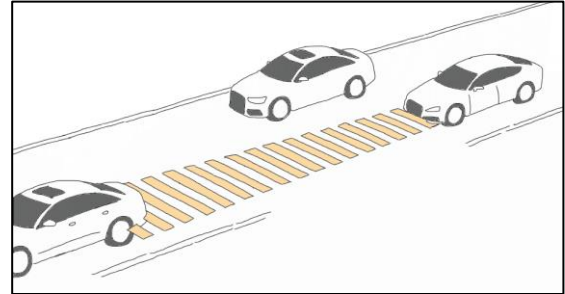


## Arbeitsblatt: Adaptive cruise control (ACC)

### Beschreibung

Adaptive cruise control dient der automatischen Regelung der Fahrzeuggeschwindigkeit. Sobald ein Hindernis vor dem Fahrzeug erkannt wird, passt sich die Geschwindigkeit des Fahrzeuges automatisch an und es wird ein sicherer Abstand eingehalten.



Grafik: R. P. Dröge



### Geschwindigkeiten

Es müssen für die Aufgabenstellung zwei Geschwindigkeiten erfasst werden: Die **Geschwindigkeit des eigenen Fahrzeugs** und die **relative Geschwindigkeit zum vorausfahrenden Fahrzeug**.

Die Geschwindigkeit ( $v$ ) des eigenen Fahrzeugs kann über die zurückgelegte Strecke ( $s$ ) pro Zeit ( $t$ ) berechnet werden.

$$v = \frac{s}{t}$$

Eine relative Geschwindigkeit ist bezogen auf einen bestimmten Standort. Beispiel: Die Geschwindigkeit des eigenen Fahrzeugs beträgt 100 km/h, ein vorausfahrendes Fahrzeug fährt mit 90 km/h. Dann ist die Geschwindigkeit des vorausfahrenden Fahrzeugs relativ zum eigenen -10 km/h, das heißt, das vorausfahrende Fahrzeug nähert sich dem eigenen mit 10 km/h.

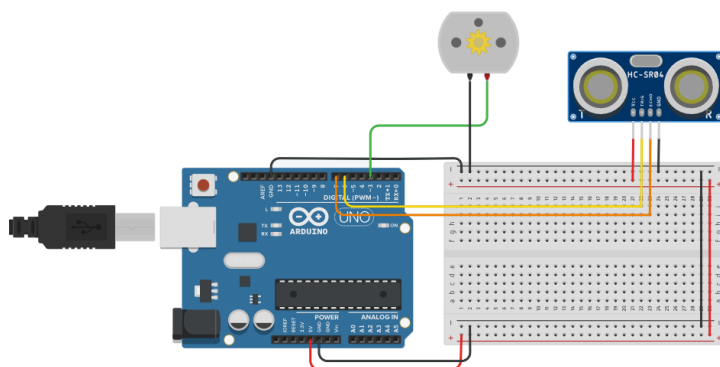


### 1. Erfassen der Relativgeschwindigkeit

Leider kommunizieren Autos nicht untereinander, daher muss die Geschwindigkeit eines vorausfahrenden Fahrzeuges durch Messungen ermittelt werden. Dies kann am einfachsten mit zeitlich versetzten Abstandsmessungen (siehe Projekt „Abblendlicht“) gemacht werden.

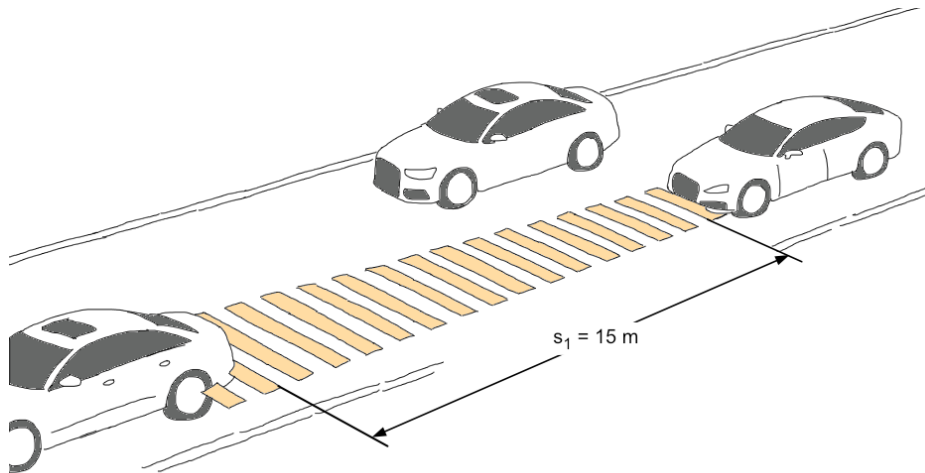
Durch zwei Abstandsmessungen ist es möglich, eine Strecke zu bestimmen. Beispiel: Erste Messung 50 m, zweite Messung 40 m. Daraus folgt: Das vorausfahrende Fahrzeug hat sich um 10 m auf das eigene zubewegt. Die Geschwindigkeit kann bestimmt werden, wenn die Zeit zwischen den Messungen bekannt ist.

