Zusammenfassung

- **Einleitung in die Ultraschallnavigation**
  - Ein neues »Seh«-System für den Mars-Rover wird eingeführt, um dessen Fähigkeit zur Hinderniserkennung zu verbessern, indem ein Ultraschallsensormodul hinzugefügt wird.
  - Das Ziel ist, die Wahrnehmung des Rovers für Hindernisse direkt vor ihm zu schärfen, um die Navigation in komplexem Gelände zu optimieren.
- **Ultraschallsensor**
  - Der HC-SR04 Ultraschallsensor kann Entfernungen von 2 cm bis 400 cm messen und ermöglicht dem Rover, seine Umgebung berührungslos zu »sehen«.
  - Die Funktionsweise des Sensors ähnelt der Echolokation einer Fledermaus, nutzt Schallwellen zur Entfernungsmessung und verbessert so die Navigation des Rovers.
- **Funktionsprinzip des Ultraschallsensors**
  - Der Sensor verwendet vier Pins (TRIG, ECHO, VCC, GND) für seine Funktion: TRIG sendet Schallwellen aus, ECHO empfängt das Echo der Wellen, VCC versorgt den Sensor mit Strom, und GND dient als Erdung.
  - Die Entfernungsmessung erfolgt durch Aussenden und Empfangen von Ultraschallwellen, wobei die Zeit zwischen Aussendung und Echoempfang zur Berechnung der Entfernung zu Objekten genutzt wird.

Berechnung

Um die Dauer des Echos zu berechnen, das der HC-SR04 Ultraschall-Entfernungssensor empfängt, wenn ein Hindernis 2 cm, 20 cm und 400 cm entfernt ist, verwenden wir die Formel:

$$\text{Dauer} = \frac{\text{Entfernung} \times 2}{\text{Schallgeschwindigkeit}}$$

Die Schallgeschwindigkeit in Luft beträgt etwa 340 m/s (oder 0,034 cm/us). Da die Schallwelle die Entfernung zum Hindernis und zurück zurücklegen muss, multiplizieren wir die Entfernung mit 2.

Beim Arbeiten mit dem HC-SR04 Ultraschallsensor und ähnlichen Sensoren ist es üblich, die Zeit in Mikrosekunden (us) zu messen, da dies die Genauigkeit der Entfernungsmessung erhöht.

Berechnung für verschiedene Entfernungen

1. **Für ein Hindernis in 2 cm Entfernung:**
   $\text{Dauer} = \frac{2\,\text{cm} \times 2}{0{,}034\,\text{cm/us}} = \frac{4}{0{,}034}\,\text{us}$
2. **Für ein Hindernis in 20 cm Entfernung:**
   $\text{Dauer} = \frac{20\,\text{cm} \times 2}{0{,}034\,\text{cm/us}} = \frac{40}{0{,}034}\,\text{us}$
3. **Für ein Hindernis in 400 cm Entfernung:**
   $\text{Dauer} = \frac{400\,\text{cm} \times 2}{0{,}034\,\text{cm/us}} = \frac{800}{0{,}034}\,\text{us}$

```
# Berechnung der Dauer des Echos für verschiedene Entfernungen in Mikrosekunden
schallgeschwindigkeit_cm_us = 0.034  # Schallgeschwindigkeit in cm/us

# Entfernungen in cm
entfernungen_cm = [2, 20, 400]
```

```
# Berechnung der Dauer für jede Entfernung in Mikrosekunden (us)
dauer_us = [(entfernung * 2 / schallgeschwindigkeit_cm_us) for entfernung in
    entfernungen_cm]

# Umrechnung der Dauer von Mikrosekunden in Millisekunden (ms)
dauer_ms = [dauer / 1000 for dauer in dauer_us]

# Umrechnung der Dauer von Millisekunden in Sekunden (s)
dauer_s = [dauer / 1000 for dauer in dauer_ms]

# Um ein besseres Verständnis für die Dauer zu bekommen (Berechnung des reziproken
    Werts der Dauer in Sekunden)
dauer_v = [1 / dauer if dauer != 0 else 0 for dauer in dauer_s]

# Ausgabe der Ergebnisse
dauer_us, dauer_ms, dauer_s, dauer_v
```

Die berechneten Ergebnisse für die Dauer des Echos bei verschiedenen Entfernungen und deren Umrechnungen sind:

