Designing and Implementing an Autonomous RC Car

Eddy Perez-Carrillo

Brookdale Community College

April 20, 2023

Mentors: Dr. Christopher Ochs and Kevin Squires

Advisors: Karina Ochs and Ana Teodorescu

Abstract

Autonomous vehicles (AVs) have become an increasingly popular topic in recent years due to advancements in technology. While there have been significant developments in this field, there are still many limitations that need to be addressed. This report will be exploring the technologies behind autonomous vehicles, their current state in the auto market and in research, as well as the limitations they face. Additionally, an autonomous RC car will be designed and constructed to gain a better understanding of the challenges associated with creating a fully autonomous vehicle. This project aims to contribute to the advancement of autonomous vehicle technology and improve the general public's understanding of its potential benefits and limitations.

I. Introduction

Since the earliest example of an autonomous vehicle dating back to 1925 [17], significant advancements have been made in AV technology, leading to the development of some of the most advanced vehicles today. Large automakers such as General Motors (GM), Tesla, and Mercedes-Benz, as well as technology companies such as Waymo and Cruise are investing heavily in research and development to bring autonomous vehicles to the mass market [9], [12].

With studies predicting AVs will be commercially available by the end of the decade [12], it is important for the general public to have an understanding of the current technology and limitations of these vehicles. This research paper aims to improve public understanding of autonomous vehicles by designing and implementing an autonomous RC car using off-the-shelf hardware and open-source software. By providing a practical and hands-on demonstration of how autonomous vehicles operate, this paper will discuss the technology of AVs as well as their current limitations.

II. Background

A. Defining Autonomous Vehicles (AVs)

An autonomous vehicle (AV), often referred to as a self-driving car, is a vehicle that can operate without the need for human intervention. AVs are equipped with many sensors, cameras, and advanced algorithms that allow it to perceive its surroundings and make decisions based on what it sees. They combine advanced hardware with artificial intelligence to perform tasks typically done by human drivers.

For an autonomous vehicle to be fully autonomous, it must operate without human input. It is designed to navigate autonomously in any environment, meaning it must be equipped with hardware that can withstand any condition it may face while driving.

B. Society of Automotive Engineers (SAE) Six Levels of Driving AutomationSAE

The SAE defines six levels of automation for vehicles, ranging from Level 0 (no automation) to Level 5 (full automation) [27]. Each level of automation represents an increase in the vehicle's ability to control itself and handle driving tasks, and a decrease in the amount of input required from the driver. As the automation level increases, the vehicle is able to handle more complex and difficult driving situations without human intervention. The six levels of driving automation include:

- Level 0: No Automation - The driver is in full control of the vehicle at all times.
- Level 1: Driver Assistance - The vehicle has some automated features (i.e, adaptive cruise control, lane departure warning, etc.), but the driver is still responsible for controlling the vehicle.

- Level 2: Partial Automation - The vehicle can control both steering and acceleration under some conditions, but the driver is still required to remain attentive and take control if necessary.
- Level 3: Conditional Automation - The vehicle can manage most driving tasks, but the driver must remain attentive and be ready to take control if the system encounters a situation it can't navigate through.
- Level 4: High Automation - The vehicle can operate without human input in most situations but may require human intervention in some situations.
- Level 5: Full Automation - The vehicle is capable of operating without human intervention in all conditions and environments.

III. Research

A. Technology Used in Autonomous Vehicles

AVs use a variety of hardware components to collect and process data about the vehicle's environment. These components work together to enable autonomy, allowing the vehicle to make real-time decisions based on the data it collects from its many sensors. AVs gain visual and distance information through sensors such as LiDAR, radar, and cameras. This gives vehicles the ability to "see" everything around them, track moving objects, and send the information collected from these sensors to computers to be processed.

1. LiDAR

LiDAR (light detection and ranging) sensors are sensors that use pulsed laser waves to map the distance of objects surrounding an AV [1]. LiDAR measures distance to an object by sending multiple pulses of laser waves per second and recording the time it takes for each pulse to reach the target and bounce back to the sensor.

LiDAR sensors create 3D point clouds, which can be interpreted as maps, to understand their surroundings, allowing AVs to recognize the distances to obstacles. One of the biggest advantages of LiDAR is that it doesn't rely on visible light to operate; it uses near infrared light to generate the 3D maps necessary for navigation [6]. LiDAR sensors also have a very large field of view. Mechanical LiDAR systems all have a full horizontal FOV of 360° and may have a limited vertical FOV [17]. Solid-state LiDAR is more compact than mechanical LiDAR and tends to have a wider vertical FOV than with mechanical LiDAR, which can be beneficial in an automotive application [17].

