Race Of Doom

IRP Presentation

Developers:

- Peter Wissman - CprE
- Gavin Petrak - CprE
- Jacob Nedder - CybE
- Andrew Kraft - EE
- Jack Doe - SE

Introduction - What is Race of Doom?

- **Project Goal:**
  - Designing an autonomous vehicle for competitive racing on a track filled with obstacles and hacking nodes.
- **Key Components:**
  - Autonomous Steering: Onboard sensors for obstacle detection and avoidance.
  - Remote Speed Control: Vehicle speed controlled remotely.
- **Expected Deliverables:**
  - Track with obstacles and hacking nodes.
  - Two autonomous vehicles with remote control capabilities and obstacle detection sensors.
- **Design Process:**
  - Flexible and open-ended design process allowing students to construct their own autonomous RC car within the allocated budget.

Introduction - Why 3 teams?

- **Track Design Team:**
  - Ensures the creation of an engaging and challenging track for the race.
- **Vehicle Design Teams:**
  - Team A: Utilizes a cheap RC car with expensive components.
  - Team B: Utilizes an expensive RC car with cheap components.
- **Comparative Analysis:**
  - Performance: Evaluate the speed and maneuverability of each vehicle design on the track.
  - Cost-effectiveness: Analyze the cost versus performance trade-offs between the two vehicle design approaches.
- **Key Benefits:**
  - **In-depth Comparison:** Allows for a thorough comparison of different design strategies.
  - **Insights for Future Projects:** Provides valuable insights into cost-effective design and performance optimization.
- **Outcome Evaluation:**
  - Compare results between vehicle design teams and track design outcomes to assess project success and identify areas for improvement.

Introduction - Purpose of the Project

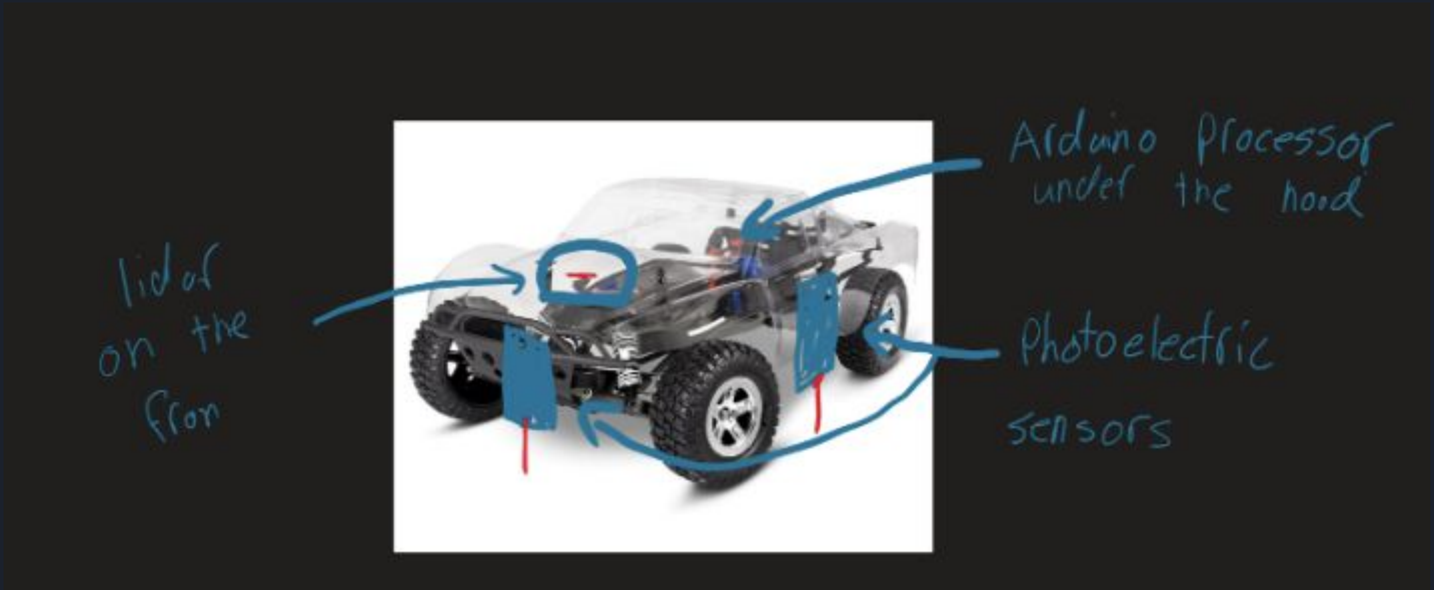
- **Objective:**
  - Designing autonomous vehicles for competitive racing in the Race of Doom.
- **Focus on Autonomous Car Technology:**
  - Harnessing the latest advancements in autonomous technology for real-world application.
  - Emphasizing the integration of onboard sensors for obstacle detection and autonomous steering.
- **Significance of Race of Doom Teams:**
  - Fostering collaboration and teamwork among students across different disciplines.
  - Providing a platform for students to apply theoretical knowledge to practical, hands-on projects.
- **Building for the Future:**
  - Establishing a foundation for future Race of Doom teams to iterate upon and improve.
  - Learning from mistakes and successes to inform future iterations and advancements in autonomous vehicle design.

