Bluetooth controlled car with distance and light sensors

Student: Veliciu Vlad

Group: 2031

Coordinator: Prof. Giurgiu Mircea

Faculty: Electronics, Telecommunications, and Information Technology

Contents

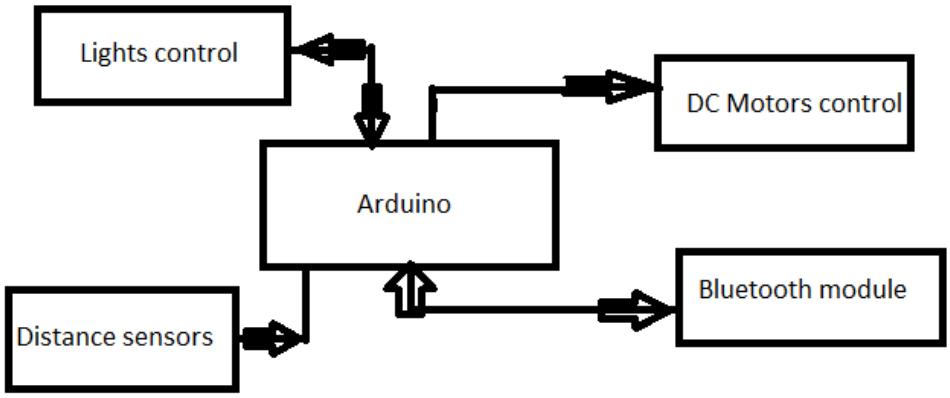
Brief Description

The purpose of this project is to expose a prototype of a Bluetooth controlled car that has some “autonomous” elements that will help the user to drive the car, such as obstacle identification sensors (implemented here with distance sensors) in the front and back and a light sensor to control the lights depending on the light intensity from the outside media. This aspect is extremely important today, as we speak more and more about autonomous driven cars that facilitate a more comfortable driving experience for the user and releases the stress that can appear. Not to say that some elements, like obstacle sensors that will stop the car immediately, are lifesaving, as the human is not able to react so fast as electronic sensors do.

Like it was said before, the project is supposed to implement a car that is controlled by the phone via Bluetooth connection. In addition, some improvements to this control have been added, such as ultrasonic distance sensors that will help the user to avoid a collision with obstacles in front or in the back in case the user loses focus and cannot control the car properly. So, this system can prevent damages to the car. Moreover, the light intensity sensor (a photoresistor) will control the turning on and off the lights (LEDs) from the front and the back of the car. These two elements bring a sort of autonomy to the car, in the sense that they help the user.

The implementation of this project is done with Arduino Uno as controller of all mechanisms that control the car and sensors. Briefly to say, there are four motors, one for each wheel, that are connected via an H-bridge with the output pins of Arduino. There is also a Bluetooth module that keeps data transmission between the phone and Arduino. Also, there are two ultrasonic sensors connected (one in the back and one in the front of the car) and a warning LED controlled by the fact that an obstacle is detected or not. The light intensity sensor will control four LEDs (two in the front, two in the back). Also, another 2 LEDs from the back will indicate when the car stopped (as for a real car). The functioning will be further explained.

Theoretical background



(Block diagram)

From the block diagram we can see the logical connections between the functional blocks of the project.

Arduino: It is the microcontroller, the brain, of the car. It will be supplied with a battery of 9V (as it is recommended) [1]. It will transmit all the necessary information to the Bluetooth module and the motors controller and the LEDs on the car. It receives information from Bluetooth module and the sensors (distance and light intensity).