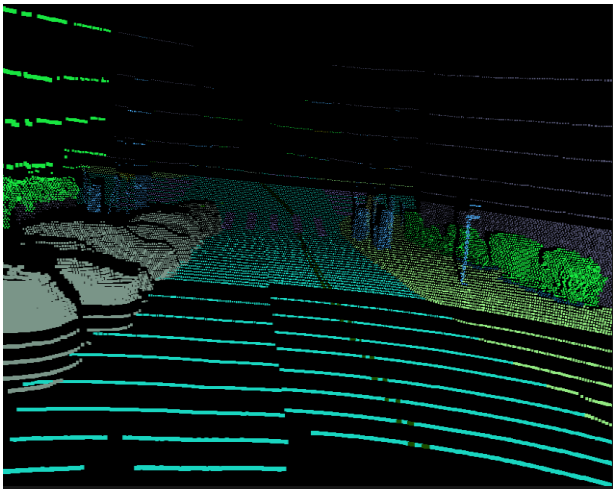

Figure 1- LiDAR Point Cloud of a street in San Francisco [28]

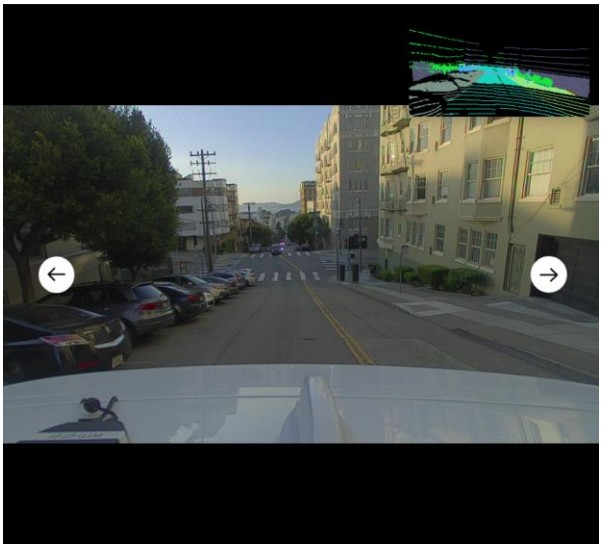

Figure 2- Camera view of same street in San Francisco [28]

2. Radar

Radar (radio detection and ranging) works very similar to LiDAR, with the main difference being that radar uses *radio waves* as opposed to near-infrared light waves. A very common form of radar used in AVs is millimeter-wave (mmWave) radar, which emits millimeter-long radio waves, allowing for the very accurate mapping [35]. This is because shorter wavelengths allows for greater resolution, but at the cost of the distance it can travel.

Radar is useful in autonomous vehicles because radar sensors can operate in all weather conditions with little change in effectiveness. They are also not affected by lighting conditions, which is ideal in creating a fully autonomous vehicle. Glares, low intensity light, and adverse weather all have negative impacts on the performance of sensors that rely on visible light, whereas radar is unaffected by any of these conditions. Radar sensors are also capable of detecting objects at ranges of up to 250m [35]. Similar to LiDAR, radar sensors are also capable of measuring relative velocities of objects, crucial for motion prediction and decision-making [6].

3. Cameras

Cameras are another crucial component used in every AV today. According to [11], they "are the most accurate way to create a visual representation of the world". AVs use multiple cameras all around the vehicle to give a full 360° view of its surroundings. These cameras range in FOV, allowing for the cameras to gain visual data from short and long ranges. While they are essential in providing visual information, cameras tend to perform poorly in more adverse environments, preventing AVs from driving in all weather conditions [6].

4. Computers

Computers are essential in autonomous vehicles. They provide the "brainpower" that allows AVs to operate on their own. Computers are responsible for collecting and processing data from all of the sensors and other components within the vehicle to make real-time driving decisions, such as controlling acceleration, braking, steering, and navigation. The computers used are systems-on-chips (SOCs), which are very small integrated circuits that combine multiple systems of a computer onto a single chip [5], allowing for powerful computers to be integrated into larger systems, such as an AV, without taking up much space. An example of one of these

powerful computers is NVIDIA's DRIVE PX2 computer, capable of multiple trillion operations per second [20].

5. Software

Software is a crucial component of AVs, serving as the "brain" that allows the vehicle to interpret data from its surroundings, perform complex tasks, and make decisions in real-time. Numerous algorithms are used for functions such as sensing, perception, localization, prediction, and control. Some of these algorithms include object detection and localization and mapping, among others [2],[3]. Object detection networks are made to detect all objects from one or several known classes in an image [3]. These networks utilize multiple sensors to accurately detect objects. A commonly used localization and mapping algorithm is SLAM (Simultaneous Localization And Mapping), which enables an AV to simultaneously map an unknown environment and position itself based on the constructed map [2]. Because of the high accuracy and resolution LiDAR offers as well as the low cost and visual information offered by cameras, the two sensors are commonly used in SLAM algorithms.