Introduction - Requirements

- The project must stay within the given CPR E program's budget
- The RC Car shall autonomously steer away from obstacles on the track
- The RC Car shall stay within the bounds of the track
- The RC Car shall be protected from the track-hacking source
- The Driver shall control only the speed and acceleration of the RC Car
- The RC Car shall make multiple laps around the track

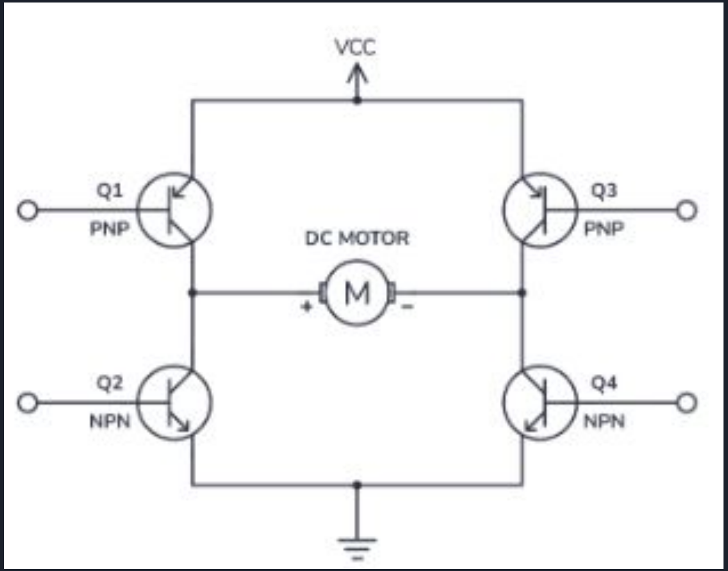
Design - Initial Design

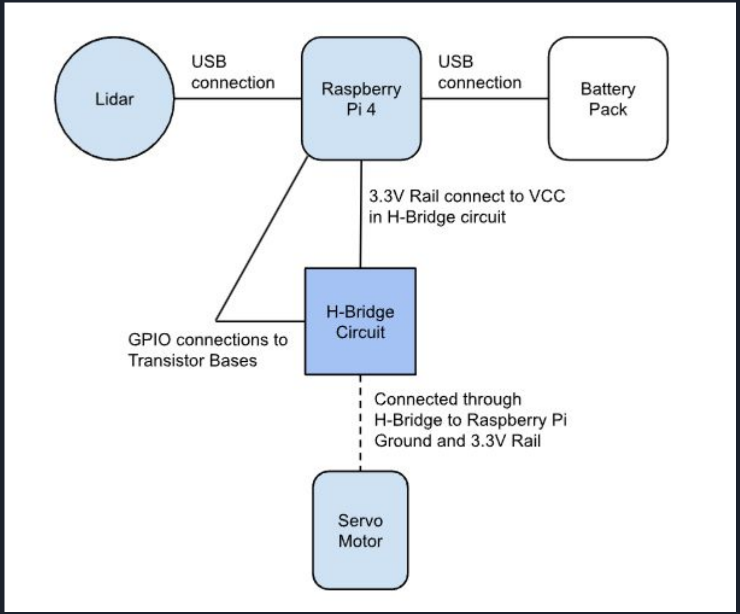
- Large car body with mounted sensors
- LiDAR for object detection, photoelectric sensors for staying in bounds
- Initial plan used an Arduino for automation
- Original code base in Python
- Design ultimately downscaled to match reduced size of final car body



