Hardware GUIDE

Raspberry Pi

PROJEKTE

PCGH GUIDE 06/15
€ 12,99
Österreich € 14,50
Schweiz CHF 22,-
Benelux € 15,-

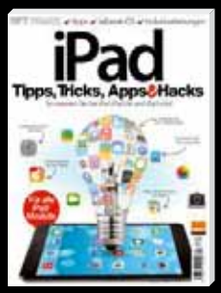
computec MEDIA EDITION

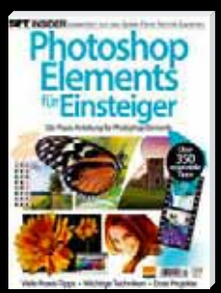
Praxis-Workshops • Python-Apps • Eigenbau-Roboter • u. v. m.

computec MEDIA

EDITION

ALLE PREMIUM-BOOKAZINES IM ÜBERBLICK





Bequem online bestellen:
shop.computec.de/edition

Oder einfach digital lesen:
epaper.computec.de

Willkommen zu

Raspberry Pi

PROJEKTE

Eine Revolution mit Himbeergeschmack findet – ausgehend von den Britischen Inseln – in der ganzen IT-Welt statt: Seit der erste Minicomputer mit dem schmackhaften Namen, dem poppigen Logo und den höchst erstaunlichen Eigenschaften Anfang 2012 aus der Serienproduktion kam, findet ein Siegeszug statt, wie man ihn in der Welt der Computerbastlerinnen, Hobbyprogrammierer und Technikbegeisterten seit langer, langer Zeit nicht mehr gesehen hat. Aber der Raspberry Pi, der für nur 35 Euro erhältliche, vollwertige PC im Format einer Scheckkarte, findet auch mehr und mehr in professionellen Bereichen und der Industrie Verwendung.

Schlüssel zum großen Erfolg des Ein-Platinen-Geräts ist seine unglaubliche Vielseitigkeit. Der Raspberry Pi kann als On-Board-Steuereinheit von Roboterfahrzeugen benutzt werden, an einem Ballon befestigt atemberaubende Fotoaufnahmen aus der Stratosphäre machen, in regelmäßigen Abständen die Topfblume gießen, als Bewegungsmelder fungieren, mit sich selbst via Twitter Gespräche führen oder einfach als Server-Computer im Dauerbetrieb mit extrem niedrigem Stromverbrauch dienen.

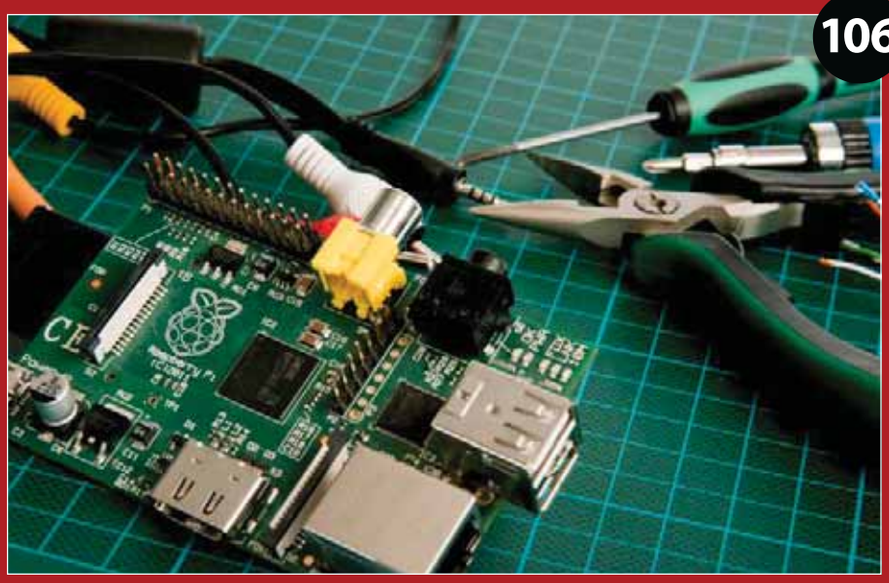
Der eigentliche Gedanke, den die Entwickler des Raspberry Pi hatten, war jedoch, Kinder und Jugendliche endlich wieder für das Programmieren, Hacken und Herumprobieren mit Elektronik zu begeistern – und dies ist ganz offensichtlich gelungen! Der Minicomputer eignet sich bestens für erste Gehversuche im Coding, aber genauso gut für umfassende, anspruchsvolle Softwareprojekte.

In diesem Bookazine stellen wir Ihnen den Raspberry Pi ausführlich vor: von der Einrichtung über Upgrades und Zubehör, bis hin zu vielen spannenden Anwendungsideen aus der Praxis. Tüfteln, bauen, basteln, experimentieren, schalten, verdrahten, programmieren, lernen und lehren – all das können Sie mit dem Raspberry Pi machen. Was sollte Sie aufhalten?

INHALT

„Alles was Sie brauchen, um Technologie zum Leben zu erwecken."

„Lernen Sie, wie man in Python programmiert, und realisieren Sie tolle Projekte."



Basics. Projekte. Ideen. Know-how.

JAHRES-ABO
über 15% Rabatt
6 Ausgaben
nur 49,90 €

ABO-VORTEILE

- Günstiger als am Kiosk
- Versandkostenfrei per Post
- Pünktlich und aktuell
- Keine Ausgabe verpassen

Jetzt bestellen!

shop.raspberry-pi-geek.de

computec MEDIA EDITION

PC Games Hardware GUIDE
DAS HARDWARE-MAGAZIN FÜR PC-SPIELER

Raspberry Pi

PROJEKTE

computec

Ein Unternehmen der MARQUARD MEDIA INTERNATIONAL AG
Verleger Jürg Marquard

Verlag Computec Media GmbH
Dr.-Mack-Straße 83, 90762 Fürth
Telefon: +49 911 2872-100
Telefax: +49 911 2872-200
E-Mail: redaktion@sft-magazin.de
shop.computec.de/edition

Geschäftsführer Rainer Rosenbusch, Hans Ippisch

Redaktionsleiter PCGH GUIDE (V.i.S.d.P.) Lars Craemer, verantwortlich für den redaktionellen Inhalt, Adresse siehe Verlagsanschrift

Mitarbeiter dieser Ausgabe MDV Textdienste, Claudia Costanza, Frank Neupert-Paries, Marc Brehme, Hansgeorg Hafner, Albert Kraus
Lektorat Claudia Brose (Ltg.), Birgit Bauer, Esther Marsch, Heidi Schmidt, Ina Hulm

Vertrieb, Abonnement Werner Spachmüller (Ltg.)
Marketing Jeanette Haag
Produktion Thomas Eder, Jörg Gleichmar
Head of Online Christian Müller

Anzeigen
CMS Media Services GmbH, Dr.-Mack-Straße 83, 90762 Fürth

Verantwortlich für den Anzeigenteil
René Behme, Adresse siehe Verlagsanschrift

Anzeigenberatung Print:
Bernhard Nusser: Tel.: 0911-2872-254; bernhard.nusser@computec.de
René Behme: Tel.: 0911-2872-152; rene.behme@computec.de
Alto Mair: Tel.: 0911-2872-144; alto.mair@computec.de
Anne Müller: Tel.: 0911-2872-251; anne.mueller@computec.de

Anzeigenberatung Online:
Ströer Digital Media GmbH
Stresemannstraße 29, 22769 Hamburg
Tel.: +49 (0) 40-46 85 67-0
Fax. +49 (0) 40-46 85 67-39
Mail: info@stroeerdigitalmedia.de
www.stroeerdigitalmedia.de

Anzeigendisposition: anzeigen@computec.de
Datenübertragung: via E-Mail: anzeigen@computec.de
Es gelten die Mediadaten Nr. 28 vom 01.01.2015

Vertrieb und Einzelverkauf
DPV Deutscher Pressevertrieb GmbH, Düsternstr. 1-3, 20355 Hamburg, Internet: www.dpv.de

Druck
Quad/Graphics Europe, 120 Pultuska Street, 07-200 Wyszków, Polen

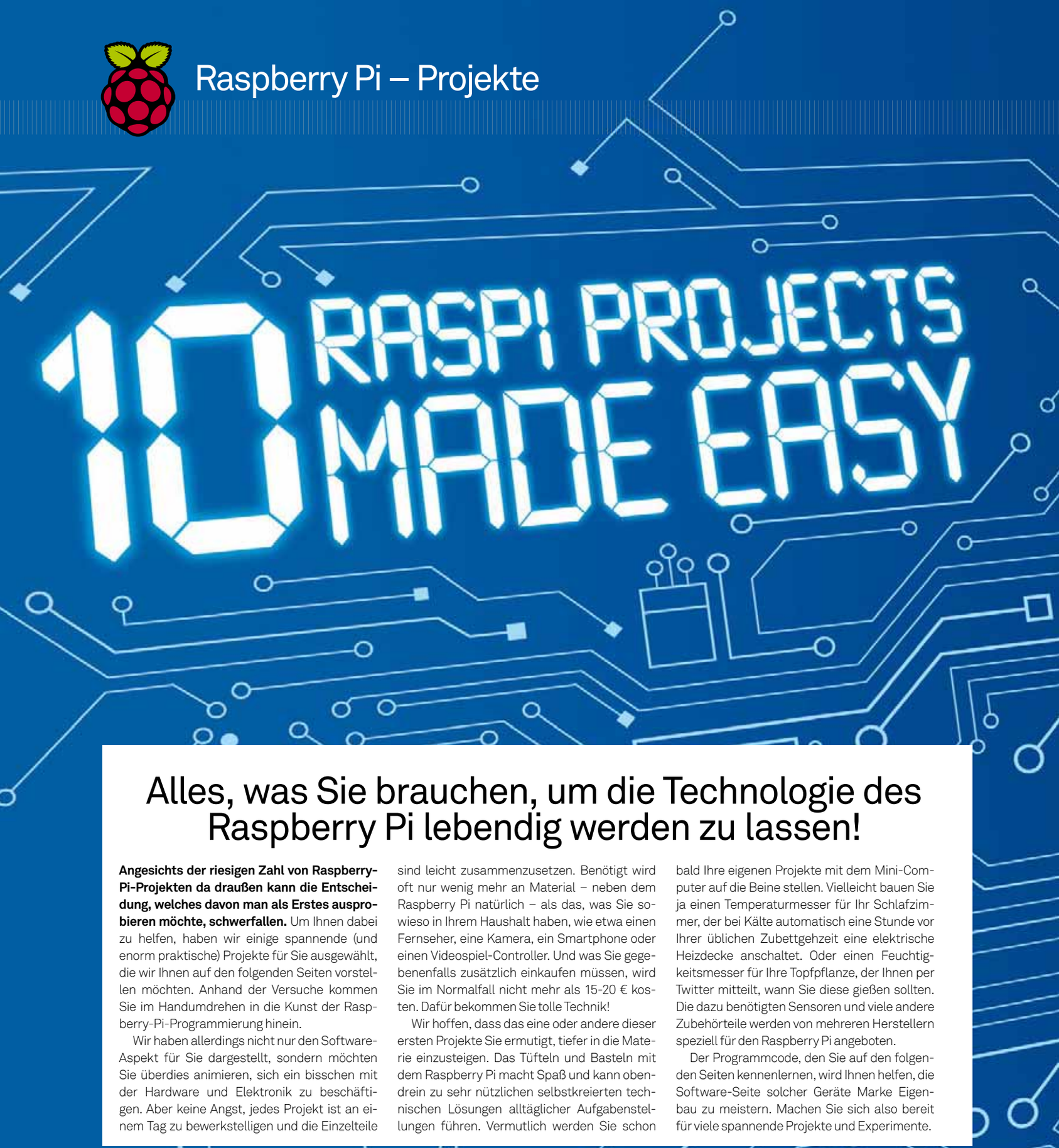
ip IMAGINE PUBLISHING

Publishing Director: Aaron Asadi, **Head of Design:** Ross Andrews, **Edited by:** Alex Hoskins & Gavin Thomas, **Assistant Designer:** Perry Wardell-Wicks, **Photographer:** James Sheppard, **Senior Art Editor:** Greg Whitaker
www.imagine-publishing.co.uk

MARQUARD MEDIA INTERNATIONAL AG

Deutschsprachige Titel: SFT, WIDESCREEN, PC GAMES, PC GAMES MMORE, PC GAMES HARDWARE, BUFFED, X3, PLAY 4, GAMES & MORE, GAMES AKTUELL, N-ZONE, XBG GAMES, LINUX-MAGAZIN, LINUXUSER, EASYLINUX, RASPBERRY PI GEEK

Internationale Zeitschriften: Polen: COSMOPOLITAN, JOY, SHAPE, HOT, PLAYBOY, CKM, VOYAGE, HARPER'S BAZAAR; Ungarn: JOY, SHAPE, ÉVA, IN STYLE, PLAYBOY, CKM, MEN'S HEALTH



Alles, was Sie brauchen, um die Technologie des Raspberry Pi lebendig werden zu lassen!