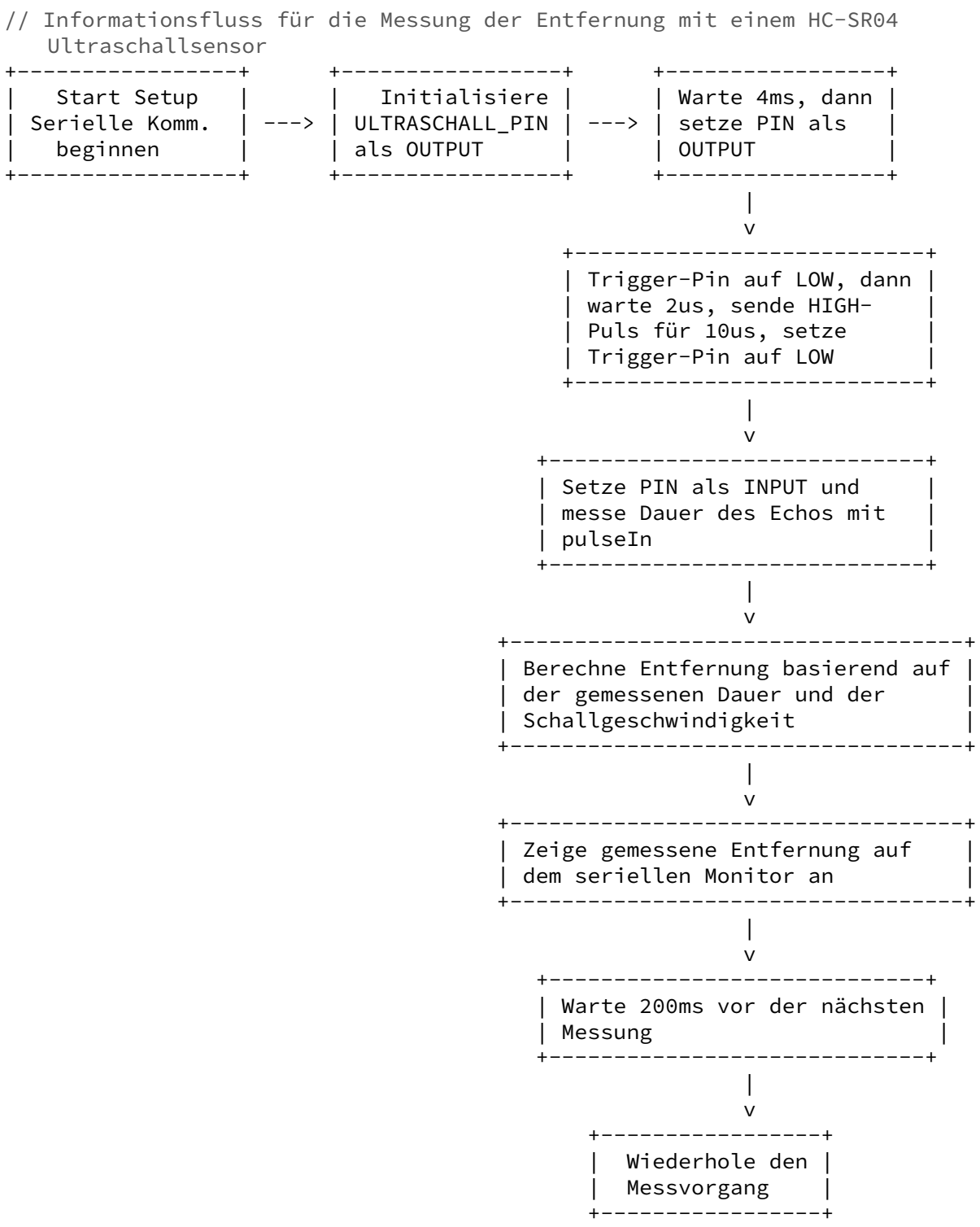
- **Dauer in Mikrosekunden (us):**
  - 2 cm Entfernung: 117.65 us
  - 20 cm Entfernung: 1176.47 us
  - 400 cm Entfernung: 23529.41 us
- **Dauer in Millisekunden (ms):**
  - 2 cm Entfernung: 0.118 ms
  - 20 cm Entfernung: 1.176 ms
  - 400 cm Entfernung: 23.529 ms
- **Dauer in Sekunden (s):**
  - 2 cm Entfernung: 0.000118 s
  - 20 cm Entfernung: 0.001176 s
  - 400 cm Entfernung: 0.023529 s
- **Reziproke Werte der Dauer in Sekunden (1/s, für ein besseres Verständnis der Dauer):**
  - Bei einer **Entfernung von 2 cm** könnte der Schall das Hindernis und zurück zum Sensor **8500 Mal pro Sekunde** reisen.
  - Bei einer **Entfernung von 20 cm** könnte dies **850 Mal pro Sekunde** geschehen.
  - Und bei einer **Entfernung von 400 cm (4 Meter)** könnte der Schall diese Strecke **42,5 Mal pro Sekunde** zurücklegen.