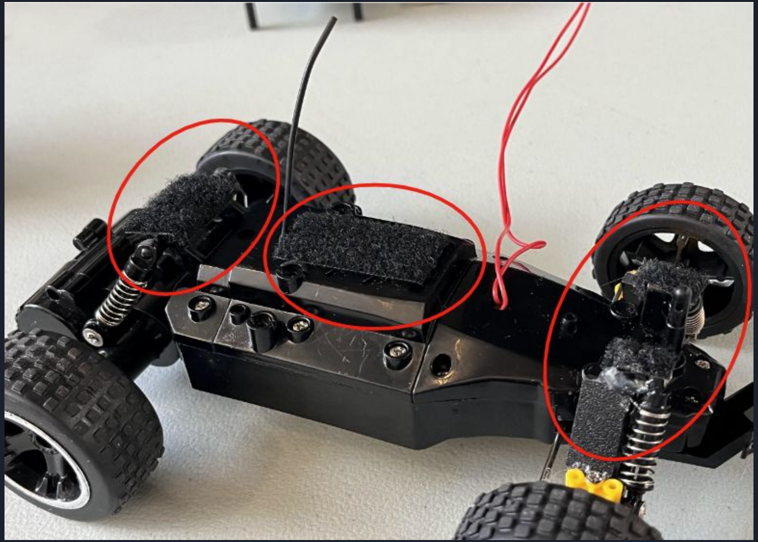
Design - Hardware - Electrical Components

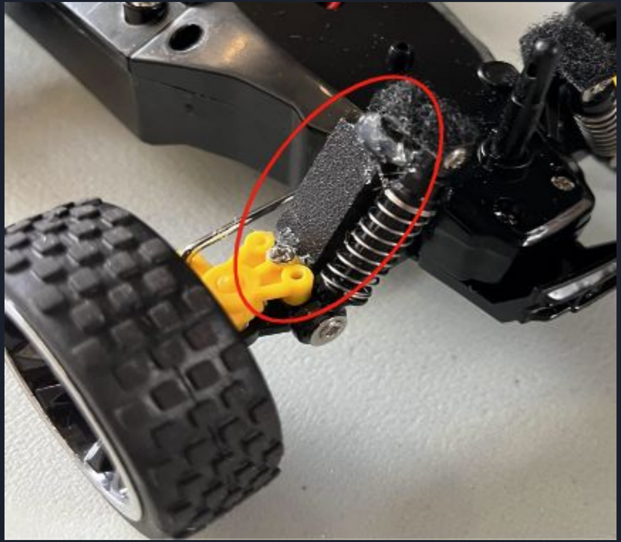
- Simple H-Bridge Circuit using NPN's and PNP's
- Power Systems Diagram
- Driving motor remains untouched

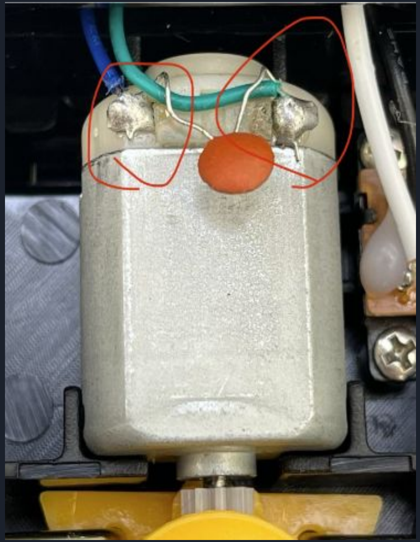




Design - Hardware - Physical Components







Design - Software

```
// continue straight
gpioWrite(17, 0);
gpioWrite(27, 0);
gpioWrite(13, 0);
gpioWrite(19, 0);
printf("Straight into the wall!\n");
```

- Initially used Python, switched to C++
  - Compatibility with LiDAR, GPIO, headers
  - LiDAR USB connector provided C++ headers
- LiDAR scans provided valuable information to use in code
  - Object distance, angle, and quality
- Created fields that determine autonomous steering based off scans
  - Used a weighting system to formulate decisions
  - Depending on environment, continue straight, left, or right
- Characteristics of code fields based on track constraints
  - “Critical Range” - 40 degrees directly in front of the car
  - Laid out special cases for track obstacles

