Spring 5-1-2021

Path Planning with Deep Neural Networks

Paul Simmerling
University of Connecticut - Storrs, paul.simmerling@uconn.edu

Brendan Sayers
University of Connecticut - Storrs, brendan.sayers@uconn.edu

Paulo Alcantara Silva
University of Connecticut - Storrs, p.hal@uconn.edu

Path Planning with Deep Neural Networks, Team 2107

Paul Simmerling[1,2], Paulo Alcântara[1], Brendan Sayers[1],
CSE Students: Yun-Hwan Kim[3], Nathan Choi[3], Tim O'Reilly[3],
Advisors: Shalabh Gupta[1], Fei Miao[3], Karl Berntorp[4]

[1]*Electrical Engineering*, University of Connecticut, Storrs, CT 06269
[2]*Physics*, University of Connecticut, Storrs, CT, 06269
[3]*Computer Science*, University of Connecticut, Storrs, CT, 06269
[4]Mitsubishi Electric Research Laboratories, Cambridge, MA 02139

Project Website: https://ecesd.engr.uconn.edu/ecesd2107/
Source Code and Documentation: https://github.com/psimmerl/merl_bot
This project is sponsored by Mitsubishi Electric Research Laboratories (*MERL*)
MERL Contact: *berntorp@merl.org*
For questions or comments regarding this project, please contact
paul.simmerling@uconn.edu

CONTENTS

LIST OF FIGURES

LIST OF TABLES

GLOSSARY

Ackermann drive	steering method used in cars. 6, 14
Arduino	single board micro-controller. 6, 7, 13, 14
autonomous	vehicle that does not need a human driver. 4, 6, 13
CAD	Computer Aided Design. 7
chassis	structure of a robot. 2, 6, 11, 13, 14
encoder	wheel rotation sensor. 4, 6, 7, 13, 14
ESC	Electronic Speed Controller for controlling motors. 2, 11
IMU	Interial Measurement Unit - determine orientation. 4, 6, 7, 13, 14
LIDAR	$360\deg$ distance sensor that uses a laser. 4, 6, 7, 8, 13, 14
Nvidia Jetson	small format computer designed for machine learning. 8
RACI chart	lists team member responsibilities. 2, 13
Raspberry Pi	small format computer. 7, 8, 13
ROS	Robot Operating System - OS used for controlling robots. 7, 9, 13
SLAM	Simultaneous Localization and Mapping. 5
ultrasonic	distance sensor that uses ultrasonic waves. 7

Abstract

This report will cover the work and plans of the ECE 2107 Senior design team. The goal of the project is to design and build a fully autonomous self-driving car. This car will have a complete sensor suite including LIDAR, an IMU, a camera, and encoders. It will be based on a multi-level system where the highest level uses a neural network for advanced signal processing and analysis. The current state of the project is discussed as well as the final results. Project management and other constraints will be briefly investigated. This team is building a self driving car testbed for MERL. The vehicle has been constructed and the sensors have been validated.

I. Introduction

Since their invention in the 1800s, cars have become a cornerstone of the modern world. Used in applications ranging from transporting goods thousands of miles to visiting a friend in a nearby neighborhood, road capable vehicles are essential for billions of people. Recent advances in data-driven machine learning and the development of precise, cheap localization sensors has led to the development of autonomous vehicles. Self-driving cars have been shown to decrease the number of car accidents resulting in fewer injuries and deaths[2]. The U.S. automotive industry market was $545.4 billion in 2018 and accounted for 2.7% of the U.S. gross domestic product [3]. It is expected that in the coming years, this market will be encompassed entirely by autonomous vehicles. As such, new models for autonomous driving must be developed and tested.

In association with Mitsubishi Electric Research Laboratories (MERL), the U.S. research arm of Mitsubishi, the University of Connecticut (UConn) Electrical and Computer Engineering (ECE) Senior Design Team 2107 will designed and developed a new self-driving car testbed. This vehicle will use a novel neural network outlined in the U.S. Patent Number 9,989,964 by Dr. Karl Berntorp[1]. This project posed a variety of challenges to the senior design team. The primary challenge was designing a robot that would be able to provide an adequate assessment of its environment and have the ability to provide real-time feedback such that the vehicle would be able to navigate and avoid obstacles. The neural network design was done in conjunction with the Computer Science and Engineering (CSE) Department.

