**Hardware Design Document**

**for “*LaneSense: Autonomous Lane Detection RC Car*”**

***Date:* January 26, 2024**

**Name:** **Dominic Lau**

**For Course/Teacher:** **Mr.Roller (TEJ4M)**

**Table of Contents**

[1. Introduction 2](#_2et92p0)

[1.1 Project Overview 3](#)

[1.2 Firmware and Platforms 3](#_jpzn0ghcq6ms)

[1.3 Works Cited 4](#_kfkwf8bbbmym)

[2. Product Design 5](#)

[2.1 Changes 5](#)

[2.2 Block Diagram 5](#_m27biveygn73)

[2.3 Schematic 6](#)

[3. User’s Guide 6](#_98mrcdbamjpr)

[APPENDIX 7](#_z337ya)

[Appendix A. Beta Testing and Reporting Table 7](#_8pucqpz4qrj5)

[Functional and Hardware Requirements 7](#_2i21v982a4oj)

[User Interface Requirements 8](#_26in1rg)

[Internal Interface Requirements 9](#_tfg0oedlsl4l)

[External Interfaces Requirements 10](#_rqego812ggfq)

[Appendix B. Reflection of Learning 10](#_u0mwsdlasfo8)

[Appendix C. Code 12](#_s1bngm64fug4)

# Introduction

## ***Project Overview***

This project targets students (high school and college levels) and hobbyists interested in learning about robotics and computer vision. The Lane Detection RC Car is an educational and recreational project featuring a remote-controlled car with a camera for autonomous lane detection. Users can switch between autonomous and manual modes for learning and fun.

Please refer to the Hardware Requirements Specification and the Preliminary Design Document for more Information about the project.

## ***Firmware and Platforms***

| Name | Firmware or Platform | Use | URL |
| --- | --- | --- | --- |
| Arduino IDE | Platform | Used for programming the Arduino Uno. | https://www.arduino.cc/en/software |
| PlatfromIO (VScode) | Platform | Used for programming the Arduino Uno | https://platformio.org/?utm\_source=platformio&utm\_medium=piohome |
| TigerVNC | Platform | Used to virtually connect to the pi as the school wifi is bad. | https://tigervnc.org/ |
| Thonny | Platform | Used to code the pi 4 | https://thonny.org/ |
| OnShape | Platform | This platform was used to cad | https://www.onshape.com/en/ |
| Cura | Platform | Used to slice the stl files that was cadded on onshape for home printer. | https://ultimaker.com/software/ultimaker-cura/ |
| Dremel | Platform | Used to slice the stl files that was cadded on onshape for school printer. | https://www.dremel.com/gn/en/digilab/software |
| RH\_NRF24.h | Firmware | RF library used to control the transceiver modules | https://github.com/epsilonrt/RadioHead/tree/master |
| Servo.h | Firmware | Used for controlling the servo position and the speed of the brushless motor | https://github.com/arduino-libraries/Servo |
| OpenCV | Firmware | An external open source python library used for lane detection. | https://opencv.org/ |
| Picamera2 | Firmware | A python library to receive inputs from a Pi camera | https://datasheets.raspberrypi.com/camera/picamera2-manual.pdf |

