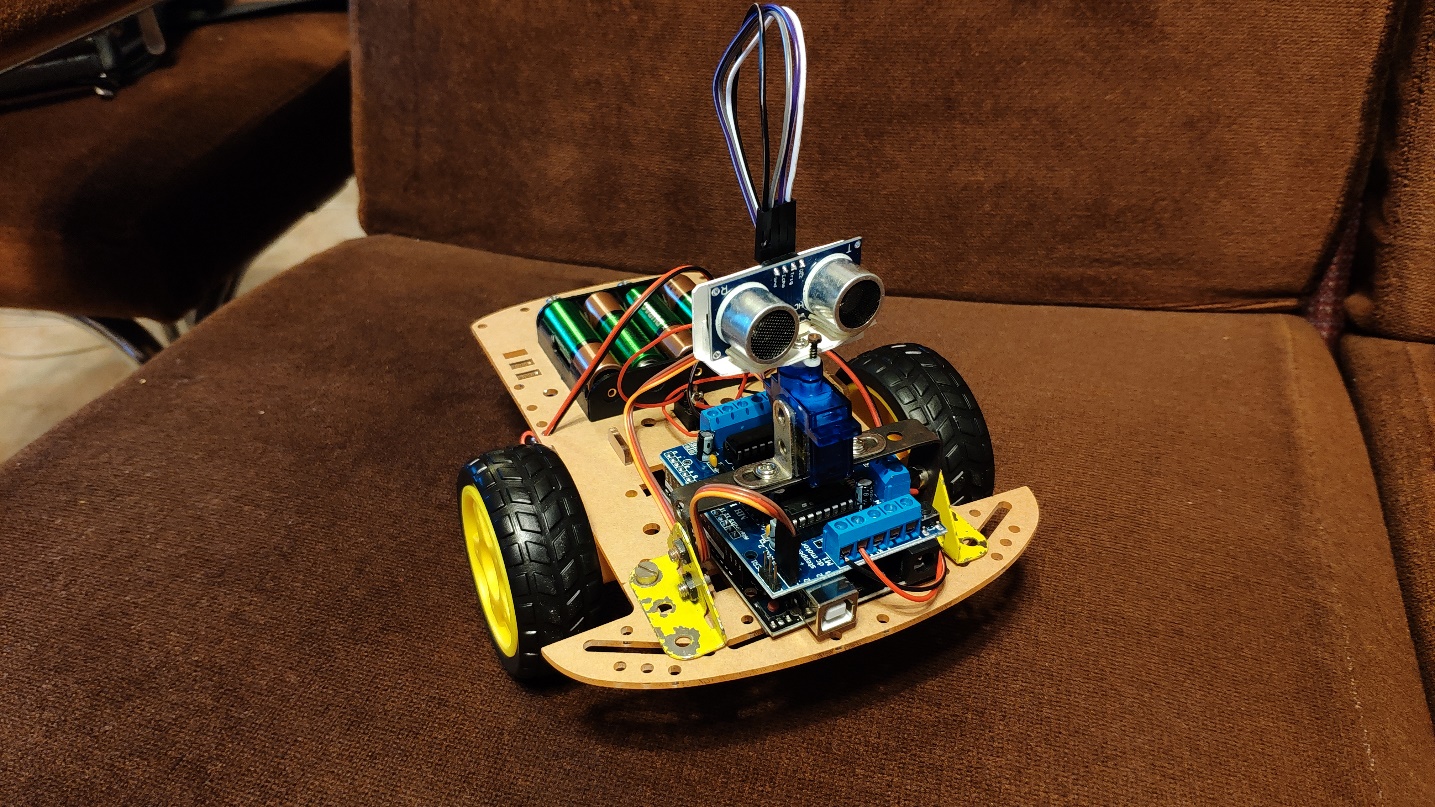
**Проектна работа за НП „Обучение за ИТ кариера“**

Робот, който заобикалящ препятствия



Изготвил:

Борис Радев

**Съдържание**

1. Въведение
2. Описание на Робота
   1. Блок схема
   2. Електрическа схема
3. Списък на съставните части
4. Сорс код – функционалност на робота
5. Заключение

Въведение

Роботът има за цел да се движи самостоятелно в среда с препятствия и да ги заобикаля. Трябва да може да излезе успешно от лабиринт, без да събаря или да се блъска в стените на лабиринта. С помощта на ултразвуков сензор да разпознава къде има препятствия и къде не. На база на получените резултати определя на къде трябва да тръгне. Роботът е с формаta на триколесна кола с много просто устройство. Серво-мотор задвижва ултразвуковия сензор наляво и надясно. Така робота обработва възможните пътища, по които робота може да тръгне.

