

HANSER



Leseprobe

zu

Open Robots für Maker Programmierspaß und smarte Elektronik mit Makeblock

Erik Bartmann
Jörn Donges

ISBN (Buch): 978-3-446-45489-7

ISBN (E-Book): 978-3-446-45612-9

ISBN (E-Pub): 978-3-446-45791-1

Weitere Informationen und Bestellungen unter

<https://www.hanser-fachbuch.de/>

sowie im Buchhandel

© Carl Hanser Verlag, München

Inhaltsverzeichnis

1	Einführung	1
1.1	Die faszinierende Welt der Robotik	1
1.2	Hinweise zum Arbeiten mit diesem Buch	3
2	Die Makeblock-Produktwelt	7
2.1	mBot – der Basisroboter	7
2.2	mBot Ranger – der große Bruder des mBot	11
2.3	Ultimate Robot Kit – für alle, die es ernst meinen	13
2.4	Kommt ein Airblock geflogen...	15
2.5	XY-Plotter Robot Kit	16
3	Die Sinne des mBot – Sensoren, Aktoren und Verbindungen ..	19
3.1	Die RJ-25-Buchsen	21
3.2	Der USB-Port	22
3.3	Die RGB-LEDs	26
3.4	Der Buzzer	27
3.5	Die Infrarotschnittstelle	28
3.6	Der Lichtsensor	29
3.7	Der Taster	29
3.8	Die Motoranschlüsse	30
3.9	Der Ein-/Aus-Schalter	32
3.10	Die externe Spannungsversorgung	33
3.11	Die Funkverbindung	36

4	Das Gehirn des mBot – die mBlock-Entwicklungsumgebung ..	41
4.1	Die Programmiersprache Scratch	42
4.2	Die Installation von mBlock	44
4.3	Die unterschiedlichen Wege, den mBot zu programmieren	45
4.4	Die Firmware an den mBot senden	46
4.5	Die Bluetooth-Verbindung herstellen	51
4.6	Der Arduino-Mode in mBlock	56
4.7	Die Arduino/Genuino-Entwicklungsumgebung	60
5	Den mBot über PC und Fernbedienung steuern (Projekt 1)	65
5.1	Die Steuerung über den PC	65
5.2	Die IR-Fernbedienung	69
6	Die Makeblock-App für Smartphones und Tablets	87
6.1	Installieren	88
6.2	Spielen	89
6.3	Erstellen	91
6.4	Bauen	94
6.5	Weitere Möglichkeiten	95
7	Der mBot folgt einer Linie – der Line-Finder-Sensor (Projekt 2)	97
7.1	Was benötigen wir?	98
7.2	Auf Linie bleiben – der Line-Finder-Sensor	99
7.3	Das mBlock-Skript	102
7.4	Weitere Ideen	107
8	Freifahrt für den mBot – der Ultraschallsensor (Projekt 3)	109
8.1	Was benötigen wir?	110
8.2	Das mBlock-Skript	113
8.3	Nützliche Zusatzinformationen	115
8.4	Mögliche Varianten	121

9	Von hell bis dunkel und alles dazwischen – der Lichtsensor (Projekt 4)	123
9.1	Was benötigen wir?	124
9.2	Der Lichtsensor auf dem mCore-Board	124
9.3	Das mBlock-Skript	129
9.4	Die Koordinatenumrechnung	133
9.5	Hell dreht schnell	134
9.6	Weitere Ideen	136
10	Analog gegen digital – Einsatz von Potenziometer, Joystick und Multimeter	137
10.1	Was benötigen wir?	139
10.2	Die ADC-Anschlüsse am Mikrocontroller	140
10.3	Die Zusammenhänge zwischen Spannung, Strom und Widerstand	141
10.4	Der Vergleich zwischen Potenziometer und Joystick	150
10.5	Das mBlock-Skript	153
10.6	Nützliche Zusatzinformationen	154
10.7	Fazit	157
11	Auf Schall reagieren – der Soundsensor (Projekt 5)	159
11.1	Was benötigen wir?	160
11.2	Ein zweiter Soundsensor kommt hinzu	162
11.3	Der tanzende mBot	165
12	Orientierung ist alles! Das Kompassmodul (Projekt 6)	167
12.1	Was benötigen wir?	167
12.2	Das mBlock-Skript	170
12.3	Nützliche Zusatzinformationen	171
12.4	Varianten	172
13	Wir bauen eine Soundmaschine (Projekt 7)	175
13.1	Was benötigen wir?	175
13.2	Das mBlock-Skript	183
13.3	Nützliche Zusatzinformationen	187

14	Periskop ausfahren! Rundumsicht-Scanner – Einsatz von Servomotor und Ultraschallsensor (Projekt 8)	189
14.1	Was benötigen wir?	189
14.2	Das mBlock-Skript	199
14.3	Das ScanLine-Skript	200
14.4	Das Ping-Skript	206
14.5	Nützliche Zusatzinformationen	208
14.6	Fazit	214
15	Sag es mit Farben! Die RGB-LEDs (Projekt 9)	217
15.1	Was benötigen wir?	217
15.2	Die Wahl des Moduls	220
15.3	Das mBlock-Skript	223
15.4	Varianten	226
16	Das Farbenspiel – weitere Einsatzmöglichkeiten der RGB-LEDs (Projekt 10)	229
16.1	Was benötigen wir?	229
16.2	Das mBlock-Skript	233
16.3	Varianten	243
17	Der mBot zeigt Gesicht – die LED-Matrix (Projekt 11)	245
17.1	Was benötigen wir?	246
17.2	Das mBlock-Skript für „zeige Text“	250
17.3	Das mBlock-Skript für „zeige Zeit“	254
17.4	Das mBlock-Skript für „zeige Zeichnung“	258
17.5	Nützliche Zusatzinformationen	260
17.6	Mögliche Varianten	261
18	Das große Krabbeln – der mBot wird zum Käfer (Projekt 12)	263
18.1	Was benötigen wir?	264
18.2	Wir montieren die Kuppelstangen	265
18.3	Das große Krabbeln beginnt	269

19	Wir bauen einen Kameraroboter – das Ultimate Robot Kit (Projekt 13)	271
19.1	Was benötigen wir?	271
19.2	Aufbau	272
19.3	Eine Kamerafahrt steuern	276
19.4	Livestreams der Kamera übertragen	277
20	Ab in die Cloud! IoT-Anwendungen mit Makeblock und Microsoft Azure (Projekt 14)	283
20.1	Die Cloud-Computing-Plattform Microsoft Azure	284
20.2	Gesichtserkennung mit Makeblock und Microsoft Azure	284
21	Das mBot-Standardprogramm	289
21.1	Was benötigen wir?	289
21.2	Aktionen über die IR-Fernbedienung ausführen	290
21.3	Nützliche Zusatzinformationen	292
22	Der mBot wird zum Wachhund – ein Alarmsystem auf Basis des PIR-Motion-Sensors (Projekt 15)	293
22.1	Was benötigen wir?	294
22.2	Programmierung mit Python	295
22.3	Das mBlock-Skript	297
22.4	Der Arduino-Sketch	299
22.5	Das Python-Skript	302
22.6	Varianten	304
	Stichwortverzeichnis	305

