Programmieren lernen mit EV3

Vom Einsteiger zum Meisterprogrammierer mit LEGO® Mindstorms® EV3

Bearbeitet von
Terry Griffin

1. Auflage 2015. Taschenbuch. XXIV, 264 S. Paperback
ISBN 978 3 86490 275 8
Format (B x L): 20,3 x 25,4 cm

6

Programmablauf

Der *Programmablauf* oder *Programmfluss* ist die Reihenfolge, in der die Programmierblöcke ausgeführt werden. Gewöhnlich geschieht dies von links nach rechts, allerdings kannst du den Ablauf dadurch beeinflussen, dass du Blöcke warten oder wiederholt ausführen oder eine Aktion aufgrund einer Bedingung auswählen lässt. Die drei wichtigsten Blöcke zur Steuerung des Programmablaufs sind der Warte-, der Schalter- und der Schleifenblock.

Wie der Warteblock funktioniert, hast du bereits gesehen. In diesem Kapitel schauen wir uns den Schalter- und den Schleifenblock genauer an. Außerdem erkläre ich den Schleifen-Interrupt-Block, da er zur Steuerung von Schleifenblöcken dient.

HINWEIS Der Schalter- und der Schleifenblock verfügen auch über einige Merkmale, die nur bei der Verwendung von Datenleitungen zum Tragen kommen. Sie werden in den Kapiteln 9 und 10 behandelt.

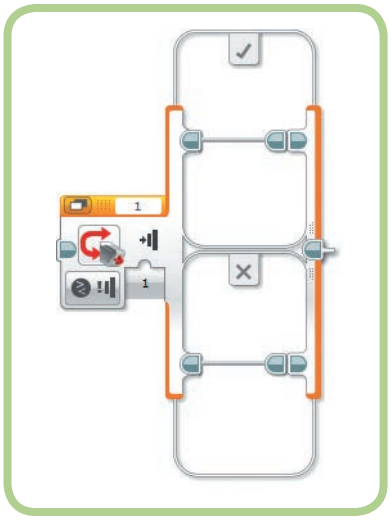


Abbildung 6-1: Der Schalterblock

Der Schalterblock

Im Schalterblock (siehe Abbildung 6-1) kann dein Programm entscheiden, welche Blöcke ausgeführt werden sollen. Dies wird als eine *bedingte* Struktur bezeichnet, da sich der Programmablauf in Abhängigkeit davon ändert, ob eine Bedingung erfüllt ist oder nicht. Der Schalterblock prüft die Bedingung und trifft dann auf der Grundlage des Ergebnisses eine Auswahl zwischen zwei oder mehr Gruppen von Blöcken, den sogenannten *Fällen*. Dadurch kann das Programm eine Entscheidung treffen und auf die Daten von den Sensoren des Roboters reagieren. Beispielsweise entscheidet das Programm *RedOrBlue* weiter hinten in diesem Kapitel anhand der Messwerte des Farbsensors, welchen Klangblock es ausführen soll.

Die Bedingung festlegen

Um in einem Schalterblock die Bedingung einzurichten, legst du zunächst in der Modusauswahl einen Sensor und einen Modus fest. Anschließend gibst du den Schwellenwert und jegliche sonstigen Parameter an, z. B. den Sensoranschluss.

Die Auswahl des Modus kannst du dir so verstellen, als würdest du eine Frage stellen. Der Block in Abbildung 6-1 entspricht der Frage: »Ist der Berührungssensor gedrückt?« Da es sich hierbei um eine Ja/Nein-Frage handelt, kann der Schalterblock nur über zwei Fälle verfügen, nämlich den für *wahr* und den für *falsch*.

Bei anderen Fragen sind mehr als zwei Antworten möglich. Da der Farbsensor acht verschiedene Werte erkennen kann (sieben Farben und *Keine Farbe*), gibt es auf die Frage »Welche Farbe hat der Farbsensor wahrgenommen?« acht mögliche Antworten. Der Schalterblock kann dann also bis zu acht Fälle umfassen.

Die Größe eines Blocks ändern

Die Größe des Schalter- und des Schleifenblocks wird automatisch angepasst, wenn du Blöcke in sie hineinziehst. Du kannst die Größe dieser Blöcke aber auch selbst anpassen, etwa sie verkleinern, nachdem du einige der darin vorhandenen Blöcke entfernt hast, um in der Registeransicht des Schalterblocks weitere Register anzuzeigen oder um Platz für Kommentare zu schaffen.

Wenn du auf einen Schleifenblock oder auf einen Fall eines Schalterblocks klickst, werden Griffe zur Größenänderung angezeigt (siehe Abbildung 6-2). Wenn du auf diese Griffe klickst und an ihnen ziehst, kannst du die Größe des Blocks verändern.

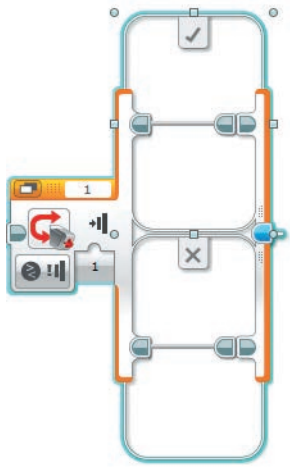


Abbildung 6-2: Griffe zur Größenänderung

Das Programm LineFollower

LineFollower ist ein einfaches Spurfolgeprogramm, das anhand eines Schalterblocks entscheidet, in welche Richtung der Roboter fahren soll. Der TriBot verwendet die Daten vom Farbsensor im Modus **Stärke des reflektierten Lichts**, um der Kante einer Linie zu folgen.

Um dieses Programm einsetzen zu können, musst du die Stoßstange mit dem Berührungssensor vom Vorderteil des TriBots abbauen und durch den Farbsensor ersetzen. Montiere den Farbsensor wie in Abbildung 6-3 so, dass er nach unten zeigt.

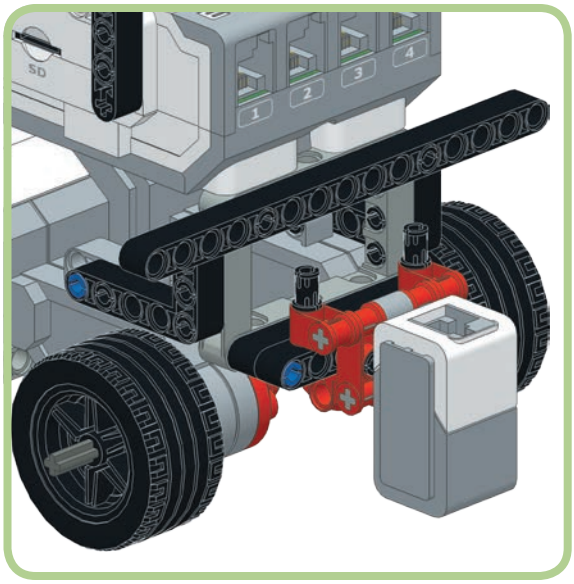


Abbildung 6-3: Der Farbsensor in der Stellung zum Verfolgen einer Linie

Um das Programm zu testen, brauchst du eine dunkle Linie, der der Roboter folgen kann. Dazu kannst du mit schwarzem Filzstift oder Klebeband ein Oval auf weißen Karton auftragen (siehe Abbildung 6-4). Die Linie sollte mindestens 3 cm breit sein und viel dunkler als der Untergrund.

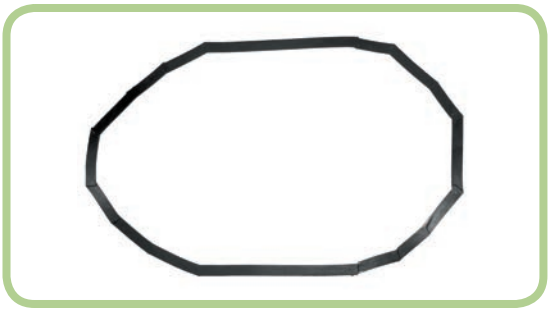


Abbildung 6-4: Eine Testlinie aus Klebeband auf Karton

HINWEIS Dieses Programm funktioniert am besten, wenn die Linie sanfte Kurven aufweist. Scharfe Ecken können Probleme verursachen. Die endgültige Version des Programms aus Kapitel 19 kann jedoch besser mit Ecken fertig werden.