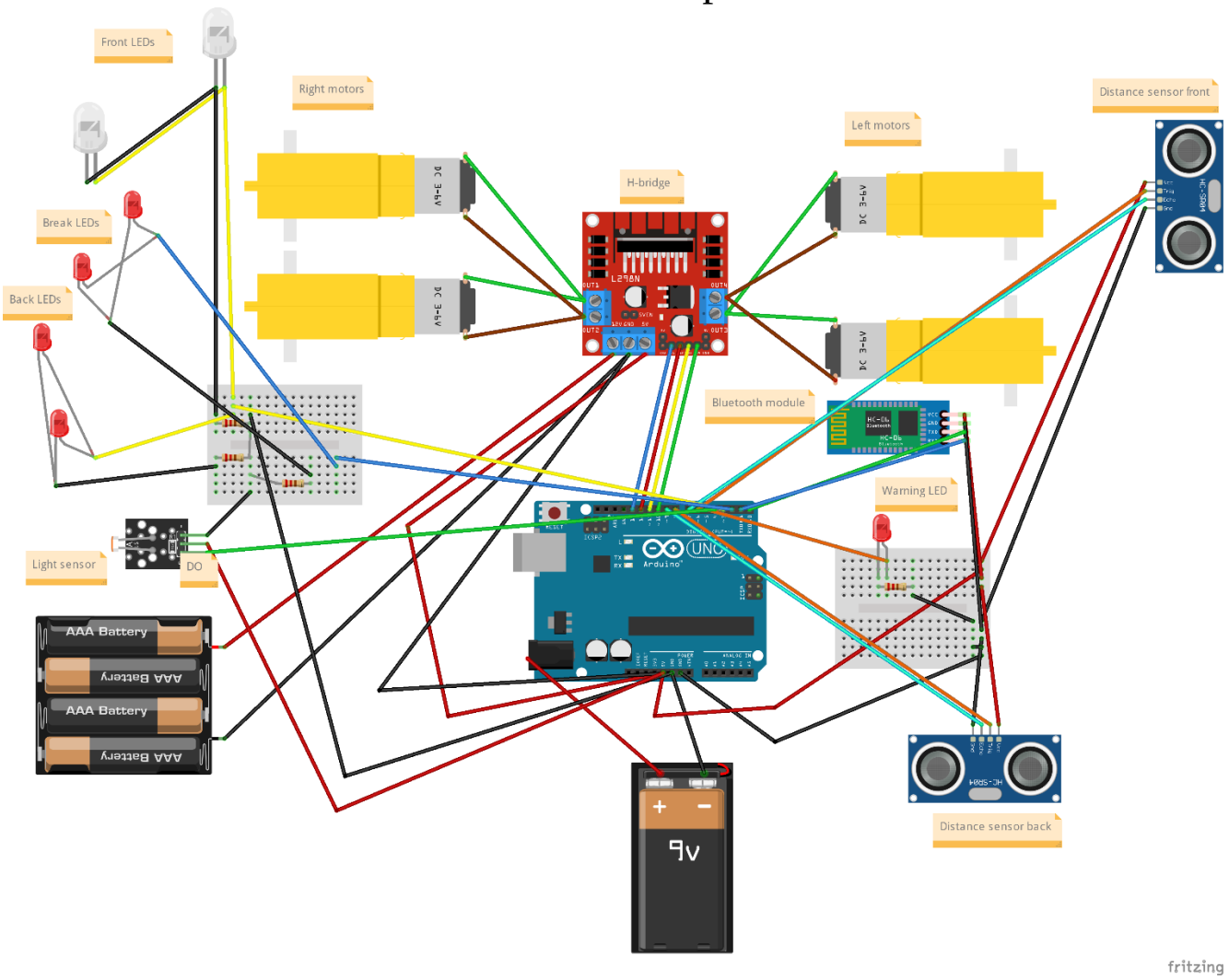
DC motors controller: The four motors are controlled by an H-bridge that is connected to Arduino. This bridge will be supplied itself with a 12V battery, as the power given by Arduino (~5V) is not sufficient to supply all motors. Depending on the state of the pins that control the bridge, this will control itself the motors (forward/backward/right/left/stop).

Bluetooth module: This module represents the communication between the user (via phone) and the microcontroller. It transmits bidirectional alerts (regarding to obstacles) and commands (for motors).

Lights control: The light intensity sensor will continuously transmit the status of the light to Arduino, that will switch on and off the four LEDs accordingly. The warning LED will be power on when one of the distance sensors will sense an obstacle to the microcontroller and the breaking LEDs will become on whenever the car stops.

Distance sensors: The two ultrasonic sensors from the back and front will continuously transmit the distance to the nearest obstacle and if this distance is below a limit, the warning LED will be power on by Arduino and a message will be sent via the Bluetooth module

Hardware and Software Implementation



(Electrical schematic)

Arduino

Arduino Uno is the microcontroller of the car. It contains the code that facilitates the control of all the components (LEDs, sensors, motors, Bluetooth module). Also, it is supplied from a 9V battery. The configuration of the pins is made in the code, as followed:

```
int trig_f=6; int echo_f=7; int trig_b=8; int echo_b=9; int warningled=5;
int lighters=3; int lightsensor=4; int right_motors_forward=13; int
right_motors_reverse=12; int left_motors_forward=11; int
left_motors_reverse=10; int stopled=2;
```

with the setup:

```
pinMode(lightsensor,INPUT);     //input from lightsensor

pinMode(trig_f,OUTPUT);    //trigger front output
```

```
pinMode(echo_f,INPUT);     //echo front input
pinMode(trig_b,OUTPUT);    //trigger back output
pinMode(echo_b,INPUT);     //echo back input
pinMode(left_motors_forward,OUTPUT);   //left motors forward
pinMode(left_motors_reverse,OUTPUT);   //left motors reverse
pinMode(right_motors_forward,OUTPUT);   //right motors forward
pinMode(right_motors_reverse,OUTPUT);   //right motors reverse
pinMode(warningled,OUTPUT);
pinMode(lighters,OUTPUT);
pinMode(stopled,OUTPUT);
```

This part of the code highlights the connections made between Arduino and the peripheric devices.