1

Einführung

■ 1.1 Die faszinierende Welt der Robotik

Wir leben in einem spannenden Zeitalter. Die Digitalisierung erfasst immer mehr Bereiche des alltäglichen Lebens und hat schon begonnen, die Gesellschaft nachhaltig zu verändern. Heutzutage ist nahezu jede Art von Fachwissen innerhalb von Sekunden verfügbar. Menschen auf der ganzen Welt sind miteinander vernetzt und tauschen sich in Echtzeit aus, egal, wie weit sie voneinander entfernt sind. Doch nicht nur die Kommunikation und das Arbeitsleben verändern sich durch die digitale Revolution. Wir sind mehr und mehr in der Lage, gefährliche oder auch sehr monotone Arbeiten auszulagern und an Maschinen abzugeben. In modernen Fabrikationshallen für Autos haben Menschen nur noch beaufsichtigende Funktionen. Es sind Roboter, die an den Fließbändern zielsicher die Fahrzeuge zusammensetzen. Auch in lebensfeindlichen Umgebungen tun sie ihren Dienst. Roboterfahrzeuge sind schon bis auf den Mond oder sogar zum Planeten Mars vorgedrungen. Auch im Haushalt treffen wir die kleinen Helfer immer häufiger an, sei es als intelligenter Staubsauger, als Rasenmäher oder Putzroboter.



Bild 1.1 Greifarm eines modernen Industrieroboters (Quelle: Pixabay)

Die meisten Menschen denken beim Begriff Roboter zuerst an humanoide Roboter, also technische Nachbildungen des Menschen. Der Cyborg oder Android hat seinen festen Platz in der Science-Fiction-Literatur und bestimmt wohl auch deshalb die Vorstellungen, die wir von Robotern haben. Tatsächlich werden die humanoide Roboter vor allem im technikbegeisterten Japan geliebt und perfektioniert.



Bild 1.2 Ein humanoider Roboter (Quelle: Pixabay)

Doch auch dort, wo man es nicht auf den ersten Blick vermutet, verrichten Roboter ihre Arbeit. Es dauert nicht mehr lange, bis Autos in der Lage sein werden, vollkommen autonom zu fahren. Der Fahrer kann sich entspannt zurücklehnen und sich vom Auto ans Ziel bringen lassen. Er muss lediglich die Fahrt überwachen und in Ausnahmesituationen eingreifen. Zunehmend beliebter werden auch kleine Multicopter/Drohnen, die heutzutage aus Filmaufnahmen schon nicht mehr wegzudenken sind und die auch für Landvermessungen benutzt werden. Mit ihren Fähigkeiten, automatisch die Höhe zu halten oder einem Sender zu folgen, sind auch sie Roboter.

Letztlich fallen auch so profane Geräte wie Brotbackautomaten oder die intelligente Waschmaschine mit Schonprogramm in diese Kategorie: Es sind Maschinen, die von einem programmierbaren Controller gesteuert werden und sich daher in ihrem Verhalten flexibel an die Gegebenheiten anpassen können. Dies bezeichnen wir im weitesten Sinne als Roboter. Sicher fallen dir noch weitere Beispiele ein. Alle haben eine gemeinsame Struktur. Da sind einmal die **Sensoren**, mit denen Roboter die Umgebung wahrnehmen und sich ein Bild von ihrer Umwelt machen können, bestehend aus Messwerten. Das können Temperatur,

Druck oder Lichtsensoren sein, aber auch komplexere Geräte wie eine Kamera oder ein Radar. Alles, was dem Roboter hilft, seine Umwelt wahrzunehmen, bezeichnen wir als Sensor.

Das Gegenteil davon sind sozusagen die Gliedmaßen, also alle Teile, durch die der Roboter auf seine Umgebung einwirken und etwas verändern kann. Diese bezeichnen wir als **Aktoren**. Dazu zählen z. B. Motoren, Greifarme oder auch Lampen und Displays, die Informationen anzeigen.

Zwischen Sensoren und Aktoren steht eine mehr oder weniger intelligente Verarbeitung. Ein wesentlicher Aspekt von Robotern ist, dass sie programmierbar sind und damit die große Welt der Software mit ins Boot holen. Es sind Algorithmen, die darüber bestimmen, wie der Roboter auf die Umweltdaten reagiert und welche Handlungen er ausführt.

Die Einsatzgebiete sind nahezu unbegrenzt und halten Einzug in vielen unterschiedlichen Bereichen, um die Arbeit des Menschen zu erleichtern oder ihn komplett zu ersetzen.

■ 1.2 Hinweise zum Arbeiten mit diesem Buch

Das erwartet dich in diesem Buch

Dieses Buch ist eine Einführung in die faszinierende Welt der Robotik. Du brauchst keine Vorkenntnisse, denn wir werden alles von Grund auf erklären und aufbauen. Doch auch wenn du schon ein wenig programmieren kannst und nun deine Ideen nicht nur am Bildschirm verwirklichen willst, sondern live und in 3D einen kleinen Roboter steuern willst, dann bist du hier richtig.

Es gibt diverse Hersteller kleiner Robotermodelle, die von purem Spielzeug bis hin zu fast professionellen Plattformen zur Entwicklung von Robotikprojekten reichen. Wir haben uns dafür entschieden, unsere Projekte mit den Robotern der Firma Makeblock zu verwirklichen. Die Makeblock-Roboter können dich einen sehr weiten Teil deiner Robotikkarriere begleiten. Sie sind einfach genug, um auch für Kinder einen spielerischen Einstieg ins Thema zu gewährleisten. Gleichzeitig bieten sie Fortgeschrittenen und Profis die notwendige Offenheit, um auch sehr komplexe Schaltungen und Programmierungen umzusetzen.

Für den Einstieg steht die grafische Programmierplattform mBlock mit der Programmiersprache Scratch bereit, für die du keinerlei Code eingeben musst. Die Anweisungen an den Roboter kannst du einfach grafisch am Bildschirm erstellen, indem du Funktionsblöcke mit der Maus verschiebst und verbindest. Dank der Nutzung von Arduino-Controllern als Hirn und Schaltzentrale der Makeblock-Roboter steht dir darüber hinaus als Maker das

riesige Arduino-Universum zur Verfügung. Es gibt etliche interessante Arduino-Projekte in Büchern und im Internet, die du nachbauen und mit deinen Makeblock-Robotern verbinden kannst. Anregungen für eigene Projekte kannst du dir z. B. im Buch *Mach was mit Arduino!* (ISBN 978-3-446-45128-5) von Robert Jänisch und Jörn Donges holen, das ebenfalls im Hanser Verlag erschienen ist.