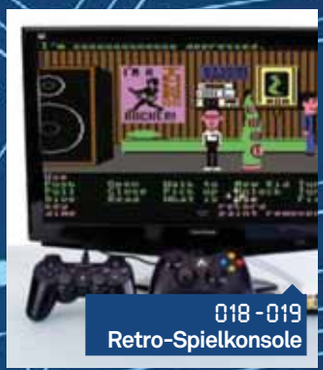
Angesichts der riesigen Zahl von Raspberry-Pi-Projekten da draußen kann die Entscheidung, welches davon man als Erstes ausprobieren möchte, schwerfallen. Um Ihnen dabei zu helfen, haben wir einige spannende (und enorm praktische) Projekte für Sie ausgewählt, die wir Ihnen auf den folgenden Seiten vorstellen möchten. Anhand der Versuche kommen Sie im Handumdrehen in die Kunst der Raspberry-Pi-Programmierung hinein.

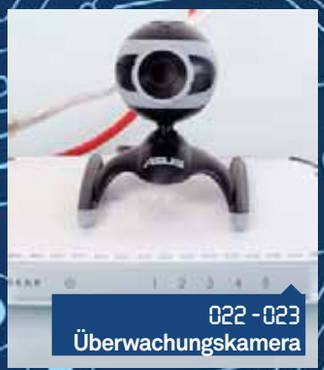
Wir haben allerdings nicht nur den Software-Aspekt für Sie dargestellt, sondern möchten Sie überdies animieren, sich ein bisschen mit der Hardware und Elektronik zu beschäftigen. Aber keine Angst, jedes Projekt ist an einem Tag zu bewerkstelligen und die Einzelteile sind leicht zusammenzusetzen. Benötigt wird oft nur wenig mehr an Material – neben dem Raspberry Pi natürlich – als das, was Sie sowieso in Ihrem Haushalt haben, wie etwa einen Fernseher, eine Kamera, ein Smartphone oder einen Videospiel-Controller. Und was Sie gegebenenfalls zusätzlich einkaufen müssen, wird Sie im Normalfall nicht mehr als 15-20 € kosten. Dafür bekommen Sie tolle Technik!

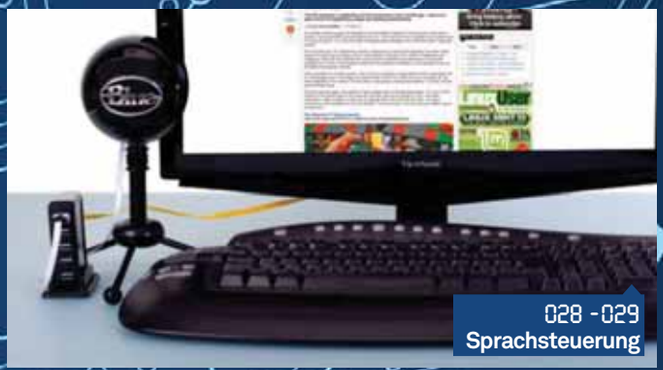
Wir hoffen, dass das eine oder andere dieser ersten Projekte Sie ermutigt, tiefer in die Materie einzusteigen. Das Tüfteln und Basteln mit dem Raspberry Pi macht Spaß und kann obendrein zu sehr nützlichen selbstkreierten technischen Lösungen alltäglicher Aufgabenstellungen führen. Vermutlich werden Sie schon bald Ihre eigenen Projekte mit dem Mini-Computer auf die Beine stellen. Vielleicht bauen Sie ja einen Temperaturmesser für Ihr Schlafzimmer, der bei Kälte automatisch eine Stunde vor Ihrer üblichen Zubettgehzeit eine elektrische Heizdecke anschaltet. Oder einen Feuchtigkeitsmesser für Ihre Topfpflanze, der Ihnen per Twitter mitteilt, wann Sie diese gießen sollten. Die dazu benötigten Sensoren und viele andere Zubehörteile werden von mehreren Herstellern speziell für den Raspberry Pi angeboten.

Der Programmcode, den Sie auf den folgenden Seiten kennenlernen, wird Ihnen helfen, die Software-Seite solcher Geräte Marke Eigenbau zu meistern. Machen Sie sich also bereit für viele spannende Projekte und Experimente.

10 spannende Projekte





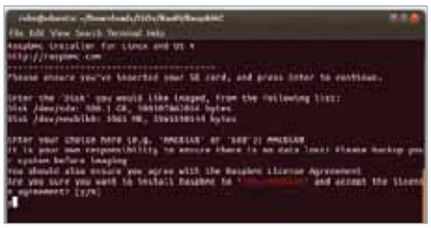


Ein komplettes Mediacenter

WAS SIE BRAUCHEN:
- Internetverbindung
- Externe Festplatte
- OmniVESA-Halterung
- HDMI-Kabel

Mit ein bisschen Software und einigen Kniffen wird Ihre Raspberry Pi zum Herzstück einer Multimedia-Anlage.

Es gibt mehrere Möglichkeiten, den Pi in einen Home-Theater-PC (HTPC) zu verwandeln. Besonders empfehlenswert sind dabei Distributionen, die auf XBMC basieren. OpenELEC zum Beispiel ist ein effizientes, schlankes Betriebssystem speziell für diesen Anwendungsbereich. In diesem Artikel behandeln wir jedoch das anpassungsfähigere Raspbmc, das auf Debian aufbaut.



01 Raspbmc installieren

Für Raspbmc gibt es ein Installer-Skript, das das jeweils aktuelle Image des Betriebssystems für den Raspberry Pi herunterlädt. Erstellen Sie ein neues Verzeichnis und holen Sie sich das Skript mit folgendem Befehl:

```
$ wget http://svn.stmlabs.com/svn/
raspbmc/release/installers/python/
install.py
```

Machen Sie es zu einer ausführbaren Datei:

```
$ chmod +x install.py
```

Starten Sie das Programm und folgen Sie den Anweisungen auf dem Bildschirm:

```
$ sudo python install.py
```

02 Einstellungen

Indem Sie Raspbmc auf diese Weise installieren, stellen Sie sicher, dass Sie die neueste Version der Software auf dem Pi haben. Der Download der dazu notwendigen Dateien dauert eine Weile, außerdem müssen Sie den Pi im Verlauf der Installation mehrfach neu starten, aber dies ist nur beim ersten Durchgang so. Wählen Sie bei XBMC noch Ihre Spracheinstellung aus.

03 Netzwerk-Sharing

Sie können im Menü über Video > Add Source die Quellen für Ihre Videodateien zu XBMC hinzufügen. Klicken Sie auf Browse, und Sie können Offline- oder Netzwerk-Files hinzufügen. Beim Netzwerk-Sharing haben Sie die Wahl zwischen den Verfahren Universal Plug and Play (UPnP) und Server Message Block (SMB), oder Sie verwenden „Add network location“, wenn Sie den direkten Pfad kennen. Bei Musik und Bildern ist das Prozedere entsprechend.

04 Medieninformationen

Nachdem Sie eine Quelle ausgewählt haben, können Sie festlegen, welchen Dienst Sie für die Informationsbeschaffung (etwa Zusammenfassungen von Folgen einer TV-Serie) anzapfen möchten. Achten Sie darauf, dass Ihre Filme und Serien Jahresangaben haben.

05 Apps auswählen

Bei jeder Medienart können Sie über „Add-Ons“ die Apps anzeigen lassen, die Sie bereits besitzen. Mit einem Klick auf „Get More“ bekommen Sie eine komplette Liste aller verfügbaren Apps, die Sie direkt von der Liste aus installieren können.

06 Fernsteuerung per Browser

Gehen Sie zu Settings > Services > Webserver. Aktivieren Sie diesen, und Sie können den Raspberry Pi per Browser bedienen. Dazu benötigen Sie die IP-Adresse des Pi (zu finden in Settings > System Info). Tragen Sie diese in die Adresszeile des Browsers ein, gefolgt von „:80“.

