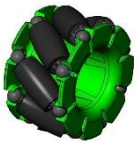
Omniwheels mit Spur- und Gestensensor

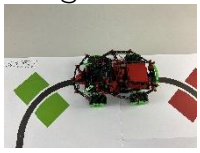
Mit diesem Modell erarbeitest du dir weitere Grundlagen der Programmierung. Verstärkt wird hier der Einsatz des Gestensensors (Farberkennung) und das Zusammenspiel mit dem Spursensor erarbeitet. Neu hinzu kommt das Steuern des Modells über ein selbst erstelltes Bedienfeld.

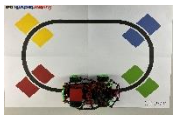
Im Modell verbaute Aktoren und technisches Zubehör.

Mini Motor	Getriebe	Gestensensor	Omniwheelrad	IR-Spursensor
				

Die Erklärung zu den Bauteilen findest du auf der Startseite.

Das Modell „Omniwheels mit Spur- und Gestensensor“ gliedert sich in 5 Programmieraufgaben:

Aufgabe 1 Omniwheels_track_gesture_sensor_1.ft	Programmierlevel 3 Das Modell fährt entlang der schwarzen Linie auf dem Parkour. Über den IR-Spursensor wird bei Verlassen der Linie die Fahrtrichtungen nachgeregelt, so dass das Modell immer auf der schwarzen Führungslinie fährt.	Hyperlink zur Aufgabe 1 
Aufgabe 2 Omniwheels_track_gesture_sensor_2.ft	Programmierlevel 3 Das Modell soll zuerst die schwarze Führungslinie suchen und dann wie in Aufgabe 1 dieser folgen.	Hyperlink zur Aufgabe 2

Aufgabe 3 Omniwheels_track_ gesture_sensor_3.ft	Programmierlevel 3 Das Modell soll über die Gestensteuerung bewegt werden. Dazu müssen die Handbewegungen ausgewertet werden.	Hyperlink zur Aufgabe 3
Aufgabe 4 Omniwheels_track_ gesture_sensor_4.ft	Das Modell folgt wie in den vorherigen Aufgaben der schwarzen Leitlinie. Trifft es auf ein Farbfeld wird die Farbe ermittelt und es wird dadurch eine Aktion ausgeführt z.B. blau - einparken	Hyperlink  zur Aufgabe 4

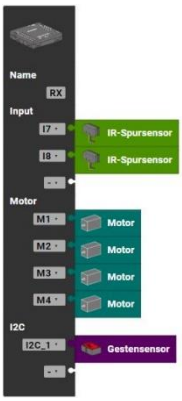
Aufgabe 1

Die erste Aufgabenstellung kennst du schon vom Modell „Advanced 2-Radroboter“ her. Der einzige Unterschied ist, dass du jetzt mit vier Motoren arbeitest.

Festlegung der Räder: M1 = Rad vorne links, M2 = Rad vorne rechts, M3 = Rad hinten links und M4 = Rad hinten rechts.

Starte dazu das Programm „Advanced_2_wheeled_roboter_2”.

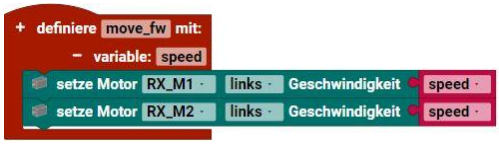
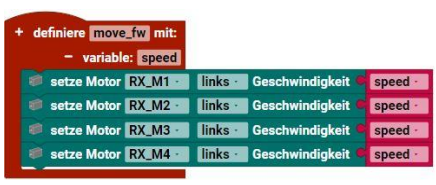
Da du die meisten Funktionen übernehmen kannst, musst du das Programm zuerst unter dem neuen Namen abspeichern bzw. benennen: Omniwheels_track_ gesture_sensor_1

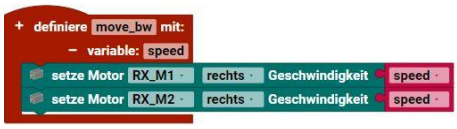
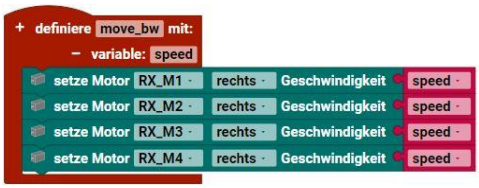
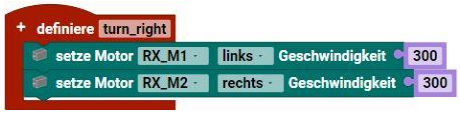
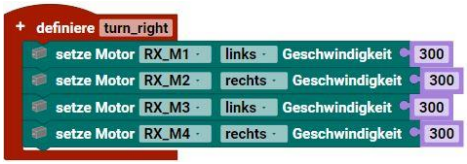
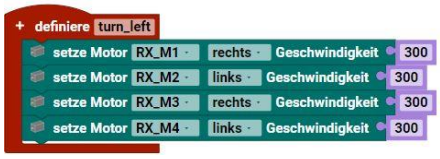
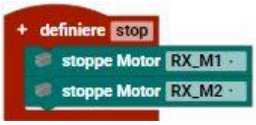
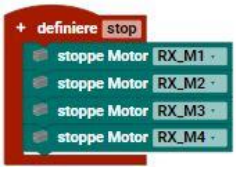


Erweitere die Controllerkonfiguration auf 4 Motoren und füge den Gestensensor an die I2C-Schnittstelle ein.