Das Grundprogramm

Dieses Programm regelt fortlaufend nach, in welche Richtung der Roboter steuert, sodass der Farbsensor immer über dem Rand der Linie bleibt (siehe Abbildung 6-5). Dadurch kann der Roboter der Linie folgen. Im Modus **Stärke des reflektierten Lichts** gibt der Messwert des Farbsensors an, wie viel Licht von einem kleinen kreisförmigen Bereich unterhalb des Sensors zurückgeworfen wird. Befindet sich der Sensor über dem weißen Untergrund, ergibt sich ein hoher Messwert, da ein Großteil des Lichts reflektiert wird. Wenn der Sensor dagegen komplett über der dunklen Linie steht, wird nur wenig Licht zurückgeworfen, sodass ein kleiner Wert gemessen wird. Die Werte zwischen diesen beiden Extremen hängen davon ab, wie weit der Sensor über der Linie steht. Während der Roboter vorwärts fährt, liest er anhand der Messwerte des Farbsensors ab, wo er sich relativ zum Rand der Linie befindet. Ist er zu weit über der Linie, lenkt er nach links zurück zur Kante. Bewegt er sich dagegen von der Linie weg, steuert er nach rechts, um zu ihr zurückzugelangen.

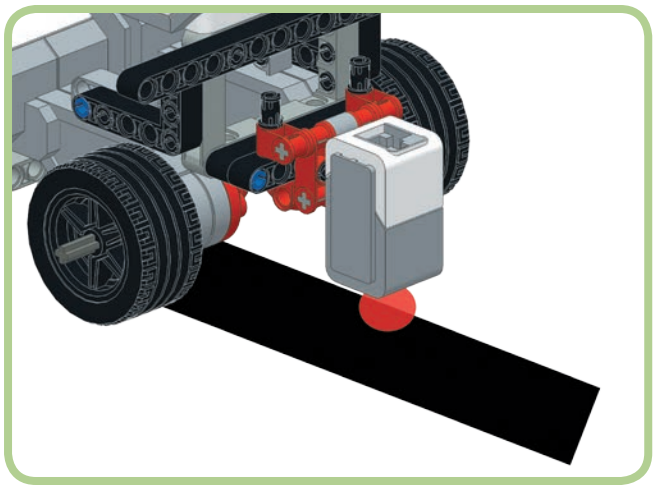


Abbildung 6-5: Der Farbsensor über dem Rand der Linie

Das Programm entscheidet sich in einem Schalterblock zwischen zwei Bewegungslenkungsblöcken, von denen der eine den Roboter nach links und der andere nach rechts fahren lässt. Abbildung 6-6 zeigt das fertige Programm. Es steht komplett in einem Schleifenblock, sodass es wiederholt ausgeführt wird, bis du es abbrichst. Der Schalterblock liest die Daten des Farbsensors und entscheidet auf dieser Grundlage, welcher Bewegungslenkungsblock ausgeführt werden soll. Dabei lenkt der Bewegungslenkungsblock im oberen Teil (im Fall *wahr*) den TriBot nach links, derjenige im unteren Abschnitt (Fall *falsch*) nach rechts.

Schreibe jetzt das Programm:

1. Lege ein neues Projekt namens *Chapter6* an.
2. Lege ein neues Programm namens *LineFollower* an.
3. Füge einen Schleifenblock hinzu.
4. Ziehe einen Schalterblock in den Schleifenblock. (Der Schleifenblock dehnt sich aus, um Platz für den Schalterblock zu schaffen.)
5. Füge in die beiden Fälle des Schalterblocks je einen Bewegungslenkungsblock ein.
6. Markiere den Schalterblock und setze den Modus auf **Farbsensor ▸ Vergleichen ▸ Stärke des reflektierten Lichts.**

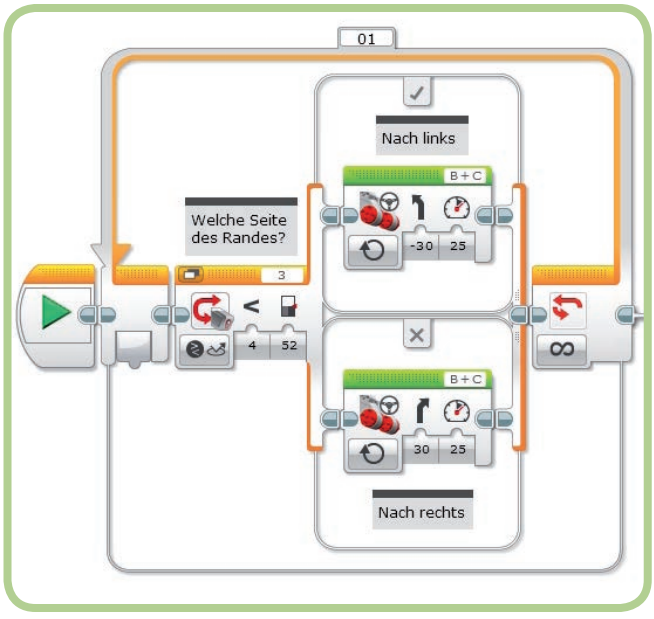


Abbildung 6-6: Das Programm LineFollower

Jetzt musst du nur noch die Einstellungen der einzelnen Blöcke vornehmen. Wie du die Werte für diese Blöcke festlegst, sehen wir uns im Folgenden an.

Einen Schwellenwert für den Farbsensor wählen

Wie bestimmen wir einen geeigneten Schwellenwert für den Schalterblock? Damit der TriBot dem Rand der Linie folgt, müssen wir den Wert herausfinden, den der Farbsensor misst, wenn er sich über dem Rand befindet. Abbildung 6-7 zeigt die Farbensensorwerte, die ich bei fünf verschiedenen Positionen des TriBots erhalten habe, von ganz außerhalb der Linie bis ganz darüber. Eine einfache, aber zuverlässige Möglichkeit, um einen geeigneten Schwellenwert zu bestimmen, besteht darin, den Durchschnitt des höchsten Werts (ganz außerhalb der Linie) und des niedrigsten (ganz darüber) zu nehmen. Mit den Zahlen aus Abbildung 6-7 ergab sich bei mir ein Schwellenwert von (92 + 13) / 2, was gerundet 52 ergibt. Das liegt auch sehr nah an dem Wert, den ich bekomme, wenn ich den TriBot über dem Rand der Linie platziere. Je nach Sensor, Testuntergrund und Beleuchtung können sich bei dir andere Werte ergeben.