Diese Zahlen veranschaulichen die Häufigkeit, mit der der Schall die angegebene Distanz innerhalb einer Sekunde hin und zurück reisen kann, und bieten eine anschauliche Perspektive auf die Geschwindigkeit der Schallausbreitung in Bezug auf die Entfernungsmessung mit dem HC-SR04 Ultraschallsensor.

Um ein besseres Verständnis zu bekommen eine Millisekunde ist ein Tausendstel einer Sekunde, daher gilt: $1\text{ ms} = 0,001\text{ s} = \frac{1}{1000}$

- **0,1 ms** entspricht $0,1 \times 0,001 = 0,0001$ etwa Sekunden **1/1000 einer Sekunde**
- **1,1 ms** entspricht $1,1 \times 0,001 = 0,0011$ etwa Sekunden **1/10000 einer Sekunde**
- **23,5 ms** entspricht $23,5 \times 0,001 = 0,0235$ etwa Sekunden **1/42 einer Sekunde**

Programmierung des Sensors



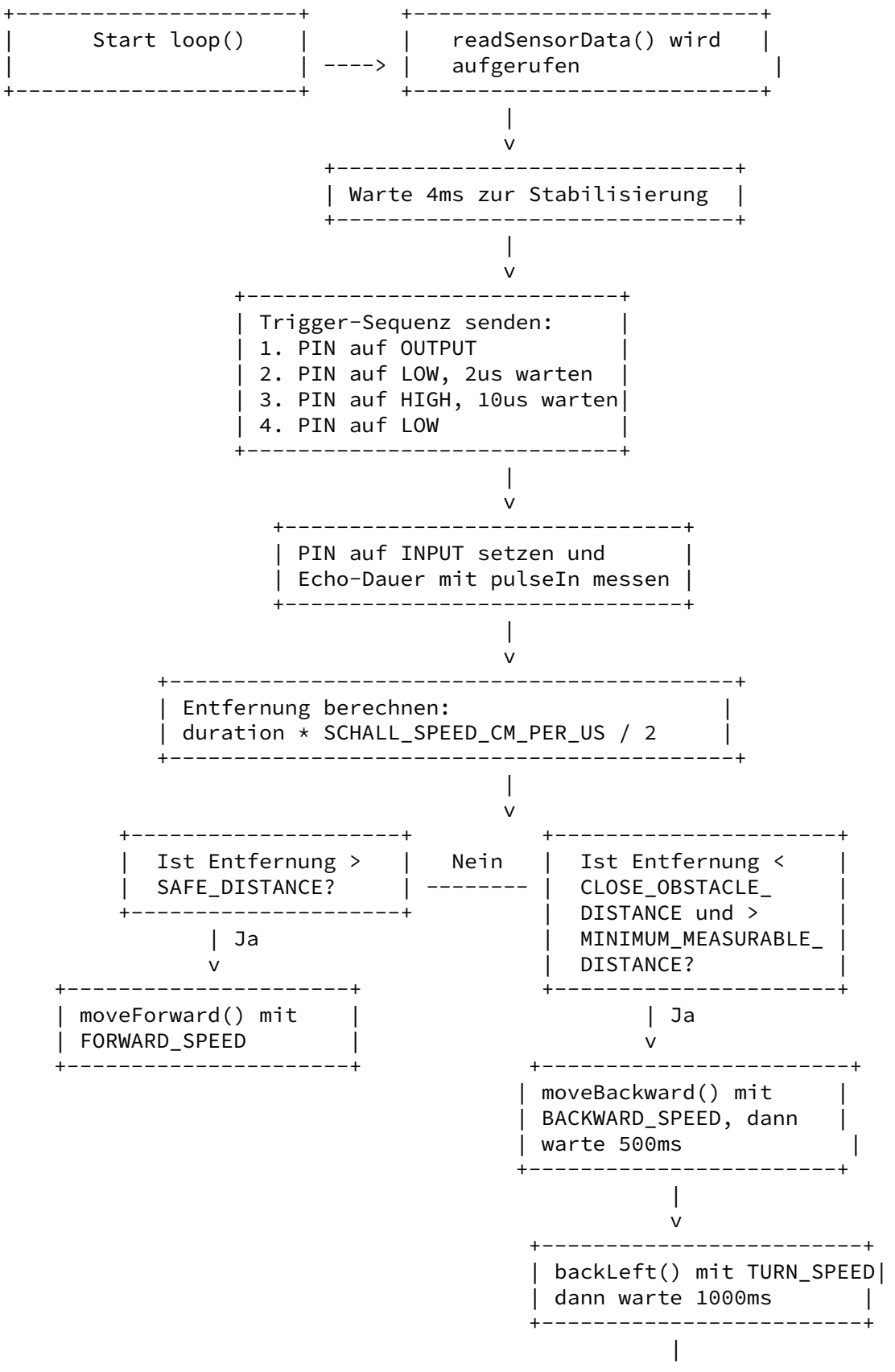
Ausgabe: Die Entfernung beträgt: 10.13cm

