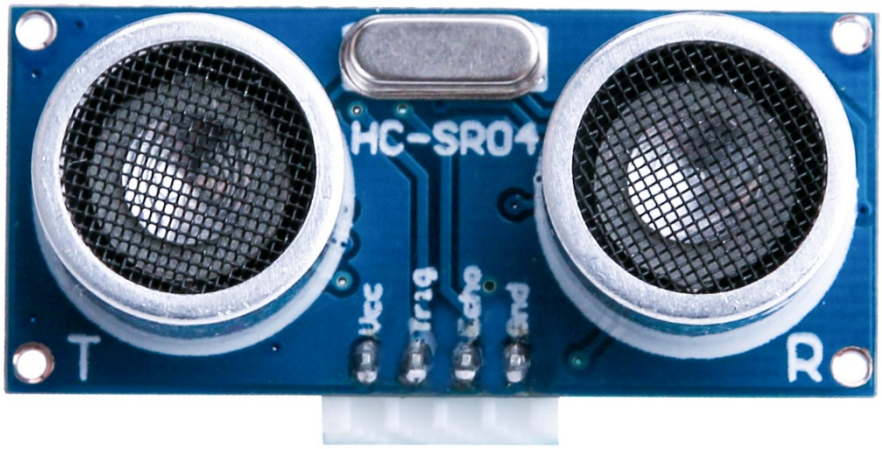
Lektion 4 Hindernis- Vermeidung



Punkte dieses Abschnitts

Die Freude am Lernen, ist nicht nur zu wissen, wie Sie Ihr Auto kontrollieren können, sondern auch wissen, wie Sie Ihr Auto zu schützen können. Also stellen Sie das Auto weit weg von Hindernissen.

Lernteile:

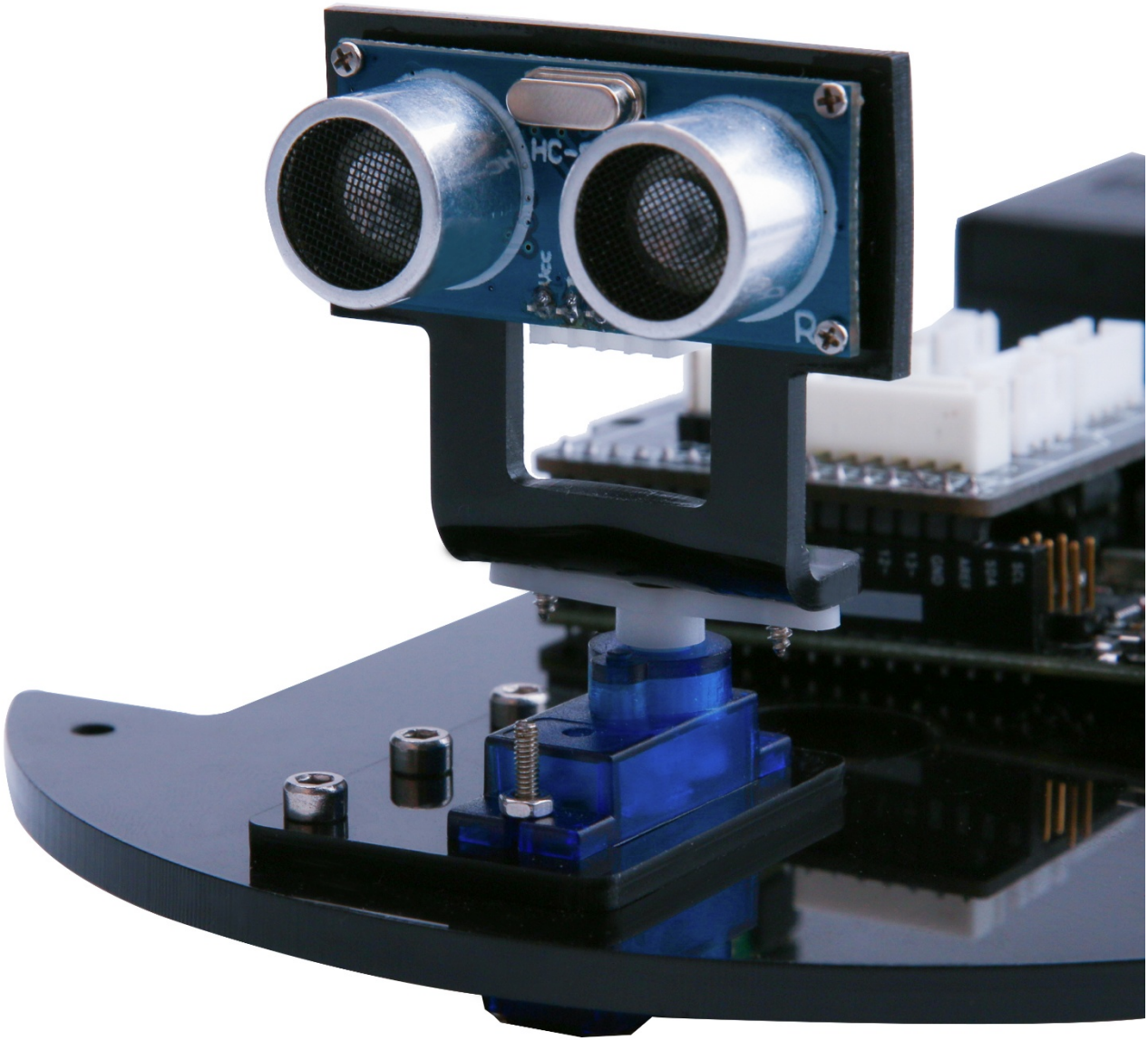
- Erfahren Sie, wie Sie das Ultraschallmodul zusammenbauen können
- Sei vertraut mit Lenkung
- Erfahren Sie mehr über das Prinzip der Hindernisvermeidung
- Verwenden Sie das Programm, um Hindernis Vermeidung Auto wahr werden zu lassen

