Embedded Project Description: Self-Parking Car with Bluetooth Control

Project Title

Self-Parking Car Using Bluetooth and Embedded System

Objective

The goal of this project is to design and develop a prototype self-parking car that receives initial control signals via Bluetooth and then autonomously detects a suitable parking space and parks itself using sensor-based feedback.

System Overview

This embedded system project focuses on a miniature car prototype that incorporates:
- Bluetooth module for wireless command input.
- Microcontroller to process instructions and control the vehicle.
- Sensors (e.g., ultrasonic or IR) to detect parking spaces and obstacles.
- Motors and motor driver circuit to navigate and park the car.
- Power supply for on-board components.

Functional Description

1. Bluetooth Communication:
   A mobile app sends a command via Bluetooth to the car (e.g., "start parking").

2. Obstacle Detection & Slot Identification:
   The car moves forward slowly, using side-facing ultrasonic sensors to detect gaps between obstacles (representing cars or parking boundaries). It identifies an available parking slot based on distance thresholds.

3. Parking Maneuver:
   Once a slot is detected, the car calculates the appropriate maneuver (parallel or perpendicular) and autonomously navigates into the parking space using preprogrammed logic.

System Components

- Microcontroller: Arduino Uno / STM32 / ESP32
- Bluetooth Module: HC-05 / HC-06
- Ultrasonic Sensors: HC-SR04 (front and sides)
- Motor Driver: L298N dual H-bridge
- DC Motors: For driving and steering
- Chassis: 4-wheel base or differential drive
- Battery Pack: Rechargeable Li-ion or AA cells
- Optional: Servo motor (for steering), IR sensors, LEDs

- any other components

Software Requirements

- Arduino IDE / STM32CubeIDE
- Bluetooth terminal app / Custom mobile app
- Embedded C / C++
- PID algorithm (optional for smoother control)

Parking Model Description

- Parking Environment: A mock parking lot with cardboard cars and designated parking slots.
- Parking Types Supported:
  - Parallel parking between two obstacles
  - Perpendicular parking between two lines
- Detection Method:
  - Car scans space while moving forward.
  - If a wide enough space is detected (e.g., ≥ 15 cm), it flags it as a valid slot.
  - Based on the type of space, it initiates a predefined maneuver.
- Road Width: The road on which the car drives is 30 cm wide.

Expected Outcomes

- Bluetooth-controlled start of parking process.
- Accurate detection of an available parking slot using sensors.
- Smooth and safe autonomous parking into the detected slot.
- Visual indication (LED or buzzer) on parking success.

Additional Requirement

An essential requirement for this project is the development of low-level drivers for all sensors and motors used in the system. These drivers must be implemented using low-level programming techniques by directly accessing and manipulating the hardware registers of the microcontroller, instead of relying on high-level Arduino libraries or predefined functions.

This includes:
- Writing register-level code for initializing and reading from the ultrasonic sensors (e.g., HC-SR04).
- Implementing motor control logic by configuring timers, PWM, and GPIO registers manually.
- Ensuring optimized and deterministic behavior suitable for real-time embedded applications.

This approach aims to enhance understanding of hardware interaction, provide better control over performance, and prepare the developer for bare-metal embedded systems programming.

Team Members

- Each team shall consist of a maximum of four members

Project Due Date

- Delivery a softcopy on google classroom with a block diagram for the system.
- Week 13, Thursday 8th of May 2025 at 9 am. The demo will be during the regular lab session.