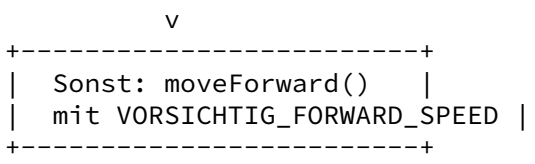
```
/**
 * @file main.cpp
 * @brief Programmierung des Ultraschallsensors
 *
 * Dieser Code ermöglicht es, die Entfernung zu einem Objekt mithilfe des HC-SR04
    Ultraschallsensors zu
 * messen. Die serielle Kommunikation wird genutzt, um die gemessene Entfernung in
    Zentimetern auszugeben.
 * Der Sensor wird durch einen kurzen High-Puls aktiviert, und die Zeit, die das
    Echo benötigt,
 * um zum Sensor zurückzukehren, wird verwendet, um die Entfernung zum Objekt zu
    berechnen.
 */
#include <Arduino.h>

// Definiere den Pin für das Ultraschallmodul
#define ULTRASONIC_PIN 10

void setup() {
  // Starte die serielle Kommunikation
  Serial.begin(115200);
}

void loop() {

  // Eine Verzögerung von 4ms ist erforderlich, sonst könnte das Ergebnis 0 sein
  delay(4);

  // Setze den Pin als Ausgang, um das Signal zu senden
  pinMode(ULTRASONIC_PIN, OUTPUT);

  // Setze den Trigger-Pin zurück auf niedrig
  digitalWrite(ULTRASONIC_PIN, LOW);
  delayMicroseconds(2);

  // Aktiviere den Sensor, indem ein High-Puls für 10us gesendet wird
  digitalWrite(ULTRASONIC_PIN, HIGH);
  delayMicroseconds(10);

  // Setze den Trigger-Pin wieder auf niedrig
  digitalWrite(ULTRASONIC_PIN, LOW);

  // Setze den Pin als Eingang, um zu lesen
  pinMode(ULTRASONIC_PIN, INPUT);

  // pulseIn gibt die Dauer des Pulses am Pin zurück
  float duration = pulseIn(ULTRASONIC_PIN, HIGH);

  // Berechne die Entfernung (in cm) basierend auf der Schallgeschwindigkeit (340 m
     /s oder 0,034 cm/us)
  float distance = duration * 0.034 / 2;

  // Zeige die Entfernung auf dem seriellen Monitor an
  Serial.print("Die Entfernung beträgt: ");
  Serial.print(distance);
  Serial.println(" cm");

  // Verzögere ein wenig, damit der Sensor sich vor der nächsten Messung
     stabilisieren kann
```

```
  delay(200);
}
```

Programmierung des Ultraschallmoduls zur Steuerung des Mars-Rovers

Flussdiagramm zeigt den Ablauf von der Messung der Entfernung mit dem Ultraschallsensor bis zur Entscheidung, wie der Rover basierend auf dieser Entfernung gesteuert werden soll. Es umfasst die Initialisierung des Sensors, das Senden und Empfangen des Ultraschallsignals, die Berechnung der Entfernung und die anschließende Steuerung des Rovers je nach gemessener Entfernung.




