RACE: Remote Access Control for Electric Cars

Austin Silva, Michael Mallette, Jesvany Colon, and Sean Chaney

Dept. of Electrical Engineering and Computer Science, University of Central Florida, Orlando, Florida, 32816-2450

***Abstract* — The RACE (Remote Access Control for Electric Cars) project revolutionizes the realm of electric RC car control by integrating advanced hardware and software technologies to create an engaging and immersive platform for internet-controlled (IC) car racing. By integrating hardware components such as microcontrollers, cameras, motors, and image sensors into customized PCB(s) for the IC cars, we have created a robust and reliable system through sophisticated implementation of communication protocols to enable users to access and control the cars remotely via an internet connection. Then, with the inclusion of a unique finish line system that incorporates a detection camera and LED matrix, coupled with software functionalities, users are able to compete against one another in races where the system accurately tracks the laps of each user and records the result of each race to users.**

***Index Terms* — IC (Internet-Controlled) Cars, Race System, Finish Line, ESP32, OV2640, Pixy 2.**

I. Introduction

Remote-controlled (RC) cars have long been a popular hobby amongst children and enthusiasts who enjoy both the technical aspects of building and modifying the vehicles, and the thrill of competing in races. As simple as a RC car may seem, it is responsible for fond memories for many individuals. Unfortunately, traditional RC car racing is limited by physical proximity, requiring participants to be in the same location. This project helps to alleviate this issue while withholding the joy of RC racing that many have held dearly.

Originating from a heavy interest in robotics and gaming, we wanted to create a project that is not only entertaining and fun but also interactive amongst people who may not be able to be in the same place together. Going into the brainstorming for the project we knew we wanted to implement race cars in some way, as a member had been doing prior research on implementing cars and internet connectivity. As avid gamers, we also felt like creating a competitive aspect where users can race head to head or against time through a complete racing system.

The motivation behind this project is simple, to combine the excitement of RC car racing with the accessibility and interactivity of online gaming, creating a dynamic and engaging platform where users can race against each other remotely. By incorporating elements such as live streaming and car controls, we aim to bring a video-game-like experience to the real world, offering an innovative, entertaining, and technically challenging product for hobbyists and competitive racers alike. This project serves as a medium between the physical world and virtual space, offering an experience unique to users since it blends elements of physical RC car racing and online video game racing.

II. RACE System Features

To fully realize the internet-controlled (IC) car racing system we wanted to create, we determined six critical features to incorporate into this project in order to meet engineering specifications and the UCF Senior Design requirements:

1) Internet-Based Remote Control: Users can access and control the IC cars from anywhere in the state via a reliable internet connection. The system utilizes a web-based interface, allowing players to send commands to the cars with minimized latency.
2) First-Person View (FPV) Camera System: Each car is equipped with a camera that streams video in real-time back to the user, providing a direct first-person view (FPV) as though they were inside the vehicle.
3) Multi-User Support with Scalable Server Architecture: The system is designed to allow multiple users to race simultaneously. The server architecture is built to scale, ensuring smooth performance even with two concurrent users.
4) Web-Based User Interface: Users interact with the system through a web browser interface. The interface provides control inputs, live video feed, and race stats.
5) Lap Tracking System: A sensor is implemented to automatically track laps. This ensures accurate race results and eliminates the need for manual lap counting.
6) Custom Printed Circuit Board (PCB): The IC cars feature custom PCBs that integrate a microcontroller responsible for receiving commands from the server and executing them on the car's driving and steering motors via a dual-motor driver. The microcontroller is also responsible for sending live video feed to the

dedicated server from the image sensor integrated onto the PCB.

III. RACE System Components

The RACE system is primarily composed of two major subsystems, the first being the IC race cars and the second being the finish line system. The IC race cars consist of DC motors, a motor driver, an IoT camera, a microcontroller (MCU), a USB-UART converter, and a power supply unit (PSU) all together on a car chassis. The finish line incorporates a single-board computer that interfaces with a color detection camera and a LED matrix to provide the features of a typical car race.