So ist dieses Buch aufgebaut

In Kapitel 2 werfen wir einen Blick auf die Makeblock-Produktwelt und gehen auf die unterschiedlichen Eigenschaften der Makeblock-Roboter und -Drohnen ein.

Kapitel 3 und 4 enthalten wichtige Grundlagen, die wir für die Verwirklichung unserer Projekte brauchen. Wir nehmen uns darin das kleine Roboterfahrzeug mBot genauer vor. Wir stellen alle Sensoren und Anschlüsse sowie die Entwicklungsumgebung vor, mit der du den mBot und andere Robotermodelle steuern und programmieren kannst.

In Kapitel 5 springen wir mitten in die Praxis und entwickeln auf zwei unterschiedliche Weisen eine Fernsteuerung für den mBot.

Kapitel 6 ist dem Einsatz mobiler Geräte gewidmet. Es stellt die wichtigsten Apps genauer vor, die man beim Arbeiten mit Makeblock-Robotern verwenden kann.

Zu den Standardanwendungen für kleine Roboterfahrzeuge gehört das automatische Folgen einer Linie auf dem Boden oder das Erkennen von Hindernissen mit einem Ultraschall-Sensor. Diese Projekte stellen wir in Kapitel 7 und 8 vor.

In Kapitel 9 bis 12 widmen wir uns den einzelnen Sensoren und experimentieren mit dem Lichtsensor, dem Schallsensor und dem Kompass-Modul.

Mit der Soundmaschine erwartest dich in Kapitel 13 ein besonderes Projekt. Wir verwandeln den mBot in ein Keyboard zum Musik machen. Dieses Projekt stellt etwas höhere Anforderungen an deine Elektronik-Fertigkeiten. Du lernst, Schaltungsprototypen mit einer Lochrasterplatine aufzubauen.

Auch Kapitel 14 stellt ein Projekt für fortgeschrittene Maker vor. Der mBot erhält eine Radar-Antenne als Rundumsicht-Scanner.

Kapitel 15 bis 17 stehen im Zeichen von Licht und Farbe. Wir lernen die RGB-LEDs und ein Farbenspiel kennen. Danach verpassen wir dem mBot eine LED-Matrix, auf der wir Muster und Laufschriften zum Leben erwecken.

In Kapitel 18 kannst du deine mechanischen Fähigkeiten unter Beweis stellen und den mBot vom Radantrieb auf einen sechsbeinigen Antrieb umstellen.

Wenn du mit einer Filmkarriere liebäugelst, dann ist der Kamera-Bot aus Kapitel 19 etwas für dich. Darin konstruieren wir auf Basis des Ultimate Robot Kit einen praxistauglichen Kamerawagen für Smartphone oder Spiegelreflexkameras. Über eine Streaming-App werden wir das Kamerabild, das der Bot aufnimmt, in Echtzeit auf einem Bildschirm streamen.

Wenn du dich für Cloud-Anwendungen und das Internet of Things (IoT) interessierst, dann solltest du auf jeden Fall das Projekt in Kapitel 20 durcharbeiten, in welchem wir die Microsoft-Azure Cloud nutzen, um Gesichter auf einem Kamerabild zu erkennen.

Kapitel 21 widmet sich dem Programm, das auf dem mBot bei Inbetriebnahme standardmäßig installiert ist.

Zum Abschluss präsentieren wir mit dem Alarmsystem in Kapitel 22 ein anspruchsvolleres Projekt, bei dem du die Meldung eines Bewegungssensors über ein in Python programmiertes Skript automatisch per E-Mail verschicken kannst.

Los geht's!

Doch nun ist es an der Zeit, den Protagonisten unserer Reise durch die Roboterwelt genauer kennenzulernen. Es ist der mBot von Makeblock, mit dem fast alle Projekte in diesem Buch (bis auf den Kameraroboter aus Kapitel 19) realisiert werden. Er ist das kleinste Modell der Makeblock-Roboter, aber auch mit ihm lässt sich schon so einiges Interessantes anstellen. Wir wünschen dir viel Spaß beim Entdecken, Nachbauen und Experimentieren!

Brühl/Hamburg, Januar 2018

Erik Bartmann

Jörn Donges

7



Der mBot folgt einer Linie – der Line-Finder-Sensor (Projekt 2)

Nachdem wir nun einen Blick auf die Hardware geworfen haben und du auch weißt, welche Möglichkeiten für Steuerung und Entwicklung bereitstehen, ist der mBot bereit für weitere Projekte. Auch die Entwicklung und Übertragung von Scratch-Skripten ist nichts Neues mehr für dich, denn mit der Fernsteuerung aus Kapitel 5 hast du das ja bereits gemacht. Zum Kennenlernen der grundlegenden Programmiermöglichkeiten und weiterer Funktionen beschränken wir uns in den folgenden Kapiteln darauf, die bereits im mCore-Board eingebauten Sensoren zu nutzen. Du benötigst also erst einmal keine zusätzlichen elektronischen Bauteile.

Zunächst richten wir unser Augenmerk auf das interessante Thema des autonomen Fahrens. In diesem Bereich wurde in den letzten Jahren viel Forschungsaufwand betrieben. Es wird nicht mehr lange dauern, bis selbstständig fahrende Autos zum alltäglichen Straßenbild gehören. Doch um ein Roboterfahrzeug autonom fahren zu lassen, brauchen wir irgendeine Form der Orientierung. Am elegantesten wäre es natürlich, mit einer Kamera oder einem Entfernungssensor die Umgebung abzutasten und so eine Karte der Umgebung anzulegen, durch die der mBot fahren kann. Dies würde tatsächlich den autonom fahrenden Pkw sehr nahekommen, denn auch diese versorgen sich per Kameras und Abstandssensoren mit Daten der unmittelbaren Umgebung. Diese Datenerfassung dient nicht nur dazu, Hindernissen auszuweichen, sondern auch, das Fahrzeug entlang der Straße zu führen. Sensoren tasten die Begrenzungen der Straßen ab und erkennen die Seiten- und Mittellinien. Dadurch kann das Roboterfahrzeug automatisch dem Straßenverlauf folgen. Wie auf virtuellen Schienen wird das Fahrzeug entlang dieser Markierungen geführt. Diesen Aspekt des autonomen Fahrens wollen wir in diesem Kapitel nachbilden. Wir werden den mBot dazu bringen, einer auf den Boden gezeichneten Linie zu folgen.

■ 7.1 Was benötigen wir?

Folgende Hardware wird benötigt:

Grundvoraussetzung	Zusatzmodule/Bauteile
<p data-bbox="454 356 511 384">mBot</p> 	<p data-bbox="962 356 1019 384">keine</p> 

Der mBot ist von Haus aus mit einem sogenannten Line-Finder ausgerüstet, der in der Lage ist, den Untergrund nach hellen bzw. dunklen Flächen abzuscannen. Wir machen uns diese Fähigkeit zunutze, um den mBot einer vorgegebenen Spur folgen zu lassen, die z. B. auf ein Stück Papier oder Pappe aufgezeichnet wurde. Eine entsprechende Vorlage liegt jedem mBot-Kit als Poster bei (siehe Bild 7.1). Kommt der mBot während seiner Fahrt über diese Vorlage von der Spur ab, reagiert der Sensor auf den Helligkeitswechsel, und es kann entsprechend gegengesteuert werden. Wir sehen uns das gleich im Detail genauer an.