**Описание на робота**

Работата на робота е базирана на данните, които той получава от ултразвуковия сензор в предната част на шасито. При липса на преграда пред робота той се движи напред. При откриване на такава той спира и се „оглежда“ за вероятни алтернативни маршрути. В случай че няма алтернативен път в ляво или дясно на робота той започва да се движи назад докато не стигне до ново препядствие.

2.1 Електрическа схема

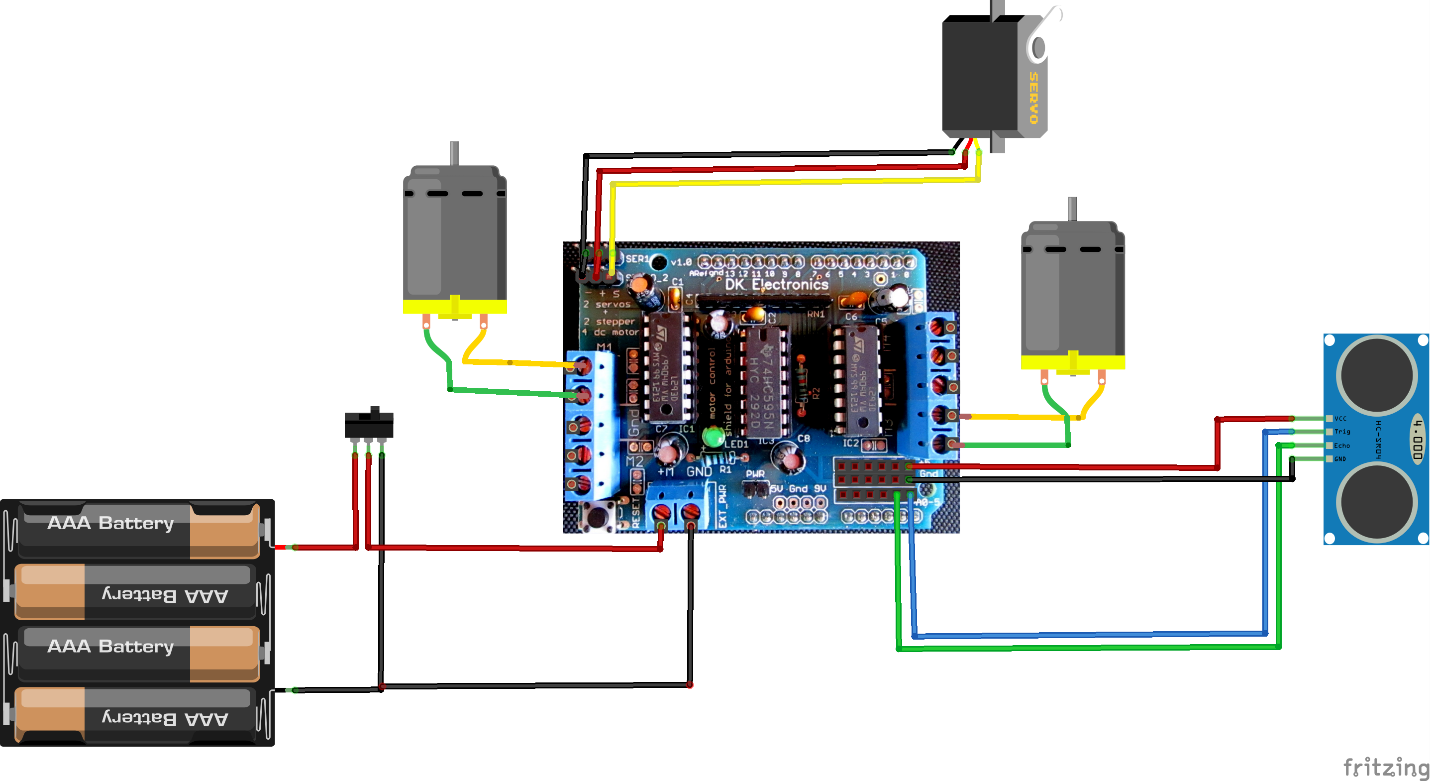


Figure 1 електрическа схема на робота

2.2 Блок схема

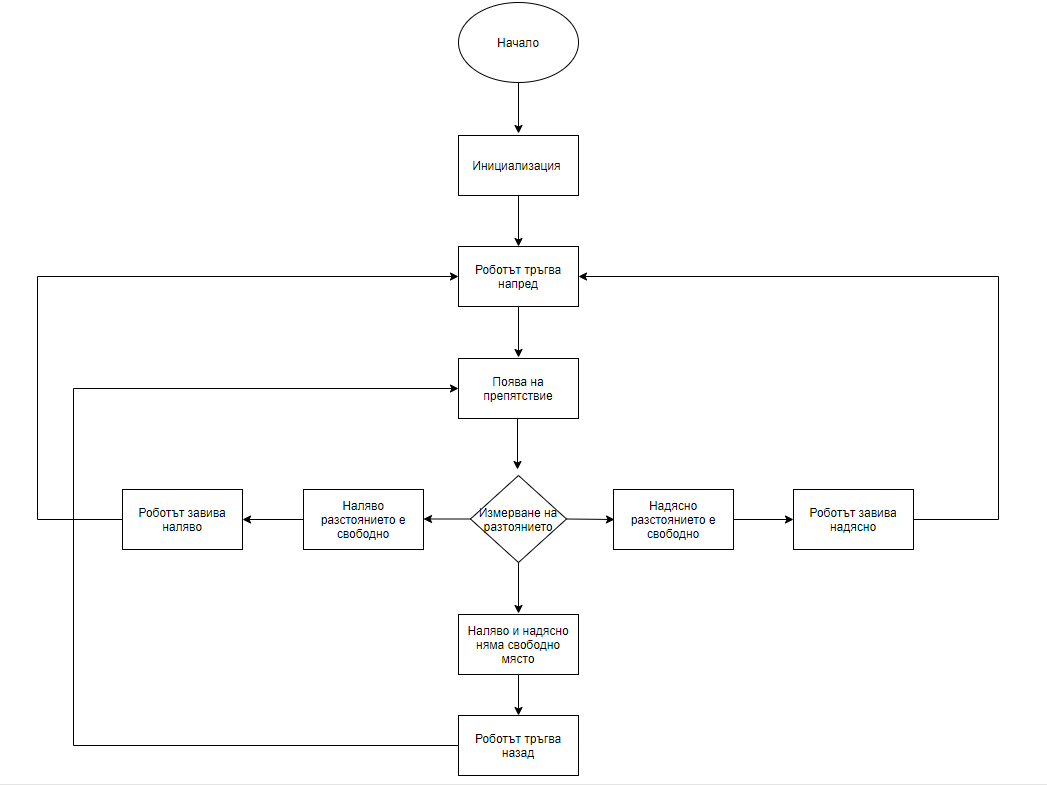


Figure 2 блок схема на програмния код

**Списък на съставните части**

1.  Elegoo Uno R3 ATmega328P ATmega16U2 Board with USB Cable Compatible with ARDUINO UNO R3
2.  MOTOR Servo/Air X L293D Motor Driver Step Motor Shield For Arduino UNO/MEGA
3.  Ultrasonic Module HC-SR04 Distance Measuring Sensor
4.  Smart Car Motor Smart Robot Car Model Robot Car Chassis Kits With Speed Encoder 1: 48
5.  Micro servo motor SG90 for RC Robot Helicopter Airplane Controls Boat Mini Servo (Control Angle 120)

**Сорс КОд – функционалност на робота**

За реализирането на програмата на робота ми бяха нужни две, външни за Arduino IDE, библиотеки:

* Motor Shield Library: <https://github.com/adafruit/Adafruit-Motor-Shield-library>
* New Ping Library: <https://bitbucket.org/teckel12/arduino-new-ping/wiki/Home#!download-install>

Motor shield Library е библиотеката, с която се управлява моторния контролер.

New Ping Library е библиотеката, с която се управлява функционалността на ултразвуковия сензор.

|  |
| --- |
| #include <AFMotor.h>  #include <NewPing.h>  #include <Servo.h>  #define TRIG\_PIN A4 // defines the trigger pin for the sonar  #define ECHO\_PIN A5 // defines the echo pin for the sonar  #define MAX\_DISTANCE 200 // defines the range of the sonar  #define MAX\_SPEED 190 // defines the maximum speed of the motors  NewPing sonar (TRIG\_PIN, ECHO\_PIN, MAX\_DISTANCE);  AF\_DCMotor motor1(1, MOTOR12\_1KHZ);  AF\_DCMotor motor2(3, MOTOR12\_1KHZ);  Servo myservo;  boolean goesForward = false;  int distance = 100;  int motorSpeed = 0; |

На редовете по-горе се инстанцират двата мотора, задвижващи колелата, и Серво-мотора, който движи ултразвуковия сензор.

|  |
| --- |
| void setup()  {  myservo.attach(9);  myservo.write(90);  delay(2000);  distance = readPing();  delay(100);  distance = readPing();  delay(100);  distance = readPing();  delay(100);  distance = readPing();  delay(100);  } |

