

LEGO[®]-Roboter

bauen, steuern und programmieren
mit Raspberry Pi und Python



Inhaltsverzeichnis

Einleitung	11
Mit dem Buch arbeiten	13
Der LEGO Digital Designer 4.3	15
Teil I Die Hardware für die Roboter	17
<hr/>	
1 LEGO als Grundlage für unsere Roboter	19
1.1 Roboter als Bausatz	19
1.2 LEGO mit Elektronikkomponenten versehen	22
1.2.1 Folgende LEGO-Elektronikkomponenten werden verwendet	22
1.2.2 Folgende Fremdkomponenten werden verwendet	23
2 Der Raspberry Pi stellt sich vor	25
2.1 Der Einplatinencomputer	25
2.2 Die GPIO-Schnittstelle	27
2.3 Den Raspberry Pi konfigurieren	30
2.3.1 Den RPi mit weiterer Hardware versehen	31
2.3.2 Die SD-Karte vorbereiten	31
2.4 Den BrickPi3 anschließen	32
3 Die elektronischen Komponenten	35
3.1 Verwendung der LEGO-Elektronik-Komponenten	35
3.1.1 Der LEGO-Colorsensor	35
3.1.2 Der LEGO-Touchsensor	36
3.1.3 Der LEGO-Infrarotsensor	36
3.1.4 Der Hitechnic-Kompass-Sensor	37
3.1.5 Der Hitechnic-Gyrosensor	37
3.1.6 Der mittlere LEGO-Motor	37
3.1.7 Der LEGO-Motor	38
3.2 Verwendung von Fremdkomponenten	40
3.2.1 Der Fototransistor	41
3.2.2 Die LED	41

3.2.3	Der Touchsensor	41
3.2.4	Der Schallgeber	42
3.2.5	Die Motoren und der Motortreiber	42
3.2.6	Die Kamera	50
3.2.7	Der BrickPi3	51

Teil II Programmieren lernen 53

4	Die Programmiersprache Python	55
4.1	Die Entwicklungsumgebung	55
4.2	Die ersten Schritte	56
4.3	Hallo, ich bin ein Roboter	58
4.4	Editieren und ausführen	60
5	Variablen	63
5.1	Datentypen	63
5.1.1	Zahlen	63
5.1.2	Strings (Zeichenketten)	67
5.1.3	Wahrheitswerte	69
5.2	Datenstrukturen	70
5.2.1	Listen	70
5.2.2	Tupel	72
5.3	Konstanten	72
6	Verzweigungen	73
6.1	Bedingungen	73
6.2	Das if-Statement	74
6.3	Das else-Statement	75
6.4	else-if-Kaskaden	79
6.5	Modulbibliotheken	80
6.6	Experiment: LED schalten	80
7	Schleifen	85
7.1	Das while-Statement	85
7.2	Das for-Statement	88
7.3	Das break-Statement	91
7.4	Das continue-Statement	92
7.5	Experiment: Blinklicht	92
7.6	Experiment: LED dimmen	94

8	Funktionen	97
8.1	Deklaration	97
8.2	Parameter	98
8.3	Rückgabewert	99
8.4	Experiment: Licht erkennen	100
9	Klassen und Objekte	107
9.1	Definition einer Klasse	107
9.2	Methoden einer Klasse	109
9.3	Vererbung	111
9.4	Experiment: Töne erzeugen	112
Teil III Projekte		119
Projekte mit dem BrickPi3 und LEGO-Komponenten		120
10	Wänden und Gegenständen ausweichen	121
10.1	Das LEGO-Modell	122
10.2	Die LED	123
10.3	Der Infrarotsensor	125
10.4	Die Motoren	127
10.5	Wänden und Gegenständen ausweichen	136
11	Himmelsrichtungen erkennen	141
11.1	Das LEGO-Modell	142
11.2	Der Kompass-Sensor	142
11.3	Der Touchsensor	147
11.4	Die übrige Hardware	148
11.5	Himmelsrichtungen erkennen	148
12	Auf dem Tisch bleiben	153
12.1	Das LEGO-Modell	154
12.2	Der Gyrosensor	154
12.3	Die übrige Hardware	157
12.4	Auf dem Tisch bleiben	157
13	Ein Labyrinth lösen mit einem Expertensystem	161
13.1	Das LEGO-Modell	162
13.2	Der Colorsensor	163
13.3	Die übrige Hardware	165

13.4	Das Expertensystem.....	165
13.4.1	Die Regelbasis.....	166
13.4.2	Der Stapelspeicher (Stack).....	168
13.4.3	Der Regelinterpreter.....	170
13.4.4	Die Move Engine.....	172
13.4.5	Die Funktion move_since_wall.....	173
13.4.6	Die Funktion move_back.....	174
13.4.7	Die Funktion rotate.....	175
13.5	Labyrinth.....	176
14	Linienverfolgung mit einem neuronalen Netz.....	185
14.1	Das LEGO-Modell.....	187
14.2	Einführung in neuronale Netze.....	187
14.3	Der Colorsensor als Lichtsensor.....	191
14.4	Die übrige Hardware.....	192
14.5	Das neuronale Netz.....	192
14.6	Linienverfolgung klassisch.....	197
14.7	Linienverfolgung trainieren.....	202
14.8	Linienverfolgung mit neuronalem Netz.....	210
15	Objekte klassifizieren mit einem neuronalen Netz.....	215
15.1	Das LEGO-Modell.....	218
15.2	Benötigte Hardware.....	219
15.3	Kategorisieren lernen.....	219
15.4	Trainingsdaten erzeugen.....	221
15.5	Das Training.....	226
15.6	Objekte klassifizieren.....	231
16	Pappkarten abschießen per Bilderkennung.....	237
16.1	Das LEGO-Modell.....	238
16.2	Die RPi-Kamera.....	240
16.3	Die übrige Hardware.....	243
16.4	Bilderkennung.....	243
16.4.1	Farben identifizieren und definieren.....	244
16.4.2	Objekt im Bild erkennen.....	252
16.4.3	Die eigentliche Bilderkennung.....	256
16.5	Pappkarten abschießen.....	262
17	Joghurtbecher sammeln per Bilderkennung.....	267
17.1	Das LEGO-Modell.....	269

17.2	Benötigte Hardware.	270
17.3	Bildererkennung.	271
17.4	Joghurtbecher sammeln	271
<hr/>		
	Projekte mit elektronischen Fremdkomponenten	287
18	Texte morsen	289
18.1	Das LEGO-Modell	291
18.2	Eine Hilfsplatine basteln.	292
18.2.1	Die Spannungsversorgung	293
18.2.2	Die Spannungsteiler.	293
18.2.3	Das I ² C-Interface	294
18.2.4	Der Taster	295
18.2.5	Die LED.	295
18.2.6	Der Piezo-Schallgeber	297
18.2.7	Der IC MCP3008	300
18.3	Touchsensoren basteln	302
18.4	Morsecode übersetzen und eingeben.	304
18.4.1	Text in Morsecode übersetzen.	304
18.4.2	Morsecode in Text überführen	307
19	Abfahren der »platonischen Flächen«	313
19.1	Das LEGO-Modell	314
19.2	Getriebemotor mit Encoder und Motortreiber	316
19.3	Die »platonischen Flächen«	321
20	Suche des hellsten Orts im Raum	325
20.1	Das LEGO-Modell	326
20.2	Der Fototransistor	326
20.3	Die hellste Lichtquelle des Raumes finden	329
21	Ausblick	333
A	Anhang	335
A.1	Download	335
A.2	Bezugsquellen	336
	Stichwortverzeichnis	338

Einleitung

Die Robotertechnologie wird in unserem alltäglichen Leben immer präsenter. Es gibt Roboter, die unseren Boden kehren, die unseren Rasen mähen oder uns in einem gewissen Umfang bedienen können. In der Industrie werden schon seit längerem Roboter zur Fertigung eingesetzt. Es werden von ihnen dort aber auch Überwachungs- und Serviceleistungen erbracht. Es spannt sich schon jetzt ein weites Feld auf, in dem Roboter eingesetzt werden können.



