RoboterTraktor

Auer, Klausner und Kostenzer

5BHW

PPM

Inhalt

Aufgabenstellung:

Die Idee dieses Projektes war es einen Roboter zu bauen, der sozusagen ein Feld abfahren kann. Der Roboter soll deshalb geradeaus fahren können, das Ende des Feldes erkennen und dort stehen bleiben. Außerdem soll er Rechts- und Linkskurven fahren können.

Als Wunschkriterien sollte der Roboter Hindernisse erkennen und vor diesen stehenbleiben oder ausweichen können. Außerdem soll er durch Abfahren des Feldes seine Fläche berechnen können. Ein weiterer Zusatz wäre, dass der Roboter bei schwachen Akku seine Ladestation automatisch aufsucht.

Umsetzung:

Für den Antrieb des Roboters wurden zwei Servo-Motoren verwendet.

Um zu überprüfen, ob der Roboter am Rand des Feldes steht und ob er eine Rechts- oder Linkskurve fahren muss, wurden drei Lichtsensoren verwendet. Diese reagieren auf helle und dunkle Oberflächen.

Damit der Roboter Hindernisse erkennt und ausweichen kann, wurde ein Ultraschallsensor verwendet, der den Abstand von Sensor zum Objekt misst.

Für die Programmierung des Roboters bzw. der Komponenten wurde ein Arduino verwendet. Als Stromquelle diente vorerst der PC, später ein Akku.

Programm:

```
#define motorLinks 6
#define motorRechts 5
#define sensorLinks 10
#define sensorRechts 11
#define trig 8
#define echo 7

#include <Servo.h>

volatile boolean kurveRechts;
volatile boolean kurveLinks;
volatile boolean anfang;
volatile boolean feldende;
volatile char letzteKurve;

Servo rechts;
Servo links;

volatile unsigned long alteZeit=0, entprellZeit = 1000;
```

```
void setup()
{
  Serial.begin(9600);
  attachInterrupt(0, ISRendeErkennen, FALLING);
  rechts.attach(motorRechts);
  links.attach(motorLinks);
  pinMode(trig, OUTPUT);
  pinMode(echo, INPUT);
  anfang = true;
  feldende = false;
  kurveRechts = false;
  kurveLinks = false;
  stoppen();
}

void loop()
{
  if(!feldende)
  {
    //Der Roboter soll stoppen, wenn ein Hindernis zwischen 1 und 10cm steht
    if(entfernungMessen()<=10 && entfernungMessen() >= 1) // 1cm weil: Wenn kein Hindernis vorhanden ist, dann
                                                          //beträgt entfernungMessen einen Wert zwischen 0 und 1
    {
      stoppen();
      delay(1000);
```

```
    }else
    {
    fahren();
    }
  }else
  {
    if(kurveRechts && !kurveLinks)
    {
     //Serial.println("Rechtskurve");
     rechtsKurveFahren();
     delay(1000);
    }else if(kurveLinks && !kurveRechts)
    {
      //Serial.println("Linkskurve");
      linksKurveFahren();
      delay(1000);
    }else
    {
     //Serial.println("Stoppen");
     stoppen();
     delay(5000);
    }
    feldende = false;
  }
}
```

```
void fahren()
{
  rechts.write(0);
  links.write(180);

}

void stoppen()
{
  rechts.write(89); //Durch Probieren kamen wir auf diese Werte
  links.write(99);  //Angegeben sind eigentlich für Stillstand 90
}

void linksKurveFahren()
{
  for(int i = 110; i >= 105; i--)
  {
    rechts.write(0);
    links.write(i);
    delay(500);
  }
}
```

```
void rechtsKurveFahren()
{
  for(int i = 80; i <= 83; i++)
  {
    rechts.write(i);
    links.write(180);
    delay(500);
  }
}

//Interrupt-Routine
void ISRendeErkennen()
{
  if((millis() - alteZeit) > entprellZeit) //Entprellen des Sensors
  {
  feldende = true; //Erkennung des Feldendes
  if(anfang)
  {
    //Wenn der Sensor links ein Signal bekommt, soll der Roboter eine Rechtskurve fahren
    // weil durch den Interrupt vorne das Feldende schon gegeben ist
    if (analogRead(A2) <= 200) //A2 entspricht Sensor links
    {
      kurveLinks = false;
      kurveRechts = true;
      letzteKurve = 'r';
    }
```

```
    //Gleich wie oben, nur mit rechtem Sensor und Linkskurve
    else if(analogRead(A0) <= 200) //A0 entspricht Sensor rechts
    {
      kurveRechts = false;
      kurveLinks = true;
      letzteKurve = 'l';
    }
    //Wenn Sensor rechts und links ein Signal bekommen, dann soll er stehen bleiben
    else if(analogRead(A2) <= 200 && analogRead(A0) <= 200)
    {
      kurveRechts = false;
      kurveLinks = false;

    }else //Default-Wert ist eine Rechtskurve
    {
      kurveLinks = false;
      kurveRechts = true;
      letzteKurve = 'r';
    }
    anfang = false; //Weil dies nur beim ersten Interrupt passieren soll
```

```
  }else //Für alle anderen Fälle (ausgenommen der erste Interrupt)
   {
     if (analogRead(A2) >= 600 && analogRead(A0) >= 600) //Abwechselndes Kurvenfahren
     {
       //kurveRechts = !kurveRechts;
       //kurveLinks = !kurveLinks;

       if(letzteKurve == 'r')
       {
         kurveLinks = true;
         kurveRechts = false;
         letzteKurve = 'l';
       }else if(letzteKurve == 'l')
       {
         kurveLinks = false;
         kurveRechts = true;
         letzteKurve = 'r';
       }
     }else //Ende des Feldes
      {
       kurveRechts = false;
       kurveLinks = false;
      }
   }
   alteZeit = millis();
  }
}
```

```
//Ultraschallsensor
double entfernungMessen()
{
  double duration;

  digitalWrite(trig, LOW);
  delayMicroseconds(2);
  digitalWrite(trig, HIGH);
  delayMicroseconds(10);
  digitalWrite(trig, LOW);
  duration = pulseIn(echo, HIGH);

  return duration/29/2;

}
```