DC motors controller

The four motors are controlled by the Dual H-bridge motor controller (L298N). This is connected to Arduino at the pins `13,12,11,10`, each pin controlling the forward and backward movement of the motors (3-6V DC motors). As it can be seen in the schematic, the 2 motors from each side have the positive and negative pins connected together to an output of the H-bridge, meaning that they will both move in the same direction. The H-bridge practically controls the direction of the motors, as followed, for example if it sees `HIGH` on pin `13` (which is `input 1` of the bridge), it will put `high` level of voltage on the `OUT01`. Consequently, if it sees `LOW` on pin `12` (`input 2`), it will put `low` level of voltage on `OUT02` [2] and in that way the motors from the right will be polarized in such a way that they will create a forward movement of the right wheels. The same happens for backwards movement. For turning left/right, the motors from right/left will be forward polarized. For each movement, the user has to introduce a letter on his device that is connected through Bluetooth (`F` – forward, `B` – backward, `R` – right, `L` – left, `S` – stop). An example of the implementation in code is presented below: (forward movement, for the other directions, see the code from the annex)

```
digitalWrite(right_motors_reverse,LOW);
digitalWrite(left_motors_reverse,LOW);
digitalWrite(left_motors_forward,HIGH);
digitalWrite(right_motors_forward,HIGH);
```

To the axe of each DC motor is attached a wheel, without any other intermediate attachments. The H-bridge is also supplied supplementary by a 12V battery, as the output power given by Arduino (~5V), is not sufficient to supply the four motors and thus affecting the correct functioning of the car.

Bluetooth module

For the Bluetooth communication between the user's device and Arduino, a dedicated module is used (HC-05 Bluetooth module via the app Serial Bluetooth Terminal). It is supplied with 5V form Arduino, and the communication is done through the pins `RX,TX`. The connection with the external device is always checked using the function `Serial.available()` and `Serial.read()` is used to get the information concerning the direction introduced by the user. The user also receives messages about direction movements and obstacle detection (done in code with the function `Serial.println("…")`). Communication with user's device realized with the code below where variable `t` keeps the information got from user (is of type `char`):

```
if(Serial.available()){//establish Bluetooth connection
  t = Serial.read(); }//get the direction introduce (permanant change)
```

Lights control

Firstly, there is a red "warning LED" that signalizes to the user that an obstacle has been detected (it relates to Arduino on pin `5`). If any of the 2 ultrasonic sensors detect an obstacle within the specified safety-distance, a high level will be put on pin 5, which turns on the LED. The LED is also put in series with a resistor of 220 Ohms (connected to the cathode) (from calculations, $V_{Arduino}$~$6V$ $(because\ of\ this\ configuartion\ not\ 5V)$, $I_{Arduino} = 20mA$, V_{redLED}~$1.6V > VTh$, $R = 220\Omega$ (V_R~4.4V)). It is mounted on the top of the car.

Secondly, there are two red LEDs in the back of the car that are on if the car stops (because of an obstacle or if commanded by the user). Their anodes are both connected to pin `2` and their cathodes are connected together and in series with a resistor of 220 Ohms (from calculations, the current passing through each LED will be ~10mA and the voltage in their cathode will be ~4.4V, as the current of 20mA will pass through the resistor).

The two white LEDs from the front and the two red LEDs from the back (light LEDs) are on if the light intensity sensor signalizes dark in the medium. This sensor is based on a photoresistor (that is orientated upwards, "to the sky") whose resistance gets very high when exposed to dark (meaning that it will put `HIGH` level on the digital output of the light sensor) and very low when exposed to light [3]. The sensor is connected to Arduino on pin `4` through which the continuous interpretation is done by code (powering on or off the lights) and supplied with 5V. The LEDs are connected in the same manner as the breaking LEDs (the 2 from the front/back are connected together to pin `4` and their cathodes in series with a 220 Ohm resistor – see annex for explanations).