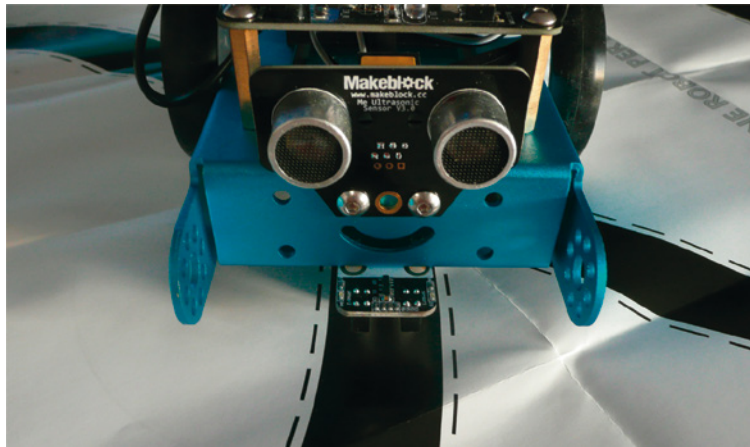


Bild 7.1 Der Line-Finder unter dem Chassis tastet den Untergrund ab.

■ 7.2 Auf Linie bleiben – der Line-Finder-Sensor

Bevor wir uns der eigentlichen Programmierung zuwenden, wollen wir das Prinzip des Line-Finders verstehen. In Bild 7.2 siehst du den mBot von unten und kannst den Line-Finder mit den Sensoren 1 und 2 erkennen.

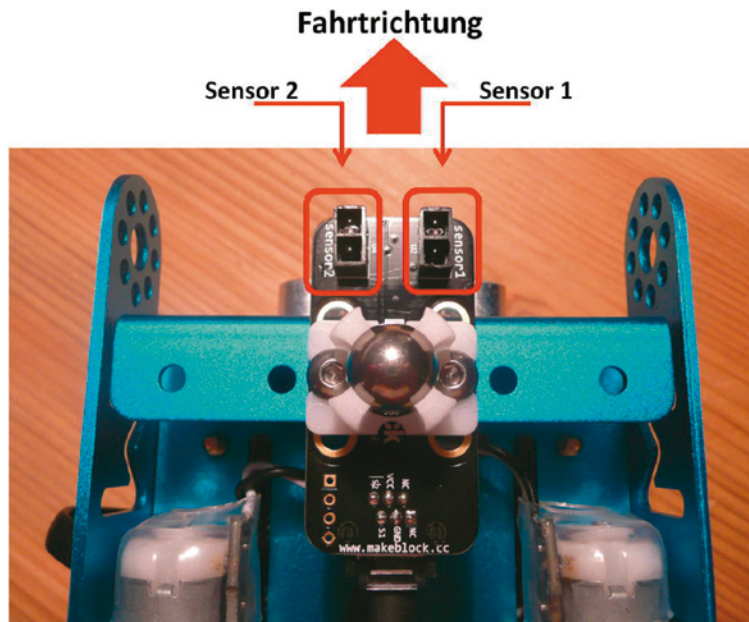


Bild 7.2 Der Line-Finder am mBot

Auf der Oberseite des Line-Finders befindet sich zur Kontrolle der Funktion pro Sensor je eine LED (siehe Bild 7.3).



Bild 7.3 Die Status-LEDs des Line-Finders

Wenn du z. B. mit der Hand einen Sensor abdeckst, erkennst du den Wechsel der LED. Setzt du den mBot mit dem Line-Finder auf eine Unterlage mit einer schwarzen Spur, reagieren die Sensoren entsprechend. Bei einem schwarzen Untergrund verlöschen die Sensoren, bei einem weißen Untergrund leuchten sie. Je nach Abweichung von der Spur kommt es zu unterschiedlichen Kombinationen, die in Richtungsänderungen während der Fahrt umgerechnet werden können. In mBlock fragst du den Line-Finder mit dem Block aus Bild 7.4 ab.

Line-Follower-Sensor Port 2

Bild 7.4 Der Line-Follower-Sensor-Block fragt die Sensoren ab.

Wie aber funktioniert ein einzelner Sensor überhaupt? In Bild 7.5 siehst du, dass ein einzelner Sensor eigentlich aus zwei Komponenten besteht.



Bild 7.5 Die zwei Komponenten jedes einzelnen Sensors des Line-Finders

Wir haben es einerseits mit einem Sender zu tun, der in Form einer Infrarot-Leuchtdiode Signale aussendet, die andererseits von einem Empfänger in Form eines Fototransistors registriert werden. Helle bzw. dunkle Flächen, die vor einem Sensor vorbeigeführt werden, stellen unterschiedliche Reflexionsmerkmale dar und werden entsprechend abweichende Reaktionen des Fototransistors nach sich ziehen. Die Infrarot-LED sendet also ihr Licht auf eine Fläche aus, das entweder von einem weißen Untergrund reflektiert oder von einem schwarzen Untergrund absorbiert wird. Der Fototransistor als Empfänger reagiert entsprechend dieser beiden Möglichkeiten, wobei die Elektronik die Signale an den Mikrocontroller weiterleitet und bewertet. Das Prinzip bzw. das Zusammenspiel beider Komponenten ist in Bild 7.6 zu erkennen.



Bild 7.6 Das Sensorprinzip des Line-Finders

Auf der linken Seite haben wir es mit einer dunklen Unterlage zu tun, wobei das Licht der IR-LED von der schwarzen Fläche verschluckt (d. h. absorbiert) wird. Der Fototransistor empfängt kein Signal. Dagegen haben wir es auf der rechten Seite mit einer hellen Unterlage zu tun, von der das Licht der IR-LED zurückgestrahlt (d. h. reflektiert) wird, was der Fototransistor als Signal empfängt. Diese beiden möglichen Zustände – Signal oder kein Signal – kann der Mikrocontroller empfangen und gemäß der Programmlogik verarbeiten bzw. darauf reagieren.

Der Rückgabewert des Sensors ist ein digitaler Wert, das heißt, jeder Fototransistor kann nur zwei Zustände zurückmelden: entweder hell oder dunkel.

Zwei Fototransistoren mal zwei Zustände – das ergibt genau vier mögliche Rückgabewerte, die den vier in Bild 7.7 bis Bild 7.10 abgebildeten Situationen entsprechen.

