Object Avoidance Robot

Autor: Topală Andrei-Tudor

Introducere

- Proiectul consta intr-un robot autonom care este capabil sa detecteze si sa evite obstacole.
- In momentul in care robotul intalneste un obstacol, acesta se va opri, va analiza o cale libera si isi va schimba directia de deplasare.
- Noua directie de deplasare a robotului va fi aleasa folosind un senzor ultrasonic controlat de un servomotor.

Descriere generala + schema bloc

- Pentru a controla cele doua motoare ale robotului, am folosit un modul cu driver cu punte H L298N. Acesta este controlat prin 6 pini, 3 pentru fiecare motor: ENA, IN1, IN2 si ENB, IN3, IN4. Motoarele sunt alimentate folosind un set de 4 beterii de 1.5V.
- Robotul foloseste un senzor ultrasonic de distanta pentru a detecta obstacole. In momentul in care senzorul detecteaza un obstacol la mai putin de 10cm, robotul se va opri, va roti senzorul ultrasonic astfel incat sa poata determina care este directia libera de deplasare. Rotirea senzorului ultrasonic este realizata de un servomotor.

Hardware Design

Lista de componente

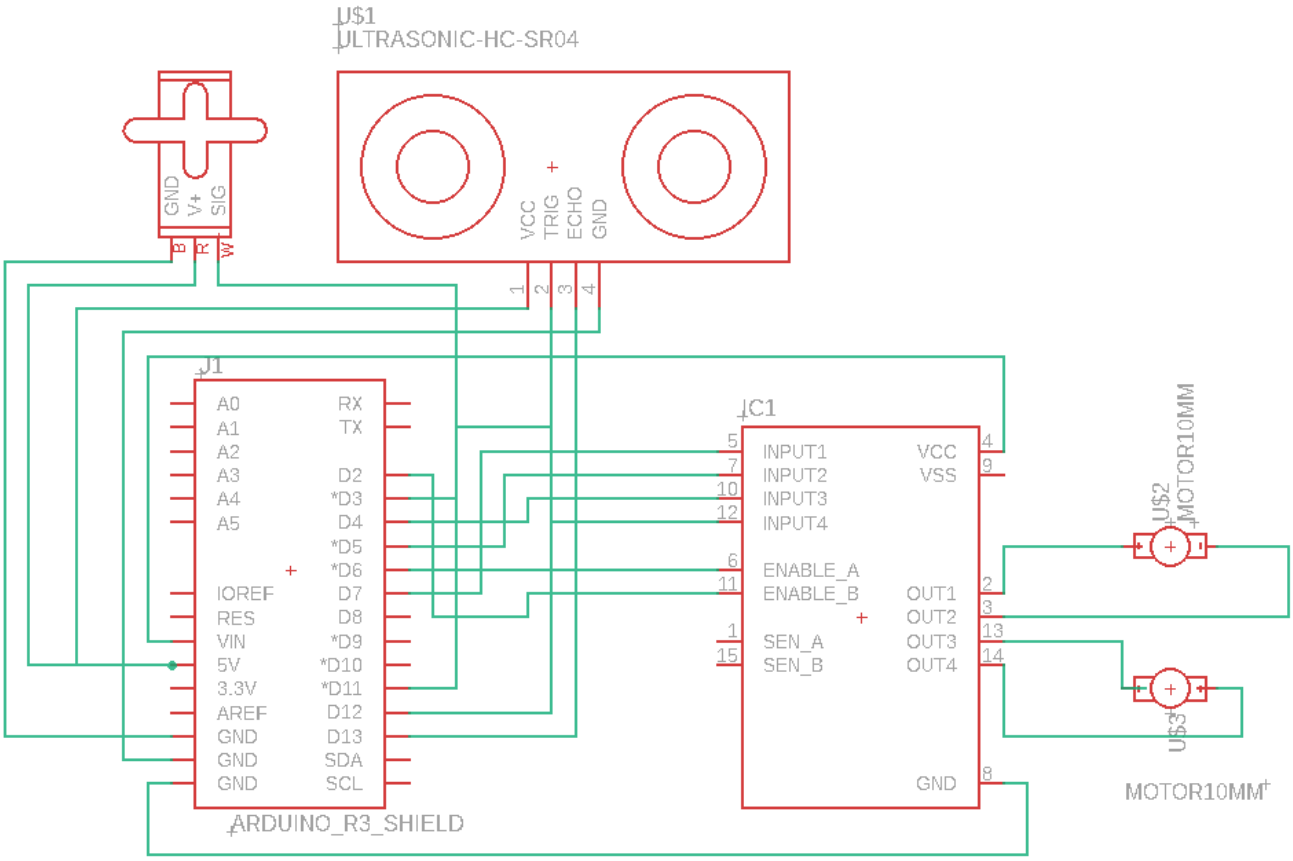
- Arduino UNO
- Senzor Ultrasonic HC-SR04
- Micro-servo motor SG90
- Sasiu + 2 motoare DC
- Driver Motor L298N
- Suport camera servomotor
- Set 4 baterii 1.5V
- Suport baterii 4xAA
- Fire tata-tata (2 seturi)