Not only is software used to allow AVs to function, it is also used to create virtual environments for the purpose of training AVs without having to physically have them drive. Simulations such as NVIDIA's "DriveSim" [7] and Waymo's "Simulation City" [34] are examples of simulations that allow for sensors to be tested in virtual environments and are used to train some of the most advanced vehicles today.

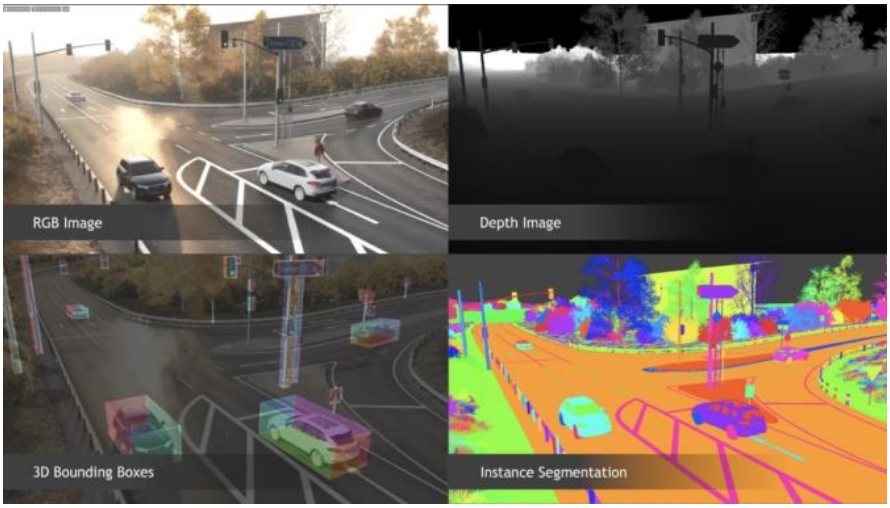

Figure 3- NVIDIA Drive Sim being used to gather data that will be used to train neural networks for autonomous vehicle perception

B. Autonomous Vehicles in the Current Auto Industry

In today's auto market, Level 2 automation is the highest level of vehicle automation that is commercially available [21]. While Level 3 automation has recently been achieved in the United States, it is currently limited to a single vehicle manufacturer, Mercedes-Benz, and operation is limited to only the state of Nevada [10]. Level 2 systems include Ford BlueCruise, Chevrolet Super Cruise, INFINITI ProPilot, and, most notably, Tesla Full Self-Driving (FSD) [23]. While some of the names of these products may suggest fully autonomous driving, these systems are all considered Advanced Driver Assistance Systems (ADAS) under SAE International standards. Drivers must remain attentive when using these ADAS features, as they are not intended to take full control over the vehicle at all times.

AV companies, such as Waymo LLC and Cruise LLC, are developing more sophisticated vehicles operating at Level 4 autonomy. Through a heavy use of simulations, as well as real-time testing on open roads, AV companies have been able to map millions of miles of driving to develop and train their systems to better navigate through their environment. Waymo is a leading company in the AV industry, with over 20 billion miles of simulated driving and over 20 million miles of

actual driving as of 2021 [13] with numbers continually increasing, far above most competitors. More recently, Waymo cars have driven over 1 million miles without a human driver behind the wheel [33]. Advancement in their AVs have allowed Waymo, among others, to begin implementing robotaxi services operating in limited areas, mainly in San Francisco and Phoenix, although other cities such as Las Vegas have also begun allowing AVs to operate on their roads [15].

C. Limitations

While recent developments have led to very intelligent AVs, they have their limitations that prevent them from becoming fully autonomous.

One of the biggest limiting factors is the impact adverse weather conditions have on AVs. Rain, snow, and fog, especially, can significantly limit the visibility of sensors, making it difficult for AVs to detect obstacles and navigate roads safely [6]. Cameras are most affected by the weather, as their accuracy and reliability greatly suffers when in unfavorable weather conditions. LiDAR sensors are less affected by weather, but their effectiveness is still reduced considerably in poor weather. Radar is able to detect obstacles in poor weather, but radar alone cannot safely control a vehicle.

Cybersecurity threats are also a limiting factor for autonomous vehicles. Software in regular passenger cars have shown to be vulnerable, as seen in one 2015 recall where Jeeps were able to be hijacked remotely, not only giving hackers almost complete control over the controls, but also allowing them to track the Jeep's location [22]. Such vulnerabilities generate fear and distrust in AVs, as they are equipped with much more technology than regular vehicles that use the internet to perform some of their functions. Having more connections, while enabling vehicles

to perform more increasingly complex tasks, also creates more vulnerabilities that could be open to malicious attacks. Combatting this issue requires the development of robust cybersecurity measures to ensure vehicles are not hacked or hijacked.