Abb. 1: Roboter, die beispielsweise in der industriellen Fertigung eingesetzt werden können (mit freundlicher Genehmigung der Firma KUKA AG, Augsburg)

Und die Einsatzmöglichkeiten werden immer ausgefeilter. Es befinden sich Roboter in der Entwicklung, die kranke und alte Menschen versorgen können, oder Roboter, die uns eine echte Küchenhilfe sein können.

Außerdem wäre die Erforschung anderer Planeten, wie beispielsweise des Mars, ohne Roboter nicht denkbar. Aktuell befindet sich der NASA-Roboter »Perseverance« (Ausdauer) auf dem Mars, um diesen nach Lebensspuren zu erkunden.

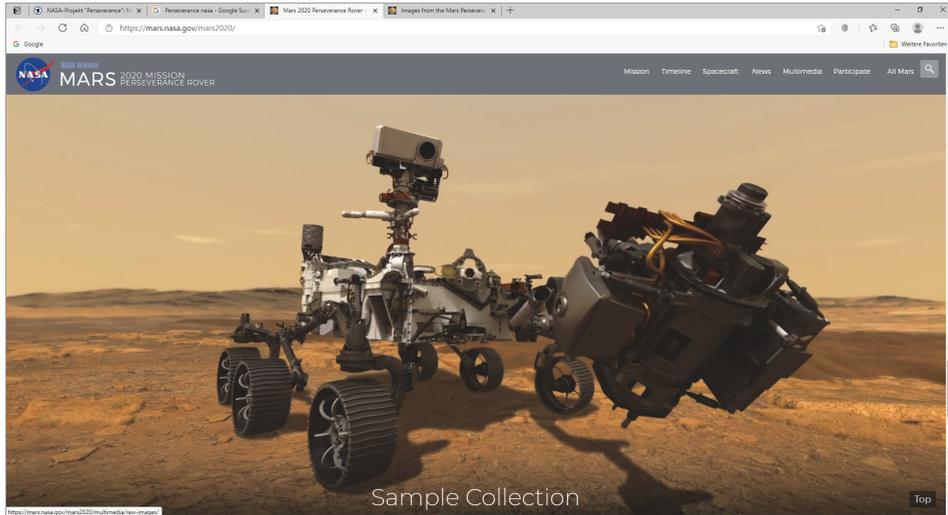


Abb. 2: Der Roboter Perseverance (Quelle: NASA-Webseite)

Wegen der großen Entfernung zwischen Mars und Erde sind die Signale, die der Roboter und das Kontrollzentrum der NASA austauschen, etwa drei Minuten unterwegs (der Mars ist zurzeit etwa 56 Mio. km von der Erde entfernt und die Lichtgeschwindigkeit beträgt etwa 300.000 km/sec). Diese Zeit ist zu lang, als dass die NASA den Roboter in Echtzeit steuern könnte. Es ist nur möglich, dem Roboter generelle Befehle zu senden, die er dann autark ausführt.

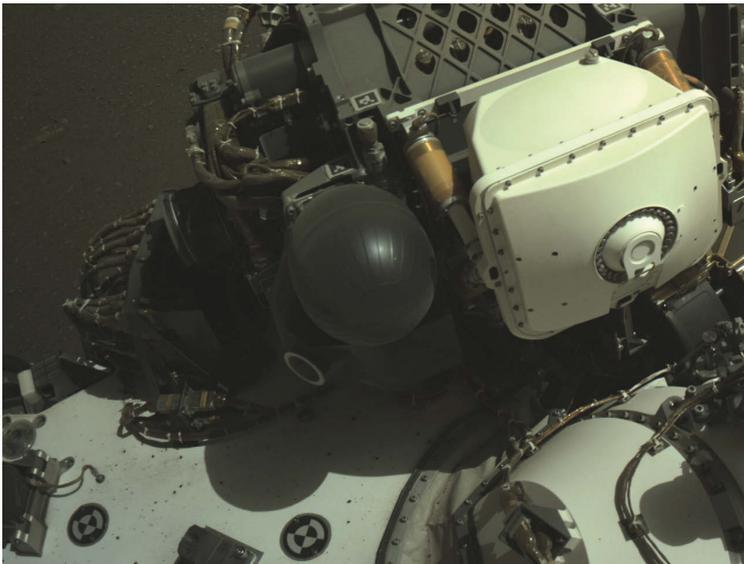


Abb. 3: Perseverance fotografiert sich selbst. (Quelle: NASA: NASA/JPL-Caltech)

Der Roboter muss, um seine Aufgaben zu erfüllen, eigenständig Entscheidungen treffen können. Dazu ist eine gewisse Intelligenz notwendig.



Abb. 4: Perseverance fotografiert die Marslandschaft. (Quelle: NASA: NASA/JPL-Caltech/ASU)

Wie Roboter so agieren können, erfahren Sie in diesem Buch. Sie lernen Grundfertigkeiten von Robotern kennen, die für verschiedene Aufgaben angewendet werden können. Vielleicht verfügt ja ein NASA-Roboter auch über eine von mir vorgestellte Grundfertigkeit.

Sie können mit diesem Buch alles praktisch erkunden, indem Sie meine Roboter mit LEGO-Teilen nachbauen, um so einen persönlichen Eindruck von der Leistungsfähigkeit auch schon kleinerer Roboter zu erlangen. LEGO-Teile haben dabei den Vorteil, dass Sie einen aufgebauten Roboter später wieder zerlegen und dann einen neuen bauen können.

Wir beschäftigen uns zunächst mit der generellen Steuerung von Robotern, später aber auch mit Themen der »Künstliche Intelligenz Forschung«. Sie werden neuronale Netze und ein Expertensystem einsetzen, um Roboter zu steuern. Solche Roboter besitzen schon eine gewisse Grundintelligenz und treffen intelligente Entscheidungen auf der Basis eigener Steuerungen.

Mit dem Buch arbeiten

Dieses Buch ist in drei Teile geteilt. Als Erstes stelle ich die Hardware für die Roboter vor. Danach gebe ich Ihnen eine Einführung in Python, die von mir zur Robotersteuerung verwendete Programmiersprache. Im dritten Teil stelle ich Ihnen elf Projekte vor, bei denen Sie selbst Roboter bauen und programmieren. Sie lernen dabei sukzessive die Roboter-Programmierung mit der Programmiersprache

Python, aber auch Konzepte kennen, die zur Steuerung von Robotern herangezogen werden.

Ich verwende zum Bau des Chassis der Roboter Bauteile aus dem LEGO-EV3-Baukasten. Sie können natürlich auch die Bauteile aus eigenen Beständen verwenden oder sich gewisse Teile, die Ihnen fehlen, von LEGO bestellen (Einzelteillisten befinden sich jeweils im Download zum Buch auf der Webseite www.mitp.de/0310). Ein Teil der Roboter wird mit den zum EV3-Baukasten gehörenden Motoren und Sensoren ausgestattet. Ich verwende aber auch LEGO-Sensoren, die dazugekauft werden können. Allgemein können Sie sich sämtliche LEGO-Komponenten separat beschaffen (richten Sie sich dazu an den oben genannten Teilelisten aus). Ein EV3-Baukasten ist natürlich nicht unbedingt erforderlich, Sie hätten aber so die Bauteile, bis auf die Hitechnic-Sensoren, komplett vorliegen.

Drei Projekte (Kapitel 18 bis 20) erstelle ich mit Fremdkomponenten im Sinne von allgemeinen Technikkomponenten, die nicht von LEGO sind. Es handelt sich dabei um Getriebemotoren und Sensoren wie einen Fototransistor oder Kommunikationsmittel wie eine LED. Wenn Sie Interesse haben, diese Projekte mit Fremdkomponenten zu erarbeiten, werden Sie in diesem Rahmen mit dem Lötcolben basteln.