A. Microcontroller(s)

To choose the right microcontroller for our cars, several factors were evaluated, including size, Wi-Fi connectivity, operating voltage, memory capacity, the number of cores, and SRAM size. After sufficient research, we decided to utilize ESP32 due to its powerful dual-core architecture, high clock speed, Wi-Fi connectivity, and number of GPIO pins.

The specific chip is the ESP32-WROVER-E, and with its dev kit's integrated camera ribbon connector, it enabled us to effectively reduce prototyping complexity and maximize accessibility with the cars selected IoT camera since it possesses much smoother integration with ESP32 over other MCUs. The ESP32-WROVER-E chip includes a PCB trace antenna as well, which has allowed us to minimize additional hardware and maximize performance in our project.

Figure 1: Freenove ESP32-WROVER Cam Board

B. RC Car Model

With the constraints involved with Senior Design, specifically time and manufacturability, instead of building a car from the ground up, we decided to purchase existing RC car models on the market, strip all the electronics and exterior away, and then test the models' chassis and DC motors to determine an optimal selection for our project.

After testing three different options, we selected a 1:24 scale Monster Jam RC car due to its voltage operating range, chassis size, and smooth control.

C. Motors & Motor Driver

Each car possesses a DC driving and steering motor, with both motors able to operate at 5 V while drawing 0.5 A and 0.8 A respectively. The voltage operating level and current requirements for these motors were deemed suitable for the cars since we intended to utilize a dual-motor driver to manage both the driving and steering functions of the cars through a single module.

We chose the DRV8833 dual H-bridge motor driver due to it operating within the voltage and current specifications required by our motors, ensuring reliable and safe performance through dual motor control operating at 5 V.

D. IoT Camera

After conducting thorough research into camera technologies that could be used for live video streaming in the cars, out of the investigated options, IoT (internet of things) cameras were selected due to their video transmission capabilities over Wi-Fi, FOV (field-of-view) for ideal first-person FPV, compact size, lower cost, and smoother interfacing with microcontrollers.

As mentioned previously in the microcontroller section, the ESP32-WROVER-E dev kit incorporates a camera ribbon connector for IoT cameras. Thus, after researching IoT camera modules that incorporate that ribbon connector, we decided on the OV2640 due to its 160° FOV, adjustable frame rate, direct ESP32 integration, and higher resolution than other OVxxxx modules that were researched.

E. USB-UART Bridge

One element of the PCB design needed for this project was a USB/UART bridge since the ESP32-WROVER chip needs RS-232 communication to program the device. Thus, by using the USB-UART bridge, CP2102N, a USB-UART converter is included in the PCB. That converter enabled us to then program the ESP32 chips via a USB connection to their respective UART interface.

F. Single-Board Computer

In order to communicate between our server and hardware for the finish line detection system, we decided upon using a single-board computer. We wanted a device that is capable of communicating via WiFi to the server, and that can also distribute power to our Pixy 2 and LED matrix. After attempting and failing to implement the ESP32-WROOM32 in this system we ultimately selected the Freenove Control Board V5 Rev4 WiFi due to its compatibility with the Pixy 2 Camera.

The Freenove Control Board V5 Rev4 WiFi is able to connect our Pixy 2 and LED Matrix via SPI and I2C connection, respectively. It offers built-in voltage regulators that enable us to power both of our peripherals with 3.3V and 5V regulated voltage outputs.

G. Color Detection Camera

To ensure accurate detection for our two IC cars at the finish line, we narrowed our research of potential methods down to color detection via a camera, and due to smoother integration and specialization in color/object detection we selected the PIXY 2 camera.

The PIXY 2 camera offers efficient color-based object tracking, line following, and marker reading with streamlined software. It detects multiple objects by recognizing color signatures, providing position, size, and color data. Its built-in line-following algorithm aids in navigation, while color-coded tags enable multi-object tracking. Pre-installed firmware is easily customizable via PixyMon, allowing real-time adjustments. Compatible with platforms like Arduino and Raspberry Pi, it integrates seamlessly with provided libraries, making it a user-friendly solution for reliable object detection.

