Workshop: ASURO-Programmieren in C

Teil 2: Praxis

Markus Becker

http://mbecker-tech.de

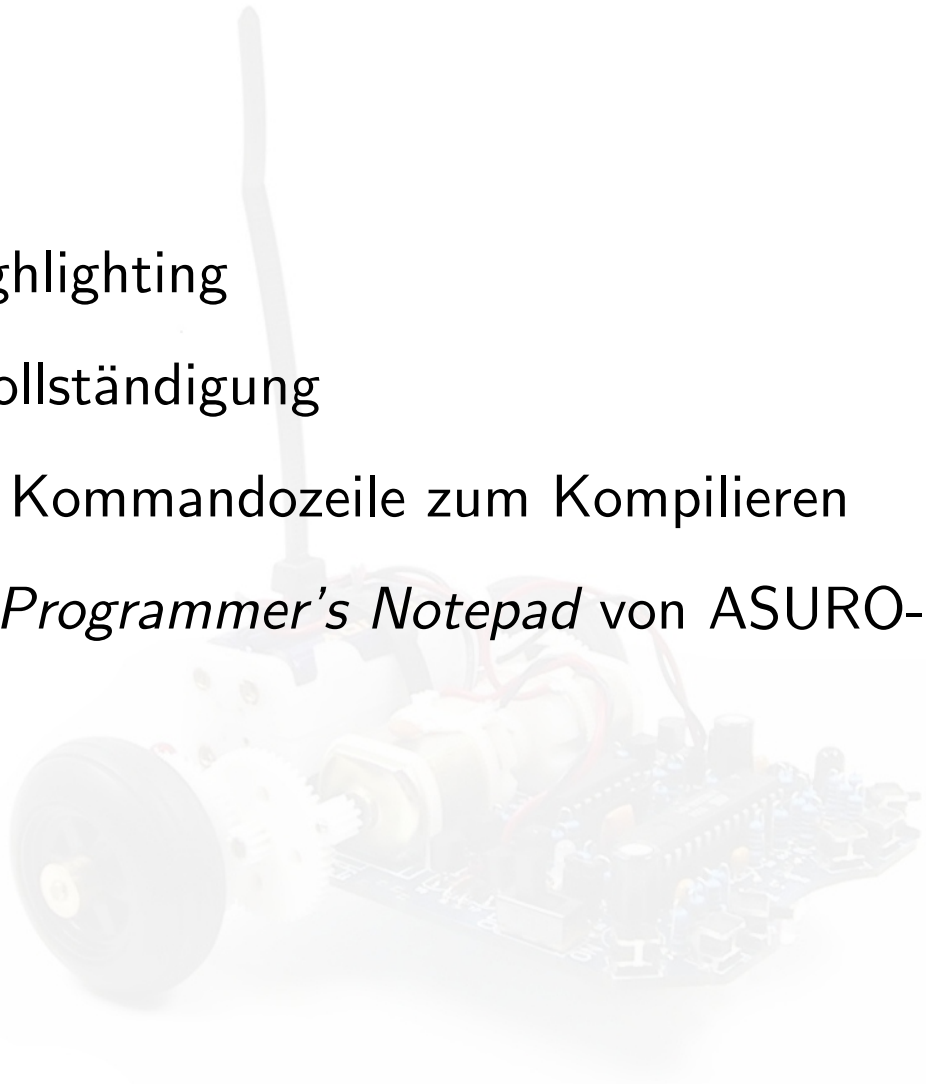
Bürgernetz Ingolstadt e. V. / ByteWerk

Stand: 24. April 2010

Inhaltsverzeichnis

Editor: Notepad++ (Windows) und Geany (Unix)

- Syntax-Highlighting
- Auto-Vervollständigung
- Integrierte Kommandozeile zum Kompilieren
- Windows: *Programmer's Notepad* von ASURO-CD auch möglich