II. Problem Statement

A. Statement of Need & Specifications

The project sponsor has developed a novel method for Simultaneous Localization and Mapping (SLAM). In this control problem, a vehicle creates a map of its surroundings while simultaneously as the car drives through it[1]. It was the primary objective of this project to design and implement a small-scale self-driving vehicle that could use the neural network outlined in the patent such that the vehicle would adhere to a variety of constraints listed by the patent, see figure 1 for these specifications.

	Specifications on motion of the vehicle
500	Stay on the road
510	Stay in the middle of the lane
520	Maintain nominal longitudinal velocity
530	Maintain safety margin to surrounding obstacles
540	Maintain minimum distance to vehicles in the same lane
550	Drive smoothly

Fig. 1: Specification on the motion of the vehicle per the neural network [1].

Several additional requirements were placed on the testbed design by the project sponsor. One of these requirements was that the vehicle must operate in the same manner as a standard road capable vehicle. The small-scale car should use sensors to generate ground truth labels and training datasets for a vehicle maneuvering on the road.

Finally, a machine learning (specifically deep neural network) system must be developed that uses the sensor information mentioned earlier to construct and plan vehicle motion in response to the sensor input. This final requirement was completed primarily on the CSE team's side.

B. Basic Limitations on Project Scope

Initial limitations hailed from concerns about processing power and budget concerns. For the vehicle to be entirely autonomous, numerous sensors and robust enough controllers must be installed onto the chassis. While some of these sensors can be reasonably inexpensive, several of them, such as the LIDAR, can be very expensive. The vehicle also needed to process the data fast enough to provide real-time feedback responses. Rapid feedback responses will require a powerful processor; quality processors can be in the range of several hundred dollars. The implementation of the vehicle required balancing budget concerns against sensor quality and processing power.

III. Approach and Design

A. Multilevel System

It was essential for the final design to meet all of the requirements and specifications listed in figure 1 and any other constraints from the sponsor. Based on this information, a design was created that uses two systems to control the vehicle's final motion.

The first system was the low-level system. This system is involved in all operations involving basic robot controls such as turning, accelerating, maintaining velocity, and maintaining course. Specifically, this system will be adhering to the specifications 510, 520, and 550 seen in figure 1. For the hardware of the system, an Ackermann drive chassis will be used. Using Ackermann drive, the requirement for simulating the driving dynamics of a standard road vehicle will be satisfied. This system included two optical encoders, one on each rear wheel. These encoders will be used for all velocity controls. The system also included an inertial measurement unit (IMU); this IMU will maintain attitude and direction such that the car can drive straight.

Additionally, this system has two batteries. The first battery powered the main motor, its electronic speed controller (ESC), and the steering servo. The second was meant for powering all of the sensors, the high-level system, and the Arduino, which will be the processor for all low-level operations.

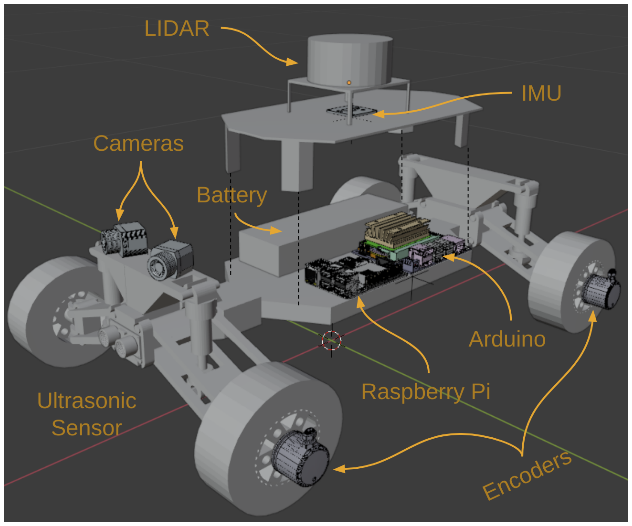
The second system was for high-level operations. A Raspberry Pi 4B 8GB was used for data processing and communication with the Arduino and an external laptop in this system. A LIDAR sensor was situated on the top of the robot for 360º of distance measurements. A camera module was also used on the front of the robot to identify obstacles and potential paths. This data was then processed on the Raspberry Pi, which was running Robot Operating System (ROS), and then pertinent information could then be communicated via wifi to a nearby laptop. This laptop was then used for more demanding tasks such as running the neural network and performing additional image processing.