H. LED Matrix

To create an experience similar to that of a real professional driving race, we wanted to implement as many components that you may see in the environment to a project of this scale. One notable feature of a professional race is a countdown with the use of traffic lights. To simulate this, we decided a LED matrix would be the best way to approach this feature.

From our research, we agreed that the Adafruit Bicolor LED Square Pixel Matrix is the best fit for our project's purposes. It has a 8x8 display with red, green, and amber pixels, controlled via I2C using the integrated HT16K33 driver. It requires only two pins (SDA, SCL) and features Qwiic/STEMMA QT connectors for easy integration with other I2C devices. Operating at 3.3V and 5V, it's compatible with various microcontrollers, including our Freenove Control Board. Its compact size (1.5" x 1.5") and support from Adafruit libraries make it ideal for simple visual displays. The integrated driver simplifies microcontroller connectivity, while its bi-color capability enables a red-green countdown for user-friendly race starts.

IV. Power Supply Unit

Device	Voltage Range	Current Draw	Peak Power
MCU	3.3 V	150-300 mA	~1 W
Camera	3.3 V	100-150 mA	~0.5 W
Driving Motor	5 V	500 mA	2.5 W
Steering Motor	5 V	800 mA	4 W
Dual Motor Driver	2.7-10.8 V	1500 mA (Max per Channel)	6.5 W (Operating at 5 V)

Table 1: Peak Power Needs from IC Cars

As shown in the table above, the IC cars require two main voltage rails, those being 3.3 V and 5 V, to supply power to their respective devices. From that table as well, the maximum current draw when all devices are powered and in use is less than 2 A, with the majority of that from the dual motor driver powering both the driving and steering motors. Thus, in determining the PSU parameters, we recognized that a 7.4 V Battery with significant capacity (mAh) would be the optimal choice for the cars power supply, to efficiently and effectively power all the devices for our cars' desired battery lifetime. We also determined that the IC Cars require 3.3 V and 5 V Voltage Regulators to step-down the 7.4 V from the battery for sufficient power delivery to their respective devices.

Now for the finish line system, we determined that a 5 V USB power supply connected to the Freenove Control Board would be the optimal choice for both power supply and power distribution within that system. That is because the control board is connected to the race moderator's laptop, and from that USB connection to the finish line system, the control board sufficiently distributes power to both the Pixy 2 camera and the LED Matrix.

A. Battery Pack(s)

Having decided upon a 7.4 V Battery to serve as the cars power supply, Lithium-based battery technology was the ideal option we discovered upon research. Thus, due to their effectiveness in RC car applications, we chose to utilize a Li-Po (Lithium-Polymer) battery product as the cars power source. Since one of our engineering specifications is for the cars' batteries to last for more than 15 minutes, we utilized a simple formula to help discern the correct capacity (mAh) to sufficiently meet the specification.

$$Battery\ Runtime = \frac{Battery\ Rated\ Capacity\ (in\ mAh)}{Load\ Current\ (in\ mA)}$$

Figure 2: Battery Capacity Formula

With a current draw of less than 2 A, 1100 mAh Battery Capacity was determined to significantly exceed our battery lifetime specification for more than 15 minutes of runtime when all devices are power and running, giving the cars at least 30+ minutes of battery lifetime.

B. Voltage Regulation

As mentioned, the cars need 3.3 V and 5 V voltage regulators to effectively power the devices on the cars PCB(s). After research and investigation, switching regulation was selected over linear regulation due to their higher efficiency and better thermal performance, and two TPS buck converters (step-down voltage regulators) were selected due to their adjustable input/output voltage range and lower switching frequency compared to other converters.

The TPS561208 serves as the 3.3 V regulator, powering the ESP32 and the OV2640, then the TPS563231 serves as the 5 V regulator, powering the dual motor driver and its connected DC motors.