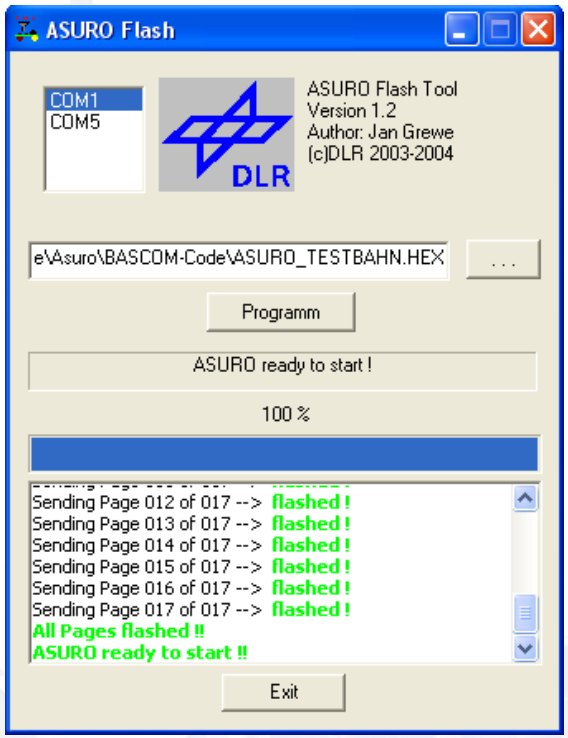


Compiler: avr-gcc

- Standard-Compiler für AVR-Micro-Controller, GPL
- Anpassung über makefile
- Installation mitsamt weiterer Tools über *WinAVR*
- Windows: `PATH`-Umgebungsvariable auf `WinAVR/utils/bin`
- Aufruf über Kommandozeile im Ordner der Quellcode-Dateien
  `C:\Asuro> make all`
- Erzeugt Maschinencode in Form einer `hex`-Datei

Flashen: Flash-Tool (Windows) und asurocon (Unix)

- Zum Übertragen der `hex`-Datei auf den Micro-Controller



Flash-Tool unter Windows

- asurocon-Aufruf: `asurocon /dev/ttyS0 hexfile.hex`
- Ablauf: erst ASURO einschalten, dann Flashen starten

Aufgabe 'Dauerlicht'

- Front-LED einschalten
- Status-LED grün
- Linke Heck-LED an, rechte Heck-LED aus

Hinweis

Da die ASURO-Library über das `makefile` verlinkt wurde ist keine Pfadangabe zur `asuro.h` nötig, es reicht:

```
#include <asuro.h>
```

```c
#include <asuro.h>

void main (void) {
  Init ();
  FrontLED (ON);
  StatusLED (GREEN);
  BackLED (ON, OFF);

  while (1); // Endlosschleife
}
```

Listing 1: Lösung zu Aufgabe 'Dauerlicht'

Aufgabe 'Lauflicht'

Nacheinander folgende LED's leuchten lassen, Zeitabstand 1s

- Front-LED
- Status-LED grün, rot, gelb
- Linke Heck-LED, rechte Heck-LED

Hinweis

Die Funktion `Sleep ()` geht nur bis maximal ca. 3.5ms (originale Library) bzw. 7ms (erweiterte Library),
Lösung: eigene Sleep-Funktion `void sleep_ms (int ms)`

```
1  #include <asuro.h>
2
3  #define DELAY 1000 // 1000ms = 1s
4
5  void sleep_ms (int ms) {
6    while (ms > 0) {
7      Sleep (72); // 1ms
8      ms = ms - 1;
9    }
10 }
11 ...
```

