Lösungsblatt

# Aufgabe 6: Buggy mit Hinderniserkennung

Die sieben Teilaufgaben der Programmieraufgabe führen Schritt für Schritt in die Synchronsteuerung der Encoder-Motoren ein und bauen unmittelbar aufeinander auf. Die Experimentieraufgabe integriert den Ultraschall-Abstandssensor, der für einen ähnlichen Zweck auch in Aufgabe 8 benötigt wird.

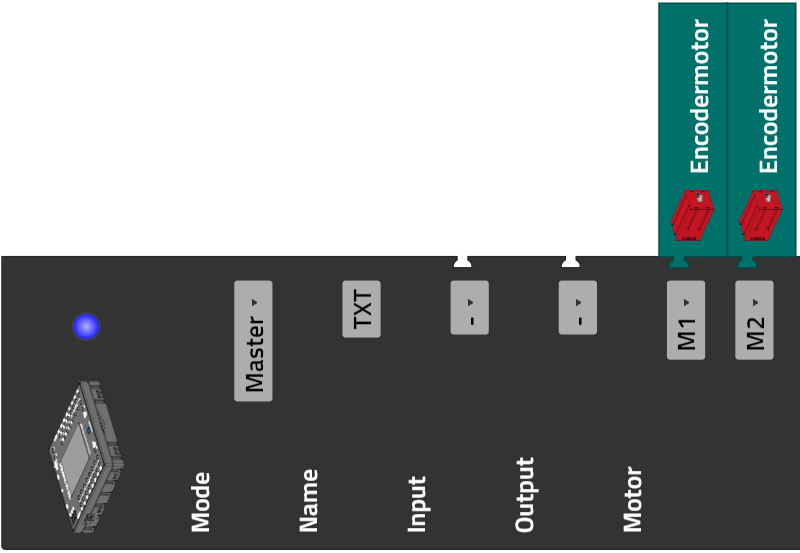
## Konstruktionsaufgabe

Siehe Bauanleitung.

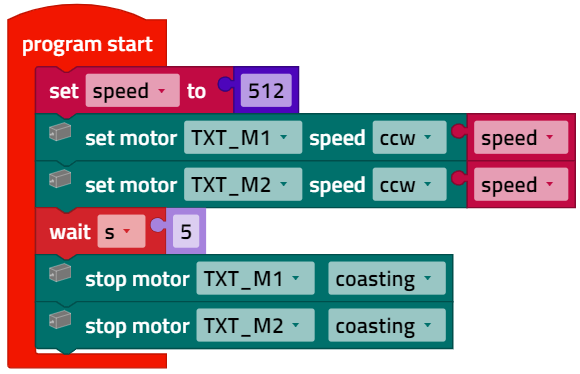
## Programmieraufgaben

**1. Einfache Geradeausfahrt**

Konfiguration der Motoren:



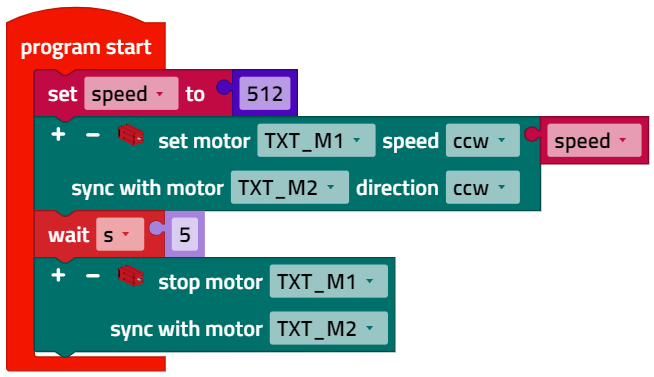
Programm (Beispiel):



*Buggy\_Driving\_Test.ft*

**2. Synchroner Antrieb**

Programm (Beispiel):

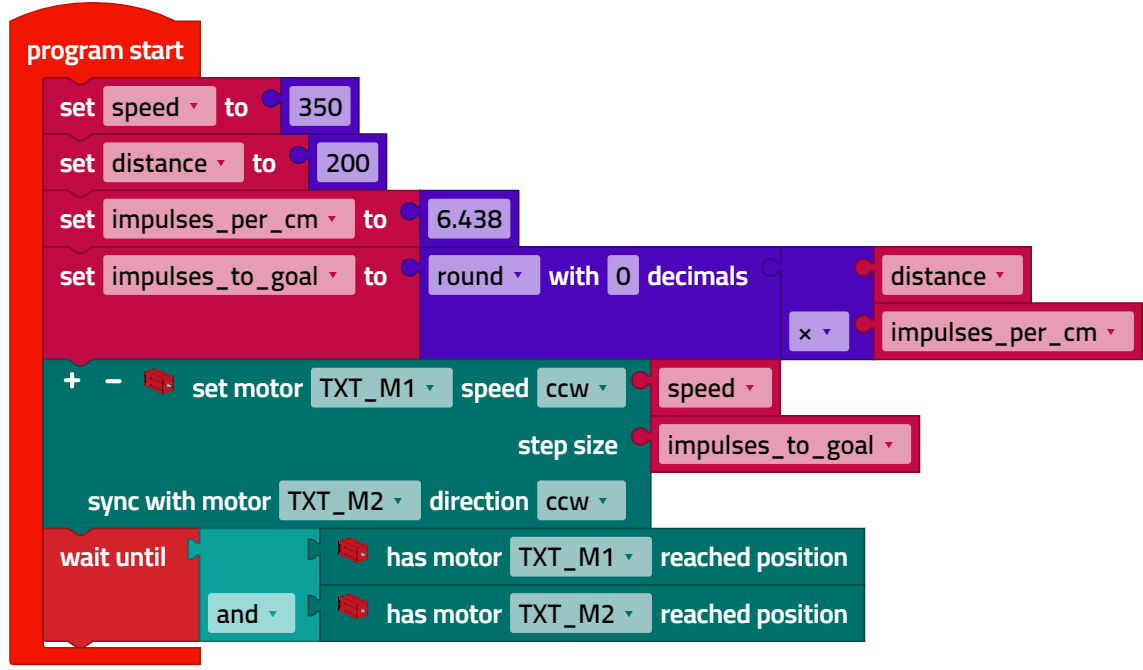


*Buggy\_Driving\_Test\_Synchronous.ft*

**3. Kontrollierte Fahrt mit Encodern**

Die Reifen des Buggys haben einen Umfang von ca. 20,5 cm. Damit liefert der Encoder, da die Räder von den Motoren mit einer Über­setzung von 1:2 ins Langsame angetrieben werden, Impulse/m (oder 6,23 Impulse/cm). Tests über eine Distanz von zwei Metern zeigen, dass der Wert auf etwa **6,438 Impulse/cm** korrigiert werden muss (dieser Wert kann von Modell zu Modell abweichen). Für eine Vorwärtsfahrt von 2 m sind damit 1288 Impulse erforderlich.

