# Responses to Reviewers

## 1. Summary

We revised the report to address the two remaining design comments. First, we added a clearer explanation of how the car keeps its forward orientation using MPU6050 yaw feedback and yaw PID correction. Second, we added a physical-design figure and a major-subsystem summary so the car structure, module placement, and subsystem responsibilities are easier to understand.

## 2. Questions for General Evaluation

| Reviewer comment | Revision focus |
| --- | --- |
| Explain how the car manages to always orient forward. | Added yaw-feedback and yaw PID explanation with the provided block diagram. |
| Provide a block diagram for the car. Provide the physical design and describe major subsystems. | Added a physical layout figure and a subsystem table. |

## 3. Point-by-point Response to Comments and Suggestions

**Comment 1: Explain how the car manages to always orient forward.**

**Response 1:** We added a description of the yaw-feedback control loop. To keep the car moving forward, the STM32 uses the MPU6050 yaw angle as feedback. When the forward command is sent, the current yaw angle is saved as the target direction. During movement, the STM32 compares the current yaw with the target yaw. If there is an error, the yaw PID controller calculates a correction value and adjusts the left and right motor speeds. This allows the car to return to the original direction when it starts to drift.

**Added explanation:** The control is a closed-loop heading correction rather than a fixed equal-speed command. The yaw error is converted into a differential speed correction, so one motor is slightly increased while the other is slightly reduced. This helps compensate for motor mismatch, floor friction, and small disturbances during forward motion.

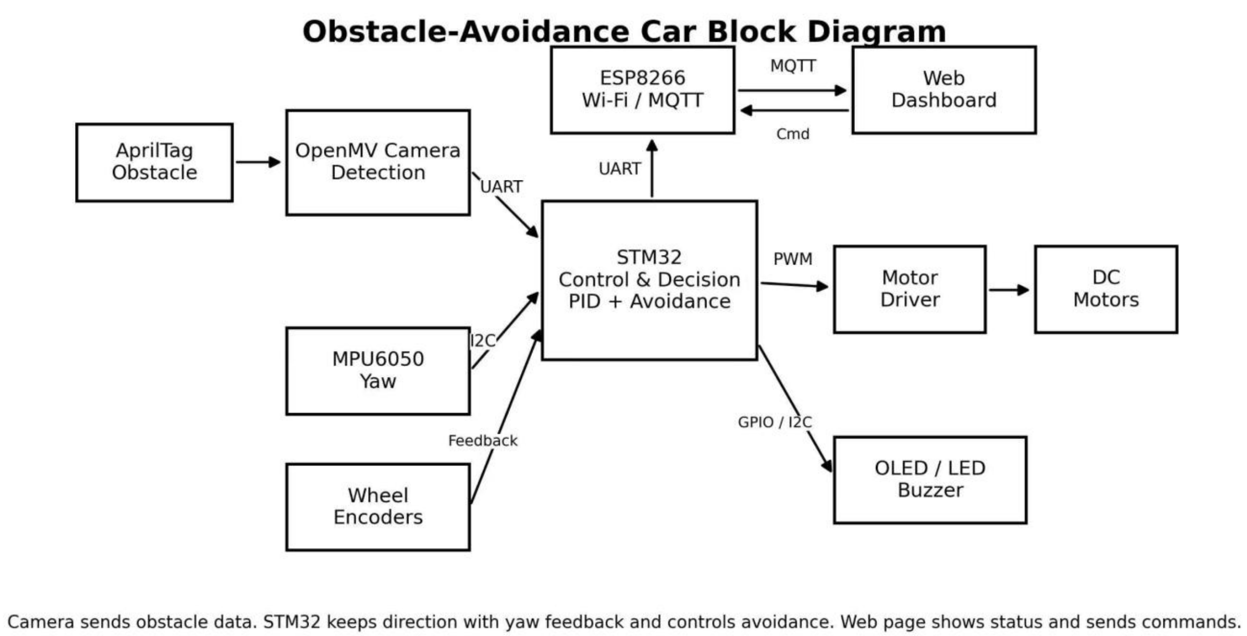


*Figure 1. Block diagram used to explain yaw-feedback forward-orientation control.*

**Comment 2: Provide the physical design and describe major subsystems.**

**Response 2:** We added a physical-design figure and a subsystem summary. The car uses a two-wheel differential-drive chassis. The left and right DC motors are mounted on the two sides and can be controlled independently. The camera module is mounted at the front to observe obstacle markers. The STM32 board is placed near the center as the main controller. The MPU6050 is fixed on the chassis to measure yaw drift, while the wheel encoders provide wheel-speed and distance feedback. The motor driver connects the STM32 to the two DC motors, and the Wi-Fi/MQTT module connects the car to the web dashboard.