Als Computer, der die Roboter steuert, verwende ich den *Raspberry Pi*-Computer. Dabei handelt es sich um einen etwa scheckkartengroßen Einplatinencomputer, der durch das Betriebssystem Linux gesteuert wird.

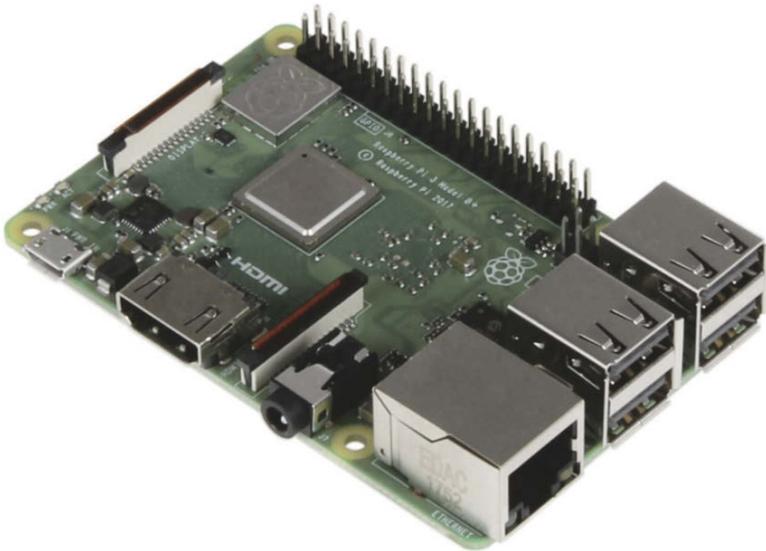


Abb. 5: Der Raspberry Pi

Er verfügt, wie Sie noch sehen werden, über eine Hardwareschnittstelle in Form von 40 kleinen Pins, an die die Roboterhardware zur Steuerung angeschlossen werden kann.

Zusätzlich verwende ich den *BrickPi3*. Das ist eine Hardwareplatine, die auf den Raspberry Pi gesteckt wird, um die LEGO-Hardware zu steuern. Der Verbund von Raspberry Pi und BrickPi3 erledigt dabei die Aufgaben, die der EV3-Baustein im LEGO-Baukasten erledigen würde. Der Vorteil bei der Verwendung des Raspberry Pi besteht darin, dass Sie zum einen eine vollwertige, momentan sehr aktuelle Programmiersprache (Python) einsetzen können, um Ihre Roboter zu steuern, der Einsatz neuronaler Netze wäre anders gar nicht denkbar. Zum anderen lernen Sie einen sehr interessanten und preiswerten Computer kennen, den Sie auch für andere Aufgaben verwenden können. Er verfügt über ein Betriebssystem, das *Raspberry Pi OS*, das eine grafische Benutzeroberfläche hat, sodass sich auch Apple- und Windows-Benutzer sehr schnell heimisch fühlen werden.

Der LEGO Digital Designer 4.3

Ich habe die Bauanleitungen zu den einzelnen Robotern mit dem Tool »LEGO Digital Designer 4.3« (<https://www.lego.com/de-de/ldd>) erstellt und diese in entsprechenden Dateien gespeichert, die dem Download zum Buch unter www.mitp.de/0310 beiliegen (die Dateien haben die Endung `.lxf`). Auf diese Weise können Sie alle Roboter-Modelle im Buch ganz einfach nachbauen.

Wenn Sie, nachdem Sie den LEGO Digital Designer installiert haben, auf die Datei doppelklicken, wird das Programm gestartet und das Modell angezeigt. Da LEGO dieses Tool per Internetserver nicht mehr unterstützt, wird eine Fehlermeldung angezeigt, die Sie mit einem Klick auf den OK-Button quittieren.

In der oberen rechten Ecke der Buttonleiste befindet sich der Button , mit dem Sie die Bauanleitung generieren können. Danach können Sie mit den Pfeilbuttons zwischen den einzelnen Steinen hin- und herschalten und das Modell erstellen. Da es für den BrickPi3 natürlich kein Symbol im Programm gibt, habe ich den Platz, den er einnimmt, im Modell immer frei gelassen.

Nun sind Sie sicherlich schon ganz gespannt auf die Themen, die ich Ihnen in meinem Buch zeigen werde, daher sollten wir sofort in die Behandlung der Technologie einsteigen. Ich wünsche Ihnen viel Vergnügen bei der Lektüre meines Buches und beim Bau und der Programmierung der vorgestellten Roboter.

Thomas Kaffka, im Juli 2021

Teil I

Die Hardware für die Roboter

In diesem Teil:

- **Kapitel 1**
LEGO als Grundlage für unsere Roboter 19
- **Kapitel 2**
Der Raspberry Pi stellt sich vor 25
- **Kapitel 3**
Die elektronischen Komponenten 35

LEGO als Grundlage für unsere Roboter

Es ist eine gute Idee, LEGO-Komponenten zum Roboterbau zu verwenden. Ein aufgebauter Roboter kann immer wieder zerlegt werden und ein neuer erfunden und gebaut werden. Der eigenen Fantasie sind dabei grundsätzlich keine Grenzen gesetzt. Daher verwende ich in meinem Buch LEGO-Komponenten, um ein Roboterchassis zu erstellen. Ich lege dabei die Bauteile des LEGO-EV3-Baukastens zugrunde.

Leser, die diesen Baukasten nicht besitzen, aber über LEGO-Bausteine verfügen oder gewillt sind, sich diese anzuschaffen, können auch ganz entspannt sein. Denn ich gebe bei jedem Roboter eine Teileliste über die LEGO- oder Fremdkomponenten an, die ich für den jeweiligen Roboter verwende. Diese Teilelisten sind im Download zum Buch (www.mitp.de/0310) sowie in den einzelnen Kapiteln zu finden.

1.1 Roboter als Bausatz

Sie könnten sich natürlich, wenn Sie sich mit der Robotik beschäftigen möchten, auch einen Bausatz zu einem fertigen Roboter anschaffen, statt eigene Roboter aus LEGO zu bauen. Diese Roboter haben aber den Nachteil, dass es sich um fertige Maschinen handelt, die in ihren Freiheitsgraden eben deshalb beschränkt sind. Wenn man ihre Motoren und sonstigen Aktoren programmiert hat, ist es nicht mehr interessant, sich mit ihnen weiter zu beschäftigen.

Dazu hier einige Beispiele von Robotern, die ich mir unter anderem angeschafft habe (Abbildung 1.1 bis Abbildung 1.3).

Der YETI (Abbildung 1.1) ist ein einfacher Roboter, der über zwei Beine verfügt. Er ist mechanisch so aufgebaut, dass er mit zwei Servomotoren gesteuert werden kann. Um zu gehen, verlagert er sein Gewicht auf eines der beiden Beine und bewegt das andere vor. Danach wird das gegenüberliegende Bein belastet. Ich habe für den Roboter zusätzlich ein Vier-Segment-Display angeschafft und eingebaut. Damit kann er Meldungen oder seinen Status mitteilen.