Im Hauptprogramm musst du anschließend alle Funktionen, die sich auf Motorbewegungen beziehen, mit 2 weiteren Motoren erweitern.

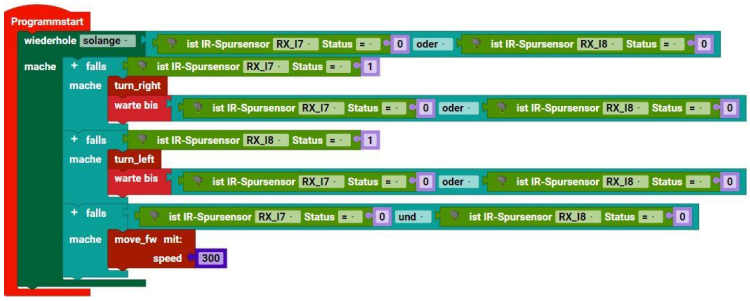
Ändere anhand der folgenden Darstellungen die Funktionen:

2-Rad move_fw	4-Rad move_fw
	

2-Rad move_bw	4-Rad move_bw
	
2-Rad turn_right	4-Rad turn_right
	
2-Rad turn_left	4-Rad turn_left
	
2-Rad stop	4-Rad stop
	

Das Modell soll entlang der schwarzen Linie auf dem Parkour fahren. Über den IR-Spursensor wird bei Verlassen der Linie die Fahrtrichtungen nachgeregelt, so dass das Modell immer auf der schwarzen Führungslinie fährt. Hier kannst du die Funktion „follow_line“ verwenden. Eine Änderung musst du nicht vornehmen.

Die Funktion „find_line“ kannst du löschen, diese wird erst in der Aufgabe 2 benötigt.



In den Programmstart musst du zuerst aus dem Block „follow_line“ die gesamte „wiederhole“-Schleife duplizieren und in „Programmstart“ einfügen. Ändere den „speed“-Wert auf „300“ ab.

Zum Schluss musst du nach der Schleife den Block „stop“ einfügen. Somit ist die Aufgabe 1 fertig definiert.

Teste anschließend das Programm. Hat alles funktioniert, kannst du die 2. Aufgabe programmieren.

Aufgabe 2

Wichtig: Da du Teile der Aufgabe 1 benötigst, speichere diese gleich unter Omniwheels_track_ gesture_sensor_2 auf deinem Rechner ab.

Kopiere zuerst dem Programm „Advanced_2_wheeled_roboter_2” die Funktion

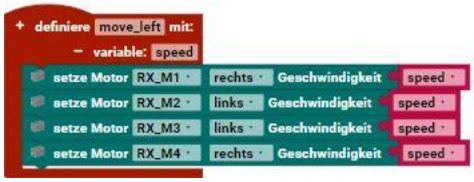
Somit sind alle Funktionen definieret und du kannst diese in deinen Programmstart einbauen.

Wie in der Aufgabenstellung gefordert, soll das Modell zuerst eine Linie suchen und dieser dann entlangfahren, d.h. die Funktion wird vor den Schleifenbefehl gesetzt.

Somit ist das Programm fertig erstellt und du kannst es testen. Speichere es nochmals ab.

Aufgabe 3

Für diese Aufgabe benötigst du neben den schon definierten Funktionen noch die Funktion „move_right“ und „move_left“. Dupliziere einfach die Funktion „move_bw“ zwei mal und ändere die Parameter ab.

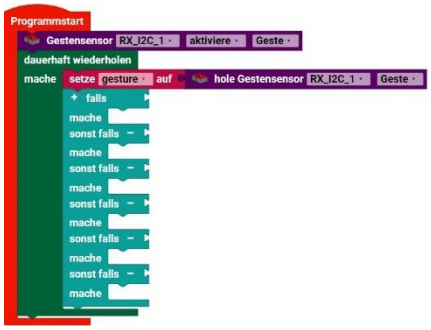
4-Rad move_right	4-Rad move_left
	

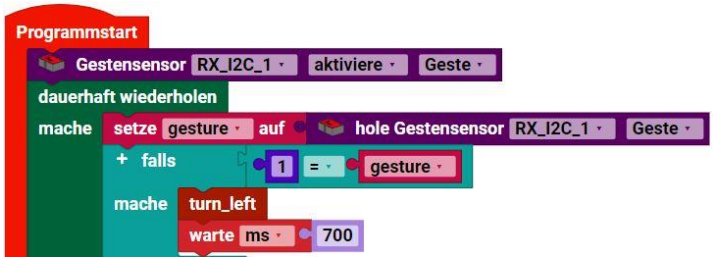
Das Hauptprogramm erstellen wir zusammen. Wie dir sicherlich noch bekannt ist, musst du Schleife „dauerhaft wiederholen …“ einfügen.