Schwierigkeiten:

Ansteuerung der Servomotoren

Im ersten Versuch wurden die Servomotoren mittels eines PWM-Signals angesteuert, dieses ließ jedoch die Servomotoren teilweise abwechselnd und/oder gar nicht drehen. Weiters müssten die Servomotoren neu verdrahtet werden, um mit beiden Rädern in dieselbe Richtung zu drehen! Deshalb wurde eine eigene Klasse für die Motoren verwendet, die es ermöglichte die Motoren zugleich zu bewegen. Die Klasse „Servo“ ermöglichte es mit dem Befehl *.write(ZAHL)* die Motoren jeweils beidseitig in die richtige Richtung antreiben zu lassen.

Prellen der Sensoren

Der Roboter schien die ISR zufällig oft aufzurufen, was uns nach längerer Recherche dazu führte, dass wir die Sensoren für die ISR entprellen müssen. Durch Einfügen eines Entprellvorganges vor der eigentlichen ISR wurde der Fehler des „zufälligen“ Auslösens behoben. Obwohl dieser Fehler recht leicht zu beheben war, wurde dieser sehr lange und intensiv gesucht, da anfangs das Prellen der Sensoren für unwahrscheinlich gehalten wurde. Nach dem sogenannten Ausschlussverfahren wurde die ISR von innen nach außen genau untersucht und am Ende der einfache Fehler detektiert und mit einer einfachen Zeitverzögerung von wenigen Millisekunden gelöst.

Wechsel der Kurvenrichtung

Die Lösung dieses Problems ist ebenso simpel wie die des Prellens, jedoch wurde dieser Fehler infolge der Suche des Fehlers vom Prellen entdeckt. Bei diesem Fehler ging es darum, dass sich die Zustände von Links- auf Rechtskurve immer mit einem NICHT (!) geändert haben. Dies war bei der einfachen Überlegung zwar richtig, falls jedoch einmal ein anderer Zustand als „Rechts- oder Linkskurve“ auftrat, kamen Zustände heraus, welche nicht zugelassen waren und somit den Roboter stoppten. Mittels Vergleiche einer Char-Variablen wurde dieser Fehler behoben!

Farbsensoren und Ultraschallsensor

Die beiden Sensortypen arbeiten mit PWM-Signalen, d.h. Abfragen mit HIGH und LOW werden nur selten erfüllt, da Werte zwischen 0V und 5V ausgegeben werden können. Diese Tatsache wurde uns kurz zum Verhängnis, da der Ultraschallsensor meist einen Wert von 0 erreicht, weshalb bei der Abfrage neben einer oberen auch eine untere Grenze gegeben ist. Derselbe Fakt gilt für die Farbsensoren!