- Fire mama-tata (1 set)

Componente auxiliare pentru montaj

- Soldering Iron Kit
- Set suruburi M2-M2.5
- Banda dublu adeziva

Schema electrica



Software Design

Motoare

- Intrucat rotile nu sunt perfect aliniate, robotul nu va merge drept in momentul in care punem ambele motoare pe 5V. De aceea, am fost nevoit sa aleg valorile potrivite pentru ca robotul sa aiba o directie de deplasare dreapta.

- Motoarele sunt controlate folosind **L298N H bridge** prin 6 pini: in1-4, enA, enB.
  - enA: Controleaza viteza motorului stang prin PWM
  - enB: Controleaza viteza motorului drept prin PWM
  - in1,2: Controleaza directia de deplasare (inainte/inapoi) a motorului stang
  - in3,4: Controleaza directia de deplasare (inainte/inapoi) a mtorului drept

```
enum Direction {
  LEFT, RIGHT
};

/* Motor control pins of
 *
 * enAPin: Left motor PWM speed control
 * enBPin: Right motor PWM speed control
 * in1Pin: Left motor Forward
 * in2Pin: Left motor Backwards
 * in3Pin: Right motor Forward
 * in4pin: Right motor Backwards
 */
const int enAPin = 6;
const int in1Pin = 7;
const int in2Pin = 5;
const int in3Pin = 4;
const int in4Pin = 2;
const int enBPin = 3;

/* Set motor speed: 255 full ahead, -255 full reverse , 0 stop */
void go(enum Direction m, int speed)
{
  digitalWrite(m == LEFT ? in1Pin : in3Pin, speed > 0 ? HIGH : LOW);
  digitalWrite(m == LEFT ? in2Pin : in4Pin, speed <= 0 ? HIGH : LOW);
  analogWrite(m == LEFT ? enAPin : enBPin, speed < 0 ? -speed : speed);
}

void moveForward() {
  go(LEFT, 200);
  go(RIGHT, 180);
}

void moveLeft() {
  go(LEFT, -200);
  go(RIGHT, 0);
  delay(600);
}

void moveRight() {
  go(LEFT, 0);
  go(RIGHT, -180);
  delay(700);
}
```

```
void moveStop() {
  go(LEFT, 0);
  go(RIGHT, 0);
}
```

Servo

- Servomotorul are 5 unghiuri (2 stanga, 2 dreapta, 1 mijloc) pentru care citeste date de la senzorul ultrasonic.
- Pentru a determina care este cea mai buna directie de deplasare, se va face **media aritmetica** a celor 2 unghiuri corespunzatoare fiecarei directii.
- Servomotorul este controlat folosind biblioteca Servo.h

```
#include <Servo.h>

#define START_ANGLE 2
#define NUM_ANGLES 5

Servo servo;
/* Servo motor that aims ultrasonic sensor.
 *
 * servoPin: PWM output for servomotor
 */
const int servoPin = 11;

int centimeter = 0;

/* Servomotor angles to masure best potential path
 *
 * sensorAngle: arrya of angles
 * distance: array of distances between sensor and object for each angle
 */
unsigned char sensorAngle[NUM_ANGLES] = {20, 60, 90, 130, 160};
unsigned int distance[NUM_ANGLES] = {0};

/* Moves ultrasonic sensor in 5 different angles
 * to get a better view of the field.
 *
 * @return: Direction (LEFT/RIGHT) for the next path
 */
unsigned int useServo() {
  static unsigned char angleIndex = START_ANGLE;
  static signed char step = 1;
  int maxDistance = 0, nextDirection = 0;
  int distanceLeft = 0, distanceRight = 0;

  for (int i = 0; i < 8; ++i) {
    angleIndex += step;
```

```
    if (angleIndex == NUM_ANGLES - 1) {
      step = -1;
    } else if (angleIndex == 0) {
      step = 1;
    }
    servo.write(sensorAngle[angleIndex]);
    distance[angleIndex] = readUltrasonicDistance() / CM_FACTOR;

    delay(500);
  }

  /* Distance mean for both directions. */
  distanceLeft = (distance[0] + distance[1]) / 2;
  distanceRight = (distance[3] + distance[4]) / 2;

  return (distanceLeft > distanceRight) ? LEFT : RIGHT;
}
```