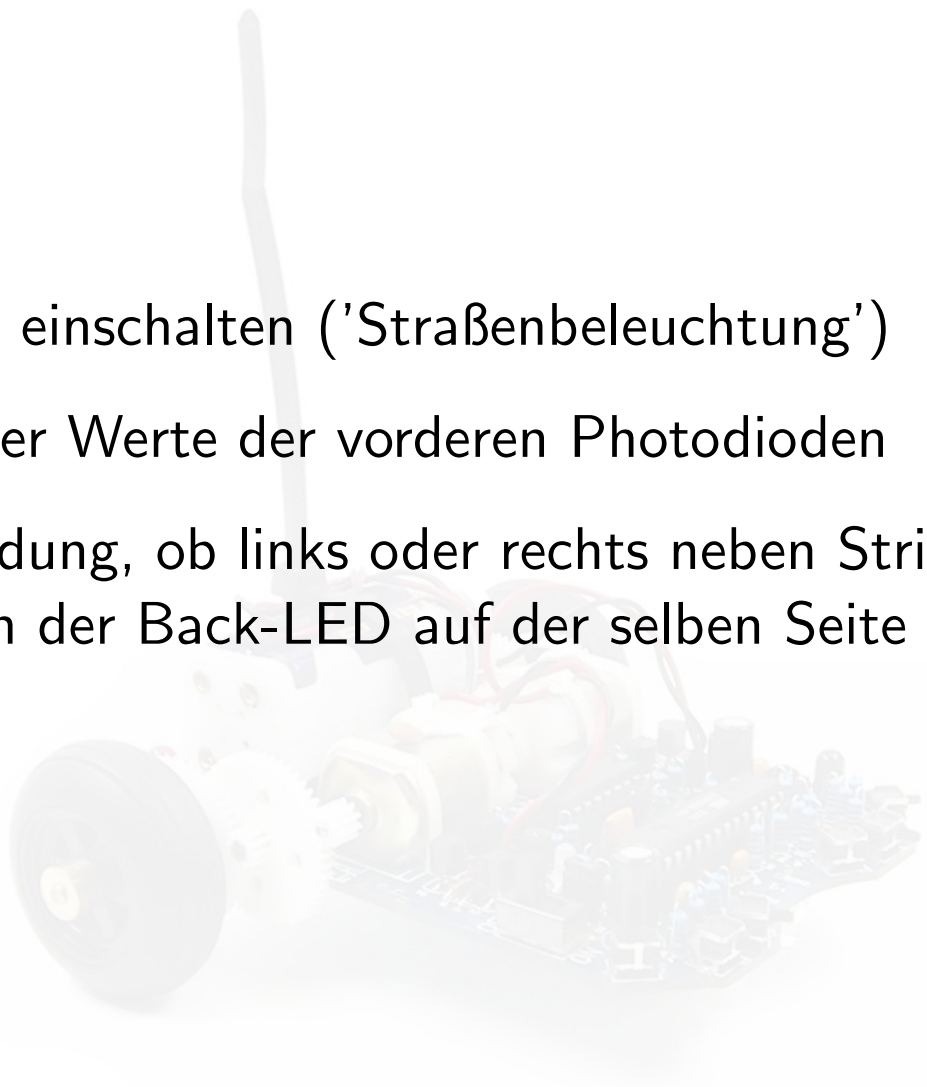
Listing 2: Lösung zu Aufgabe 'Lauflicht': ms_sleep ()

```
1  void main (void) {
2    Init ();
3    while (1) {
4      FrontLED (ON);
5      sleep_ms (DELAY);
6      FrontLED (OFF);
7      StatusLED (GREEN);
8      sleep_ms (DELAY);
9      StatusLED (RED);
10     sleep_ms (DELAY);
11     StatusLED (YELLOW);
12     sleep_ms (DELAY);
13     StatusLED (OFF);
14     BackLED (ON, OFF);
15     sleep_ms (DELAY);
16     BackLED (OFF, ON);
17     sleep_ms (DELAY);
18     BackLED (OFF, OFF);  }  }
```

Listing 3: Lösung zu Aufgabe 'Lauflicht' main ()

Aufgabe 'Linienerkennung'

- Front-LED einschalten ('Straßenbeleuchtung')
- Auslesen der Werte der vorderen Photodioden
- Unterscheidung, ob links oder rechts neben Strich und Einschalten der Back-LED auf der selben Seite