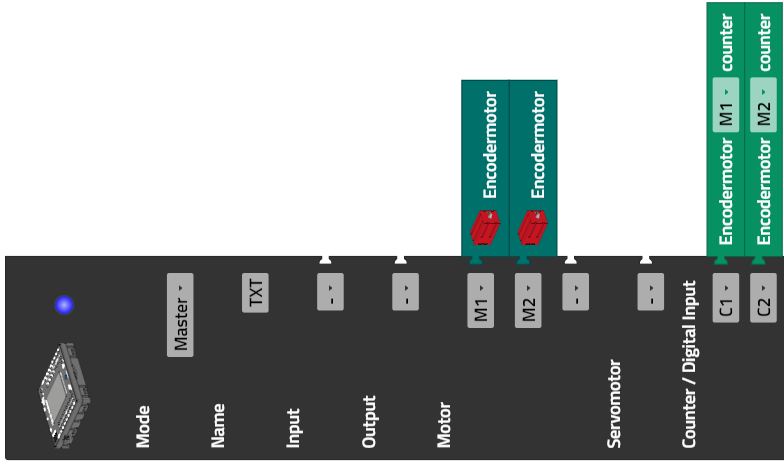
Programm (Beispiel):



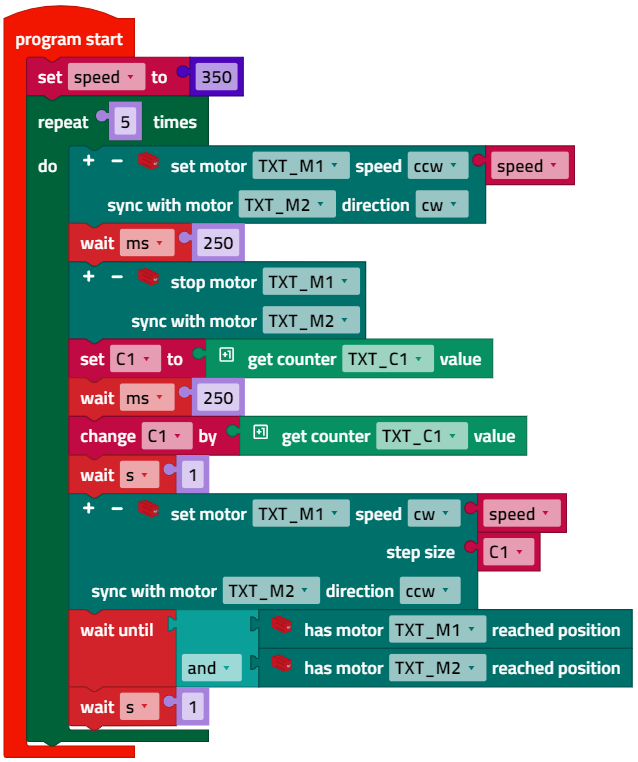
*Buggy\_Driving\_Test\_Synchronous\_Distance.ft*

**4. Synchrones Drehen auf der Stelle**

Konfiguration der Encoder (Counter):



Programm (Beispiel):

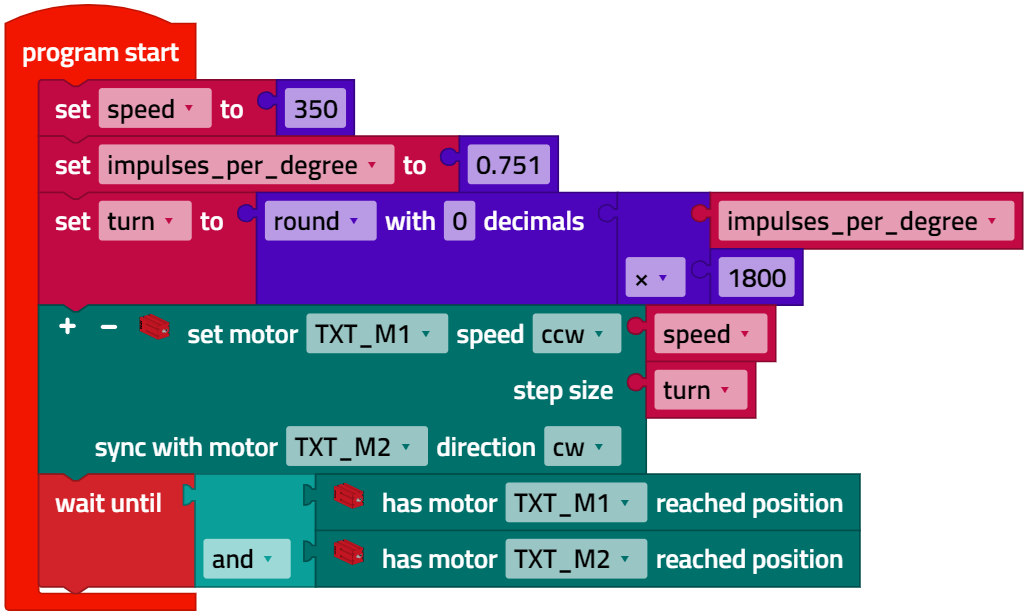


*Buggy\_Turning\_Test\_Synchronous.ft*

**5. Drehen um vorgegebene Winkel**

Beim Drehen auf der Stelle fahren die Reifen – bezogen auf die Reifenmitte – einen Kreis mit einem Durchmesser von etwa 14 cm ab. Das Abfahren des Kreisumfangs entspricht damit etwa 14·π·6,132 ≈ 269,7 Impulsen, also Impulse/Grad. Der Wert ist nur eine grobe Näherung, da die Reifen konstruktionsbedingt nicht genau in der Mitte abrollen. Er kann mit einem einfachen Programm, das den Roboter bspw. fünfmal um die eigene Achse drehen lässt, experimentell überprüft werden.