Vorbereitungen:

- Ein Auto (mit Batterie)
- Ein USB cable
- Ein beweglicher Ultraschall-Sensor

Ⅰ. Verbindung

servo



Ⅱ. Programm hochladen

```
#include <Servo.h> //servo library
Servo myservo; // create servo object to control servo
int Echo = A4;
int Trig = A5;
int in1 = 6;
```

```
int in2 = 7;
int in3 = 8;
int in4 = 9;
int ENA = 5;
int ENB = 11;
int ABS = 150;
int rightDistance = 0,leftDistance = 0,middleDistance = 0 ;
void _mForward()
{
  analogWrite(ENA,ABS);
  analogWrite(ENB,ABS);
    digitalWrite(in1,HIGH);//digital output
    digitalWrite(in2,LOW);
    digitalWrite(in3,LOW);
    digitalWrite(in4,HIGH);
  Serial.println("go forward!");
}

void _mBack()
{
  analogWrite(ENA,ABS);
  analogWrite(ENB,ABS);
    digitalWrite(in1,LOW);
    digitalWrite(in2,HIGH);
    digitalWrite(in3,HIGH);
    digitalWrite(in4,LOW);
  Serial.println("go back!");
}

void _mleft()
{
  analogWrite(ENA,ABS);
  analogWrite(ENB,ABS);
```