Bevor du mit der Werteabfrage des Gestensensor beginnst, musst du die Variable „gesture“ erzeugen. Die Variable „speed“ ist ja schon definiert.

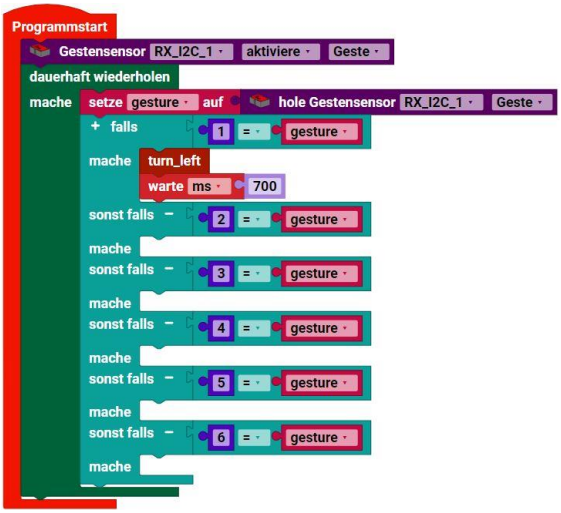
Füge die Befehlsfolge „setze … auf“ und „hole Gestensensor …“ ein.

Insgesamt sollen 6 Bewegungen, die vom Gestensensor ermittelt werden, ausgewertet werden und eine Aktion des Modells auslösen. Dazu fügst du „6“ mal den Befehl „falls mache“ ein.

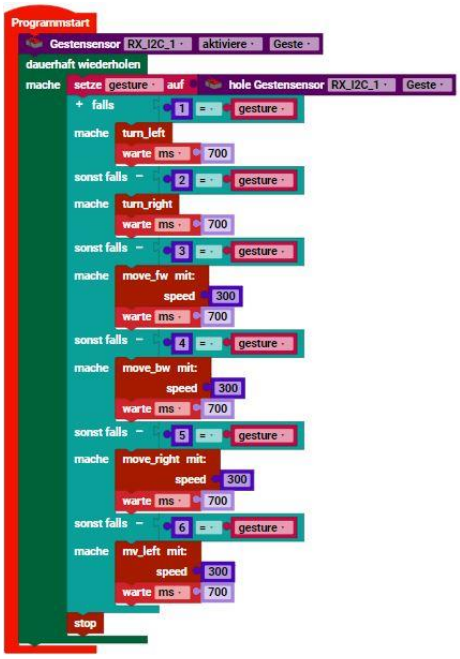




Die erste Abfrage soll ermitteln, ob der Sensorwert „1“ ist. Ist dies der Fall, soll das Modell „700 ms“ nach links drehen. Füge dazu den Befehl „=“ ein. In die erste Andockstelle kommt der Befehl „eine Zahl“ und in die zweite Andockstelle die Variable „gesture“. Anschließend folgt der Befehl „warte …“ ändere hier auf „ms“ und den Wert „700“ um.



Definiere die weiteren Abfragen. Dupliziere die erste Abfrage und füge sie an den jeweiligen Stellen ein. Ändere anschließend die Parameter.

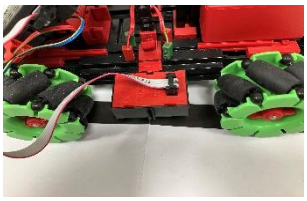


Zum Schluss musst du noch die Leerräume von „mache“ definieren. Dazu kannst du schon einmal den „warte“-Befehl duplizieren und an die einzelnen Stellen einfügen. Für den Wert „2“ fügst du „turn_right“ ein, für „3“ „move_fw, für „4“ „move_bw“, für „5“ „move_right“ und für „6“ „move_left“ ein. Die „speed“-Parameter müssen mit „300“ festgelegt werden. Am Ende der Schleife fügst du noch die Funktion „stop“ ein.

Somit ist das Programm fertig erstellt und du kannst es testen. Speichere das Programm unter Omniwheels_track_ gesture_sensor_3

Aufgabe 4

Jetzt kommst du zur letzten Aufgabe.

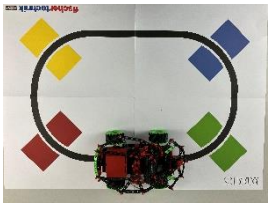


Wichtig: Damit der Gestensensor die Farbflächen erkennen kann, musst du wie in der Bauanleitung gezeigt, den Gestensensor umpositionieren.