## ***Works Cited***

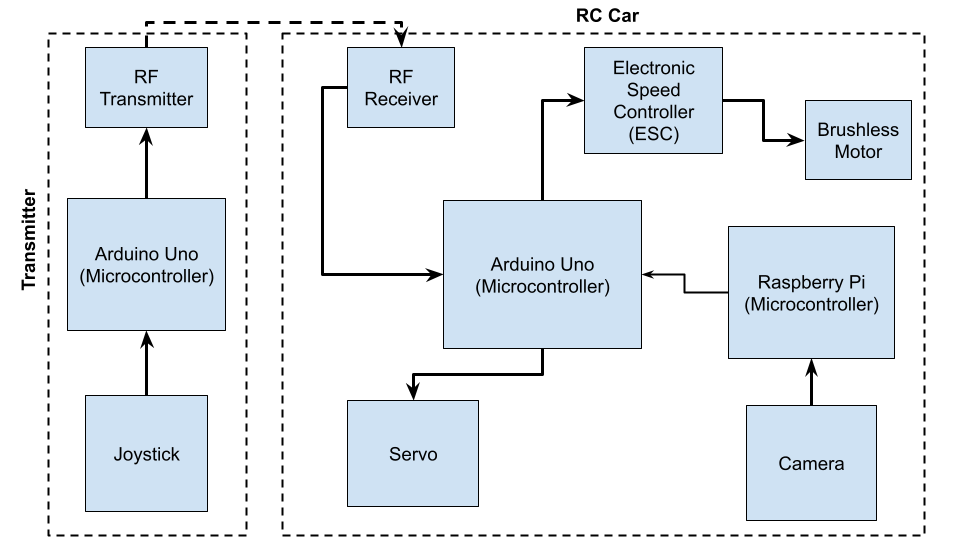
| Name | Description/Use | Author | URL |
| --- | --- | --- | --- |
| nRF24L01 – How It Works, Arduino Interface, Circuits, Codes | An article on how to set up the NRF24L01 modules. | Dejan | https://howtomechatronics.com/tutorials/arduino/arduino-wireless-communication-nrf24l01-tutorial/ |
| I PRINTED an RC Car... How FAST can it go?? | Used the CAD files for the development of the 3D printed RC Car | Michael Rechtin | [I PRINTED an RC Car... How FAST can it go??](https://www.youtube.com/watch?v=-pxNIrBaDrA) |
| Building a lane detection system | Used parts of the code to write the lane detection algorithm | Arun Purakkatt | https://medium.com/analytics-vidhya/building-a-lane-detection-system-f7a727c6694 |
| Chat GPT | Helps in understanding tutorial codes, features, and debugging. | OpenAi | https://openai.com/chatgpt |