07 Android-Fernbedienung

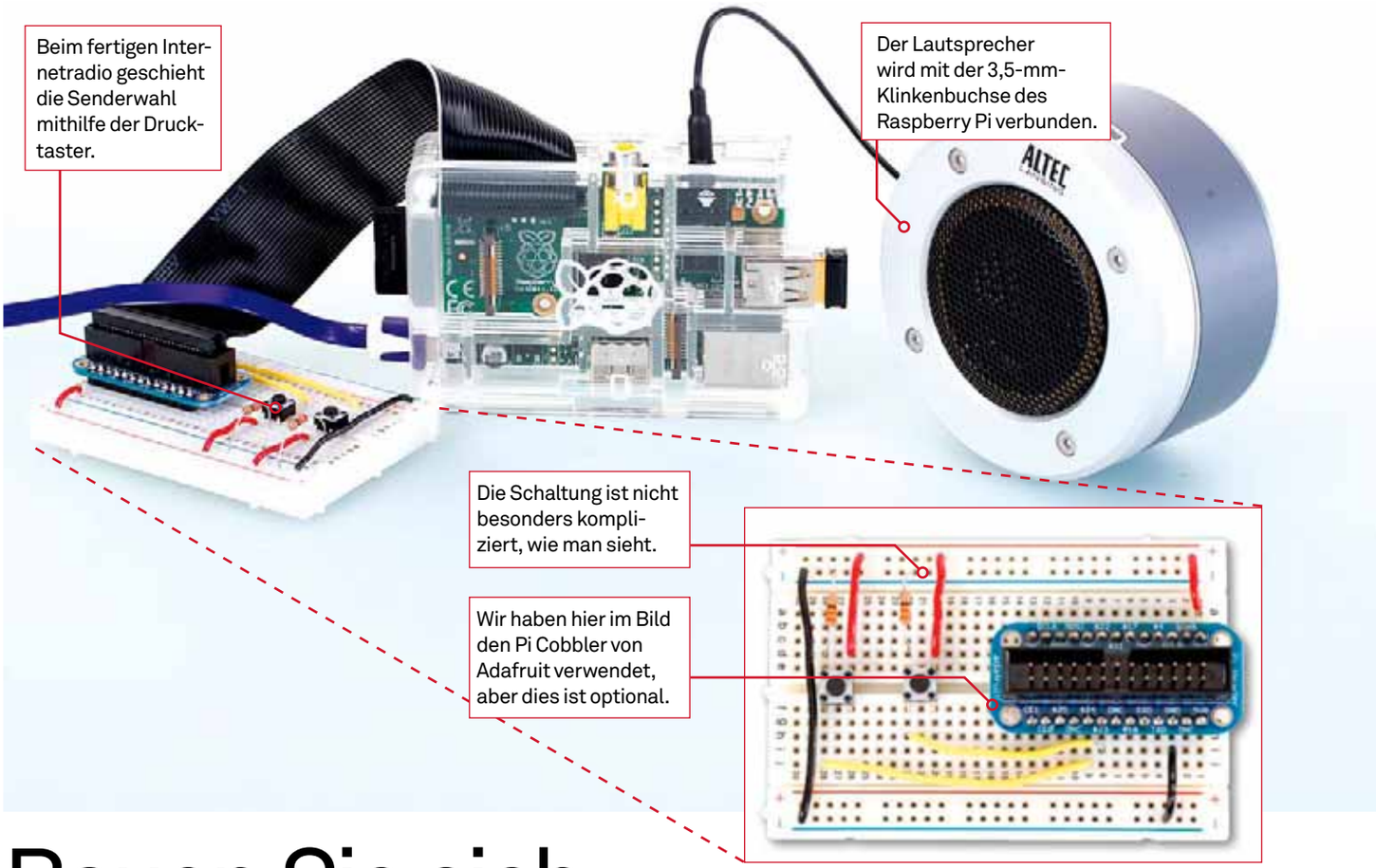
Es gibt eine offizielle Android-Fernbedienungs-App im Google Play Store. Damit können Sie XMBC steuern, sobald der Webserver läuft. Laden Sie die App auf Ihr Smartphone und installieren Sie sie. Fügen Sie in den Einstellungen einen neuen Host hinzu und geben Sie IP-Adresse und Port (standardmäßig: 80) ein wie in Schritt 06.

08 CEC verwenden

Falls Ihr TV-Gerät den CEC-Standard unterstützt (siehe **kodi.wiki/view/CEC**), können Sie auf Ihren XBMC-Pi per HDMI zugreifen, somit also das gesamte Mediacenter mit einer Fernbedienung kontrollieren.

09 Klassisch per Infrarot

XBMC unterstützt auch etliche klassische Universal-Fernbedienungen, die mit USB-Infrarot-Empfängern funktionieren. Sehr empfehlenswert ist der USB-Dongle FLIRC, ein programmierbarer Infrarotempfänger.



Bauen Sie sich ein Internetradio

So machen Sie den Raspberry zu einem WLAN-Streaming-Radio.

Im Netz stehen Tausende frei empfangbare Radiosender zur Verfügung, und wir stellen Ihnen hier ein kleines Projekt vor, bei dem Sie den Raspberry Pi in einen handlichen Receiver verwandeln können.

01 Vorbereitungen

Wenn Raspbian installiert ist und der Pi mit dem Internet verbunden ist, wechseln Sie in der Kommandozeile zum root user:

```
sudo su
```

Aktualisieren Sie Ihre Pakete und bringen Sie den Pi auf den neuesten Stand:

```
apt-get update && apt-get upgrade -y
```

02 Extrapakete installieren

Mit dem folgenden Befehl installieren Sie die Python-Pakete, die für den Zugriff auf die GPIO-Kontaktstifte nötig sind:

```
apt-get install python-rpi.gpio
```

Installieren Sie anschließend den Mediaplayer MPlayer:

```
apt-get install mplayer
```

WAS SIE BRAUCHEN:

- eine drahtlose Internetverbindung
- 2 Drucktaster
- 1 Steckplatine
- 4 Kabel (weiblich zu männlich) (um den Raspberry Pi mit der Steckplatine zu verbinden)
- 4 Kabel (männlich zu männlich)
- 2 Widerstände (220 Ohm)
- Lautsprecher (mit Anschlusskabel und 3,5-mm-Klinkenstecker)

Dieser Code verbindet die GPIO-Pins mit dem MPlayer, der die Audiowiedergabe besorgt:

Die GPIO-Bibliothek für den Raspberry Pi wird importiert und auf BCM-Modus gesetzt.

Hier werden die GPIO-Stifte 23 und 24 eingestellt, die für die Sendersuche zuständig sind.

Wenn der Taster für Stift 23 gedrückt wird, wird das hier ausgeführt.

Das Skript bricht etwaige offene Prozesse des MPlayers ab und lädt den Sender.

```
#!/usr/bin env python
import time import sleep
import os
import RPi.GPIO as GPIO
# I found loads of BBC Radio streams from http://bbcstreams.com/
GPIO.setmode(GPIO.BCM)
GPIO.setup(23 , GPIO.IN)
GPIO.setup(24 , GPIO.IN)
while True:
    if GPIO.input(23)==1:
        os.system('sudo killall mplayer')
        os.system('mplayer -playlist http://bbc.co.uk/radio/listen/live/r1.asx &')
    if GPIO.input(24)==1:
        os.system('sudo killall mplayer')
        os.system('mplayer -playlist http://bbc.co.uk/radio/listen/live/r6.asx &')
    sleep(0.1);
GPIO.cleanup()
```

03 Hardware

Wir verbinden die GPIO-Stifte 23 und 24 mit jeweils einem Drucktaster. Mit 4 Kabeln (weiblich zu männlich) schließen wir die Kontakte 23, 24, 3V3 (3,3 Volt) und GND (Masse) an die Steckplatine an. Für die beiden 220-Ohm-Widerstände benötigen wir weitere 4 Kabel (männlich zu männlich) (siehe Schaubild links).

04 Software

Fügen Sie den oben angezeigten Code in eine neue Datei namens **radio.py** ein und legen Sie diese in Ihrem Home-Verzeichnis ab. Sie können den Code später nach Wunsch anpassen. Öffnen Sie nun die Datei, die die Konfiguration der Netzwerkschnittstelle enthält:

```
nano /etc/network/interfaces
```

05 WLAN-Konfiguration

Der Raspberry Pi soll sich beim Booten automatisch via WLAN mit Ihrem Router verbinden.

Ändern Sie dazu die Datei **/etc/network/interfaces** wie folgt:

```
auto lo
iface lo inet loopback
iface eth0 inet dhcp
allow-hotplug wlan0
auto wlan0
iface wlan0 inet dhcp
wpa-ssid "ssid"
wpa-psk "password"
```

Ersetzen Sie dabei **ssid** und **password** mit Ihrer Geräte-ID und Ihrem Passwort und lassen Sie die Anführungszeichen stehen.

06 Start beim Booten

Gehen Sie im Terminal zu **/etc/init.d/** und generieren Sie mit dem Nano-Editor eine Datei namens **radio**.

```
nano radio
```

Fügen Sie folgenden Code in die Datei ein:

```
#! /bin/bash
modprobe snd_bcm2835
amixer cset numid=3 1
python /home/pi/radio.py
```

Damit wird das Kernel-Modul für die Soundkarte geladen. Der Mixer amixer legt als Output die 3,5-mm-Klinkenbuchse fest (dafür steht die 1; HDMI wäre 2). Zuletzt ruft der Code das Python-Skript auf.

07 Datei ausführbar machen

Speichern Sie die Datei im Verzeichnis **/etc/init.d** und machen Sie sie dann mit dem folgenden Befehl ausführbar:

```
chmod 755 radio
```

Danach legen Sie fest, dass die Datei während des Bootprozesses gestartet werden soll:

```
update-rc.d radio defaults
```

08 raspi-config

Ändern Sie in der Konfigurationsdatei **raspi-config** das Bootverhalten des Raspberry Pi. Wir brauchen den Desktop nicht, sondern nur ein Terminal. Nachdem Sie die entsprechende Einstellung vorgenommen haben, booten Sie den Pi erneut.

> **"Wir stellen ein Projekt vor, bei dem Sie den Pi in einen handlichen Receiver verwandeln."**

09 Der erste Test

Nachdem der Raspberry Pi komplett hochgefahren ist, drücken Sie einen der Taster auf der Steckplatine. Nach ein paar Sekunden sollte ein Audiosignal aus dem Lautsprecher kommen, den Sie an die 3,5-mm-Klinkenbuchse angeschlossen haben. Dann hat es geklappt – Sie haben ein drahtloses Internetradio!

Wenn Sie möchten, können Sie noch weitermachen: etwa mithilfe eines zusätzlichen Tasters und amixer **(manpages.ubuntu.com/manpages/gutsy/man1/amixer.1.html)** einen Stummschalter einbauen oder sogar ein LCD-Display anschließen **(rpiblog.com/2012/11/interfacing-16x2-lcd-with-raspberry-pi.html)**, das die Senderinfos anzeigt.

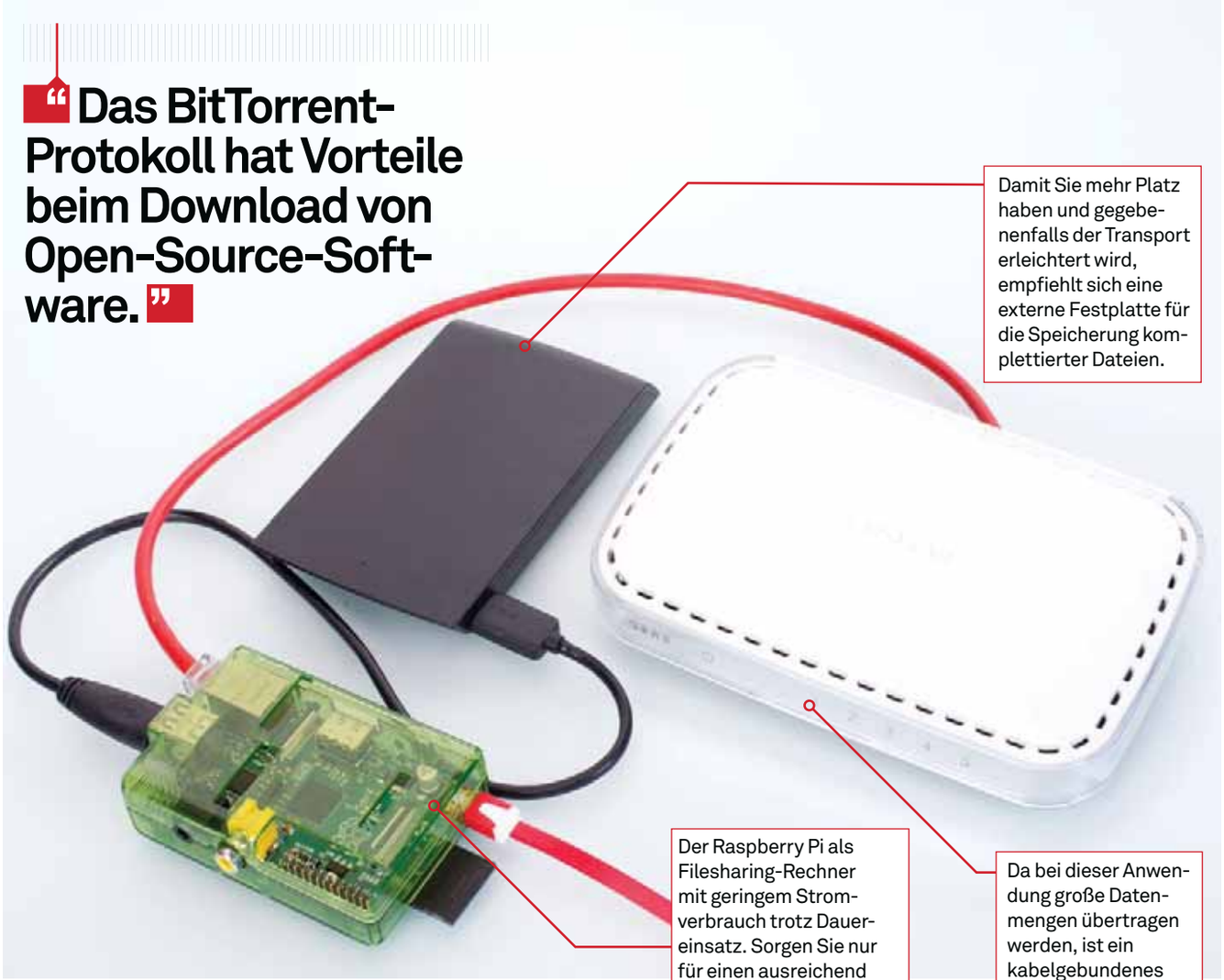
Filesharing im Dauerbetrieb

Holen Sie sich aktuelle Distros, Pakete und Beta-Versionen im Handumdrehen mit dem Pi!