Hier geht es darum, dass das Modell wieder anhand der Führungslinie den Parkour abfährt.

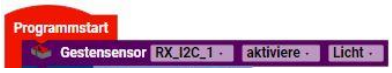
Wird eine farbige Fläche erreicht, soll jeweils ein Ereignis ausgeführt werden.

- Trifft der Sensor auf „Gelb“, fährt das Modell schneller.
- Trifft der Sensor auf „Rot“, führt das Modell Tanzbewegungen auf.
- Trifft der Sensor auf „Blau“, wird die Parkfunktion angefahren.
- Trifft der Sensor auf „Grün“, fährt das Modell langsamer.

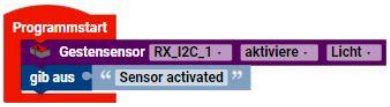


Für dieses Programm wirst du bis auf einige Funktionen zur Bewegung des Modells neue Funktionen und das Hauptprogramm erstellen. Dazu kannst du das vorherige Programm unter dem neuen Namen abspeichern: Omniwheels_track_gesture_sensor_4

Beginne damit, dass du aus dem „Programmstart“ alle Befehle löschen kannst, bis auf den Befehl zur Initialisierung des Gestensensors.

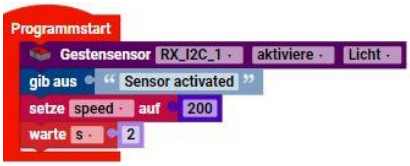


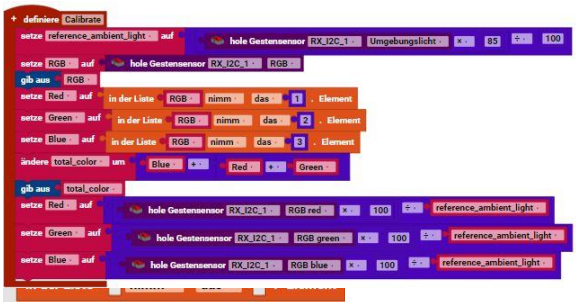
Hier kommt ein neuer Befehl „gib aus“ aus dem Block „Verarbeitung“ – „Text“ hinzu. In den Freiraum kannst du einen in der Konsole darzustellenden Text einbinden, z.B. auf das Programm bezogen “Sensor activated“.



Im Verlauf des Gesamtprogramms wirst du weitere Textausgaben einarbeiten.

Definiere die schon erstellte Variable „speed“ mit dem Wert „200“. Füge anschließend einen „warte“- Befehl von „2“ Sekunden ein.



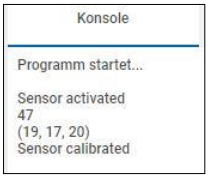


Erzeuge im nächsten Schritt eine Funktion „Calibrate“. Erzeuge, bevor du den Programmteil definierst, folgende Variablen:

„reference_ambient_light“, „RGB“, “Red”, “Green”, “Blue”, “correct_r”, “correct_b”, “current_g”.

Im ersten Befehl wird der Referenzwert des Umgebungslichtes gemessen, den der Sensor liefert. Dazu baust du die Variabe „setze reference_ambient_light“ zuerst ein. Die Variable wird auf den ausgegebenen Wert des Gestensensors für das „Umgebungslicht“ gesetzt. In der Konsole kannst du den Wert ausgeben lassen.

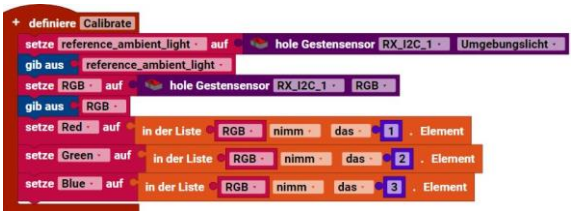
Füge anschließend den Befehl „setze RGB auf“ und in den Freiraum den Befehl „hole Gestensensor … RGB“ ein. Anschließend kannst du dir die ermittelten Werte für die drei Farben in der Konsole ausgeben lassen.

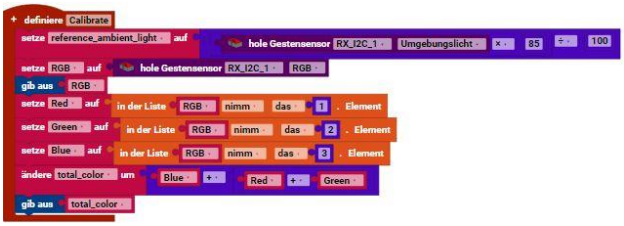


Die nächsten 3 Befehle sind Zuweisungen aus dem Befehl „in der Liste … nimm“ aus dem Block „Verarbeitung“ – „Datenstruktur“. Das erste Element in der Liste (RGB) ist der Wert für „Rot“. Dieser wird der Variablen „Red“ zugewiesen.