One of the possible benefits of AVs is to improve road safety because they would reduce the error caused by humans. However, AVs face the challenge of having to travel in uncertain environments filled with many unpredictable variables, such as other vehicles and pedestrians [21]. In one case with Uber, a jay-walker was struck and killed because the AV being tested had not accounted for the pedestrian to be walking in the street outside of crosswalks and was unable to plan the right path around them [24]. Additionally, according to [17], "the most important aspect of AVs is liability for accidents". In AVs, software is the main driver of the vehicle, responsible for making important driving decisions. It brings up the question of who is liable in the event an AV causes an accident.

IV. Implementation

This project describes the design and implementation of an autonomous RC car that is able to detect objects and map its surroundings. To do this, off-the-shelf hardware and mainly open-source software was used. Currently, the car is able to detect and classify objects in its path and stop and go based on if it detects an obstacle. It detected objects using detectNet v2, an object detection network developed by NVIDIA.

A. Hardware

The budget for the car was limited to about $120[1], thus it could only be equipped with some of the hardware commonly found on modern AVs and the hardware was relatively basic, as

[1] Note- While the project budget was limited to about $120, the reComputer J1010 Nano and the RPLiDAR A1 were purchased out of pocket and were not included in the project budget.

compared to commercially produced RC cars. Emphasis in the design was placed on a camera and LiDAR sensor for visual information and a computer that was suited for machine learning applications. While this hardware focus allowed for partial automation, it did not allow for full automation, as the vehicle did not have enough sensors or computational power to obtain the large amounts of information needed to complete complex tasks, such as decision-making. Given the hardware that could be obtained within the budget, the car would be capable of detecting objects in its path, identifying the object and its confidence, and mapping its distance from the object. It is important to note the J1010 For the hardware components, the following were selected:

- Slamtec RPLiDAR A1 ($99)
- Seeed Studio IMX2190169 8MP Camera ($24.90)
- Seeed Studio reComputer J1010 Nano ($199)
- Arduino Uno Rev3 ($20.70)
- Arduino Motor Driver L298N ($11.39)
- Bemonoc DC Motor with Encoder ($14.88/ea.)

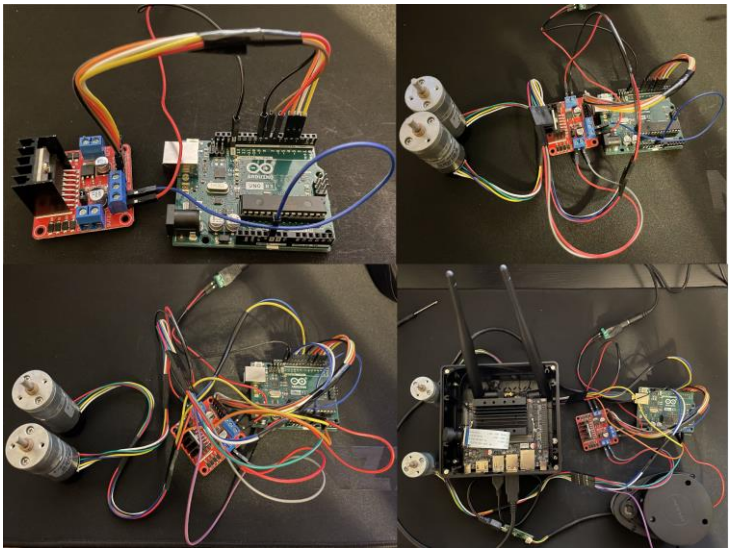

Figure 4- Connecting hardware components

Slamtec's RPLiDAR A1 is an accurate, low-cost, mechanical LiDAR sensor, capable of a 360° horizontal FOV, but no vertical FOV. The LiDAR has a 12m range and an angular resolution of about 1°. Because it is a mechanical LiDAR, the RPLiDAR A1 is only capable of creating a 2D map.

The camera the RC car is equipped with is an 8MP Sony IMX-219 image sensor with a 160° FOV. It is capable of a resolution of 3280 × 2464 and utilizes the MIPI CSI-2 interface. The camera can capture high quality images and the large FOV allows the camera to record a wider frame.

The J1010 computer is a computer developed by Seeed Studio that utilizes NVIDIA's Jetson Nano module, designed for basic, entry-level AI applications. NVIDIA's Jetson Nano runs on their own Jetson Linux OS and uses their JetPack SDK, a software development kit with numerous AI development tools, such as TensorRT and OpenCV.

Arduino is an open-source hardware and software company that manufactures electronic devices, such as the Uno Rev3 and Motor Driver microcontrollers. An Uno Rev3 is used to allow the Jetson Nano computer to send commands to the motor controller to enable the motors to run at different speeds or reverse directions. The motor controller controls two Bemonoc 12V DC motors, capable of up to 600 RPM.