Filesharing per BitTorrent hat keinen besonders guten Ruf, da es häufig mit Software- und Medienpiraterie in Verbindung gebracht wird. Doch das Verfahren bietet einige wichtige Vorteile beim Download von Open-Source-Software: Es ist tendenziell schneller, verringert die Bandbreite und erlaubt das Zurückteilen mit der Community. Distributionen, Pakete und anderes sind als Torrents verfügbar, und aus dem Raspberry Pi kann man einen kleinen, stromsparenden, allzeit bereiten Torrent-Rechner machen. In diesem Tutorial wollen wir den Pi so einrichten, dass wir von einem Linux-Computer auf ihn zugreifen und somit per Fernsteuerung die Torrents verwalten können.

WAS SIE BRAUCHEN:
- » eine externe Festplatte
- » Raspbian raspberrypi.org/downloads
- » einen Desktop-PC mit installiertem BitTorrent-Client Deluge

“Das BitTorrent-Protokoll hat Vorteile beim Download von Open-Source-Software.”



Damit Sie mehr Platz haben und gegebenenfalls der Transport erleichtert wird, empfiehlt sich eine externe Festplatte für die Speicherung komplettierter Dateien.

Der Raspberry Pi als Filesharing-Rechner mit geringem Stromverbrauch trotz Dauereinsatz. Sorgen Sie nur für einen ausreichend belüfteten Standort.

Da bei dieser Anwendung große Datenmengen übertragen werden, ist ein kabelgebundenes Netzwerk ein Muss.

01 Raspbian installieren

Raspbian ist ein geeignetes Betriebssystem für unseren BitTorrent-Pi. Spielen Sie das Image auf eine SD-Karte und nehmen Sie das Setup vor, wobei SSH aktiviert und die grafische Benutzeroberfläche deaktiviert werden sollte.

02 Fernzugriff

Tippen Sie `ifconfig` in die Kommandozeile des Pi, um die IP-Adresse herauszufinden. Ab jetzt könnten Sie den Monitor ausstöpseln und per Fernzugriff mit der Einrichtung fortfahren. In jedem Fall erreichen Sie den Pi mit dem folgenden Befehl sowie der Eingabe Ihres Passwortes:

```
$ ssh [Benutzer]@[IP-Adresse]
```

03 Festplatte einhängen

Falls Sie Ihre externe Festplatte nicht neu formatieren möchten, müssen Sie den Pi fit für NTFS machen. Geben Sie ein:

```
$ sudo apt-get install ntfs-3g
```

Öffnen Sie mit `sudo nano /etc/fstab` die Datei **/etc/fstab** und fügen Sie dort die Festplatte hinzu, indem Sie Folgendes eintragen:

```
/dev/[Adresse der Festplatte]
[Einhängepunkt] auto noatime 0 0
```

Verwenden Sie fdisk, um den Namen des Speichers zu ermitteln, und erzeugen Sie mit mkdir einen neuen Einhängepunkt à la **/home/pi/torrents**. Booten Sie neu, um die Änderung wirksam werden zu lassen.

04 Deluge installieren

Als BitTorrent-Client verwenden wir Deluge. Installieren Sie es mit:

```
$ sudo apt-get install deluged
deluge-console
```

Starten und schließen Sie Deluge, sodass es eine Konfigurationsdatei erstellt, die Sie editieren können:

```
$ deluged
$ sudo pkill deluged
```

Kopieren Sie abschließend die Konfigurationsdatei für den Fall, dass etwas schiefgeht:

```
$ cp ~/.config/deluge/auth ~/.config/
deluge/auth.old
```

05 Basis-Konfiguration

Öffnen Sie die Datei mit:

```
$ nano ~/.config/deluge/auth
```

Fügen Sie am Ende hinzu:

```
[Benutzer]:[Passwort]:10
```

Damit beschränken Sie den Zugang. Starten Sie nun Deluge erneut:

```
$ deluged
$ deluge-console
```

06 Verbindung herstellen

Nun sind Sie innerhalb des Clients. Geben Sie die folgenden drei Befehle ein:

```
config -s allow_remote True
config allow_remote
exit
```

Starten Sie den Deluge-Daemon neu:

```
$ sudo pkill deluged && deluged
```

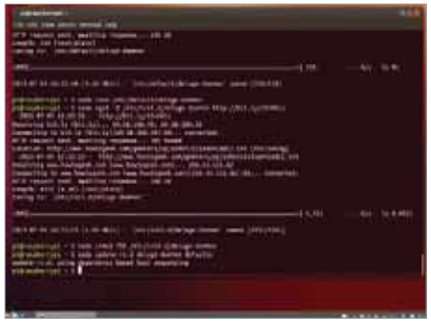
Öffnen Sie anschließend den grafischen Client auf Ihrem Linux-Rechner.

07 Schnittstelle für Fernzugriff

Gehen Sie im Menü zu Edit > Preferences > Interface, deaktivieren Sie den „Classic Mode“, und starten Sie Deluge neu. Klicken Sie im Connection Manager den Button „Add“, und geben Sie bei Hostname die IP-Adresse an sowie den eben eingerichteten Benutzer. Klicken Sie dann auf „Connect“, um sich etwaige Torrents anzeigen zu lassen, die gerade hoch- oder heruntergeladen werden.

08 Speicherort festlegen

Gehen Sie wieder über Edit > Preferences zum Reiter „Downloads“. Tragen Sie als Download-Ordner das Verzeichnis ein, in das Sie in Schritt 03 die Festplatte eingehängt haben. Aktivieren Sie die Funktion „Auto add .torrents from“ und geben Sie das gewünschte Verzeichnis ein.



09 Deluge beim Booten starten

Mithilfe eines init-Skripts von Ubuntu können Sie festlegen, dass Deluge beim Booten automatisch gestartet werden soll. Laden Sie das Skript mit folgenden Kommando herunter:

```
$ sudo wget -O /etc/default/delugedaemon
http://bit.ly/13nKOSj
```

Öffnen Sie **/etc/default/deluge-daemon** im Editor und ändern Sie den Benutzernamen in den zuvor festgelegten. Speichern Sie die Datei, downloaden Sie dann das komplette init-Skript, und führen Sie ein Update durch:

```
$ sudo wget -O /etc/init.d/delugedaemon
http://bit.ly/13nKKlz
$ sudo chmod 755 /etc/init.d/deluge-
daemon
$ sudo update-rc.d deluge-daemon
defaults
```

Zeitrafferaufnahmen mit dem Pi

So werden Zeitraffervideos mit Ihrer DSLR-Kamera zum Kinderspiel.

WAS SIE BRAUCHEN:
- Steckplatine, Anschlüsse, Jumper-Kabel
- DSLR-Kamera
- passendes Auslöserkabel
- Raspbian mit Python 2.7

Falls Sie denken, dass faszinierende Zeitrafferideos wie die von Vincent Laforet (**laforetvisuals.com**) oder John Eklund (**theartoftimelapse.com**) für den Normal-User unerreichbar sind, sind Sie nicht alleine – aber mithilfe des Raspberry Pi und ein wenig Python-Code rücken solche Aufnahmen in greifbare Nähe. In dieser Anleitung nutzen wir den Pi, um eine handelsübliche Spiegelreflexkamera (eine Canon EOS) anzusteuern und damit perfekte Zeitrafferaufnahmen zu erstellen.

01 Raspberry Pi einrichten

Für diese Anleitung gehen wir davon aus, dass Sie eine aktuelle Version von Raspbian nutzen (erhältlich auf **raspberrypi.org/downloads**). Geben Sie im Terminal Ihres mit Tastatur, Maus und Monitor ausgestatteten Pi ein:

```
sudo apt-get update
```

02 RPi.GPIO installieren

Prüfen Sie dann, ob die Entwicklungsumgebung eingerichtet ist. Sie sollte eigentlich bereits in Raspbian enthalten sein. Mit dem folgenden Befehl im Terminal stellen Sie dies sicher:

```
sudo apt-get install python-dev
```

Dann: `sudo apt-get install python-rpi.gpio`

03 Pi Cobbler einrichten

Die Entwicklungsumgebung läuft, wenden wir uns also der Hardware zu. In dieser Anleitung verwenden wir ein eine simple Steckplatine und einen Adafruit Pi Cobbler (**learn.adafruit.com/adafruit-pi-cobbler-kit/overview**), mit dem wir leichter an die GPIO-Pins des Raspberry Pi kommen. Wie Sie im Bild sehen, sitzt der Cobbler in der Mitte der Steckplatine und wird per Flachbandkabel angeschlossen.

04 Anpassen der Steckplatine

Damit die Kamera per GPIO angesteuert werden kann, muss ein Schaltkreis erstellt werden, der einen Pin des GPIO (hier ist es Pin 23 auf dem Cobbler, der dem tatsächlichen Pin 16 entspricht) mit der Spitze des Auslöserkabels verbindet. Der Widerstand ist dabei optional und nicht notwendig. Der untere Teil des Kabels ist immer die Masse, erden Sie diesen also durch Verbindung des mittleren Pins der Steckerbuchse mit dem „GND"-Pin auf dem Cobbler.

05 Das Tool

Wir haben ein kurzes Python-Programm namens „Time-lapse Photography Tool" geschrieben (siehe rechts), das die Eingabe von Bildanzahl und Verschlussfrequenz erlaubt. Es erstellt aus diesen Informationen einen for-Loop, der über den GPIO-Pin 16 den Auslöser betätigt. Wenn Sie das Projekt im „Straßeneinsatz" verwenden möchten, empfehlen wir die Android-App ConnectBot, die per SSH Steuerbefehle mit dem Pi austauscht. Vergessen Sie aber nicht, mit `sudo python time_lapse_camera.py` Ihr Skript zu starten.