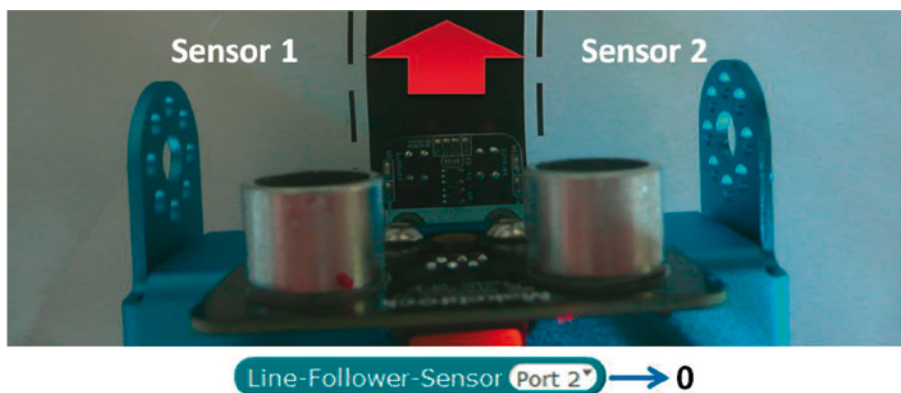


Bild 7.7 Möglichkeit 1: Genau in der Spur, Rückgabewert 0

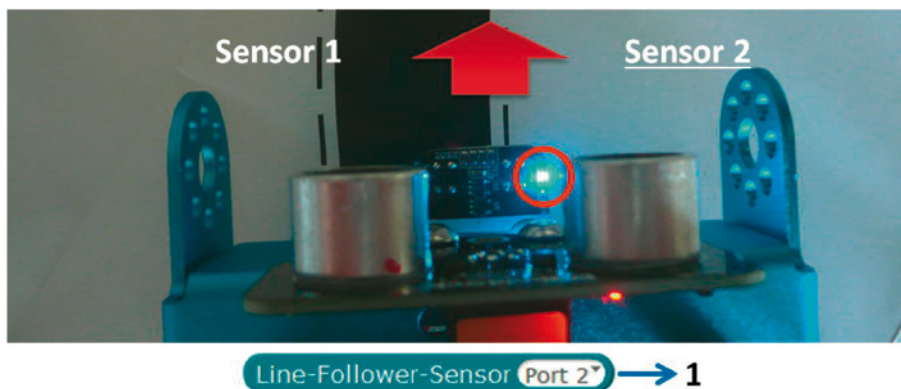


Bild 7.8 Möglichkeit 2: Rechts neben der Spur, Rückgabewert 1

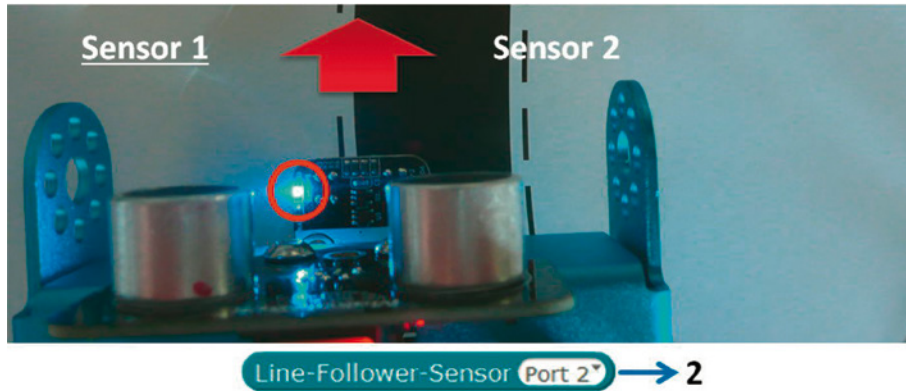


Bild 7.9 Möglichkeit 3: Links neben der Spur, Rückgabewert 2

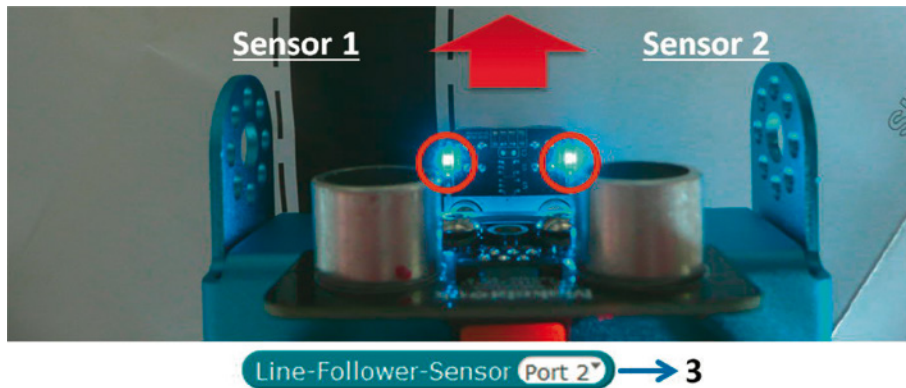


Bild 7.10 Möglichkeit 4: Komplett neben der Spur, Rückgabewert 3

■ 7.3 Das mBlock-Skript

Über das folgende Skript können wir das Verhalten des Line-Finders mithilfe der Unterlage sehr gut testen.



HINWEIS: Achte darauf, dass das Umgebungslicht den Sensor nicht zu stark stört. Ein zu helles Licht einer Lampe oder der direkte Schein der Sonne kann die Messergebnisse beeinflussen und zu unerwünschten Reaktionen führen.

Zur besseren Kontrolle des Rückgabewertes des Sensors legen wir zunächst die Variable *LineFinderValue* an und machen sie mit einem Häkchen sichtbar (siehe Bild 7.11).



Bild 7.11 Die Variable für den Rückgabewert des Line-Finders



Variablen sind ein wichtiges Konzept in der Programmierung. Mit ihnen können wir veränderliche Werte unter einem frei wählbaren Namen abspeichern und jederzeit darauf zugreifen. Mit einer Variablen weisen wir einem Speicherbereich des Computers einen festen Namen zu, um ihn im Programm bequem anzusprechen und auslesen zu können. Es handelt sich also um Behälter bzw. Container zur Aufnahme der unterschiedlichsten Werte. Der Name eines einzelnen Behälters ist dabei konstant, der Inhalt kann sich jedoch ändern.

Doch zurück zu unserem Skript. Das Skript fragt kontinuierlich den Line-Finder ab, der sich hier an Port 2 befindet, und speichert den Rückgabewert in die zuvor erstellte Variable. Die Port-Nummer musst du natürlich gegebenenfalls anpassen (siehe Bild 7.12).



Bild 7.12 Der Abfrageblock für den Line-Finder

Die folgende Tabelle erläutert die in Bild 7.12 dargestellten Codeblöcke.

Block	Funktion
	Über den <i>wiederhole fortlaufend</i> -Block wird die nachfolgende Abfrage des Sensors kontinuierlich ausgeführt.
	Über den <i>setze auf</i> -Block wird die Variable <i>LineFinderValue</i> mithilfe des <i>Line-Follower-Sensor</i> -Blocks, der mit Port 2 konfiguriert ist, initialisiert.