Programm (Beispiel) zum Testen des Umrechnungsfaktors (5 Drehungen):

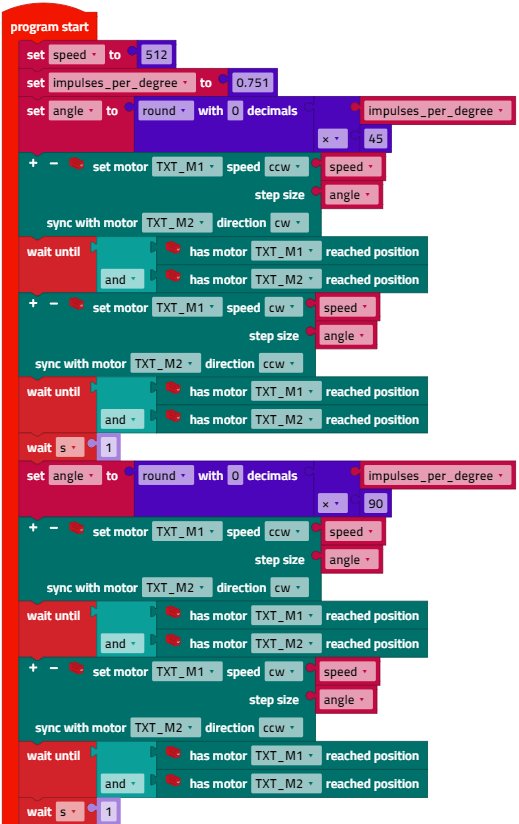
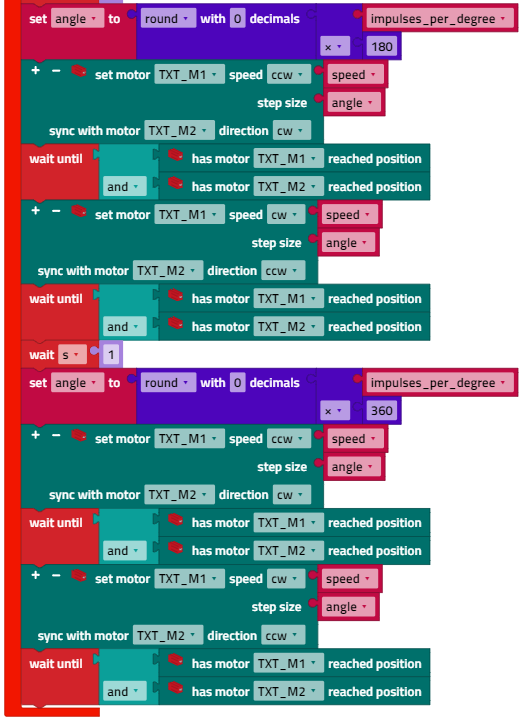


*Buggy\_Synchronous\_Turning\_Calibration.ft*

Die Tests zeigen, dass der Umrechnungsfaktor auf etwa **0,751 Impulse/Grad** angepasst werden muss (dieser Wert kann von Modell zu Modell abweichen).

Eine 90°-Drehung erfordert bei gegen­läufigem Antrieb der Encoder-Motoren also 68 Impulse, eine 45°-Drehung etwa 34 Impulse, eine 180°-Drehung 135 Impulse und eine ganze Drehung 270 Impulse.

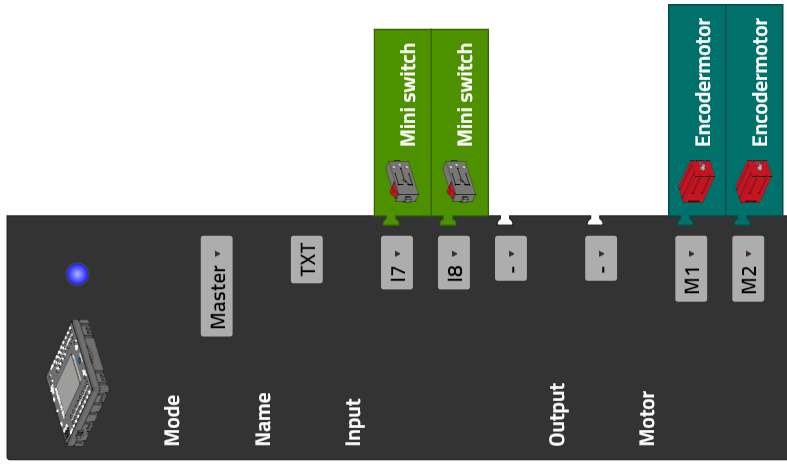
Programm (Beispiel):

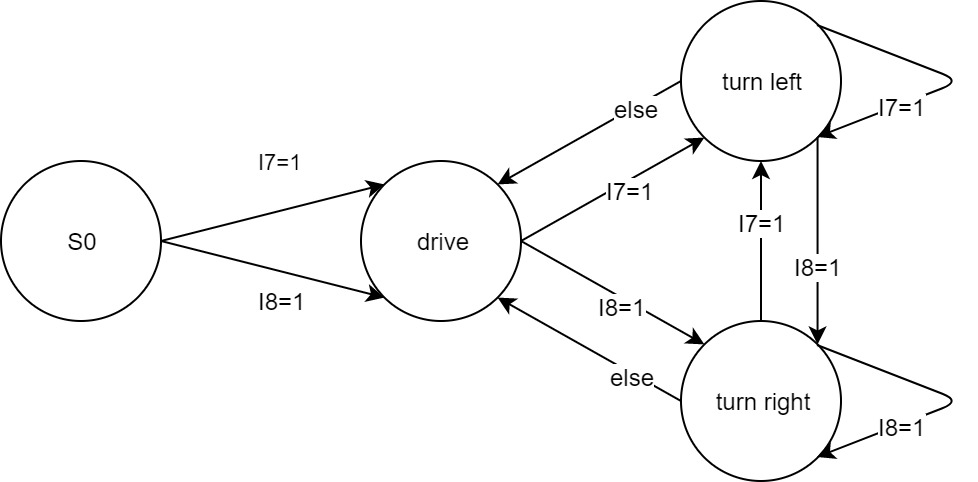
*Buggy\_Turning\_Test\_Synchronous\_Angle.ft*

**6. Buggy mit Hinderniserkennung**

Konfiguration der Taster:



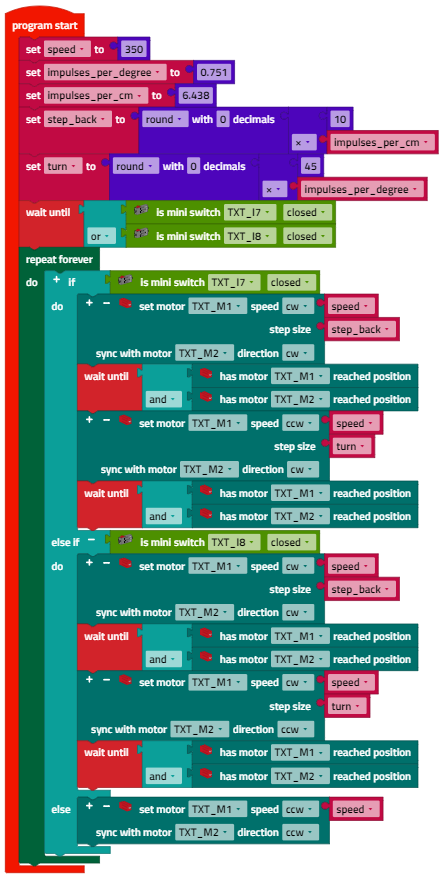
6a. Zustandsübergangsdiagramm:



*State-Transition\_Diagram\_Buggy\_with\_Pushbuttons.drawio*

6b. Das Zurücksetzen des Buggy um 10 cm entspricht 10·6,132 ≈ 61 Impulsen.

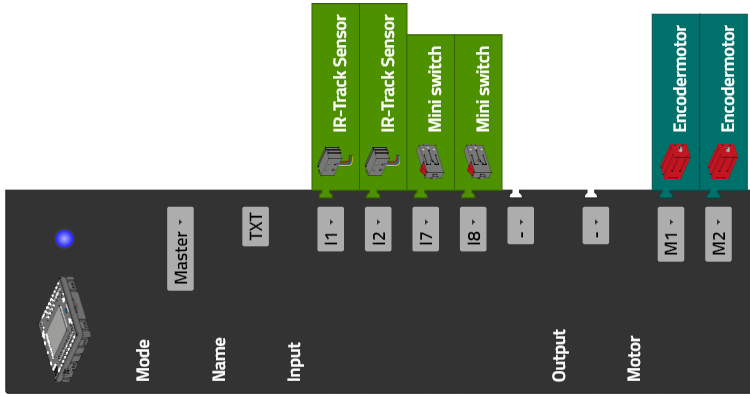
Programm (Beispiel):



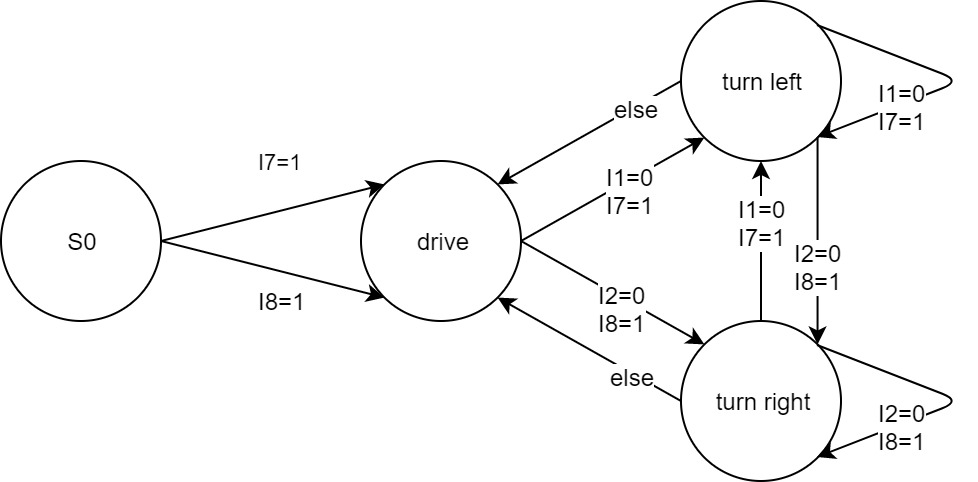
*Buggy\_with\_Pushbuttons.ft*

**7. Buggy mit Linien-/Kantenerkennung**

Konfiguration der IR-Spursensoren:

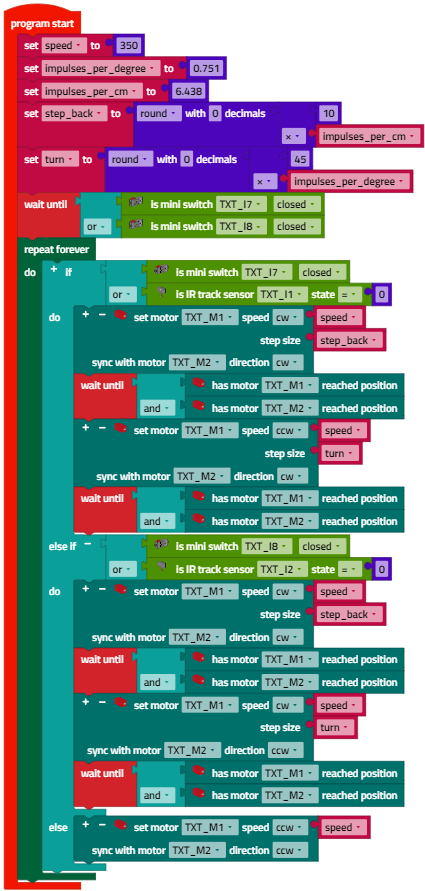


7a. Angepasstes Zustandsübergangsdiagramm:



*State-Transition\_Diagram\_Buggy\_with\_Pushbuttons\_and\_IR\_Sensors.drawio*

7b. Programmerweiterung mit IR-Sensoren:

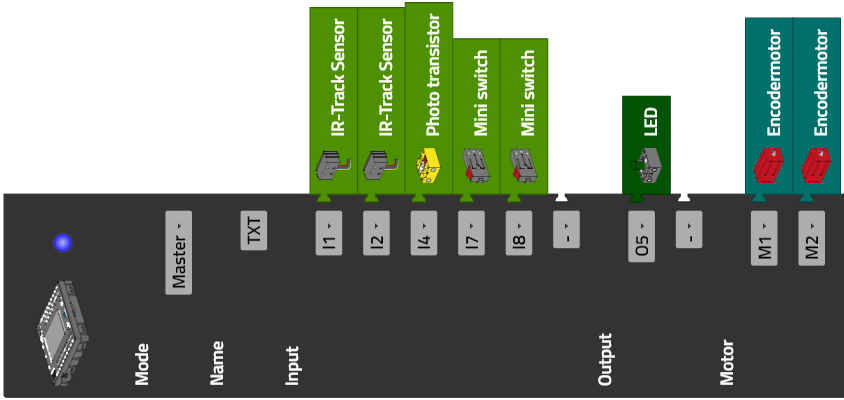


*Buggy\_with\_Pushbuttons\_and\_IR\_Sensors.ft*

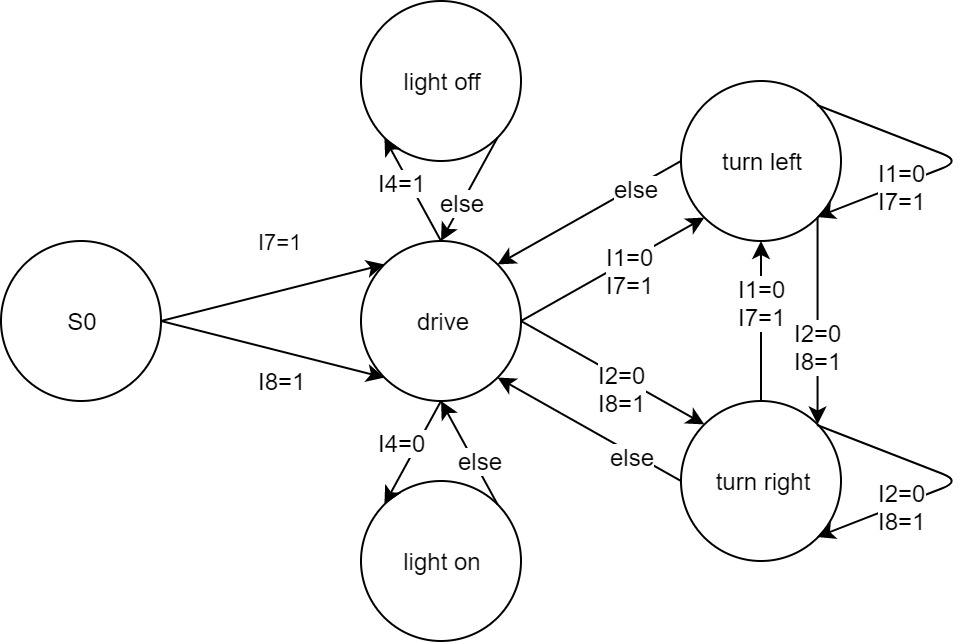
## Experimentieraufgaben

**1. Buggy mit Frontstrahler**

Konfiguration des Fototransistors:

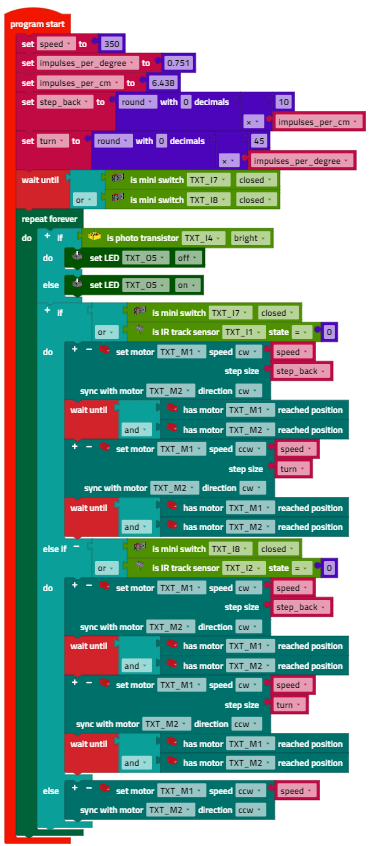


1a. Angepasstes Zustandsübergangsdiagramm:



*State-Transition\_Diagram\_Buggy\_with\_Pushbuttons\_IR\_Sensors\_and\_Light.drawio*

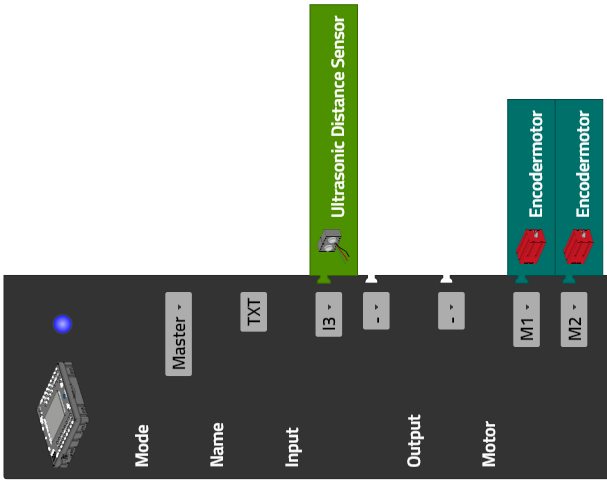
1b. Programmerweiterung mit Fototransistor:



*Buggy\_with\_Pushbuttons\_IR\_Sensors\_and\_Light.ft*

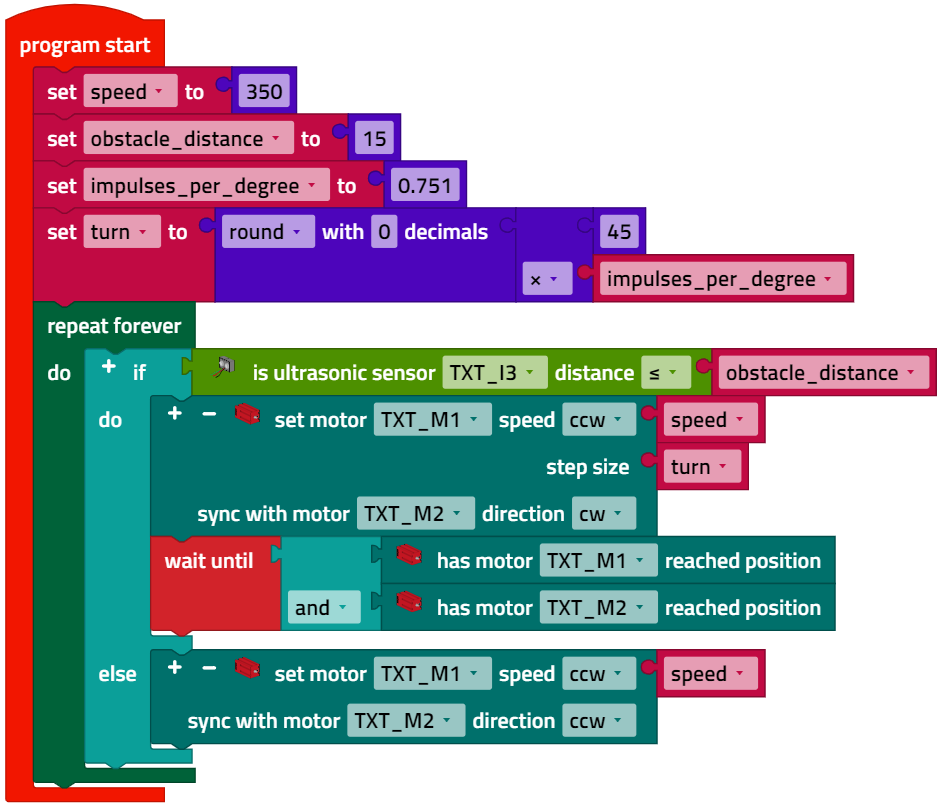
**2. Hinderniserkennung mit Ultraschall**

