounted at 60° from the vehicles center at a birds-eye view angle; it allows the cameras to detect objects approaching from the next lane, as well as objects directly next to the car. All cameras work in unison and aid each other in object detection and lane changing.

The goal of the design is to free up budget space for more camera sensors without risking the quality of the camera sensors, so the only option was to downgrade the NVIDIA Jetson TX2 to the NVIDIA Jetson Nano, while eliminating the Arduino UNO R3 Microcontroller and the Slamtec RPLIDAR A1. The downgrade from the Jetson TX2 to the Jetson Nano is a reasonable downgrade that saves $200 without sacrificing a significant amount of computing power. The Nano has less GPU capability with 128 NVIDIA CUDA cores compared to the TX2's 256 NVIDIA CUDA cores. The Nano's CPU also has less capabilities with a Quad-core ARM Cortex-A57 MPCore processor compared to the TX2's Dual-core Denver 2 64-bit CPU and quad-core ARM A57 complex. The Nano has half the memory and storage of the TX2 and does not come with an onboard Wi-Fi chip. However, the rest of the specifications are nearly identical or do not apply for the use case. It is clear some computing power is sacrificed with

the downgrade. However, the Jetson Nano can handle the demands of the added camera sensors along with controlling the autonomous vehicle.

The biggest change in the third design is the combination of sensors used. Using three cameras instead of a variety of sensors may seem to take away from the capabilities of the autonomy of the vehicle. However, knowing that two or more depth cameras have the same mapping capabilities as a LIDAR shows that the depth camera sensors used in the design can do anything another sensor can, with the added benefit of object recognition and detection. These depth cameras detect objects and map its environment up to a radius of ten meters with a viewing angle of 160° for the structure core, and 90° for the Intel RealSense. Both cameras come with an on-board Bosch BMI055 IMU; adding two more IMU's to the original design solidify the localization of the vehicle and give the design a more accurate navigation system. Building off the placement of the cameras mentioned above, with the viewing range of 160° for the front positioned camera and 90° for the two side cameras, with strategic placement of an overlap of 30° of the viewing angles, the design gets a total viewing angle of 280° of the cars environment, which is beyond what the design needs for full autonomy.

There are some benefits of the design compared to the initial design. The money saved from downgrading the Jetson TX2 to the Jetson Nano allows for more higher-end sensors to be purchased, leading to more accurate data and overall better autonomy of the vehicle. Also, using three cameras instead of a combination of sensors allows the data taken to be less diverse, thus combining the data together is less work compared to taking data from several types of sensors. Using the Jetson Nano to drive the vehicle instead of the Arduino UNO R3 is beneficial because the programming language for Arduino is the C language, whereas the language decided for the project is Python. Communication between these two devices is integral to the success of the project, so eliminating the need of cross device communication relieves some stress on the project. Also, uniformity of the programming language makes designing and writing code for the project easier. Being able to attain more accurate data, easier data fusion and analysis, and eliminating cross device communication, while adding programming language uniformity, gives the third design possible benefits that prove effective for the goal of the project.

Although the design saves money due to eliminating LIDAR and the more advanced TX2, there are issues the modifications cause. First, relying solely upon image processing demands a more advanced computer algorithm. Vast amounts of data have to be compiled to run through the algorithm, which requires a significantly larger amount of computing power and graphics processing. In order to afford more cameras, the design proposes downgrading the TX2 to the NVIDIA Nano, which might not be able to handle an even more advanced algorithm than before. A more advanced algorithm not only means problems for the downgraded computer, it also means problems for the algorithm writer. The additional time and energy needed to write a more robust and smarter algorithm may take up too much design time, resulting in the inability to meet time constraints. Another problem it introduces is industry compatibility. A goal for the project is to create a lateral control system scalable to full scale vehicles. Relying solely upon image processing is not acceptable in industry, as cameras cannot be relied on by themselves during adverse weather conditions. According to [5], autonomous vehicles that rely primarily on image processing and lack LIDAR sensors are inherently more dangerous, as even the smartest of algorithms fail. The possibility of not being able to run a more advanced

algorithm, time constraints, and the lack of industry compatibility give the alternative design hang-ups that prove ineffective for the goal of the project.

The optimal design decided upon for the project is a synthesis of different parts of all three alternative designs presented above. The optimal design includes the NVIDIA TX2 that handles all algorithm computing, sensor fusion, and control of the autonomous vehicle. The design also includes the intel RealSense D435 depth camera, the intel RealSense T265 tracking camera, and the Slamtec RPLIDAR A2M8. Both the depth camera and the tracking camera are mounted on the front of the vehicle and are used for lane keeping and lane centering, as well as object detection and localization. The LIDAR is mounted in the center of the vehicle to get a 360° view of the vehicle's surroundings and is responsible for object detection within a given radius of the vehicle.

The optimal design uses the Jetson TX2 instead of the Jetson Nano because the time the processor takes to compute the various algorithms in the design is crucial for the real time application of the project. The TX2 is the superior computer when it comes to speed and capacity, so the real-time constraints of the project are not a concern for the computer, and the TX2 has more than enough capacity to handle the number of sensors included in the design. The sensors chosen in the design give the vehicle a highly accurate variety of data that make the vehicle efficiently autonomous. The decision to use the intel RealSense depth and tracking cameras instead of the Structure Core depth camera is beneficial because now the design uses two separate cameras to handle two different tasks, all while being $50 cheaper. The tracking camera handles the localization and lane keeping of the vehicle, while the depth camera aids in object detection and lane keeping/changing. Having two cameras control two different tasks adds to the modularity of the project and gives more accurate data with respect to each task. The decision to use the Slamtec RPLIDAR A2M8 sensor, which is a midrange LIDAR sensor, is because LIDAR sensors are good for object detection at a variety of distances and various weather conditions. Although the vehicle is tested indoors, one of the project goals is to have it be scalable to a full-size vehicle. Since the weather influences the camera sensors, the LIDAR is a necessity to ensure full autonomy in all weather conditions.

Unfortunately, due to outside circumstances mid-project, the optimal design was not able to be realized as expected. The team could no longer work on the hardware components, and the project transitioned into a simulation. The team worked in CARLA, an open-source simulator for autonomous driving [14]. With the transition, CARLA was used to develop an autonomous longitudinal and lateral controller for a simulated vehicle that would change lanes upon detection of a stationary vehicle in its current lane of travel. A sensor was used in CARLA to represent that hardware components of the project as best as possible.

The following subsections provide further details of the components of the optimal design.

2.1 Optimal Design

2.1.1 Objective

The goal of the project is to develop a small-scale, self-driving car with full lateral autonomy. The vehicle must be able to "lane keep," or stay within the boundaries of a road's

lanes, and it must also be able to detect stationary objects within its path of travel. Then, the most important outcome of the project is the vehicle's ability to safely and efficiently change lanes without user input. Once an object has been detected in its path of travel, the vehicle must recognize if there is a lane available to change to. The vehicle must change lanes only if there are no objects detected that may interfere with or pose a threat to both the vehicle and/or the object. The vehicle must come to a stop if it cannot change lanes safely, but if the vehicle does change lanes, it is expected to remain in that lane as it did the previous lane until it reaches another obstruction.

Along with the remote-controlled (RC) car, the optimal design uses a depth camera, a tracking camera, a LIDAR (Light Detection and Ranging), and a NVIDIA Jetson TX2. The cameras are to be positioned in the front of the car. The depth camera's main task is to lane keep, although it may be also be used to aid in object detection. Furthermore, when the vehicle needs to change lanes, the depth camera's data must be used to identify the lane line to determine if a lane change is permissible. The tracking camera task is to be used for localization, so that tasks as lane keeping can be accomplished. The LIDAR is to be mounted on the chassis of the vehicle, and since it scans 360 degrees, its primary job is to detect any objects in the vehicle's path of travel. The main computing hardware, and the hardware physically controlling the vehicle, is the NVIDIA Jetson TX2. When the sensors take data, they must send it to the TX2 to perform the necessary calculations based on algorithms for lane changing/keeping and object detection. The TX2 must decide several things simultaneously in order to keep the car autonomous. The TX2 must decide if the car is in the center of the lane and if there are any objects obstructing its path; if there are objects in its path, then the TX2 must decide if the next lane is available or not. When these decisions are made, the TX2 must then drive the motor that controls the wheels and the steering mechanism to whatever it has decided the vehicle to do.

A block diagram illustrating the components and their functions in the overall system is shown in Figure 1.

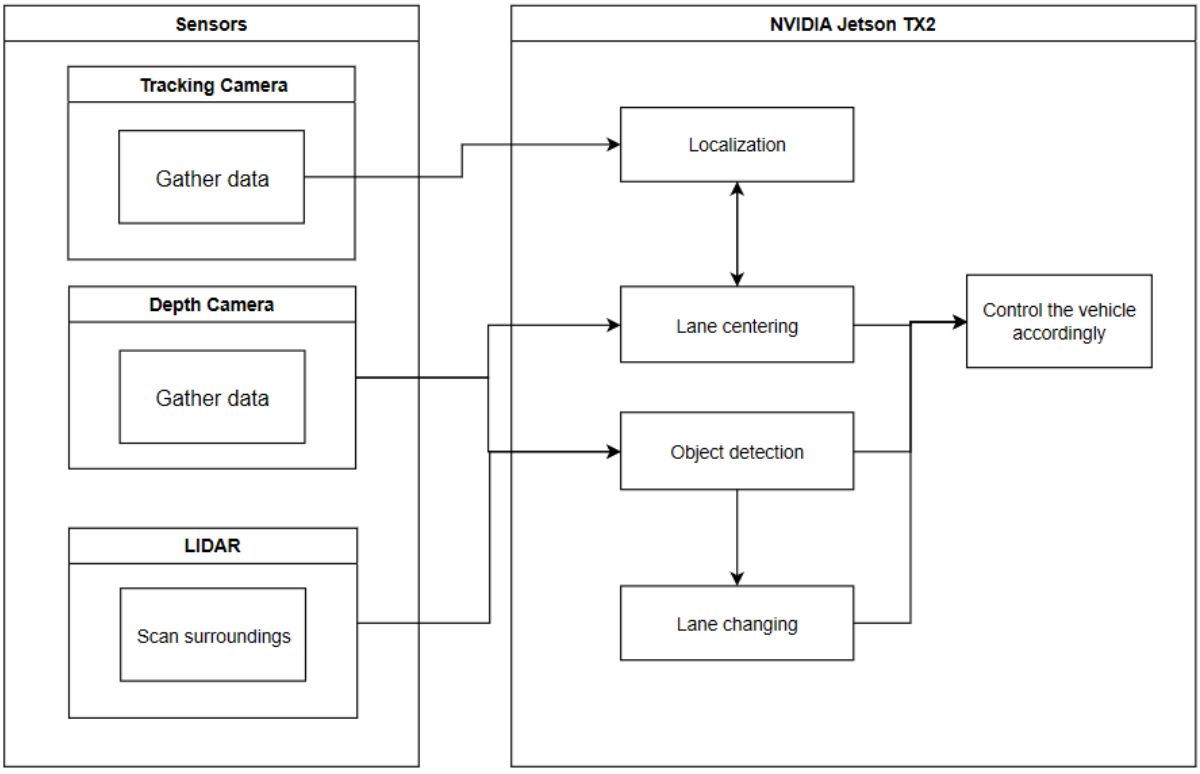

Figure 1. A block diagram illustrating the sensors and processor for the system and their functions.

While the team was not able to realize the design with the sensors, TX2, and RC car due to the necessary change in direction mid-project, these components were represented by the CARLA simulator as best as possible to successfully complete the objective of the project.

2.1.2 Subunits

2.1.2.1 LIDAR

The autonomous lane changing system for the project must have at least one device to behave as its “eyes.” The LIDAR serves to perform the intended task.

The project uses the Slamtec RPLIDAR A2M8 360 Degree Laser Scanner Kit. The device is a two-dimensional laser scanner and uses a triangular measurement system. The measurement system is illustrated below in Figure 2 from [11] and works in the following manner: Laser light is emitted by the LIDAR and reflected off an object. A linear Charge-Coupled Device (CDD) sensor detects the reflected light. The distance between the laser and where the light hit the CCD is used to calculate the distance between the LIDAR and the object. The laser has been tested to reach and is safe in the event it shines in the eyes of a human or animal [11].