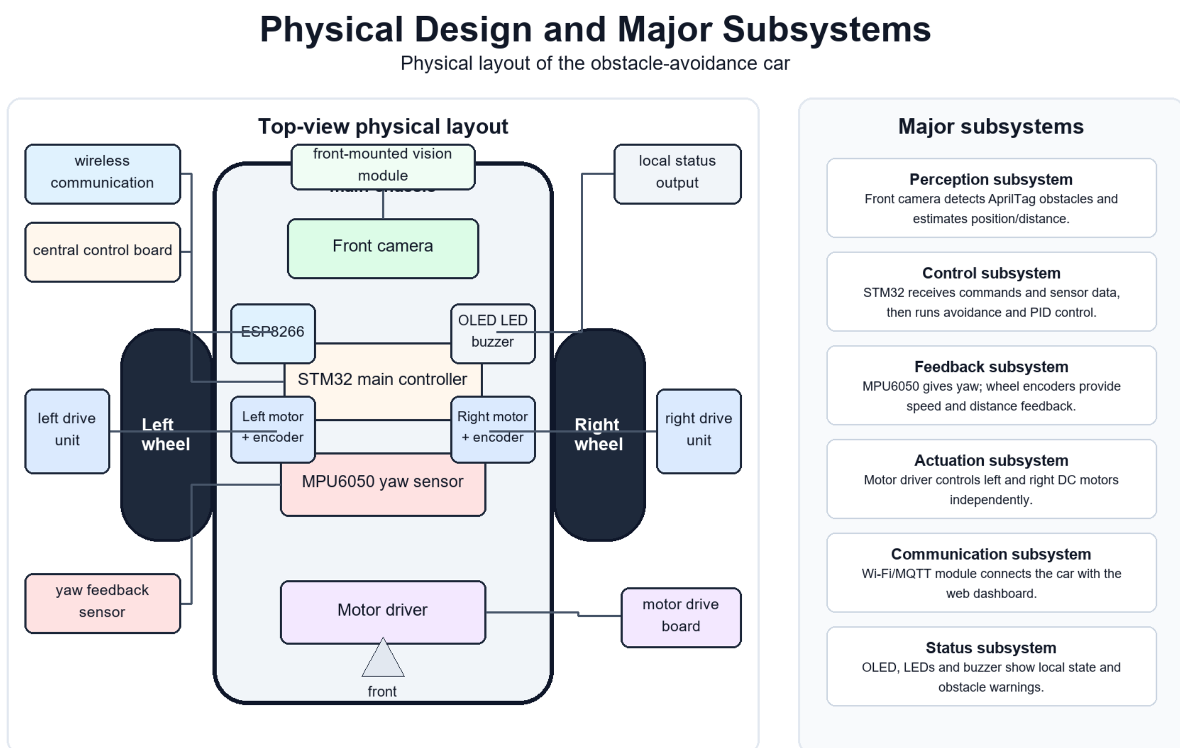


*Figure 2. Physical layout of the car and major subsystem roles.*

The major subsystems are perception, control, feedback, actuation, communication, and status output. These are summarized below.

| Subsystem | Main hardware | Role |
|---|---|---|
| Perception | Front camera module | Detects obstacle markers and provides obstacle information to the controller. |
| Control | STM32 main controller | Runs command handling, yaw correction, motor control, and obstacle-avoidance decisions. |
| Feedback | MPU6050 and wheel encoders | Measures yaw, wheel speed, and travel distance for closed-loop control. |
| Actuation | Motor driver and left/right DC motors | Drives the differential wheels and executes corrected speed commands. |
| Communication | Wi-Fi/MQTT module and web dashboard | Sends remote commands and receives car status data. |
| Status output | OLED, LEDs, and buzzer | Displays local state and gives obstacle or operation warnings. |