Wie wir schon in Bild 7.7 bis Bild 7.10 gesehen haben, liefert der Sensor die folgenden Werte zurück:

Möglichkeit	Rückgabewert	Ergebnis
Sensor 1 und Sensor 2 befinden sich innerhalb der schwarzen Spur.	0	Der mBot fährt entlang der schwarzen Spur.
Sensor 2 befindet sich außerhalb der schwarzen Spur.	1	Der mBot befindet sich rechts neben der schwarzen Spur.
Sensor 1 befindet sich außerhalb der schwarzen Spur.	2	Der mBot befindet sich links neben der schwarzen Spur.
Sensor 1 und Sensor 2 befinden sich außerhalb der schwarzen Spur.	3	Der mBot hat die schwarze Spur komplett verlassen.

Wir wollen nun ein Skript erstellen, das auf die Werte des Line-Finders reagiert und den mBot genau entlang der schwarzen Spur fahren lässt. Dazu legen wir ein paar neue Variablen an. Eine davon nimmt die Geschwindigkeit des mBot auf, die anderen dienen dazu, die Rückgabewerte 0 bis 3 des Sensors etwas verständlicher darzustellen (siehe Bild 7.13).

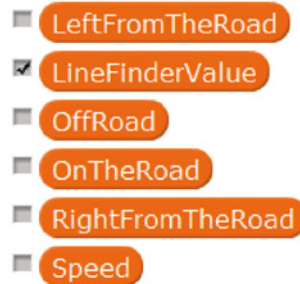


Bild 7.13 Die Variablenliste für den Line-Follower

Bei der Variablen *LineFinderValue* setzen wir das Häkchen, um den Rückgabewert am Bildschirm zu verfolgen. Bei den anderen ist dies unnötig, da sie sich im Programmverlauf nicht ändern.

Der nun folgende Block dient der Initialisierung. Dies ist ein ganz typischer Programmaufbau. Als Erstes müssen wir den Roboter und das Skript in einen definierten Zustand versetzen, von dem die weitere Verarbeitung ausgeht. Dazu werden den verwendeten Variablen zunächst einmal ihre Startwerte zugewiesen (siehe Bild 7.14).

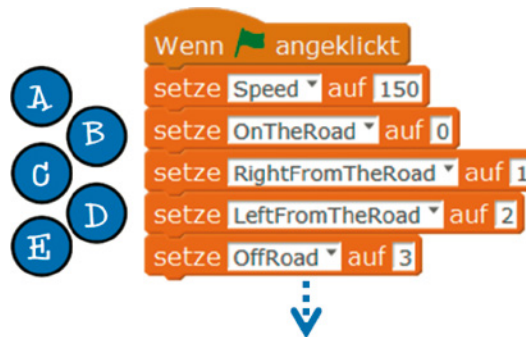







Bild 7.14 Startwerte der beteiligten Variablen

Die folgende Tabelle erläutert die in Bild 7.14 dargestellten Codeblöcke.

Block	Funktion
	Über den <i>setze auf</i> -Block wird die Variable <i>Speed</i> , die für die Geschwindigkeit des mBot verantwortlich ist, auf den Wert 150 gesetzt. Bei einem zu hohen Wert kann es passieren, dass der mBot sehr schnell die Spur verlässt und nicht wieder zurückfindet.

Block	Funktion
	Über den <i>setze auf</i> -Block wird die Variable <i>OnTheRoad</i> mit dem Wert 0 initialisiert, was bedeutet, dass der mBot genau auf der Spur fährt.
	Über den <i>setze auf</i> -Block wird die Variable <i>RightFromTheRoad</i> mit dem Wert 1 initialisiert, was bedeutet, dass der mBot rechts neben der Spur fährt.
	Über den <i>setze auf</i> -Block wird die Variable <i>LeftFromTheRoad</i> mit dem Wert 2 initialisiert, was bedeutet, dass der mBot links neben der Spur fährt.
	Über den <i>setze auf</i> -Block wird die Variable <i>OffRoad</i> mit dem Wert 3 initialisiert, was bedeutet, dass der mBot die Spur komplett verlassen hat.

Beachte, dass hier der Line-Finder noch nicht zum Einsatz kam. Wir haben lediglich den möglichen Rückgabewerten 0, 1 und 2 Namen gegeben, damit wir uns nicht merken müssen, für welchen Zustand die Zahlen stehen. Das ist eine oft verwendete Technik, um Programme lesbarer zu machen, man nennt das auch „sprechende“ Variablennamen.

Sehen wir uns nun den Teil des Skripts an, der für die Steuerung des mBot verantwortlich ist (siehe Bild 7.15).

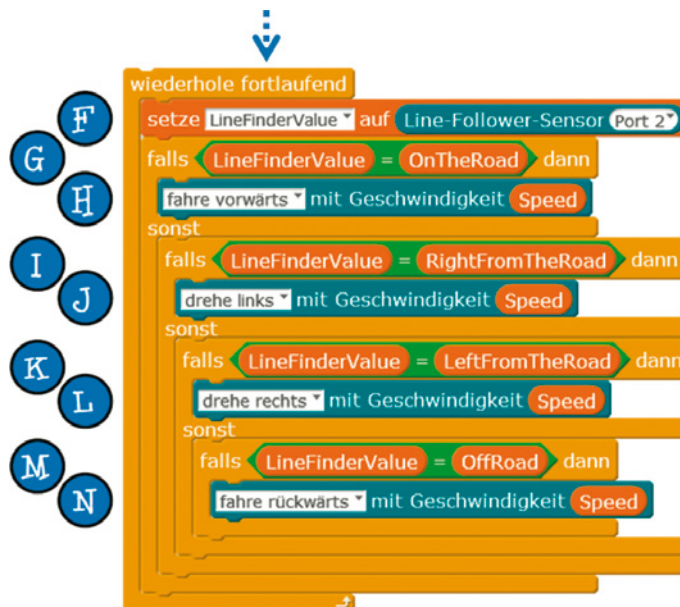


Bild 7.15 Abfrage des Line-Sensors und Steuerung des mBot

Die folgende Tabelle erläutert die in Bild 7.15 dargestellten Codeblöcke.

Block	Funktion
	Über den <i>setze auf</i> -Block wird die Variable <i>LineFinderValue</i> mit dem Sensorwert versorgt, der später ausgewertet wird.
	Über den <i>falls-dann-sonst</i> -Block wird der Wert der Variablen <i>LineFinderValue</i> mit dem Wert der Variablen <i>OnTheRoad</i> verglichen.
	Stimmen die Werte überein, erfolgt über den <i>fahre vorwärts</i> -Block ein Geradeausfahren mit der Geschwindigkeit, die in der Variablen <i>Speed</i> hinterlegt ist.
	Über den <i>falls-dann-sonst</i> -Block wird die Variable <i>LineFinderValue</i> mit dem Wert der Variablen <i>RightFromTheRoad</i> verglichen.
	Stimmen die Werte überein, erfolgt über den <i>drehe links</i> -Block eine Linksdrehung mit der Geschwindigkeit, die in der Variablen <i>Speed</i> hinterlegt ist.
	Über den <i>falls-dann-sonst</i> -Block wird die Variable <i>LineFinderValue</i> mit dem Wert der Variablen <i>LeftFromTheRoad</i> verglichen.
	Stimmen die Werte überein, erfolgt über den <i>drehe rechts</i> -Block eine Rechtsdrehung mit der Geschwindigkeit, die in der Variablen <i>Speed</i> hinterlegt ist.
	Über den <i>falls-dann-sonst</i> -Block wird die Variable <i>LineFinderValue</i> mit dem Wert der Variablen <i>OffRoad</i> verglichen.
	Stimmen die Werte überein, erfolgt über den <i>fahre rückwärts</i> -Block ein Rückwärtsfahren mit der Geschwindigkeit, die in der Variablen <i>Speed</i> hinterlegt ist.