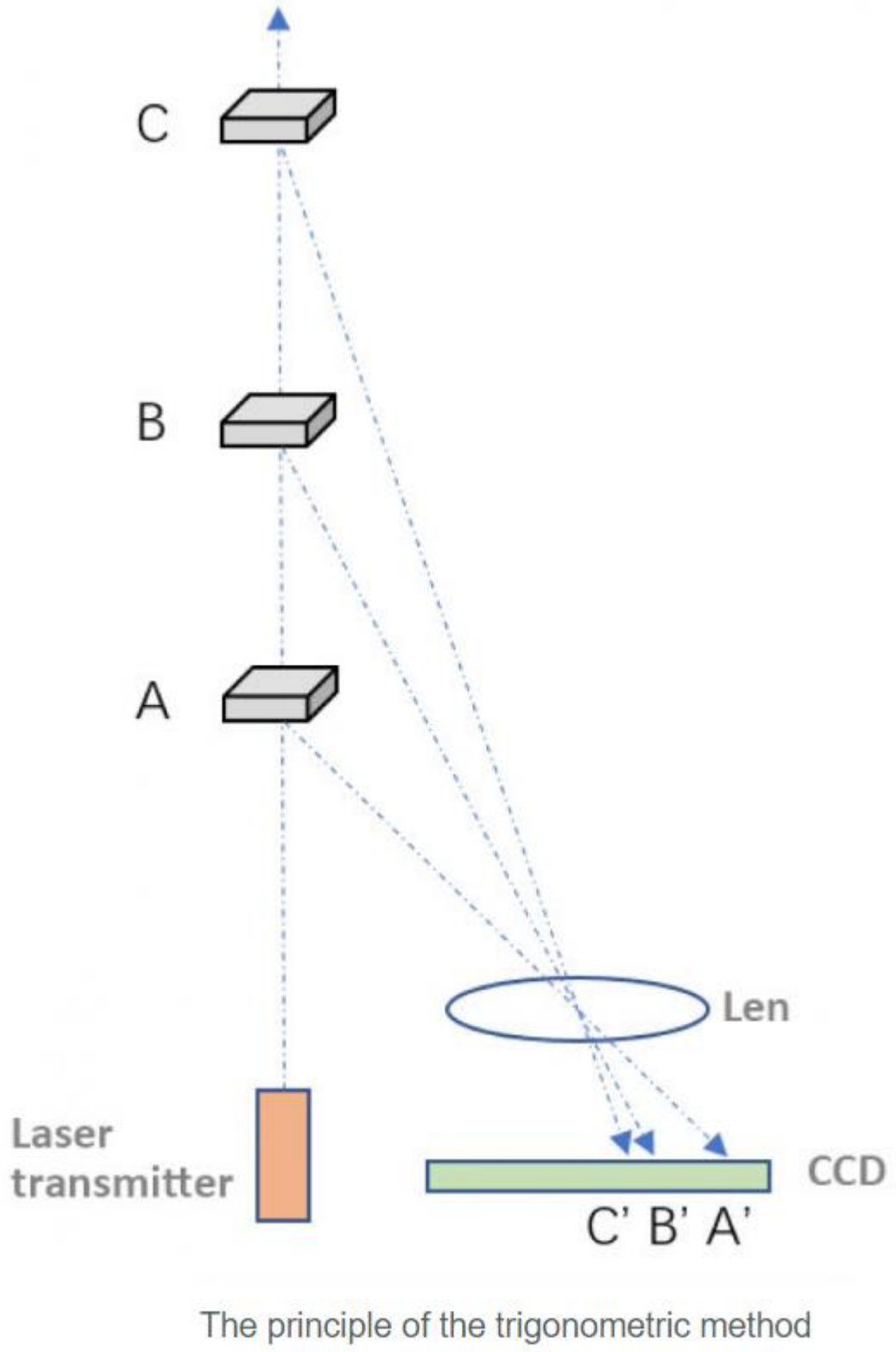

Figure 2. The triangular measurement system used by the Slamtec RPLIDAR A2M8 [11].

The sample of laser ranging per second is usually around 4,000, though the sample frequency can be between 2,000 and 8,000 Hz. Each sample duration is 0.25 ms. The scan rate falls between 5 – 15 Hz, and in most cases is 10 Hz. While the LIDAR's angular range is a full 360°, the distance ranges from 0.15 – 12 m. The distance range is based on white objects with 70% reflectivity. Its angular resolution is typically 0.9°, yet it can be between 0.45 – 1.35°. The LIDAR rotates with a brushless motor with a non-contact drive [12]. Brushless motors are beneficial due to their low noise and extended lifetimes, as there is no friction inside. The values discussed in section 2.1.2.1, with the additional of the LIDAR's weight and power consumption, are in Table 1 below. The LIDAR was tested by being connected to the NVIDIA Jetson TX2 that is discussed later in the subunits section.

Sample rate	4000 Hz
Sample duration	0.25 ms
Scan rate	10 Hz
Angular resolution	0.9°
Angular range	360°
Distance range	0.15 - 12 m (white objects w/ 70% reflectivity)
Power	5 V, 450 mA - 2W
Motor	Brushless
Data	2D point cloud
Measurement System	Triangular
Weight	190 g

Table 1. Summary of Slamtec RPLIDAR A2M8 specifications.

2.1.2.2 Depth Camera

Typically, digital cameras yield pictures as a two-dimensional grid of pixels. There are three values, Red, Green, and Blue (RGB) affiliated with each pixel. The Intel RealSense D435 depth camera has both an RGB sensor and two other sensors. It creates a two-dimensional grid of pixels as well but adds an additional numerical value called depth (D), or the distance from the camera. Therefore, the pixels have four values associated with them, RGBD [27].

The D435 camera uses stereo depth to know how far objects are from it. Its two sensors are called the left and right imagers. The camera takes the images produced by each sensor and compares them. The comparison uses the distance between the right and left imager and the disparity between them to calculate the distance the object is away from the camera, similar to the way the LIDAR functions [13, 27].

The color sensor resolution is 1920 x 1080 at 30 frame per second (fps), with a 77° diagonal field of view. The depth resolution is 1280 x 720 at up to 90 fps, and a field of view of 95° ± 3°. The minimum depth distance is 0.105 m and the maximum range is 10 m. The high end of the range is approximate since it depends on lighting conditions [13]. The camera's data is integral to the lane detection and lane keeping technologies of the project, and it aids the LIDAR in object detection. The values discussed above, as well as the camera's weight and power consumption, are presented in Table 2 below. This sensor was tested by attaching it to the NVIDIA Jetson TX2 and making sure it was outputting properly.

Depth resolution	1280x720 @ up to 90 fps
Visible (RGB) resolution	1920x1080 @ 30 fps
Depth FOV	87°±3° x 58°±1° x 95°±3°
Visible FOV	64.4° x 42.5° x 77 ° (±3°)
Depth range	0.105 - 10m
Weight	72 g
Power	1.5 W

Table 2. Summary of Intel RealSense D435 specifications.

2.1.2.3 Tracking Camera

The second camera for the project is the Intel RealSense tracking camera and its importance stems from its ability to determine its location. The camera uses Visual Inertial Odometry Simultaneous Localization and Mapping (V-SLAM) to know where it is with respect to its environment. The camera has a visual processing unit to run its V-SLAM at low power. There are two imagers with fisheye lenses to give the camera a 163 ± 5° field of view [26].

An image processing unit (IMU) is included with the Intel RealSense tracking camera. The IMU detects movements and rotations in 3 axes and 6 degrees of freedom, respectively [26]. Through statistical methods, it uses the data gathered about the acceleration and velocity of the camera to estimate the trajectory of the camera over time. Prediction is essential to lane centering and lane detection. Similar to the LIDAR and the depth camera, the tracking camera was hooked up to the NVIDIA Jetson TX2 to check its output.

2.1.2.4 NVIDIA Jetson TX2

A lateral control system relies on three main types of hardware: devices that enable vast amounts of data acquisition, devices that can compile the data and decide actions based on the data, and devices that carry out actions. The device that brings in data from sensors and cameras, processes it, and dictates actions for the RC car to carry is called the microprocessor. For the application, a microprocessor is needed that is powerful, fast, reliable, and relatively compact. The microprocessor needs a central processing unit (CPU) and a graphics processing unit (GPU) capable of handling data from different sensors and artificial intelligence algorithms of varying levels of complexity.

The perfect processor to accomplish the previously listed tasks is the Nvidia Jetson TX2. The TX2 is a microprocessor made by the well-established AI hardware creator Nvidia. It is a powerful computer capable of handling complex deep neural networks with its NVIDIA Pascal architecture GPU and quadcore processing power, while having a relatively small power consumption [18]. Not only does it have powerful processing power, but it also has a relatively small footprint that could easily be integrated into an autonomous vehicle build. The TX2 boasts several input/output configurations making it both versatile and highly applicable to a fusion of

sensors. As depicted in Figure 3, the TX2 is the centerpiece of the project. It is the place where software and data, in turn spurring on navigation decisions in real time.

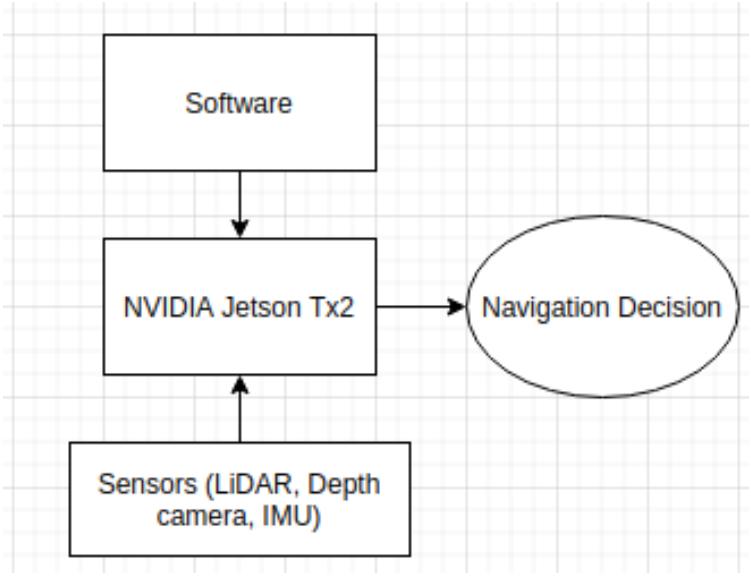

Figure 3. A flow chart showing the relationship between data from sensors and software driven by the TX2.

2.1.2.5 Flipsky ESC

When designing an autonomous vehicle on a 1:10 scale RC car, the car must move slowly. Without a vehicle capable of moving at slow speeds, testing would be impossible. Many adjustments to sensors and other sensitive pieces of technology would become not feasible. Even with higher torque low speed RC cars their acceleration is still too great to make testing a reasonable process. Therefore, it is critical that the autonomous car is equipped with a FLIPSKY ESC. The open source electronic speed control enables limits on motor voltage, in turn giving the motor less power. As seen in Figure 4, the FLIPSKY ESC sits in-between the motor and the processor, allowing the algorithm to take control of the RC car's steering, speed, braking, and acceleration. Since the FLIPSKY ESC allows for the customization of the motor's power, the car can be tested at the lowest speeds imaginable. The FLIPSKY ESC also collects performance data that could prove valuable when testing both software and hardware.

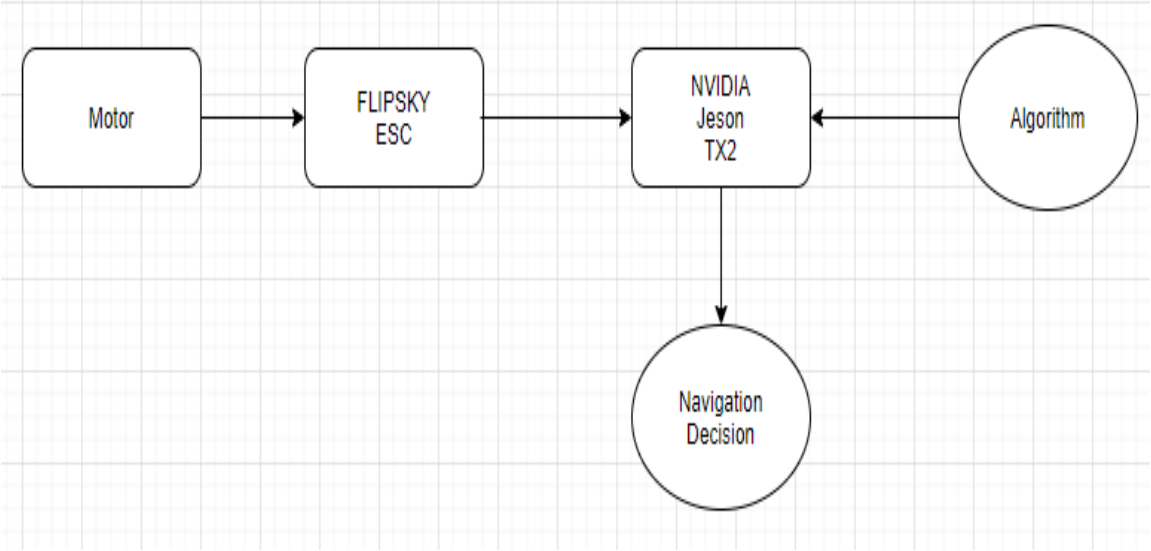

Figure 4. A flow chart showing the critical placement of the FLIPSKY ESC.

2.1.2.6 RC Car

The RC car plays an integral part in the success and overall usefulness of a lateral control system. The RC car is the hands and feet of the project. Without hands and feet, the sights humans see, the scents people smell, and the sounds people here cannot be acted upon or reacted to. Similarly, without an RC car the most advance lateral control system could do nothing with the data it receives from LIDAR and imaging units. Clearly, the optimum RC car to compliment an array of advanced sensors, processors, and controllers must be selected carefully.

A few very important things must be considered before choosing an RC car. The motor of the car must meet certain criteria, and the car's body must be of a shape that is conducive for modification. First off, the motor of the RC car must be a Brushed Motor. When looking at electric motors there are two types: brushed and brushless. Brushless motors tend to attract RC car hobbyists as they are capable of faster speeds and are low maintenance, but for the purposes of building an autonomous vehicle brushed motors are optimal. Brushed motors are easier to control with outside hardware and are capable of lower speeds. A slower RC car is more desirable for designing a fully autonomous lateral control system as testing and tweaking can only be carried out on a vehicle that is capable of lower speeds. The brushed motor must also have adjustable spur and pinion gear ratios to ensure a high torque and lower speed. Next the RC car must have a body shape that enables modifications and a custom chassis mount. RC car bodies come in all shapes and sizes, some of them have flat wide belly bodies while others have deep wheel wells. For the application, a vehicle is needed with a flat body style, as it accommodates modifications with ease. Mounting a custom chassis would prove to be incredibly difficult on a RC car with deep wheel wells or a non-flat body.