Das zweite Element in der Liste ist der Wert für „Grün“. Dieser wird der Variablen „Green“ zugewiesen. Das dritte Element in der Liste ist der Wert für „Blau“. Dieser wird der Variablen „Blue“ zugewiesen.

Füge zuerst dreimal den Befehl „setze … auf“ ein, ändere auf „Red“, „Green“ und „Blue“. Füge anschließend die Variable „RGB“ ein. Zum Schluss folgt noch dreimal der Befehl „Eine Zahl“. Ändere den ersten Wert auf „1“, den zweiten Wert auf „2“ und den dritten Wert auf „3“ ab.



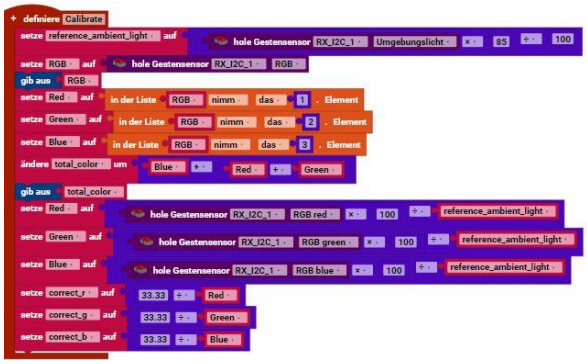


Anschließend addierst du die drei Farbwerte in die Variable „total_color“. Den ermittelten Wert kannst du über „gib aus …“ in der Konsole anzeigen lassen.

Im nächsten Schritt setzt du die Variable „Red“, „Green“ und „Blue“ auf den Wert des Gestensensors für die Farbe „RGB red“, „RGB green“ und „RGB blue“. Diesen Wert multiplizierst du mit „100“ und teilst ihn durch den Wert „reference_ambient _light“.

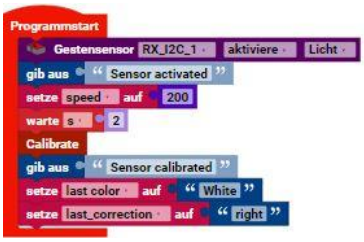
Bei einer perfekten weißen Farbe sollten die drei Werte gleich sein, d. h. der gesamte Lichtwert, 1/3 (oder 33,3 %) jeder Farbe.

Sobald du den Prozentsatz jedes Kanals berechnet hast, teilst du die theoretischen 33,33 % jedes Kanals durch den durchschnittlichen Lichtwert und erhälst so einen Korrekturfaktor für das Licht im Raum. Die Berechnungen werden den Variablen „correct_r“, „correct_g“ und „correct_b“ zugewiesen.



Wenn das Licht überwiegend blau ist, ist der Anteil dieses Kanals höher als 33,3 und der Korrekturwert ist kleiner als eins. Im Gegensatz dazu sind die anderen Kanäle niedriger als 33,3 und ihre Korrekturwerte sind höher als eins. Was ist dabei zu beachten?

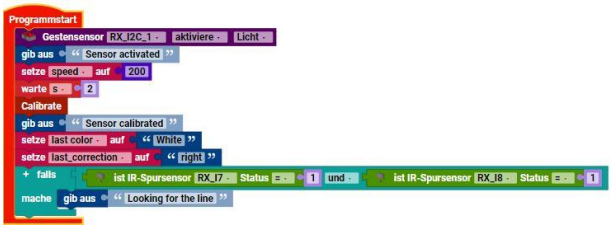
Das Modell startet immer in einem weißen Bereich. Mit dem Korrekturfaktor reduzierst du die Auswirkungen der Lichtfarbe und der Sensormesswerte auf die Farben erheblich.



Die Funktion „Calibrate“ fügst du in den Programmstart ein. In der Konsole kannst du einen Text „Sensor calibrated“ ausgeben lassen.

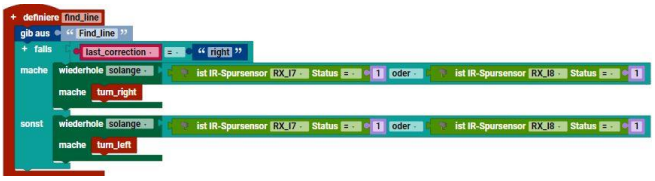
Definiere zwei weitere Variablen „last color“ und „last correction“. Füge beide Variablen nach „Calibrate“ ein und setze sie auf „White“ und „right“.

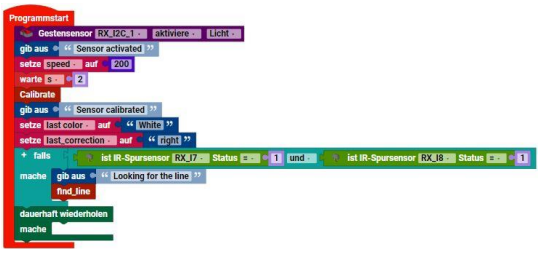
Als Nächstes soll das Modell über den IR-Sensor die schwarze Linie finden. Dazu fügst du zuerst den Befehl „falls mache“ ein. Für „falls“ baust du den Mehrfachbefehl zur Ermittlung der IR-Logik ein. Anschließend gibst du in der Konsole aus, dass die Linie gesucht wird.