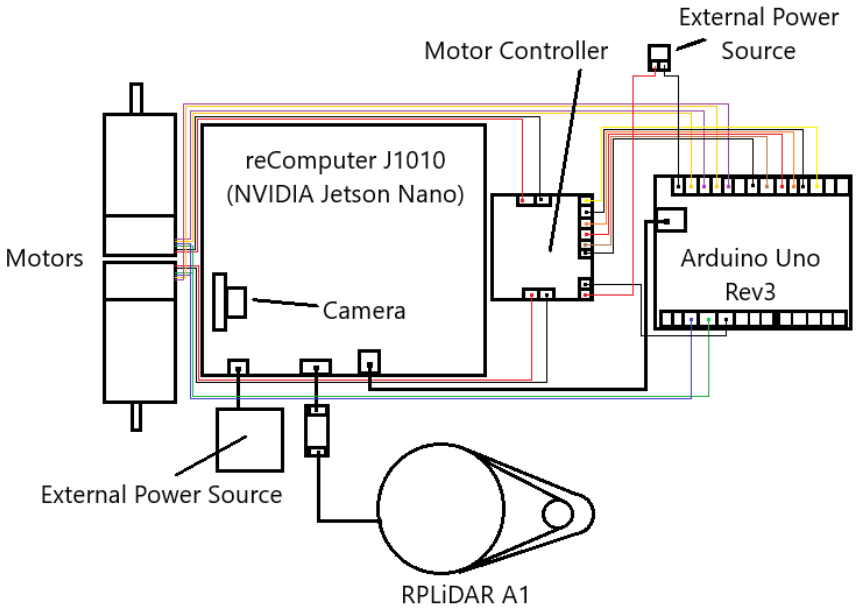

Figure 5- Diagram of Hardware for RC car

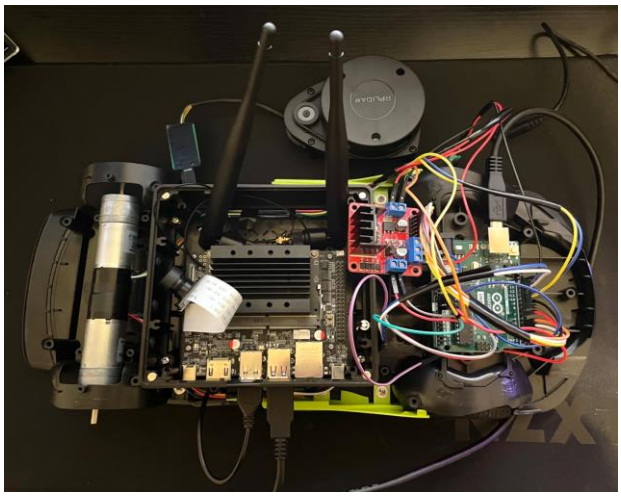

Figure 6- Fitting Parts on RC Car

Figure 5 shows a schematic for how each component was connected together and Figure 6 shows the actual hardware. Jumper wires are used to connect the motors with the Arduino Uno and motor controller. The Arduino Uno is connected to the NVIDIA Jetson Nano using a USB-A to USB-B cable. The Jetson Nano is connected to the RPLiDAR through a USB-A to Micro USB cable. The LiDAR also uses a Micro USB to UART bridge to make transmitting data easier. This is because the bridge is a communication path between the two communication protocols [31],

allowing for LiDAR data sent using the UART protocol to be received by the Jetson Nano via USB.

B. Software

For writing the code for the RC car, Visual Studio Code is the IDE used and the code for the car is written in the Python programming language. Pyfirmata, OpenCV, and Jetson Inference by DustyNV on Github are used for performing functions such as opening the camera and running the object detection network. Pyfirmata is a python interface for the firmata protocol, meant for communicating with microcontrollers through software on a computer [29]. It allows the Jetson Nano computer to easily communicate with the Arduino Uno microcontroller. OpenCV is an open-source, computer vision library, aimed at providing the tools needed for computer vision applications [26]. In this project, it was used for displaying the camera feed using the GStreamer framework; converting the frames captured from one subpixel layout to another one that could be used by the Jetson Nano's detection network; and mapping out boxes that indicate the object(s) the Jetson Nano is detecting at that moment.

```
import time
import pyfirmata
import jetson_inference
import jetson_utils
import cv2

board = pyfirmata.Arduino('/dev/ttyACM0')

en_a_pin = board.get_pin("d:9:p")
en_b_pin = board.get_pin("d:3:p")
in_1_pin = board.get_pin("d:8:o")
in_2_pin = board.get_pin("d:7:o")
in_3_pin = board.get_pin("d:5:o")
in_4_pin = board.get_pin("d:4:o")
motor_1A = board.get_pin("d:12:i")
motor_1B = board.get_pin("d:13:i")
motor_2A = board.get_pin("d:10:i")
motor_2B = board.get_pin("d:11:i")
```