06 Vom Foto zum Video

Ihre Kamera ist also randvoll mit Bildern, die jetzt gesammelt und zu einer Videodatei kombiniert werden sollen. Der Raspberry Pi ist rein technisch dazu in der Lage, aber es empfiehlt sich, die Bilder auf einen separaten Linux-Rechner zu übertragen und dort die Schwerarbeit zu verrichten. Dazu wird FFmpeg verwendet. Öffnen Sie das Terminal in Ihrem Bilder-Ordner und geben Sie ein: `ffmpeg -f image2 -i image%04d.jpg -vcodec libx264 -b 800k video.avi`. Dies setzt voraus, dass libx264 auf Ihrem Rechner installiert ist. Der Befehl „image%04d.jpg" gibt das Dateiformat und die Anzahl der Ziffern im Namen an (in diesem Fall: „image0001.jpg"). Auf **ffmpeg.org/ffmpeg.html** finden Sie eine komplette Dokumentation der Funktionen von FFmpeg, unser Code sollte aber bereits gute Ergebnisse liefern.

Code-Übersicht Zeitrafferfotos

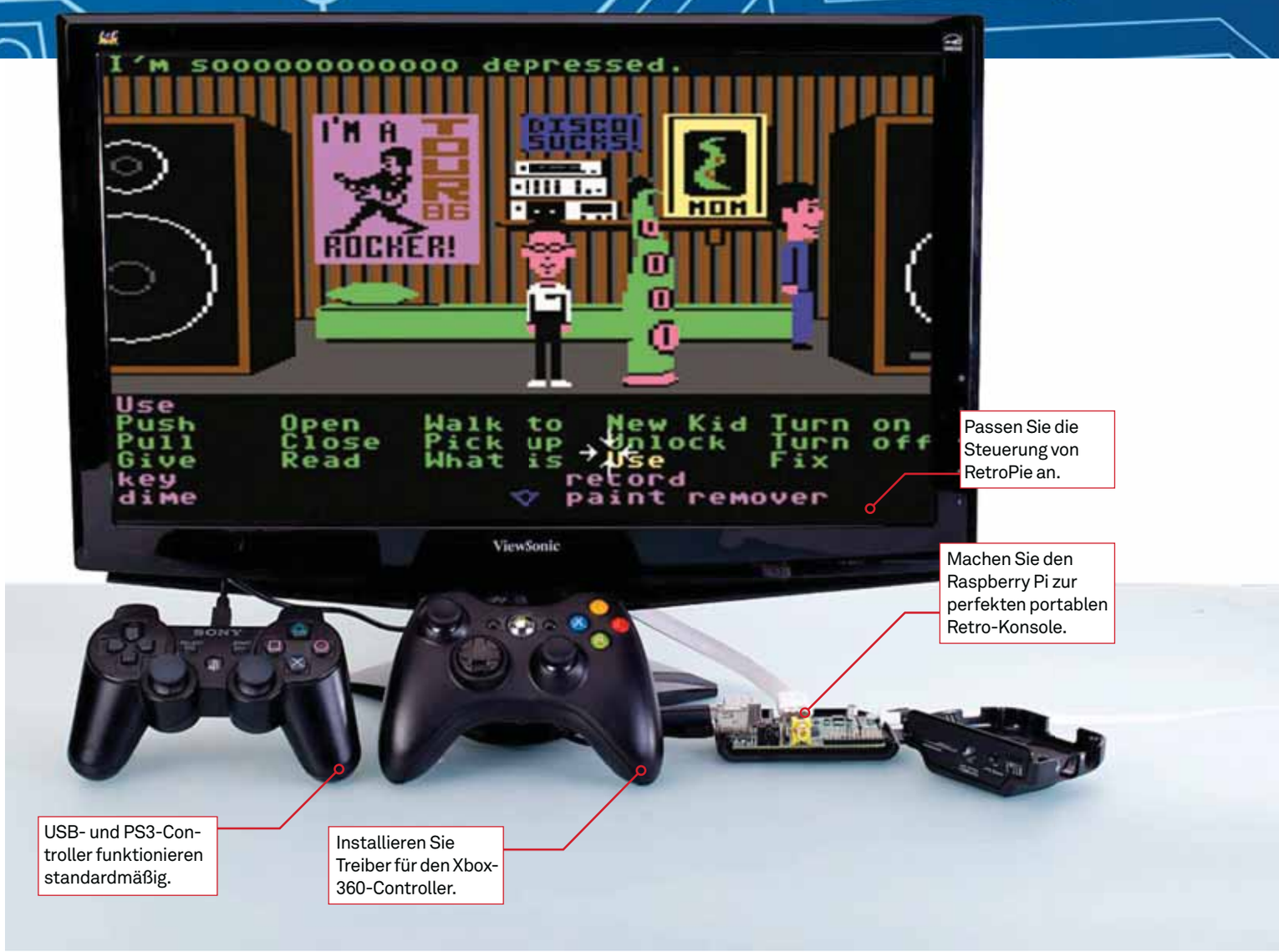
Das simple Skript braucht kaum Imports. Das GPIO-Modul wird für den Auslöser benötigt, das Time-Modul ist für das Festlegen des Aufnahmeintervalls zuständig.

```
import RPi.GPIO as GPIO
import time

print '\n Willkommen!'
print "Wählen Sie Bildanzahl und Aufnahme-Intervall.\n"

def main():
    shots = raw_input('Wie viele Bilder möchten Sie aufnehmen?\n ->')
    interval = raw_input('In welchem Intervall (in Sekunden) möchten Sie auslösen?\n ->')

    if shots.isdigit() and interval.isdigit():
        shots = int(shots)
        interval = int(interval)

        print "Gesamte Aufnahmedauer: %d Minuten.\n" % (shots * interval / 60)
        answer = raw_input('Fortsetzen? (yes/no):')
        confirm = answer.lower() in ['yes', 'y']

        if confirm:
            GPIO.setmode(GPIO.BOARD)
            GPIO.setup(16, GPIO.OUT)
            taken = 1
            print
            print 'Starte Sequenz von %d Bildern' % (shots)

            for i in range(0, shots):
                print
                print 'Bild %d of %d' % (taken, shots)
                taken +=1
                GPIO.output(16, GPIO.HIGH)
                time.sleep(0.5)
                GPIO.output(16, GPIO.LOW)
                time.sleep(interval)
            GPIO.cleanup()
        else:
            print "Versuchen Sie es erneut (oder beenden Sie mit Strg+C):\n"
            main()
    else:
        print "Hoppla - Sie können nur Zahlen eingeben. Versuchen Sie es erneut:\n"
        main()

    print
    print 'Fertig. Danke, dass Sie dieses Tool genutzt haben!\n'
    again = raw_input('Weitere Zeitraffersequenz aufnehmen? (yes/no):\n -> ')
    proceed = again.lower() in ['yes', 'y']

    if proceed:
        main()
    else:
        print '\nSee you next time!\n'
        quit()

if __name__ == '__main__':
    main()
```

Beachten Sie (GPIO.BOARD) – es legt den physikalischen Pin Nummer 16 fest, im Gegensatz zur dokumentierten Nummer (in diesem Fall 23).

In diesem Teil des Skripts werden Aufnahmen gezählt und das Auslöseintervall berechnet.

Am Ende der hauptsächlichen if- und for-Loops folgt eine if/else-Abfrage, die eine neue Aufnahme oder das Programmende einleitet.



Bauen Sie sich eine Retro-Spielkonsole

Holen Sie sich mit RetroPie Ihre Dosis Retro-Gaming und spielen Sie die Klassiker auf Ihrem Raspberry Pi.

Immer mehr Leute bauen sich eigene Arcade-Maschinen oder basteln sich selbst Retro-Spielkonsolen, und der Raspberry Pi ist in Sachen Rechenleistung und Größe dafür perfekt geeignet. Mit dieser Anleitung machen Sie Ihren Pi zu einer wahren Emulationsmaschine!

WAS SIE BRAUCHEN:
» RetroPie
blog.petrockblock.com/retropie/retropie-downloads/

01 RetroPie installieren

Laden Sie das aktuelle Image von der RetroPie-Seite herunter und entpacken Sie es. Kopieren Sie es dann – genau wie bei anderen Raspberry-Pi-Distros – einfach mit folgendem Befehl auf die SD-Karte:

```
$ sudo dd bs=4M if=[Image-Pfad]
of=/dev/[Pfad der SD-Karte]
```

02 Ersteinrichtung

Beim ersten Start werden sie aufgefordert, ein Eingabegerät zu konfigurieren, entweder eine Tastatur oder einen USB-Controller oder einen PS3-Controller. Die Ersteinrichtung legt ein sehr spärliches Eingabeschema fest; später können Sie es mit einem separaten Tool anpassen.

03 Controller kalibrieren

Der eben festgelegte Menü-Button bringt Sie zurück zum Terminal-Interface. Schließen Sie ein USB-Eingabegerät an und geben Sie ein:

```
$ cd RetroPie/emulators/RetroArch/
tools
```

Danach:

```
$ ./retroarch-joyconfig >> ~/
RetroPie/configs/all/retroarch.cfg
```

Folgen Sie dann den Bildschirmanweisungen, um den Controller einzurichten.

04 Xbox-360-Controller

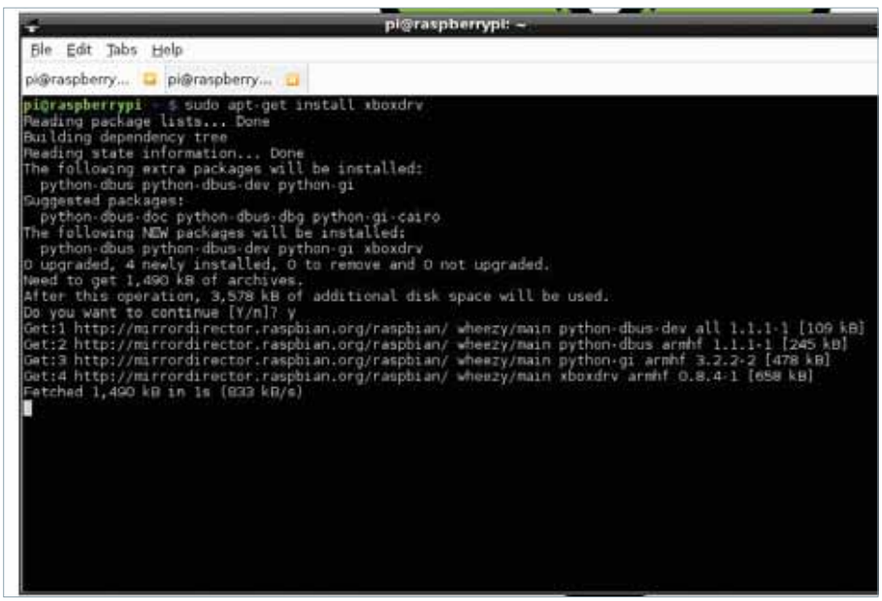
Möchten Sie einen kabelgebundenen oder PC-kompatiblen Xbox-360-Controller verwenden, benötigen Sie Treiber:

```
$ sudo apt-get install xboxdrv
```

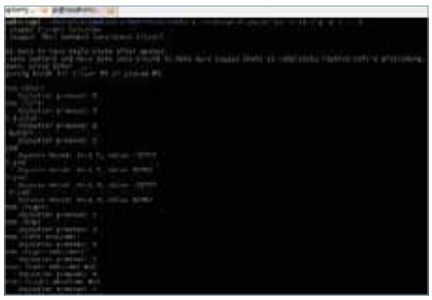
Fügen Sie dann in **/etc/rc.local** – direkt vor exit 0 – Folgendes hinzu:

```
xboxdrv --trigger-as-button --wid 0
--led 2 --deadzone 4000 --silent &
sleep 1
```

Ändern Sie für kabelgebundene Controller das --wid in --id um. Starten Sie neu.