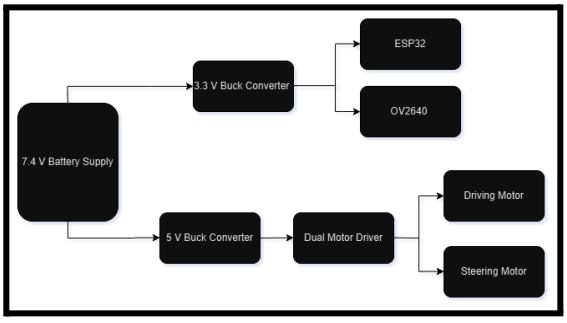


Figure 3: Power Distribution Diagram

V. Hardware Design

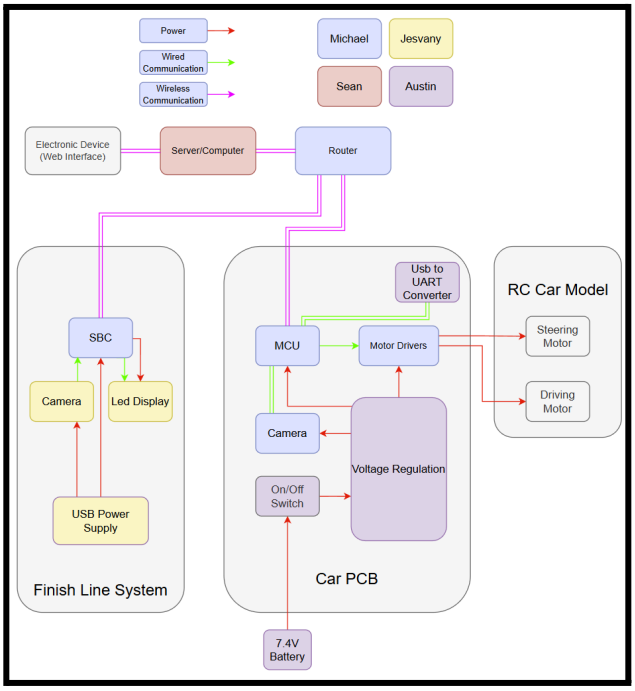


Figure 4: Hardware Block Diagram

The figure above shows a detailed outline of the layout of the hardware design for this project, including colored arrows that represent power distribution, wired communication, and wireless communication amongst all of the hardware components in the RACE system.

A. IC Cars Design

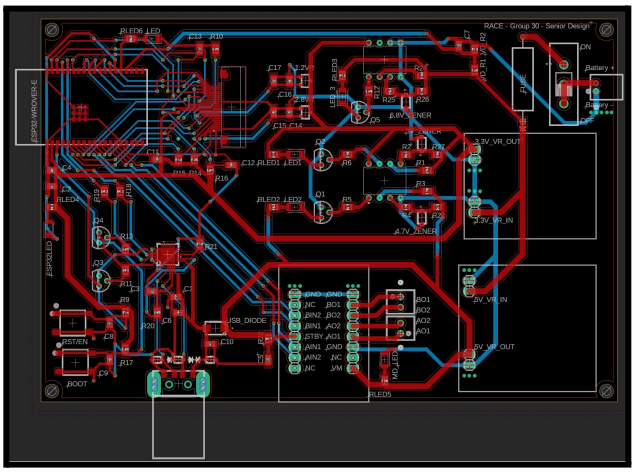


Figure 5: Complete PCB Design

As shown in the figure above, it details the complete board layout and routing for the car's main PCB. Starting from the top right corner, the battery's JST connector

plugs into the board, and once the switch is flipped to the 'ON' position, the 7.4 V battery supply goes directly to the input of the two voltage regulators breakout PCBs that are connected via male and female headers onto the main board. All three voltage levels are then monitored by comparator LED circuits in the top middle portion of the board, providing status indicators that the 7.4 V, 5 V, and 3.3 V are steady at their correct voltage level. At the bottom middle portion is the motor driver breakout PCB connection with terminal blocks for the DC motor terminals. To the left of the motor driver is a USB Type A plug that is connected to the CP2102N USB-UART bridge along with protection diodes and push-buttons for resetting and booting the ESP32. The ESP32-WROVER-E chip rests at the top leftmost portion of the board, with its antenna left at the edge to eliminate interference, and to the right of the ESP32 is the 24-position camera connector for the OV2640. The OV2640 does require 2.8 V and 1.2 V for analog and digital power supply, thus LDO (low drop-out) regulators are incorporated onto the PCB to meet those voltage requirements.