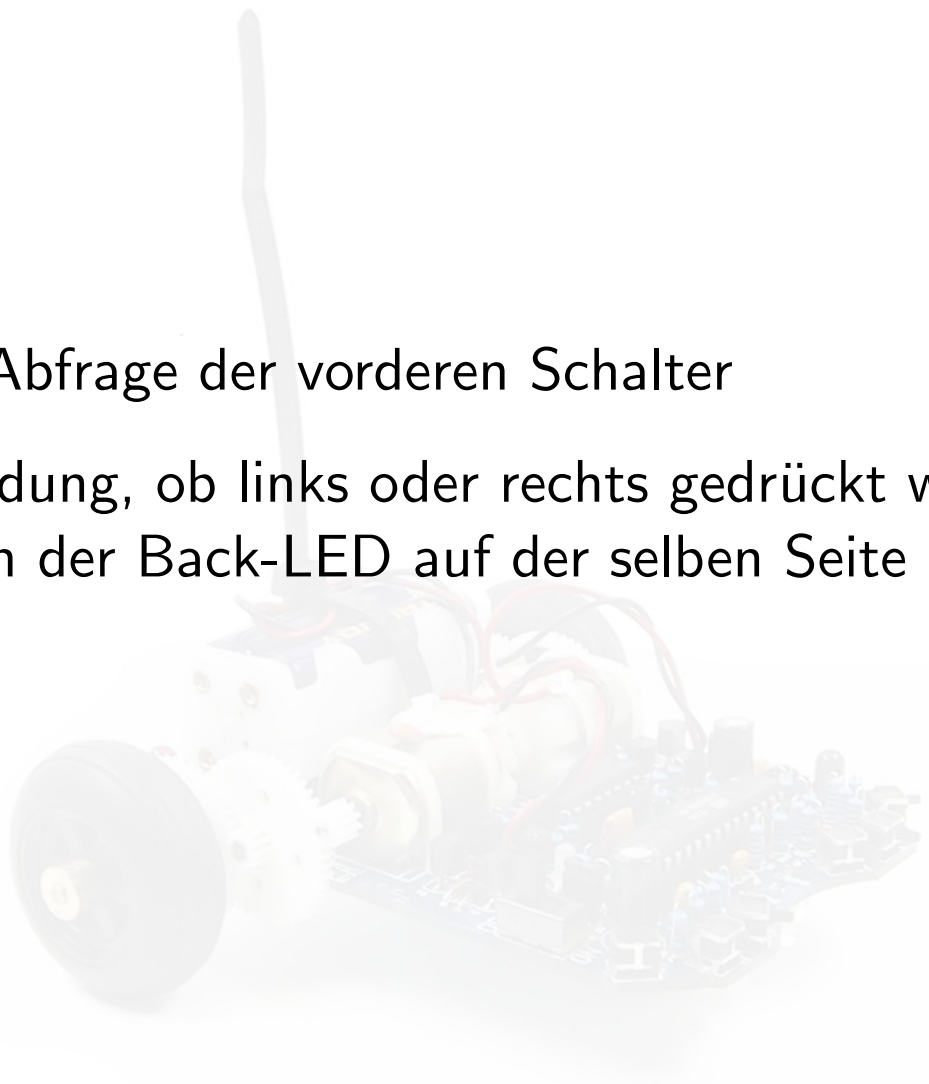


```
1  #include <asuro.h>
2  #define OFFSET 5
3  void main (void) {
4    int data[2];
5    Init ();
6    FrontLED (ON);
7    while (1) {
8      data = LineData ();
9      if (data[0] > data[1] - OFFSET)
10       BackLED (ON, OFF);
11     else if (data[0] < data[1] + OFFSET)
12       BackLED (OFF, ON);
13     else
14       BackLED (OFF, OFF);
15   }
16 }
```

Listing 4: Lösung zu Aufgabe 'Linienerkennung'