v
Sonst: moveForward()
mit VORSICHTIG_FORWARD_SPEED

```
/**
 * @file main.cpp
 * @brief Programmierung des Ultraschallmoduls zur Steuerung des Mars-Rovers
 *
 * Dieser Code nutzt den Ultraschallsensor, um die Entfernung zu Hindernissen zu
    messen
 * und den Mars-Rover entsprechend zu steuern.
 * Es wird entschieden,
 * ob der Rover vorwärts, rückwärts oder in eine bestimmte Richtung bewegt werden
    soll,
 * basierend auf der gemessenen Entfernung.
 */
#include <Arduino.h>
#include <SoftPWM.h>

// Definiere den Pin für das Ultraschallmodul
#define ULTRASCHALL_SENSOR_PIN 10

// Definition der Pins für die Motoren
#define LEFT_MOTOR_FORWARD_PIN 2
#define LEFT_MOTOR_REVERSE_PIN 3
#define RIGHT_MOTOR_FORWARD_PIN 5
#define RIGHT_MOTOR_REVERSE_PIN 4

// Konstanten für die Entfernungsmessung
const float SCHALL_SPEED_CM_PER_US = 0.034; // Schallgeschwindigkeit in cm/µs
const int SAFE_DISTANCE = 50;               // Sicherheitsabstand in cm
const int CLOSE_OBSTACLE_DISTANCE = 15;    // Abstand eines nahen Hindernisses in
   cm
const int MINIMUM_MEASURABLE_DISTANCE = 2; // Mindestmessbare Entfernung in cm
const int FORWARD_SPEED = 200;              // Geschwindigkeit für Vorwärtsbewegung
const int VORSICHTIG_FORWARD_SPEED = 70;    // Vorsichtige Vorwärtsbewegung
const int BACKWARD_SPEED = 70;             // Geschwindigkeit für Rückwärtsbewegung
const int TURN_SPEED = 70;                 // Geschwindigkeit für Drehung

// Prototypen der Funktionen
float readSensorData(); // Liest die Daten vom Ultraschallsensor
void moveForward(int speed); // Bewegt den Rover vorwärts
void moveBackward(int speed); // Bewegt den Rover rückwärts
void backLeft(int speed); // Dreht den Rover nach links
void backRight(int speed); // Dreht den Rover nach rechts

void setup() {
  SoftPWMBegin(); // Initialisiere SoftPWM für Motorsteuerung
  Serial.begin(9600); // Starte die serielle Kommunikation zur Fehlersuche
}

void loop() {
  float distance = readSensorData();

  if (distance > SAFE_DISTANCE) { // Wenn der Weg frei ist, vorwärts bewegen
    moveForward(FORWARD_SPEED);
  } else if (distance < CLOSE_OBSTACLE_DISTANCE && distance >
     MINIMUM_MEASURABLE_DISTANCE) { // Wenn ein Hindernis nahe ist, aber über dem
     Mindestabstand, rückwärts bewegen
    moveBackward(BACKWARD_SPEED);
    delay(500); // Warte kurz, bevor versucht wird zu drehen
    backLeft(TURN_SPEED);
    delay(1000); // Wartezeit nach der Drehung
```

```
  } else { // Bei mittleren Distanzen vorsichtig vorwärts bewegen
    moveForward(VORSICHTIG_FORWARD_SPEED);
  }
}


// Funktionen-Implementierungen

// Liest die Entfernung vom Ultraschallsensor und berechnet sie
float readSensorData() {
  delay(4); // Wartezeit, um den Sensor zu stabilisieren

  // Trigger-Sequenz senden
  pinMode(ULTRASCHALL_SENSOR_PIN, OUTPUT);
  digitalWrite(ULTRASCHALL_SENSOR_PIN, LOW);
  delayMicroseconds(2);
  digitalWrite(ULTRASCHALL_SENSOR_PIN, HIGH);
  delayMicroseconds(10);
  digitalWrite(ULTRASCHALL_SENSOR_PIN, LOW);

  // Echo lesen
  pinMode(ULTRASCHALL_SENSOR_PIN, INPUT);
  float duration = pulseIn(ULTRASCHALL_SENSOR_PIN, HIGH);

  // Entfernung berechnen und zurückgeben
  return duration * SCHALL_SPEED_CM_PER_US / 2;
}

// Bewegt den Rover vorwärts
void moveForward(int speed) {
  SoftPWMSet(LEFT_MOTOR_FORWARD_PIN, speed);
  SoftPWMSet(LEFT_MOTOR_REVERSE_PIN, 0);
  SoftPWMSet(RIGHT_MOTOR_FORWARD_PIN, speed);
  SoftPWMSet(RIGHT_MOTOR_REVERSE_PIN, 0);
}

// Bewegt den Rover rückwärts
void moveBackward(int speed) {
  SoftPWMSet(LEFT_MOTOR_FORWARD_PIN, 0);
  SoftPWMSet(LEFT_MOTOR_REVERSE_PIN, speed);
  SoftPWMSet(RIGHT_MOTOR_FORWARD_PIN, 0);
  SoftPWMSet(RIGHT_MOTOR_REVERSE_PIN, speed);
}

// Dreht den Rover nach links
void backLeft(int speed) {
  SoftPWMSet(LEFT_MOTOR_FORWARD_PIN, 0);
  SoftPWMSet(LEFT_MOTOR_REVERSE_PIN, 0);
  SoftPWMSet(RIGHT_MOTOR_FORWARD_PIN, speed);
  SoftPWMSet(RIGHT_MOTOR_REVERSE_PIN, 0);
}

// Dreht den Rover nach rechts
void backRight(int speed) {
  SoftPWMSet(LEFT_MOTOR_FORWARD_PIN, speed);
  SoftPWMSet(LEFT_MOTOR_REVERSE_PIN, 0);
  SoftPWMSet(RIGHT_MOTOR_FORWARD_PIN, 0);
  SoftPWMSet(RIGHT_MOTOR_REVERSE_PIN, 0);
```

```
}
```