Abb. 1.1: Der YETI von AREXX Engineering

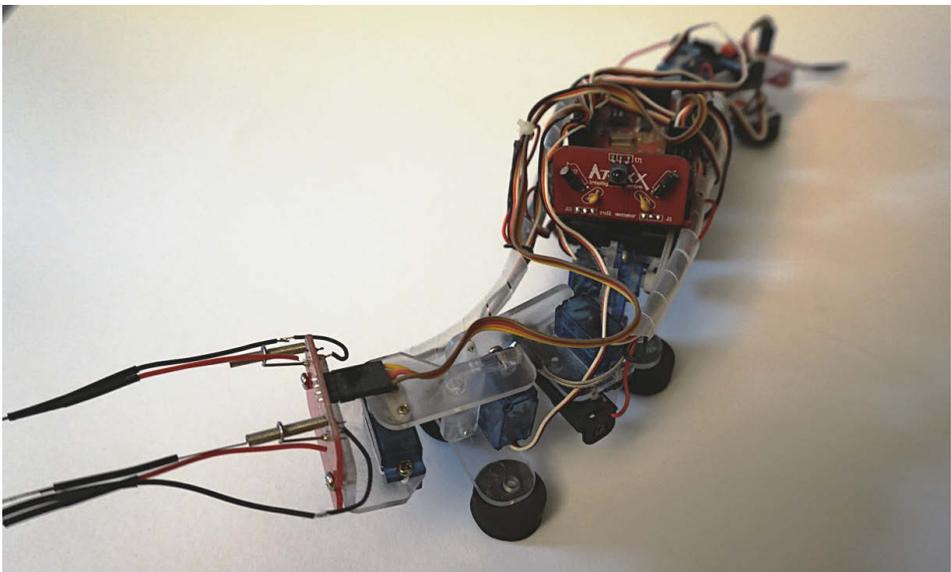


Abb. 1.2: Die Caterpillar von AREXX Engineering

Die Caterpillar ist ein wurmförmiger Roboter. Sein Körper ist in verschiedene Segmente geteilt, die sich durch Servomotoren jeweils auf und ab oder links und rechts bewegen lassen. Durch eine koordinierte Ansteuerung der Motoren ist der

Roboter in der Lage, sich fortzubewegen. Seine Sensoren sind zwei Antennen vorne und eine Antenne hinten sowie ein Rollsensor.

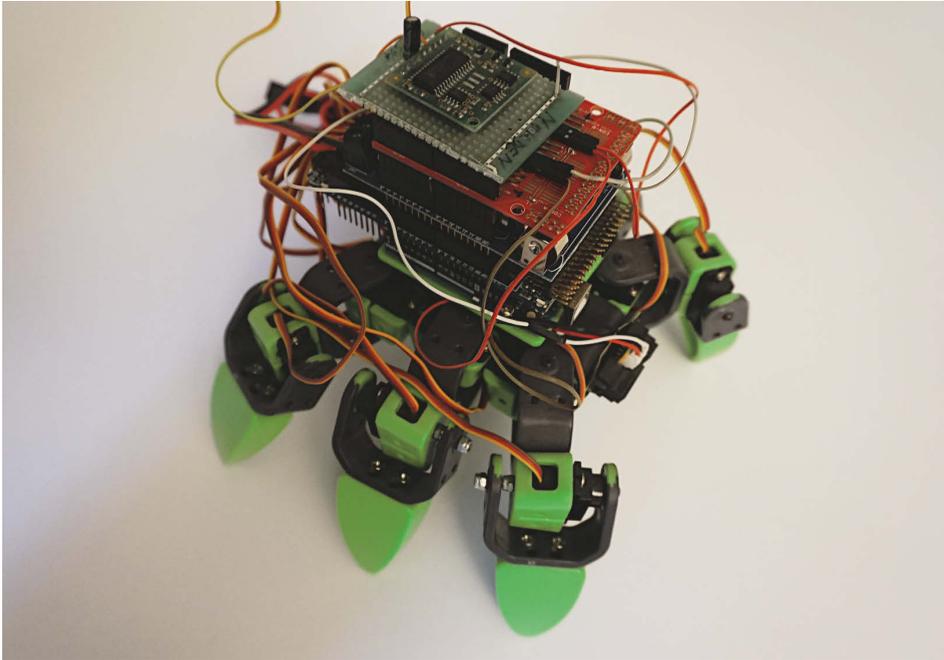


Abb. 1.3: Der ALLBOT von Velleman

Der ALLBOT ist ein sechsbeiniger Roboter, der entsprechend anspruchsvoll programmtechnisch gesteuert wird. Die sechs Beine müssen sich ja koordiniert bewegen, damit er sich vorwärtsbewegt. Dem Roboter habe ich einen Infrarotsensor (vorne am Chassis) spendiert, damit er Abstände zu einem Hindernis messen kann, sowie einen Kompass-Sensor (oberste Platine), der ihn befähigt, die Richtung zu bestimmen, in der er sich bewegt.

Natürlich kann man mit den vorgestellten Robotern vortrefflich experimentieren und diese auch mit zusätzlichen elektronischen Komponenten erweitern. Aber irgendwann hat man sie ausgereizt und dann stehen sie nur noch herum und sehen schön aus. Das ist schade.

Daher gehe ich in diesem Buch einen anderen Weg und verwende LEGO-Komponenten sowie elektronische Fremdkomponenten, um Roboter zu bauen. LEGO-Roboter haben den Vorteil, dass man sie, nachdem man sich ausführlich mit ihnen beschäftigt hat, zerlegen und dann einen neuen Roboter bauen kann. Wenn man LEGO für seine Roboter verwendet, wird dieses interessante Hobby niemals langweilig.

1.2 LEGO mit Elektronikkomponenten versehen

Das LEGO-Mindstorms-System bietet Computer, Motoren und Sensoren für den Bau von Robotern aus LEGO-Technikteilen an. Diese originalen LEGO-Bauteile sind im Vergleich zu Fremdkomponenten sehr teuer. Daher können sowohl die Originalteile als auch – als Alternative – elektronische Fremdkomponenten verwendet werden, wie Sie in den letzten drei Kapiteln zu den Roboterprojekten sehen können.

1.2.1 Folgende LEGO-Elektronikkomponenten werden verwendet

Für die Modelle im Buch verwende ich die folgenden LEGO-Komponenten.



Abb. 1.4: Der LEGO-Colorsensor



Abb. 1.8: Der mittlere Motor von LEGO EV3



Abb. 1.5: Der LEGO-Infrarotsensor



Abb. 1.9: Der LEGO-EV3-Motor



Abb. 1.6: Der LEGO-Touchsensor



Abb. 1.10: Der Hitechnic-Gyrosensor



Abb. 1.7: Der Hitechnic-Kompass-Sensor

1.2.2 Folgende Fremdkomponenten werden verwendet

Ein echter Elektronik-Bastler hat keine Angst vor Komponenten von Drittanbietern. Diese Bauteile haben den enormen Vorteil, dass sie teilweise nur einen Bruchteil der originalen LEGO-Komponenten kosten. Man muss allerdings bereit sein, auch mal etwas zu löten, da beispielsweise der Lichtsensor zusammengebaut werden muss. Aber keine Bange, das kriegen Sie hin.

Ich verwende beim elektronischen Basteln einen 16-Watt-LötKolben. Seine Leistung darf nicht zu hoch sein, damit die elektronischen Bauteile keinen Schaden nehmen. Dazu benötigt man noch eine Spule Lötzinn und wer es ganz komfortabel haben möchte, der legt sich noch eine Abisolierzange zu, fertig.

Als Fremdkomponenten setze ich die folgenden Bauteile ein.



Abb. 1.11: Der Lichtsensor (Fototransistor)



Abb. 1.14: Getriebemotor plus Reifen und Encoder



Abb. 1.12: Der Touchsensor (Mikroschalter)



Abb. 1.15: Die LED



Abb. 1.13: Die Raspberry-Pi-Kamera



Abb. 1.16: Der Piezo-Schallgeber

Teil III

Projekte

Sie haben jetzt einen Überblick über den Raspberry Pi sowie den BrickPi3 erhalten. Sie haben bereits etwas programmiert und ein paar Hardwarekomponenten eingesetzt. Das war sicherlich schon spannend, aber es geht natürlich noch mehr.