> **Spiele auf den Raspberry Pi zu laden, ist ganz einfach.**



05 Xbox-Controller einbinden

Um den Xbox-360-Controller einzubinden, öffnen Sie die Datei tools mit:

```
$ cd ~/RetroPie/emulators/RetroArch/
tools
```

Geben Sie Folgendes ein:

```
$ ./retroarch-joyconfig -o p1.cfg -p
1 -j 0
```

Fügen Sie die Dateien dann zu RetroArch hinzu:

```
$ sudo cat p*.cfg >> ~/RetroPie/
configs/all/retroarch.cfg
```

Speichern Sie und starten Sie neu.

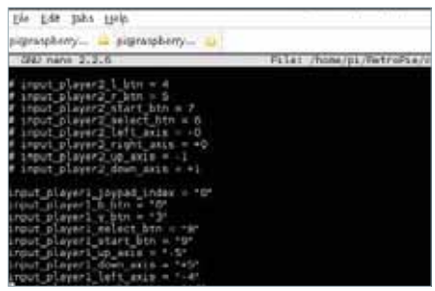
06 Spiele hinzufügen

Spiele auf den Raspberry Pi zu laden, ist ganz einfach. Nehmen Sie die SD-Karte aus dem Pi und öffnen sie sie mit Ihrem PC. Kopieren Sie ROMs oder kompatible Dateien in die jeweiligen Ordner in:

[Pfad der SD-Karte]/home/pi/RetroPie/roms/

07 In RetroPie navigieren

RetroPie ordnet Spiele automatisch dem richtigen Emulator zu und erlaubt direkten Zugriff darauf. Mit links oder rechts wechseln Sie Emulatoren, um jeweils andere Spiele laden zu können.



08 Spielen zu zweit

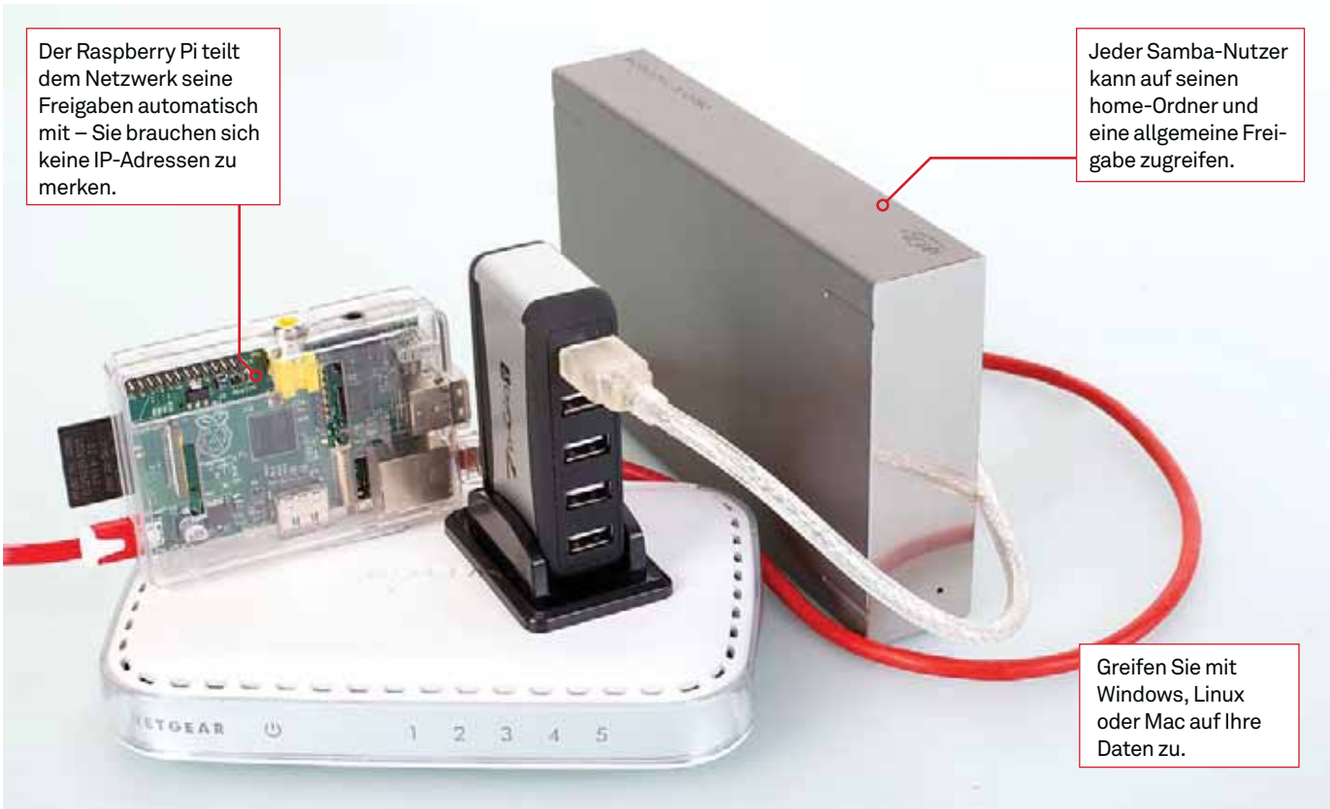
Um Konfigurationsprobleme zu vermeiden, sollten Sie zwei gleiche Controller verwenden. Kopieren Sie ans Dateiende von **~/RetroPie/configs/all/retroarch.cfg** den Code, der mit `input_player1_joypad_index = "0"` beginnt. Ändern Sie jeweils player1 in player2, um den zweiten Controller verwenden zu können.

09 Sicherer Neustart

Manche Emulatoren lassen sich nicht beenden, sondern Sie müssen manuell neu starten. Eine Tastenkombination zum Beenden der Emulatoren schafft Abhilfe: Fügen Sie am Ende von **retroarch.cfg** hinzu:

```
input_enable_hotkey_btn = „X“
input_exit_emulator_btn = „Y“
```

Dabei sind X und Y die jeweilige Anzahl von Tasten auf Ihrem Controller.



Ihre eigene Mini-Cloud

Keine Lust mehr auf Datenübertragung per USB-Stick? Ein Server auf Basis des Raspberry Pi löst dieses Problem!

Ein zentraler Datenserver im Haus macht das Leben leichter. Daten zwischen verschiedenen Rechnern auszutauschen, ist so einfach wie nie zuvor, weil Sie alles in Ihrer eigenen Mini-Cloud im Heimnetzwerk abspeichern können. Der Raspberry Pi ist dafür wie gemacht.

WAS SIE BRAUCHEN:

- einen Router oder Switch (um den Pi mit dem Netzwerk zu verbinden)
- externes Medium (optional) (Festplatte oder USB-Stick)
- USB-Verteiler mit Strom (für externe Festplatte)
- aktuelle Raspbian-Version

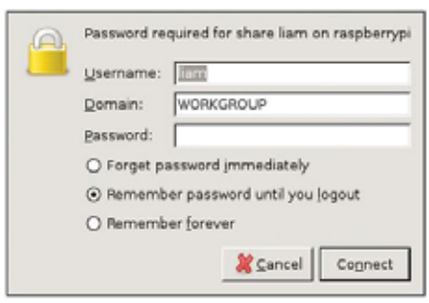
01 Installation der nötigen Software

Loggen Sie sich mit dem Usernamen „pi“ und dem Password „raspberry“ in Raspbian ein. Aktualisieren Sie die Paketinformationen mit `sudo apt-get update`. Installieren Sie dann mit `sudo apt-get install samba` die nötigen Pakete. Samba fungiert als Datenserver und beinhaltet Software, die für die Netzwerkfreigabe zuständig ist.

02 Anschließen externer Medien

Um externe Speichermedien zu verwenden, müssen diese für die Nutzung unter Linux vorbereitet werden. Beachten Sie dabei, dass das Medium bis zur erneuten Formatierung nur unter Linux lesbar ist. Ist das Speichergerät mit dem Rechner verbunden, finden Sie mit dmesg seine Bezeichnung heraus. (Wenn sie dmesg mit `tail -n 3` verwenden, zeigt es die letzten 3 Zeilen an.) Wahrscheinlich ist es /dev/sda.

```
pi@raspberrypi ~ $ dmesg | tail -n 3
[ 1707.371370] sd 1:0:0:0: [sda] No
Caching mode page present
[ 1707.371403] sd 1:0:0:0: [sda]
Assuming drive cache: write through
[ 1707.371422] sd 1:0:0:0: [sda]
Attached SCSI removable disk
```



03 Formatieren externer Medien

Mit Parted können Sie nun eine neue Partitionstabelle auf Ihrem Medium anlegen und eine Partition erstellen, die den gesamten Speicher umfasst. Laden Sie mit partprobe die Partitionstabelle neu und erstellen Sie ein ext4-Dateisytem auf der neuen Partition. Alle Daten auf Ihrem Medium werden dadurch gelöscht.

```
pi@raspberrypi ~ $ sudo parted /dev/
sda
(parted) print
Disk /dev/sda: 16.1GB
(parted) mktable msdos
Warning: The existing disk label on
/dev/sda will be destroyed and all
data on this disk will be lost. Do
 you want to continue?
Yes/No? Yes
(parted) mkpart
Partition type?  primary/extended?
primary
File system type?  [ext2]? ext2
Start? 0GB
End? 16GB
(parted) quit
pi@raspberrypi ~ $ sudo partprobe
pi@raspberrypi ~ $ sudo mkfs.ext4 /
dev/sda1
```

04 Einbinden externer Medien

Verlegen Sie jetzt **/home** auf Ihr externes Medium. Dazu muss es unter **/mnt/storage** eingebunden, die Dateien von **/home** nach **/mnt/storage** verschoben und die Partition dann als **/home** eingebunden werden.

```
pi@raspberrypi ~ $ sudo su
root@raspberrypi:/home/pi# cd /
root@raspberrypi:/# mkdir /mnt/home
root@raspberrypi:/# mount /dev/sda1 /
mnt/home/
root@raspberrypi:/# mv /home/* /mnt/
home/
root@raspberrypi:/# umount /mnt/home/
root@raspberrypi:/# rmdir /mnt/home/
root@raspberrypi:/# echo '/dev/sda1 /
home ext4 defaults 0 1' >> /etc/fstab
root@raspberrypi:/# mount -a
root@raspberrypi:/# mount | grep sda1
/dev/sda1 on /home type ext4
(rw,relatime,data=ordered)
root@raspberrypi:/# ls /home
lost+found  pi
root@raspberrypi:/# exit
```