A CAD drawing of the proposed design can be found in figure 2a/b. Figure 2a provides an isometric view of the full robot, whereas figure 2b is the unexploded view.

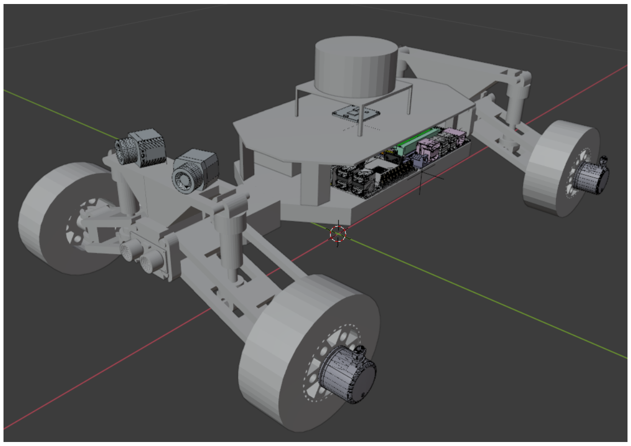
The two systems could communicate with each other using serial UART. The low-level system would receive orders from the high-level system regarding changing velocity and heading. The low-level system would also send its encoder and IMU data to the high-level system. The high-level system then functioned as a wifi access point and would perform simple sensor fusion, such as determining if any emergency stop code needs to be sent to the robot to avoid crashing. It would also send its data to the laptop, which would record all data for training and process the neural network/camera vision. The laptop would send the neural network's output back to the Raspberry Pi and, ultimately, the Arduino. These systems would be able to allow for obstacle avoidance and path planning. Figure 3 is the signal flow diagram of the final system.

B. *Alternative Design Solutions*

While the vehicle's primary plan has been approximately the same throughout all iterations, several variations have been considered. One variation proposed and spearheaded by one team member was to use a tank drive system where a separate motor will control each wheel. These motors would allow the robot to independently set each wheel's velocity, therefore, control the turning speed and radius of the car. Unfortunately, this variation does not align with the requirement that the vehicle has the same dynamics as a standard road vehicle; thus, it was not appropriate to implement. Another variation used ultrasonic sensors in addition to the LIDAR module. The theory was that this would allow for more training data and give measurements for nearby obstacles. It was decided not to include ultrasonic on the final design since the overlap with LIDAR is significant, and the ultrasonic sensor would not provide meaningful data. The



(a) Exploded view with component labels.



(b) Assembled view.

Fig. 2: CAD model of the current proposed design.

last alternative was using an Nvidia Jetson Nano instead of a Raspberry Pi 4B for processing the data for the neural network, cameras, and LIDAR. The Jetson would give a superior performance at the cost of being more expensive. It was decided to go with the Raspberry Pi 4B 8GB version since the price was too high, and the idea of using a remote laptop to process the data was preferred.

C. Constraints

As a result of the coronavirus pandemic, a significant constraint is placed on the vehicle's construction and recording training data. Typically, the construction would be done in a shared