Senzor ultrasonic

- Seonzorul citeste date din mediul exterior folosind sunete ultrasonice.
- Acesta intoarce numarul de microsecunde parcurs de sunet de la senzor pana la obiect si inapoi.
- Pentru a putea converti din microsecund in cm, putem sa ne folosim de urmatoarea formula: Distance = (Time x SpeedOfSound) / 2

```
#define CM_FACTOR 58.00

/* Ultrasonic Module pins
 *
 * triggerPin: 10 microsecond high pulse causes sharp sound , wait 50 us
 * echoPin: Width of high pulse indicates distance
 */
const int triggerPin = 12;
const int echoPin = 13;

long readUltrasonicDistance() {
  /* Clear the trigger */
  pinMode(triggerPin, OUTPUT);

  digitalWrite(triggerPin, LOW);
  delayMicroseconds(2);
  digitalWrite(triggerPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(triggerPin, LOW);
  pinMode(echoPin, INPUT);

  return pulseIn(echoPin, HIGH) / CM_FACTOR;
}
```

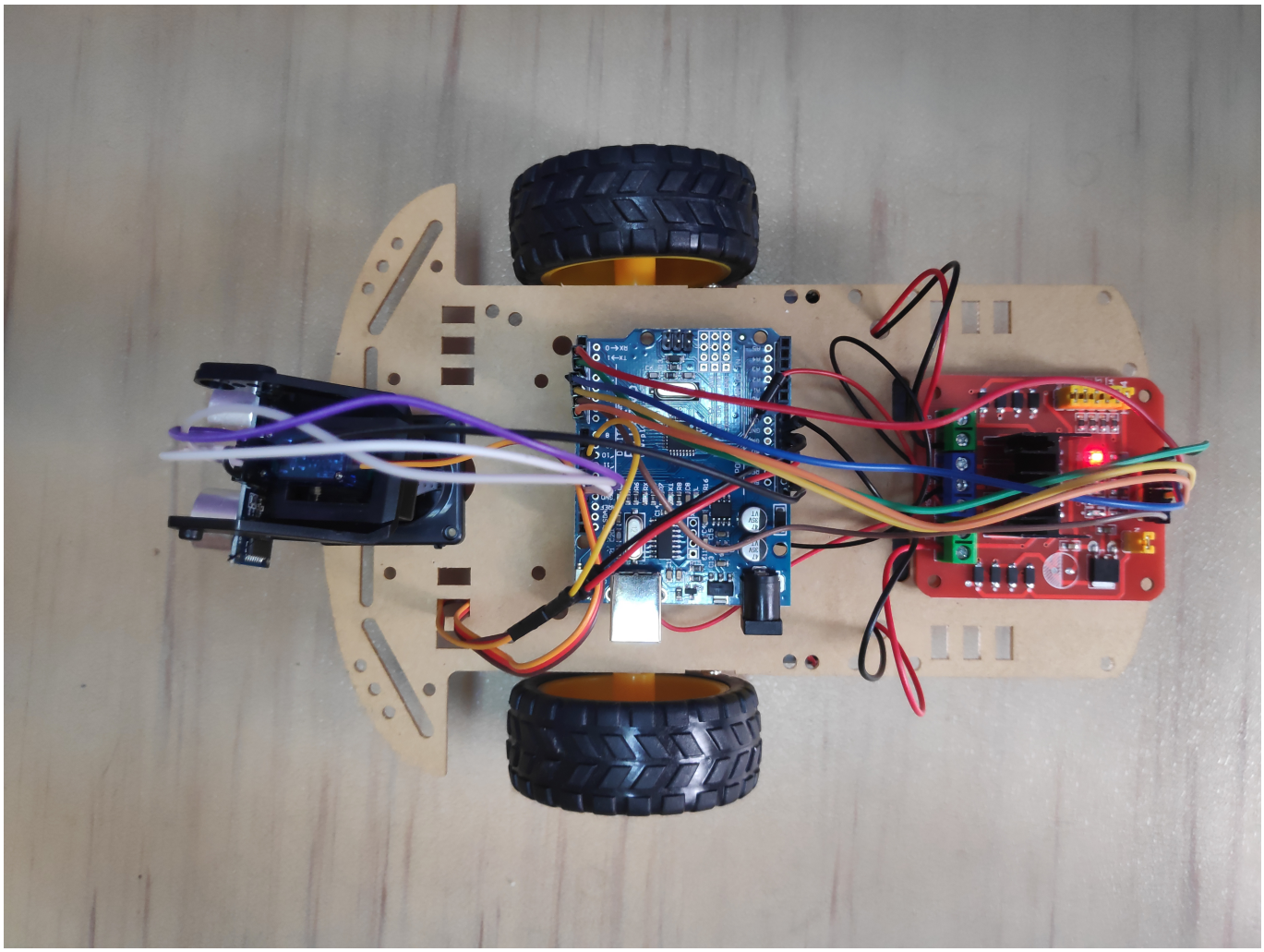
Loop

- Robotul merge inainte pana detecteaza un obiect.
- In momentul in care a detectat un obiect la mai putin de 10cm, se opreste, isi schimba directia de deplasare si isi continua drumul.

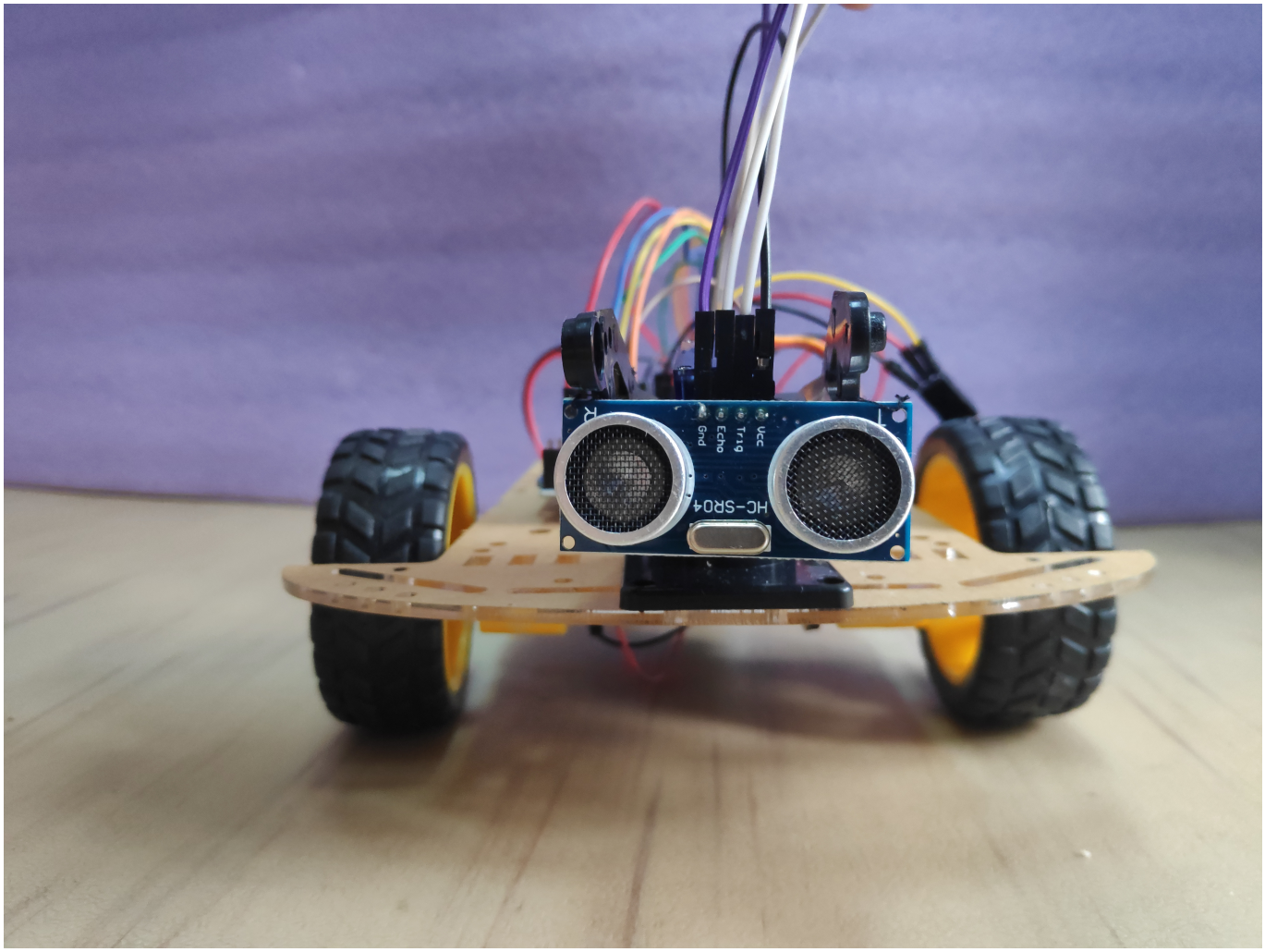
```
#define MIN_COLLISION_DISTANCE 10

void loop() {
    centimeter = readUltrasonicDistance();
    if (centimeter < MIN_COLLISION_DISTANCE) {
      moveStop();

      int nextDirection = useServo();

      if (nextDirection == LEFT) {
        moveLeft();
      } else {
        moveRight();
      }
      moveStop();
    } else {
      moveForward();
    }
}
```