Figure 7- Beginning of code for RC car. The code began with importing necessary libraries and setting pins as variables

```
def gstreamer_pipeline(
        capture_width=3264,
        capture_height=1848,
        display_width=960,
        display_height=616,
        framerate=24,
        flip_method=0,
):
    return (
        "nvarguscamerasrc ! "
        "video/x-raw(memory:NVMM), "
        "width=%d, height=%d, "
        "format=NV12, framerate=%d/1 ! "
        "nvvidconv flip-method=%d ! "
        "video/x-raw, width=%d, height=%d, format=BGRx ! "
        "videoconvert ! "
        "video/x-raw, format=BGR ! appsink"
        % (
            capture_width,
            capture_height,
            framerate,
            flip_method,
            display_width,
            display_height,
            )
    )
    #Detection netork and camera
cap = cv2.VideoCapture(gstreamer_pipeline(flip_method=0), cv2.CAP_GSTREAMER)
net = jetson_inference.detectNet('ssd-mobilenet-v2', threshold = 0.6)
```

Figure 8- Creating camera feed and initializing object detection network

After importing the necessary libraries and setting up the Arduino board, the camera feed was created. The camera's parameters were set before initializing the camera feed using OpenCV's "VideoCapture" function and the GStreamer pipeline. Once the video began, the jetson inference library was used to initialize the detection network (detectNet). The detectNet network utilizes TensorFlow's object detection model, running on the SSD-MobileNet-V2 architecture and the COCO 2017 dataset. The confidence threshold was set to 0.6, meaning any objects detected with a confidence above 60% would be detected by the network. Due to the limited dataset used, the detectNet network was unable to detect more than 91 different objects, many of which are not relevant in an automotive application.

```
60      #Forward at Full Speed
61  en_a_pin.write(1)
62  en_b_pin.write(1)
63  in_1_pin.write(0)
64  in_2_pin.write(1)
65  in_3_pin.write(0)
66  in_4_pin.write(1)
```

Figure 9: Starting the motors at full speed

```
67
68  while True:
69      ret, img = cap.read()
70
71      img = cv2.cvtColor(img, cv2.COLOR_BGR2RGBA)
72      tensor = jetson_utils.cudaFromNumpy(img)
73      #Detect objects in frame
74      detections = net.Detect(tensor, img.shape[1], img.shape[0])
75
76      for detection in detections:
77              #Draw boxes around detected objects
78          left = int(detection.Left)
79          top = int(detection.Top)
80          right = int(detection.Right)
81          bottom = int(detection.Bottom)
82          cv2.rectangle(img, (left, top), (right, bottom), (0, 255, 0), 2)
83          print("Detected object:", detection.ClassID, "Condifence:", detection.Confidence)
84          time.sleep(0.5)
85
86          #Display image
87      img = cv2.cvtColor(img, cv2.COLOR_RGBA2BGR)
88      cv2.imshow("Object Detection", img)
89
90          #Stop motors if object is detected and wait 3 seconds
91      if detections:
92          en_a_pin.write(0)
93          en_b_pin.write(0)
94          time.sleep(3)
95          #Start motors if no objects detected
96      else:
97          en_a_pin.write(1)
98          en_b_pin.write(1)
99
100     if cv2.waitKey(1) == ord('q'):
101         en_a_pin.write(0)
102         en_b_pin.write(0)
103         break
104
105 cap.release()
106 cv2.destroyAllWindows
```

Figure 10: While loop running to capture frames from the camera and detect objects in the frames captured. Once an object is detected, or no longer detected, output commands are sent to the Arduino Uno.

The two motors are initially set to full speed, before a while loop runs continuously to detect objects as long as the camera is capturing the video feed. The loop begins by reading a frame from the camera and converting that image from BGR to RGBA, then converting the image from a Numpy array to a tensor format, to be interpreted by the Jetson Nano. Afterwards, the detectNet network detects objects found in the frame and bounding boxes are drawn around the detected objects. The class ID of the object detected and the network's confidence are also recorded. If objects are detected in the frame, the motors stop and wait until there are no objects in its frame. If the RC car is connected to a monitor, the image is also converted back to BGR and displayed on the monitor.

C. Results

In this project, an initial design for the RC car was created, as shown in Figure 2. The hardware was successfully tested and integrated into the RC car. Open-source software was combined with code I wrote to allow all of the components to communicate with each other and allow the RC car to move and perform necessary functions.

However, there were some setbacks, such as a lack of documentation for some of the hardware, not having all of the hardware needed readily available, and having to learn how to code, among others, through the process of adding hardware to the RC car to enable autonomous driving. Additionally, while the RPLiDAR A1 was able to connect to a desktop computer, it was not able to connect to the Jetson Nano.

Throughout this project, it became evident that designing and creating an autonomous vehicle is a difficult task that requires a significant amount of time and planning. Nevertheless, this project has had many successes, resulting in a functioning RC car that can detect some objects and respond accordingly.