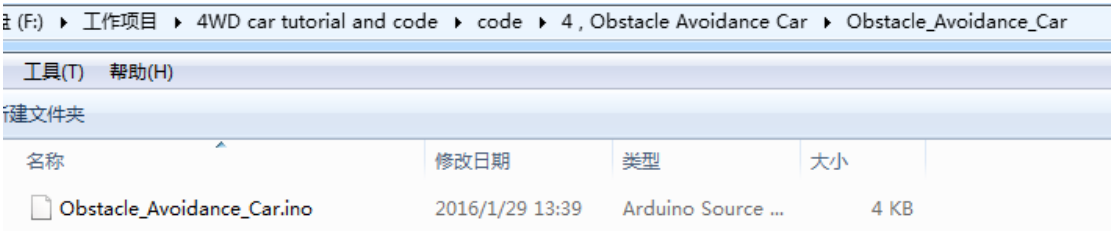
```
  digitalWrite(in1,HIGH);
  digitalWrite(in2,LOW);
  digitalWrite(in3,HIGH);
  digitalWrite(in4,LOW);
 Serial.println("go left!");
}

void _mright()
{
 analogWrite(ENA,ABS);
 analogWrite(ENB,ABS);
  digitalWrite(in1,LOW);
  digitalWrite(in2,HIGH);
  digitalWrite(in3,LOW);
  digitalWrite(in4,HIGH);
 Serial.println("go right!");
}
void _mStop()
{
  digitalWrite(ENA,LOW);
  digitalWrite(ENB,LOW);
  Serial.println("Stop!");
}
 /*Ultrasonic distance measurement Sub function*/
int Distance_test()
{
  digitalWrite(Trig, LOW);
  delayMicroseconds(2);
  digitalWrite(Trig, HIGH);
  delayMicroseconds(20);
  digitalWrite(Trig, LOW);
  float Fdistance = pulseIn(Echo, HIGH);
  Fdistance= Fdistance/58;
```

```
  return (int)Fdistance;
}

void setup()
{
  myservo.attach(3);// attach servo on pin 3 to servo object
  Serial.begin(9600);
  pinMode(Echo, INPUT);
  pinMode(Trig, OUTPUT);
  pinMode(in1,OUTPUT);
  pinMode(in2,OUTPUT);
  pinMode(in3,OUTPUT);
  pinMode(in4,OUTPUT);
  pinMode(ENA,OUTPUT);
  pinMode(ENB,OUTPUT);
  _mStop();
}

void loop()
{
    myservo.write(90);//setservo position according to scaled value
    delay(500);
    middleDistance = Distance_test();
    #ifdef send
    Serial.print("middleDistance=");
    Serial.println(middleDistance);
    #endif

    if(middleDistance<=20)
    {
      _mStop();
      delay(500);
      myservo.write(5);
```

```
        delay(1000);
        rightDistance = Distance_test();

        #ifdef send
        Serial.print("rightDistance=");
        Serial.println(rightDistance);
        #endif

        delay(500);
          myservo.write(90);
        delay(1000);
        myservo.write(180);
        delay(1000);
        leftDistance = Distance_test();

        #ifdef send
        Serial.print("leftDistance=");
        Serial.println(leftDistance);
        #endif

        delay(500);
        myservo.write(90);
        delay(1000);
        if(rightDistance>leftDistance)
        {
          _mright();
           delay(360);
         }
          else if(rightDistance<leftDistance)
         {
          _mleft();
           delay(360);
         }
```

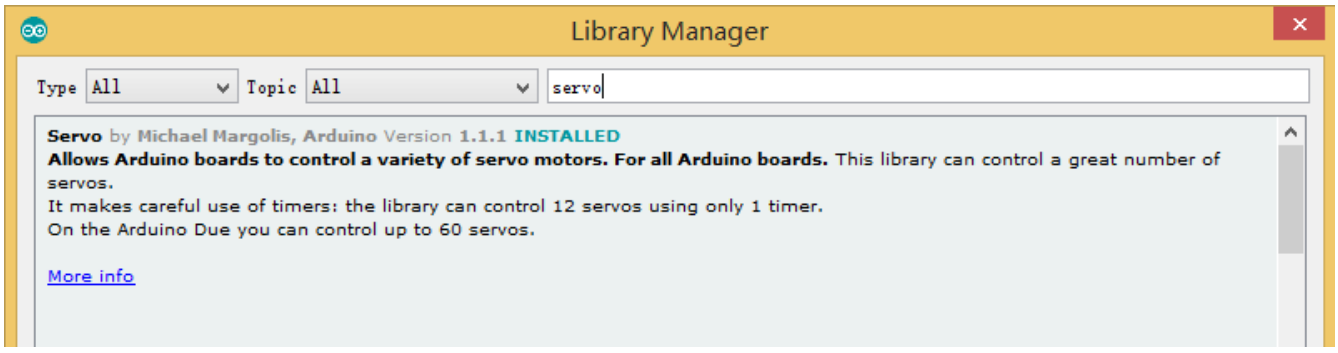
```
        else if((rightDistance<=20) | | (leftDistance<=20))
        {
          _mBack();
          delay(180);
        }
        else
        {
          _mForward();
        }
    }
    else
        _mForward();
}
```