Aufbau:

Verwendete Komponenten:

- Arduino
- 2 Servo-Motoren
- Ultraschallsensor
- Lichtsensor
- Kabel
- Steckbrett

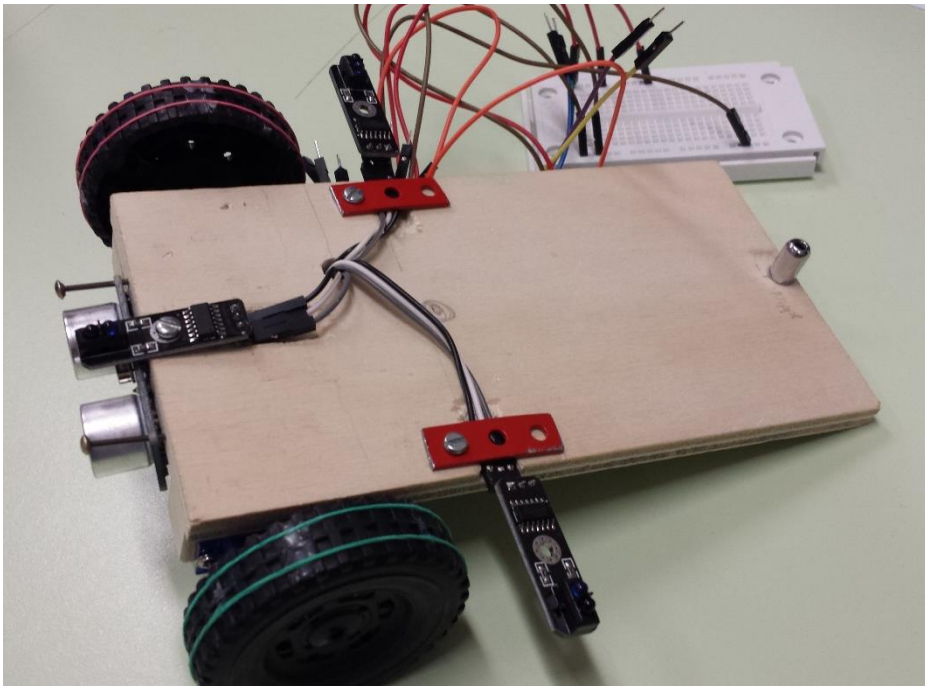
Das Ziel war es den Roboter so leicht wie möglich zu bauen und deshalb wurde eine dünne Holzblatte verwendet, an der die Servo- Motoren befestigt wurden.

Als Räder wurden leichte Kunststoffräder verwendet, die auf die Servomotoren aufgesteckt wurden.

Auf der Unterseite wurde hinten ein „Nippel“ aufgeklebt, damit der Roboter nicht das ganze „Hinterteil“ mitschleifen muss.

Als nächstes wurden die Lichtsensoren auf die Unterseite der Holzblatte angeschraubt und die Kabel durch ein Loch in der Platte gelegt.

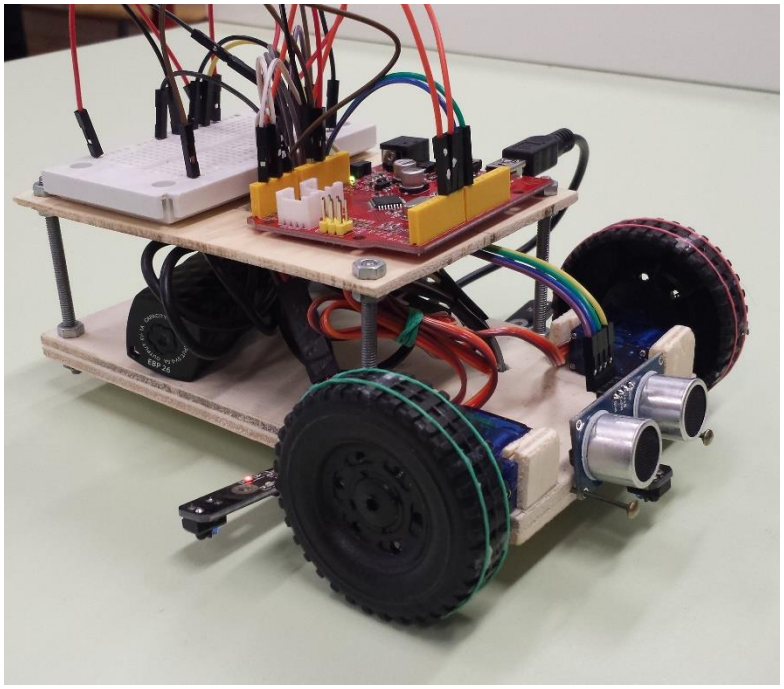
Auch der Ultraschallsensor wurde an der Vorderseite mittels zwei Nägel befestigt.