The car that meets these specifications closely is the Traxxas Slash. The Slash is a two-wheel drive short course race truck with a flat body, brushed motor, and has the option to switch out pinion and spur gears. Not only does the vehicle have optimum specifications as is, but other autonomous vehicle builds such as MIT's RACECAR [15] and Jetsonhack's project

of the same name [16] have significant documentation on the Traxxas Slash's ability to be converted to an autonomous vehicle with ease.

2.1.2.7 Power Consumption and Weight

When bringing together several pieces of advanced hardware and placing them onto an RC car, it is critical to keep track of two things: how much power a piece of hardware draws, and how much it weighs. Hardware such as the NVIDIA TX2 carry out several advanced computational functions at once; therefore, it draws more power than a less complex component. It is also important to pay attention to the weight of all the hardware being used, as too much weight could prevent the RC car from moving in an effective matter. Table 4 below lists both the weight and power consumption of each subunit previously listed.

Hardware Name	Weight (Pounds)	Power Consumption (Watts)
Nvidia TX2	1.030	15
LIDAR	0.42	2
Tracking Camera	0.16	1.5
Depth Camera	0.16	1.5
FLIPSKY ESC	0.18	7.5
Total	1.987	32

Table 4. Summary of hardware weight and power consumption specifications.

2.1.2.8 Power Supply

The hardware listed above cannot be powered by an RC car battery, as RC car batteries do not have standard USB or DC outputs. This poses a need for another mobile power source. For this, the Krisdonia 25000 miliampere hours (maH) portable power supply is used. This reliable and small power supply has several outputs that can supply wattages up to 94 watts, and only weighs 1.4 pounds [32]. This power supply allows all the hardware aboard the vehicle to be fully powered while remaining mobile.

2.1.2.9 CARLA simulation

Due to the COVID-19 pandemic the project had to pivot to a virtual simulation of the above hardware. The simulation environment chosen to do this is the CARLA simulator, an open-source simulator for autonomous driving research. CARLA has a wide variety of options and capabilities when designing an autonomous vehicle or an environment the autonomous vehicle is in. The user can control the sensor suite put on the vehicle, the vehicle's physical attributes, the town environment the vehicle gets spawned into, and the weather conditions the town experiences.

The simulator itself is run on a gaming engine and demands more powerful graphical computing than traditional personal computers have to offer. Fortunately, CARLA has a pre-compiled lite version that the team was able to download and use with enough capability to simulate the planned autonomous vehicle. CARLA uses a Python application programming interface (API) to easily incorporate Python scripting into the simulation. The Python API provided everything the team needed to create the hardware described above in the virtual environment. The ability to select similar sensors that were described above and the ease of

using the Python API to customize the simulation is what lead the team to choose CARLA for the simulation and virtual build.

2.1.3 Software

The programming language chosen for the project is Python. The design, however, uses the Robot Operating System (ROS) as a framework to help organize and facilitate the workspace and workflow of all the programming needed for the hardware of the design. Python is the best choice for the objectives of the project because it is one of the best languages used for machine learning and data analytics. Also, the syntax and readability of the language is simple and easy to use. Python has a vast selection of libraries and frameworks that can be used to aid in the algorithms for lane keeping, lane changing, and object detection. Lastly, Python has cross-platform capability with seamless integration. It is one of the most popular languages today, which is helpful if the project runs into any problems that require troubleshooting.

ROS is a great way to make programming complex robotic systems easy and simple, without sacrificing the functionality of the project. Without ROS, the file space and hierarchy of the files and components would have to be built from scratch specific to the design. With ROS, however, the user can run a couple commands that create the workspace for them and sets up their project to be ready to code quickly. It takes the “busy work” out of the beginning stages of programming. ROS also has many tools and libraries that are specific to robotic systems, so robotic applications, especially related to the design of the project, such as computer vision and data fusion become easier when using ROS. ROS also makes it simple to have different devices communicate with each other. Considering that the design has five different sensors and a whole other control system, there are many devices that need to communicate back and forth constantly and quickly throughout the operation of the project, and ROS makes it possible without any difficulty. ROS also has a thriving community with a helpful wiki website that can be used to guide the programmers in the code implementation portion of the project.

Python is compatible with the NVIDIA Jetson TX2, which is going to be used for computing the algorithms of the vehicle. Using Python with the TX2 is going to be the biggest challenge in the design because the TX2 must handle all the algorithm computation of the vehicle, which is more program intensive compared to motor control. However, there are plenty of libraries and frameworks for Python that are great tools for heavy data analysis/computation.

Python is also the programming language used in the CARLA API. This means that the integration of the algorithms originally designed in Python will be able to run in CARLA. The Python API gives the user access to all of the methods CARLA has to offer, which gives the user the ability to customize the simulation and extract data from the simulation easily. Running python scripts while the simulation is running is what will create the autonomous vehicle in CARLA, setup the environment, extract the necessary data, and run the data through the algorithms for full lateral autonomy. The implementation of the software on the physical hardware is shown in figure 5, and the implementation of software in the CARLA simulator is shown in figure 6. Notice the similarities between the two implementations.

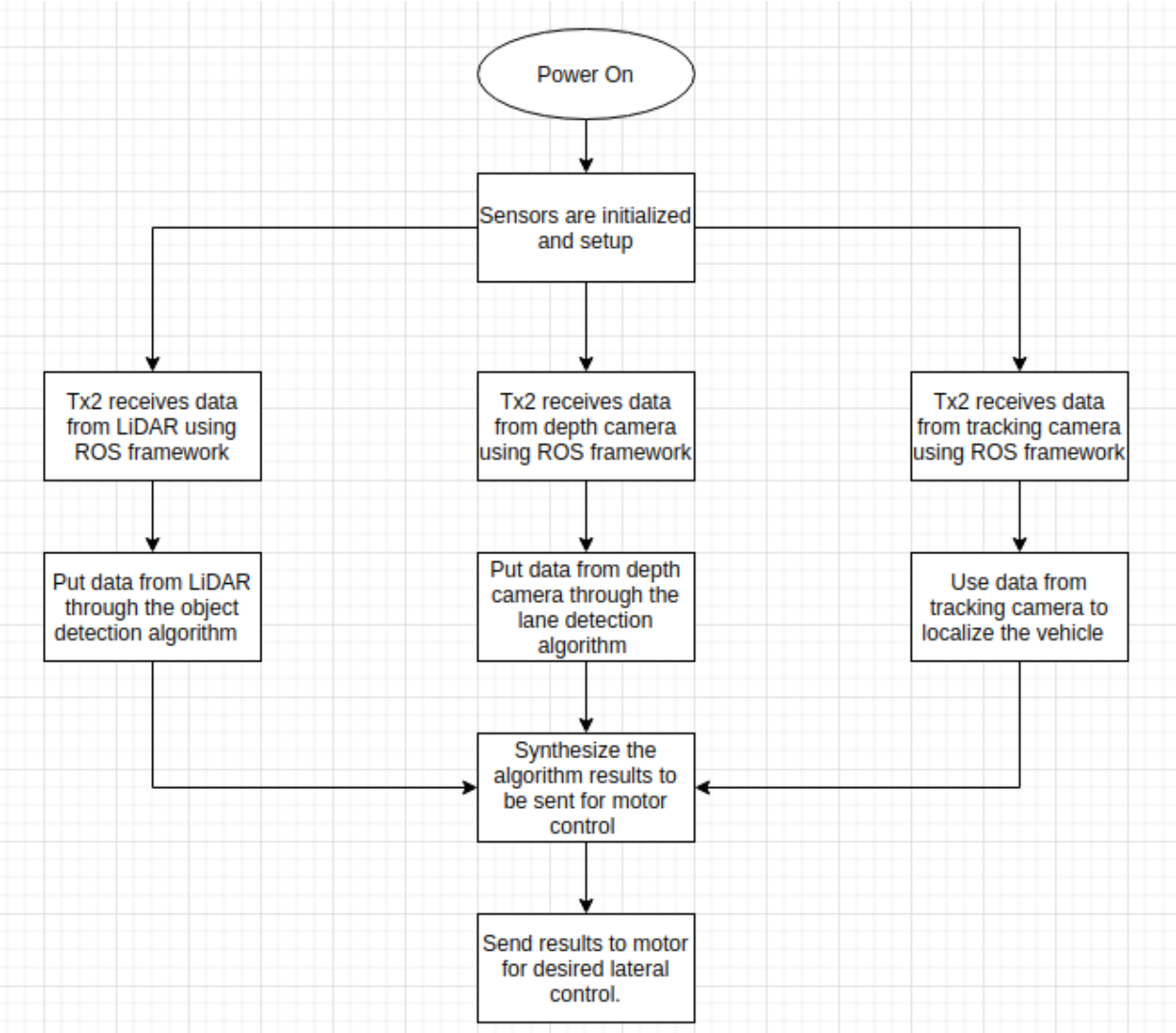

Figure 5. Software implementation for the physical hardware of the device.

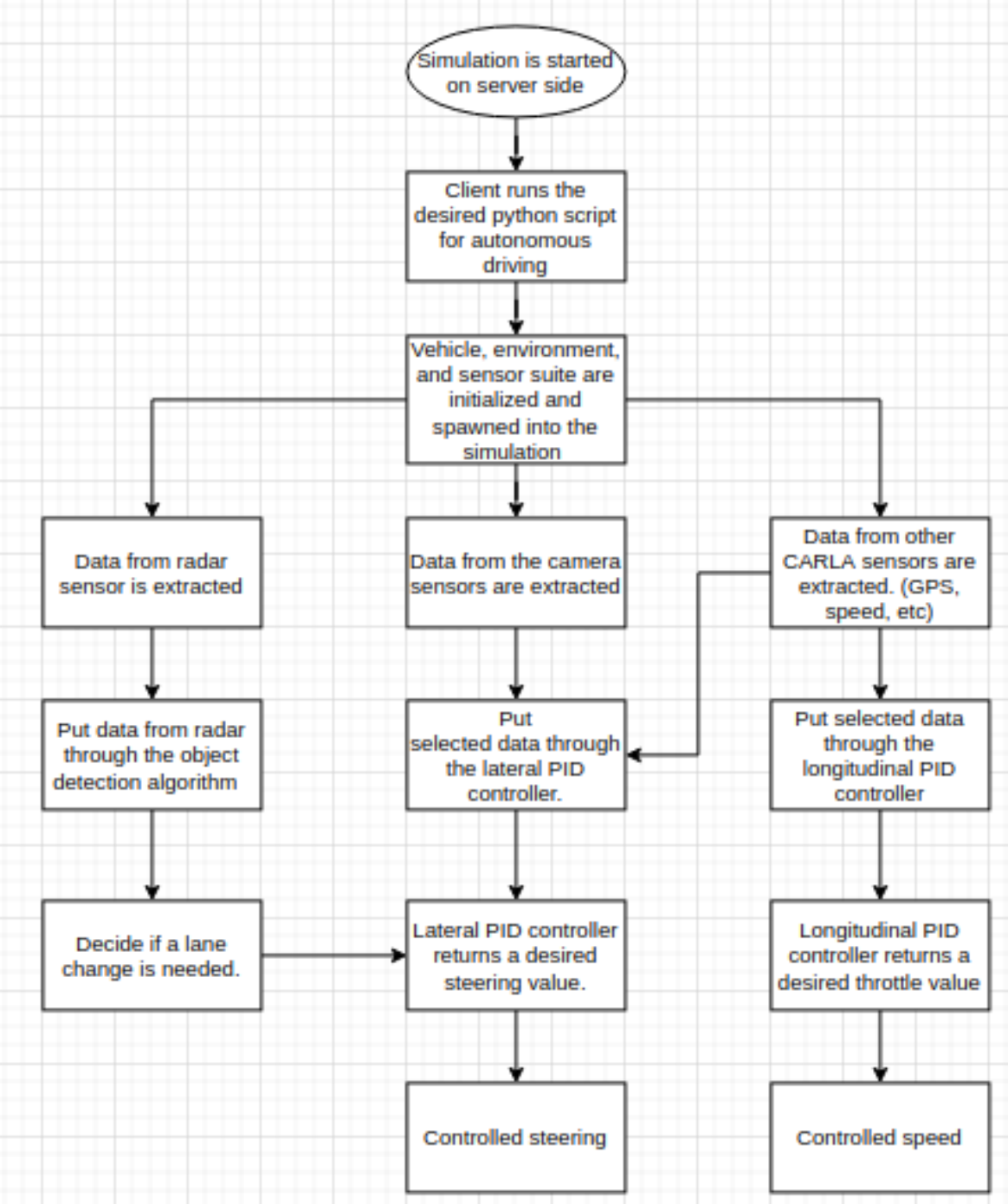

Figure 6: Software implementation for the CARLA simulation.

2.1.3.1 Algorithms

Selecting optimal sensors is a very important step in the project. The number, combination, and type of sensors were chosen to optimize functionality and reliability of the prototype. Another key step is then processing the data obtained from the sensors. Data from the depth camera is to be used for lane keeping and lane centering, data from the tracking camera is to be used for localization, data from the LIDAR is to be used for object detection, and

data from the sensors also is to be used for implementing longitudinal control. These algorithms must be reliable yet simple enough to meet the time constraints for the project.