Öffnen Sie die Datei Obstacle_Avoidance_Car \ Obstacle_Avoidance_Car.ino



Da das Programm die Bibliothek <servo.h> verwendet, müssen wir die Bibliothek zunächst installieren.

Öffnen Sie die Sketch --- Include Library --- Verwalten von Bibliotheken



Servo suchen und dann die neueste Version installieren.

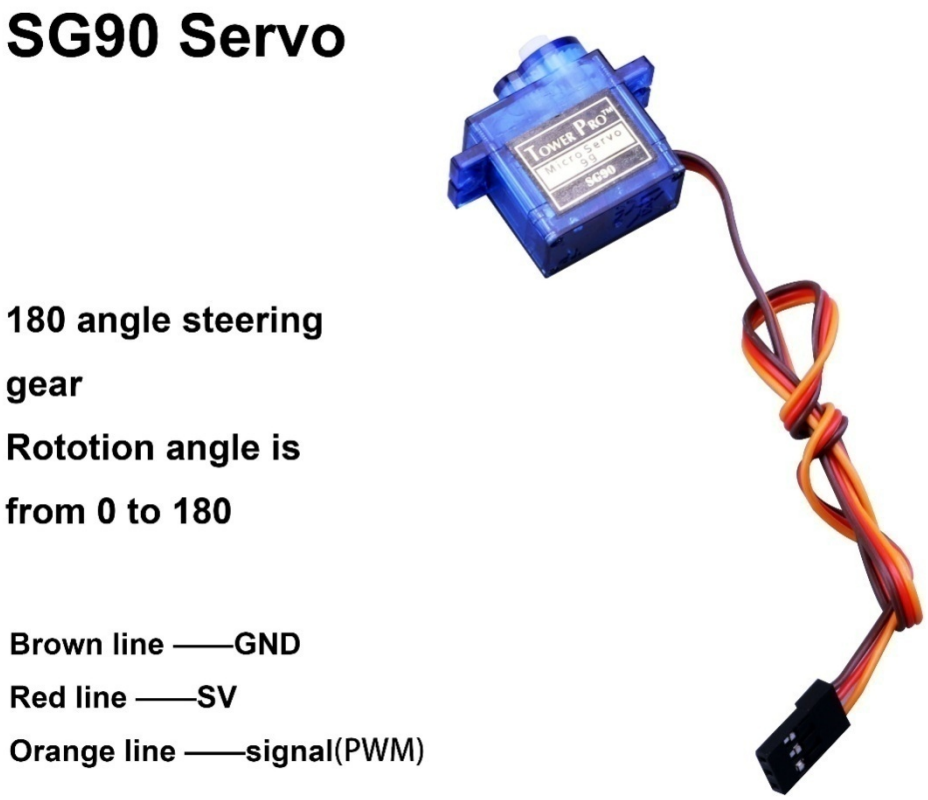
Nach dem Hochladen des Programms auf die UNO-Steuerplatine, trennen Sie das Kabel, stellen Sie das Fahrzeug auf den Boden und schalten Sie die Stromversorgung ein.

Sie werden sehen, dass das Fahrzeug vorwärts fährt und die “Wolken”platform mit

dem Ultraschallsensor dreht sich, um die Distanzmesssensoren kontinuierlich zu betreiben. Wenn es Hindernisse gibt, wird die Cloud-Plattform stoppen und das Fahrzeug wird seine Richtung ändern, um das Hindernis zu umgehen. Nach der Umgehung des Hindernisses wird sich die Cloud-plattform wieder drehen und das Fahrzeug wird auch weiterfahren.