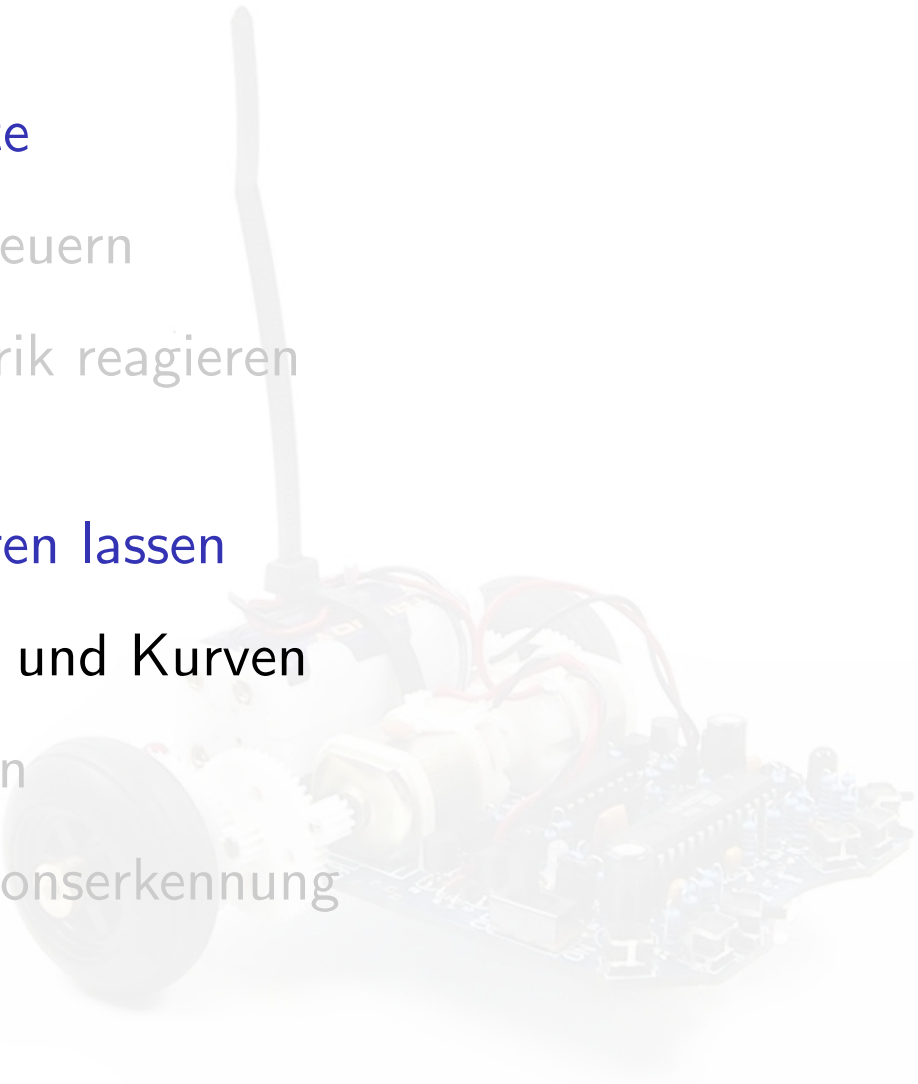
Aufgabe 'Taster erkennen'

- Zyklische Abfrage der vorderen Schalter
- Unterscheidung, ob links oder rechts gedrückt wurde und Einschalten der Back-LED auf der selben Seite



```c
#include <asuro.h>
void main (void) {
  unsigned int switch;
  Init ();
  while (1) {
    switch = PollSwitch ();
    if (switch != 0) { // Schalter gedrueckt
      if (switch <= 7) // Tastsignal RECHTS
            BackLED (OFF, ON);
      else // Tastsignal LINKS
            BackLED (ON, OFF);
    }
    else
      BackLED (OFF, OFF);
  }
}
```

Listing 5: Lösung zu Aufgabe 'Taster erkennen'



Aufgabe 'Indianapolis'

- 2.5s geradeaus fahren (halbe Geschwindigkeit)
- 0.4s Linkskurve (innen Stopp, außen halbe Geschwindigkeit)
- 2.5s geradeaus fahren (halbe Geschwindigkeit)
- 0.4s Linkskurve (innen Stopp, außen halbe Geschwindigkeit)
- ...

Achtung

Nicht auf dem Tisch einschalten bzw. nur wenn festgehalten, damit der ASURO nicht vom Tisch fährt / fällt.

```c
#include <asuro.h>
#define SPEED 127
#define GERADE 2500
#define KURVE 400
void main (void) {
  Init ();
  MotorDir (FWD, FWD);
  while (1) {
    MotorSpeed (SPEED, SPEED);
    sleep_ms (GERADE); // Gerade
    MotorSpeed (0, SPEED);
    sleep_ms (KURVE);  // Kehre
  }
}
```

Listing 6: Lösung zu Aufgabe 'Indianapolis'

Aufgabe 'Linie folgen'

- Front-LED einschalten ('Straßenbeleuchtung')
- Auslesen der Werte der vorderen Photodioden
- Wenn links oder rechts neben Strich: entsprechenden Kurve fahren (*2-Punkt-Regler*)
- Ansonsten: geradeaus fahren