The project is to use the Open Source Computer Vision Library (OpenCV) to achieve lane keeping and lane centering. The tasks to lane keep and lane center can be broken up into two main steps: line detection and control. The first step that needs to be taken is to identify and locate the lane markings. However, before the depth camera's images are used for line detection, the camera must be calibrated to account for lens distortion [21]. The camera is calibrated by taking a chessboard, or any other known object, to create a model that undistorts an image. Once distortion has been accounted for, the image needs to be transformed from the vehicle view to bird's eye view (BEV). The image is then converted into the HSV (Hue Saturation Lightness) color space to apply the Sobel operator [21]. The Sobel operator computes the gradient of the image to filter out the lane lines. A histogram is then taken of the bottom portion of the image; the peaks in the histogram identify the starting points of the lanes. To locate the entire lane lines, the sliding search algorithm is used. Starting from the initial points obtained from the histogram, a window slides up the lines to determine all the coordinates for the lane. These locations are then used to determine the best fit polynomial for the lane lines [21].

OpenCV has commands to then control the vehicle. The steering angle needed to maintain the center of the lane is computed. Furthermore, functions are to be used to stabilize the steering [28]. Stabilization is a critical aspect that ensures the RC car drives smoothly with no sharp turns; it is not desirable for the vehicle to oscillate within the lane lines.

Figure 6a illustrates the steps that are taken to lane keep and lane center, and Figure 6b illustrates the results obtained for lane line detection on the indoor track that was used for the project.

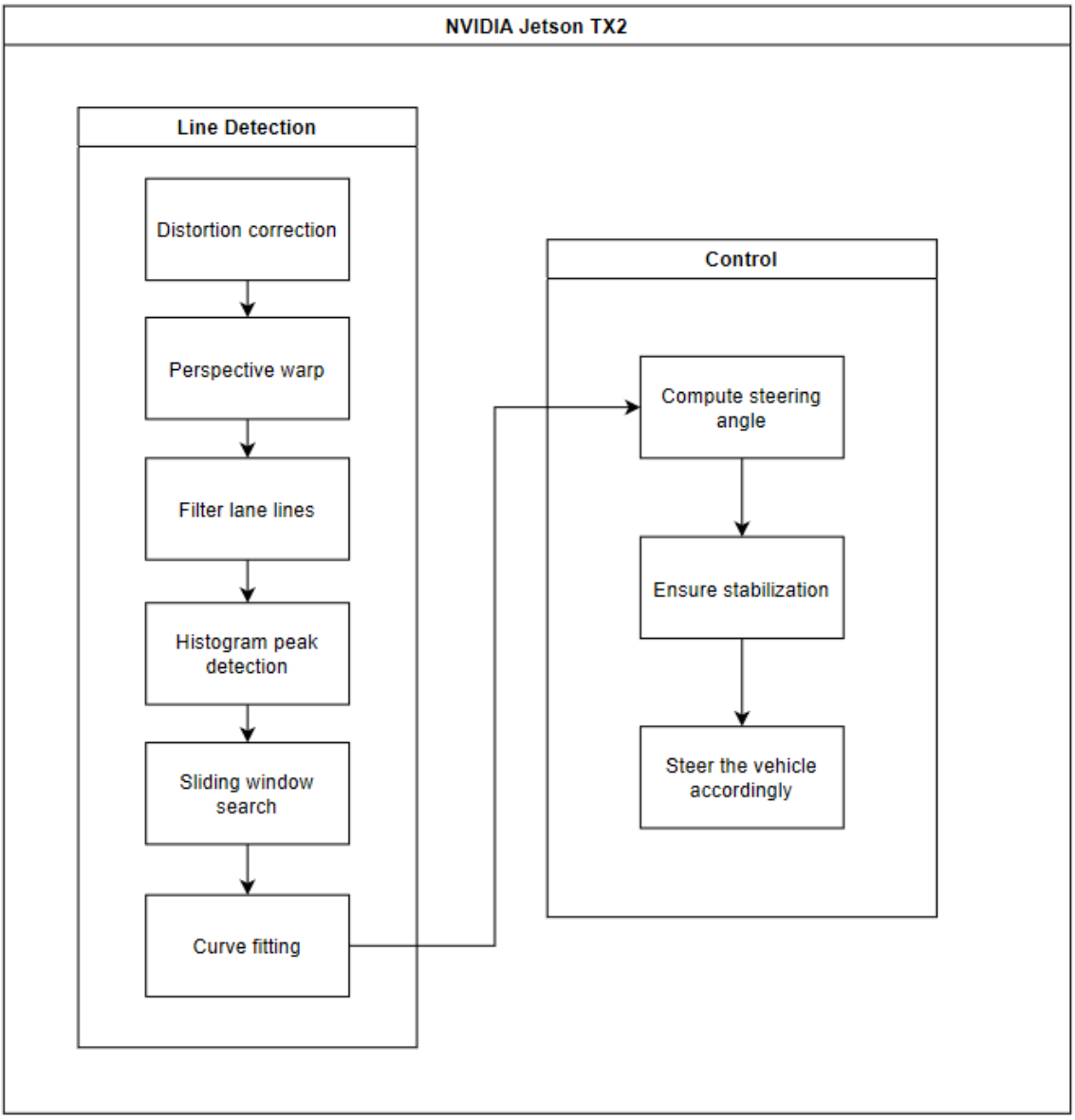

Figure 6a. The steps the autonomous vehicle needs to take to achieve lane centering.

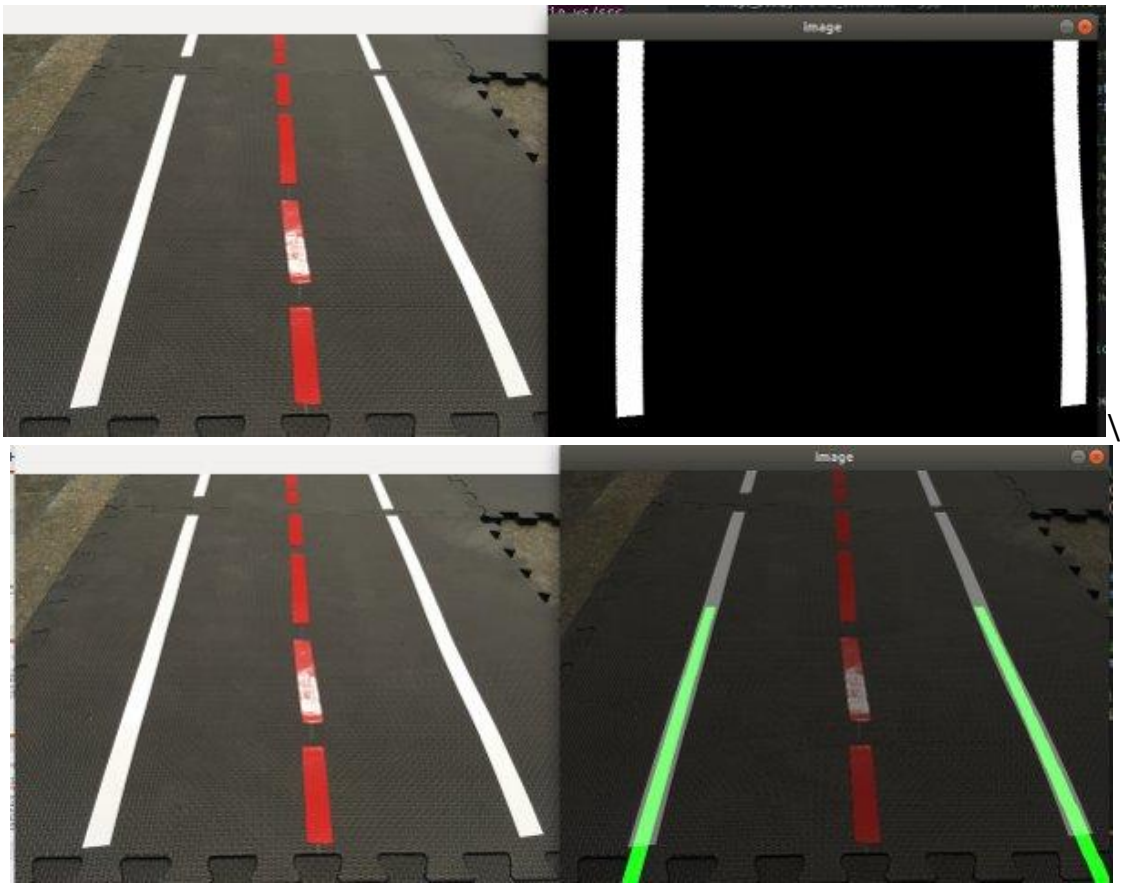

Figure 6b. The implementation of lane line detection for the project.

When the team successfully detected the lane lines, the project moved to the CARLA simulator. In CARLA, similar steps were implemented to achieve lane keeping for the virtual vehicle. First, the vehicle needed to be able to detect the lane lines. This was accomplished through the CARLA libraries; the vehicle was able to obtain the coordinates for the center of its current driving lane as an identifier of where it needs to drive. Next, a proportional derivative integral (PID) controller was implemented for the task of lane keeping. The PID lateral controller was fed with the error of the distance between the vehicle's center to the lane's center. The error determined the steering angle of the vehicle, which then led it to drive in the center of the lane.

The lateral controller was then extended in the simulator to allow for autonomous lane changing. In order to represent obstacles in the vehicle's path of travel, stationary vehicles were spawned in random locations within the lanes. When the vehicle detected a stationary vehicle close ahead, the error for the PID lateral controller became the distance between the vehicle's venter to the adjacent lane's center. This process permitted for autonomous lane changing within CARLA.

Another important algorithm for the project development is object detection. For the scope of the project, object detection is best achieved through an occupancy grid. An occupancy grid breaks up the environment into an array of independent cells [17]. Data obtain

from sensors are used to determine the probability that an independent cell within the environment is occupied. The map can then be used to determine if there are any objects within the vehicle's path of travel. Figure 7 below illustrates the concept of an occupancy grid [17].

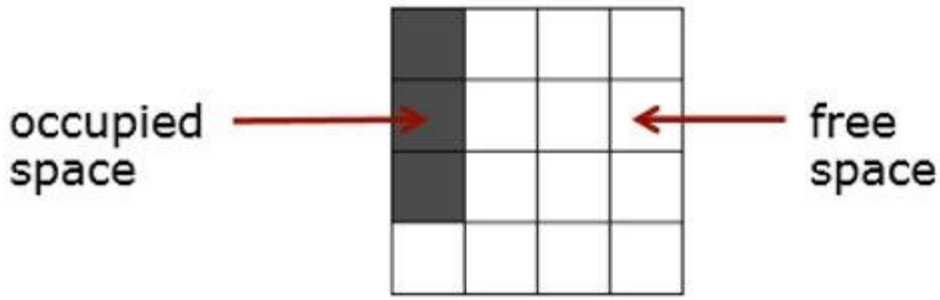

Figure 7. A visual representation for the concept of occupancy grid mapping.

The team generated the occupancy grid through ROS libraries and by using a depth camera and a LIDAR. The ROS libraries enabled data from the depth camera to be used to create a 3D map and allowed data from the LIDAR to create a 2D occupancy grid. Figure 8 depicts an example of an occupancy grid that was generated during the project. When generating this occupancy grid, the sensor configuration was on an elevated platform. While the depth camera captures the room in a 3D map, the LIDAR only sees a flat plane in front of it.

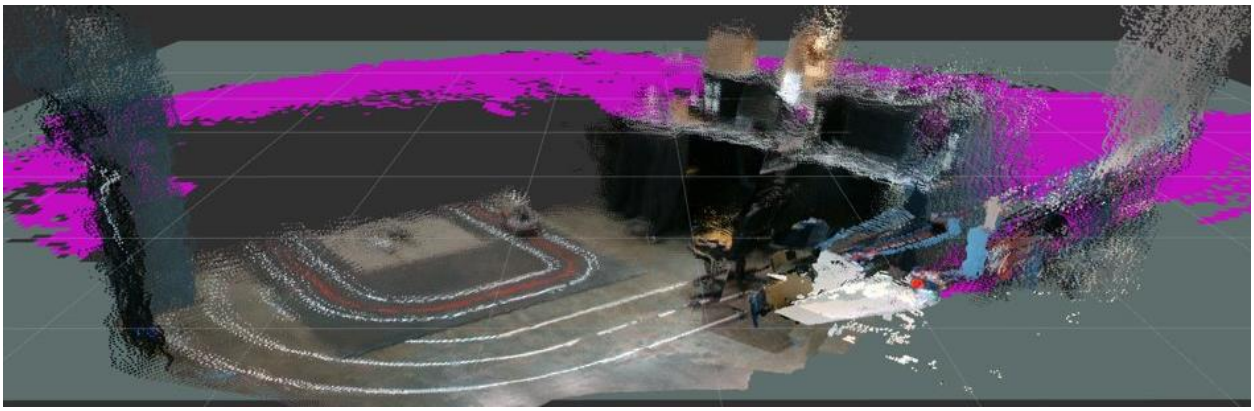

Figure 8. An occupancy grid that was generated with the team's hardware components.

Upon working in the CARLA simulator, the team did not use the occupancy grid for object detection. A sensor was mounted on the virtual vehicle that was able to detect any objects in its forward path of travel. The sensor was able to detect the distance between itself and the object ahead. The distance was used to determine when a lane change needed to occur.

The last important algorithm development needed for the project is longitudinal control. A simple PID controller was implemented to obtain constant speed in CARLA. The error

between the vehicle's current speed and the target speed was fed into the PID controller, which then set the throttle value for the vehicle.

