4 Design

4.1 Design Content

For our project, we need to design a RC car that will semi-autonomously navigate through a track with obstacles and hacking. The driver of the RC car will have access to increase or decrease the speed of the vehicle but have no control over the direction the vehicle goes in. The direction will be changed based on a sensor on the front of the bot that detects where there is an obstacle and where a path is available for the bot to travel through. There will also be sensors on the side of the bot that will detect the borders of the track. The border will consist of a specific color of tape that can be detected by a simple photoelectric sensor. The bot will have a processor on board that will compute everything related to detecting obstacles, borders, and deformities. This processor will tentatively be an Aurdino until we have found a better solution.

What we need based off of requirements:

- A way to detect the borders of the track
- A way to detect objects in front of us
- A way to detect deformities in the track, for example: Holes, bumps, craters
- Steering is handled autonomously
- User controls speed of bot
- Protections against remote tampering
- Protections against potential physical hazards

Track team must guarantee:

- Car must stay flat to the track (can have ramps, banked turns, etc). The bot shouldn't be flipped from the track having a split levels
- The car shouldn't be severely damaged by obstacles

4.2 Design Complexity

This project meets the requirements for technical complexity as it consists of multiple components and subsystems that work together using distinct scientific and engineering principles. Our end product is a partially-autonomous RC car that can quickly navigate through a hostile environment to reach a goal. In order to accomplish this, we will need to modify an existing RC car, taking out its main control components and replacing them with a customized Arduino board. The only communication between our remote control and the car will be control of acceleration, which will be handled by the preexisting radio communication. The car itself will use LIDAR and photoelectric sensors to identify obstacles and remain on the track, feeding this data to the onboard systems that will make the decisions on when and where to turn the vehicle. These decisions will need to be programmed into the board, meaning our group will have to make use of an embedded coding language. Beyond this basic maneuverability, our bot will need to be durable enough to survive any of the obstacles included as part of the track design. Our car must be capable of navigating sharp turns, vertical changes in terrain, moving around holes, and avoiding other obstacles designed to damage the vehicle. This means we will need to redesign the shell and suspension of the car, implementing mechanical principles to minimize the damage of impacts and maximizing the chance of recovery from a flipped position.

4.3 Modern Engineering Tools

What modern engineering tools were used for this design? Their roles.

Embedded Programming Language: For coding the Arduino onboard processor, we plan to use the Arduino IDE. This software allows us to write code in a simplified version of C and C++. It's perfect for controlling the RC car's behavior, processing sensor data, and making autonomous decisions.

Sensor Data Processing: We'll be relying on the Arduino IDE to process data from our sensors, such as LIDAR and photoelectric sensors. This is where we write code in C/C++ to interpret data and make real-time decisions about obstacle detection and path planning.

Remote Control Interface: To create a user interface for remote control of the car's speed, we intend to use programming languages like Python or JavaScript. These languages are great for building a user-friendly control interface that communicates with the car through wireless communication methods.

Cybersecurity Implementation: Implementing cybersecurity measures to protect against remote tampering will be a crucial part of our project. We'll incorporate encryption and authentication protocols within our Arduino code to ensure the security of the communication.

Simulation and Testing Software: To test our algorithms and simulate the RC car's behavior before deploying it in the real world, we're considering using software tools like ROS (Robot Operating System) or simulation environments like Gazebo. These tools will help us iron out any issues before putting the car on the track.

Data Visualization and Debugging: For visualizing and debugging sensor data, we're looking at using software like MATLAB or Python with libraries like Matplotlib. These tools will help us analyze sensor data and identify any issues in the system.

Version Control: We'll use Git for version control. It's a great tool for managing code revisions, tracking changes, and merging contributions from our team members, ensuring a smooth collaborative development process.

Programming Environment: The Arduino IDE will be our primary programming environment for the Arduino onboard processor. We may also use text editors like Visual Studio Code or Sublime Text for writing and editing code, as they offer useful features like syntax highlighting and code completion.

Algorithm Development: If we create custom algorithms for obstacle detection and path planning, we'll use C/C++ within the Arduino IDE. We may also integrate additional libraries or external tools as needed to enhance our algorithms.

Firmware Updates: When it's time to update the firmware on the Arduino board or other microcontrollers, we'll use the Arduino IDE. This process involves uploading new code to the hardware components to implement improvements and updates.

