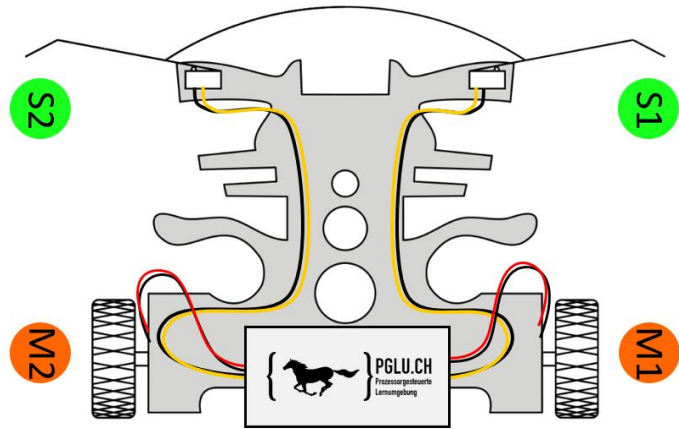


1

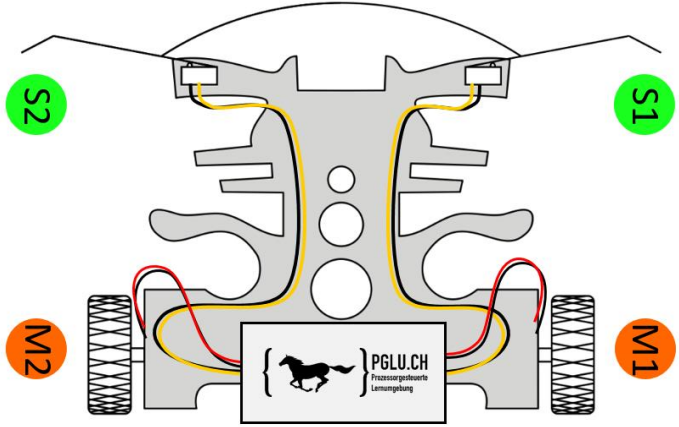


Variante Rückwärts bei Berührung und Vereinfachung mit Funktionen

Jede Fahrrichtung wird in eine Funktion ausgelagert. Das macht das Programm übersichtlicher und einfacher zu erweitern.

The screenshot shows a block-based programming environment with the following components:

- Main Loop:**
 - Wenn Sensor 1 = EIN (checked standard):
 - rueck500
 - links500
 - Sonst: vorwärts
 - Wenn Sensor 2 = EIN (checked standard):
 - rueck500
 - rechts500
 - Sonst: vorwärts
- Functions:**
 - vorwärts:**
 - setze Motor 1 auf 100 %
 - setze Motor 2 auf 100 %
 - rueck500:**
 - setze Motor 1 auf -100 %
 - setze Motor 2 auf -100 %
 - pausiere 500 ms
 - links500:**
 - setze Motor 1 auf 100 %
 - setze Motor 2 auf -100 %
 - pausiere 500 ms
 - rechts500:**
 - setze Motor 1 auf -100 %
 - setze Motor 2 auf 100 %
 - pausiere 500 ms

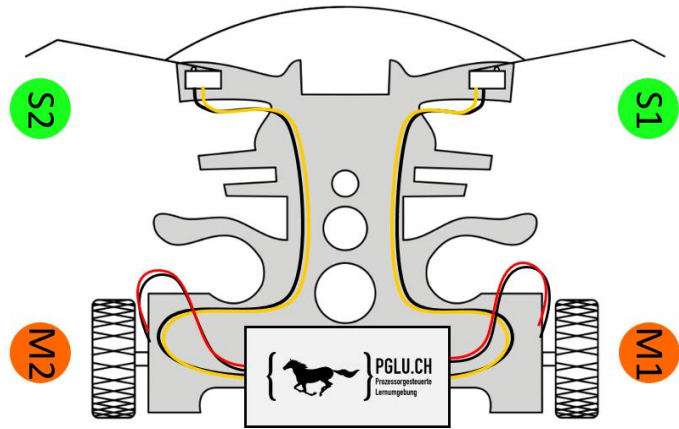


Variante mit Lauflicht bei Drehung – wie im Video auf Webseite

Was nach viel Arbeit aussieht, ist schnell erledigt: du kannst Blöcke und Gruppen mit einem Rechtsklick (lange Berührung) kopieren!

