

Lesson 4 indernis- 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 e
- Sei vertraut mit Lenkungen
- Erfahren Sie mehr über das Prinzip der Hindernisvermeidung
- Verwenden Sie das Programm, um Hindernis Vermeidung Auto wahr werden zu lassen

Vorbereitungen:

е

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



I. Verbindung

Beim Zusammenbau des Ultraschallsensor-Modul-Halters sollte das Servo auch austariert werden, um sicherzustellen, dass der Server um 180 Grad drehen kann.



STEP1: Verbinde den UNO mit dem Computer und öffne die Servo_debug-Code-Datei im Pfad "\ Lektion 4 Hindernisvermeidungs-Auto \ Servo_debug \ Servo_debug.ino".





Servo_debug code preview:

```
//www.elegoo.com
#include <Servo.h>
Servo myservo;
void setup(){
  myservo.attach(3);
  myservo.write(90);// move servos to center position -> 90°
}
void loop(){
```

STEP2: Wähle "Tool" --> "Port" und"Board" in der Arduino IDE.

∞ Servo_debug Arduino 1.8.2 - □ >								
File E	dit Sketch	Tools Help						
	• 🖬 🖸	Auto Format	Ctrl+T			Q		
		Archive Sketch						
Ser	vo_debug	Fix Encoding & Reload						
1	// <u>www. e</u>	Serial Monitor	Ctrl+Shift+M					
2	#includ	Serial Plotter	Ctrl+Shift+L	_				
3	Servo r	WiFi101 Firmware Updater						
4		Board: "Arduino/Genuino Uno"]	>				
5⊡	void se	Port: "COM6"		2	Serial ports			
6	myser	Get Board Info			COM1			
7	myser	Programmer: "USBasp"		. <u></u>	COM6	-> 90°		
8	}	Burn Bootloader						
9⊡	void lo	op () {						

STEP3: Klicken Sie auf die Pfeiltaste, um den Code auf die UNO-Controller-Karte

hochzuladen.

}





Nach dem Hochladen wird das Servo auf 90 Grad gedreht und bleibt dort.

SCHRITT 4: Das Ultraschallsensormodul bei 90 Grad zusammenbauen.

Der Winkel jeder Zähne auf Mikro-Servo ist 15 Grad und wenn Sie es auf der Mitte der Richtung von 90 Grad installieren, wird es sich jeweils nach links oder rechts um 15 Grad drehen, was bedeutet, dass der tatsächliche Grad der Installation der Mikro-Servo 15 Grad Oder 105 Grad ist.



FAQ zum Servomotor.

- 1 Warum dreht sich der Mikro-Servo gegen den Uhrzeigersinn um jeweils 15 Grad, wenn ich die Stromversorgung einschalte?
 - Dies ist normal beim SG90 Micro Servo und es wird keinen Einfluss auf die normale Verwendung mit Programmen geben.
 - Wenn du es nicht mit Programmen kontrollierst, kannst du es mit deiner Hand wieder normal drehen oder die mit dem Mikro-Servo verbundenen Drähte abziehen, bevor du das Gerät



einschaltest.

2 Das Mikro-Servo ist außer Kontrolle und dreht sich weiter.

Verwenden Sie "myservo.write (angle)", um den Mikron-Servo auf den Winkelgrad zu befehlen, der einen Bereich von 0 bis 180 hat. Wenn er den Bereich überschreitet, erkennt der Micro-Servo diesen Winkel nicht und wird sich drehen.

II. Programm hochladen

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

Öffnen Sie die Arduino Software

∞ sketch_may18a Arduino 1.8.2 - □	×
File Edit Sketch Tools Help	
	P
sketch_may18a	
1 void setup() {	^
2 // put your setup code here, to run once:	
3	
4 }	
5	
6 void loop() {	
7 // put your main code here, to run repeatedly:	
8	
9 }	
	~
Arduino/Genuino Uno on	COM12



∞ sketch_r	nay18a Arduino 1.8.2		—		×
File Edit Sk	etch Tools Help				
	Verify/Compile	Ctrl+R			Ø
	Upload	Ctrl+U			_
sketch_	Upload Using Programmer	Ctrl+Shift+U	Manage Libra	ries	
1⊟ vc	Export compiled Binary	Ctrl+Alt+S	Add .ZIP Libra	arv	
2	Show Sketch Folder	Ctrl+K	And the library		
3	Include Library)	Arduino librar	les	
4 }	Add File		Bridge		



Warten auf "Laden von Bibliotheksindex" zum fertigstellen.

∞ Library Manager	×
Type All V Topic All V Filter your search	
Arduino Low Power by Arduino Power save primitives features for SAMD and nRF52 32bit boards With this library you can manage the low power states of newer Arduino boards More info	^
Arduino SigFox for MKRFox1200 by Arduino Helper library for MKRFox1200 board and ATAB8520E Sigfox module This library allows some high level operations on Sigfox module, to ease integration with existing projects More info	
Arduino Uno WiFi Dev Ed Library by Arduino This library allows users to use network features like rest and mgtt. Includes some tools for the ESP8266. Use this library only with Arduino Uno WiFi Developer Edition. <u>More info</u>	
ArduinoCloud by Arduino Easly connect your Arduino/Genuino board to the Arduino Cloud Easly connect your Arduino/Genuino board to the Arduino Cloud More info	•
Downloading libraries index Downloaded 12kb of 332kb. Cancel	L

Servo suchen und dann die neueste Version installieren. Das folgende Bild zeigt, dass die

Servobibliothek bereits installiert ist.