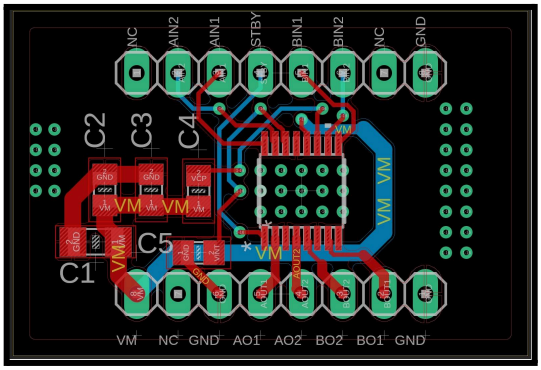
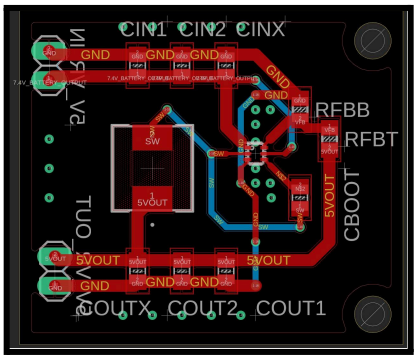


Figure 6: Motor Driver PCB Design

This next figure shows the dual motor driver breakout PCB that gets plugged into the main PCB, with its design being based on a DRV8833 motor driver module that was used during the prototyping and testing phase. Using the datasheet, we included a strong ground plane connection under the chip by incorporating thermal vias that act as a heat sink for the chip.



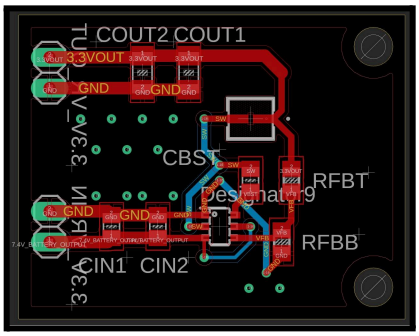


Figure 7: Voltage Regulators PCB Design

Figure 7 shows the two voltage regulator breakout PCBS that get plugged into the main PCB. Both of the designs were based directly off of the datasheets recommended PCB layout with the corresponding resistor, capacitor, and inductor values matching the datasheet.

The PCB is securely connected onto the car chassis by a 3D-printed structure, with the OV2640 camera having a dedicated slot for direct video feed from the "driver's POV" at the front of the car. The DC motors and battery pack is securely attached within the car chassis as well, completing the hardware integration of the IC cars.

B. Finish Line Design

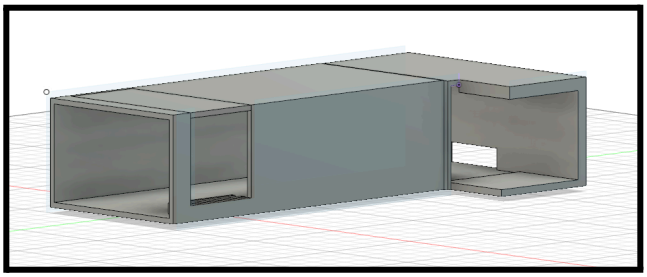


Figure 8: Finish Line Enclosure

The finish line system is designed to accurately detect cars as they cross the finish line, process lap data, and display race results in real-time. The system utilizes the

Pixy 2 camera to identify the different colors on the car structures, ensuring precise and consistent detection. The detection signals are sent to our Freenove Control Board's ESP32 module, which processes the data via WiFi to a WebSocket server, which is responsible for managing lap tracking, determining race winners, and updating race progress dynamically through the LED Matrix. The LED matrix display shows essential race information, including the initial countdown, lap updates, and final results.

All of these components are housed in an enclosure, as shown in Figure 8, to maintain a stable environment for our finish line hardware components. This design allows the camera angle to be adjusted to give flexibility in the positioning of the track and the enclosure itself. Also, users will be able to see the LED Matrix at the finish line via the onboard camera. Lastly, it has a small opening for easy connection from the Freenove Control Board to our computer.