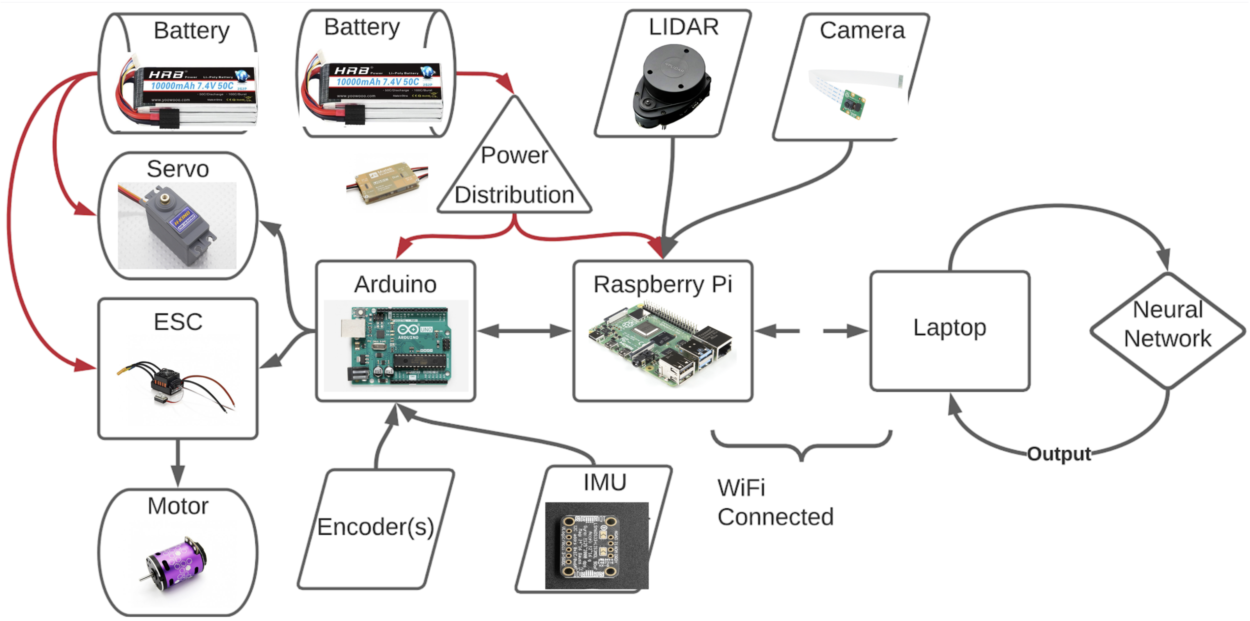


Fig. 3: Signal flow graph of final system.

space such as a workbench/laboratory provided by the University of Connecticut Electrical and Computer Engineering Department; unfortunately, this is no longer a valid option. In order to work around this issue, team 2107 split the robot into three separate sections for each team member to build and diagnose. Once each section was completed, the device was then delivered to a Paul Simmerling's residence for assembly. When the lab's in the basement of ITE opened up for group work, team 2107 would meet on a weekly basis to finish the construction of the robot. While this method worked, we believe it stifled the productivity and creativity on the project significantly.

D. Final Results

Once the robot chassis was assembled by Brendan Sayers, the team was able to work collaboratively on the robot. Simmerling was able to set up the Raspberry Pi to host a wifi network and run Robot Operating System (ROS). Simmerling also wrote the code for the nodes on the Pi as well as the laptop. ROS had many built in libraries that allowed for easy diagnostics of the camera as well as the LIDAR. Simmerling wrote the control code for the Arduino and calibrated the IMU to work with a PID loop to drive the robot straight.