Ⅲ. Einführung des Grundprinzips

Zuerst lasst uns etwas über das SG90 Servo lernen:



Klassifizierung: 180 Lenkgetriebe

Normalerweise hat das Servo 3 Steuerungskabel: Stromversorgung, Masse und Signal.

Definition der Servo Pins: braunes Kabel - GND, rotes Kabel - 5V, orange - Signal.

So funktioniert das ganze:

Der Signalmodulationschip im Servo empfängt Signale von der Anschaltbaugruppe, dann erhält der Servo die Grundgleichspannung. Es gibt auch eine Referenzschaltung innerhalb des Servos, die eine Standardspannung erzeugt. Diese beiden Spannungen werden miteinander verglichen und die Differenz wird ausgegeben. Dann erhält der Motorchip den Unterschied und entscheidet über die Drehzahl, die Richtung und die Achse. Wenn es keinen Unterschied zwischen den beiden Spannungen gibt, hört das Servo auf.

Wie man den Servo steuert:

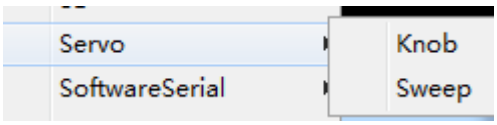
Um die Servoumdrehung zu steuern, musst du den Zeitimpuls um etwa 20ms und die Hochpegelimpulsbreite auf etwa 0,5 ms ~ 2,5 ms einstellen, was mit dem vom Servo begrenzten Winkel übereinstimmt.

Bei der Verwendung von 180-Winkelservo ist die entsprechende Steuerbeziehung wie folgt:

0.5ms	0 Grad
1.0ms	45 Grad
1.5ms	90 Grad
2.0ms	135 Grad
2.5ms	180 Grad

Das Programm:

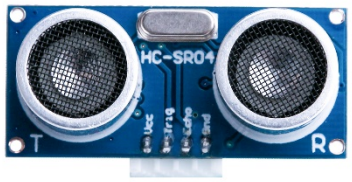
Arduino hat eine Bibliotheksdatei. <Servo.h>




```
Servo myservo;  // create servo object to control servo
 myservo.attach(3);  // attach servo on pin 3 to servo object
 myservo.write(90); //set servo position according to scaled value
```

Sie können Lenkgetriebe in 9 Worten fahren.

Als nächstes wollen wir uns das Ultraschallsensormodul ansehen.

Funktion des Moduls: Testabstand, Hochpräzisionsmodul.

Anwendung der Produkte: Roboter-Hindernisvermeidung, Objekttestabstand, Flüssigkeitsprüfung, öffentliche Sicherheit, Parkplatzprüfung.

Haupttechnische Parameter

(1): verwendete Spannung: DC --- 5V

(2): statischer Strom: weniger als 2mA

(3): Pegelausgang: höher als 5V

(4): Pegelausgang: kleiner als 0

(5): Erfassungswinkel: nicht größer als 15 Grad

(6): Erkennungsabstand: 2cm-450cm

(7): hohe Präzision: bis zu 0,2 cm

Methode der Verbindung von Kabel: VCC, Trig (das Ende der Kontrolle), Echo (das Ende des Empfangs), GND