Hardware: Things like the sensors being used for data collection, battery/power source, signal transmission between RC car and controller, and the prebuilt RC car to modify haven't been decided yet. The list below mentions the main components we will probably need once the criteria, requirements, and funding are finalized..

- Sensors: We will need distance sensors and light sensors at the very least for measuring distance and steering..
- Motors: We may need to replace the one that comes with the RC car already. Also, mounting a sensors on a servo motor will help us with data collection
- 3D printer: Designing the skeleton of the RC car and any of the physical components that may need to be remodeled inside and outside of the RC car
- Signal Connector: We will need a way for the controller to connect to the RC car without any interference from the track team
- Battery: May need to replace the battery depending on the power requirements of the added components
- Basic components and connectors: Just things like wires, cables, screws, etc, that will be needed for assembly and connecting devices to each other

4.4 Design Context

Area	Description	Examples
Public health, safety, and welfare	In our evolving society, self-driving cars and autonomous vehicles are becoming a reality more and more every day. With this emerging technology, creating additional test situations such as our senior design project can help find reliable solutions for public safety and welfare by minimizing hacking attempts and malfunctions.	Further developing knowledge in the world of automotive vehicles, reducing risks of malfunction and/or hacking from third parties.
Global, cultural, and social	At the moment, self-driving vehicles and the technology behind them is only available to a select few, such as the upper class and possibly farmers when it comes to automated farming.	Development of this test is smaller scale to reduce risks of harming humans, animals, or the environment.
Environmental	Our project will not use any gases or fossil fuels in general, and will be run 100% on electricity. Utilizing the least amount of energy for optimization is a priority.	No usage of fossil fuels, minimizing the amount of electricity used when the car is running is essential.
Economic	Even on our small scale, it is likely this experiment will cost at least $1000. Reasons like this is why automated vehicles are still in their infancy, so we will try to cut down on these costs as much as possible without sacrificing safety features.	Minimizing cost allows for further development of automated vehicles heading in a consumer-level market, but all safety requirements must still be met.

4.5 Prior Work/Solutions

Include relevant background/literature review for the project

– If similar products exist in the market, describe what has already been done

– If you are following previous work, cite that and discuss the **advantages/shortcomings**

– Note that while you are not expected to "compete" with other existing products / research groups, you should be able to differentiate your project from what is available. Thus, provide a list of pros and cons of your target solution compared to all other related products/systems.

Detail any similar products or research done on this topic previously. Please cite your sources and include them in your references. All figures must be captioned and referenced in your text.

When it comes to autonomous vehicles, a few clear examples come to mind. The recent rise of Tesla's self-driving cars is likely the most well-known example, who's engineers are attempting to allow fully automatic driving in any real-world scenario. Although Tesla brings forward some of the most advanced self-driving technology to the table, there are still plenty of bugs that prevent drivers from completely relying on the functionality of this software.

- Pros: Ideas for navigating a car on roads, knowledge how sensors could be used to keep the car moving in a straight line, examples for visualizing obstacles in a computer environment.
- Cons: High budget like the ones used for Teslas is not achievable with our funding, comes from a large-scale model where constraints are likely different.

If we were to scale down the high aspirations of Tesla, another example of a group attempting to develop autonomous vehicles could be the show BattleBots. Here, groups compete against each other to use sensors and traps to fight their opposing vehicle. Our project is very similar to this example, where we must race around a track as fast as possible. We will be able to use ideas from a show like BattleBots and transform them into our own designs for a race. However, we must also keep in mind budget is a large factor for a project like this, and we likely do not have the same funding as groups in the BattleBots show.

- Pros: Applications for modding/creating RC cars has been explored a bit, observable results from the show are easy to come across, sensors are similar to ones we may be using.
- Cons: Objective of these RC cars is not the same as the project we have been handed, budget is likely not as high as those observed in the show.

4.6 Design Decisions

List key design decisions (at least three) that you have made or will need to make in relation to your proposed solution. These can include, but are not limited to, materials, subsystems, physical components, sensors/chips/devices, physical layout, features, etc.

- Car selection: What kind/model of RC car would we like to modify? In turn, how much work will we need to put in when it comes to adding external sensors and software?