05 Nutzer für Samba anlegen

Für den Zugriff auf die Samba-Freigaben müssen Sie Nutzer mit Login erstellen. Folgen Sie der Anleitung für jeden Nutzer, den Sie anlegen möchten. Der Schritt usermod ist nur notwendig, wenn der Nutzer Zugriff auf allgemeine Freigaben haben soll.

```
pi@raspberrypi ~ $ sudo adduser liam
Adding user `liam' ...
Adding new group `liam' (1002) ...
Adding new user `liam' (1001) with
group `liam' ...
Creating home directory `/home/liam'
...
Copying files from `/etc/skel' ...
sh: 0: getcwd() failed: No such file
or directory
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated
successfully
Changing the user information for
liam
Enter the new value, or press ENTER
for the default
   Full Name []: Liam Fraser
   Room Number []:
   Work Phone []:
   Home Phone []:
   Other []:
Is the information correct? [Y/n] Y

pi@raspberrypi ~ $ sudo usermod -a
-G sambashare liam

pi@raspberrypi ~ $ sudo pdbedit -a
-u liam
new password:
retype new password:
Unix username:        liam
Full Name:            Liam Fraser
Home Directory:       \\raspberrypi\
liam
```

06 Allgemeine Freigaben

Erstellen Sie nun ein Verzeichnis, das alle Nutzer der Gruppe **sambashare** öffnen und bearbeiten können. Wenn Sie den Ordner mit setgid freigeben (chmod g+s), werden neue Dateien und Unterverzeichnisse gleich mit der Gruppen-ID versehen. Das bedeutet, dass jeder Nutzer in der Gruppe sambashare sie öffnen und bearbeiten kann.

```
pi@raspberrypi /home $ sudo mkdir /
home/allusers
pi@raspberrypi /home $ sudo chown
root:sambashare /home/allusers/
pi@raspberrypi /home $ sudo chmod
770 /home/allusers/
pi@raspberrypi /home $ sudo chmod
g+s /home/allusers/
```

07 Konfiguration von Samba

Öffnen Sie mit sudo **/etc/samba/smb.conf** im Editor. Scrollen Sie zum Abschnitt „Authentication" und kommentieren Sie die Zeile `security = user` aus. Setzen Sie dann unter „Share Definitions" im Abschnitt [homes] `read only` auf no. Fügen Sie schließlich am Dateiende einen Abschnitt für den Freigabeordner hinzu:

```
[allusers]
    comment = Shared Folder
    path = /home/allusers
    read only = no
    guest ok = no
    browseable = yes
    create mask = 0770
    directory mask = 0770
```

Starten Sie dann Samba neu:
`sudo /etc/init.d/samba restart`

08 Zugriff auf die Freigaben

Der Raspberry Pi teilt dem Netzwerk seine Freigaben automatisch mit. Jeder Nutzer hat Zugriff auf seinen **home**-Ordner und das Verzeichnis **allusers**. Ist der Nutzer nicht in der Gruppe sambashares, hat er keinen Zugriff. Die Syntax für den Samba-Zugriff lautet wie folgt:

Windows: `\\raspberrypi\share_name`

Mac: `smb://raspberrypi/share_name`

Linux: `smb://raspberrypi/share_name`

Der Zusatz share_name ist optional – Sie können sich auch einfach durchklicken. Der Zugriff sollte sogar ohne Syntax möglich sein, indem Sie einfach in der Netzwerkübersicht Ihres Dateimanagers „raspberrypi" auswählen.

Überwachungskamera

Möchten Sie einen Raum ganz bequem über Ihren Webbrowser überwachen? Sie brauchen nur einen Raspberry Pi und eine Webcam!

In diesem Artikel erfahren Sie, wie Sie mithilfe von MJPG-Streaming-Software Videosignale von einer oder sogar gleichzeitig von mehreren Webcams über den Raspberry Pi in Ihren PC-Browser oder auf Ihr Mobilgerät streamen können. Die Streams lassen sich außerdem aufnehmen.

WAS SIE BRAUCHEN:

- einen Router oder Switch in Ihrem Netzwerk, um den Raspberry Pi anzuschließen
- eine Linux-kompatible Webcam (zum Beispiel Logitech C270)
- einen USB-Hub mit Stromversorgung (da die Stromversorgung der Webcam allein nicht ausreichen könnte)
- die aktuelle Version von Raspbian (Download: raspberrypi.org/downloads)

01 Netzwerkinformationen

Zunächst müssen wir einige Informationen über das Netzwerk herausbekommen, in dem der Raspberry Pi sich befindet, um ihm eine statische IP-Adresse zuweisen zu können.

```
pi@raspberrypi ~ $ ip addr show dev eth0 | grep inet
    inet 172.17.173.94/24 brd 172.17.173.255 scope global eth0
pi@raspberrypi ~ $ ip route | grep default
default via 172.17.173.1 dev eth0
pi@raspberrypi ~ $ cat /etc/resolv.conf
nameserver 172.17.173.1
```

02 Statische IP-Adresse zuweisen

Auf Basis der so ermittelten Konfiguration können wir die statische IP-Adresse zuweisen. Verändern Sie mit einem Editor wie nano in **/etc/network/interfaces** folgende Zeile:

```
iface eth0 inet dhcp
```

Und zwar, indem Sie eine solche Konfiguration stattdessen einfügen (eigene Werte verwenden!):

```
iface eth0 inet static
    address 172.17.173.94
    netmask 255.255.255.0
    network 172.17.173.0
    broadcast 172.17.173.255
    gateway 172.17.173.1
```

03 Software installieren

Rebooten Sie den Pi und loggen Sie sich mit „pi“ als User und „raspberry“ als Passwort ein. Die MJPG-Streaming-Software liegt nicht als fertiges Paket für den Pi vor, daher müssen wir sie selber kompilieren. Aktualisieren Sie den Paketindex mit dem Befehl `sudo apt-get update`. Wir müssen Subversion installieren, um den Quellcode herunterzuladen, sowie libjpeg und imagemagick, die vom MJPG-Streamer benötigt werden. Tippen Sie:

```
sudo apt-get install subversion libjpeg8-dev imagemagick
```

04 MJPG-Streamer installieren

Dann den MJPG-Streamer downloaden und installieren:

```
pi@raspberrypi ~ $ svn checkout svn://svn.code.sf.net/p/mjpg-streamer/code/mjpg-streamer-code
pi@raspberrypi ~ $ cd mjpg-streamer-code/mjpg-streamer
pi@raspberrypi ~/mjpg-streamer-code/mjpg-streamer $ make clean all
```

05 Testen

Starten Sie so den MJPG-Streamer:

```
export LD_LIBRARY_PATH=.
./mjpg_streamer -i “input_uvc.so” -o “output_http.so -w ./www”
```

Exportieren Sie anschließend die Library-Path-Variable zum aktuellen Verzeichnis (.), damit die Input- und Output-Plugins verwendet werden können. Geben Sie die IP-Adresse des Raspberry Pi, gefolgt von :8080, in die Adresszeile Ihres Browsers ein und klicken Sie auf den Stream-Tab. Sollte der Stream nicht angezeigt werden, testen Sie das JavaScript-Beispiel. Dieses sollte sogar auf Android-Geräten laufen.

06 Streamer beim Booten starten

Editieren Sie die Datei **/etc/rc.local** (dazu müssen Sie sudo verwenden) wie folgt, aber achten Sie darauf, dass am Ende das „exit 0“ stehen bleibt.

```
export STREAMER_PATH=/home/pi/mjpg-streamer-code/mjpg-streamer
export LD_LIBRARY_PATH=$STREAMER_PATH
$STREAMER_PATH/mjpg_streamer -i “input_uvc.so” -o “output_http.so -w $STREAMER_PATH/www” &s
```

Booten Sie den Pi erneut. Spätestens jetzt sollte der Stream angezeigt werden.

Auf eLinux.org finden Sie eine Liste von Webcams, die mit dem Pi kompatibel sind.

07 Stream aufnehmen

Sie können leicht einen MJPG-Stream herunterladen und ihn mithilfe von VLC in ein nützlicheres Format umwandeln:

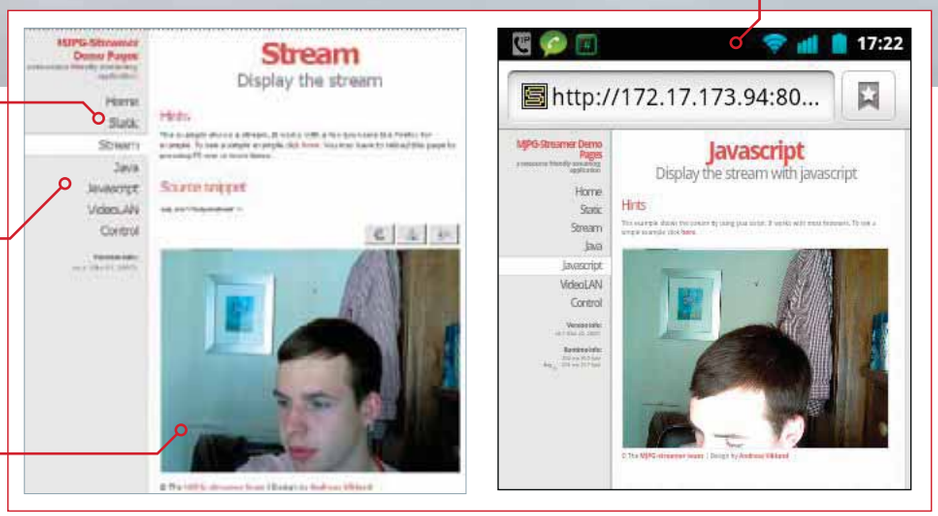
```
cvlc http://172.17.173.94:8080/?action=stream --sout file/mp4:stream.mp4
```

Oder Sie downloaden die URL des Streams mithilfe von wget. Tippen Sie dazu:

```
wget http://172.17.173.94:8080/?action=stream
```

“Streamen Sie Videosignale von einer Webcam in Ihren Browser.”

Auch auf Mobilgeräten lassen sich die Streams ansehen.



Mit dem MJPG-Streamer haben Sie auch die Möglichkeit, Standbilder zu erstellen.

Die Software bietet diverse Optionen, wie Sie auf die Streams zugreifen können. Jede davon ist mit einem Beispiel belegt.

Streamen Sie von der Webcam direkt zum Webbrowser.

Um allerdings diese Rohdaten wiederum als Video laufen lassen zu können, müssen Sie die Bildrate pro Sekunde festlegen.

08 Mehrere Streams parallel

Hier sehen Sie, wie Sie mit HTML mehrere Streams auf einer Webseite anzeigen. Die Streams könnten etwa von diversen Raspberry Pis stammen, die Sie im Haus platziert haben.

```
<html>
<h1>Streaming Example</h2>
<table border="1">
<tr>
<td><h2>Stream 1</h2></td>
<td><h2>Stream 2</h2></td>
</tr>
<tr>
<td><img src="http://172.17.173.94:8080/
?action=stream"/></td>
<td><img src="http://172.17.173.94:8080/
?action=stream"/></td>
</tr>
</table>
</html>
```

09 Ausblick: SSL

Das Setup, das wir beschrieben haben, ist noch nicht perfekt. So gibt es weder Authentifizierung noch SSL-Verschlüsselung – daher sollten die Streams nicht über das Internet geteilt werden. Das HTTP-Modul des MJPG-Streamers ist recht simpel, somit können Sie SSL mithilfe eines Reverse Proxy wie Pound bekommen. Eine einzelne Instanz von Pound kann mehrere Raspberry Pis SSL-mäßig abdecken, die unterschiedlichen Geräte würden dann zum Beispiel mittels „stream1“, „stream2“ etc. in der URL angesprochen werden.

Pi als VoIP-Server

So verwenden Sie Ihren Raspberry Pi als Voice-over-IP-Server. Machen Sie den Minirechner zur häuslichen Telefonanlage!