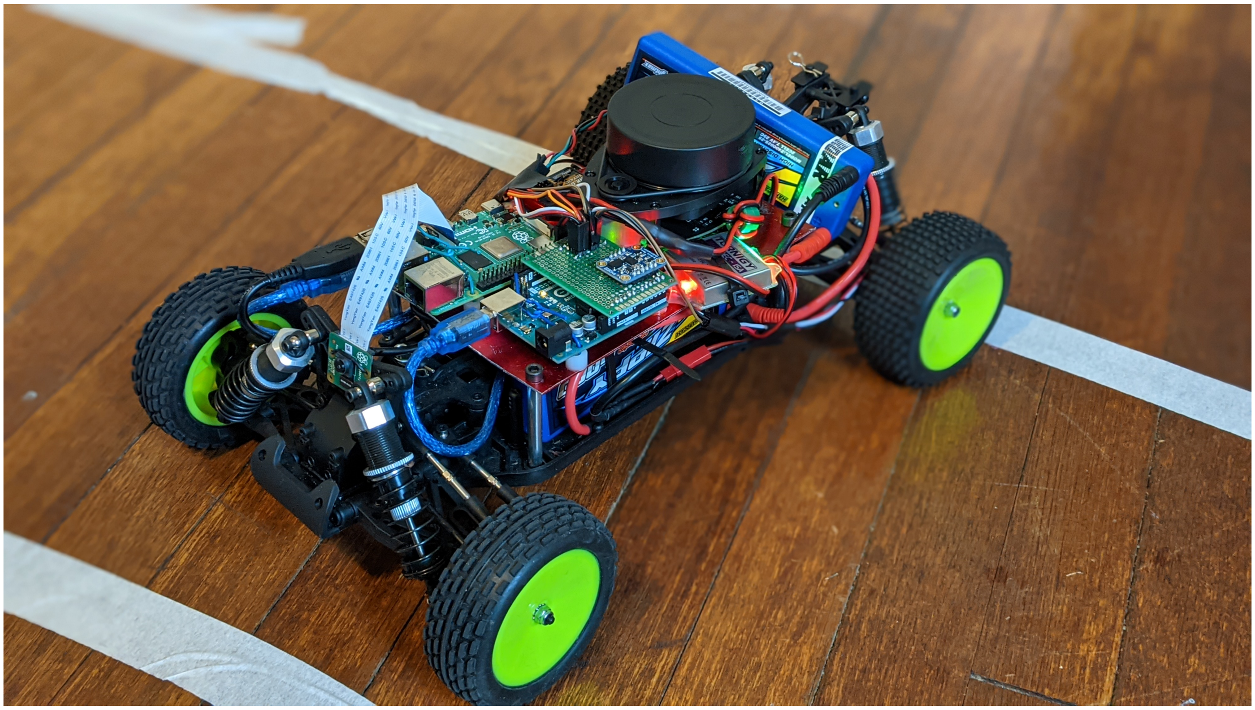


Fig. 4: Side view of the robot.

The robot was then assembled. The final robot can be seen in figures 4 and 5. These figures show the robots in the completed state. This robot set into two modes; training, and testing. In the training mode an X-Box controller can be used to drive the robot and record data for testing. In the testing mode, the trained neural net can steer the car.

Using ROS, we were able to set up SLAM using HectorSLAM. With HectorSLAM, we were able to generate a map of the surrounding terrain using the LIDAR. This allowed us to find the x/y coordinates of the vehicle (in reference to the starting position), which was essential for the neural net to work. See figure 6 for an example of a map generated using HectorSLAM.

The CSE team also made headway on their design of the neural network. They were able to used the NVidia PilotNet model to train the robot. Using PilotNet, they have been able to test and train a model car using the simulation software Carla. However, they used only camera data whereas the vehicle team focused their efforts into the LIDAR. This meant the vehicle team had to change their final product to work with the CSE teams needs.

