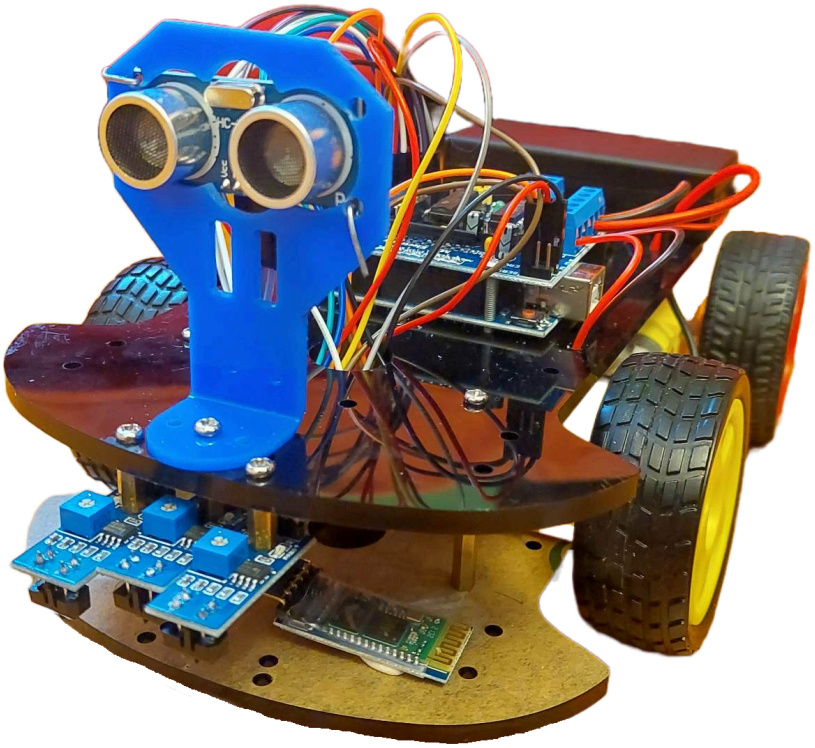
Arduino Bluetooth-Controlled Car with Obstacle Detection and Line Following

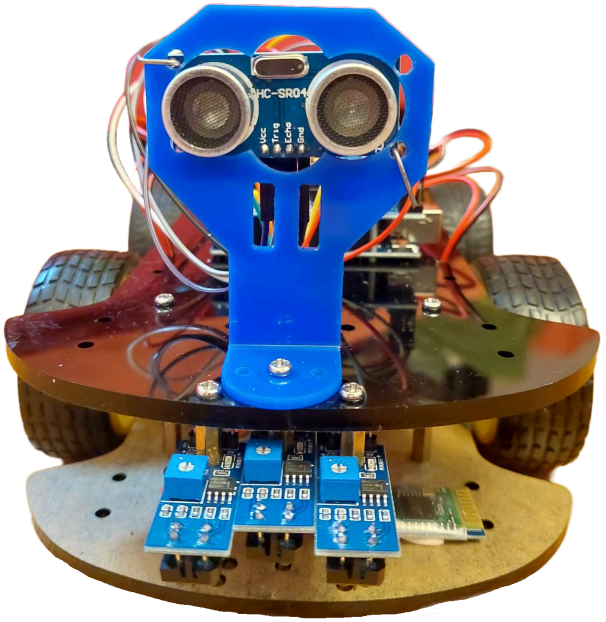


Collaborators:
Manousos Linardakis
Christos Kazakos

Contents:

Introduction:

This Robot-Car is a bluetooth-controlled vehicle that was designed and built from scratch as part of an Electronics and IoT course at Harokopio University of Athens (department of Informatics and Telematics). It is equipped with an ultrasonic sensor for obstacle detection and three line sensors for line following. In addition, an Android application was developed to control the Robot-Car with buttons for forward, backward, left, and right movements, along with an autopilot and line-following mode. The source code for the project is available on GitHub (see this repository).



Hardware Components:

- Motors (with Wheels):
  The Robot-Car is equipped with four motors, each with its wheel, providing it with movement capabilities in different directions (forwards, backwards, left and right).

- Rechargeable Batteries (18650 3.7V):
  Power source for the motors and other components. The Robot-Car is powered by rechargeable batteries that can be recharged using a charger.

- Motor Driver L293D:
  The Motor Driver L293D is an essential component of the Robot-Car, as it enables the Arduino to control the motors that drive the car's movement. This dual H-bridge

driver provides precise control over the direction and speed of each motor using the AFMotor library for Arduino. In addition to controlling the motors, the L293D also serves as a convenient interface for connecting the ultrasonic sensor and line detectors, which are all connected to the analogue pins of the motor driver. With the L293D and the AFMotor library, the Robot-Car is able to move with precision and respond to external stimuli such as obstacles and lines with ease.