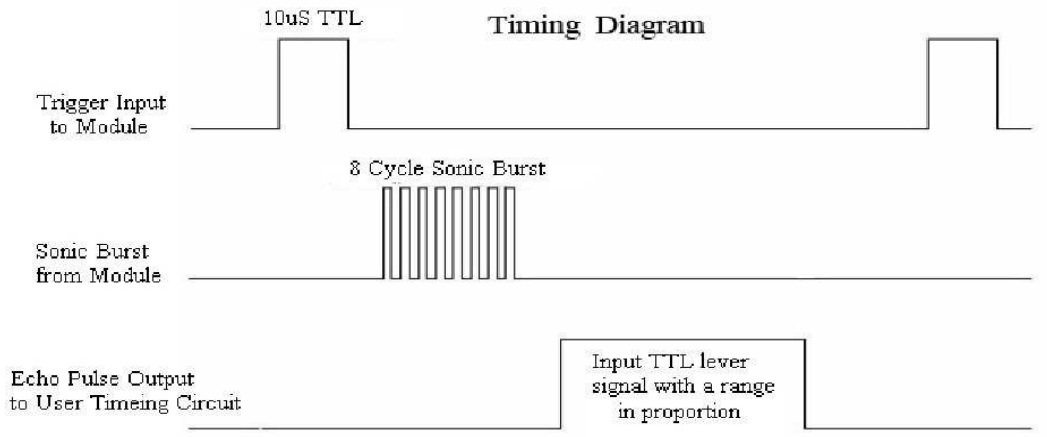
Wie funktioniert das Modul:

(1) IO-Port von TRIG verwenden, um ein High-Level-Signal auszugeben für mindestens 10us einmal;
(2) Das Modul sendet 8 quadratische Wellen von 40kz automatisch, prüft, ob das Signal automatisch zurückgegeben wird;
(3) Wenn Signale empfangen werden, gibt das Modul einen Hochpegelimpuls über den IO-Port von ECHO aus, die Zeitdauer des Hochpegelimpulses ist die Zeit zwischen dem Wellensenden und dem Empfang. So kann das Modul den Abstand nach der Zeit kennen.

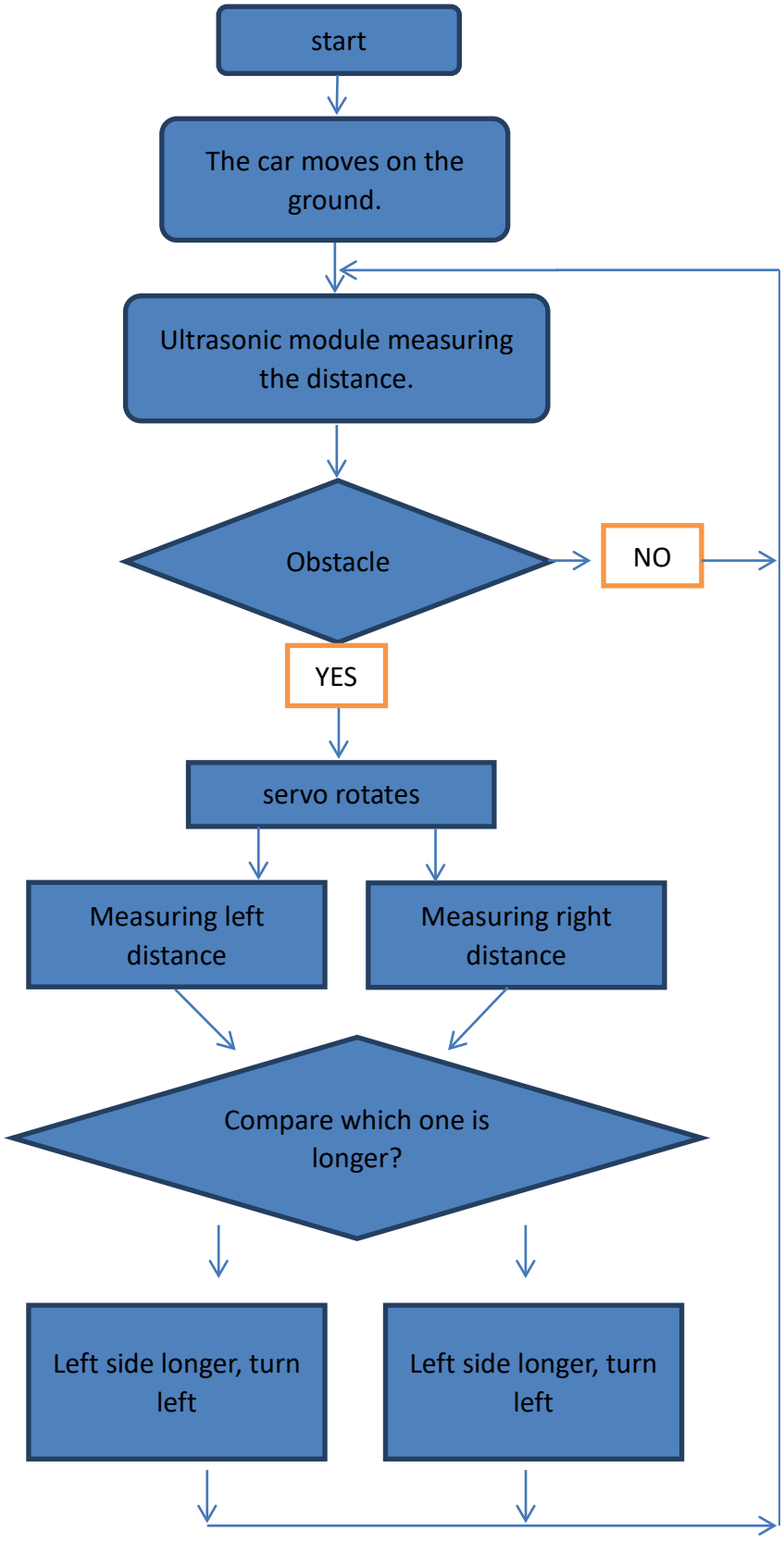
Testing distance= (high level time* velocity of sound (340M/S))/2);