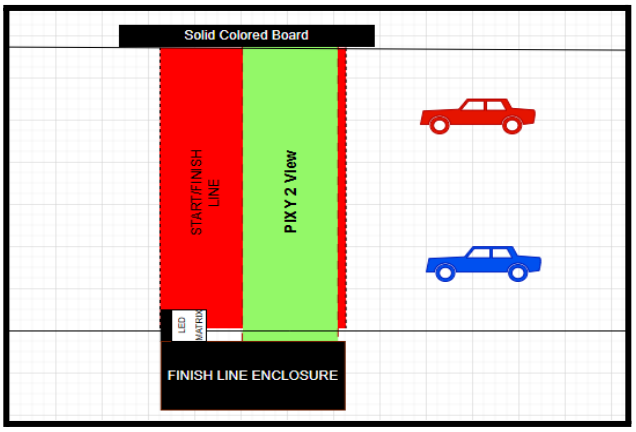


Figure 9: Finish Line System Portrayal

The next figure illustrates the layout and positioning of the finish line system, ensuring the Pixy 2 camera effectively detects the IC cars and the LED matrix is properly displayed. Positioned along the other side of the finish line, the system incorporates a solid-colored board on the opposite side of the track, parallel to the enclosure, to enhance detection accuracy by minimizing potential interference. The LED matrix will be prominently placed at the side of the track, facing the IC cars at the starting position for clear visibility to the users and in-person spectators.

VI. Software Design

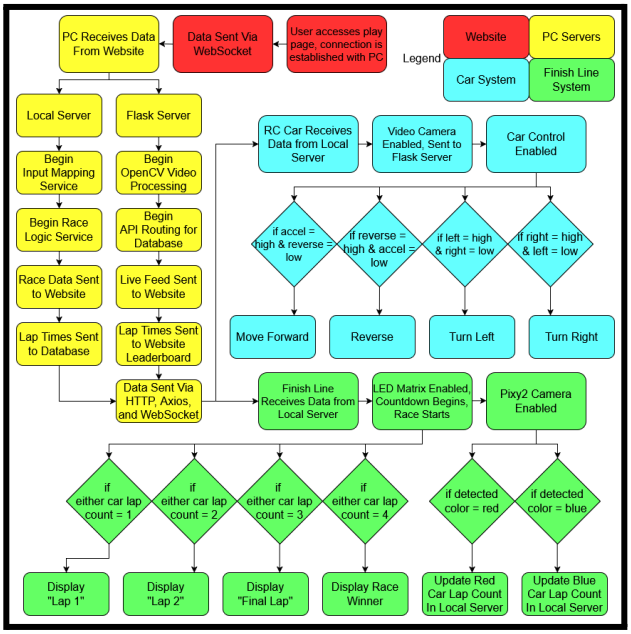


Figure 10: Software Flowchart

Our goal for the RACE system was to build a system that is easy to understand, maintain, and extend, while meeting all project requirements. We focused on clear organization of modules, reliable communication between components, well-defined software functions, and a user-friendly interface that makes the system accessible to both technical and non-technical users.