В метода Setup() се задават пина, към който е свързан серво-мотора. Също задаваме и ъгъла на изходна позиция на мотора . В случая той е 90°, тъй като мотора има ход само 180° и то в една посока . Затова, за да може да се завърта и наляво и на дясно го слагаме в позиция, която е в средата на максималния ход. Тук ултразвуковия сензор приема първите сигнали, които по-късно ще обработим.

|  |
| --- |
| void loop() {  int distanceR = 0;  int distanceL = 0;  delay(40);  if(distance <= 15)  {  moveStop();  delay(100);  moveBackward();  delay(300);  distanceR = lookRight();  delay(200);  distanceL = lookLeft();  delay(200);  if(distanceR > distanceL)  {  turnRight();  moveStop();  }else  {  turnLeft();  moveStop();  }  }else  {  moveForward();  }  distance = readPing();  } |

В метода по-горе обработваме случаите, в които пред робота се появява препятствие, което сензорът засича. Когато преградата пред робота е на по-малко от 15см, той спира. След това се оглежда наляво и надясно. Сравнява резултатите и преценява на къде дистанцията е по-голяма. Тогава завива в дадената посока и след това отново тръгва направо. Накрая отново започва да приема данни за разстоянието пред робота докато се движи напред.

|  |
| --- |
| int lookRight()  {  myservo.write(0);  delay(500);  int distance = readPing();  delay(100);  myservo.write(90);  return distance;  }  int lookLeft()  {  myservo.write(180);  delay(500);  int distance = readPing();  myservo.write(90);  return distance;  delay(100);  } |

Това са методите, които описват процесите за „оглеждане“ на робота, в търсене на алтернативен маршрут. В случай, че робота „гледа“ надясно, той завърта ултразвуковия сензор към неутрално положение на 0°. След това получава обратен сигнал за свободното разстояние в дясно на робота. Когато роботът „поглежда“ наляво, действията които се случват са същите като тези в метода за надясно. В този случай сензорът се завърта на 180°. След това събира данни за ситуацията наляво.

|  |
| --- |
| int readPing()  {  delay(70);  int cm = sonar.ping\_cm();  if(cm == 0)  {  cm =250;  }  return cm;  } |

С този метод четем и обработваме данните за разстояние, получени от сензора. Ако разстоянието е равно на 0 робота се отдръпва от препятствието.

|  |
| --- |
| void moveStop()  {  motor1.run(RELEASE);  motor2.run(RELEASE);    }  void moveForward()  {  if(!goesForward)  {  goesForward = true;  motor1.run(FORWARD);  motor2.run(FORWARD);  for(motorSpeed = 0; motorSpeed < MAX\_SPEED; motorSpeed +=2)  {  motor1.setSpeed(motorSpeed);  motor2.setSpeed(motorSpeed);  delay(5);  }  }  }  void moveBackward(){  goesForward = false;  motor1.run(BACKWARD);  motor2.run(BACKWARD);  for(motorSpeed = 0; motorSpeed <= MAX\_SPEED; motorSpeed += 2)  {  motor1.setSpeed(motorSpeed);  motor2.setSpeed(motorSpeed);  delay(5);  }  } |

Тук описваме логиката на робота, когато се движи. Поради желанието за по-икономична полза на батериите, робота никога не тръгва на максимална скорост, а ускорява постепенно като при по натоварен терен не се достига максималната скорост. По същия начин работи и метода за движение назад, но той почти не се използва. Само в случаите когато се налага да се отдалечи от преграда.

|  |
| --- |
| void turnRight()  {  motor1.run(FORWARD);  motor2.run(BACKWARD);  delay(300);  motor1.run(FORWARD);  motor2.run(FORWARD);  }  void turnLeft()  {  motor1.run(BACKWARD);  motor2.run(FORWARD);  delay(300);  motor1.run(FORWARD);  motor2.run(FORWARD);  } |

Тук описваме методите за завиване на робота. В този случай независимо от посоката на завиване, то се случва стационарно. Докато процеса на завиване на е завършен, роботът не се движи нито напред нито назад.

Заключение

Роботът успешно открива прегради и пречки на пътя си. Успява да завива и да избира по-добрия вариант за маршрут. Недостатък е че не може да завива в движение, но за сметка на това не се блъска в препятствията. Сензорът отчита точно данните за разстоянието, а серво-мотора върти сензора в нужните посоки. Тъй като серво-моторът е ограничен да се върти само до 180° движението назад с избягване на препятствия е невъзможно. Затова хода назад е ограничен до толкова, че да само да се отдалечава от засечена преграда. След това отново тръгва напред, но в нова посока. Сензорът прави изключение с по-ниски и по-тесни предмети. Тъй като тях понякога не ги засича и се достига до лек сблъсък. Това са частни случай, които се случват рядко. Те не влияят на цялостната работа на робота.