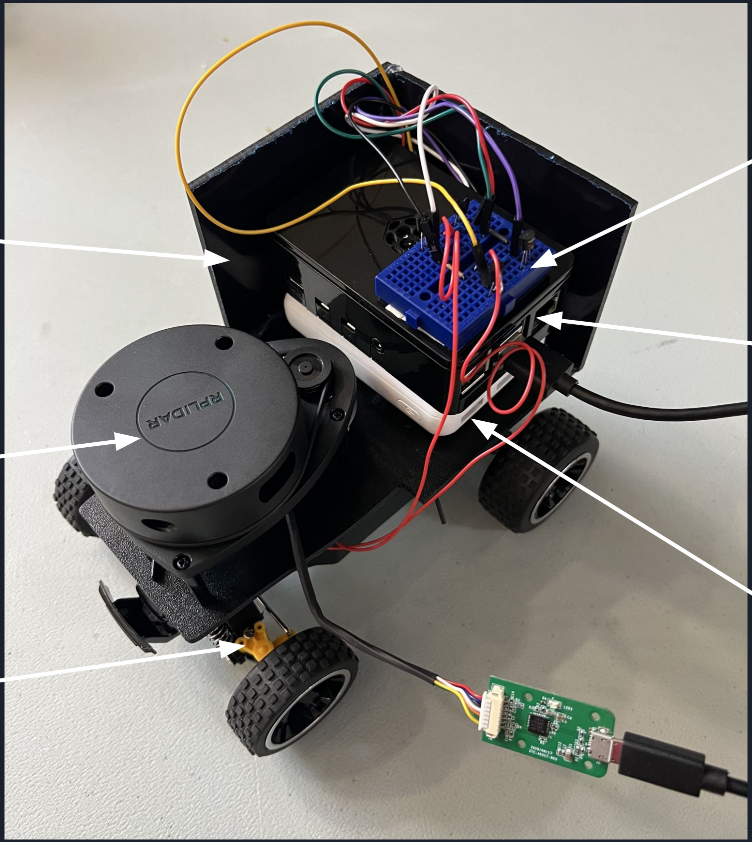
```
int rightWeight = 0;
int leftWeight = 0;
int rampWeight = 0;
```

Design - Final Design

ABS plastic protective shell cut and glued in place

LiDAR sensor connected through daughter board

RC car base with reinforced suspension



Breadboard with H-bridge turning circuit

Raspberry Pi 4 with wiring running from GPIO ports to Breadboard

External battery pack to power Raspberry Pi

Testing - Hardware

- Circuit Components tested by:
  a. Finding all power requirements for devices
     - Transistors, LiDAR, Raspberry Pi, Servo Motor
  b. Figuring out what circuit design works best
     - H-Bridge
  c. Testing using Multimeter and Voltmeter
     - Initial testing substituted LED's for turning, and voltmeter to power transistors and Servo
  d. Placing final components into circuit
- Physical Components tested by applying force with hand or outside object
  a. Suspension Upgrades

Before Results	After Results
Top Speed:1.5 m/sec	Top Speed: 0.78 m/sec
Turning Radius: 1.143 m	Turning Radius: 1.27 m
Total Weight: 689.46 grams	Total Weight: 1176.77 grams

Testing - Software

```
* Placeholder Values for Distance:
*
* Ground < 1000 away.
* Wall < 500 away.
* Angle: 300 - 360, 0 - 60
* Critical Wall Angle: 340 - 360, 0 - 20
*
```

- Initial iterative testing to determine LiDAR Ranges
- Tests to verify correct interface with GPIO pins
- Most testing was integrated testing of whole RC car
  - Senior Design Lab basic turning
  - Day before race track testing

```
if ((((nodes[pos].angle_z_q14 * 90.f) / 16384.f) > 340)) {
  // Critical wall angle to the right.
```

```
            } else if (((nodes[pos].angle_z_q14 * 90.f) / 16384.f) <
20) {        // Critical wall angle to the left.
```

Conclusion - Final Testing/Demonstration Outcome

- **Final Testing**
  - Final testing revealed difficulty driving across bubbles in track material
  - RC car provided insufficient power to drive up ramp
  - Efficiently navigates track and Faraday cage, avoiding walls
  - Performance overall exceeded expectations
- **Demonstration Outcome**
  - Final Demo lap times:
    - Run 1: 220 sec lap + 60 sec violations = 280 sec total time
    - Run 2: 113 sec lap + 35 sec violations = 148 sec total time
    - Run 3: 135 sec lap + 60 sec violations = 195 sec total time
  - Irregularity of the track cause acceleration issues
  - Issues with card not turning

Conclusion - Design Flaws

- **Small RC car body led to several issues**
  - Insufficient engine power to drive up ramp
  - Additional components added enough weight to heavily impact vehicle performance on uneven terrain
  - Highly vertical design leads to occasional issues with overhanging track elements
- **Code has difficulty differentiating walls and full-track-length obstacles**
  - Ex. the entrance to the Faraday cage
  - Ex. the ramp

Conclusion - Future Iterations

- Upgrade the RC car
    - Larger Size for more components
    - More Powerful motors and better steering
    - More space for protecting the components
- Add more components for object detection
    - Ping sensor to cross check our lidar results.
    - Accelerometer for detecting when we reverse
    - Some sort of color detection
        - Could be used in tandem with the track team to know when an obstacle will show up
- Full Autonomy

Conclusion - The Future of Race of Doom

- Plans to continue Race of Doom for future senior design projects
- New teams can make their own decisions based on the project results
  - Avoiding Design Flaws
  - Enhance the code that was used
- More iterations of the project ⇒ Better Requirements and direction
  - Project has been very open to interpretation

Thank you for your time!

Questions?