Konfiguration des Ultraschallsensors:



2a. Programmvariante mit Ultraschall-Sensor zur Hinderniserkennung:

Da das Hindernis vom Ultraschallsensor bereits in einem Abstand von einigen cm erkannt wird, ist ein Zurücksetzen des Buggy nicht erforderlich. Daher wird das Programm sehr kurz.



*Buggy\_with\_Ultrasonic.ft*

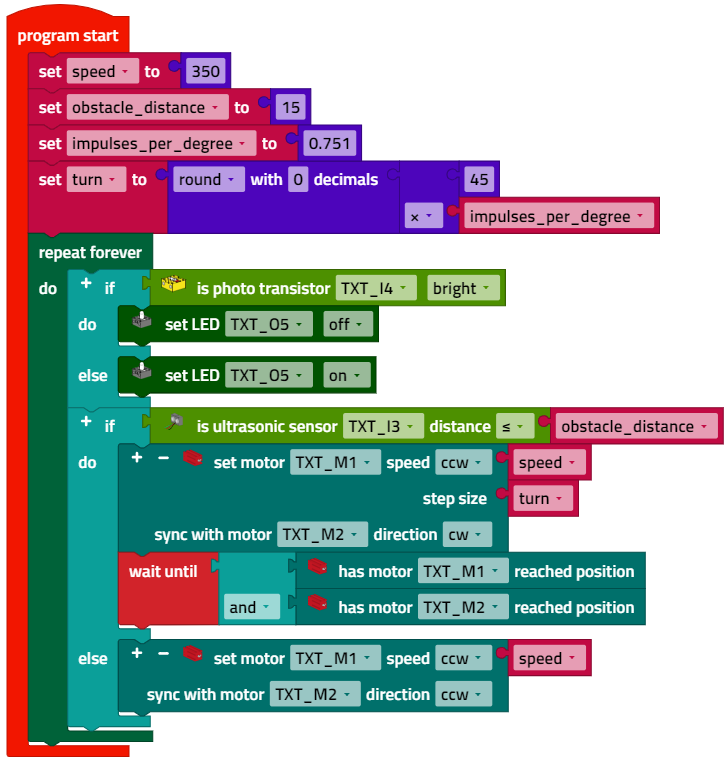
*Achtung*: Der Ultraschallsensor kann Entfernungen von bis zu 4 m messen. Die Laufzeit eines Signals über 2 x 4 m = 8 m liegt für eine Abstandsmessung also bei mindestens 23 ms; ein Messintervall dauert daher 30 ms. Daher kann es bei einer sehr kurzen Distanz dazu kommen, dass das Messergebnis des Sensors sich auf einen über 55 ms zurückliegenden Zeitpunkt bezieht. Die Geschwindigkeit des Buggy muss daher so gewählt werden, dass er während dieses Zeitintervalls nicht zu dicht an das Hindernis heranfährt.

2b. Ergänzung eines Fototransistors und zweier Frontstrahler

Konfiguration des Fototransistors:

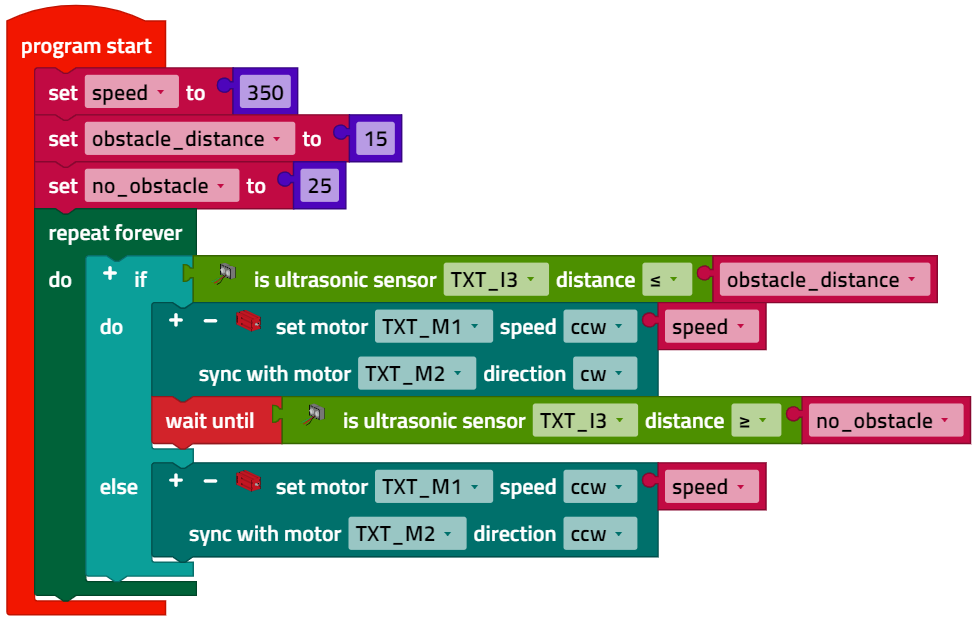


Programm (Beispiel): Ultraschall-Hinderniserkennung, Frontstrahler



*Buggy\_with\_Ultrasonic\_and\_Light.ft*

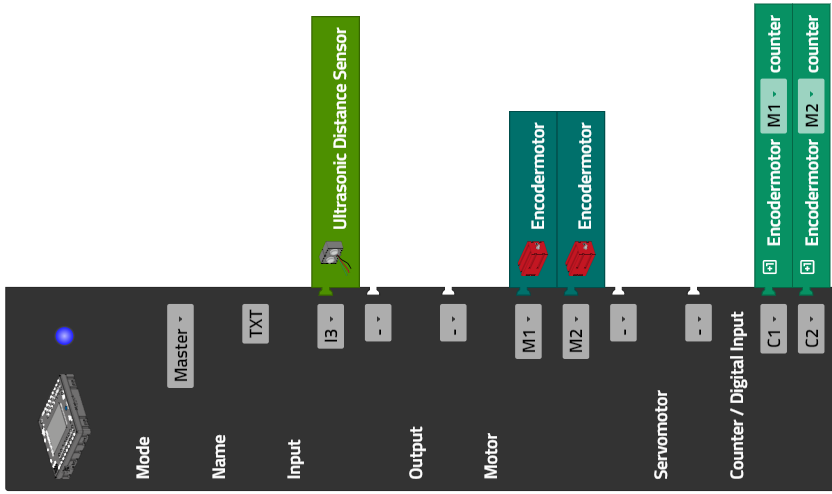
2c. Programm (Beispiel): Drehung, bis kein Hindernis innerhalb von 25 cm Abstand



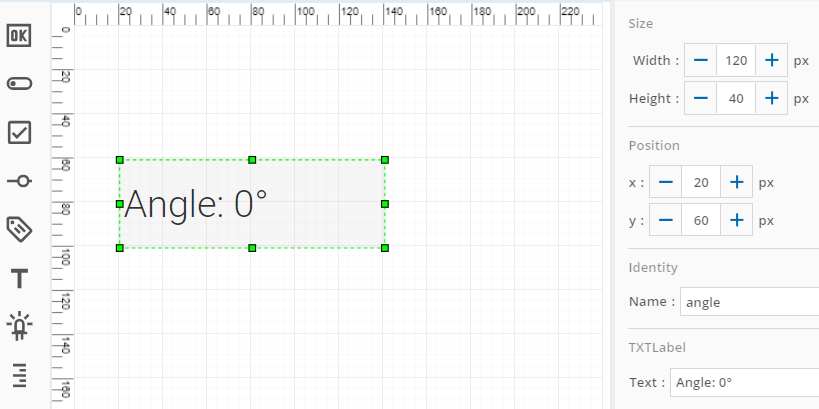
*Buggy\_with\_Ultrasonic\_flexible\_Turn.ft*

2d. Die Umrechnung der Drehimpulse in einen Winkel ist einfach: 360° entsprechen mit unserem experimentell korrigierten Wert etwa 270,36 Impulsen (dieser Wert kann von Modell zu Modell abweichen), wie wir aus der Program­mieraufgabe 5 wissen. Damit erhalten wir aus den Impulsen des Encoders während der Drehung den Winkel mit:

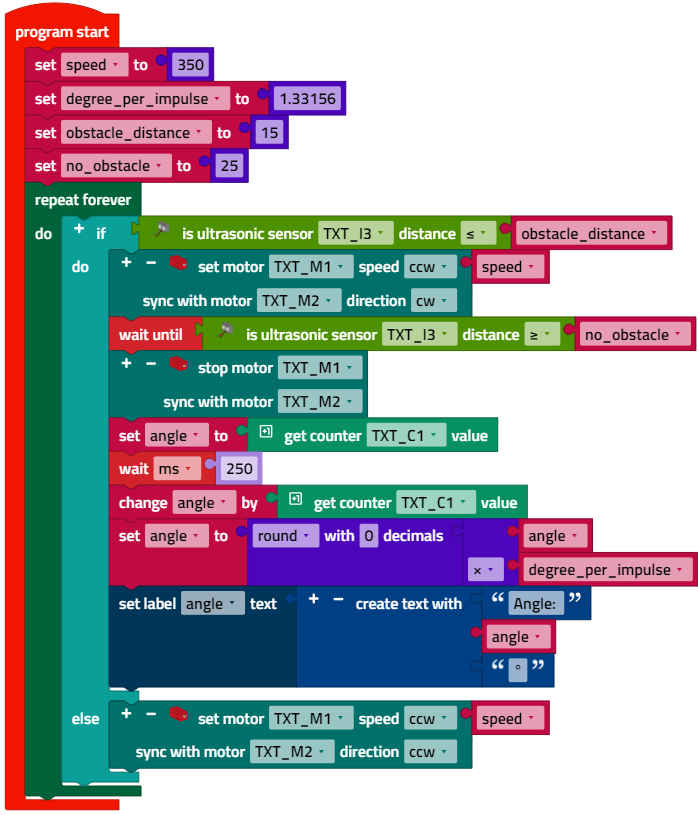
Konfiguration der Zähler:



Konfiguration der Display-Anzeige:



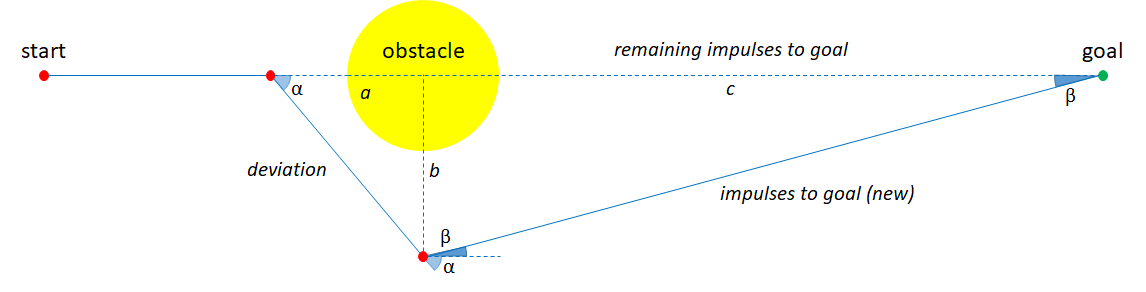
Programm (Beispiel): [Nach Korrektur des Zähler-Bugs Programm anpassen!]



*Buggy\_with\_Ultrasonic\_flexible\_Turn\_display\_Angle.ft*

**3. Buggy mit Encoder-Navigation**

3a. Es gibt unterschiedliche Lösungen für diese Aufgabe. Ein mathematisches Modell (Lösungsbeispiel) zeigt die folgende Skizze:



*Mathematical\_Model\_Target\_Reacher.jpg*

Bei einer Distanz von 3 m ist das Ziel beim Start des Roboters 1869 Impulse entfernt (*impulses to goal*).