2.2 Prototype

While the team did not have the opportunity to assemble the hardware components into a final prototype, the autonomous lateral control system was implemented in the CARLA simulator. The vehicle that the team programmed is spawned in a simple town on a road with two lanes in each direction. Sensors are also created and put on the virtual vehicle to represent the depth camera, tracking camera, and LIDAR that the RC car would have used. These sensors include a camera, an IMU sensor, and a radar. Additionally, three stationary vehicles are spawned on the same road to represent obstacles in the vehicle's path of travel. The programmed vehicle drives in its lane until it senses a nearby vehicle in its path ahead. Upon detection, the programmed vehicle then switches to either the right or left lane depending on the lane it is currently in. The vehicle remains in the new lane until another vehicle is detected ahead. The programmed vehicle operates with total autonomy.

A PID controller is used to enable autonomous longitudinal control. The error fed into the PID controller is the difference between the vehicle's current speed, which is obtained through CARLA methods, and the vehicle's target speed, which is defined by the user beforehand. If the current speed is above the target speed, the vehicle's throttle value decreases, and if the current speed is below the target speed, the vehicle's throttle value increases. The process always allows for the vehicle to drive at or close to the target speed.

A PID controller is also used to enable autonomous lateral control. The error fed into the PID controller is the difference between the vehicle's center and the current driving lane's center, both of which values are obtained through the predefined methods in CARLA. The PID controller is always optimized for minimal oscillation and minimal error. The PID lateral controller is also extended for autonomous lane changing. When the vehicle's sensor detects a nearby vehicle ahead, the error fed into the PID controller becomes the difference between the vehicle's center and the adjacent lane's center. If the vehicle is in the left lane, the right lane's center is obtained, and if the vehicle is in the right lane, the left lane's center is obtained. Like before, these values are obtained through predefined methods in CARLA. Since the error uses the adjacent lane's center, the vehicle successfully changes lanes with full autonomy to avoid the vehicle in its path. When the programmed vehicle has detected that it is in the new lane, the error fed into the PID controller again becomes the difference between the vehicle's center and the current driving lane's center. The process enables for the vehicle to continue driving in its current lane. The vehicle only changes lanes upon detection of an obstacle in its path.

Figure 9 demonstrates the successful operation of the programmed vehicle in CARLA. The vehicle begins in the left lane, and it approaches a stationary vehicle soon after it starts to move. Once the vehicle's sensor detects the stationary vehicle, it successfully changes into the right lane. The vehicle then continues in the right lane until it approaches another stationary

vehicle. The vehicle changes in the left lane after it has detected it and then remains in the left lane.

Figure 9. Screenshots of the CARLA simulator demonstrating operation of the vehicle.

Another representation of the vehicle's operation is shown in Figure 10. The red line is the vehicle's path, the three black dots represent the stationary vehicles that were spawned to put obstacles in the programmed vehicle's path, and the blue dashed lines are the lane lines. The plot demonstrates the programmed vehicle successfully kept its lane and changed lanes upon detection of any obstacles.

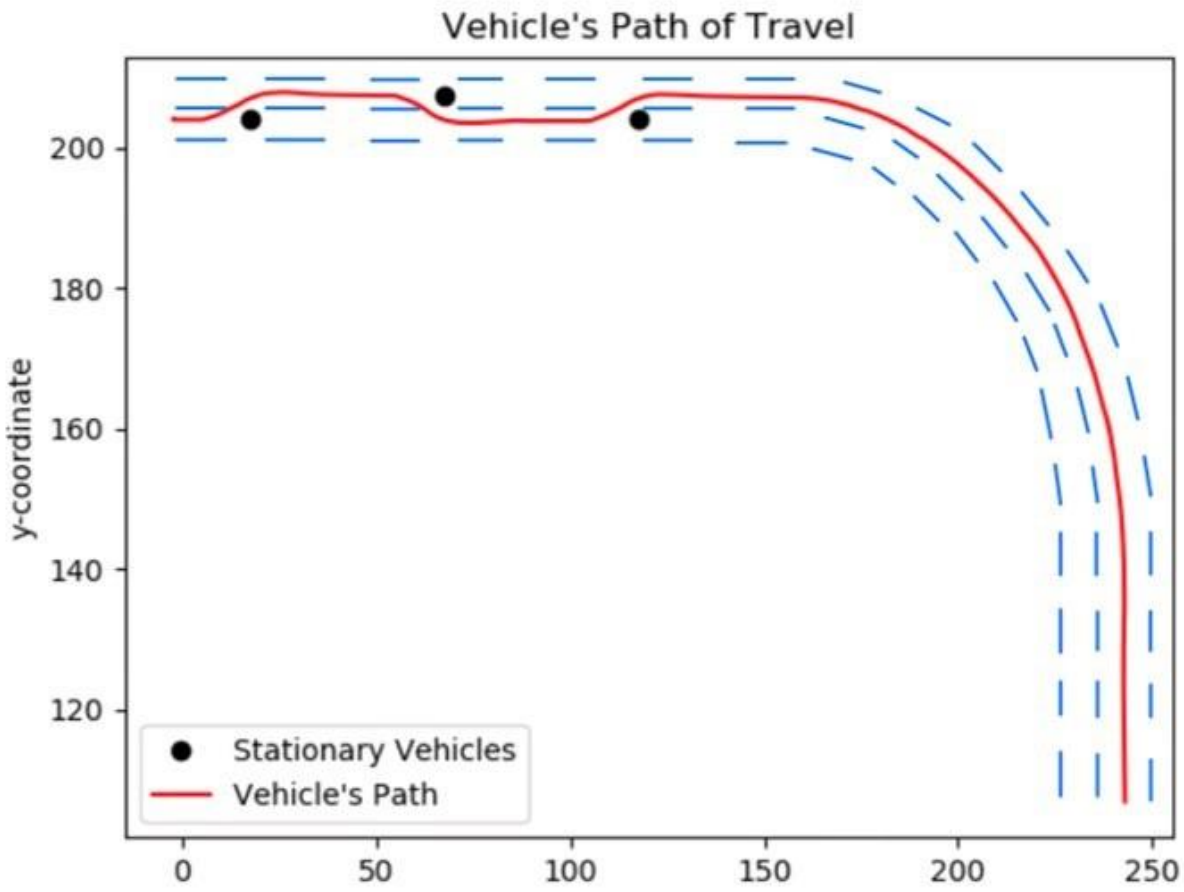

Figure 10. A plot representing the vehicle's path of travel.

After the team implemented autonomous longitudinal control in CARLA, they were able to program the vehicle to change lanes, but it was not with full autonomy. The vehicle only changed into the left or right lane when the a or d key was pressed, respectively. The client requested that additional vehicles are spawned on the same road as the programmed vehicle, and that the vehicle autonomously changes lanes as it approaches the stationary vehicles.

In order to satisfy the client's request, the team spawned three stationary vehicles, all in the left lane. The programmed vehicle drove in the left lane until it reached the stationary vehicles, and then it changed into the right lane. Because all stationary vehicles were in the left lane, though, only one successful autonomous lane change was demonstrated. The client then requested that the spawned stationary vehicles are in different lanes to demonstrate that more than one lane change can occur.

The team responded to the client and spawned three stationary vehicles on the same road as the programmed vehicle. This time, however, the first vehicle was in the left lane, the second vehicle in the right lane, and the third vehicle in the left lane. The programmed vehicle started in the left lane, and therefore, was able to demonstrate three successful lane changes to avoid the stationary vehicles.

The team used the CARLA simulation as a means for a virtual build for their originally planned design. CARLA's platform allowed the team to virtually build similar sensors to those that the initial design called for, and the methods of autonomous control implemented in CARLA can be smoothly transferred to those physical sensors. There are certain things that the virtual build will not be able to demonstrate with the actual hardware available in the project. For example, the vehicles in CARLA are full size vehicles, and they behave much different than the 1:10 scale car provided in the project. Also, the steering and throttle aspect in CARLA is simpler to implement than in the physical project. This is because their motoring is all digital and rely solely on the raw values inputted through code, whereas controlling the physical motors in the project would require additional hardware and more rigorous code.

The CARLA simulator allowed the team to build the depth camera sensor, an IMU sensor and a radar sensor. The depth camera is similar to the D435 depth camera used in physical hardware, the IMU sensor is similar to the D265 tracking camera, which was primarily used for its onboard IMU, and the radar sensor is similar to the LiDAR sensor used in the physical hardware. The data returned by the virtual sensors was similar to the data returned by the physical sensors, so the conclusion was drawn that the algorithms done in CARLA can be fully transferred over to the physical sensors. This means that both lateral and longitudinal PID controllers can be used in conjunction with the physical hardware mentioned in the paper.

There must be some things taken into consideration before this virtual build can fully be implemented onto the physical hardware. One thing is that the longitudinal controller is set to about 12 kph and would be extremely fast if implemented on a 1:10 scale car, so the speed should be slowed to a sustainable level for proper testing. Also, the lateral controller may need some adjustment for the new vehicle because its gain values were determined for the physical attributes of the vehicle used in the simulation, and the switch from virtual simulation to real world testing may have some unforeseen impacts on the PID controller performance. Finally, the algorithm for deciding lane changes may need to be adjusted for the same reasons as the lateral PID controller.

Overall, the virtual build was successful in accomplishing the goal of the project and it was designed for implementation on the physical hardware, which will help future research and development for physical project at NIU.

3 REALISTIC CONSTRAINTS

3.1 Engineering Standards

Many companies are currently researching and developing technology to further the advancement of autonomous vehicles. A key aspect of autonomous vehicles is the number, type, and combination of sensors used. Each industry has their unique take and combination, but the best practices are those that bring multiple sensors together [10]. Each sensor has different strengths and weaknesses; some work better in short range, some are less reliable in low visibility, etc. Therefore, bringing multiple sensor data together, referred to as

sensor fusion, is best to create a more reliable view of the surrounding environment [10]. While much fewer sensors are used in the project due to its small scale, it does follow the practice of using a combination of sensors. Two different sensors are used to, as explained above, increase the reliability of the prototype.

Commonly used sensors among the companies are cameras and LIDARs [10]. Cameras are commonplace, and as of 2018, all cars, whether autonomous or not, are required to have a camera for reverse [10]. Cameras are not optimal in all conditions; however, they play an important role in a vehicle's sensing capabilities, particularly regarding object detection and road marking identification. A 360-degree rotating LIDAR is the most popular sensor used in autonomous vehicles today. It is typically the most expensive of sensors yet only a few companies do not incorporate it, such as MobileEye [10]. Other sensors are used in autonomous vehicles, but the two sensors described here are most used [10]. Prior to working in the simulator, the team was working with and planned to use both the camera and LIDAR to optimize the sensing capabilities of the RC car.

3.2 Economic Constraints

The economic restraint on the project starts with the provided budget of $1000; an autonomous vehicle requires many hardware components in order to fully function automatically. For the project, the vehicle is an RC unit to be provided along with its control units, so the vehicle is not a factor in the budget. The hardware components for the project include a LIDAR, a camera, and a NVIDIA Jetson TX2. There are several options to choose from for each hardware component, but specific parts were chosen for the project that fit within the $1000 budget without sacrificing functionality.

3.3 Environmental Constraints

While the vehicle ended up being developed in the CARLA simulator, the RC vehicle was planned to be tested on an indoor track. Indoors, it cannot be fully tested according to how it would operate in a real-world outdoor environment. Therefore, the ability to smoothly translate the prototype to a full-size vehicle would have been limited. The vehicle cannot attain high speeds during testing. An indoor environment also does not account for potholes, skid marks, and other blemishes on a typical road that may disorient the lane-centering feature. Additionally, a constraint that the industry is currently facing is that lane-changing algorithms are either too conservative or too complex. A constraint on the project, therefore, is the lane-changing algorithm must be able to operate in an efficient, accountable way.

3.4 Sustainability Constraints

The main sustainability constraint is the limitations of the battery of the RC vehicle. The sensors all have low power consumptions, so these devices add minimal constraint to the battery. The main power consumption comes from controlling the vehicle. However, the purpose of the project is not to maximize the time the vehicle can operate but to make sure the vehicle can function long enough for testing and demonstration on a single charge. Another issue that comes up when dealing with the sustainability of the project is that the RC vehicle must be constructed in a way that it is easily maintainable for years to come. One of the goals

of the project is to create a platform for NIU students to research and develop autonomous vehicles. Thus, if a prototype had been built, it would have been crucial to design the vehicle in a way where any piece of hardware can be isolated, replaced or tested for future use. The constraint was also kept in mind when considering the software of the project. The software had to be implemented in the simplest, yet most functional way possible for future students to be able to grasp how the software functioned. Commenting within the program is very important for easy understanding of the code. All these constraints were considered when designing and planning the project.

3.5 Manufacturability Constraints

The project's focus was to develop a lane-changing algorithm, and it was designed to do so by building off, and only working with, one RC car. The prototype needed to function properly; however, another critical aspect was its manufacturability. It was expected to be built in a clear, straightforward manner so that it can be duplicated in the future. Since research is being conducted to further advancement in autonomous vehicle technology, a prototype must have been created such that it can be built upon easily and/or applied to a larger scale. Furthermore, a next step for NIU is researching connected vehicle technology. The prototype had to be simple enough to reproduce for further development and understanding of vehicle to vehicle communication.