[illegible]

3



Variante für parallele Prozesse

So würde ein Profi das Selbstfahrende Auto programmieren. Da das Programm nie pausieren muss, können die Sensoren immer abgefragt werden und sind dadurch permanent «scharf»

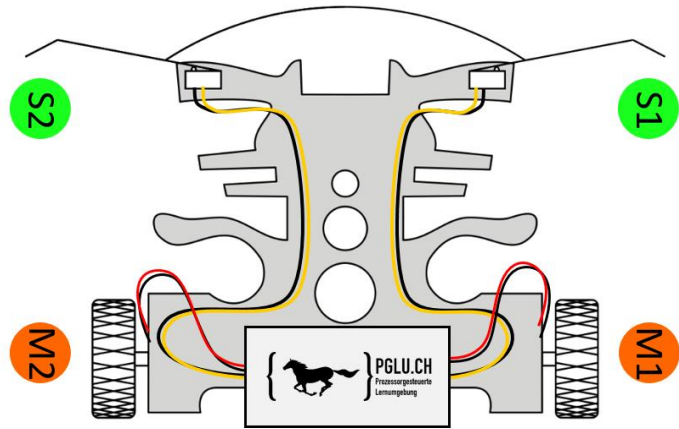
Verwende dieses Programm, wenn du parallel zum Fahrprozess eine Lichtanimation mit Speedypixel programmieren möchtest!

Dieses Programm muss nie pausieren? Wozu dient dann der Block «pauchiere 1ms»? Er garantiert, dass nach 500 Loop-Durchgängen genau 500ms verstrichen sind, was es erleichtert, die Zeit des Rückwärtsdrehens präzise einzustellen.

The screenshot shows the PGLU.CH programming interface with the following components:

- Left Sidebar:** A vertical menu with icons for Sensor, Motor, LED, Pixel, Logik, Frage, Ablauf, Zahl, Variable, Mathe, Struktur, Funktion, Loop, and Spezial.
- Top Tabs:** Grafik, Code, Simulator, and Programme.
- Main Code Area:**
 - Hauptloop: 100'000 mal pro Sekunde** (Blinkcode: kurz 1, lang 1)
 - schreibe Timer1 = Timer1 + 1
 - schreibe Timer2 = Timer2 + 1
 - wenn Sensor 1 = EIN prüfe standard**
 - schreibe Timer2 = 0
 - schreibe Timer1 = 200
 - wenn Sensor 2 = EIN prüfe standard**
 - schreibe Timer1 = 0
 - schreibe Timer2 = 200
 - wenn Timer1 < 500**
 - setze Motor 1 auf -100 %
 - sonst**
 - setze Motor 1 auf 100 %
 - wenn Timer2 < 500**
 - setze Motor 2 auf -100 %
 - sonst**
 - setze Motor 2 auf 100 %
 - pauchiere 1 ms**
 - Vor Hauptloop: 1x**
 - schreibe Timer1 = 500
 - schreibe Timer2 = 500
- Annotations:**
 - A yellow arrow points from the text "Setze für die Simulation hier 20 ein!" to the value 500 in the Timer1 and Timer2 comparison blocks.

4



Variante Lightfollower «Motor so schnell wie hell..»

Zwei Lichtsensoren schauen im 90° Winkel in Fahrtrichtung. S1 sieht Lichtquellen, die 45° von der linken Seite her leuchten und S2 schaut 45° nach vorne rechts. Das Auto folgt der Lichtquelle. Siehe Video 13.

Grafik

Code

Simulator

Programme

Aktion

Sensor

Motor

LED

Pixel

Logik

Frage

Ablauf

Zahl

Variable

Mathe

Struktur

Funktion

Loop

Spezial

Jeder Motor läuft so schnell wie die Helligkeit, die sein gegenüberliegender Sensor misst. Dies würde funktionieren, wenn die Lichtkontraste so wären, dass die Sensoren immer Werte zwischen 0-100% lieferten.

```

Hauptloop: 100'000 mal pro Sekunde  Blinkcode: kurz 1 lang 1
setze Motor 1 auf lese Wert von Sensor 2 in %
setze Motor 2 auf lese Wert von Sensor 1 in %