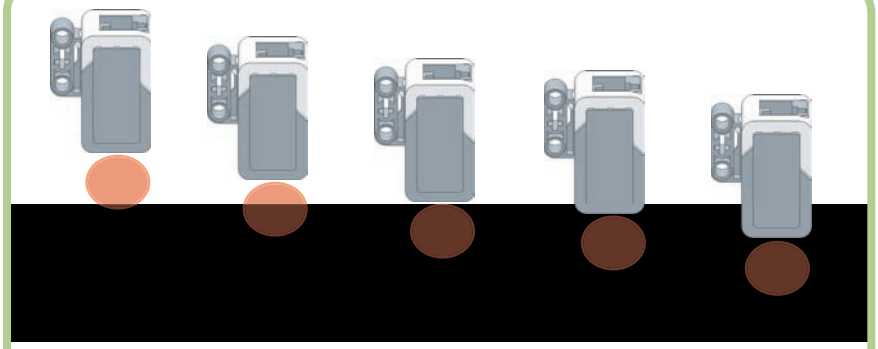


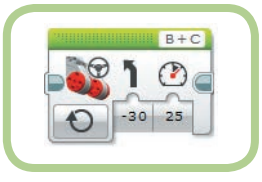
Abbildung 6-7: Farbsensor-Messwerte in verschiedenen Stellungen

7. Setze den Schwellenwert für den Schalterblock. Der Block sollte jetzt wie folgt aussehen:

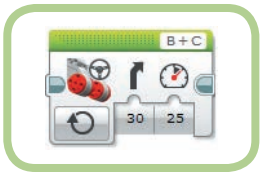
Die Bewegungslenkungsblöcke einrichten

Die beiden Bewegungslenkungsblöcke weisen die gleichen Einstellungen auf, nur die Lenkrichtung ist genau entgegengesetzt. Die Drehzahl der Motoren und der Lenkungswert haben erheblichen Einfluss darauf, wie gut der TriBot der Linie folgt. Bei einem zu niedrigen Lenkungswert dreht sich der TriBot nicht schnell genug, um einer Kurve in der Linie folgen zu können. Ist der Lenkungsparameter dagegen zu hoch eingestellt, schwankt der Roboter ständig von einer Seite zur anderen, da er von einem Extrem ins andere fällt. Auch bei einer zu hohen Geschwindigkeit wird es für den Roboter schwerer, auf Änderungen in der Linienführung zu reagieren. Beginne mit einem absoluten Lenkungswert von 30 für die beiden Richtungen (also 30 und -30) und einem Leistungsparameter von 25:

8. Markiere den Bewegungslenkungsblock im oberen Fall.
9. Setze den Modus auf **An**.
10. Setze den Parameter *Leistung* auf **25** und den Parameter *Lenkung* auf **–30**. Der Block sieht damit wie folgt aus:



11. Markiere den Bewegungslenkungsblock im unteren Fall und setze seinen Modus auf **An**.
12. Setze den Parameter *Leistung* auf **25** und den Parameter *Lenkung* auf **30**. Der Block sieht damit wie folgt aus:



Das Programm testen

Lade das Programm herunter und führe es aus, um zu prüfen, wie gut es funktioniert. Möglicherweise musst du noch kleine Änderungen dahingehend vornehmen, wie schnell der TriBot fährt und wie abrupt er sich dreht. Dabei musst du darauf achten, dass du beide Bewegungslenkungsblöcke auf die gleiche Weise änderst. Wenn das Programm funktioniert, kannst du ausprobieren, wie stark du die Geschwindigkeit erhöhen kannst, ohne dass der Roboter die Fähigkeit verliert, der Linie zu folgen.

Mehr als zwei Wahlmöglichkeiten

Mit der ersten Version des Programms *LineFollower* schwankt der TriBot immer ein wenig nach links und rechts, während er einer geraden Linie folgt, da er ständig seine Lenkung anpasst. Die Bewegung wird glatter, wenn du drei Bewegungslenkungsblöcke verwendest, von denen einer nach links und einer nach rechts lenkt und der dritte den Roboter geradeaus fahren lässt.

Im Modus **Farbsensor ▸ Vergleichen ▸ Stärke des reflektierten Lichts** kann der Schalterblock nur zwei Gruppen von Blöcken enthalten, zwischen denen je nach dem Messwert des Farbsensors entschieden wird. Um zwischen drei Optionen wählen zu können, brauchst du zwei Schalterblöcke. Der erste entscheidet, ob der Robo-

ter nach links fahren soll, der zweite ist für die Entscheidung zwischen Geradeausfahrt und dem Abschwenken nach rechts zuständig. Diese Struktur wird in der EV3-Programmierung (und in der Programmierung allgemein) häufig genutzt, um anspruchsvolle Entscheidungen zu treffen.

Als Erstes nimmst du dazu folgende Änderungen am Programm *LineFollower* vor:

13. Platziere im unteren Abschnitt des Schalterblocks, rechts neben dem Bewegungslenkungsblock, einen weiteren Schalterblock.
14. Setze den Modus des neuen Schalterblocks auf **Farbsensor ▸ Vergleichen ▸ Stärke des reflektierten Lichts**.
15. Ziehe den vorhandenen Bewegungslenkungsblock (der den Roboter nach rechts fahren lässt) in den unteren Abschnitt des neuen Schalterblocks.
16. Platziere einen neuen Bewegungslenkungsblock im oberen Abschnitt des neuen Schalterblocks.
17. Setze den Modus des neuen Bewegungslenkungsblocks auf **An** und den Leistungsparameter auf **25**.

    Der Schalterblock muss jetzt wie in Abbildung 6-8 aussehen.

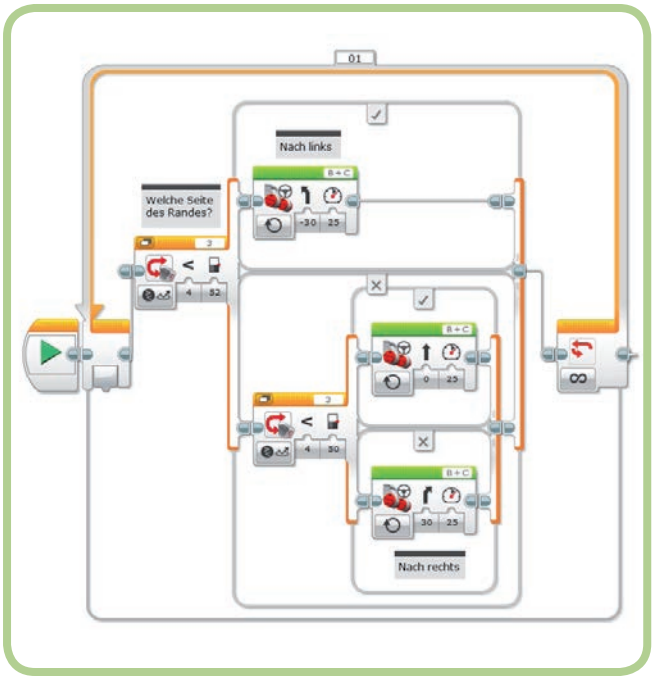


Abbildung 6-8: Die Schalterblock mit den neu hinzugefügten Blöcken

Du musst die Schwellenwerte der beiden Schalterblöcke nun so einstellen, dass der TriBot geradeaus fährt, wenn er sich auf dem Rand der Linie befindet, und sich zur Seite dreht, wenn er sich von diesem Rand entfernt (wenn er sich also entweder dem Inneren der Linie zuwendet oder sie ganz verlässt).

Ursprünglich hatte ich einen einzigen Schwellenwert, nämlich 52, der in der Mitte zwischen den beiden Extremwerten 13 und 92 liegt. Als neue Schwellenwerte können wir diejenigen wählen, die auf halbem Wege zwischen den Extremwerten und dem Mittelwert liegen, also 32 und 72. Ein Vergleich mit Abbildung 6-6 zeigt, dass diese Werte tatsächlich sinnvoll sind, da sie sehr nah an den Messwerten liegen, die ich bekommen habe, als sich der Sensor zum größten Teil außerhalb bzw. zum größten Teil über der Linie befand. Daher richte ich die Schwellenwerte so ein, dass der Roboter in gerader Linie vorwärts fährt, wenn der Messwert zwischen 32 und 72 liegt, und sich außerhalb dieses Bereichs nach links bzw. rechts wendet. Tabelle 6-1 zeigt, wie sich das Programm je nach den Werten des Farbsensors verhalten soll. Da die Messergebnisse von der Beleuchtung, dem Untergrund und dem Material für die Linie abhängen, musst du für dein Programm deine eigenen Werte verwenden.