3.6 Ethical Considerations and Constraints

Ethical constraints on the project mostly stem from the concern of safety for those operating the vehicle. In the scope of the project, no one is physically in vehicle, but the safety aspect is still crucially important because the goal is to have the system easily transferrable to a full-size car. Ethically, designing the project evolved around safe operation and control of the vehicle and the potential person in the vehicle, which means that the vehicle has a smooth, non-abrupt operation and makes decisions centered around the safety of the potential person operating the vehicle.

3.7 Health and Safety Constraints

The main health constraint the project faces would only become an issue if the solution were to be mass-produced for industry. If users were to trust a vehicle's total autonomy, especially in lateral movement, would it count as operating heavy machinery? Surgeon general warnings for prescription drugs and alcohol almost always include cautioning against operate heavy machinery while using any kind of substance. The constraint comes in to play in how a manufacturer ought to advise the general public. Automotive companies that utilize a fully autonomous lateral control algorithm would have to warn users about the potential danger of trusting a self-driving car's lateral control while unable to operate a vehicle. As for general safety, the vehicle must be designed in conformance with all component specs to ensure safe operation. The project's difficulty increases as more sensors are added, for they all must fit within the specs of the system.

3.8 Social Constraints

The concern for safety creates a social constraint to the project. Due to events that receive media coverage concerning autonomous vehicles, some people have developed a skepticism about the reliability and safety of autonomous vehicles. For example, there have been negative consumer reports about Tesla's autonomous cars as a result of the three fatal accidents when the automation features were in use. The stigma could affect the economic interest of the vehicle and the industry of autonomous vehicles for consumer use. Even though the project is not a fully autonomous vehicle, the purpose is to design it to be transferred to a vehicle that is, so it needs to be considered throughout the project.

3.9 Political Constraints

Political constraints could affect the research and development of fully autonomous lane changing for self-driving cars in the future. As the technology grows, it is possible for politicians to put limits on the degree of autonomy that can be deemed road safe. If these policies were to be put into place during any stage of the research and development of a lateral control algorithm, complications would arise. Additionally, states could differ in road safety policies regarding autonomous vehicles. These could complicate designs and force the development to include varying degrees of autonomy.

4 SAFETY ISSUES

The optimum design selected for the project does not have many safety concerns for both the designers and the users of the product. For the users of the product, the biggest safety concern is the vehicle running into objects that it is not supposed to, causing a risk of minor injuries to those around it. However, these issues would only arise in a malfunction of the vehicle because one of the main goals of the project that has been implemented is object avoidance. Another concern for the designers of the project was the vehicle running into objects causing minor injuries during the extensive period of testing before the prototype was finalized. However, because of the change in direction for the project in mid-March, the tasks of object avoidance and lane changing were implemented in the CARLA simulator. Therefore, the safety concern was for the virtual environment and the objects in the simulation that served as a representation of the physical world.

Additionally, while the prototype was never physically built, the designers discussed the safest possible spot on the vehicle to pick it up without causing damage to the vehicle or themselves and about implementing a kill switch to cut the power to the vehicle. The other possible safety concern for the designers of the vehicle was electrostatic discharge (ESD) of the microelectronic circuits onboard the vehicle. The design called for many microelectronic components and any microelectronic device has a potential risk of ESD. Although ESD does not pose a serious threat to the health of the designers, it can still generate enough energy to cause a little harm to humans and permanently damage the components of the vehicle. Thus, the proper ESD safety straps would have always been worn while wiring and testing the microelectronics onboard the vehicle.

One of the goals for the design was to have the autonomous RC vehicle be fully scalable to a full-size vehicle. With the goal in mind, it was important to design the RC vehicle with the

same safety risks of a full-size vehicle. The biggest issue was mimicking every aspect of a full-size car on a small-scale car. For example, the RC car chosen for the design was ten times smaller than the size of a full-size vehicle, so the car was extremely light weight compared to a real vehicle. With an electric motor, the light weight can make the car accelerate at unsafe rates for a full-size vehicle. There also must have been safety standards implemented by the designers that governed the safe operation of the autonomy of the vehicle. For example, factors such as when it is most practical to initiate a lane change or what the appropriate distance to be from an object is when stopped were considered. Designing the RC vehicle in a controlled manner mimicking that of a full-size vehicle, while implementing the safety standards of autonomy, were the main safety risks when designing the RC vehicle with a goal of full-size scalability. Although the team did not work on the RC vehicle upon the change in the project's direction, the same precautions were considered with the vehicle in the simulation.

The best way the safety concerns were addressed was through the software implementation of the vehicle. The software is what governs the operation and autonomy of the vehicle. Therefore, while designing and planning out the programming for the design, it was crucial for the designers to keep the safety concerns listed above in mind. Although the team ended up working in the CARLA simulation tool, it was recognized that in order to maintain steady control of the vehicle without rapid acceleration, the hardware associated with the drivetrain of the vehicle of the design must have been chosen specifically to have the capability of low speeds and acceleration while giving the designers more control of the motors via programming. To address the safety standards for autonomy of the vehicle, the designers extensively researched any safety standards that are already present in the autonomous vehicle industry. Furthermore, once the vehicle demonstrated some level of autonomy, it was not hard to adjust the already implemented operation of the vehicle. Addressing the safety concerns for the scalability of the vehicle was important to the success of the project. It was kept in mind that at full scale, real people would be in the vehicle. Therefore, safe reliable operation is of the upmost importance.

5 IMPACT OF ENGINEERING SOLUTIONS

Upon completion and successful implementation of the optimum design of the project, a low-cost solution for lane keeping/changing and object recognition for full size vehicle had to be presented. The solution would make for a safe, reliable, and effortless operation of a full-size car and completely change the way cars are thought about. The solution has global, economic, environmental, and societal impacts. The impact the solution has globally is that there are not many autonomous vehicles in the world, and most of the ones that are out there are still in the testing phase. The solution the project provides can also stretch across different platforms besides vehicles. Soon autonomy is to be introduced into everything a machine handles in the daily lives of every human, making the human experience better overall. Globally, there would be an increase in autonomy applications beyond cars, and the way engineers think about autonomy and control would be impacted.

The optimal design differs from other autonomous solutions because of its low-cost aspect. The leading factor that inhibits the growth and productivity of autonomous vehicles in the market is making an economical autonomous vehicle. Industrial LIDAR's are very expensive, and autonomy requires more sensors as well, which adds to the cost. By choosing a

careful array of sensors, the number of sensors used brings the overall cost of production down. With the cost getting lower, lower companies can then invest more into the research and production of autonomous vehicles, which would change the market for cars drastically. Also, more production of autonomous vehicles may lead to rise of more electric vehicles on the road, which drives the demand for gasoline down and the demand for electric charge stations up. Overall, the solution provided by the project has lasting changes in the market.

Less demand for gasoline leads to the impact the solution has on the environment. When gasoline is burned, such as in a traditional combustion engine, the byproducts emitted from the exhaust contain carbon dioxide. The extra carbon dioxide being emitted into the atmosphere is the leading cause of climate change and global warming. With the solution presented, there is more of an incentive for consumers to purchase autonomous vehicles, which takes gasoline cars off the road improving the emission of carbon dioxide.

Finally, the solution presented in the design has an impact on society. The way people think about cars changes; never have autonomous vehicles been readily available to the public at an affordable cost. With more autonomous vehicles on the road,
there are fewer vehicles driven by humans, so the probability of human error decreases. If every autonomous car follows the same autonomy protocol, then there may be no discrepancy when the cars are on the road, making the road safer overall. Overall, the solution presented here impacts the everyday lives of individuals who usually operate a vehicle. It permits people to do other tasks during their commute without worrying about the safety of themselves or others. As mentioned above, the solution of autonomy can also be applied to platforms outside of vehicles, so the impact of the solution on society almost seems endless. It is clear the solution presented in the optimum design may have lasting impacts on global, economic, environmental, and societal scales. The impacts may change the way the world operates for the better.

6 LIFE-LONG LEARNING

The senior design program is a transition from learning as a student to working as an engineer. It provides the opportunity to take learned material and apply it to a real project that is accomplished over the course of the year. The program teaches soft skills, technical skills, as well as the process that is taken for a design project in industry.

The program taught communication and listening skills. There were four main areas of focus for the project: hardware, sensors, software, and algorithms. The four team members of the project were each responsible for one of the areas listed. It was expected that everyone regularly informed the team of the research they had done and important information that needed to be considered for the project. Additionally, everyone needed to listen to all the ideas each team member presented to ensure all options were considered and the best choices were made for the prototype. While communication within the team was critical, regular communication with the faculty sponsor was just as important. Ideas and research had to be presented to the client minimally once a week to demonstrate the status of the project, as well as present any questions. Furthermore, regular communication with the faculty advisor was critical to ensure the team was on track and progressing in the right direction. The communication was again strengthened when the semester transitioned to a remote workspace.

Another opportunity the program presented was to learn a variety of technical skills. The programming language of choice for the project was Python, and majority of the team members did not have experience with the language. Throughout the course of the project, each team member became more familiar working with Python. Within the software, the project depended on the algorithms that are implemented to ensure the car operates with full autonomy. The team acquired understanding of lane keeping, object detection, and lane changing algorithms and the methodology that must be taken to implement such algorithms. The project also presented the opportunity to learn the hardware side. The team has learned the different types of sensors and controllers that were planned to be used to satisfy the project specifications.

Finally, the senior design program illustrated the process that is taken for a design project in industry. The project statement must first be defined to understand the problem that needs to be solved. Time must then be spent to fully comprehend the design requirements and the prototype specifications; a working solution must meet the operational and technical guidelines outlined by the client. Additionally, before a design is finalized, multiple designs and components must be considered and analyzed to ensure the optimal design is selected.

7 BUDGET AND TIMELINE

7.1 Budget

The project is limited to a $1000 budget. The budget includes hardware components, as well as any necessary supplies to complete the autonomous vehicle lateral control system. The budget for the project is shown below in Table 5.

Table 5. The Budget for the Autonomous Vehicle Lateral Control System.

Materials	**Cost**
NVIDIA Jetson TX2 dev kit	$299
Intel Realsense Depth and Tracking Bundle	$367.07
Slamtec RPLidar A2	$320
LANMU USB to DC Power Cable	$8.99
Total Cost:	**$995.06**

For an autonomous vehicle to be able to change lanes without user input, there must be great deal of thought put into the components needed. In order to have a scale vehicle have any degree of autonomy, let alone advanced lane changing capabilities, there are many items that must be acquired. The first requirement is a 1:10 scale vehicle to serve as the chassis for additional hardware. The hardware includes microprocessors, image detection units, a LIDAR unit, and other key pieces of technology that make the project possible.

First, a piece of hardware that is necessary for the completion of a fully autonomous lateral control system for a self-driving vehicle is a microprocessor. It is a key piece of hardware that serves as both the heart and brain of an autonomous vehicle. Without it, information could not be collected or synthesized, nor could a decision algorithm be implemented to use said information to make an intelligent decision. Not only does the microprocessor need to be compact enough to fit on a small 1:10 scale vehicle, but it also

must be powerful. No ordinary microprocessor would suffice, as the amount of computing power it takes to gather in live data from a fusion of sensors and implement AI algorithms to apply this data is fast. That is why an NVIDIA Jetson TX2 is the best choice for an autonomous vehicle build, as it can handle artificial intelligence applications and robust data acquisition, which can be clearly seen in [29].

Second, imaging units are an absolute necessity for any artificially intelligent vehicle. They act as the frontal vision of an autonomous vehicle. When it comes to changing lanes with full autonomy in a safe and effective manner or knowing where the RC car is in respect to its lane, cameras could not be more critical. Whether it be detecting objects or seeing obstacles in an adjacent lane, the two cameras act as one of the pillars that hold the project. Therefore, the best and necessary choices are the Intel RealSense Depth Camera D435 and the Tracking Camera T265. The cameras pair well together, as the V-SLAM of the T265 can contextualize the point cloud created by D435 [26]. The tracking camera also boasts an on-board inertial measurement unit which helps with an autonomous vehicle's speed control.

Third, a LIDAR sensor is necessary for an autonomous vehicle as the sensor acts as both forward and peripheral/rear vision. A LIDAR unit is a sensor that uses laser light to gather data on objects within a certain radius 360 degrees around the sensor itself, which can be seen in [30]. A LIDAR is needed to aid in detection where the imaging unit cannot see. It becomes critical when considering lane changing applications, as peripheral vision is necessary to make a safe and effective lane change. Therefore, it is critical for a lateral control system that a LIDAR be mounted onto the chassis of the autonomous vehicle.

Another budget item that must be purchased is a USB to DC input converter. The USB to DC input converter enables the power supply to connect to and power the USB Hub. In conclusion, every budget item above is crucial to the completion of a small-scale autonomous vehicle with the ability to change lanes without user input.

7.2 Timeline

In the beginning stages of the project, the team met with the client to understand the problem and the need for a prototype. The team members each then spent several hours familiarizing themselves with autonomous vehicles to understand their background and their current state in the market. Given the client's needs and the information obtained from research, operational and technical specifications were developed for the project.

After the project requirements were established, research was done to understand possible design solutions. Different RC cars, sensors, processors, and microcontrollers were considered. Analysis was done on each possible design to weigh the advantages and disadvantages of using different combinations of hardware components. Furthermore, discussion with the client helped the team to solidify the best design solution, which was then analyzed to ensure satisfaction of all project specifications.

Selecting the optimal design included selecting the type of hardware components that are to be used. Since a LIDAR, two cameras, and a processor were chosen along with the RC car, significant research was done to select specific parts to order.