- Arduino Uno:
  The Robot-Car is powered by the Arduino Uno, which acts as the "brain" of the vehicle, processing the commands sent via Bluetooth and controlling its motors, sensors, and other components. The Arduino Uno is connected to the L293D motor driver, which facilitates the control of each motor's direction and speed. It can be programmed using the Arduino IDE.

- Bluetooth Module (HC-05):
  The Bluetooth module is an essential component that enables wireless communication between the Robot-Car and the user's phone. It receives commands from the app on the user's phone and transmits them to the Robot-Car, which then executes them. Without the Bluetooth module, the Robot-Car would not be able to be controlled remotely and would lose a significant part of its functionality.

- Ultrasonic Sensor:
  The ultrasonic sensor is an important component of the Robot-Car that detects obstacles in its path and signals the Arduino to change direction (when the user chooses autopilot mode). This sensor uses sound waves to detect objects by sending out a pulse and measuring the time it takes for the pulse to bounce back. The Arduino then uses this information to calculate the distance of the obstacle and determine the appropriate action to avoid it. The ultrasonic sensor is a highly reliable and accurate sensor, making it an essential part of the Robot-Car's obstacle avoidance system.

- Line Sensor (3 pieces):
  The Robot-Car is equipped with three line sensors that help it follow a desired path. The sensors are positioned at the left, right, and center of the car's underside, and are used to detect and follow a black line on a white surface. When a sensor detects the line, the Robot-Car adjusts its direction to stay on track. If no detection occurs on the left or right sensor, the car rotates clockwise in order to find the line. We used analog output for the sensors to allow for greater flexibility in determining the threshold values for detecting the dark line.

- Car Body:
  The chassis is an essential part of the Robot-Car, serving as the physical structure that houses all of the electronic components.

- Battery holder (with switch):
  Used to connect the batteries to the arduino.

Software Components:

- Arduino IDE:
  Used for coding the Arduino Uno. The Arduino IDE is an open-source software development environment based on the Processing programming language.

- Android Studio:
  The Android application was developed using Android Studio, a popular integrated development environment (IDE) for Android app development. The knowledge and skills required for developing the app were acquired through the Android Development course taken at Harokopio University. This course provided us with a solid understanding of Android app development.

Functionality:

The Robot-Car is designed to be controlled remotely via Bluetooth using an Android application. The user can control the car's movements by sending commands through the app, such as forward, backward, left, or right. In addition, the app features an autopilot mode that allows the car to drive forward and rotate whenever it encounters an obstacle. The ultrasonic sensor detects obstacles in the car's path and sends signals to the Arduino, which controls the motor driver to change the car's direction.

The Robot-Car is also equipped with three line sensors that enable it to follow a black line on a white surface. The line sensors send signals to the Arduino, which then controls the motor driver to keep the car on the line. If the car loses track of the line, it will rotate to the right until it detects the line again. The line following mode can be enabled through the app, and the car will move along a predetermined path based on the line sensor readings.

Code (Arduino):

The code uses the AFMotor library and the SoftwareSerial library in the Arduino IDE to control four DC motors. The AFMotor library provides functions to control DC motors via a motor driver. The SoftwareSerial library allows communication with a Bluetooth module via a serial channel.

- First, the motor pins are initialized. Then, variables trigPin, echoPin, ultraPilot, max_obs_dist, rot_delay_ultraPilot, motor speeds, rotation speed, various sensors, and a command that accepts the command we will give are initialized. Some of the most important are:
- The trigPin and echoPin correspond to the output and input pins of the ultrasonic distance sensor.