In den jetzt folgenden Kapiteln stelle ich jeweils einen Roboter vor. Ich gehe auf das Basteln des Roboters aus LEGO-Komponenten und in den letzten drei Kapiteln auf die Verwendung von Fremdkomponenten ein. Aber die Roboter werden natürlich auch programmiert.

Sie erfahren, wie Sie einen Roboter mit Motoren bewegen und Sie lernen die verschiedenen Sensoren kennen, mit denen ein Roboter gesteuert werden kann. Beispielsweise, wie ein Roboter mithilfe einer Kamera »sehen« kann.

Dabei lernen Sie die grundsätzliche Steuerung eines Roboters kennen, aber auch anspruchsvollere Anwendungen. Ich stelle Ihnen neuronale Netze vor, mit denen Roboter intelligent gesteuert werden können. Weiterhin arbeiten Sie sich in das Prinzip eines Expertensystems ein, das auch ein Instrument der Künstlichen Intelligenz ist.

Das alles soll Ihnen dieses spannende Gebiet näherbringen und so lernen Sie ganz nebenbei auch noch das Programmieren mit Python.

Projekte mit dem BrickPi3 und LEGO- Komponenten

In diesem Teil:

- **Kapitel 10**
Wänden und Gegenständen ausweichen 121
- **Kapitel 11**
Himmelsrichtungen erkennen 141
- **Kapitel 12**
Auf dem Tisch bleiben 153
- **Kapitel 13**
Ein Labyrinth lösen mit einem
Expertensystem 161
- **Kapitel 14**
Linienverfolgung mit einem neuronalen Netz. . . . 185
- **Kapitel 15**
Objekte klassifizieren mit einem
neuronalen Netz. 215
- **Kapitel 16**
Pappkarten abschießen per Bilderkennung. 237
- **Kapitel 17**
Joghurtbecher sammeln per Bilderkennung 267

Wänden und Gegenständen ausweichen

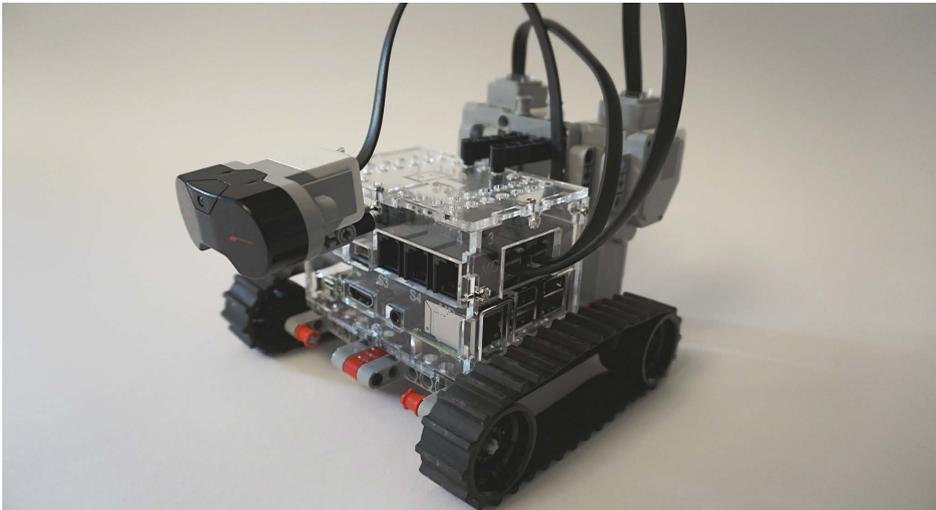


Abb. 10.1: Wänden und Gegenständen ausweichen

Der in diesem Kapitel vorgestellte Roboter hat die Aufgabe, Wänden und Gegenständen auszuweichen. Er verfügt über zwei Motoren, die ihn antreiben. Diese setzen dazu zwei Raupenriemen in Bewegung. Vorne am Roboter ist der Infrarotsensor befestigt, um den Abstand zu den Wänden und Gegenständen zu messen.

Starten Sie das Programm dieses Kapitels zunächst mit Tastatur und Bildschirm. Ziehen Sie dann die Kabel vom Raspberry Pi ab. Das sind: das HDMI-Kabel, das Tastaturkabel, das Mauskabel sowie das Kabel des Netzteils. Der Batterieschalter rechts neben dem Batterieanschluss muss dabei eingeschaltet und das Batteriepack angeschlossen sein. Wenn Sie den Roboter auf den Boden setzen und danach Ihre Hand weniger als fünf Zentimeter vor den Infrarotsensor halten, wird der Roboter eingeschaltet und fährt los. Trifft er auf eine Wand, stoppt er, setzt zurück und wendet ein wenig. Danach fährt er weiter geradeaus.

Das ist genau das Prinzip, nach dem die modernen Staubsaugerroboter funktionieren. Es fehlt nur noch einen Saugmotor unter dem Roboterchassis und fertig ist der Staubsaugerroboter.

Sie können den Roboter ausschalten, indem Sie Ihre Hand wieder weniger als fünf Zentimeter entfernt vor den Infrarotsensor halten. Danach können Sie die erwähnten Kabel wieder einstecken.

Der Roboter in Aktion



Sie können ein Video mit dem Roboter im Einsatz sehen, wenn Sie dem QR-Code folgen.

10.1 Das LEGO-Modell

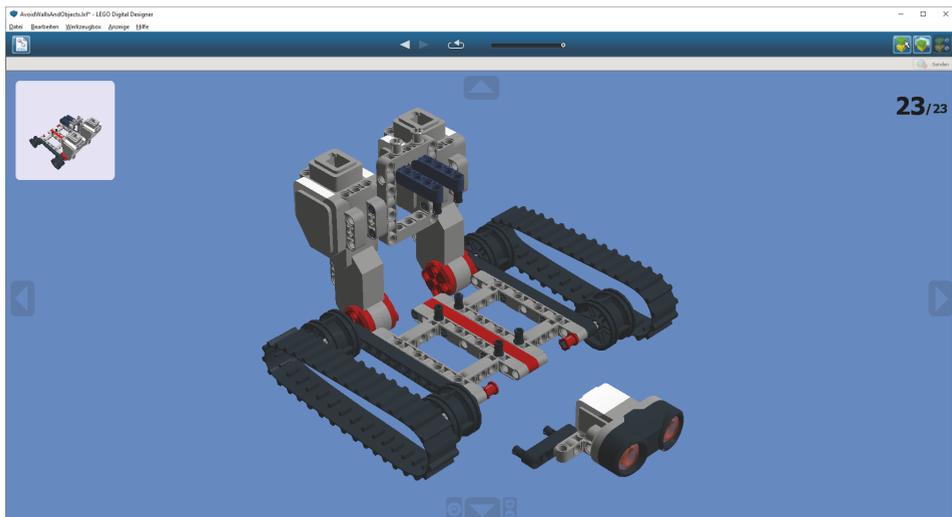


Abb. 10.2: Die Bauanleitung

Die Bauanleitung des Robotermodells rufen Sie mit dem LEGO Digital Designer auf. Wie Sie das Programm bedienen, habe ich in der Einleitung erläutert. Zum aktuellen Roboter gehört die Datei `AvoidWallsAndObjects.lxf`.

Den BrickPi3 platzieren Sie so, wie in Abbildung 10.1 dargestellt wird. Vorne am Gehäuse des BrickPi3 wird der Infrarotsensor von LEGO angebracht. Verbinden Sie ihn mit dem Port S1. Verbinden Sie den in Fahrtrichtung rechten Motor mit dem Port MB und den in Fahrtrichtung linken Motor mit MC. Die Maus und die

Tastatur verbinden Sie besser mit den oberen beiden USB-Ports, damit die Kabel nicht auf dem Riemen scheuern.

10.2 Die LED

Auf der Platine des BrickPi3 befindet sich links neben dem Batterieanschluss eine gelbe LED, die auch von den Programmen angesteuert werden kann. Bei unserem Roboter benutzen wir sie, um seinen Betriebsmodus anzuzeigen.