The sequence of code from below highlights how the status of the light sensor is read (`digitalRead()`) and then interpreted resulting in powering on.ogg the LEDs:

```
int light=digitalRead(4);
  if (light==HIGH) {
    digitalWrite(lighters,HIGH); } else { digitalWrite(lighters,LOW); }
```

Distance sensors

The two ultrasonic sensors (HC-S04) (front/back) are supplied with 5V. They continuously measure the distance to the nearest object and based on this measurement, Arduino will decide (based on the code) to stop the car or not. The ultrasonic sensor practically will emit an ultrasonic wave which will reflect on the object in the front of the sensor. Knowing the speed of sound and measuring the distance the wave travels from the moment it has been sent, then reflected, and then received back to the sensor, the distance can be calculated [5]. This is done in the code as followed (example for distance in the front. Similar case for distance in the back):

```
digitalWrite(trig_f, LOW); delayMicroseconds(2);

digitalWrite(trig_f, HIGH); delayMicroseconds(10);

digitalWrite(trig_f, LOW);

duration1 = pulseIn(echo_f, HIGH);

distance1 = (duration1 / 2) / 29;//calculate distance to front obstacle
```

First, the `trigger(trig_f)` level has to be put `LOW` to ensure that there is no previous other level for the sensor. Then, we put it `HIGH`, meaning that a sound wave is sent. After a certain delay (10ms, enough time to ensure that the wave doesn't get back in that time), `trigger` level is put back to `LOW`. The duration is measured by waiting the `echo(echo_f)` level to be `HIGH` and measuring thus the time it takes for the wave to come back (using the function `pulseIn()`; waits till the `echo` signal changes to `HIGH`, wave received). Knowing the speed of sound (340 m/s -> 29 ms/cm), the distance can be calculated by considering half of the distance traveled, so that the distance in cm is the time measured (in ms)/29 (ms/cm)/2.($d = v * t => d_{cm} = t_{ms}/2 * v * 10^{-2}$). The condition put to stop the car is a distance to the obstacle of 30cm (or below). The code below highlights how the distance get from the sensor from the front (similar case for the back) is interpreted and how communication with the user is achieved (variable `ct1` is used to control the rate of messages sent to the user (1/detected obstacle) and `currentstate` keeps the current direction of movement of the car)

```
if (distance1 < 30){
  if (currentstate == 'F'){
    /*stop the motors*/
    digitalWrite(stopled,HIGH);//turn on the breaking LEDs
    digitalWrite(warningled,HIGH);//turn on the red warning LED
    if (ct1==0){//control the rate of messages sent to the device
      Serial.println("Obstacle in front!");
      ct1=1; } } } else { digitalWrite(warningled,LOW); ct1=0; }
```

Conclusions

The project implements thus a prototype for a car controlled via Bluetooth and sensor that help the user to drive the car. Ultrasonic distance sensors for distance measuring are effective through the fact that they highlight an easy way for distance measuring by making use of a sound wave and then calculating the distance. The light intensity sensor makes use of a photoresistor that will signal the presence of light and thus powering on or off the LEDs. And the H-bridge gives through its structure power to the motors according to the signals it receives from the microcontroller (commands given by the user via Bluetooth).

So, it can be said that this implementation is quite straight forward and easy to understand, as it uses simple modules and a clear code for giving and interpreting commands. For sure there can be made improvements to the project, such as adding a sensor for interpreting the colors of a traffic light or controlling the speed of the car.

In my opinion, this project highlights in a proper manner the functioning of a remote-controlled car with some elements of “autonomy” (collision avoiding and automated light control), offering the users a satisfying experience by seeing on their own how remote control works and how handling some autonomous features is made.

References

1. https://support.arduino.cc/hc/en-us/articles/360018922259-What-power-supply-can-I-use-with-my-Arduino-board-https://www.sciencedirect.com/topics/engineering/pulse-width-modulation
2. L298 Dual H-Bridge Motor Driver IC Pins & Working (how2electronics.com)
3. https://eepower.com/resistor-guide/resistor-types/photo-resistor/#
4. https://projecthub.arduino.cc/samanfern/c71cd04b-79fd-4d0a-8a4b-b1dacc2f7725
5. https://www.instructables.com/Arduino-Ultrasonic-Sensor-HC-sr04-LEDs-Distance-Me/
6. https://www.sigmanortec.ro/ (for components)
7. https://play.google.com/store/apps/details?id=de.kai_morich.serial_bluetooth_terminal&hl=ro&gl=US&pli=1 (the app)