Tabelle 6-1: Programmverhalten in Abhängigkeit von den Farbsensorbereichen

Messwert des Farbsensors	Programmverhalten
0–31	Steuert nach links
32–72	Fährt geradeaus
73–100	Steuert nach rechts

Vervollständige das Programm nun wie folgt:

18. Markiere den äußeren Schalterblock und setze den Schwellenwert auf die untere Grenze des Bereichs, in dem der Roboter geradeaus fahren soll (in meinem Fall 32).
19. Markiere den inneren Schalterblock und setze den Schwellenwert auf die untere Grenze des Bereichs, in dem der Roboter nach rechts fahren soll (in meinem Fall 73).

Das Programm sieht jetzt aus wie in Abbildung 6-9.

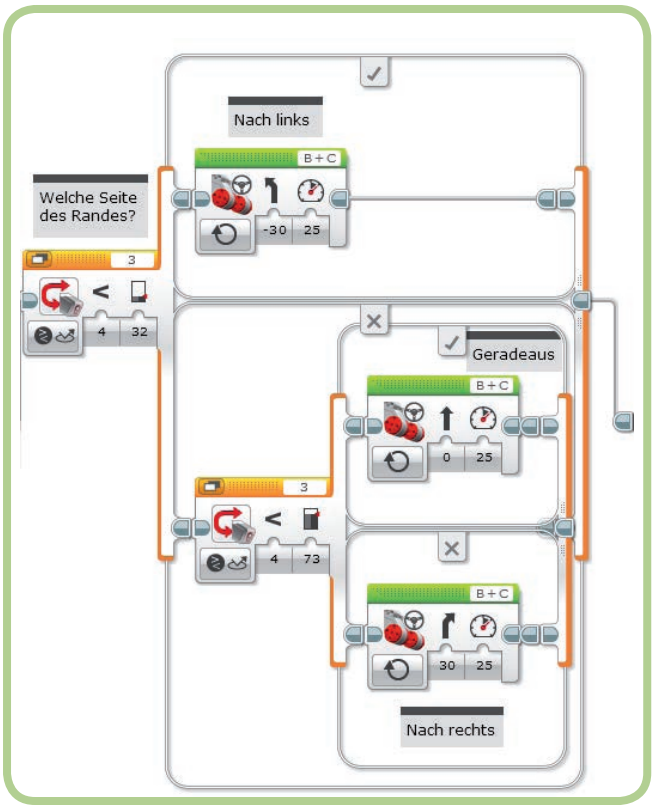


Abbildung 6-9: Die Schwellenwerte festlegen

HINWEIS In diesem Programm lesen beide Schalterblöcke den Farbsensorwert, um ihre Entscheidungen zu treffen. Das bedeutet, dass der Sensor zweimal abgelesen wird, wobei die beiden Werte nicht unbedingt exakt gleich sind. Das sollte jedoch kein Problem darstellen, da sich der Sensorwert in der Zeit, die der EV3-Stein zum Ausführen der beiden Blöcke benötigt, nicht allzu sehr geändert haben sollte.

Das Programm testen

Wenn du das Programm jetzt ausführst, wirst du feststellen, dass sich der Roboter viel geschmeidiger bewegt, während er der Linie folgt. Probiere verschiedene Schwellenwerte, Leistungs- und Lenkungsparameter aus, um zu sehen, wie schnell du den TriBot machen kannst, ohne dass er von der Linie abkommt und sich verirrt.

Die Registeransicht

Das Programm *LineFollower* verwendet *verschachtelte* Schalterblöcke, bei denen sich ein Schalterblock in einem anderen befindet. Das verschlingt viel Platz auf dem Bildschirm, weshalb es schwierig werden kann, mit dem Rest des Programms zu arbeiten. Um die Schalterblöcke zu verkleinern, kannst du sie in die Registeransicht umschalten (klicke dazu auf die Ansichtsauswahl). Dadurch nimmt das Programm auf dem Bildschirm viel weniger Platz ein, wie du in Abbildung 6-10 sehen kannst.

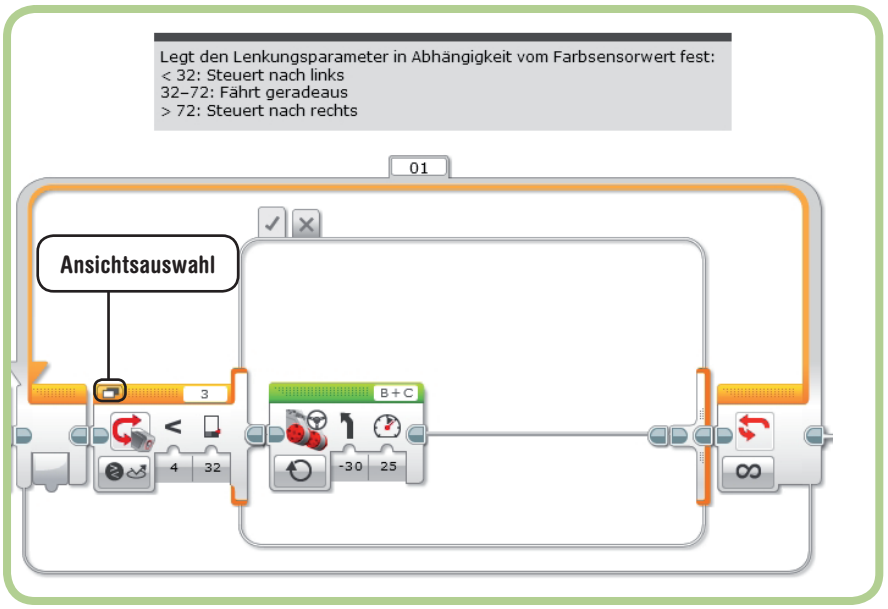


Abbildung 6-10: Der Schalterblock in der Registeransicht

HINWEIS **Wenn du einen Schalterblock in der Registeransicht darstellst, solltest du alle Kommentare über den äußersten Schalterblock stellen. Da immer nur ein Register auf einmal angezeigt wird, können Kommentare innerhalb des Schalterblocks dadurch nicht mehr sichtbar sein. Abbildung 6-10 zeigt einen Kommentar für unser Spurfolgeprogramm.**

Aufgabe 6-1

In der ersten Version von *LineFollower* wurde bei jedem Schleifendurchlauf eine von zwei möglichen Aktionen ausgewählt, nämlich eine Drehung nach links oder nach rechts. Die zweite Version lässt den Roboter eleganter fahren, da sie eine dritte Möglichkeit bietet, nämlich die Geradeausfahrt. Verbessere das Programm noch weiter, indem du noch zwei Fälle hinzufügst, sodass es insgesamt fünf Wahlmöglichkeiten gibt: scharfes Wenden nach links, sanftes Wenden nach links, Geradeausfahrt, sanftes Wenden nach rechts und scharfes Wenden nach rechts. Dazu kannst du die Bewegungslenkungsblöcke, die den Roboter nach links und rechts fahren lassen, durch Schalterblöcke ersetzen, die je nach dem Messwert des Farbsensors zwischen einer scharfen und einer sanften Drehung entscheiden.

Das Programm RedOrBlue

In diesem Abschnitt schreiben wir das Programm *RedOrBlue*, das erkennen kann, ob ein Objekt rot oder blau ist. Als Ausgangspunkt dient uns dazu das Programm *IsItBlue* aus Kapitel 5 (siehe Abbildung 6-11), da es bereits in der Lage ist, blaue Objekte zu erkennen. Als Erstes ändern wir das Programm so, dass es auch rote Objekte identifizieren kann, und dann sorgen wir dafür, dass es irgendetwas Sinnvolles tut, wenn das Objekt weder rot noch blau ist.

Das Programm *IsItBlue* ist im Projekt *Chapter5* gespeichert. Als Erstes müssen wir daher das Programm in das Projekt *Chapter6* kopieren und umbenennen.

1. Öffne das Projekt *Chapter6*, falls es noch nicht geöffnet sein sollte.
2. Öffne das Projekt *Chapter5*.
3. Öffne die Eigenschaftenseite des Projekts *Chapter5*, indem du auf das kleine Schraubenschlüsselsymbol links neben den Programmregistern klickst.
4. Wähle in der Liste der Programme *IsItBlue* aus.
5. Klicke am unteren Rand des Fensters auf **Kopieren**.
6. Wechsele zum Projekt *Chapter6*.
7. Öffne die Eigenschaftenseite des Projekts *Chapter6*.