Rezultate

Poze dispozitiv

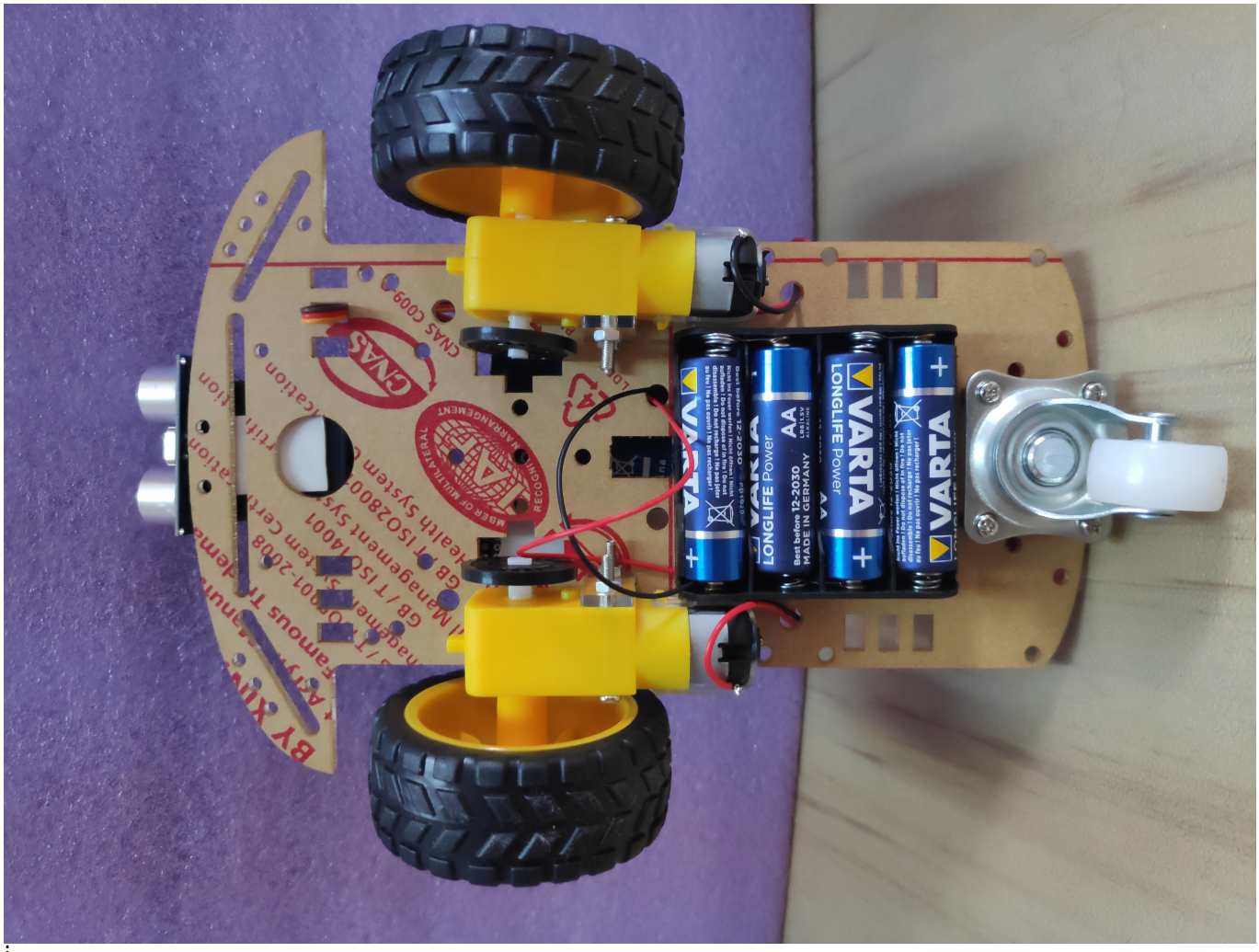


ta de la baterii prin pinul Vin si este conectata la driver-ul de motoare, la servomotor si la senzorul ultrasonic.



s

at pe un suport de camera controlat de un servomotor capabil sa se roteasca pe planul orizontal.



i
pite pe spatele sasiului pentru a economisi spatiu. Bateriile alimenteaza atat motoarele, cat si microcontrolerul.

Videoclip de prezentare

Download

Cod Sursa (arhiva)

Jurnal

- 17.05.2021: Testarea separata a fiecarei componente pentru a observa modul de functionare al elementelor
- 19.05.2021: Realizarea schemei electrice, lipirea firelor pe motoare si asamblarea componentelor pe sasiu

- 20.05.2021: Calibrarea directiei de deplasare a robotului folosind un Alimentator la priza de 1A pentru a nu consuma baterii (o alegere care s-a dovedit sa fie nefericita spre final)
- 22.05.2021: Testarea per ansamblu al tuturor functionaliatilor (tot conectat la priza)
- 29.05.2021: Alimentarea robotului folosind baterii.
  - Initial am incercat sa alimentez placuta la o baterie de 9V, dar din pacate aceasta nu avea suficienta putere sa faca motoarele sa se miste.