Reflexion Ultraschallsensor

- Wie erkennt ein Infarotsensor Hindernisse? Können Sie das zugrunde liegende Konzept erläutern? (bei einem Wert von 0 wird ein Hindernis angenommen, und der Rover reagiert entsprechend, um Kollisionen zu vermeiden.)
- Wie erkennt ein Ultraschallsensor Entfernungen? Können Sie das zugrunde liegende Konzept erläutern?
- Wie unterscheidet sich das Hindernisvermeidungssystem mit Infarotsensor von dem Ultraschallsensor? Was sind ihre jeweiligen Vor- und Nachteile?
- Ist es machbar, diese beiden Hindernisvermeidungssysteme (Infarotsensor und Ultraschallsensor) zu kombinieren?
- **Erkennung von Hindernissen durch Infarotsensoren:**
  - Infarotsensoren (IR-Sensoren) erkennen Hindernisse durch die Aussendung von Infrarotlicht und die Messung des reflektierten Lichts. Wenn ein Hindernis nahe genug ist, wird das ausgesendete Infrarotlicht zurück zum Sensor reflektiert. Bei einem Wert von 0 wird angenommen, dass ein Hindernis vorhanden ist, was eine entsprechende Reaktion des Systems auslöst, um Kollisionen zu vermeiden.
- **Erkennung von Entfernungen durch Ultraschallsensoren:**
  - Ultraschallsensoren messen Entfernungen durch die Aussendung von Ultraschallwellen. Diese Wellen treffen auf ein Objekt, werden reflektiert und zurück zum Sensor geschickt. Die Zeit, die vom Aussenden bis zum Empfangen des Echos vergeht, wird gemessen und genutzt, um die Entfernung zum Hindernis zu berechnen.
- **Unterschiede zwischen Infarot- und Ultraschallsensoren im Hindernisvermeidungssystem:**
  - **Infarotsensoren:**
    - Vorteile: Günstig, klein, niedriger Energieverbrauch.
    - Nachteile: Beeinträchtigt durch externe Lichtquellen, begrenzte Reichweite, kann Schwierigkeiten haben, dunkle oder nicht-reflektierende Oberflächen zu erkennen.
  - **Ultraschallsensoren:**
    - Vorteile: Funktioniert gut bei verschiedenen Lichtverhältnissen, kann größere Entfernungen messen, effektiv bei der Erkennung von Objekten unabhängig von deren Farbe und Oberflächenbeschaffenheit.
    - Nachteile: Größer und teurer als IR-Sensoren, möglicherweise anfällig für Ultraschallechos in engen Räumen, die zu falschen Messungen führen können.
- **Kombination von Infarot- und Ultraschallsensoren:**
  - Die Kombination beider Sensorentypen in einem Hindernisvermeidungssystem ist machbar und kann die jeweiligen Schwächen ausgleichen. Durch die Nutzung beider Sensortypen kann ein robusteres System erstellt werden, das in einer Vielzahl von Umgebungen und unter verschiedenen Bedingungen effektiv funktioniert. Die Kombination ermöglicht eine präzisere und zuverlässigere Hinderniserkennung und -vermeidung.