```

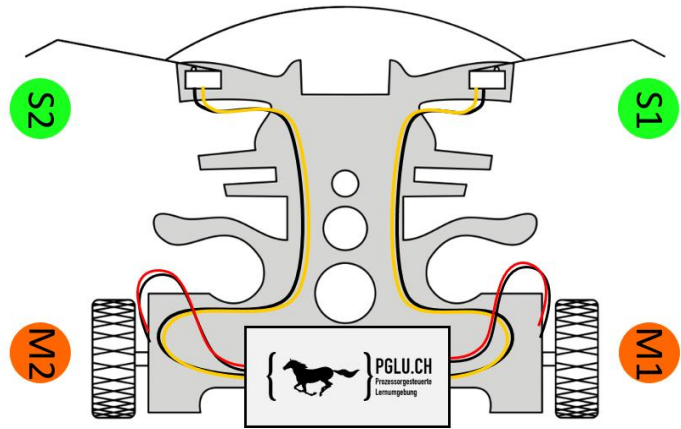
Programm für Umgebungen mit weniger Kontrast. Mit dem Faktor (hier 1.2) wird die Empfindlichkeit der Sensoren erhöht.

```

Hauptloop: 100'000 mal pro Sekunde  Blinkcode: kurz 1 lang 1
setze Motor 1 auf begrenze lese Wert von Sensor 1 in % * 1.2 auf Min 0 und Max 100 %
setze Motor 2 auf begrenze lese Wert von Sensor 2 in % * 1.2 auf Min 0 und Max 100 %

```

5



Variante Linefollower Grundprogramm

Ein Auto mit Lichtsensor fährt einer Linie entlang. Der Fotowiderstand will dabei stets auf der Kante zwischen Boden und Linie (hell und dunkel) bleiben. Driftet er Richtung Linie (hell) korrigieren die Motoren Richtung Boden (dunkel) und umgekehrt. Siehe Video 14. Sensoren siehe Anschluss-schema.

Dieses Grundprogramm funktioniert nur, wenn der Kontrast zwischen Boden und Linie perfekt schwarz-weiss ist.

{ }
Grafik
Code
Simulator
Programme

Aktion

Sensor

Motor

LED

Pixel

Logik

Frage

Ablauf

Zahl

Variable

Mathe

Struktur

Funktion

Loop

Spezial

Hier sind die bekannten Blöcke anders dargestellt, damit sie nicht zu lang werden: Klassischen Block nehmen > Rechtsklick/Longtouch auf Block > «vertikale Darstellung»