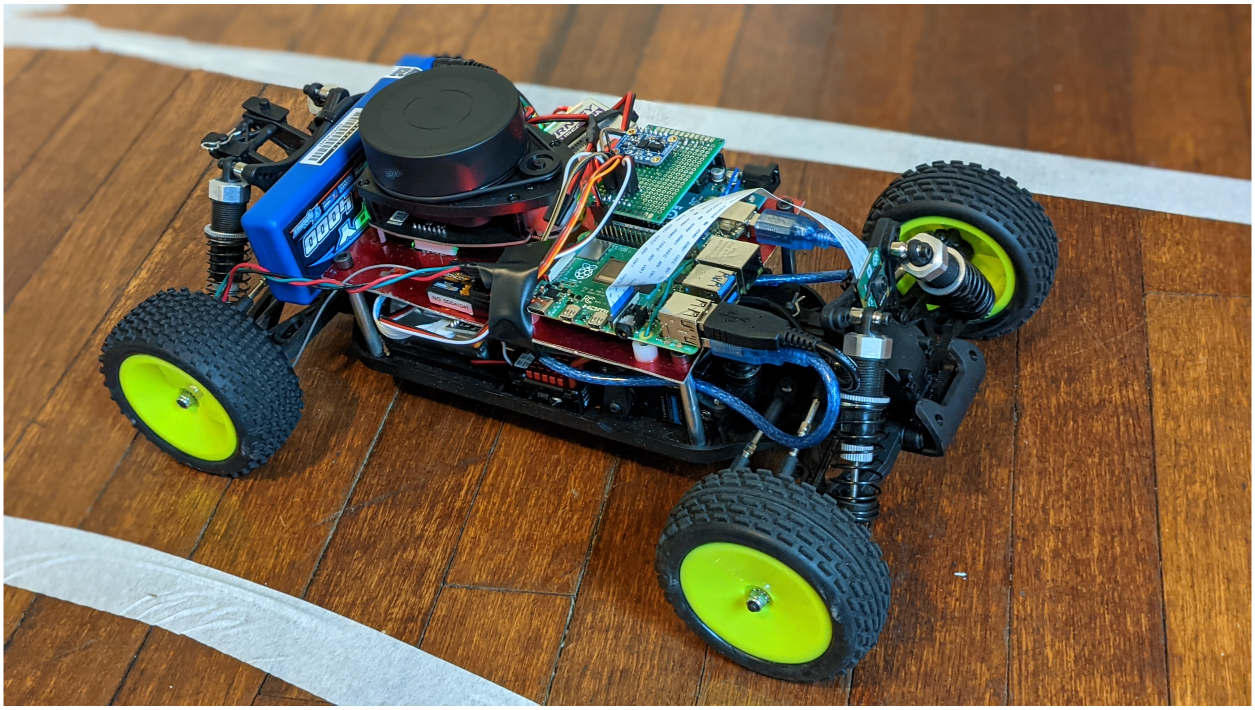


Fig. 5: Second side view of the robot.

Part	Name	Cost	I_{AVE}	P_{AVE} (W)
High-Level Controller	Raspberry Pi 4B 8GB	$75	1500mA	7.5W
Low-Level Controller	Arduino Uno R3	$22.50	50mA	250mW
LIDAR	RPILIDAR A1M8	$99	450mA	2,250mW
Camera	Raspberry Pi Camera Module V2	$25	133mA	440mW
IMU	Adafruit LSM6DS33 + LIS3MDL	$9.95	1.25mA	4.1mW
Chassis	Quanum Vandal 1/10	$113.55	–	–
Batteries	HRB 2S2P 7.4V 10000mah Lipo	$52 (x2)	500A (max)	3.7kW (max)
Power Distribution	UBEC DUO	$16.70	10mA	74mW
Miscellaneous	–	$100	–	–
TOTAL	–	$570.20	2.14A	10.52W

TABLE I: Parts list and budget. Note, miscellanous includes parts such as breadboards and wires, and chassis includes servo, motor, & ESC.

IV. Project Management

Based on the design discussed above, a list of parts, price budget, and power budget have been created. This information can be seen in table 1.