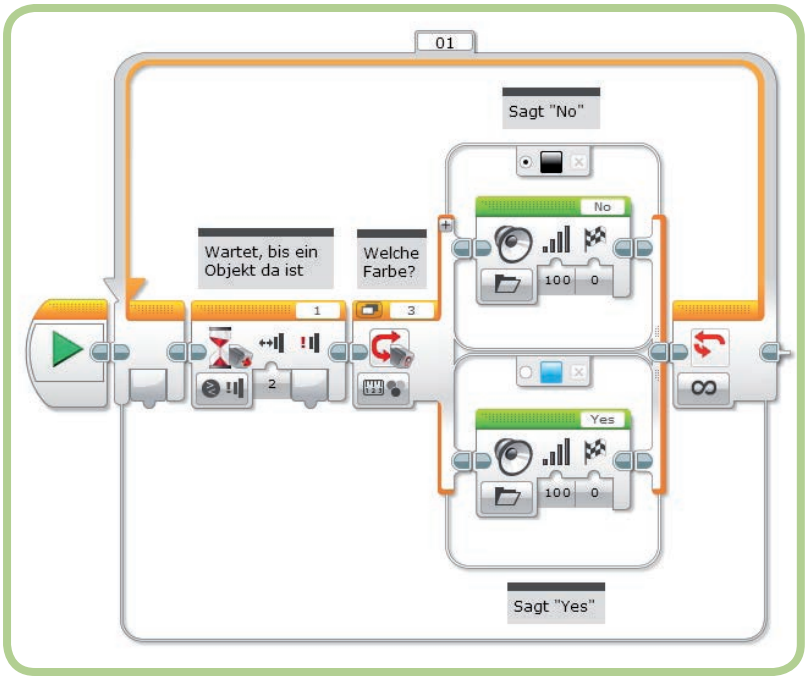


Abbildung 6-11: Der Ausgangspunkt für das Programm RedOrBlue

8. Klicke auf **Einfügen**. Das Programm *IsItBlue* wird dem Projekt hinzugefügt.
9. Öffne das Programm *IsItBlue* und benenne es in *RedOrBlue* um.
10. Schließe das Projekt *Chapter5*.

Rote Objekte erkennen

Der untere Fall des Schalterblocks gilt bereits für blaue Objekte. Daher verwenden wir den oberen Fall für rote Objekte.

1. Klicke auf das schwarze Feld am oberen Rand des Schalterblocks und wähle die Farbe **Rot** aus (siehe Abbildung 6-12).

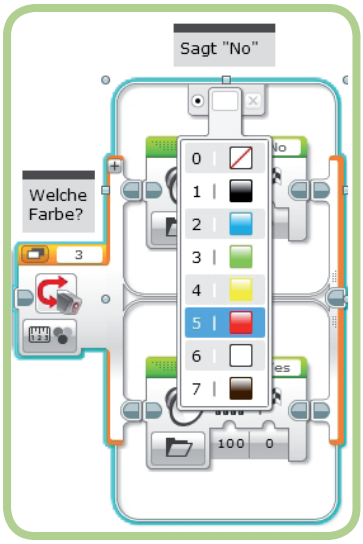


Abbildung 6-12: Die Farbe Rot für den oberen Fall auswählen

Die Antworten »Yes« und »No« waren für das Programm *IsItBlue* sinnvoll, sie sind aber nicht für ein Programm geeignet, das mehr als eine Farbe erkennen kann. Ändere die Klangblöcke so, dass das Programm bei roten Objekten »Red« sagt und bei blauen »Blue«.

2. Markiere den Klangblock im oberen Fall und ändere die Klangdatei in *Red*.

3. Markiere den Klangblock im unteren Fall und ändere die Klangdatei in *Blue*.

Das Programm sieht jetzt so aus wie in Abbildung 6-13. Wenn du es ausführst, identifiziert es blaue und rote Objekte.

Einen neuen Fall hinzufügen

Im jetzigen Zustand kann das Programm nur blaue Objekte korrekt identifizieren, denn es deklariert alle nicht blauen Objekte als rot (da »Rot« als Standardfall gekennzeichnet ist). In diesem Abschnitt ändern wir das Programm so, dass es, wenn es keine Farbe erkennen kann, »Uh-oh« sagt. Der Schalterblock enthält zurzeit zwei Fälle, nämlich einen für rote und einen für blaue Objekte. Über die Schaltfläche *Fall hinzufügen*, die in Abbildung 6-14 eingekreist ist, kannst du den Block um einen weiteren Fall ergänzen. Hast du einen Fall hinzugefügt und willst ihn später wieder entfernen, klickst du auf die Schaltfläche mit dem kleinen x auf der rechten Seite des Registers am oberen Rand des Falls.