Ok, auch das ist geschafft. Jetzt benötigst du als nächsten Programmschritt die Funktion „find_line“. Zum Üben des Programmierens erzeuge einfach die Funktion anhand der Darstellung.

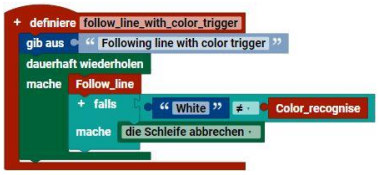
In den Programmstart fügst du die Funktion „find_line“ ein. Startest du das Programm, sucht das Modell die schwarze Linie. Die Funktion wird in der Konsole angezeigt.





Somit ist der Kopf des Programmstarts definiert und du kannst als weiteren Schritt eine Schleife einbauen. Erzeuge die Variable „Color“, “time”, „red_sensor“, „green_sensor“, „blue_sensor“ und „ambient_light“ sowie die Funktion „follow_line_with_color_trigger“.

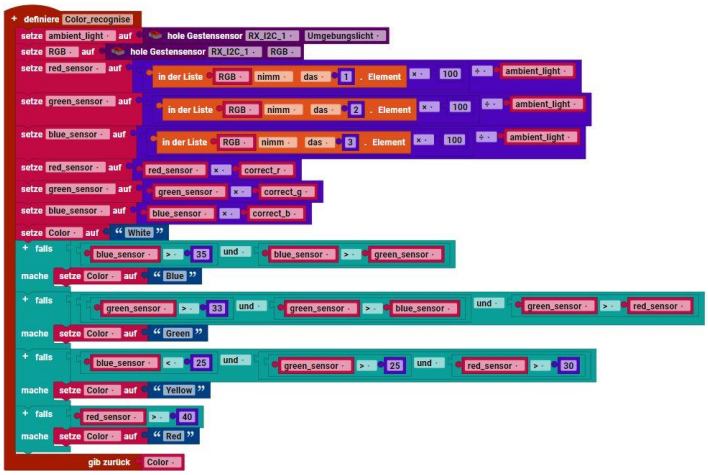
Wichtig: Die weiteren Programmdarstellungen des Programmstarts beziehen sich auf den Schleifeninhalt.



In die Schleife wird zuerst die Funktion „Follow_line_with_color_trigger“ eingebaut. Diese besteht aus nebenstehenden Programmzeilen. Doch Achtung: In dieser Funktion benötigst du noch die Funktion „Follow_line“. Somit musst du beide Funktionen erzeugen.

In die „falls mache“-Abfrage musst du noch eine Wertezuweisung aus der Funktion „Color_recognise“ abfragen. Falls der Wert aus der Funktion „≠“ „White“ ist, soll die Schleife abgebrochen werden.

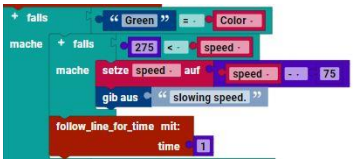
Wie in der Aufgabe festgelegt, soll das Modell der schwarzen Linie entlangfahren. Trifft es auf ein Farbfeld, muss der Farbwert ermittelt werden, um je nach Farbe ein Ereignis zu starten. Dazu musst du die Funktion „Color_recognice“ (Farberkennung) erzeugen.

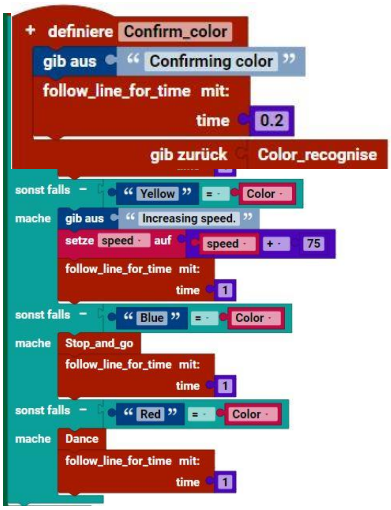


Die benötigten Variablen hast du schon erzeugt, so dass du die nebenstehende Funktion programmieren kannst. Über die mathemischen Funktonen wird der aktuelle Farbwert ermittelt und der Variable „Color“ übergeben.

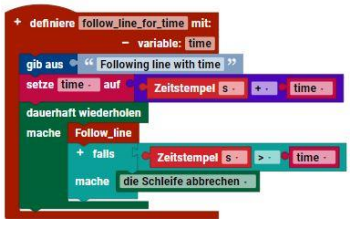
Der Wert für „Color“ wird anschließend im Programmstart ausgewertet und entsprechende Ereignisse gestartet.