### Hardwarekonfiguration

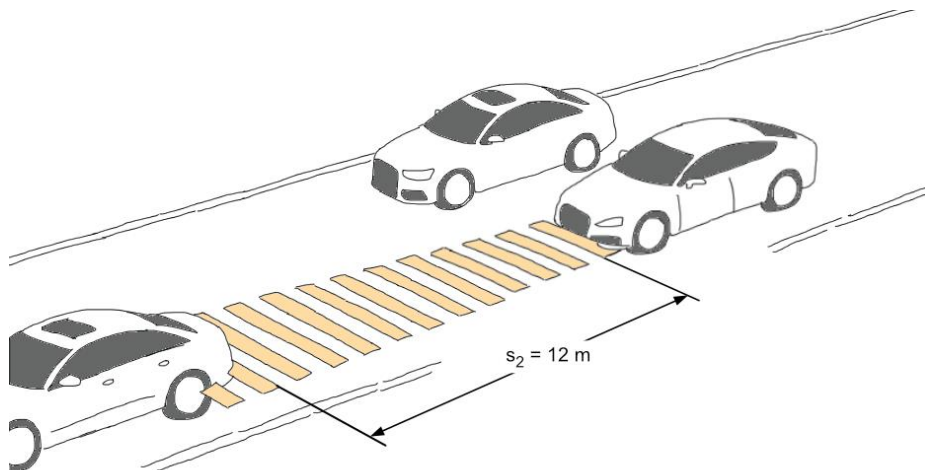


Quelle: BSZ Bietigheim

## Messung 1



## Messung 2 nach 1 s



Grafik: R. P. Dröge

In diesem Beispiel nähert sich das blaue Auto dem roten Auto mit einer relativen Geschwindigkeit von  $-3 \text{ m/s}$ .



### Aufgabe 1:

Erstelle den Programmabschnitt, mit welchem die Relativgeschwindigkeit berechnet werden kann. Diese soll in  $\text{m/s}$  ausgegeben werden. Nutze die folgenden Variablen:

- `s [1]`: erste gemessene Distanz [mm]
- `s [2]`: zweite gemessene Distanz [mm]
- `srel`: zurückgelegte Strecke relativ zum eigenen Fahrzeug als long [mm]
- `vrel`: Relative Geschwindigkeit als int [mm/ms]
- `t`: Zeit zwischen den Messungen als int [ms]

```
srel = s [1] - ;           // Berechnung der relativen Distanz [mm]
vrel =  / ;       // Berechnung der relativen Geschwindigkeit [mm/ms = m/s]
```



## 2. Abstandsmessungen mit einer For-Schleife

Sobald im vorderen Abstandsmesser ein Hindernis auftaucht, sollen zwei Messungen durchgeführt werden, um dadurch die relativ zurückgelegte Strecke zu bestimmen (srel). Dies kann mit einer „For Schleife“ realisiert werden, da dann der Code zur Messung nur einmal im Programm hinterlegt werden muss.

Die For-Schleife kann das, was in ihr steht, eine bestimmte Anzahl oft wiederholen. Sie sieht beim Arduino folgendermaßen aus:

```
for (Startwert; Wahrheitsbedingung; Fortsetzung) {
    „hier steht später der auszuführende Code“
}
```

- *Startwert*

Für den Startwert legen wir eine neue Variable an:

```
„int i = 1“
```

- *Fortsetzung*

Jedes Mal, wenn die Schleife durchlaufen wird, soll i mit 1 addiert werden. Also schreiben wir bei Fortsetzung:  $i = i + 1$ . Die Arduino IDE erlaubt es auch eine Abkürzung dafür zu verwenden, sie lautet:

```
„i++“
```

- *Wahrheitsbedingung*

Wir wollen zwei Messungen durchführen und haben schon eine Variable, die bei jedem Durchlauf der Schleife um 1 größer wird. Die Schleife soll so lange ausgeführt werden, bis  $i = 2$  ist. Die Wahrheitsbedingung ist hier also: „Durchlaufe die Schleife, solange i kleiner oder gleich 2 ist“. In code ausgedrückt sieht das ganze dann so aus:

```
„i<=2“
```



### Aufgabe 2:

Erstelle den Aufruf der For-Schleife:

```
for (  ;  ;  ) {
}
```



Informationen zur  
FOR- Schleife (Ardu-  
ino)



### 3. Variablendeklaration

#### ARRAY

Bei Arrays handelt es sich im Grunde nicht um einen eigenen Variablentyp, sondern um eine Gruppierung mehrerer Variablen eines Typs.