Throughout the initial design process, research was also being done on the software and algorithms needed for the project. As the team had decided to use ROS and Python, the team

spent December learning the framework and language. It was a critical step that helped the team design the prototype. A strong foundation eases the transition into program implementation. Regarding program planning, research was done on object detection and lane line detection. The team continued to research throughout December to understand localization, decision making, and control modules.

January through April was the team's opportunity to build the prototype. January was dedicated to the initial building and testing. The team planned the chassis layout, and then the first prototype was going to be assembled once all hardware components had been received. Additionally, each of the sensors was tested to ensure they were calibrated and working properly.

The bulk of the work was completed February through March. The team started to implement the algorithms during February. Object detection with the LIDAR and the depth camera were first broken into two separate tasks. The team needed to ensure that everything functioned as expected before they combined the sensors to work together in detecting objects. Localization with the tracking camera was simultaneously being worked on during the stage, as well as lane keeping with the RGB camera. Several hours of testing was completed during the initial program implementation to make sure the prototype development was on track. Due to the sequence of events halfway through the semester, however, the initial build of the prototype was never completed after these sensor tasks. The RC car had not come in yet, and thus, the prototype build never occurred before the project changed directions.

With a new direction in mid-March, the team began to extensively work in Carla simulator to implement autonomous longitudinal and lateral control. The first tasks were implementing a simple longitudinal controller to obtain constant speed and implementing a controller for lane keeping. Then the controller for lane keeping was expanded to enable lane changing upon detection of an object in the vehicle's path of travel. A significant amount of time was dedicated to algorithm writing and testing. Final prototype testing occured in April, and the team was ready to present their final prototype on Friday, May 1, 2020.

The timeline for the project is outlined below in Figure 11. It includes an estimation of the hours worked to complete each task.

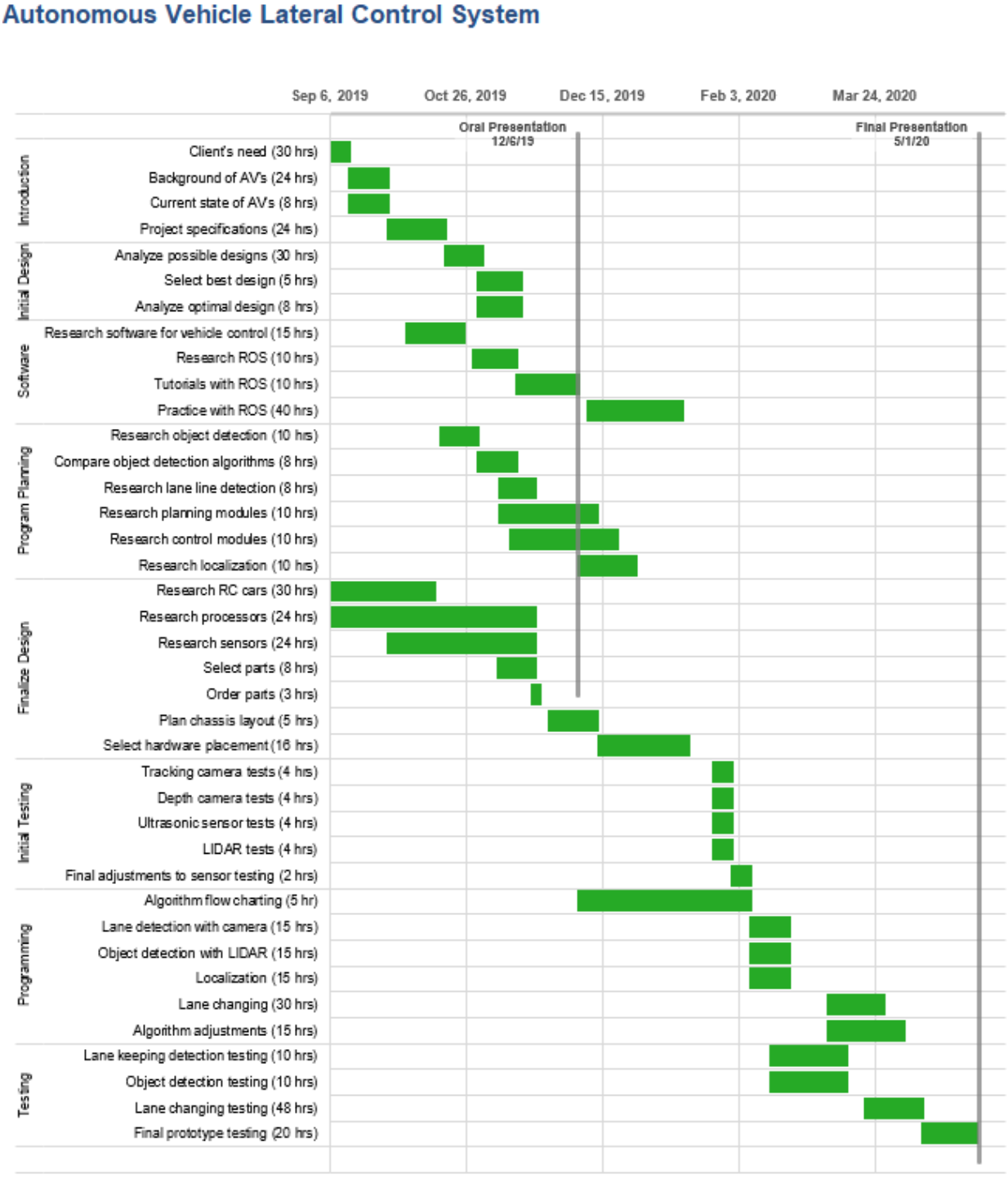

Figure 11. The project schedule with the hours for each task.

8 TEAM MEMBERS CONTRIBUTIONS TO THE PROJECT

8.1 Team Member 1

At the beginning of the project, Dylan did extensive research into how an autonomous vehicle works. He investigated autonomous cars that are available today and what technology they currently use to achieve autonomy. He then further researched that technology in terms of sensors used on the autonomous vehicle and the software achieving that autonomy. Once a firm background of autonomous vehicles was established, Dylan then specified his research to the project on hand. He looked into small scale autonomous vehicles and the hardware and software associated with them. He also researched patents that would gain him a further understanding of the software workflow in making a vehicle autonomous. After Dylan felt confident in the project at hand, he then started to research and begin hands on work with the software portion of the project, which is his part of the project.

Upon researching the plethora of software options that could be used with the project, Dylan had to make some decisions regarding which programming language to use to program the vehicle and which framework to use with that language in order to make programming the vehicle smart and robust. There were many things Dylan had to take into consideration before making these decisions. He had to make sure the programming language had machine learning capabilities, the language is extremely efficient so there is no lag time for computing intense algorithms, the language is compatible with the hardware chosen for the project, the language is well documented in case there is an issue down the line with the project, and the language was simple enough for the team to learn and understand quickly in order to commence the project as soon as possible. Keeping each aspect in mind, Dylan decided that the best language to use was Python with the PyTorch framework for machine learning, along with the Robot Operating System (ROS) workspace for quick speed and seamless integration.

Once Dylan decided on the software for the project, he started to do some more extensive research into those programming languages and frameworks to better understand their implementation and functionality. It led Dylan to begin various beginner tutorials on the ROS main website and PyTorch's Main website, for these were the frameworks he was less familiar with. The goal for Dylan is to hopefully gain a firm foundation and understanding with every aspect of the software being used within the project to a point where he feels comfortable enough to code and design the software functionality of the project. So far everything is on track for Dylan and the software of the project should propel the project to be successful with its objectives.

When the new semester began there was a delay in receiving the hardware ordered for the project, so with the permission of faculty and some graduate students, Dylan was able to start testing some software on the Jetson Nano and RPLiDAR A1 that were available in the autonomous vehicle laboratory. This allowed Dylan to get familiar with the ROS architecture and how to exactly work with these sensors in ROS. Dylan was able to learn how the Jetson products worked with ROS and the sensors and figured out how to properly build and install the source code for each sensor. These first few weeks of the semester prepared Dylan for when the ordered sensors finally arrived.

Once the sensors arrived, the first task was to initialize the Tx2 the same way Dylan did the Jetson Nano. Once all the sensors and Tx2 board were configured properly, Dylan started working on fusing the sensor data together and generate a combined occupancy grid with localization. The occupancy grid was successfully generated and showed how the sensors data can be properly fused together and handled by the powerful Tx2. The project was starting to rely on the 1:10 scale vehicle that was yet to be ordered, but then COVID-19 changed the course of the project.

After the logistics of the new direction of the project were sorted out, Dylan and the team started researching into CARLA and how to use it as a virtual build for the planned prototype. After weeks of learning the basics of CARLA Dylan started working on the longitudinal PID controller. He was able to use the CARLA Python API to extract the data he needed to control the speed of the car. When the vehicle had longitudinal and lateral autonomy, Dylan worked on spawning obstacles for the autonomous vehicle to recognize and change lanes to avoid. After that was achieved, he started working on the algorithm for lane changing using the radar sensor. After rigorous testing of the vehicle and adjustments made to the algorithm, the final vehicle was complete with total lateral and longitudinal autonomy.

8.2 Team Member 2

At the beginning of the project, Nora familiarized herself with the concept of autonomous vehicles. She researched their background, their current state in the market, as well as the five levels of autonomy for vehicles. With a base level of understanding, she then began to research aspects and systems that are needed to develop an autonomous vehicle. Nora learned the need for perception, localization, decision making, and control modules.

Within perception, Nora dove into object detection. Given that the project uses a LIDAR and a depth camera for object detection, she researched image processing algorithms that can be used to locate stationary objects. Considering the advantages and disadvantages of many different processes, she learned that creating an occupancy grid with the LIDAR was the best option given the scope of the project, and thus, researched it more in depth.

Within planning, Nora researched lane line detection. The project specifications outline the vehicle must be able to detect any lane line, such as straight, curved, and dashed lines. Therefore, she studied how OpenCV (Open Source Computer Vision) and Python can be used to find the best fit polynomial that identifies the lane lines.

When the hardware components were received in January, Nora dug into implementing lane keeping. She was successfully able to detect both straight and curved lane lines through ROS and OpenCV. However, after implementing lane keeping, the direction of the project changed to a simulation in CARLA, a vehicle simulator.

As soon as the project changed, Nora familiarized herself with CARLA and the new goal of the project. After the team was able to spawn a vehicle in the simulator, Nora began to implement a lateral PID controller for lane keeping. After she finished lane keeping, she then expanded it to lane changing. Furthermore, Nora worked on plotting the desired versus actual paths of the vehicle to demonstrate the outcome of the project.

8.3 Team Member 3

At the onset of the project, Jake researched the limitations and capabilities of the RC car that is serving as the base of the project. These limitations and capabilities include, but are not limited to, weight, spring modification, motor modification, turning power, and speed control. Jake also selected the necessary hardware that must serve as a foundation for the autonomous vehicle. The pieces of hardware include the FLIPSKY ESC and the TX2.

Jake recognized the stock RC car the project is being built on operates at far too high of speeds to be made into an artificially intelligent vehicle. He researched and found that an open source electronic speed control, called the FLIPSKY ESC, would not only be able to give total control over the speed of the vehicle; it would also allow control over output voltages and power dissipation. Jake also selected the NVIDIA TX2 as the processor. The process required continual research on his part, as there were three options to choose from. In the end, Jake was able to choose the TX2 based on its powerful artificial intelligence handling, its convenient size, and incredibly fast computational speed.

Jake also worked on familiarizing himself with basic autonomous vehicle concepts, such as lane keeping, object detection, localization. He also spent time researching artificial intelligence, machine learning, python, and deep learning. Lastly, Jake also aided other group members in sensor selection and planned the layout of the chassis on the RC car.

Next, Jake spent time learning the installation process and usage of the NVIDIA TX2's operating system Ubuntu 18.4. He familiarized himself with this operating system's command terminal, as this would serve as the communication point for the rest of the hardware. Once the LIDAR, tracking camera, and depth camera came in Jake familiarized himself with all these components by generating their outputs via the command line. Once this was achieved, he researched and became proficient in both GitHub and ROS. Jake learned how to share and access scripts from GitHub and learned about how to use ROS to bridge the gap between the sensors and processor.

Once Jake became proficient with ROS, he spent a much of his time helping the generation of outputs and gathering of data from the LIDAR, tracking camera, and depth camera using different ROS packages. Most of his time was spent mastering the tracking camera in ROS. Jake discovered not only how to retrieve data from this camera in ROS, but also discovered how to overcome errors the camera was giving. Once the project moved from a physical to the virtual environment CARLA, Jake spent most of his time understanding this open source program. He was able to load CARLA, learned how to manipulate the environment CARLA generated, and was able to test python scripts pulled from GitHub. Finally, Jake assisted in the checking and testing of the lateral and longitudinal control scripts.

8.4 Team Member 4

Mercer began by researching autonomous vehicles in order to gain an understanding of what is currently on the market and what had been done previously. Once she had a grasp on the big picture of the project, she started to research the details. Her area of focus was the different types of sensors. Comparing and contrasting the leading manufacturers of autonomous vehicles and their choice in sensors, Mercer found that LIDARS and cameras are helpful to achieving autonomy in vehicle designs.

Next, she investigated the different models and types of each of those three sensors. Throughout the process, Mercer consulted her team members and the faculty advisor to make sure the decisions she was making were the best given the information acquired. With their input, paired with the advantages and disadvantages of the sensors, she chose one LIDAR and two cameras to use. A LIDAR is important to the project as it acts as the eyes of the car. It has 360° angular range and outputs a two-dimensional point cloud of data used in object detection. The depth camera and the tracking camera are essential to lane detection and localization. She found the sensors for the lowest price and added them to the parts order form.