A. Communication Network

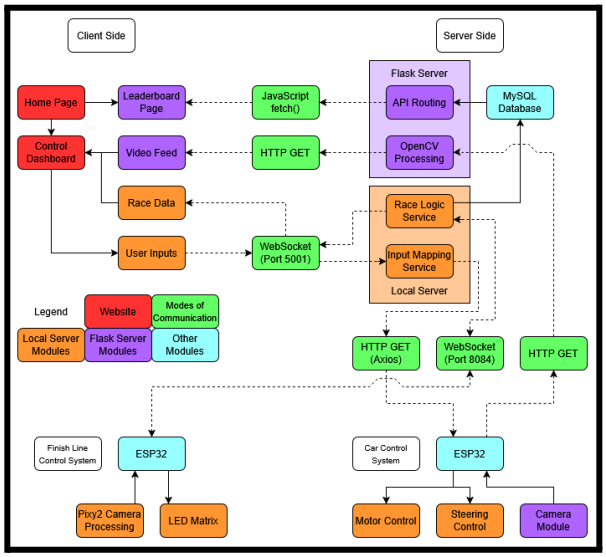


Figure 11: Communication Network Diagram

The RACE system uses a local Wi-Fi network where ESP32 modules in the cars and finish line connect directly to a local server using HTTP, Axios, and WebSockets. Each car ESP32 hosts a web server handling HTTP GET requests using Axios from a local server hosted on a personal computer to control car movements. Commands like move_forward, turn_left, and stop are sent directly to the cars via specific URL requests. The finish line ESP32 communicates race updates to the local server through WebSocket messages, which then processes those updates, and communicates back to the finish line ESP32 to display results such as lap counts or race winners on the LED matrix. Lastly, an additional local Flask server, also hosted on the personal computer, manages video streaming from the cars to the user's browser, and also uses HTTP GET requests.

The communication network diagram seen in Figure 11 details our entire communication network. The communication flow starts with the user's web browser, which sends keyboard input via WebSocket port 5001 to the local server over the internet. The server interprets these inputs and forwards corresponding Axios HTTP GET requests directly to the targeted car's ESP32 module. Upon receiving these requests, the ESP32 executes the commands to control the car's movements. Then, each car's ESP32 module captures live camera images and makes them available via an HTTP endpoint. The local Flask server continuously fetches these images, processes them into a video stream using OpenCV, and serves the stream to the user's web browser. This setup allows users to view real-time video feeds directly from their selected car.

Additionally, the local server communicates with a MySQL database hosted locally on the personal computer to track user lap times and race results. As each lap is completed, the server records the data into the database, enabling us to display user rankings and the fastest lap times on the website. To achieve this, we added an additional API route to the local flask server. This route queries the MySQL database and fetches the top lap times as JSON via JavaScript fetch(). This database integration facilitates a leaderboard page on the website, which encourages user engagement and competition.

B. Software Functionalities

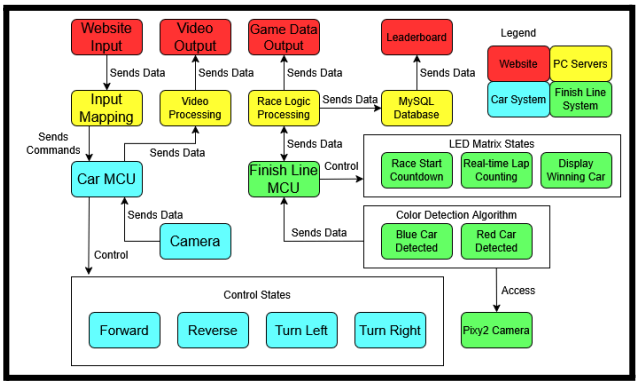


Figure 12: Software Functionality Block Diagram

The core functionalities of the RACE system involve controlling the cars, video streaming, and managing race conditions. Car control is executed through direct HTTP commands sent to the ESP32 web servers embedded in each car, triggering actions like acceleration, braking, turning, or stopping all movement. Live video streaming from each car's camera is processed by the local Flask server using Python's OpenCV library, ensuring smooth and reliable real-time video feeds on the website. Each frame from the ESP32 camera is fetched via HTTP, decoded, and re-encoded into a continuous video stream. The Flask server continuously monitors the responsiveness of each car and handles errors gracefully, ensuring minimal disruption.

Race management functionalities include real-time lap counting, recording lap time, and updating race status. The local server tracks laps completed by each car based on data received from the finish line ESP32 and Pixy2 camera. This data triggers server-side logic to determine race progress, record lap times, and announce race winners. The results are immediately sent back to the finish line for visual display on the LED matrix, which will be visible from the cars live video feed and any in-person spectators, enhancing the racing experience and increasing interactivity. The recorded lap times will be sent to the MySQL database to be featured on the leaderboard page of the website.