# Product Design

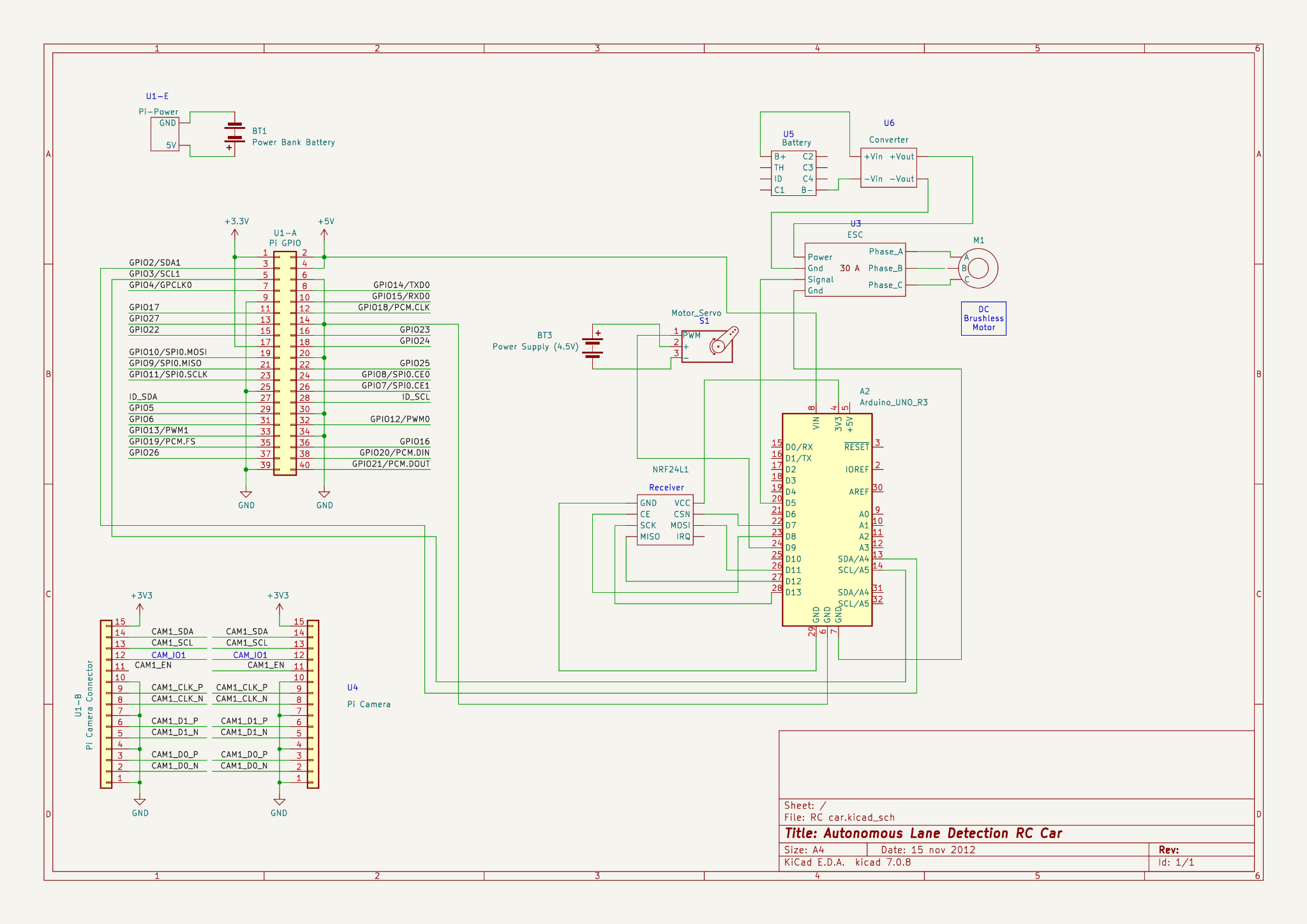
## ***Changes***

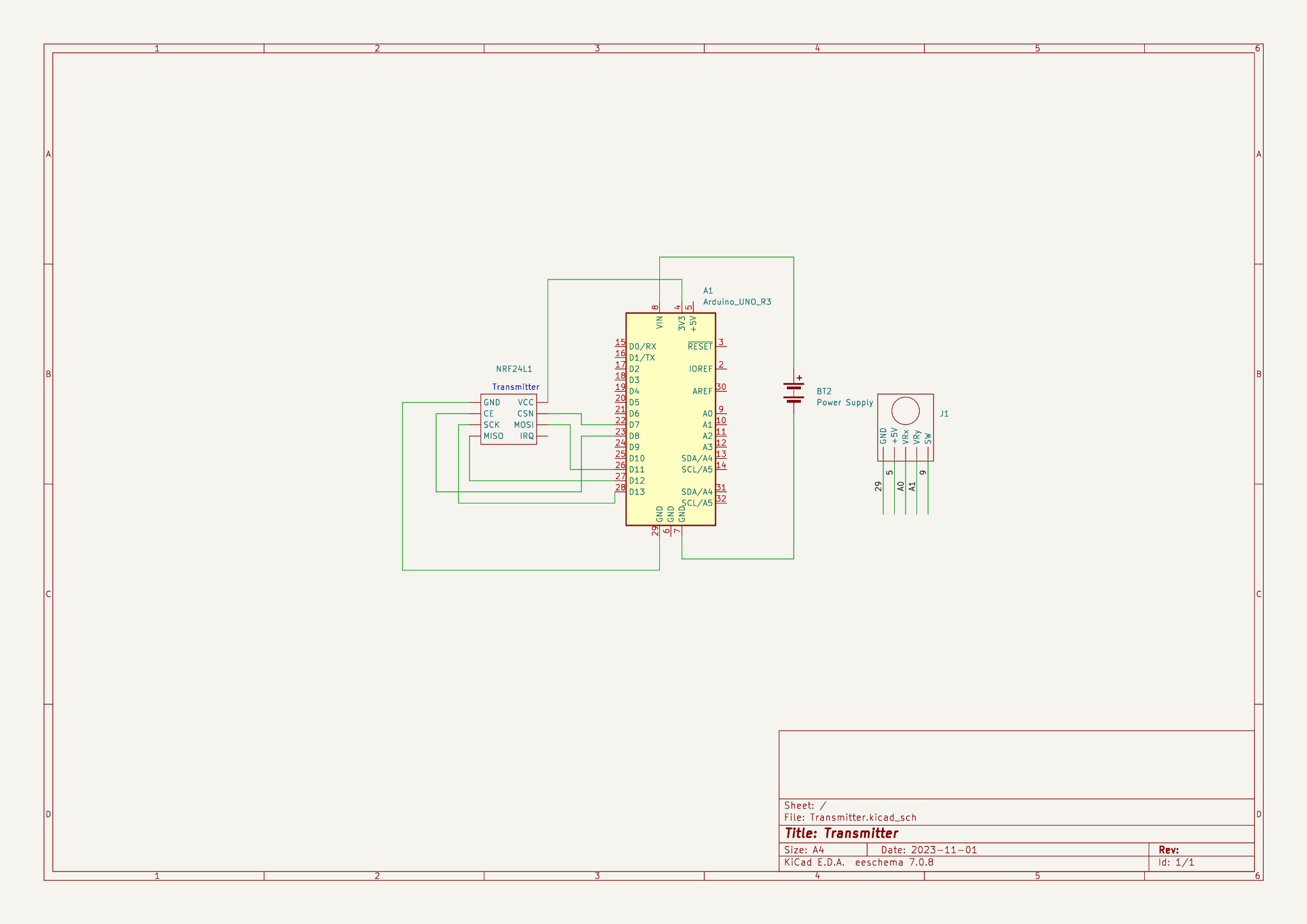
The design changes include the addition of an Arduino microprocessor on the RC car used to control the servo and speed of the ESC and receive data from the RF receiver. The Raspberry Pi is only used for running the lane detection algorithm. This change occurred because the Pi was laggy, and it is easier to work with the transmitter using libraries. A 4.5V battery (3 AA batteries) was added to power the servo motor. (For more in-depth development details, read the 1-9 Journal Entries).

## ***Block Diagram***



## ***Schematic***





# User’s Guide

* 1. Plug in all the batteries (power bank, drill battery, and 3 AA batteries).
  2. Using the command line in the terminal, connect to the Raspberry Pi through SSH.
  3. Type "python3 lane\_detection.py" in the terminal to start the lane detection algorithm.
  4. Plug in the Arduino transmitter through USB.
  5. Start controlling the car using the joystick (note that the RC car cannot go backwards).
  6. To switch from user control to autonomous mode, press the joystick, and vice versa.

# APPENDIX

## Beta Testing and Reporting Table

### ***Functional and Hardware Requirements***

| **Req#** | **Requirement** | **Hardware Components Involved in Testing (with**  **Schematic Reference Designators)** | **Status**  **(Pass/Fail or**  **Pass/Fail with explanation)** | **Date Tested** |
| --- | --- | --- | --- | --- |