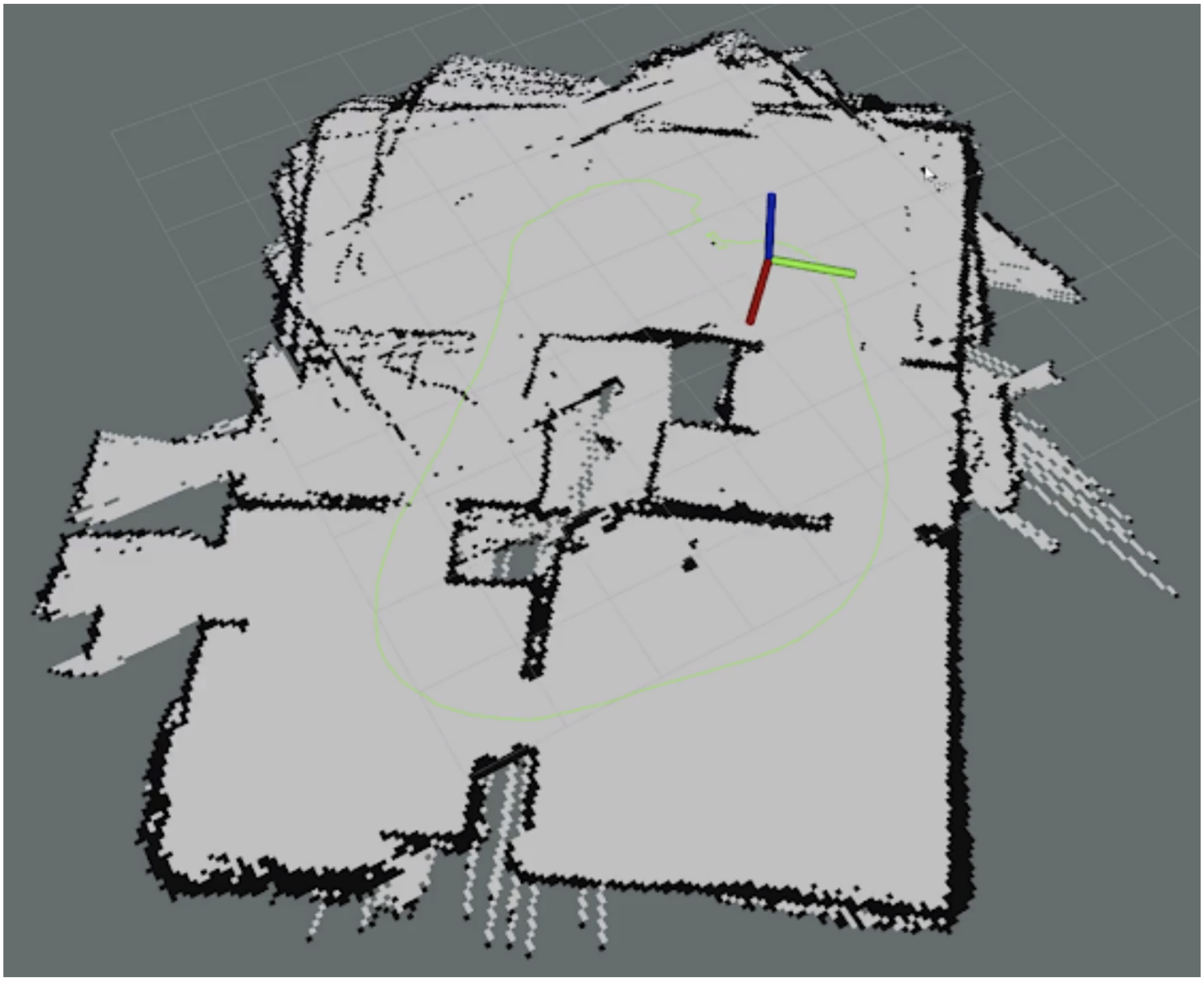


Fig. 6: Map of terrain surrounding the robot using the LIDAR and HectorSLAM.

From table 1, it is easy to see that the project is well within the $1000 budget. There is some leeway where if a device breaks or upgrades need to be made, they can be done by being below budget. The UBEC power distribution board allows for an average current draw of 4A. Table 1 shows that the robot's sensors and processors will remain at or below the board's current constraints.

The project plan and operation has primarily been focused on syncretic work, except for the CSE team, where all members attack each task and provide their feedback. Then the final design could then be chosen based on majority vote, which is often unanimous. Moving forward, the project has been split among the three group members.

The first member to be discussed is Brendan Sayers. Brendan Sayers has a background with hardware systems and, as such, will be in charge of the chassis, motor, and batteries. Sayers ordered and built the chassis, and installed the encoders onto the wheels.

Paulo Alcântara Silva was responsible for the low-level sensor subsystem. Alcântara wrote the initial calibration fro the IMU. Additionally, he was in charge of the team website.

The final group member, Paul Simmerling, was accountable for the high-level and low-level systems, including setting up ROS on the Raspberry Pi, testing the camera and LIDAR, writing the code for communication with the Pi and Arduino, coding the ROS nodes for Pi and the laptop, and developing the control code on the Arduino which included finishing the encoder and IMU codes. He was responsible for the deliverables, such as creating weekly meeting powerpoints, writing all papers, creating the poster, recording the videos, contacting the faculty advisor, and sponsor (for both CSE and ECE). Since he was involved with the high-level system, Simmerling worked with the CSE team on setting up the neural network on the robot and what sensors to use.

The Computer Science and Engineering team will design and write the code for the neural network. A RACI chart visualization can be seen in figure 6.