  - A doua varianta a fost sa alimentez modulul de drivere folosind 4 baterii de 1.5V, iar placuta la bateria de 9V. Totusi, bateria de 9V .nu a fost nici de aceasta data capabila sa alimenteze cum trebuie placuta Arduino. Existau momente in care tensiunea pe baterie scadea sub 7V, iar placuta se reseta.
  - A treia varianta (si cea finala) a fost sa folosesc cele 4 baterii de 1.5V si pentru alimentarea in acelasi timp a motoarelor si a placutei Arduino (prin pinul **Vin**). Problema acestei solutii este faptul ca bateriile nu mai au suficienta putere pentru a da viteza marita motoarelor. Astfel, toate calibrarile anterioare pentru deplasarea robotului au devenit inutile si a trebuit sa reiau munca de la capat.
- 30.05.2021: Finalizare proiect: filmat, fotografiat, descriere ocw

Concluzii

- Proiectul a fost interesant, am avut destule lucuri de invatat din el: de la lipit fire pana la facut debugging folosind multimetrul. De asemenea, am invatat sa rezolv problemele de hardware (roti nealiniate) folosind software-ul.

Bibiliografie/Resurse

- ATmega328P Datasheet (ADC & Timers)
- L298 H-Bridge Datasheet
- Servo.h library

Export to PDF

From:
http://ocw.cs.pub.ro/courses/ - **CS Open CourseWare**

Permanent link:
http://ocw.cs.pub.ro/courses/pm/prj2021/avaduva/objectavoidancerobot

Last update: **2021/06/02 14:26**