Mercer used some time to get up to speed on python coding by doing many tutorials. She added the operating system Ubuntu to her computer and learned how to work with Linux and code in her terminal. She gathered information about ROS and how to push and pull code on Github.

Once the sensors arrived, Mercer started experimenting with the sensors to become familiar with how they worked. She studied ROS packages and tested how different ones work with the sensors. Mercer juxtaposed these packages with the goal to find the best ROS packages for creating an occupancy grid used in object detection. It was determined to use the package rtabmap for generating a 3D map with the D435 camera, and the package hectorslam for building the 2D occupancy grid.

9 CONCLUSION

Creating a lateral control system that allows a self-driving car to change lanes safely and with full autonomy was the project's principal motivation. The project used virtual sensors fixed on a car conjointly with a lane changing algorithm in order to achieve the purpose. Since the algorithms in the autonomous car industry are either too complex or too conservative, the algorithm in the project must have been simple enough to achieve the goal safely and effectively. Additionally, the distinctiveness of the project originated from its cost-effectiveness and its ability to be tested on a college campus. The Autonomous Vehicle Lateral Control System can be implemented applied to a 1:10 scale RC car.

The sensors that are simulated in the project were a camera, an IMU sensor, and a radar. They stood in for the depth camera, tracking camera, and LIDAR, respectively. The LIDAR and virtual radar are at the helm of object detection. Cooperatively, the cameras are responsible for acquiring the information that is used to perform the lane centering algorithm. Both the lane centering and lane changing algorithms lead to the RC car operating autonomously.

10 REFERENCES

[1] Bavar, B., Batel, E., Zhang, X., Nix, M., Sweeney, M., Nagy, B, *US Patent No. 10,438,493 B2,* Washington, DC: U.S. Patent and Trademark Office, 2019, (pdfpiw.uspto.gov/.piw?PageNum=0&docid=10438493&IDKey=941143BF0C91%0D%0A&HomeUrl=http%3A%2F%2Fpatft.uspto.gov%2Fnetacgi%2Fnph-Parser%3FSect1%3DPTO2%2526Sect2%3DHITOFF%2526p%3D1%2526u%3D%25252Fnetahtml%25252FPTO%25252Fsearch-bool.html%2526r%3D8%2526f%3DG%2526l%3D50%2526co1%3DAND%2526d%3DPTXT%2526s1%3D%252522Autonomous%252BVehicles%252522%2526OS%3D%252522Autonomous%252BVehicles%252522%2526RS%3D%252522Autonomous%252BVehicles%252522)

[2] Ferguson, D.I., Zhu, J., *US Patent No. 8,825,265.* Washington, DC: U.S. Patent and Trademark Office, 2014, (pdfpiw.uspto.gov/.piw?PageNum=0&docid=08825265&IDKey=5A3A9441D9D8%0D%0A&HomeUrl=http%3A%2F%2Fpatft.uspto.gov%2Fnetacgi%2Fnph-Parser%3FSect2%3DPTO1%2526Sect2%3DHITOFF%2526p%3D1%2526u%3D%25252Fnetahtml%25252FPTO%25252Fsearch-bool.html%2526r%3D1%2526f%3DG%2526l%3D50%2526d%3DPALL%2526S1%3D8825265.PN.%2526OS%3DPN%2F8825265%2526RS%3DPN%2F8825265)

[3] Howard, B., "How does lane departure warning work?" *ExtremeTech*, 27 February 2017, (www.extremetech.com/extreme/165320-what-is-lane-departure-warning-and-how-does-it-work)

[4] Construciton Equipment, "Amazon Patents Autonomous Vehicle Lane-Changing Technology," *Construction Equipment*, 18 Jan 2017, (www.constructionequipment.com/amazon-patents-autonomous-vehicle-lane-changing-technology)

[5] Tesla, "Autopilot and Full Self-Driving Capability," *Tesla,* 2019, (www.tesla.com/support/autopilot)

[6] Curlander, J.C., Russell, R.S., Bathurst, A.S., Madan, U., Graybill, J.C., Norwood, J.B., Lauka, W.S., Mishra, P.K., Canavor, D.E., *US Patent No. 9,547,986.* Washington, DC: U.S. Patent and Trademark Office, 2017, (pdfpiw.uspto.gov/.piw?PageNum=0&docid=09547986&IDKey=8303676DC9C8&HomeUrl=http%3A%2F%2Fpatft.uspto.gov%2Fnetacgi%2Fnph-Parser%3FSect2%3DPTO1%2526Sect2%3DHITOFF%2526p%3D1%2526u%3D%2Fnetahtml%2FPTO%2Fsearch-bool.html%2526r%3D1%2526f%3DG%2526l%3D50%252)

[7] Kaslikowski, A., "Everything you need to know about autonomous vehicles," *Digital Trends,* 30 June 2019, (www.digitaltrends.com/cars/the-current-state-of-autonomous-vehicles/)

[8] Massachusetts Institute of Technology, "Driverless cars change lanes more like humans do: Algorithm computes 'buffer zones' around autonomous vehicles and reassess them on the

fly," *ScienceDaily*, 22 May
2019, (www.sciencedaily.com/releases/2018/05/180522114829.htm)

[9] "The Grand Challenge," *Defense Advanced Research Projects Agency, (*www.darpa.mil/about-us/timeline/-grand-challenge-for-autonomous-vehicles)

[10] Dawkins, T., "Autonomous Cars 101: What Sensors Are Used in Autonomous Vehicles?" *Level Five Supplies*, 14 Jan 2019, (https://levelfivesupplies.com/sensors-used-in-autonomous-vehicles/)

[11] Wu, E., "All You Need To Know About Slamtec RPLIDAR , Mapper, and Slamware," *Seeed Studio Blog*, 5 Aug. 2019, (https://www.seeedstudio.com/blog/2019/08/05/all-you-need-to-know-about-slamtec-rpLIDAR-mapper-and-slamware/)

[12] "RPLiDAR A2M8 360 Degree Laser Scanner Kit - 12M Range," *Seeed Studio Blog*, (https://www.seeedstudio.com/RPLIDAR-A2M8-360-Degree-Laser-Scanner-Kit-12M-Range-p-3074.html)

[13] "Depth Camera D435," *Intel*, (https://www.intelrealsense.com/depth-camera-d435/)

[14] "CARLA," *Carla*, (https://carla.org/)

[15] "RACECAR, A Powerful Platform for Robotics Research and Teaching," *MIT,* (https://mit-racecar.github.io/)

[16] "Jeston Hacks, Developing for Nvidia Jetson: RACECAR/J," *Jetson Hacks,* (https://www.jetsonhacks.com/racecar-j/)

[17] Stachniss, Cyrill. "Robot Mapping." Autonomous Intelligent Systems, http://ais.informatik.uni-freiburg.de/teaching/ws12/mapping/pdf/slam11-gridmaps-4.pdf

[18] "Jetson TX2: High Performance AI at the Edge," *NVIDIA, (*https://www.nvidia.com/en-us/autonomous-machines/embedded-systems/jetson-TX2/)

[19] "Why GPIO Zero Is Better Than RPi.GPIO for Raspberry Pi Projects," *makeuseof.com*, (https://www.makeuseof.com/tag/gpio-zero-raspberry-pi/)

[20] "Top 10 Machine Learning Frameworks," *hackernoon.com*, (https://hackernoon.com/top-10-machine-learning-frameworks-for-2019-h6120305j)

[21] "Curved Lane Detection," *Hackster,* 24 May 2018, (https://www.hackster.io/kemfic/curved-lane-detection-34f771)

[22] "YOLO Deep Learning: Don't Think Twice," *Missinglink*, (https://missinglink.ai/guides/computer-vision/yolo-deep-learning-dont-think-twice/)

[23] Martin, L., "The state of 3D object detection," *Towards Data Science,* (https://towardsdatascience.com/the-state-of-3d-object-detection-f65a385f67a8)

[24] "NTSB Report On Tesla Autopilot Accident Shows What's Inside And It's Not Pretty For FSD," *Forbes,* 6 Sept. 2019, (https://www.forbes.com/sites/bradtempleton/2019/09/06/ntsb-

report-on-tesla-autopilot-accident-shows-whats-inside-and-its-not-pretty-for-fsd/#6180ec6d4dc5)

[25] “LIDAR VS RADAR for Applied Autonomy,” *Semcon*, (https://semcon.com/what-we-do/applied-autonomy/LIDAR-vs-RADAR-for-applied-autonomy/)

[26] “Tracking T265,” *Intel,* (https://www.intelrealsense.com/tracking-camera-t265/)

[27] “Beginner’s Guide to Depth (Updated),” *Intel,* 15 July 2019, (https://www.intelrealsense.com/beginners-guide-to-depth/)

[28] Tian, D., “DeepPiCar – Part 4: Autonomous Lane Navigation via OpenCV,” *Towards Data Science,* (https://towardsdatascience.com/deeppicar-part-4-lane-following-via-opencv-737dd9e47c96)

[29] NVIDIA, “Jetson TX2,” (https://www.nvidia.com/en-us/autonomous-machines/embedded-systems/jetson-TX2/)

[30] “RPLIDAR-A1 360-degree laser range scanner,” *Slamtec,* (www.slamtec.com/en/LIDAR/A1)

[31] “Jetson RACECAR Part 3 – ESC Motor Controller,” *Jetson Hacks,* 26 Jan 2016, (www.jetsonhacks.com/2016/01/26/jetson-racecar-part-3-esc-motor-control/)

[32] “Krisdonia Portable Laptop Charger,” *Krisdonia*, (https://www.amazon.com/Krisdonia-Portable-TSA-Approved-25000mAh-Smartphone/dp/B074PQBRJV)

11 ACKNOWLEDGEMENTS

We would like to thank our academic advisor, Dr. Hasan Ferdowsi, for his constructive feedback and guidance during this project. We would also like to acknowledge our teaching assistant, Witenberg Santiago Rodrigues Souza, for his hard work and help during the completion of this project

12 APPENDIX

12.1 Updated Specifications

Physical:

Materials: Plastic, Rubber, Metal

Mechanical:

Size: 35.5 cm -56 cm
Weight: 3.2 kg
Speed:2.2 m/s - 4.5 m/s
Electric Motor Torque Rating: 8.6 kg-cm

Electrical:

Brushed Electric Motors
7 Cell NiMH battery
FLIPSKY ESC Version 4.12

Environmental:

Storage Temperature: 291 K to 300 K
Operating Temperature:283 K to 305 K
Operating Environment: Indoor

Software:

User interfaces: Linux Ubuntu 18.4
Hardware Interfaces: RC Vehicle, Lidar, ultra-sonic sensors, depth camera, tracking camera, Flipsky FLIPSKY ESC Nvidia TX2.
Communication Protocols: Wi-Fi
Features: Lane keeping, Lane changing algorithm, slow/stop object detection algorithm, data input reading/analyzing.
Computer Requirements:
Operating system: Linux Ubuntu 18.4
CPU:2 Denver 64-bit CPUs + Quad-Core A57 Complex
GPU: NVIDIA Pascal™ Architecture
Memory: 8 GB L128 bit DDR4
Flash Storage: 32 GB eMMC 5.1
Programming Languages: Python, ROS

Maintenance:

Keeping the electric motor brushes healthy
Keeping batteries charged
Making sure the chassis is stable while supporting the processors/sensors

Upgrading the suspension springs in the vehicle

12.2 Purchase Requisitions and Price Quotes

Figure 12 documents all the items purchased out of the team's senior design budget. It includes each item's vendor, their part numbers, and their prices. Team 46 was able to keep under the budget of $1,000 as directed by the College of Engineering and Engineering Technology at Northern Illinois University.

ORDER LIST

VENDOR	PART #	PART DESCRIPTION	SPECIFICATIONS	QTY	UNIT PRICE	TOTAL PRICE	WEBLINK TO PART
Intel RealSense	D435 + T265	Depth + Tracking Camera Bundle	N/A	1	359.00	367.07	https://store.intelrealsense.com/buy-intel-realsense-depth-and-tracking-bundle.html?_ga=2.163251583.2129293452.1574229107-1880265580.1573238944
Robot Shop	A2M8	Lidar	N/A	1	$320.00	320.00	https://www.robotshop.com/en/rplidar-a2m8-360-laser-scanner.html?gclid=CjwKCAiAws7uBRAkEiwAMlbZjjfKQZPizzFCk0H4QJBHaREiHGYUe1jWLQxFmTJ_SehcgZ1Dv_f1HhoCMOUQAvD_BwE
NVIDIA	TX2	NVIDIA TX2 Developer Kit Processor	Developer Kit	1	$299.00	299.00	https://developer.nvidia.com/embedded/jetson-tx2-developer-kit STUDENT DISCOUNT CODE from NVIDIA: hphu6a7qt
Lanmu	5823869367	DC to usb cable	N/A	1	$8.99	8.99	https://www.amazon.com/LANMU-Universal-Interchangeable-Connectors-Electronics/dp/B07KXBKHJZ/ref=sr_1_3?crid=2WNKRCBFS0TTI&keywords=dc%2Bto%2Busb%2Badapter&qid=1574799418&sprefix=dc%2Bto%2Busb%2Caps%2C136&sr=8-3&th=1
				GRAND TOTAL:		$995.06	

Figure 12. Purchase order sent to the Electrical Engineering Department at NIU.