| LS-3.1.2 | The RC car shall have the ability to switch between autonomous mode (lane following) and manual mode (user-controlled). | A1 Arduino\_Uno\_R3  NRF24L1 Transmitter  J1 Joystick  A2 Arduino\_Uno\_R3  NRF24L1 Receiver | Pass. By pressing a button, the RC car switches between manual mode and autonomous mode, and vice versa. | Jan 24, 2024 |
| LS-3.1.3 | The RC car shall have most of the chassis 3D printed. | N/A | Pass. The RC car chassis is 95% built out of 3D printed PLA. The tires were made with 3D printed TPU. | Jan 10, 2024 |
| LS-3.1.4 | The RC car shall be equipped with one brushless motor to drive the drivetrain, providing sufficient power and control. | M1 - Brushless Motor | Pass. The RC car uses one brushless motor for sufficient driving power. | Jan 24, 2024 |
| LS-3.1.5 | It shall be powered by a Raspberry Pi 2 to control all electronics within the RC car, including sensor data processing and communication. | U1 - Raspberry Pi 4  A2 Arduino\_Uno\_R3 | Modification. The Raspberry Pi 2 has been replaced with a Raspberry Pi 4 and an Arduino Uno. The Arduino Uno communicates with all the electronics except for the Pi camera. | Jan 24, 2024 |
| LS-3.1.6 | It shall have a transmitter that uses an Arduino and RF transmitter. | A1 Arduino\_Uno\_R3  NRF24L1 Transmitter | Pass. Fully functional RF module with Arduino Uno. | Jan 24, 2024 |
| LS-3.1.7 | It shall have a servo that steers the car. | S1 - Servo  A1 Arduino\_Uno\_R3 | Pass. The Arduino Uno communicates with the servo to adjust the steering of the car. | Jan 24, 2024 |