∞ Library Manager	×	
Type All V Topic All V servo		
Servo Built-In by Michael Margolis, Arduino Version 1.1.2 INSTALLED Allows Arduino boards to control a variety of servo motors. For all Arduino boards. This library can control a great number of servos. It makes careful use of timers: the library can control 12 servos using only 1 timer. On the Arduino Due you can control up to 60 servos. More info	~	

Verbinde die UNO-Controller-Karte mit dem Computer, öffne die Code-Datei im Pfad "\ Lektion 4 Hindernis-Vermeidungs-Auto \ Obstacle_Avoidance_Car \ Obstacle_Avoidance_Car.ino". Laden Sie das Programm auf die UNO-Karte.

	Lesson 4 Obstacle Avoidance Car > Obstacle_Avoidance_Car				
	* ^	名称	^		修改日期
	The	💿 Obstacle	e_Avoidance_Ca	r.ino	2017/5/18 15:5
Code preview:	de l				_
//www.elegoo.	com				
#include <ser Servo myservo</ser 	vo.h> / ; //	/servo lib create se	rary rvo object	to control	servo
int Echo = A4 int Trig = A5	; ;				
-					



```
#define ENA 5
#define ENB 6
#define IN1 7
#define IN2 8
#define IN3 9
#define IN4 11
#define carSpeed 150
int rightDistance = 0, leftDistance = 0, middleDistance = 0;
void forward(){
 analogWrite(ENA, carSpeed);
 analogWrite(ENB, carSpeed);
 digitalWrite(IN1, HIGH);
 digitalWrite(IN2, LOW);
 digitalWrite(IN3, LOW);
 digitalWrite(IN4, HIGH);
 Serial.println("Forward");
}
void back() {
 analogWrite(ENA, carSpeed);
 analogWrite(ENB, carSpeed);
 digitalWrite(IN1, LOW);
 digitalWrite(IN2, HIGH);
 digitalWrite(IN3, HIGH);
 digitalWrite(IN4, LOW);
 Serial.println("Back");
}
void left() {
 analogWrite(ENA, carSpeed);
 analogWrite(ENB, carSpeed);
 digitalWrite(IN1, LOW);
 digitalWrite(IN2, HIGH);
 digitalWrite(IN3, LOW);
 digitalWrite(IN4, HIGH);
 Serial.println("Left");
}
void right() {
 analogWrite(ENA, carSpeed);
 analogWrite(ENB, carSpeed);
 digitalWrite(IN1, HIGH);
```



```
digitalWrite(IN2, LOW);
 digitalWrite(IN3, HIGH);
 digitalWrite(IN4, LOW);
 Serial.println("Right");
}
void stop() {
 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);
 stop();
}
void loop() {
   myservo.write(90); //setservo position according to scaled value
   delay(500);
   middleDistance = Distance_test();
   if(middleDistance <= 20) {</pre>
```



```
stop();
 delay(500);
 myservo.write(10);
 delay(1000);
 rightDistance = Distance_test();
 delay(500);
 myservo.write(90);
 delay(1000);
 myservo.write(180);
 delay(1000);
 leftDistance = Distance_test();
 delay(500);
 myservo.write(90);
 delay(1000);
 if(rightDistance > leftDistance) {
   right();
   delay(360);
 }
 else if(rightDistance < leftDistance) {</pre>
   left();
   delay(360);
 }
 else if((rightDistance <= 20) || (leftDistance <= 20)) {</pre>
   back();
   delay(180);
 }
 else {
   forward();
 }
}
else {
   forward();
}
```

}

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

Sie werden sehen, dass das Fahrzeug vorwärts geht und die Ultraschall-Sensor-Plattform sich dreht, um die Distanzmesssensoren kontinuierlich zu betreiben. Wenn es Hindernisse gibt, wird die Cloud-Plattform aufhören und das Fahrzeug wird seine Richtung ändern, um das Hindernis zu



umgehen. Nach der Umfahrung des Hindernisses wird sich die Cloud-plattform wieder drehen und das Fahrzeug wird auch weiterfahren.

III. Einführung des Prinzips

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 Beispielprogramm:

Öffne die Arduino IDE und wähle "File->Examples->Servo->Sweep"

∞ sketch_may19	b Arduino 1.8.2				×
File Edit Sketch	Tools Help	-			
New	Ctrl+N				Ø
Open	Ctrl+O				
Open Recent	t 🍾				
Sketchbook	;	∆ Built-in Examples			^
Examples	3	01 Basics	>		
Close	Ctrl+W	02 Digital	>		
Save	Ctrl+S	03 Analog	>		
Save As	Ctrl+Shift+S	04.Communication	>		
Page Setup	Ctrl+Shift+P	05.Control	>		
Print	Ctrl+P	06.Sensors	>		
		07.Display	>7		
Preferences	Ctrl+Comma	08.Strings	>		
Quit	Ctrl+Q	09.USB	>		
5 1		10.StarterKit_BasicKit	>		
		11.ArduinoISP	>		
		Examples for any board			
		Adafruit Circuit Playground	>		
		Bridge	>		
		Esplora	>		
		Ethernet	>		
		Firmata	>		
		GSM	>		~
		LiquidCrystal	>		
		Robot Control	>		
		Robot Motor	>		
		SD	>		
		Servo	3	Knob	
		SpacebrewYun	>	Sweep	
		Changes			



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: μ s / 58 = Zentimeter oder μ s / 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 = pulseln(Echo, HIGH);



Fdistance= Fdistance/58; return (int)Fdistance;

}







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.

Code preview:

```
if(rightDistance > leftDistance) {
  right();
  delay(360);
}
else if(rightDistance < leftDistance) {
   left();
   delay(360);
}
else if((rightDistance <= 20) || (leftDistance <= 20)) {
   back();
   delay(180);
}
else {
   forward();
}</pre>
```