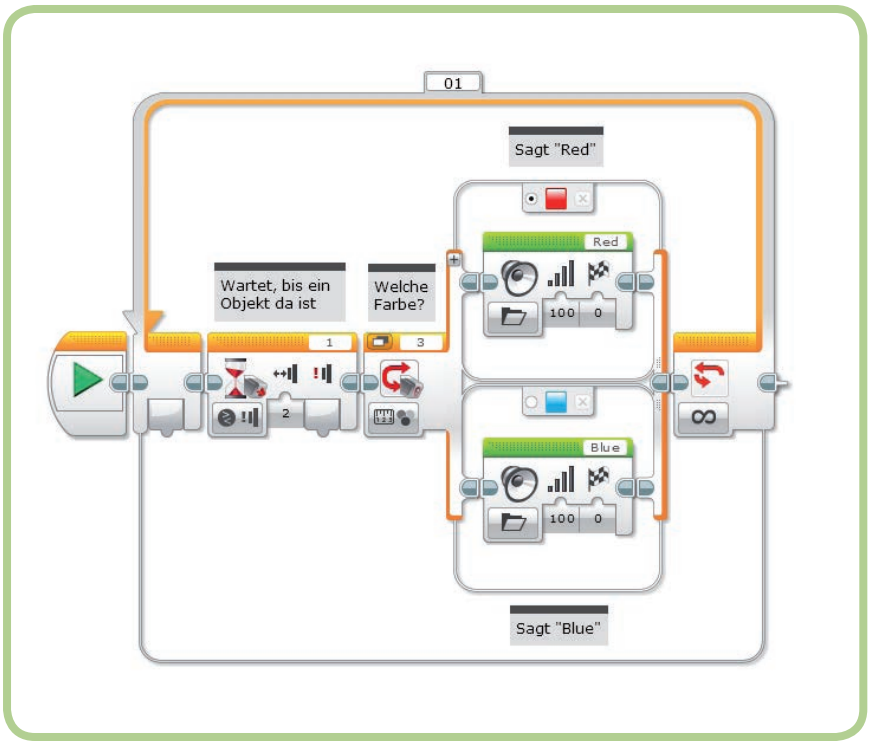


Abbildung 6-13: Rote und blaue Objekte identifizieren

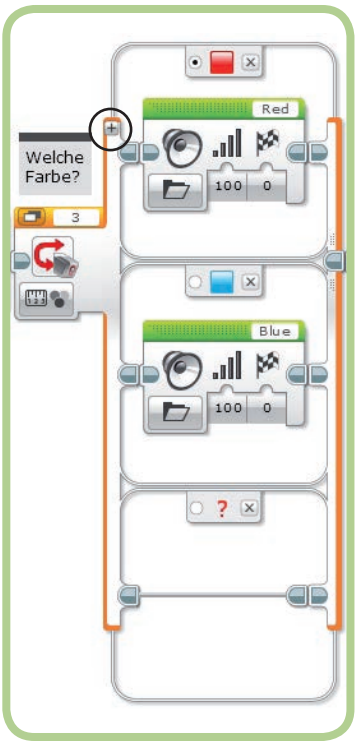


Abbildung 6-14: Einen neuen Fall hinzufügen

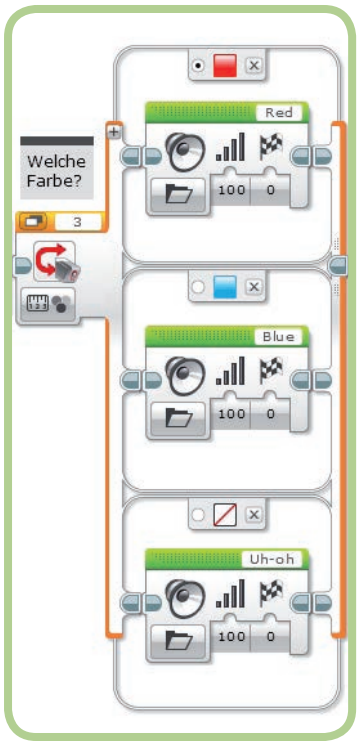


Abbildung 6-15: Der Fall Keine Farbe

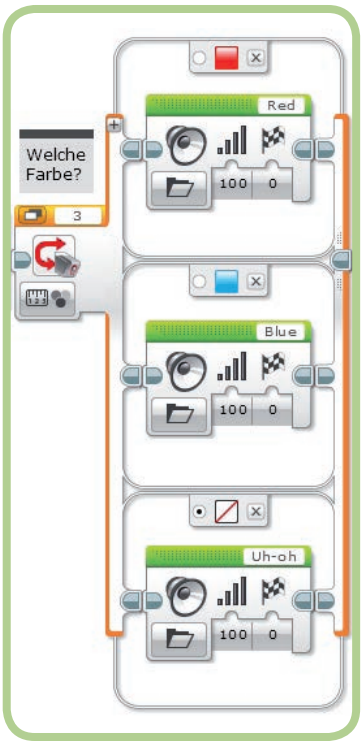


Abbildung 6-16: Keine Farbe als Standardfall

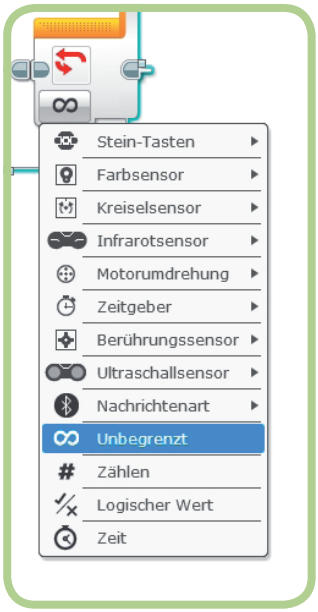


Abbildung 6-17: Den Modus eines Schleifenblocks auswählen

HINWEIS Die Schaltfläche *Fall hinzufügen* erscheint nur bei Schalterblöcken mit Modi, in denen mehr als zwei Fälle möglich sind.

Zurück zu unserem Programm:

4. Klicke auf *Fall hinzufügen*. Der Schalterblock sieht danach aus wie in Abbildung 6-14.
5. Klicke auf das rote Fragezeichen im Register am oberen Rand des neuen Felds. Wähle in dem daraufhin eingeblendeten Menü die Option **Keine Farbe**.
6. Füge im Fall *Keine Farbe* einen Klangblock hinzu.
7. Klicke auf das Feld für den Dateinamen und wähle die Datei **Uh-oh** aus, die du im Ordner *Ausdrücke* findest. Das Programm sieht jetzt aus wie in Abbildung 6-15.

Der Standardfall

»Rot« ist immer noch der Standardfall des Programms. Das bedeutet, dass das Programm die Blöcke dieses Falls ausführt, wenn keiner der angegebenen Fälle zutrifft. Es sagt also auch dann »Red«, wenn der Farbsensor irgendetwas anderes als Rot, Blau oder eine unbekannte Farbe erkennt (z. B. Gelb oder Grün). Es ist viel sinnvoller, *Keine Farbe* als Standardfall zu verwenden, damit das Programm immer dann »Uh-oh« sagt, wenn das Objekt eine andere Farbe hat als Rot oder Blau.

8. Klicke auf den Optionsschalter **Standardfall** des Falls *Keine Farbe*.

Abbildung 6-16 zeigt die endgültige Einrichtung des Schalterblocks.

Aufgabe 6-2

Erweitere das Programm *RedOrBlue* so, dass es alle sieben Farben erkennt, die der Farbsensor wahrnehmen kann.

Der Schleifenblock

Mit dem *Schleifenblock* kannst du eine Gruppe von Blöcken ständig wiederholen lassen. Die Blöcke innerhalb des Schleifenblocks werden als *Schleifenrumpf* bezeichnet. Mit einer Bedingung legst du fest, wie oft der Rumpf ausgeführt wird und wann das Programm zu dem Block übergeht, der auf die Schleife folgt.

Wie du in Abbildung 6-17 siehst, weist ein Schleifenblock die gleichen Modi auf wie ein Schalterblock, darüber hinaus aber noch vier weitere:

Abbildung 6-18: Eine fehlerhafte Version des Programms LineFinder

- **Unbegrenzt:** Die Schleife wird wiederholt, bis das Programm endet oder bis die Schleife durch einen Schleifen-Interrupt-Block abgebrochen wird (mehr darüber im folgenden Abschnitt).
- **Zählen:** Der Block wird so oft ausgeführt wie angegeben.
- **Logischer Wert:** Ein Wert, der der Schleife über eine Datenleitung übergeben wird, bestimmt, ob die Schleife beendet werden soll. (Schleifenblöcke mit Datenleitungen werden in Kapitel 10 beschrieben.)
- **Zeit:** Die Schleife wird so viele Sekunden lang ausgeführt wie angegeben.

Bei einem Schleifenblock in einem Sensormodus prüft das Programm den Sensormesswert, nachdem der Rumpf ausgeführt worden ist. Das hat zur Folge, dass der Rumpf auf jeden Fall mindestens einmal ausgeführt wird. Danach entscheidet das Programm auf der Grundlage des Sensorwerts, ob es einen weiteren Durchlauf vornehmen oder die Schleife verlassen soll. Welchen Wert der Sensor misst, während der Schleifenrumpf läuft, spielt überhaupt keine Rolle. Einzig und allein der Messwert am Ende der Schleife wirkt sich auf den Programmablauf aus. Wenn du nicht aufpasst, kann das dazu führen, dass dein Programm fehlschlägt. Als Beispiel dafür siehst du in Abbildung 6-18 eine fehlerhafte Version des Programms *LineFinder*, bei der es häufig vorkommt, dass die Linie nicht gefunden wird. Da der Farbsensor nur am Ende der Schleife geprüft wird, ist es sehr wahrscheinlich, dass der Roboter sie überfährt, während der Klang- oder der Warteblock ausgeführt wird, und er sie daher nicht wahrnimmt.

Der Schleifen-Interrupt-Block

Der *Schleifen-Interrupt-Block* aus Abbildung 6-19 gibt dir eine weitere Möglichkeit, um eine Schleife zu verlassen. Er hat nur einen einzigen Parameter, nämlich den Namen der Schleife, die abgebrochen werden soll. Oben an jedem Schleifenblock befindet sich ein kleines Feld, das den Namen des Blocks angibt (siehe Abbildung 6-20). In der Voreinstellung heißen die Schleifenblöcke *01*. Um den Namen zu ändern, klickst du auf das Feld.