Beginne mit der Auswertung von „Grün“. Wurde die Farbfläche erkannt, steht in der Variable „Color“ „Green“. Jetzt muss überprüft werden, ob die Variable „speed“ „<“ (kleiner) „275“ ist. Ist dies der Fall, musst du „speed“ auf den Wert „speed – 75“ setzen. Mit diesem aktualisierten Wert wir jetzt die Bewegung des Modells durchgeführt.



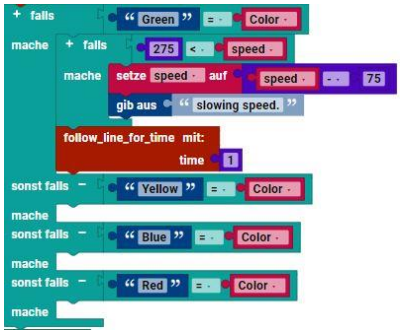


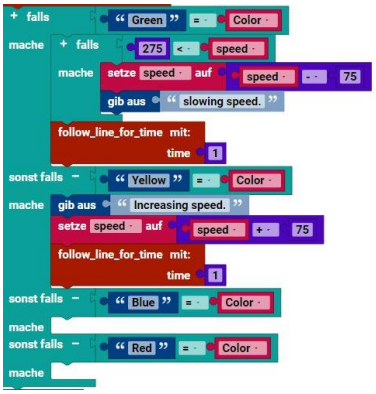
Die Programmteile schreibst du in eine „falls mache sonst“-Bedingung. Wie du erkennen kannst, benötigst du noch die Funktion „follow_line_or_time mit“-„time“. Erzeuge die Funktion und füge sie an entsprechender Stelle in Startprogramm ein.



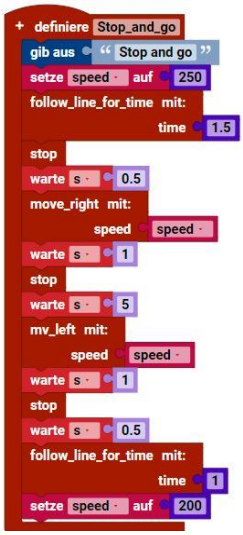
Im Startprogramm änderst du den Wert für „time“ auf „1“ ab.

Füge in das Startprogramm noch drei weitere Auswertungen ein. Ändere „sonst falls“ auf „Yellow = color“, „Blue = color” und “Red = color” um. Dazu duplizierst du einfach die benötigen Befehle und fügst sie entsprechend ein. Die zweite Auswertung kannst du aus der ersten duplizieren. Hier musst du nur die Geschwindigkeit erhöhen, als „75“ zu „speed“ addieren und dann den Wert „speed“ zuweisen.





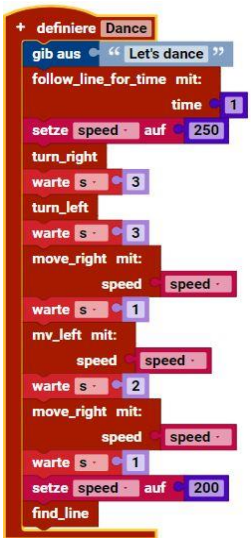
Bald geschafft: Noch müssen zwei Auswertungen definiert werden. Dazu erzeugt du eine Funktion mit dem Namen „Stop_and_go“ für die Farbe „Blue“ und eine Funktion mit dem Namen „Dance“ für die Farbe „Red“. Erzeuge die Funktionen anhand der Programmvorgabe.



Die beiden Funktionen fügst du entsprechend in den Programmstart ein, gefolgt von jeweils der Funktion „follow_line_for_time …“.

In „Stop_and_go“ bewegt sich das Modell nach rechts, stoppt und dann nach links und fährt anschließend der schwarzen Linie entlang.

In „Dance“ dreht sich das Modell in verschiedenen Richtungen. Die Bewegungen ähneln einem Tanz.