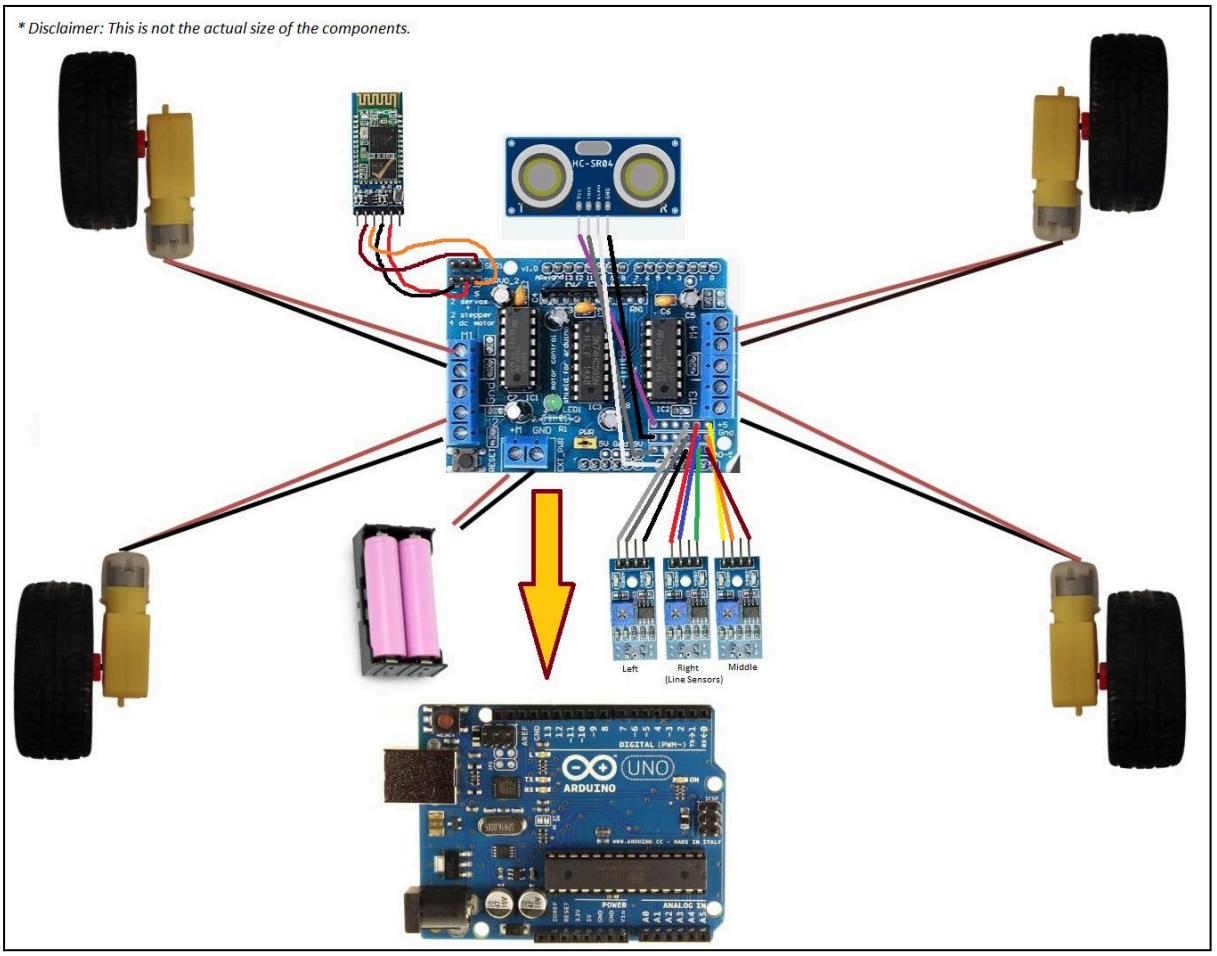
- The ultraPilot variable is a logical variable used to enable/disable the robot's automatic movement based on obstacle detection.
- max_obs_dist is the maximum distance for obstacle detection by the ultrasonic sensor.
- rot_delay_ultraPilot is the delay used to allow the robot to turn around an obstacle before continuing its movement.

The code contains the setup function, which sets the settings for serial communication and the ultrasonic sensor, left, right, forward, back functions that handle the direction of the robot's movement, the stop function that stops the robot, a loop function that accepts commands from the android application.

Code (Android):

Initially, in the code, we set various variables that we will use, such as buttons, switches, various strings, integers, etc. First of all, in onCreate(), we initialize the bluetoothDevices map, which contains the names of the Bluetooths and their corresponding Bluetooth objects with their characteristics and methods. Then we call the initMovementButtons() method, which initializes the necessary buttons in the application. Each movement button contains 2 methods, startAction() and stopAction(), where commands are sent to the Arduino via the sendCharToArduino(char c) method while the user has the button pressed and it operates accordingly. Additionally, there are 2 more buttons, autoSwitch and lineSwitch. The autoSwitch allows the robot to navigate automatically, while the lineSwitch is used when the robot needs to follow a line. When the user releases the button, no commands are sent to the robot. Furthermore, we initialize the bluetoothList with a default value, which contains the list of available Bluetooths. Finally, we call the initBluetoothList(bluetoothList) method, which populates the list with all available Bluetooths so that the user can later select the desired Bluetooth. Then there is the connectToAddress(String deviceAddress) method, which takes as input the Bluetooth selected by the user and is responsible for connecting the application with a Bluetooth (in our case, with the Arduino). Finally, the handleButtons(Switch s, boolean enable) method is called for the autoSwitch and lineSwitch buttons, which enables/disables all buttons to make the autoSwitch and lineSwitch function properly without any issues.

Circuit Diagram:



For a better view of the circuit diagram, please refer to this link.

Source Code (& video link):

All of the code developed for this project can be found in this github repository. For the components used in the project, please refer to the hardware components section.
Here is also a short video showcasing the robot-car.