V. Future Work

Future work will focus on achieving full automation with the RC car by adding more sensors, better motors, and improving the software the RC car uses. Rather than using a single mechanical LiDAR, adding a solid-state LiDAR sensor would be much better, because the car would have a horizontal and vertical FOV and therefore be able to create 3D maps of its environment. Extra cameras would also allow for better object detection from multiple angles, and adding radar sensors would help the car determine the velocities of objects detected within its view. Replacing the current motors with more powerful motors would help the car move faster with the added weight from the sensors. Additionally, the car would be scaled up to a golf cart or small car once the RC car has an acceptable level of automation, further upgrading the hardware to automotive-grade sensors and computers.

VII. Conclusion

This project focused on designing and constructing an autonomous RC car. Despite facing many challenges, the project was completed with most components integrated successfully. To reach full automation, the vehicle would need to be equipped with more sensors, as well as more complex algorithms that would allow the vehicle to make real-time decisions. Further training would also be required to ensure networks and algorithms would be able to perform at levels greater than or equal to that of a human driver.

Acknowledgments

This work was supported by the New Jersey Space Grant Consortium and Brookdale Community College. I would like to thank my mentors, Professor Kevin Squires and Dr. Christopher Ochs, as well as my advisors, Professor Karina Ochs and Professor Ana Teodorescu.

References

1. Aalerud, Atle, et al. “Reshaping Field of View and Resolution with Segmented Reflectors: Bridging the Gap between Rotating and Solid-State Lidars.” *Sensors*, vol. 20, no. 12, 15 June 2020, p. 3388, https://doi.org/10.3390/s20123388.
2. Alsadik, Bashar, and Samer Karam. “The Simultaneous Localization and Mapping (SLAM)-an Overview.” *Journal of Applied Science and Technology Trends*, vol. 2, no. 04, 2021, pp. 120–131, https://doi.org/10.38094/sgej1027.
3. Amit, Yali, et al. “Object Detection.” *Computer Vision*, 9 Mar. 2020, pp. 1–9, https://doi.org/10.1007/978-3-030-03243-2_660-1.
4. Arnold, Eduardo, et al. “A Survey on 3D Object Detection Methods for Autonomous Driving Applications.” *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 10, 22 Jan. 2019, pp. 3782–3795, https://doi.org/10.1109/tits.2019.2892405.
5. Badawy, Wael, and Graham Jullien. *System-on-Chip for Real-Time Applications*. Kluwer Academic Publishers, 2003.
6. Campbell, Sean, et al. “Sensor Technology in Autonomous Vehicles : A Review.” *2018 29th Irish Signals and Systems Conference (ISSC)*, 2018, https://doi.org/10.1109/issc.2018.8585340.
7. Cragun, Matt. “Nvidia Drive Sim Powered by Omiverse.” *NVIDIA Blog*, 15 Nov. 2022, blogs.nvidia.com/blog/2021/04/12/nvidia-drive-sim-omniverse-early-access/.
8. Fan, Yingchun, et al. “Semantic Slam with More Accurate Point Cloud Map in Dynamic Environments.” *IEEE Access*, vol. 8, 17 June 2020, pp. 112237–112252, https://doi.org/10.1109/access.2020.3003160.
9. “GM Announces Additional Investment in Cruise.” *GM Corporate Newsroom*, 18 Mar. 2022, news.gm.com/newsroom.detail.html/Pages/news/us/en/2022/mar/0318-cruise.html.
10. Group, Mercedes-Benz. “Benz World’s First Automotive Company to Certify SAE Level 3 System for U.S. Market: Mercedes-Benz Group.” *Mercedes*, 26 Jan. 2023, group.mercedes-benz.com/innovation/product-innovation/autonomous-driving/drive-pilot-nevada.html.
11. Gujarathi, Aniket, et al. *Design and Development of Autonomous Delivery Robot*, 16 Mar. 2021, https://doi.org/10.48550/arXiv.2103.09229.
12. Guo, Yuntao, et al. “Paving the Way for Autonomous Vehicles: Understanding Autonomous Vehicle Adoption and Vehicle Fuel Choice under User Heterogeneity.” *Transportation Research Part A: Policy and Practice*, vol. 154, 2021, pp. 364–398, https://doi.org/10.1016/j.tra.2021.10.018.
13. Holt, Kris. “Waymo’s Autonomous Vehicles Have Clocked 20 Million Miles on Public Roads.” *Engadget*, 19 Aug. 2021, www.engadget.com/waymo-autonomous-vehicles-update-san-francisco-193934150.html.