Project/Deliverable	Prof. Gupta	Dr. Berntorp	Paul S.	Paulo A.	Brendan S.	CSE
UConn Meetings	C	C	R	A	A	I
MERL Meetings	I	C	R	A	A	A
Website/Meeting Minutes	I	I	I	R	I	I
CAD Drawings	C	I	I	R	I	I
Final Deliverables	I	I	R	A	A	I
Sensors+Control Code	C	I	I	R	I	I
Chassis+Motors+Motor Code	C	I	I	I	R	I
RPi+Cameras+LIDAR System	C	C	R	I	I	C
Neural Network	I	C	A	I	I	R
Assemble Robot	I	I	I	I	R	I
Record Training Data	I	C	R	R	R	C

Responsible
R
Accountable
A
Consulted
C
Informed
I

Fig. 7: RACI chart for team 2107.

V. Summary

Autonomous vehicles are rapidly becoming an integral part of both future and current cars. The University of Connecticut Electrical and Computer Engineering Team 2107 designed and developed a new vehicle testbed that will process a new deep neural network for obstacle

avoidance and path planning in conjunction with a Computer Science and Engineering team. Mitsubishi Electric Research Laboratories sponsored this project. The final design takes a three-pronged approach. The first is a low-level system that used encoders, IMU, an Arduino, and Ackermann drive chassis to replicate a standard road vehicle's driving dynamics. The second prong is a medium-level system that provided immediate feedback from the LIDAR and camera to avoid collisions as well as had the SLAM processing. The final prong is an external laptop connected via wifi hosted on the Pi that will perform advanced neural network and image processing to provide path planning and obstacle avoidance.

The final design was highly successful in training. The device was capable of being tele-operated by a driver using nothing but the camera and the LIDAR. There are some steps that need to be made moving forward, the encoder system designed by Sayers is unable to record distances accurately and needs to be replaced by a commercial encoder. Moreover, the device has issues with connectivity to the laptop. To solve connection issues, an external router needs to be purchased and placed onto the robot. Finally, the device needs to record enough training data for the neural network.

Despite these few issues, the vehicle designed by team 2107 met the requirements requested by MERL. It can be easily driven using sensor data and a gamepad controller. With Robot Operating System, the device can process LIDAR and camera data and stream it to an external device also running ROS. Moreover, this data can be easily saved on this external device for the eventual training of a deep neural network such as the one designed by the CSE group.

REFERENCES

[1] K. Berntorp, M.-Y. Liu, and A. Weiss, "System and method for controlling vehicle using neural network," Jun 2018.

[2] M. Lynberg, "Automated vehicles for safety," Jun 2020. [Online]. Available: https://www.nhtsa.gov/technology-innovation/automated-vehicles-safety

[3] A. Jadhav, "Autonomous vehicle market size, share and analysis: Forecast 2026," May 2018. [Online]. Available: https://www.alliedmarketresearch.com/autonomous-vehicle-market

Appendix

Senior Design Project Checklist

Project name: 2107 Path Planning with Deep Neural Nets

Sponsor: Mitsubishi Electric Research Laboratories (MERL)

Team members (majors/programs): Paul Simmerling (Electrical Engineering, Physics), Paulo Alcântara (Electrical Engineering), Brendan Sayers (Electrical Engineering), Yun-Hwan Kim (Computer Science), Nathan Choi (Computer Science), Tim O'Reilly (Computer Science)

Faculty advisor(s): Professor Shalabh Gupta (ECE), Professor Fei Miao (CSE)

Skills:	
Analog circuit design and troubleshooting	✓
Digital circuit design and troubleshooting	✓
Software development/programming	✓
Embedded Systems/Microcontrollers	✓
Web design	✓
RF/wireless hardware	
Control systems	✓
Communication systems	
Power systems	✓
Signal processing	✓
Machine shop/mechanical design	✓
Other (please specify):	
Constraints:	
Economic (budget)	✓
Health/safety	✓
Manufacturability	
Environmental (e.g., toxic materials, fossil fuels)	
Social/legal (e.g., privacy)	
Standards:	
List standards/electric codes that you used (e.g., IEEE 802.11, Bluetooth, RS-232, VHDL, etc.)	IEEE 802.11, USB 2.0

TABLE II: Skills, Constraints, and Standards