Achtung

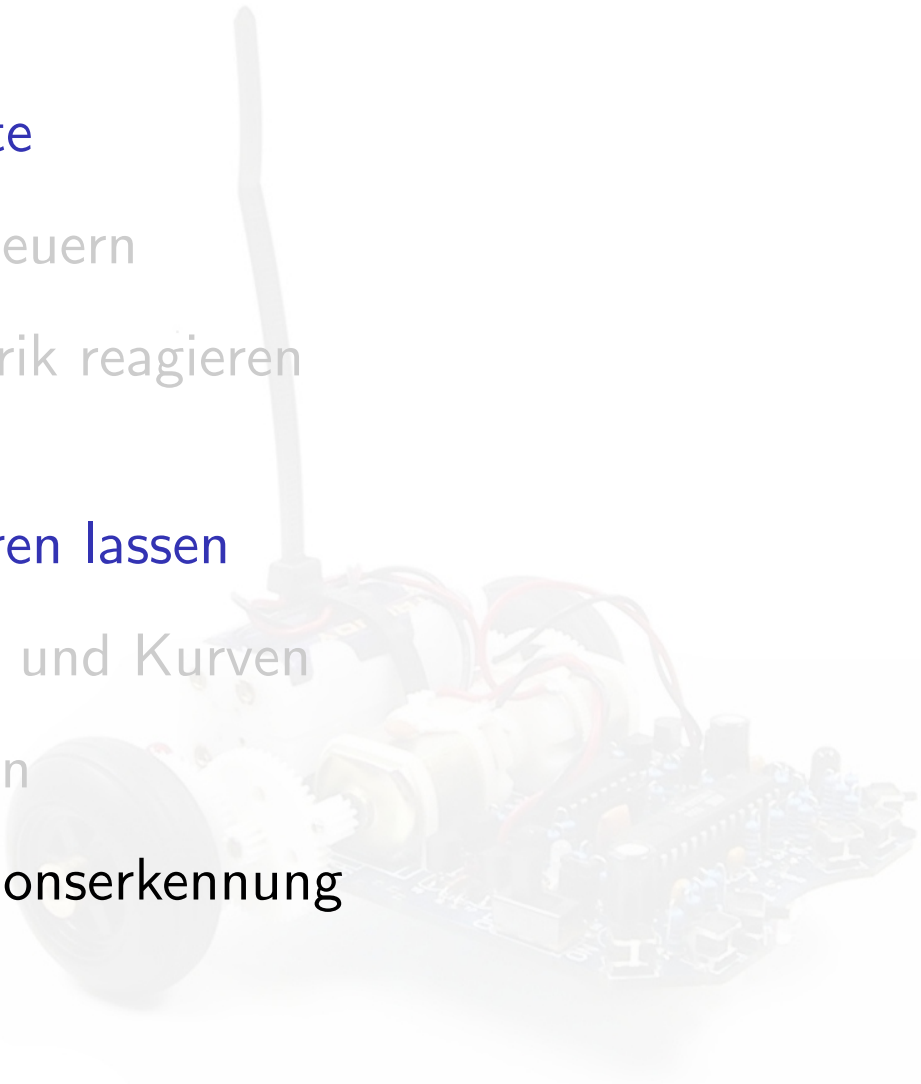
Nicht auf dem Tisch einschalten bzw. nur wenn festgehalten, damit der ASURO nicht vom Tisch fährt / fällt.

```c
#include <asuro.h>
#define SPEED 127
#define OFFSET 5
void main (void) {
  int data[2];
  Init ();
  FrontLED (ON);
  MotorDir (FWD, FWD);
  MotorSpeed (SPEED, SPEED); // losfahren
  while (1) {
    data = LineData ();
    if (data[0] > data[1] - OFFSET)
      MotorSpeed (SPEED, 0); // Rechtskurve
    else if (data[0] < data[1] + OFFSET)
      MotorSpeed (0, SPEED); // Linkskurve
    else
      MotorSpeed (SPEED, SPEED); // Geradeaus
  }
}
```

Listing 7: Lösung zu Aufgabe 'Linie folgen'

Verbesserung des simplen Reglers

- Schwachstellen:
  - Nur beschränkte Geschwindigkeit möglich
  - Keine engen Radien möglich (ASURO verlässt sonst die Linie)
  - Keine Erkennung, ob noch auf der Linie
  - 'Harte' Regelung
- Verbesserungsmöglichkeiten:
  - Zusätzliche Erkennung, ob noch auf der Linie
  - Zustandsautomat zur Speicherung des letzten Zustands (auf der Linie, links oder rechts)
  - Mehrere Regelstufen