In unserem Programm sollen zwei Messungen innerhalb einer Schleife durchgeführt werden. Um diese zwei Messungen separat speichern zu können, wird die Variable „s“ (Entfernung zum Objekt) als Array angelegt. So kann innerhalb der Schleife die Variable einfach an der Stelle „i“ der gespeichert werden.

Beispiel:

```
long s[3] // Array mit drei Spalten, mit Werten vom Typ long
```

s[0]	s[1]	s[2]
Wird nicht verwendet	Distanz 1 [mm]	Distanz 2 [mm]



Informationen zu  
ARRAY (Arduino)



#### Aufgabe 3: ACC-Programm

Trage die Ergebnisse aus Aufgabe 1 und 2 in das Programm ein und vervollständige alle Kommentare. Teste im Anschluss dein Ergebnis.



```
int trigger=6;
int echo=7;
long dauer=0; // Zeitdauer aus der Schallmessung
long srel;
long vrel;
long s[3]; // Anlegen des Arrays s
int t=800; // Zeit in ms zwischen den Messungen
byte v=1000; // Geschwindigkeit in m/s
byte dcmot=3; // Ausgang DC Motor
void setup()
{
  Serial.begin(9600);
  pinMode(dcmot, OUTPUT);
  pinMode(trigger, OUTPUT);
  pinMode(echo, INPUT);
  analogWrite(dcmot,v); // Motor wird gestartet und mit Geschwindigkeit v
                          betrieben
}
void loop()
{
  for ( [ ] ; [ ] ; [ ] ) { // For-Schleife mit zwei Durchgängen anlegen
    digitalWrite(trigger,LOW); // [ ]
    delay(10);
```

```

digitalWrite(trigger, HIGH); // 
delay(5);
digitalWrite(trigger, LOW);
dauer = pulseIn(echo, HIGH); // 
s[i] = 0.001*dauer*343/2; //Berechnung des Abstandes mit der Formel s = v*t/2
                                analog zur Aufgabe „Abstandsmessung“ an der
                                jeweiligen Stelle von „s“.

delay(t); //Abwarten der Zeit t bis zur erneuten Messung
}
srel = ; // Berechnung der relativen Distanz
vrel = srel / (t*0.05); // Berechnung der relativen Geschwindigkeit
                        "0,05" Anpassung an die Simulation
v = v - vrel; // Berechnung der neuen Geschwindigkeit
analogWrite(dcmot, v); // Ansteuerung des Motors mit neuer
                        Geschwindigkeit
                        //Ausgabe von Variablen an Serial Monitor

Serial.print(srel);

Serial.println("__[mm] srel ");
Serial.print(vrel);
Serial.println("__[m/s] vrel ");
Serial.print(v);
Serial.println("__[m/s] v ");
Serial.println();
}

```



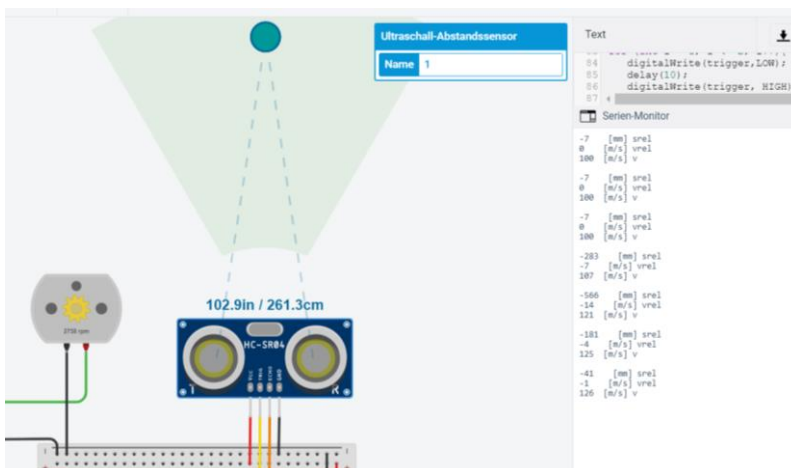
#### Aufgabe 4:

Wie verändert sich die Geschwindigkeit des eigenen Fahrzeugs (v), wenn sich das zweite Fahrzeug nähert?

Das Fahrzeug wird langsamer.

☐

Das Fahrzeug wird schneller.

☐


Quelle: BSZ Bietigheim



**Für Profis:**

Erweitere das Programm, sodass  $v$  sofort auf 0 gesetzt wird, sobald der Abstand zum vorausfahrenden Fahrzeug  $< 30$  cm beträgt. Zusätzlich soll mit einer weiteren FOR-Schleife eine LED fünf Mal im Sekundentakt blinken. Dieser Programmteil simuliert eine automatische Vollbremsung bei einem Hindernis mit Warnblinker.