Hauptloop: 100'000 mal pro Sekunde Blinkcode: kurz 1 lang 1

setze Motor 1

auf

ändere

lese Wert von Sensor 4 in %

von Min 0

Max 100

zu

Min 0

Max 50

setze Motor 2

auf

ändere

lese Wert von Sensor 4 in %

von Min 0

Max 100

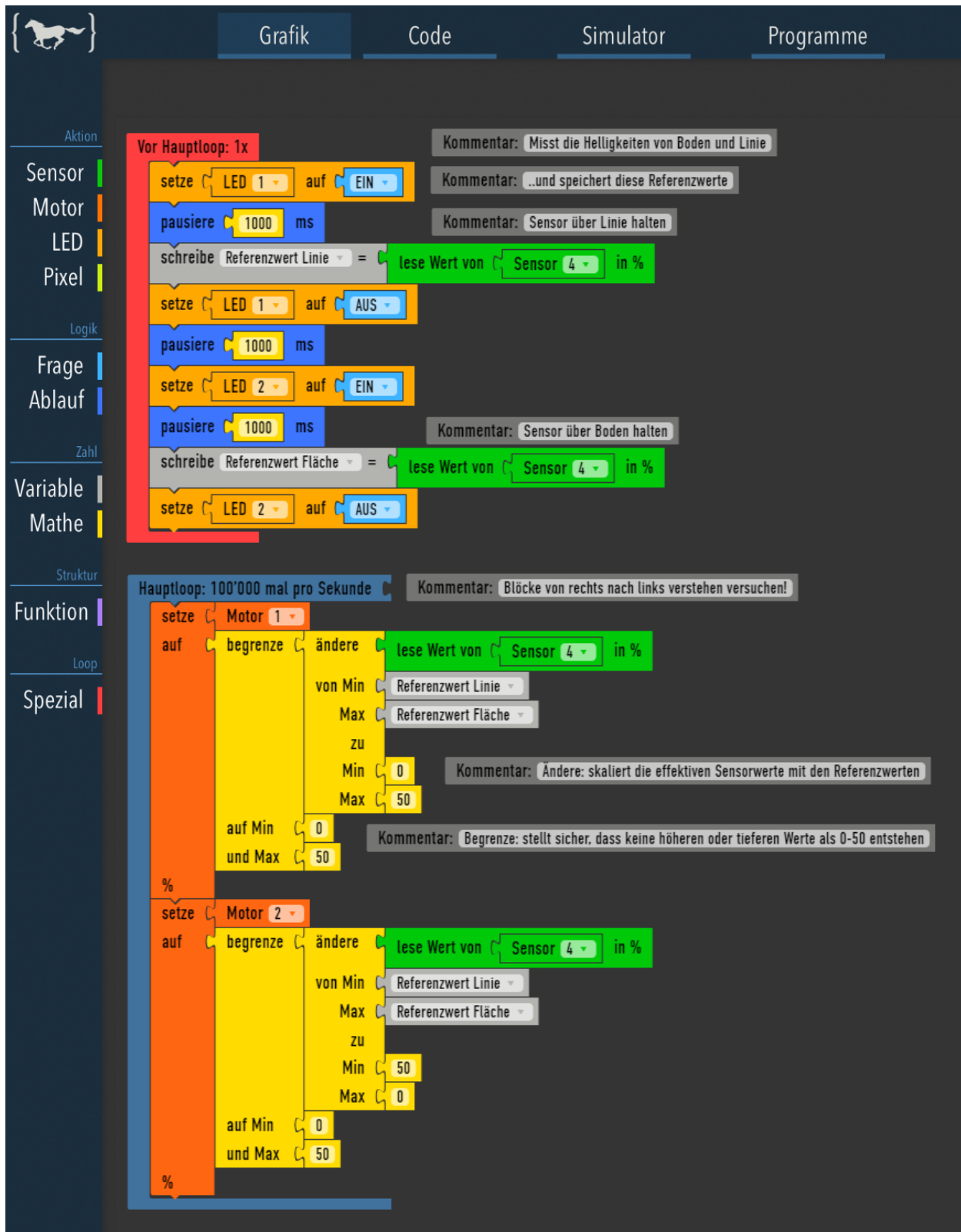
zu

Min 50

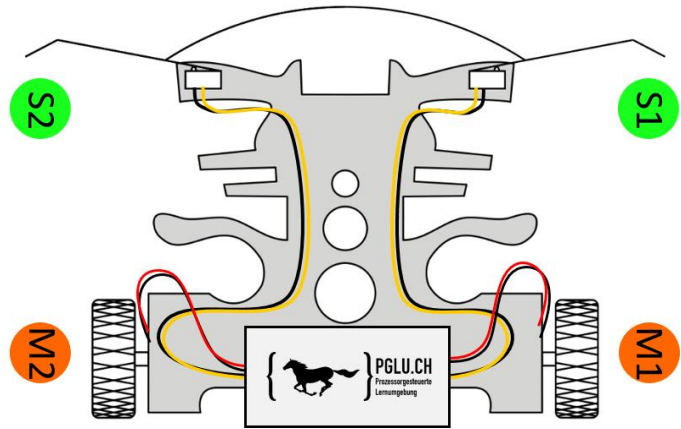
Max 0

Variante Linefollower für Boden und Linie mit wenig Kontrast

Vor dem eigentlichen Programmstart misst der Sensor die Helligkeit von Boden und Linie je eine Sekunde lang. Die Messwerte werden in zwei Variablen gespeichert. Minimale und Maximale Messwerte (dunkel und hell) werden mit dem Block «ändere...» auf die Motorengeschwindigkeiten 0-50 skaliert.



6



Variante Park Pilot

Dieses Auto fährt einen fix vorprogrammierten Parcours. Siehe Video 15

Grafik

Code

Simulator

Programme

Aktion

Sensor

Motor

LED

Pixel

Logik

Frage

Ablauf

Zahl

Variable

Mathe

Struktur

Funktion

Loop

Spezial

Hauptloop: 100'000 mal pro Sekunde

Blinkcode: kurz 1 - lang 1

fahre gerade mit: Zeit = 3000 Tempo = 0

fahre gerade mit: Zeit = 2000 Tempo = 60

fahre gerade mit: Zeit = 700 Tempo = -60

drehe rechts mit: Zeit = 500 Tempo = 60

fahre gerade mit: Zeit = 700 Tempo = -60

Name: fahre gerade mit: Zeit, Tempo

setze Motor 1 auf Tempo %

setze Motor 2 auf Tempo %

pausiere Zeit ms

Name: drehe rechts mit: Zeit, Tempo

setze Motor 1 auf +/- Tempo %

setze Motor 2 auf Tempo %

pausiere Zeit ms

Für Optionen: hier klicken und Variable unten anfügen

In Video 15 nicht verwendete Funktionen

drehe links mit: Zeit = 500 Tempo = 100

beschleunige mit: Zeit = 1000 Tempo A = 30 Tempo B = 100

Name: beschleunige mit: Zeit, Tempo A, Tempo B

zähle Tempo von Tempo A bis Tempo B alle Absolutwert Tempo A - Tempo B / Zeit

setze Motor 1 auf Tempo %

setze Motor 2 auf Tempo %

pausiere 1 ms

Name: drehe links mit: Zeit, Tempo

setze Motor 1 auf Tempo %

setze Motor 2 auf +/- Tempo %

pausiere Zeit ms