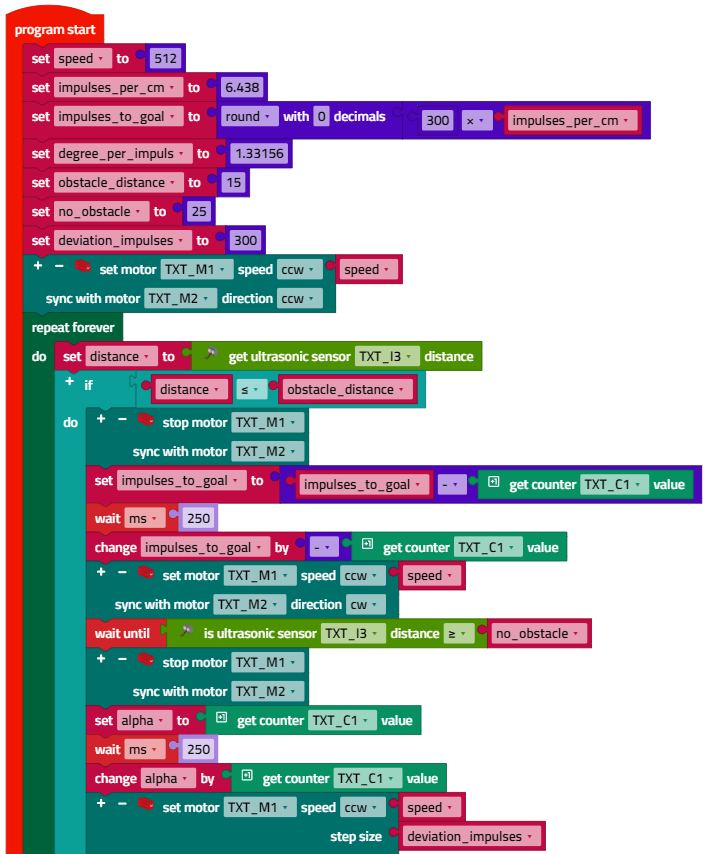
Wenn auf der Fahrt zum Ziel (Luftlinie) ein Hindernis erkannt wird, stoppt der Roboter, bestimmt die verbleibenden Impulse bis zum Ziel (*remaining impulses to goal*) durch Abzug der bereits gefahrenen Impulse und dreht so lange auf der Stelle nach rechts, bis kein Hindernis mehr sicht­bar ist. Dann bestimmen wir den Drehwinkel *α* und der Roboter fährt eine „Umleitungs“-Strecke (*deviation*) vorge­gebener Länge (z. B. 300 Impulse). Anschlie­ßend lassen wir den Roboter nach links um den Winkel *α* zurückdrehen.

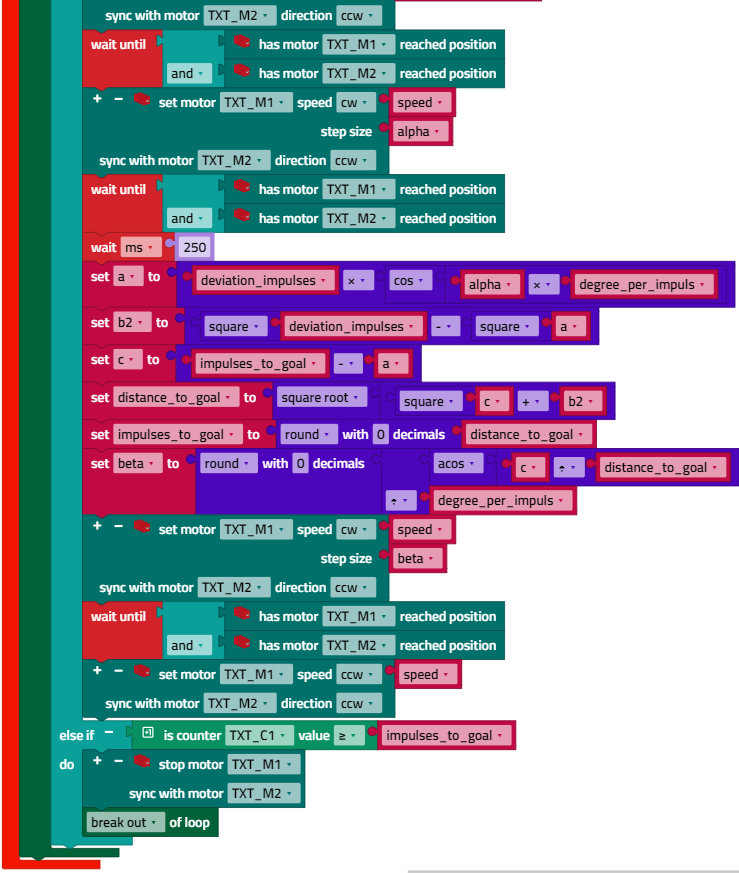
Nun können wir für das – im Bild rechte – rechtwinklige Dreieck die gesuch­ten Abstände zum Zielpunkt und die benötigten Winkel berechnen. Den neuen Abstand zum Ziel erhalten wir nach dem Satz des Pytha­goras:

Dabei erhalten wir aus dem im Bild linken rechtwinkligen Dreieck:

Den Winkel *β*, um den wir den Roboter noch drehen müssen, damit er wieder Richtung Ziel fährt, können wir dann wie folgt bestimmen:

3b. Programm (Beispiel): „Zielfinder“ [Nach Korrektur des Zähler-Bugs Programm anpassen!]





*Buggy\_Target\_Reacher.ft*

**Hinweis**: Die Präzision der Lösung hängt vor allem von der Genauigkeit des Umrechnungsfaktors für die Drehung („degree\_per\_impulse“) ab. Der Wert kann von Modell zu Modell konstruktionsbedingt deutlich abweichen, daher sollte er mit dem Programm zu Programmieraufgabe 5 überprüft werden, falls der Buggy den Zielpunkt verfehlt.

Anlagen

# Aufgabe 6: Buggy mit Hinderniserkennung

## Erforderliches Material

* PC für Programmentwicklung, lokal oder über Web-Schnittstelle.
* USB-Kabel oder BLE- bzw. WLAN-Verbindung für die Übertragung des Programms auf den TXT4.0.
* Parcours-Bogen mit schwarzer, 2 cm breiter, geschlossener Kreislinie
* Hindernis (Karton, Dose, …)

## Weiterführende Informationen

[1] Online-Diagrammeditor zur Erstellung von Zustandsübergangsdiagrammen (Format drawio): <https://www.diagrammeditor.de/>