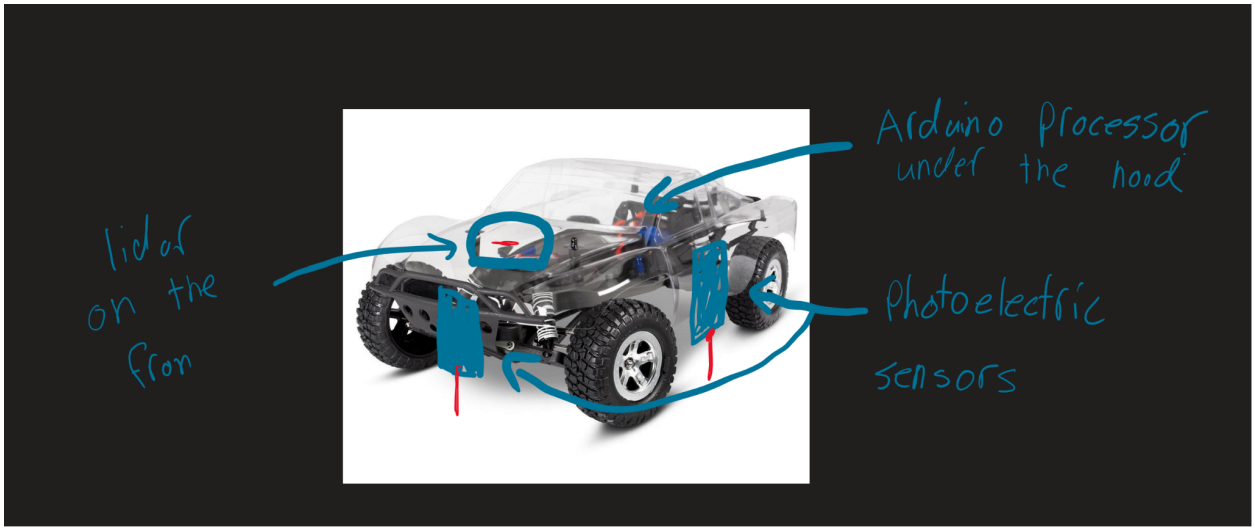
- Sensor selection: What kind of sensors would we like to have on our car? Are there any specific obstacles where a special type of sensor could come in handy? How easily can said sensors be connected to our Arduino to be integrated into our software design?
- Computer connection: What software would we need to connect our RC car and computer? Are there any limitations to this software? What programming languages (C, Java, Python, etc.) are we going to use for both connection and software development?

4.7 Proposed Design

Our team has begun to look into possible RC cars to buy. We've communicated with the other RC team what they are looking into buying but neither team has made any decisions. We've also begun looking into what processor we want to use. We've been thinking about using an arduino as our processor as it's cheap and effective. We've also had discussion on what sort of sensors will be on the bot, Lidar, photoelectric sensors, etc.

4.7.1 Design 0 (Initial Design)

Design Visual and Description



Here we have the very simple mock up of our car's design. We haven't decided on specific parts yet, but as a base line we have the following:

- Lidar on the front of the bot for object detection
    - This will alow the bot to detect obstacles on the track
- Photoelectric sensors on both sides of the bot as well as the front of the car
    - detects the borders of the track or holes
- Our processor will be hidden away underneath the hood of the car in order to protect it from damages
    - There are thoughts of incasing the processor with some sort of protection against crashing
- The possibility for other sensors are still being considered like an IR sensor for example

Functionality

Our user will only be allowed to control the car's acceleration. All the turning and manuevering will be completely controlled by the microcontroller inside of the RC's body. This current design shows where the possible components will be placed onto the car but lacks many details for a functional requirements.

4.7.2 Design 1 (Design Iteration)

Our most updated design is found in section 4.7. We have not been able to get farther then what was previously stated

Design Visual and Description

Include a visual depiction of this design as well highlighting changes from Design 0. Describe these changes in detail. Justify them with respect to requirements.

There are no new details to discuss.

4.8 Technology Considerations

We should have access to a many different ways to code our microcontroller. Were hoping for an easy way to hijack the RC car to feed data from the processor to the car's motors.

4.9 Design Analysis

We have not done an analysis of our design yet. We'd like out next design to be more detailed.