Ist der Roboter »ready to go« oder fährt nach vorne, blinkt die LED in einem Abstand von einer Sekunde. Fährt der Roboter rückwärts, blinkt sie mit einem schnellen und einem langsamen Blink, gewissermaßen als Warnung.

Zur Unterstützung der Steuerung der LED habe ich die Klasse Led entwickelt (siehe Listing 10.1). Diese wurde im Sinne der objektorientierten Programmierung (siehe Kapitel 9) erstellt.

```
#!/usr/bin/python
#####
# class led                                #
# file: Led.py                             #
# Author: Th. Kaffka, Cologne, Germany #
# Date: 02.11.2020                         #
#####

import brickpi3                            # ❶
import time
from threading import Thread

BP = brickpi3.BrickPi3()                   # ❷
ok = True                                  # ❸

class Led:

    def __init__(self):                     # ❹
        pass

    def __del__(self):                       # ❺
        BP.set_led(0)

    def ledOn(self):                         # ❻
        BP.set_led(100)
```

```

def ledOff(self):                                # 7
    BP.set_led(0)

def blinkOn(self, time1, time2):                # 8
    self.t = Thread(target=self.doBlink, args=(time1,time2,))
    self.t.setDaemon(True)
    self.t.start()

def blinkOff(self):                              # 9
    global ok
    ok = False

def doBlink(self, time1, time2):                # 10
    global ok
    ok = True
    while ok:                                    # 11
        self.ledOn()
        time.sleep(time1)
        self.ledOff()
        time.sleep(time1)
        self.ledOn()
        time.sleep(time2)
        self.ledOff()
        time.sleep(time2)

```

Listing 10.1: LED-Steuerung (Led.py)

- ❶ Im Programm werden zunächst die nötigen Modulbibliotheken importiert. Neben der Bibliothek für den BrickPi3 und der Zeitbibliothek für die Zeitsteuerung wird die Bibliothek `threading` importiert. Diese dient dazu, das Blinken in einem parallelen Programmstrang zu verarbeiten. Das ist besser, da dann das Blinken unabhängig vom übrigen Programm verarbeitet wird und die LED gleichmäßig blinkt, während der Roboter gleichzeitig andere Aktionen durchführen kann.

Hinweis

Die Zeitsteuerung, die wir hier verwenden, ist das Statement `time.sleep()`. Das bewirkt, dass das Programm eine gewisse, angegebene Zeit anhält und danach weitermacht.

- 2 Es wird ein Objekt in der Variablen BP erzeugt, das zur Steuerung des BrickPi3 dient.
- 3 Dann wird die globale Variable ok auf True gesetzt.
- 4 Der Konstruktor wird nur zu dokumentarischen Zwecken eingefügt. Ich will Ihnen damit zeigen, dass der Konstruktor existiert. Er verarbeitet hier aber nichts, daher das Statement pass.
- 5 Im Destruktor wird die LED auf den Wert 0 (nicht leuchten) gesetzt. Dieser wird aufgerufen, wenn das Objekt gelöscht oder entfernt wird.
- 6 In der Methode ledOn wird die LED auf 100% Leuchten gesetzt. Man kann der LED auch Werte zwischen 0 und 100 zuweisen, um sie beispielsweise schwächer leuchten zu lassen.
- 7 Die Methode ledOff setzt die LED auf 0% Leuchten. Die LED ist dann aus.
- 8 Die Methode blinkOn erhält zwei Parameter. Es handelt sich um zwei das Blinken steuernde Zeitparameter. Haben sie dieselben Werte, ist später das Blinken gleich lang. Sind deren Werte unterschiedlich, beispielsweise 1 und 0.1, dann blinkt die LED mit unterschiedlicher Geschwindigkeit, einmal lang, einmal kurz. Diese Methode startet als paralleles Programm (Thread) die Blinkmethode doBlink und übergibt ihr beide Parameter. Ich habe die Verarbeitung des Blinkens in einen Thread verpackt, damit das Hauptprogramm von dieser Steuerung entlastet wird und das Blinken immer mit denselben Zeiten stattfindet.
- 9 Die Methode blinkOff setzt die globale Variable ok auf False und das Blinken hört auf.
- 10 doBlink ist die Methode, die im Thread (also parallel) verarbeitet wird. Sie setzt zunächst die globale Variable ok auf True.
- 11 In einer Schleife, die ausgeführt wird, solange ok True ist, wird das Blinken mit den zwei Zeitparametern durchgeführt.

10.3 Der Infrarotsensor



Abb. 10.3: Der Infrarotsensor

Der Infrarotsensor (IR) hat die Aufgabe, Infrarotsignale zu erkennen. Wir benutzen ihn bei diesem Roboter, um den Abstand zu einem Gegenstand oder einer Wand zu bestimmen. Seine vom BrickPi3 verarbeiteten Werte liegen zwischen 100 (weit entfernt) und 0 (sehr nah). Er kann Gegenstände in einer Entfernung von bis zu 70 cm erkennen.

Auch für diesen Sensor habe ich eine Klasse erstellt, um ihn komfortabler handhaben zu können.

```
#!/usr/bin/python
#####
# class lego infrared sensor          #
# file: LegoInfraRedSensor.py       #
# Author: Th. Kaffka, Cologne, Germany #
# Date: 21.10.2020                  #
#####

import brickpi3                      # ❶
import time

BP = brickpi3.BrickPi3()

class LegoInfraRedSensor:

    def __init__(self, port):         # ❷
        self.port = port
        BP.set_sensor_type(port, BP.SENSOR_TYPE.EV3_INFRARED_PROXIMITY)

    def __del__(self):
        pass

    def getPort(self):               # ❸
        return self.port

    def getDistance(self):           # ❹
        return BP.get_sensor(self.port)
```

Listing 10.2: IR-Sensor-Steuerung (LegoInfraRedSensor.py)

- ❶ Zuerst werden die Modulbibliotheken importiert und danach die globale Variable BP erstellt, in der sich das Objekt zur Steuerung des BrickPi3 befindet.
- ❷ In dem Konstruktor wird der Sensor mit seinem Port, der als Parameter übergeben wird, assoziiert. Dabei wird auch der Sensortyp definiert.
- ❸ Mit der getPort-Methode kann später auf die Portnummer zugegriffen werden.
- ❹ Mit dieser Methode wird die aktuelle Distanz zwischen dem IR-Sensor und dem Gegenstand oder der Wand ermittelt.

Stichwortverzeichnis

`__del__` 109
`__init__` 108

A

Abbruchbedingung 85
`abs()` 97
Abschusseinrichtung 239
Absolutwert 97
Abstandsmessung 125, 219
Acrylgehäuse 32
ADC 30, 100, 301
Addition 66
Akustisches Signal 297
ALLBOT 21
Analog-Digital-Wandler 30, 100, 301
Analoger Eingangsport 30
Analoger Sensor 101
Anführungszeichen 56, 67
Animal (Klass) 110
Anode 81
Arbeiten
 iteratives 60
Argument 90
Attribut 108
Audioausgang 29
Ausgabe-Pin 83
Auskommentieren 59
Ausnahmesituation 94
Äußeren Schleife 90

B

Backpropagation-Netz 197
Ball
 verfolgen 333
Basis 103
Batterieanschluss 121
Batteriepack 27, 121
Batterieschalter 121
Bedingung 73, 85
 abfragen 76
 auswerten 74
Beispieldaten 188
 erstellen 220

Betriebssystem
 Raspberry Pi OS 25
 Raspbian 25
Bezugsquelle 41
Bibliothek 80
Bildererkennung 237, 243, 252, 271
 Klasse 256
Bilderkennungssoftware 245
Bildschirm
 Auflösung einstellen 33
Bildverarbeitung 244
Binär 63
Binärcode 108
Binärdarstellung 64, 107
Blinklicht (Experiment) 92
BOARD 83, 94, 297
break 91, 132
BrickPi3 15, 25, 27, 29, 51, 80, 142
 anschließen 32
 Bibliotheken 80
 Ports 34
 Symbol 187
Buchhaltungsprogramm 76