Tatsächlicher Betrieb:

Das Zeitdiagramm ist unten dargestellt. Sie müssen nur einen Short10uS-Puls an den Trigger-Eingang liefern, um die Messung zu starten, und dann sendet das Modul einen 8-Zyklus-Ultraschall-Ultraschall mit 40 kHz aus und hebt dessen Echo an. Das Echo ist ein Distanzobjekt, das die Pulsbreite und der Bereich im Verhältnis ist. Sie können den Bereich über das Zeitintervall zwischen dem Senden des Triggersignals und dem Empfang des Echosignals berechnen. Formel: µs / 58 = Zentimeter oder µs / 148 = Zoll; Oder: der Bereich = hohe Pegelzeit * Geschwindigkeit (340M / S) / 2; Wir empfehlen, über 60ms Messzyklus zu verwenden, um ein Triggersignal zum Echosignal zu verhindern.




```
/*Ultrasonic distance measurement Sub function*/
int Distance_test()
{
  digitalWrite(Trig, LOW);
  delayMicroseconds(2);
  digitalWrite(Trig, HIGH);
  delayMicroseconds(20);
  digitalWrite(Trig, LOW);
  float Fdistance = pulseIn(Echo, HIGH);
  Fdistance= Fdistance/58;
  return (int)Fdistance;
}
```


start
The car moves on the ground.
Ultrasonic module measuring the distance.
Obstacle
NO
YES
servo rotates
Measuring left distance
Measuring right distance
Compare which one is longer?
Left side longer, turn left
Left side longer, turn left

Im Bild oben, können wir sehen, dass das Prinzip des Hindernis Vermeidung Auto sehr einfach ist. Das Ultraschallsensormodul erkennt den Abstand zwischen dem Wagen und den Hindernissen immer wieder und sendet die Daten an die Anschaltbaugruppe, dann stoppt das Auto und dreht das Servo, um die linke und rechte Seite um die Umgebung zu scannen. Nach dem Vergleich der Distanz zu der anderen Seite, dreht sich das Auto zu der Seite, die die längere Distanz hat und dann gehts wieder vorwärts . Dann erkennt das Ultraschallsensormodul den Abstand wieder.

```
if(rightDistance>leftDistance)
        {
          _mright();
          delay(360);
         }
         else if(rightDistance<leftDistance)
         {
          _mleft();
          delay(360);
         }
         else if((rightDistance<=20) | | (leftDistance<=20))
         {
          _mBack();
          delay(180);
         }
         else
         {
          _mForward();
         }
     }
     else
          _mForward();
}
```