C. User Interface

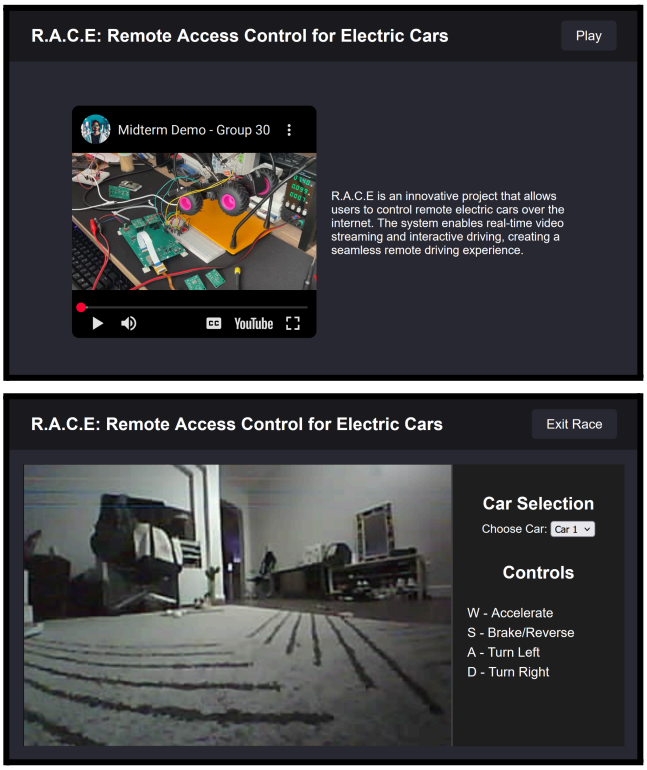


Figure 13: Website UI Layouts

The user interface of the RACE system, simple in design, is built with standard HTML, CSS, and JavaScript, and focuses on ease of use and clarity. The home page features a video that users can watch to gain a better understanding of how the RACE system works, as well as a brief overview of how to utilize the system. To the right of the navigation bar at the top of the page is the play button. The "Play" button will take the user to the play page, where the website will automatically connect to the rest of the system, providing a seamless experience.

The play page features a live video feed section that provides real-time visuals from the selected car's onboard camera. Users interact intuitively using familiar keyboard controls (W, A, S, and D), which trigger movement commands sent via WebSocket connections to the local server. The interface includes a dropdown menu that allows quick selection between multiple cars, automatically updating the live video feed and corresponding control channels without manual refreshes. Clear instructions of the controls are displayed in the sidebar, ensuring users can immediately understand and interact with the system. Lastly, the "Exit Race" button will bring the user back to the home page. The design ensures consistent functionality and readability across various screen sizes, browsers, and devices, providing a seamless user experience.

VII. Conclusion

The RACE project successfully integrates sophisticated hardware and software into the creation of an interactive and immersive IC car racing experience. The RACE system has met functional requirements and engineering specifications, successfully establishing a framework for expanding the possibilities of internet-controlled systems. This project demonstrates how thoughtful engineering, design, and teamwork can transform traditional RC racing into a dynamic, interactive, and unique experience.

VIII. Biography

Austin Silva is a 22-year old electrical engineering student who will be graduating with his bachelor's degree in May of 2025. After graduation, Austin will be starting his career in power system engineering as a Protection & Control engineer with Florida Power & Light.

Michael Mallette is a 23-year old electrical engineering student who will be graduating with his bachelor's degree in May of 2025, with plans to pursue a master's in Robotics and Autonomous Systems. Michael is now aiming to evolve the project into a startup that delivers an interactive and remotely accessible experience to users, starting with construction vehicles.

Jesvany Colon is a 24-year old electrical engineering student who will be graduating with his bachelor's degree in May of 2025. After graduation, Jesvany will be taking some time for travel while continuing his career search in the field of semiconductors, audio devices or construction.

Sean Chaney is a 25-year old computer engineering student who will be graduating with his bachelor's degree in May of 2025. After graduation, Sean will be moving back to his hometown Jacksonville, FL and will be working full time at Publix while he continues his job search.

Acknowledgement

The authors wish to acknowledge the assistance and support of the University of Central Florida faculty in the ECE department, with special recognition to Dr. Arthur Weeks and Dr. Chung Yong Chan.