C

Caterpillar 20
Chassis 287
class 108
Codeschnipsel 35
colorama 77
Colorsensor 35, 161, 163, 185, 191
Compiler 55
continue 92
Control 250

D

Darstellung
 Zahlen 64
Daten
 fehlerhafte bei neuronalen Netzen 220
Datentyp 63, 98
 Listen 70
def 97

Definition 97
 Deklaration 97
 Desktop 33
 Destruktor 109, 125
 Dexter Industries 25, 32, 142
 Display 216
 Division 66
 ganzahlige 64
 normale 64
 Divisionsrest 131
 ganzahliger 66
 Dodekaeder 313
 Doppelpunkt 74, 97
 Doppeltes Gleichheitszeichen 73

E

Editor 56
 Eingangsport 30, 50
 Einmaleins 85, 89
 Einrückung 74
 Element 70, 188
 Listen 71
 elif 79
 else 75
 else-if-Kaskade 79, 160, 174
 Emitter 102
 Encoder 23, 43
 Belegung 44
 Beschaltung 48
 magnetische 294
 magnetische Encoder-Scheiben 43
 optischer 43
 Encoder-Ereignis 50
 Encoder-Position 131
 Encoder-Scheibe 43
 Encoder-Stellung 131
 Encoder-Steuerung 131
 Encoder-Tick 131
 end= 90
 Endlosschleife 62, 86, 91, 94, 105
 EV3 14, 19
 except 94
 Exception 94
 Experiment
 Blinklicht 92
 LED dimmen 94
 LED schalten 80
 Licht erkennen 100
 Töne erzeugen 112
 Tonfolge 117
 Expertensystem 161, 165
 Labyrinth 177
 Expertenwissen 165

Exponent 64
 Exponentialdarstellung 64

F

Fahrzeugsteuerung 127, 133
 False 69
 Farbdefinition
 Lichtverhältnis 252
 Farbe 192
 erkennen 164
 Objekt 244
 Farbkanal 163
 Farbton 245
 Fehler 73, 94
 semantischer 62
 Fehlerart 62
 Fehlerbehandlung 94
 Fehlererklärung 62
 Filesystem 33
 finally 94
 First in, First out 168
 Flachbandkabel 43
 Fläche
 platonische 313
 Flag 64
 for 88, 91
 Foto 237
 auswerten 244
 Klasse 242
 Fotosensor 326
 Fototransistor 41, 100, 102, 326
 Fremdkomponente 14, 22, 23, 25, 29, 35, 40
 Frequenz 117
 Funktion 97, 98, 100, 105
 Name 97
 Variablenverwendung 99
 Funktionsergebnis 99

G

Ganzzahlige Division 64, 66
 Ganzzahliger Divisionsrest 66
 Geschwindigkeit 136
 Motor 151
 Gewichtsmatrix 188
 Gleich 74
 Gleichheit
 prüfen 73
 Gleichheitszeichen
 doppeltes 73
 Gleitkommadarstellung 63
 Global 100
 GND 29

goal_stack 169
 GPIO-Nummern 30
 GPIO-Schnittstelle 27, 80, 93, 116
 PWM 95
 zurücksetzen 94
 Größer 74
 Größer oder gleich 74
 GUI 26
 Gyrosensor 37, 153, 154

H

Halbleiterbaustein 102
 Handlungsanweisung 184
 Hardware-Encoder 131
 HDMI-Kabel 121
 HDMI-Monitor 25, 31
 Helligkeitssteuerung 95
 Heuristik 166
 Hexadezimal 63
 Hexadezimale Darstellung 64
 Hidden-Schicht 188
 Hilfsplatine
 basteln 292
 Himmelsrichtung 141
 abfragen 143
 Himmelsrichtungen 37
 Hindernis 147
 Hitechnic-Gyrosensor (Klasse) 156
 Hitechnic-Kompass-Sensor 22
 Horse (Klasse) 111, 112

I

I2C 143
 I2C-Interface 294
 I2C-Schnittstelle 28, 30
 IC 28, 301
 IDE 55
 if 74, 85, 91
 Ikosaeder 313
 Index 68, 70
 Infrarotsensor 21, 36, 121, 125
 Klasse 137
 Scannen 219
 Initialisieren 83, 108
 Initialzustand 184
 Innere Schleife 90
 Input-Schicht 188
 Instanz 108
 Instanziierung 107
 Interpreter 55, 56, 74
 Interrupt Exception 94
 IR-Sensor 36

is_moving 175
 Iteratives Arbeiten 60
 IZ 184

J

Joghurtbecher
 einsammeln 267

K

Kabel 121
 Kalibrieren 143, 145
 Kalibrierungsprozess 143
 Kamera 23, 27, 50, 237, 241
 aktivieren 241
 installieren 241
 Kanone 237
 Kategorisierungsprogramm 227
 Kathode 81
 Keyboard-Interrupt 40
 Kinderklasse 111
 Klasse 107
 Bildererkennung 256
 Fotos 242
 Infrarotsensor 137
 Kinderklassen 111
 neuronales Netz 192
 Sensortyp 191
 Vaterklassen 111
 Klassendefinition 108, 111
 Klassenname 108
 Klassifikation
 Ergebnis 219
 Klassifizierung Objekte 215
 Klaue 267
 einbauen 270
 Kleinbuchstaben 67
 Kleiner 74
 Kleiner oder gleich 74
 Kollektion 88
 Kollektor 102
 Kommentar 58
 Kompass-Klasse 146
 Kompassposition 146
 Kompass-Sensor 21, 37, 141, 145
 Komplexe Zahl 63
 Konstante 72
 Konstruktor 108, 111
 Körper
 platonischer 313
 Künstliche Intelligenz 13, 161

L

Labyrinth 161
 Ausgang finden 176
 Länge
 Listen 71
 Strings 68
 LAN-Kabel 25
 Last in, First out 168
 Laufband 215, 219
 Laufvariable 88
 Lautsprecher 42, 297
 LED 23, 41, 81, 92, 123, 296
 Steuerung, Programm 296
 Led (Klasse) 123
 LED dimmen (Experiment) 94
 LED schalten (Experiment) 80
 LEGO Digital Designer 15, 187
 LEGO EV3
 mittlerer Motor 22
 Motor 22
 LEGO-Colorsensor 22
 LEGO-Infrarotsensor 22
 LEGO-Touchsensor 22
 leJOS 145
 Lernerfolg
 verbessern bei neuronalen Netzen 220
 Lernergebnis
 speichern 226
 Lernfortschritt
 anzeigen 226
 Lernschritt 186
 nötige Anzahl 221
 Licht erkennen (Experiment) 100
 Lichteinfall
 scannen 325
 Lichtsensor 23, 35, 185, 191
 Lichtstärke 185
 Lichtverhältnis
 Farbdefinition 252
 LIFO 168
 Linie 185
 Linienverfolgung 185
 klassische 197
 Lernschritte 189
 Training 202
 Linux 14, 25
 Liste 70
 Element 71
 Länge 71
 Listing 60
 Logische Operation 69
 Lokale Variable 99