## 

### ***User Interface Requirements***

| **Req#** | **Requirement** | **Hardware Components Involved in Testing (with**  **Schematic Reference Designators)** | **Status**  **(Pass/Fail or**  **Pass/Fail with explanation)** | **Date Tested** |
| --- | --- | --- | --- | --- |
| LS-3.2.1 | The product shall have an autonomous mode, allowing it to operate without direct user intervention. | U1 - Raspberry Pi 4  U4 - Pi Camera | Fail. The user needs to manually push the car because turning on the brushless motor caused the camera feed to become laggy. | Jan 24, 2024 |
| LS-3.2.2 | Users shall have the capability to remotely control the product using a designated interface or device. | J1 - Joystick | Pass. The user can use the joystick to remotely control the RC car without being physically connected to the car. | Jan 24, 2024 |
| LS-3.2.3 | The product shall provide a manual control mode, enabling users to control its movements directly. | A1 Arduino\_Uno\_R3  NRF24L1 Transmitter  J1 Joystick  A2 Arduino\_Uno\_R3  NRF24L1 Receiver | Pass. The RC car can be controlled manually. | Jan 24, 2024 |
| LS-3.2.4 | The user interface shall be designed to be intuitive and user-friendly, ensuring ease of operation for users of varying technical backgrounds. | J1 Joystick | Pass. The use of joystick allows for an intuitive and user friendly interface. | Jan 24, 2024 |
| LS-3.2.5 | Users shall have the ability to switch between autonomous mode, manual control mode, and any other relevant modes with the push of a button. | J1 Joystick | Pass. The integrated momentary button is used to switch between modes. | Jan 24, 2024 |

### ***Internal Interface Requirements***

| **Req#** | **Requirement** | **Hardware Components Involved in Testing (with**  **Schematic Reference Designators)** | **Status**  **(Pass/Fail or**  **Pass/Fail with explanation)** | **Date Tested** |
| --- | --- | --- | --- | --- |
| 3.3.1.1 | The microcontroller shall communicate with the Electronic Speed Controller (ESC). | U3 - ESC  A2 Arduino\_Uno\_R3 | Pass. The Arduino communicates with and sets the speed of the Electronic Speed Controller. | Jan 24, 2024 |
| 3.3.1.2 | The Electronic Speed Controller shall control the DC motor using PWM (Pulse Width Modulation) signals that are connected to the wheels. | U3 - ESC  M1 - Brushless Motor | Pass. The ESC can effectively change the speed of the Brushless Motor. | Jan 24, 2024 |
| 3.3.1.3 | The Arduino (microcontroller) shall communicate with the transmitter to transmit information | A1 Arduino\_Uno\_R3  NRF24L1 Transmitter | Pass. The Arduino can communicate with the transmitter to send x and y position of the joystick. | Jan 24, 2024 |