Abbildung 6-19: Der Schleifen-Interrupt-Block

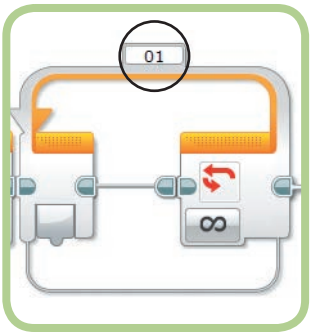


Abbildung 6-20: Der Name eines Schleifenblocks

Im Schleifen-Interrupt-Block werden nur die ersten drei Buchstaben des Namens der Schleife angezeigt, die unterbrochen werden soll. Daher ist es gewöhnlich eindeutiger, Zahlen oder Abkürzungen als Schleifennamen zu verwenden, da es sonst unter Umständen nicht ganz klar sein kann, auf welche Schleife sich der Interrupt-Block bezieht.

Das Programm BumperBot3

Damit du die Funktionsweise des Schleifen-Interrupt-Blocks kennenlernst, nimmt du im Folgenden einige Änderungen am Programm *BumperBot2* vor. Um dieses Programm einsetzen zu können, musst du den TriBot wieder in die ursprüngliche Gestalt zurückverwandeln, also mit der Stoßstange und dem Berührungssensor vorn und dem Farbsensor an der Seite (siehe Abbildung 6-21).

Abbildung 6-21: Aufbau des TriBots für das Programm BumperBot3

Durch die Änderungen soll die Schleife abgebrochen und das Programm beendet werden, wenn du das Licht im Zimmer ausschaltest. Abbildung 6-22 zeigt den Teil des ursprünglichen Programms, den wir ändern werden. Im jetzigen Zustand lässt das Programm den Roboter vorwärts fahren und wartet darauf, dass der Berührungssensor gedrückt wird. Wenn das geschieht, setzt der Roboter zurück, und das Programm beginnt wieder von vorn.

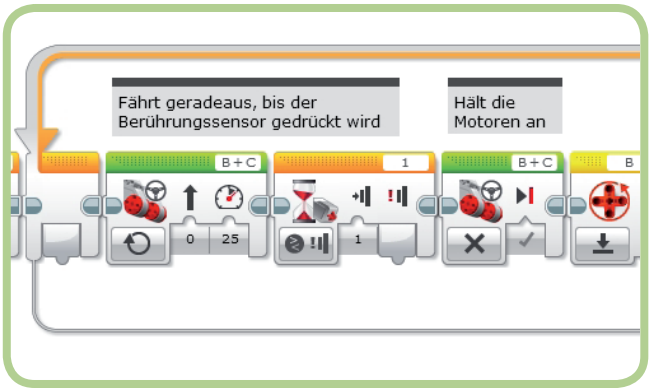


Abbildung 6-22: Das Programm BumperBot2

Nun wirst du das Programm so ändern, dass der Roboter den Farbsensor überprüft, während er wartet. Liefert der Farbsensor einen sehr niedrigen Wert, ist das ein Zeichen dafür, dass das Licht ausgeschaltet ist. In diesem Fall unterbricht der Interrupt-Block die Schleife. Da sich hinter der Schleife keine weiteren Blöcke mehr befinden, wird dadurch auch das Programm beendet.

Anstatt mit einem Warteblock zu bestimmen, ob die Stoßstange berührt wurde, verwenden wir einen Schleifenblock, der wiederholt wird, bis der Berührungssensor gedrückt wird (siehe Abbildung 6-23). Im Rumpf der Schleife prüfen wir mit einem Schalterblock im Modus **Stärke des Umgebungslichts**, wie hell der Raum beleuchtet ist. Nimmt der Farbsensor nicht mehr genügend Licht wahr, hält das Programm die Motoren an, sagt »Goodbye« und verlässt die Schleife.

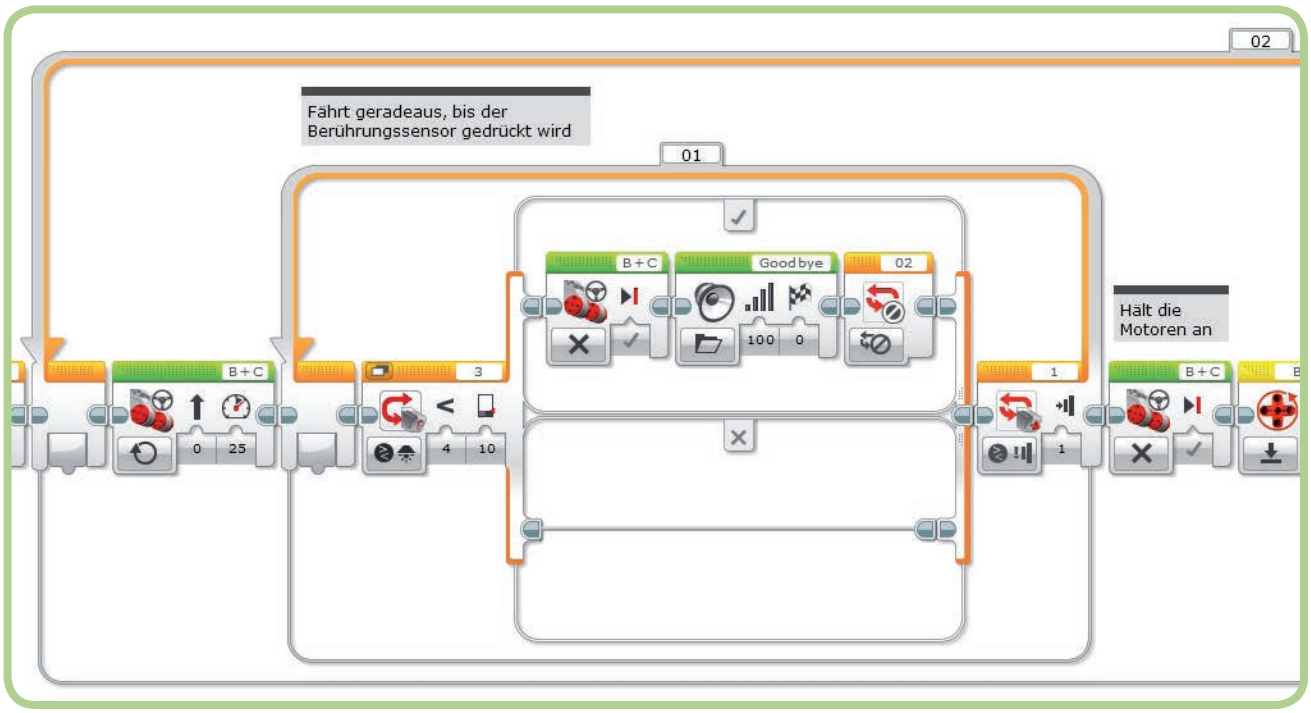


Abbildung 6-23: Das Programm BumperBot3

Das Programm *BumperBot3* aus Abbildung 6-23 erstellst du wie folgt:

1. Kopiere das Programm *BumperBot2* aus dem Projekt *Chapter5* in das Projekt *Chapter6*.
2. Ändere den Namen des Programms in *BumperBot3*.
3. Ändere den Namen des Hauptschleifenblocks in *02*.
4. Lösche den Warteblock.
5. Füge einen Schleifenblock an der Stelle ein, an der sich der Warteblock befand.
6. Setze den Modus des Schleifenblocks auf **Berührungssensor ▸ Vergleichen ▸ Zustand**.

Dieser Teil des Programms muss jetzt so aussehen wie in Abbildung 6-24.

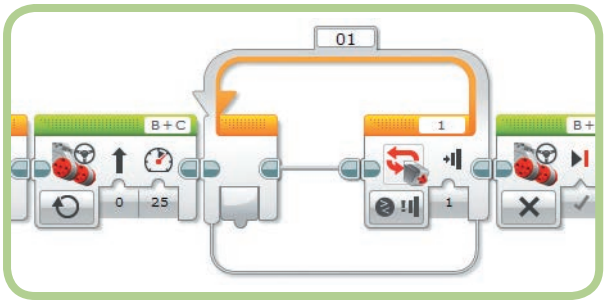


Abbildung 6-24: Den Warteblock durch eine Schleife ersetzen

Der leere Schleifenblock macht jetzt das Gleiche wie der Warteblock, den er ersetzt: Er wartet einfach darauf, dass der Berührungssensor gedrückt wird. Innerhalb der Schleife können wir jetzt aber Blöcke hinzufügen, die ausgeführt werden, während das Programm wartet.