Löten 287
 Spannungsversorgung 293
 LötKolben 23
 Lötzinn 23

M

Masse 29
 Masseanschluss 316
 Master Mind 334
 math 99
 Mauskabel 121
 Mausclick 251
 MCP3008 28
 Pinbelegung 301
 Mehrdimensionale Liste 71
 Methode 108
 Mikroschalter 23
 Mikro-SD-Karte 26, 31
 Mindstorms 22
 Minuspol 81
 Mittlerer Motor 38
 Modell der Wirklichkeit 107
 Modulbibliothek 40, 77, 80, 99, 124
 Modulo-Operator 231
 Momentaner Zustand 184
 Morsealphabet 289
 Morseapparat 289, 304
 Morsecode
 in Text überführen 307
 Text übersetzen 304
 übersetzen 304
 Morsen 289
 Motor 38, 42
 Encoder 127
 großer 127
 mittlerer 38
 Motorgeschwindigkeit 40
 Motorspannung 27
 Motortreiber 44
 Befestigung 315
 Beschaltung 45
 Move Engine 172
 move_back 174
 move_since_wall 173
 Multiplikation 66, 90
 MZ 184

N

Name
 Funktion 97
 Nervenleitung 188

- Netz
 - neuronales 185, 187, 215
- Netzteil 31
- Netzteilkabel 121
- Netzwerkstruktur 188
- Neuron 188
- Neuronales Netz 185, 187, 215
 - fehlerhafte Daten 220
 - Generalisierung 220
 - Schichten 188
 - Training 188, 226
- Nicht gleich 74
- Nicht-Verknüpfung 70
- Nimmspiel 333
- Normale Division 64

- O**
- Oberflächenfarbe 244
- Objekt 107
 - Druckdatei 216
 - erkennen 215, 220
 - Farbe 244
 - herstellen 216
 - klassifizieren 215, 231
- Objektorientierte Programmierung 107
- Oder-Verknüpfung 70
- Oktaeder 313
- Oktal 63
- Oktale Darstellung 64
- OOP 107
 - Instanz und Objekt 109
 - Vererbung 111
- Operation 66
 - logische 69
- Operator 69
 - Bedingungen 73
 - Modulo 231
- Optischer Encoder 43
- Ordnerstruktur 34
- Output-Schicht 188

- P**
- Pappkarte 237
- Parameter 98, 110
- Parcours 35, 267
- pass 74, 85, 94
- Piezo-Schallgeber 23, 112, 297
- Pin-Nummern 30
- Pixel 245
- Platonische Fläche 313
- Platonischer Körper 313
- Pluspol 81
- Pluszeichen 67, 72

- popStack 169
- Potenz 66
- Potenzieren 99
- Powerpack 38
- Problemlösung 165, 184
- Problemlösungsprozess 165
- Produktionssystem 165
- Programm 58
- Programmbefehl 55
- Programmfehler 94
- Programmierung 53
 - objektorientierte 107
- Programmnamen 59
- Programmstil 58
- Programmversion 60
- Projekt
 - Auf dem Tisch bleiben 153
 - Ausweichen 121
 - Ball verfolgen 333
 - Hellsten Ort suchen 325
 - Himmelsrichtung erkennen 141
 - Joghurtbecher einsammeln 267
 - Labyrinth 161
 - Linienverfolgung 185
 - Linienverfolgung mit der Kamera 333
 - Master Mind 334
 - Morsen 289
 - Nimmspiel 333
 - Objekte erkennen 215
 - Pappkarten abschießen 237
 - platonische Flächen abfahren 314
 - Tic-Tac-Toe 334
 - Türme von Hanoi 334
 - Zahl raten 334
- Promptzeichen 56
- Prozessor 55
- Pull-down-Widerstand 50
- Puls 94
- Pulsweitenmodulation 30, 94
- Pulswelle 94
- pushStack 168
- PWM 30, 94
- Python 26, 38, 53, 55
 - Modulbibliotheken 77
- Python-Statement 170

- Q**
- Quasi-Konstante 72

- R**
- Rad 315
- range 88

Raspberry Pi 14, 26
 Kamera 50, 237, 241
 konfigurieren 30
 Pull-down-Widerstände 50
 Python-IDE 56
 Schnittstelle 80
 Raspberry Pi 3 B+ 25, 26
 Raspberry Pi OS 15, 26
 Raupenriemen 121, 137
 Rechner-App 64
 Rechtecksignal 94, 117
 Regelbasis 166, 170
 Regelinterpreter 165, 168, 170
 Reifen 23
 Reifenumfang 131
 Reihenfolge 170
 Restschleife 92
 rotateOnly 175
 Rotation 133
 RPi.GPIO 80
 Rückgabewert 97, 99

S

Scannen 219, 271
 Lichteinfall 325
 Schallgeber 42
 Schicht 188
 Elemente 188
 Schleife 85, 90
 abbrechen 91
 Schleifendurchlauf 132
 Schleifenkopf 86
 for 89
 Schleifenrumpf 86
 Schleifenstatement 85
 Schlüsselsituation 173
 Schreibfehler 62
 SD-Karte 25
 SD-Kartenleser 32
 Sechseck 314
 Semantischer Fehler 62
 Sensor 35
 analoger 101
 Sensortyp
 Klasse 191
 shebang-Zeile 58, 77
 Shell 56, 75, 79
 Signal
 akustisches 297
 Sonderzeichen 67
 Sourcecode 60, 107
 Spannung 29, 83, 103
 Spannungsteiler 294

Spannungsversorgung 316
 löten 293
 Speed 131
 einstellen 173
 SPI 102
 SPI-Baustein 28
 SPI-Interface 329
 Stack 168
 Standardmethode 108
 Stapelspeicher 168
 Statement 55, 74, 97
 ausführen 75
 Staubsaugerroboter 121
 Steckkontakt 81
 Stiflleiste 43, 44
 Stoppzeichen 230
 Störsignalquelle 50
 String 67
 Stromversorgung 27
 Subjektorientierung 107
 Subtraktion 66
 sudo raspi-config 242
 super() 111
 Syntaxfehler 62
 sys 115
 sys (Bibliothek) 139
 System
 wissengestütztes 165
 Szene 244

T

Tastatur 33
 Tastaturkabel 121
 Tastaturlayout 32
 Taster 295
 Teileliste 19
 Tetraeder 313
 Text 67
 in Morsecode übersetzen 304
 Thonny 38, 55, 56
 Thread 26, 125, 307
 Tic-Tac-Toe 334
 time.sleep() 124
 Timeout 168
 Ton
 erzeugen 112, 297, 300
 Tonfolge (Experiment) 117
 tool4robot.py (Bibliothek) 145, 151
 top_stack 169
 Touchsensor 23, 36, 42, 147
 basteln 302
 Scannen 219

Training 188
 Beispieldaten, Anzahl 190
 Fehler 190
 Lernkurve 190
 Lernschritt 188
 Linienverfolgung 202
 neuronales Netz 226
Trainingsdaten
 erstellen 221
Transferfunktion 197
Transistor 102
True 69
try 94
try / except / finally 94, 105
Tupel 70, 72
Turm 238
Türme von Hanoi 334
Turmkanone 237

U

UART 33
Überlauf 147
Überschreiben 112
Umrechnungsfaktor 230
Und-Verknüpfung 69
Unterziel 167, 177
USB-Akku 27
USB-Maus 25, 31
USB-Netzteil 25, 27
USB-Tastatur 25, 31

V

Variable 63, 66
 Laufvariable 88
 lokale 99
Vaterklasse 111
Vererbung 111
Verknüpfung 69
Verstärker 297
Verzweigung 73

W

Wahrheitswert 69, 73, 74
Wenden 175
Wenn-Dann-Regel 166
while 85, 92, 151, 170
Win32 Disk Imager 32
Wissengestütztes System 165
Wissensbasis 165
WLAN 25, 33
Würfel 313

Y

YETI 19

Z

Zahl 63
 Darstellung 64
 komplexe 63
 raten 334
Zahlentyp 64
Zeichenkette 67
Zeilenwechsel 90
Zeitbibliothek 124
Zeitspanne 93
Zeitsteuerung 124
Ziel
 aufteilen 167
Zielen 238
Zielzustand 184
Zufallszahl 82
Zukunftstechnologie 235
Zustand 184
 Expertensystem 184
Zustandsraum 184
Zuweisung 73
ZZ 184