### ***External Interfaces Requirements***

| **Req#** | **Requirement** | **Hardware Components Involved in Testing (with**  **Schematic Reference Designators)** | **Status**  **(Pass/Fail or**  **Pass/Fail with explanation)** | **Date Tested** |
| --- | --- | --- | --- | --- |
| 3.3.2.1 | The camera module shall communicate with the microcontroller to provide video data for lane detection. | U1 - Raspberry Pi 4  U4 - Pi camera | Pass. The Pi camera provides sufficient data for lane detection. | Jan 24, 2024 |
| 3.3.2.2 | The Arduino shall read the inputs from the joystick to transmit to the receiver. | A2 Arduino\_Uno\_R3  NRF24L1 Receiver | Pass. The Arduino can read inputs from the receiver. | Jan 24, 2024 |
| 3.3.2.3 | The receiver shall receive inputs from the transmitter and relay it back to the Raspberry Pi. | A2 Arduino\_Uno\_R3  NRF24L1 Receiver | Pass with modification. The receiver receives information and sends it to the Arduino. | Jan 24, 2024 |

## Reflection of Learning

This semester's course immersed me in hardware development, allowing me to explore the entire development process. As I look back on this journey, here are my key learnings and insights:

Expectations:

* Initially, I anticipated a technical exploration into the nuances of hardware design, but the course exceeded my expectations by integrating not only technical aspects but also project management, scheduling, testing, and product demonstration.

Course Enjoyment:

* The most fulfilling aspect was seeing it function. Witnessing the evolution of a project from idea to reality provided a sense of accomplishment.

Course Challenges:

* Unforeseen technical challenges and tight schedules presented difficulties. However, these challenges pointed towards the importance of planning and adaptability.

Product Development:

* Did Not Go Well: I think that I was very annoyed with the building process of the car. It took much longer than expected.

Process Improvements:

* Could Have Done Differently: Reflecting on the process, I definitely could have scheduled better especially for the building portion.

**What I Learned:**

* Brushless Motors:
  + I gained in-depth knowledge about brushless motors and learned how to effectively control them using Arduino. This involved understanding the nuances of power supply, potentiometer adjustments, and addressing issues related to responsiveness and torque.
* 3D Modelling and Printing:
  + Continuing my 3D modelling development, I started on another platform, Onshape. I gained expertise in modifying, converting, and exporting 3D models into printable G-code files. This showcased my versatility with materials like PLA and TPU. I also improved 3D print quality by mastering printer settings and troubleshooting common issues such as stringing.
* Design Process:
  + I learned the importance of a structured design process by creating block diagrams, schematics, hardware requirement specifications, design documents, and timelines. This approach ensures efficient project management, clear communication, and sets a solid foundation for development, mirroring professional practices in computer and hardware engineering.
* Microcontrollers (Arduino):
  + I learned the more about microcontrollers, particularly Arduino, distinguishing between digital and analog inputs and outputs. I also gained hands-on experience in finding and installing specific libraries for driving servos and RF modules, showcasing practical skills in microcontroller programming.
* OpenCV and Visual Recognition:
  + Through online tutorials on platforms like YouTube and GitHub, I implemented various visual recognition algorithms using OpenCV. This exploration provided insights into the capabilities of image processing tools, offering a glimpse into the future ahead of us.
* Programming Languages:
  + I extensively used Python for its ease of use and access to libraries, accelerating the development process. Additionally, I practised C++ as the programming language for the Arduino board, showcasing my versatility in coding languages relevant to hardware projects.
* Computer System Understanding:
  + I utilized tools like SSH and VNC, gaining a profound understanding of how computer systems function.

Preparation for the Future:

* The course's comprehensive nature, combining technical skills and project management, has prepared me for post-secondary education and the workforce. The logs showcase the development of skills crucial for future educational pursuits and professional roles. It also enhances **soft** **skills** including problem solving, time management, and communication skills.

Comparison with Industry:

* While the course simulated a real-world hardware development environment, I recognize that industry experiences may involve more extensive collaboration, larger project scales, and exposure to a broader range of technologies. The industry setting could demand a higher degree of specialization, emphasizing the need for continual learning.

## Code

[Github Link](https://github.com/DominicTYLau/Self-Centring-RC-Car)