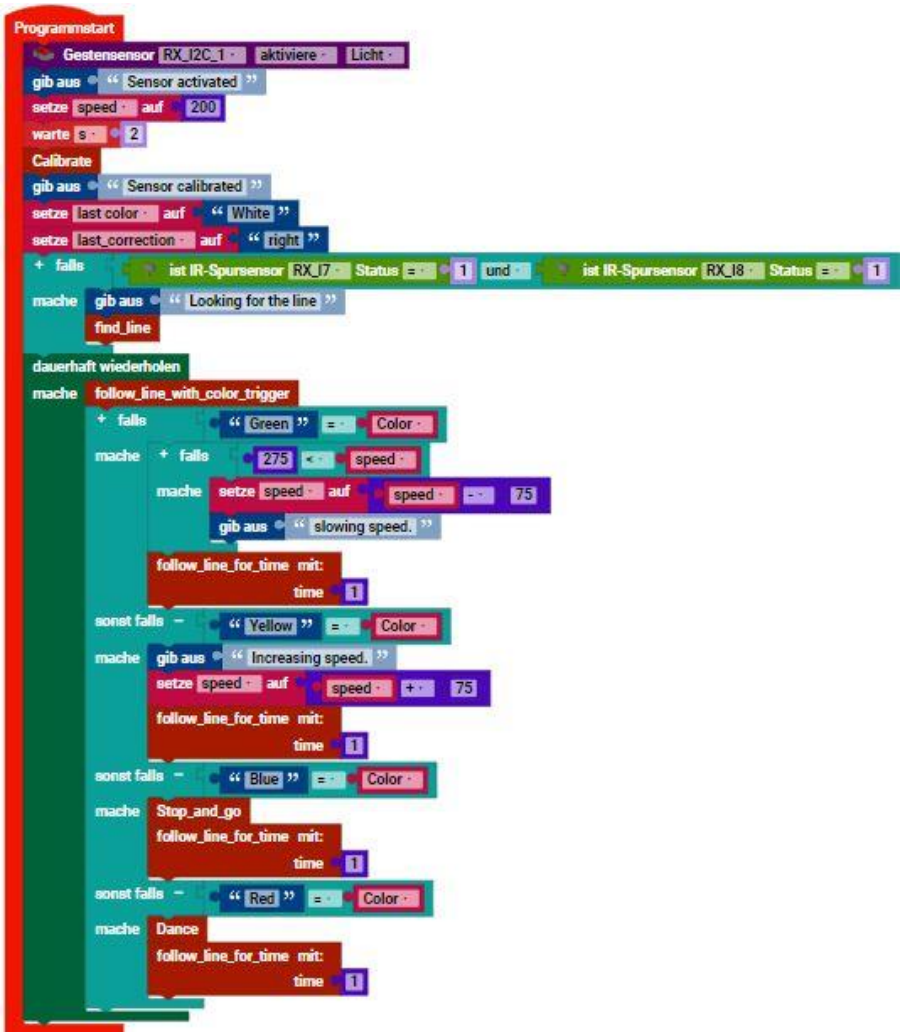


Eine kleine Funktion musst du noch erstellen. Diese bestätigt innerhalb der Farbermittlung die Farbe. (Welche Funktion/Screenshot)

Somit wären die wesentlichen Funktionen erstellt und du kannst das Programm nochmals abspeichern. „Omniwheels_track_ gesture_sensor_4.rpp“

Teste das geladene Programm

Zum Schluss noch einmal das gesamte Programm des Programmstarts.



Herzlichen Glückwunsch, dass du es bis hierher geschafft hast. Das Thema Farberkennung ist wirklich ziemlich kompliziert, und hier haben wir nur eine mögliche Lösung gesucht. Die Idee, den Sensor zu Beginn automatisch zu kalibrieren, hilft uns, die Farben besser zu erkennen, aber es ist nicht perfekt. Zum Beispiel, wusstest du, dass vor einigen Jahren Fernsehkameras, bevor sie außerhalb des Studios aufnahmen, ein weißes Blatt vor die Kamera hielten, um den Weißabgleich zu machen? Das war im Grunde, damit die Kamera die Farbe des Lichts kannte und ein weißes T-Shirt im Fernsehen auch weiß und nicht gelb aussah. Unternehmen investieren auch heute noch viel Zeit in die Entwicklung von Software, damit zum Beispiel dein Handy, wenn es ein Foto macht, automatisch die Farbe Weiß erkennen und somit diese Farbe und alle anderen korrekt darstellen kann.

Warum erzähle ich dir das? Weil vielleicht die Farberkennung trotz der Kalibrierung nicht richtig funktioniert hat. Aber das macht nichts, du musst nur die Grenzen der Farberkennungsfunktion leicht anpassen, also wie viel Rot, Grün oder Blau zum Beispiel Gelb hat. Hier musst du selbst mit diesen Werten spielen, bis du bessere Ergebnisse erzielst. Du kannst auch ausprobieren, wie unter verschiedenen Lichtfarben, dem Licht im Zimmer oder unter der Sonne oder an einem bewölkten Tag, es besser oder schlechter funktioniert.

Und schließlich, denke daran, dass dies eine fischertechnik Baukasten ist. Was bedeutet das? Es bedeutet, dass du hier die Grundlagen der Programmierung gelernt hast, aber von hier aus öffnen sich dir viele Möglichkeiten. Entwerfe deinen eigenen Roboter, programmiere ihn, spiele mit ihm, aber bleibe nicht nur bei den Modellen, die wir gesehen haben. Genieße es, den Roboter zu erschaffen, kombiniere ihn mit anderen fischertechnik-Komponenten, die du hast, und erschaffe neue, komplexere, lustigere Roboter. Genieße es, zu experimentieren und Fehler zu machen, um die beste Lösung zu finden, damit der Roboter das tut, was du willst!

Das gesamte fischertechnik-Team wünscht dir viel Spaß bei allem, was du noch entdecken wirst, und bei all den wunderbaren Robotern, die du mit ihnen bauen wirst!