Da der Arduino nicht so viele Steckplätze hat, wurde ein kleines Steckbrett verwendeten.

Aus Platzgründen wurde eine „zweite Ebene“, auch aus Holz, auf die erste Holzplatte mittels Gewindestangen gesetzt(siehe Bild). Somit ist auch noch genügend Platz für einen Akku. Die Kabel wurden durch einen Spalt in der „oberen Ebene“ verlegt, für ein sauberes und ordentliches Erscheinungsbild.

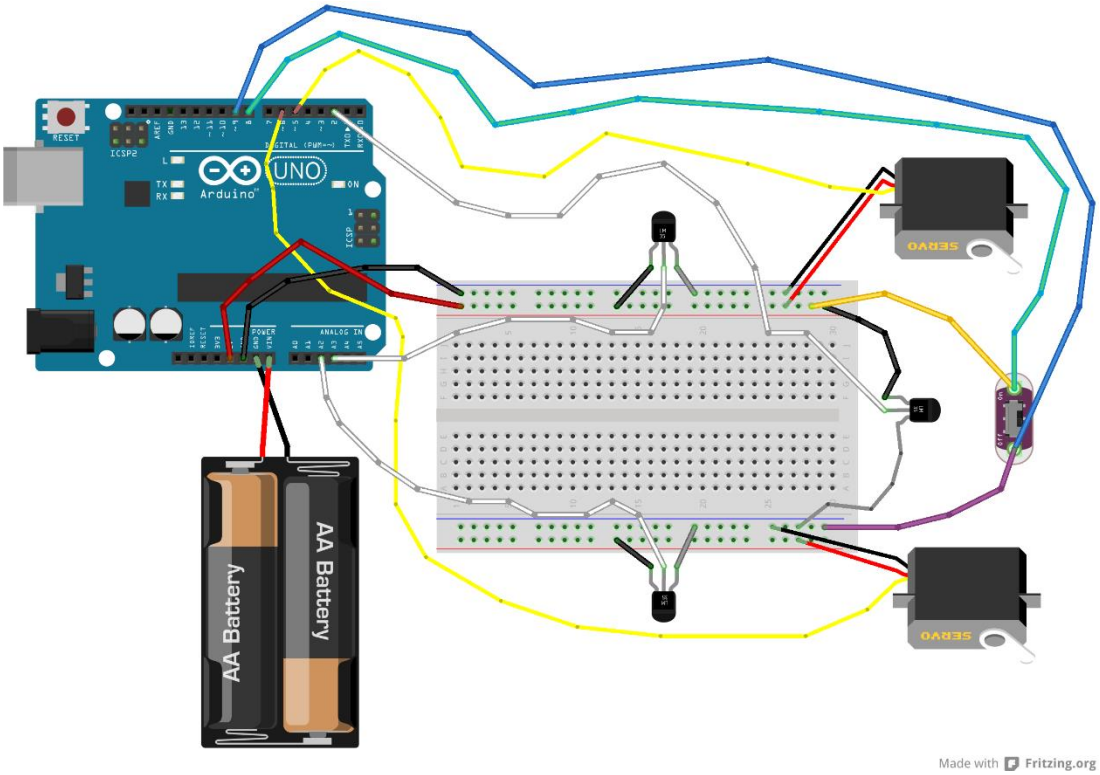
Da die Räder auf glatten Boden durchdrehen, wurden zwei Gummibänder für mehr Gripp angebracht.



Natürlich wurde auch beim Design nicht gespart und der Kreativität wurde freien Lauf gelassen. Um das Erscheinungsbild zu verschönern, wurden auf beiden Seiten Grafiken angebracht.

Schaltplan:

Im folgenden Schaltungsplan ist zu erkennen wie die Komponenten miteinander verbunden wurden.



Ergebnis:

Am Ende des Projektes hat der Roboter alle Musskriterien und auch ein Wunschkriterium erfüllt. Er fährt einen begrenzten Bereich (z.B. mit schwarzem Klebeband) ab. Durch den UItraschallsensor bleibt er ca. 10 cm vor einem Hindernis stehen.

Ausblick:

Wenn man einen Schrittmotor verenden würde, könnte man durch die Schritte, die Entfernung die der Roboter zurückgelegt, messen.

Außerdem wäre es möglich die Spannungsversorgung des Roboters zu messen. Sobald die Spannung einen gewissen Wert unterschreitet, soll der Roboter zu einem gewissen Punkt fahren.

Für mehr Stabilität könnte man hinten (statt dem Nippel) am Roboter ein oder zwei zusätzliche Räder montieren.