Aufgabe 'Linie folgen und Hindernissen ausweichen'

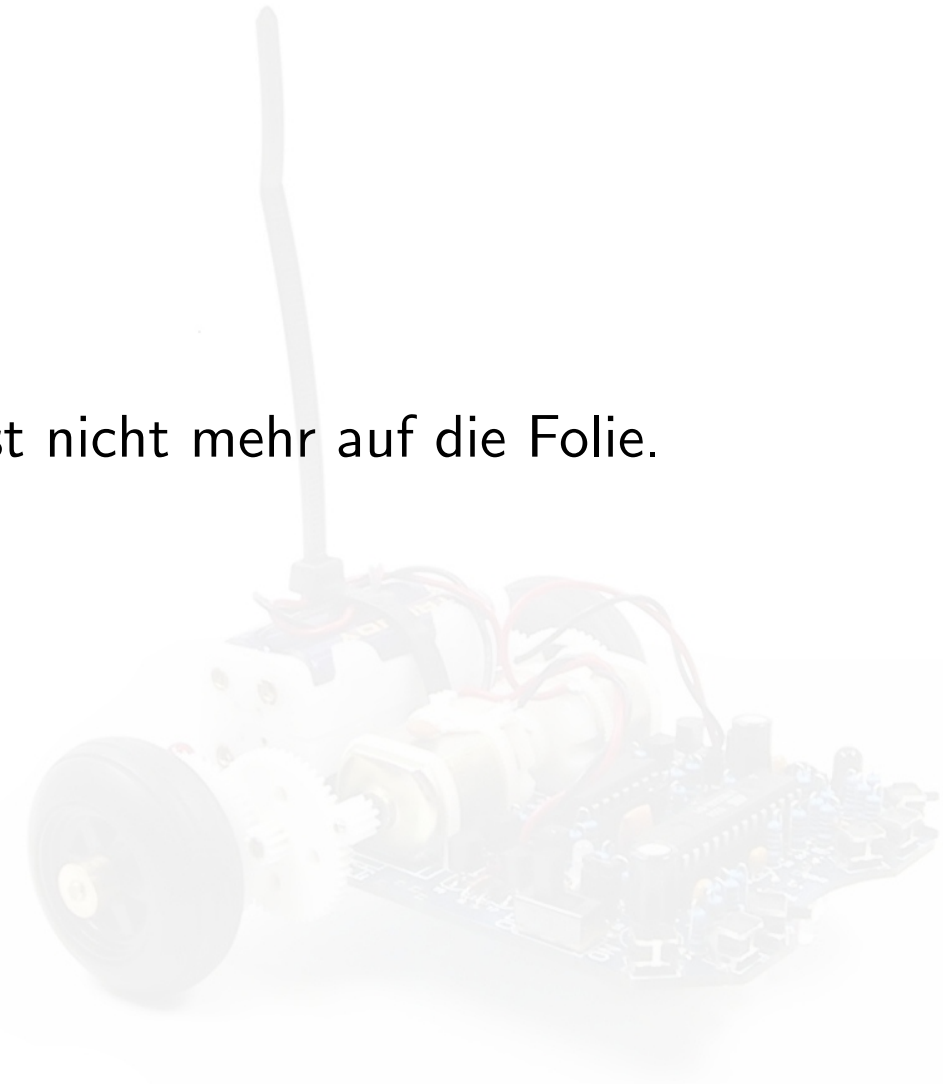
- Front-LED einschalten ('Straßenbeleuchtung')
- Auslesen der Werte der vorderen Photodioden
- Wenn links oder rechts neben Strich: entsprechenden Kurve fahren (*2-Punkt-Regler*)
- Ansonsten: geradeaus fahren
- Taster zyklisch abfragen, um Kollision zu erkennen
- Bei Kollision entsprechend ausweichen und Linie wieder finden, bspw. bei Schalterkontakt rechts: nach 'hinten rechts' fahren

Achtung

Nicht auf dem Tisch einschalten bzw. nur wenn festgehalten, damit der ASURO nicht vom Tisch fährt / fällt.

Lösung zu 'Linie folgen und Hindernissen ausweichen'

Quellcode passt nicht mehr auf die Folie.



Vielen Dank für die Aufmerksamkeit!