14. Ignatious, Henry Alexander, et al. “An Overview of Sensors in Autonomous Vehicles.” *Procedia Computer Science*, vol. 198, 1 Nov. 2022, pp. 736–741, https://doi.org/10.1016/j.procs.2021.12.315.
15. John, Krafcik. “Waypoint - the Official Waymo Blog: Waymo Is Opening Its Fully Driverless Service to the General Public in Phoenix.” *Waypoint – The Official Waymo Blog*, 8 Oct. 2020, blog.waymo.com/2020/10/waymo-is-opening-its-fully-driverless.html.
16. Kato, Shinpei, et al. “An Open Approach to Autonomous Vehicles.” *IEEE Micro*, vol. 35, no. 6, 29 Dec. 2015, pp. 60–68, https://doi.org/10.1109/mm.2015.133.
17. Khayyam, Hamid, et al. “Artificial Intelligence and Internet of Things for Autonomous Vehicles.” *Nonlinear Approaches in Engineering Applications*, 2019, pp. 39–68, https://doi.org/10.1007/978-3-030-18963-1_2.
18. Li, You, and Javier Ibanez-Guzman. “Lidar for Autonomous Driving: The Principles, Challenges, and Trends for Automotive Lidar and Perception Systems.” *IEEE Signal Processing Magazine*, vol. 37, no. 4, 29 June 2020, pp. 50–61, https://doi.org/10.1109/msp.2020.2973615.
19. Liu, Liangkai, et al. “Computing Systems for Autonomous Driving: State of the Art and Challenges.” *IEEE Internet of Things Journal*, vol. 8, no. 8, 2021, pp. 6469–6486, https://doi.org/10.1109/jiot.2020.3043716.
20. Liu, Shaoshan, et al. “Computer Architectures for Autonomous Driving.” *Computer*, vol. 50, no. 8, 1 Aug. 2017, pp. 18–25, https://doi.org/10.1109/mc.2017.3001256.
21. Mallozzi, Piergiuseppe, et al. “Autonomous Vehicles: State of the Art, Future Trends, and Challenges.” *Automotive Systems and Software Engineering*, 18 July 2019, pp. 347–367, https://doi.org/10.1007/978-3-030-12157-0_16.
22. Miller, Charlie, and Chris Valasek. 2015, *Remote Exploitation of an Unaltered Passenger Vehicle*.
23. Mueller, Alexander S, et al. *Habits, Attitudes, and Expectations of Regular Users of Partial Driving ...*, Oct. 2022, www.iihs.org/api/datastoredocument/bibliography/2261.
24. National Transportation Safety Board, Tempe, AZ, 2018, *Collision Between Vehicle Controlled by Developmental Automated Driving System and Pedestrian*.
25. NVIDIA. “DetectNet_v2.” *NVIDIA Docs*, docs.nvidia.com/tao/tao-toolkit/text/object_detection/detectnet_v2.html. Accessed 12 May 2023.
26. Pulli, Kari, et al. “Realtime Computer Vision with OpenCV.” *Queue*, vol. 10, no. 4, 2012, pp. 40–56, https://doi.org/10.1145/2181796.2206309.
27. SAE International. “Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles.” *Surface Vehicle Recommended Practice*, Apr. 2021, https://doi.org/10.3403/30443080.
28. Scale. “Pandaset Open Datasets - Scale.” *ScaleAI*, scale.com/open-av-datasets/pandaset. Accessed 12 May 2023.
29. Seeed Studio. “Firmata for Arduino Compatible Boards: Seeed Studio Wiki.” *Seeed Studio Wiki RSS*, wiki.seeedstudio.com/ODYSSEY-X86J4105-Firmata/. Accessed 13 May 2023.

30. Su, Peng, et al. “Real-Time Dynamic Slam Algorithm Based on Deep Learning.” *IEEE Access*, vol. 10, 17 Aug. 2022, pp. 87754–87766, https://doi.org/10.1109/access.2022.3199350.
31. Texas Instruments, 2021, *USB-to-UART Bridge Made Simple With MSP430 MCUs*.
32. Van Brummelen, Jessica, et al. “Autonomous Vehicle Perception: The Technology of Today and Tomorrow.” *Transportation Research Part C: Emerging Technologies*, vol. 89, Feb. 2018, pp. 384–406, https://doi.org/10.1016/j.trc.2018.02.012.
33. Waymo Team. “Waypoint - the Official Waymo Blog: First Million Rider-Only Miles: How the Waymo Driver Is Improving Road Safety.” *Waypoint – The Official Waymo Blog*, 28 Feb. 2023, blog.waymo.com/2023/02/first-million-rider-only-miles-how.html.
34. Waymo Team. “Waypoint - the Official Waymo Blog: Simulation City: Introducing Waymo’s Most Advanced Simulation System yet for Autonomous Driving.” *Waypoint – The Official Waymo Blog*, 6 July 2021, blog.waymo.com/2021/06/SimulationCity.html.
35. Wei, Zhiqing, et al. “MmWave Radar and Vision Fusion for Object Detection in Autonomous Driving: A Review.” *Sensors*, vol. 22, no. 7, 25 Mar. 2022, p. 2542, https://doi.org/10.3390/s22072542.