In den nächsten Schritten fügst du einen Schalterblock in die Schleife ein, um den Farbsensor abzufragen:

7. Ziehe einen Schalterblock in den Schleifenblock.
8. Setze den Modus des Schalterblocks auf **Farbsensor ▸ Vergleichen ▸ Stärke des Umgebungslichts**.
9. Setze den Schwellenwert auf **10**. Wenn dieser Wert zu hoch oder zu niedrig sein sollte, kannst du einige Tests ausführen und ihn entsprechend anpassen. Die Schleife sieht jetzt aus wie in Abbildung 6-25.

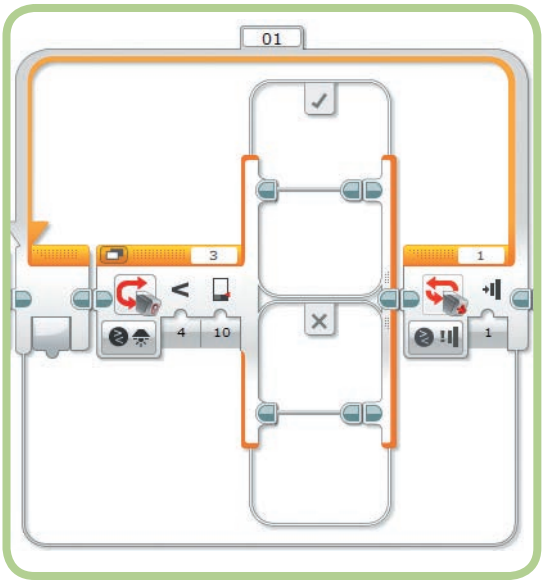


Abbildung 6-25: Den Schalterblock hinzufügen

Wenn der Schalterblock ausgeführt wird, prüft das Programm die Werte des Farbsensors, und wenn die Lichtintensität unter 10 fällt, werden die Blöcke im oberen Fall ausgeführt. Füge dort nun die folgenden Blöcke hinzu:

10. Ziehe einen Bewegungslenkungsblock in den oberen Fall des Schalterblocks und setze den Modus auf **Aus**.
11. Füge hinter dem Bewegungslenkungsblock einen Klangblock ein. Wähle aus der Liste der Klangdateien **Goodbye** aus (unter **LEGO-Klangdateien ▸ Kommunikation**).
12. Füge hinter dem Klangblock einen Schleifen-Interrupt-Block ein und setze den Schleifennamen auf **02**.

Abbildung 6-26 zeigt den Schleifenblock mit diesen Änderungen.

Wenn du das Programm jetzt ausführst, sollte der Roboter »Goodbye« sagen und anhalten, sobald du das Licht löschst. Zum Testen kannst du den TriBot auch einfach in einer Hand halten und den Farbsensor mit der anderen Hand zudecken, nachdem die Räder angefangen haben, sich zu drehen. Dadurch wird die Lichtstärke für den Farbsensor zu gering, weshalb das Programm »Goodbye« sagen und enden sollte.

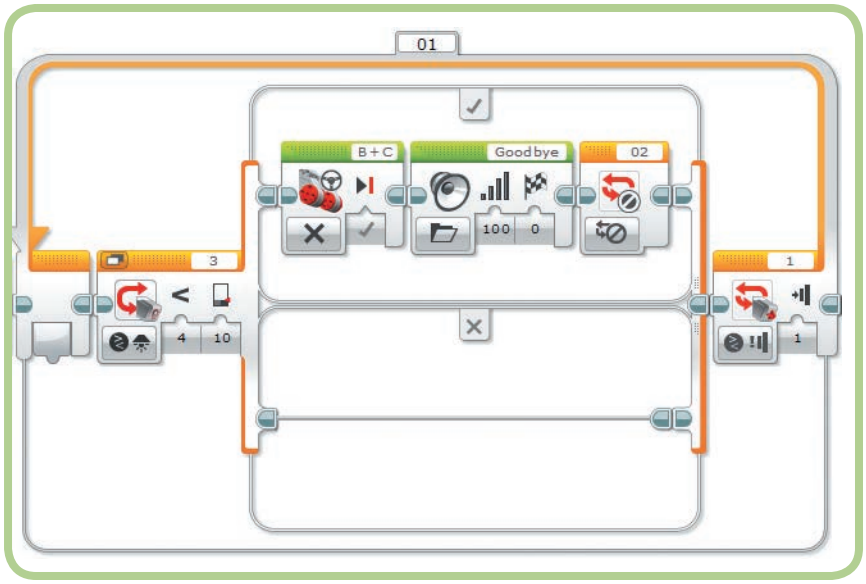


Abbildung 6-26: Die Motoren anhalten und »Goodbye« sagen

Noch mehr zum Ausprobieren

Nachdem du die Feinheiten des Schalter- und des Schleifenblocks kennengelernt hast, kannst du den Umgang damit nun mit den folgenden Aufgaben weiter üben:

1. Schreibe ein Programm, damit der Roboter dir folgt. Er soll nahe bei dir bleiben, aber dir nicht zu nahe kommen, also etwa einen Abstand von dreißig bis sechzig Zentimetern halten. Verwende den Infrarot- oder den Ultraschallsensor und einen Schalterblock, damit der Roboter vorwärts oder rückwärts fährt, wenn du zu nahe gekommen bist oder dich zu weit entfernt hast. Der Roboter sollte sich nicht bewegen, solange du dich in dem gewünschten Entfernungsbereich befindest.

2. Füge zum Programm *BumperBot3* eine Pausen- und Wiederaufnahmefunktion hinzu, die über die Infrarotfernbedienung gesteuert wird. Ergänze das Programm dazu um einen weiteren Schalterblock, der prüft, ob eine Taste auf der Fernsteuerung gedrückt wurde. Wenn ja, soll das Programm die Motoren anhalten und dann warten, bis eine weitere Taste betätigt wird. Dann soll es die Motoren wieder starten. Die erste Taste fungiert also als Pausentaste, die zweite als Wiederaufnahmetaste.

Zusammenfassung

In diesem Kapitel hast du gelernt, wie du den Schalterblock verwendest, um Entscheidungen zu treffen. Damit kannst du zwischen zwei und mehr Gruppen von Blöcken auswählen. Die Anzahl der Wahlmöglichkeiten kannst du erhöhen, indem du mehrere Schalterblöcke verschachtelst oder indem du einem Schalterblock weitere Fälle hinzufügst.

Der andere wichtige Block zur Ablaufsteuerung ist der Schleifenblock, mit dem du eine Gruppe von Blöcken wiederholt ausführen lassen kannst, bis eine vorgegebene Bedingung erfüllt ist. Bei dieser Bedingung geht es gewöhnlich darum, dass ein Sensormesswert einen bestimmten Schwellenwert erreicht, aber du kannst einen Schleifenblock auch so einrichten, dass er eine bestimmte Anzahl von Wiederholungen ausführt oder für einen bestimmten Zeitraum läuft. Die große Vielseitigkeit von EV3-Programmen ist vor allem diesen vielen Wahlmöglichkeiten bei der Konfiguration von Blöcken zu verdanken. Der Schleifen-Interrupt-Block bietet noch eine weitere Möglichkeit, um eine Schleife zu verlassen, und gibt dir daher eine noch größere Flexibilität, um den Ablauf deiner Programme zu beeinflussen.

Du hast jetzt gelernt, wie du die EV3-Motoren und die Sensoren verwendest, und kennst die Programmierblöcke, die du brauchst, um anspruchsvolle Programme zu schreiben. Im nächsten Kapitel setzt du dein Wissen ein, um den TriBot so zu programmieren, dass er selbstständig aus einem Labyrinth herausfindet.