Unser Skript trifft Entscheidungen anhand von Bedingungen, die entweder logisch *wahr* oder *falsch* waren. Derartige Abfragen spielen eine sehr große Rolle innerhalb der Programmierung, denn sie ermöglichen eine Steuerung des Programmablaufs. Derartige Konstrukte werden *Kontrollstrukturen* genannt. Sie leiten Verzweigungen im sonst geradlinigen Skriptverlauf ein. Wir haben dieses Konstrukt über den *falls-dann*-Block in unserem Skript verwendet. Der *wiederhole fortlaufend*-Block fällt ebenfalls in diese Kategorie. Er stellt eine sogenannte *Schleife* dar.

Wenn du das Skript eingegeben hast, kannst du den mBot auf die Unterlage setzen und es ausprobieren. Jedem mBot-Kit liegt ein Poster mit einer Linie bei, du kannst aber auch selbst kreativ werden. Da die Linie eine gewisse Dicke haben muss, ist schwarzes Isolierband dafür gut geeignet. Besorge dir also eine große weiße Pappe und klebe mit dem Isolierband die gewünschte Fahrspur auf. Probiere ruhig aus, wie eng die Kurven sein dürfen, damit der mBot sie noch schafft. Sind sie zu eng, kannst du im Skript den Wert für die Geschwindigkeit heruntersetzen.

Vielleicht hast du dich gefragt, warum der letzte Schritt zur Abfrage, ob der mBot mit seinem Sensor komplett neben der Spur liegt, überhaupt notwendig ist. Wenn er anfänglich einmal korrekt auf die schwarze Spur positioniert wurde, liegt er ja während der Fahrt theoretisch immer entweder rechts oder links daneben. Es kann aber trotzdem vorkommen, dass durch irgendein Fahrmanöver die Spur für den Sensor nicht mehr sichtbar ist. Dann wird mit einer Rückwärtsfahrt versucht, den mBot wieder auf die Spur zu bekommen. Wird dieser Schritt weggelassen, kommt es zu merkwürdigen und unbeabsichtigten Verhalten. Das wäre sicher interessant für einen kleinen Test. Probiere es einfach aus. Lass den letzten Block mit der *OffRoad*-Abfrage weg und untersuche das Verhalten des mBot.

■ 7.4 Weitere Ideen

Du kannst auch eine schwarze Pappe als Unterlage verwenden und die Fahrspur mit weißem Klebeband darauf anbringen. Dann musst du allerdings das Skript anpassen und die Drehungen in entgegengesetzter Richtung durchführen.

Dies sind einige weitere Ideen mit dem Line-Finder für Fortgeschrittene:

- Statt einer Linie kannst du eine „Straße“ aufkleben, also zwei Linien im genügend großen Abstand voneinander. Der mBot soll dann dazwischen bleiben und dem Straßenverlauf folgen.
- Umrande mit dem Klebeband ein Rechteck oder eine andere große geschlossene Form auf die Unterlage. Das ist die Begrenzung für den mBot. Entwickle nun ein Skript, das den mBot im Inneren des Rechtecks hält, ihn also beim Annähern der Grenzlinie wenden lässt. Dies wäre auch schon eine gute Vorbereitung auf ein späteres Projekt, bei dem wir den mBot mit dem Ultraschallsensor ausrüsten und realen Hindernissen ausweichen lassen.

Stichwortverzeichnis

Symbole

2.4G 40
3D-Darstellung 95
4-Button-Modul 243
7-Zip 44
9g Micro Servo Pack 192

A

Abisolierzange 177
Ablaufsequenz 112
Acrylplatte 246
ADC 138
ADCx 141
Airblock 15
Aktoren 3
Algorithmus 111
Aluminiumprofile 9
Aluminiumträgerteile 263
Ampere 36
analog 138
Analog/Digital-Wandler 138
Android 88, 278
Ankathete 196
App 87
Apple 88
Arduino/Genuino-Entwicklungsumgebung 60
Arduino-Mode 56
Arduino-Quellcode 57
Arduino Uno 46
Argumente 82
Array 231
ATmega328-AU 140
Atmel MEGA 328P 45
Auffahrschutz 71
Ausgangsspannung 146
Azure-Cloud 286

B

Batteriehalterung 33
Batterien 33
Baud 300
Bauen 94
Bauteiltoleranz 153
Bedienelemente 91
Bedienfeld 92
Beep 80
Beschleunigungssensor 304
Bewegung 127
Blöcke 10
Bluetooth 37
Bluetooth-Adapter 38
Bluetooth-Modul 37
Breadboard 176
Button 29
Buzzer 27, 71
Byte 227

C

Chassis 19
Cloud 283
Community 9
Continuous Rotation 209
Cosinus 197

D

Dampflock 265
Dampfmaschine 265
Daten & Blöcke 80
Datenleitung 168
Default-Firmware 290
Dezimalwerte 249
digital 137
Digitalisierung 1
Drehbereich 209

Drehbewegung 118, 264
 Drehmoment 265
 Dreieck 116
 Drohne 15

E

Echobild 190
 Ein-/Aus-Schalter 32
 E-Mail 294
 Entfernungsvektor 196
 Entwicklungsumgebung 60, 87
 Entwurf 91
 Erdmagnetfeld 167
 Ereignis 68
 Erstellen 91
 Erweiterungs-Port 139
 Event-Handler 69
 Events 68
 Extensions 10, 48

F

Fadenkreuz 130
 Fahrabweisung 72
 Fahren 89
 Fahrgeräusche 161
 Fahrzeugachse 265
 Farbe 130
 Farbkodierung 139, 156
 Farbmarkierungen 23
 Farbringe 156
 Farbsequenz 231
 Fernbedienung 65, 70
 Fernseher 278
 Fernsteuerung 67
 Feuchtigkeitssensor 304
 Filmaufnahmen 278
 Firmware 46
 Firmware-Upload 22, 218
 Flammensensor 304
 Fortbewegung 263
 Fotografie 285
 Fototransistor 100
 Frequenz 210
 Füllmodus 130
 Funkmodule 37

G

Gassensor 304
 Gegenkathete 196
 Gelenkverbindungen 266
 Geradeauslauf 275
 Geräusch 161

Gesichtserkennung 284
 Getriebe 265
 Gleichspannung 141
 Gleichstrommotor 30, 208
 GMail 302
 GND (Ground) 144
 Grafik-Editor 130, 260
 Greifer 14
 Grid 258
 Grundfarben 219
 Grüne Flagge 57

H

Häkchen 113
 Hat-Block 56, 65
 Header 25
 Header-Datei 62
 Helligkeitssensor 92
 HIGH 180
 Hin- und Herbewegung 264
 HMC5883L 171
 Honeywell 168
 Humanoide Roboter 2
 Hypotenuse 196

I

I²C-Bus 168
 IDLE-Python-GUI 296
 Infrarotfernbedienung 70
 Infrarot-Leuchtdiode 100
 Infrarotschnittstelle 28
 Installation 44
 Intensitätswert 223
 Internet of Things 15
 Interrupt 69
 iOS 278
 IoT 15, 283
 IP-Adresse 280
 IP Webcam 278
 IR-Fernbedienung 69
 IR_R 28
 IR_T 28
 ISM-Band 37

J

Joystick 150

K

Kamerafahrt 271
 Kameraroboter 271
 Kartesisches Koordinatensystem 195

Ketten 263
Keyboard 90
Klatsch-Schalter 160
Kolben 265
Kollisionskontrolle 79
Kollisionssensor 292
Kompass 167
Kompassmodul 167
Kompassrose 170
Kontrollstruktur 106
Kopfblock 65
Kreis 130
Kreiswerkzeug 130
Künstliche Intelligenz 284
Kuppelstangen 265

L

Laptop 284
Laser-Engraver 16
Laufen 263
Lautsprecher 162
LED 218
LED-Matrix 245
Lenkachse 273
Lichtschwankungen 128
Lichtsensoren 29
Lichtstreuung 246
Lifelong Kindergarten Group 42
Line-Finder 22
Line-Follower-Modus 292
Liste 231
Listenname 234
Lithium-Akku 34
Livestreaming 277, 278
Lochrasterplatine 177
LOW 180

M

MAC-Adresse 54
Mac OSX 44
Magnetkompass 167
mAh 36
Makeblock 7, 9
Makeblock-App 87, 276
Malspuren 202
Manuelle Kontrolle 291
Massachusetts Institute of Technology (MIT) 42
Masse 144
mBlock 44
mBot 7
mBot Ranger 11
mCore 20

Me Auriga 12
MegaPi-Board 14
Melodie 90
Microsoft Azure 283
Mignon AA 1,5 V 33
Mikrofon 159
Mikrofontaste 91
Mikrotaster 175, 187
Minuspol 34, 142
Minute 255
Minutenwert 256
Mittelpunkt 130
Motor 119
Motoranschlussbuchse 31
Motor Controller 30
Motoren-Encoder 274
Multimeter 154
Musiker 90
Mutter 246

N

Nachkommastellen 249
Nachrichtenversand 203
Neue Nachricht 203
Neuer Block 83
Niete 167
Nummernanzeige 93

O

ohmsches Gesetz 143
Ohm (Ω) 143
On-Board-LED 217
On-Board-Spannungsregelung 168
Open-Source-Technologie 9

P

Pairing 38
Pandabär 129
Patch-Kabel 176
Pegel 180
Periodendauer 120, 210
Pfad 260
PIR-Motion-Sensor 294
Plastikniete 189
Platine 176
Pleuelstangen 265
Plotter 16
Pluspol 34, 142
Port 23
Port-Nummer 280
Positionierung 253

Poster 98
 Potenziometer 139, 145
 Pull-down-Widerstand 180
 Pulsdauer 119
 Puls-Weiten-Modulation 119
 Puls-Width-Modulation 210
 PyCharm 296
 PyScripter 296
 PySerial 295
 Python 295
 Python-Modul 295
 Python-Tutorial 302

R

Radketten 263
 Ranger 11
 realtime face 287
 Receiver 71
 Rechtwinkliges Dreieck 196
 RGB 218
 RGB-LED 26
 RGB-LED-Modul 217
 RJ-25-Adapter 189
 RJ-25-Adaptermodul 25
 RJ-25-Buchse 21, 126
 RJ-25-Stecker 21
 Roboterfahrzeug 8

S

ScanLine 200
 Schallamplitude 159
 Schallsensor 164
 Schallsignale 164
 Schieberegler 178
 Schleife 106
 Schleifer 145
 Schrauben 246
 Schwellenwert 112
 SCL 168
 Scratch 9, 42
 SDA 168
 Sensor 100
 Sensoren 2
 Sensorsystem 294
 Servo 189, 212
 Servomotor 119, 193, 212
 Sicherungsmuttern 266
 Signalverlauf 137
 Sinus 197
 Six-legged Robot 264
 Sketch 59
 Skript-Tab 131

Slot 194
 Smart Home 283
 Smartphone 87
 Soundsensor 159
 Spannung 119, 139
 Spannungsteiler 144
 Spannungsversorgung 33, 209
 Sperrholz 246
 Spiegelreflexkamera 272
 Spielen 89
 Sprachbefehle 91
 Spracherkennung 284
 Spracherkennungsalgorithmen 284
 Sprachsteuerung 91
 Spur 100
 STA 51
 Stack 69
 Stange 162
 Stangenantrieb 265
 Stativkopf 272
 Steckbrücken 176
 Steuerknopf 151
 Steuerkonsole 89
 Steuerkreuz 70, 89
 Steuersignal 209
 Steuertasten 198
 Steuerung 65
 Steuerungssoftware 22
 Stift 129
 Stiftdicke 130
 Stiftleisten 25
 Strom 139
 Stunde 255
 Systemkamera 272
 Systemzeit 255

T

Tablet 87
 Taktleitung 168
 Taster 29
 Textdatei 260
 Tondauer 81
 Tonerzeugung 90
 Tragekonstruktionsbauteile 264
 Transmitter 71
 Treiber 43

U

Überwachungskamera 278
 Uhrmacherschraubendreher 297
 Ultimate Robot Kit 13
 Ultraschallsensor 22, 111, 190

Umgebungslicht 102
Umpolung 119
Umschalttaste 130
UND-Verknüpfung 174
Untergrund 100
USB-Verbindung 22

V

Variablenüberlauf 227
Vektor 195
Vibrationsalarm 87
Video 277
Videostream 279
Vorkommastellen 249

W

Webcam 278, 284
Wendekreis 275
Wertebereichsumwandlung 134
Widerstand 139, 175
Widerstand (R) 142
Widerstandsleiter 180
Widerstandswert 156
Windows 44
Winkel 162, 167, 189
Winkelwert 191
Wissenstransfer 9
WLAN-Router 280
ws2812 221

X

x-Achse 127
XY-Plotter 16

Y

y-Achse 127

Z

Zeichenkette 250
Zeichnen und Laufen 89
Zeit 254
Zeitformat 254
Zufallsfunktion 231
Zufallswerte 122