RasPBX ist eine Lösung, die die weitverbreitete Internet-Telefonie-Software Asterisk und dessen nutzerfreundliche Front-End-Option FreePBX auf den Raspberry Pi bringt. Es gibt Voice-over-IP-Clients (VoIP) für viele Plattformen einschließlich Hardware, die VoIP mit konventionellen Telefonleitungen kompatibel macht. Jedes angeschlossene Gerät bekommt eine Telefonnummer zugeordnet und kann andere Client-Geräte anrufen. Weiterhin sind Konferenzschaltungen möglich.

WAS SIE BRAUCHEN:

- einen Router oder Switch in Ihrem Netzwerk, an den der Raspberry Pi angeschlossen werden kann
- Smartphones, Computer, VoIP-Telefone als Client-Geräte im selben Netzwerk
- USB-Hub mit eigener Stromversorgung (falls Sie eine externe Festplatte verwenden)
- das neueste RasPBX-Image (Download: raspberry-asterisk.org/downloads)

01 Installation und Einrichtung

RasPBX ist im Prinzip startklar, sobald Sie das Image auf die SD-Karte gepackt haben. Es gibt ein Web-Interface für die Konfiguration, und per SSH ist ein Fernzugriff möglich. Da wir einen Server einrichten, loggen wir uns mit dem Usernamen „root“ und dem Passwort „raspberry“ ein und ändern die IP-Adresse in eine statische, sodass wir immer wissen, wo im Netzwerk der Pi sich befindet.

```
root@raspbx:~# ip addr show dev eth0
| grep inet
    inet 172.17.173.94/24 brd
172.17.173.255 scope global eth0
    inet6
fe80::ba27:ebff:fef3:9016/64 scope
link
root@raspbx:~# ip route | grep
default
default via 172.17.173.1 dev eth0
root@raspbx:~# cat /etc/resolv.conf
nameserver 127.0.0.1
nameserver 8.8.8.8
nameserver 8.8.8.4
```

02 Statische IP-Adresse zuweisen

Nun da wir die Netzwerkkonfiguration kennen, können wir die statische IP-Adresse zuweisen. Öffnen Sie dazu die Datei **/etc/network/interfaces** in einem Editor wie beispielsweise nano und ersetzen Sie die Zeile

```
iface eth0 inet dhcp
```

durch folgende Einträge (eigene Werte verwenden!):

```
iface eth0 inet static
address 172.17.173.94
netmask 255.255.255.0
network 172.17.173.0
broadcast 172.17.173.255
gateway 172.17.173.1
```

Da die Google-Nameserver verwendet werden, brauchen wir uns über DNS keine Gedanken zu machen. Laden Sie die neue Netzwerkkonfiguration mit diesem Kommando:

```
sudo /etc/init.d/networking restart
```

03 Web-Interface

Geben Sie die statische IP-Adresse des Raspberry Pi in die Adresszeile des Browsers ein. Daraufhin landen Sie auf dem Web-Interface von FreePBX für die Einrichtung von Asterisk. Klicken Sie auf die Schaltfläche „FreePBX Administration“ und geben Sie dann den Usernamen „admin“ und das Passwort „admin“ ein. Über „User Control Panel“ können Benutzer Ihre Sprachnachrichten abhören.

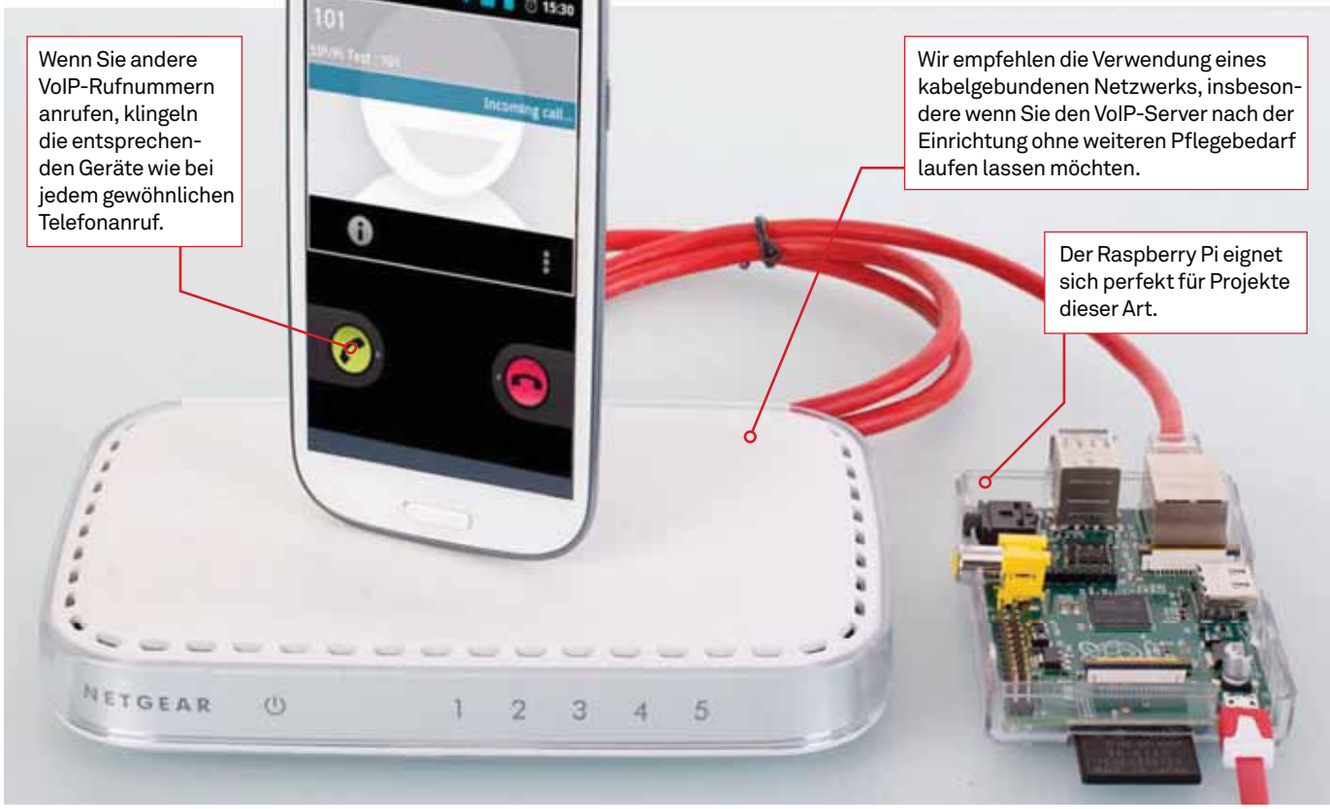
User Extension	100
Display Name	100

04 Nebenstellen einrichten

Für jedes Gerät, das angeschlossen werden soll, muss eine separate Nebenstelle eingerichtet werden. Gehen Sie hierfür im Menü zu Applications > Extensions. Wählen Sie „Generic SIP Device“ als Gerätetyp aus und klicken Sie auf „Submit“. Jeder Nebenstelle ist eine Durchwahlnummer zugeordnet. Sie müssen einen Zugangscode (Secret) eingeben, der dem Gerät die Verbindung gestattet. Dieser wird automatisch generiert, Sie können ihn aber abändern. Es lassen sich beliebig viele Nebenstellen einrichten.

05 Konferenzschaltungen

Gehen Sie im Menü zu Applications > Conferences. Wählen Sie eine Nummer und einen Namen für die Konferenz sowie eine PIN, falls Sie möchten, das Teilnehmer sich damit



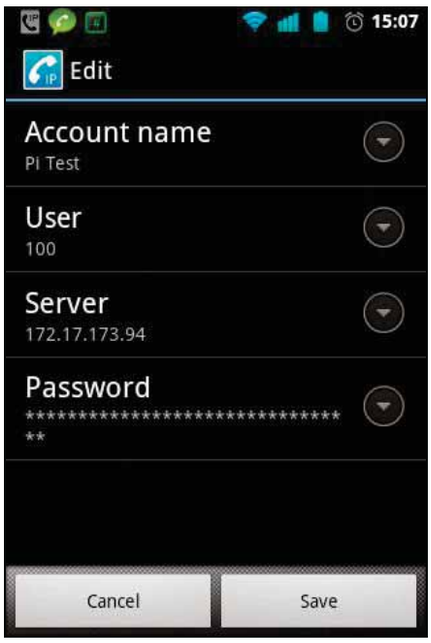
anmelden müssen, um an der Konferenzschaltung teilnehmen zu können. Speichern Sie die Änderungen mit „Submit“.

06 Wirksamkeit der Änderungen

Änderungen wie die Einrichtung neuer Nebenstellen oder Konferenzen werden erst wirksam, wenn Sie den Button „Apply Config“ anklicken. Denken Sie also stets an diesen Schritt, bevor Sie das Web-Interface schließen.

07 Anlage testen

Wenn Sie ein Android-Smartphone benutzen, empfehlen wir den kostenlosen Client CSipSimple (erhältlich auf Google Play), für Linux-Clients ist Linphone geeignet. Wir nehmen zum Test zwei Android-Smartphones. Wenn Sie ein Benutzerkonto in CSipSimple anlegen, scrollen Sie runter zur Abteilung „Generic Wizards“ und wählen Sie „Basic“. Geben Sie Ihre Infos ein (siehe Abbildung). Sind zwei Geräte eingerichtet, versuchen Sie, mit dem einen das andere anzurufen.



> **Jedes Gerät bekommt eine Telefonnummer zugeordnet und kann andere Client-Geräte anrufen.**

08 Konferenzschaltung testen

Rufen Sie die Konferenznummer, die Sie in Schritt 05 eingerichtet haben, mit einem Ihrer Geräte an. Geben Sie nötigenfalls die PIN ein und danach ein „#“. Sie erhalten zunächst den Hinweis, dass Sie der einzige Teilnehmer der Konferenz sind, und bekommen eine Meldung, wenn weitere Teilnehmer dazustoßen oder die Konferenz verlassen.

Machen Sie den Pi zum Access Point

Mithilfe eines WLAN-Adapters können Sie einen Raspberry Pi in einen Wireless Access Point für andere Geräte verwandeln.

WAS SIE BRAUCHEN:
- » Raspberry Pi (Modell B)
- » einen kompatiblen WLAN-Adapter adafruit.com/products/814
- » Raspbian raspberrypi.org/downloads

Dank seiner geringen Abmessungen ist der Raspberry Pi perfekt geeignet, um ihn als portablen Funk-Router und Access Point einzusetzen, zum Beispiel im Hotel oder beim Besuch von Freunden ohne WLAN.

01 Raspbian installieren

Bei diesem Projekt können wir unseren Access Point mit Raspbian betreiben. Installieren Sie das Raspbian-Image auf eine SD-Karte, nehmen Sie das Setup vor und aktivieren Sie dabei SSH.

02 Mittels SSH verbinden

Ermitteln Sie die IP-Adresse des Raspberry Pi, indem Sie `ifconfig` in die Kommandozeile eingeben, und notieren Sie sie. Schalten Sie anschließend den Pi aus, stecken Sie den Wireless-Adapter ein, und fahren Sie den Pi wieder hoch. Tippen Sie danach an einem mit dem Netzwerk verbundenen Computer in dessen Terminal:

```
$ ssh [User]@[IP-Adresse]
```

Geben Sie Ihr Passwort ein, um sich einzuloggen.

03 DHCP installieren

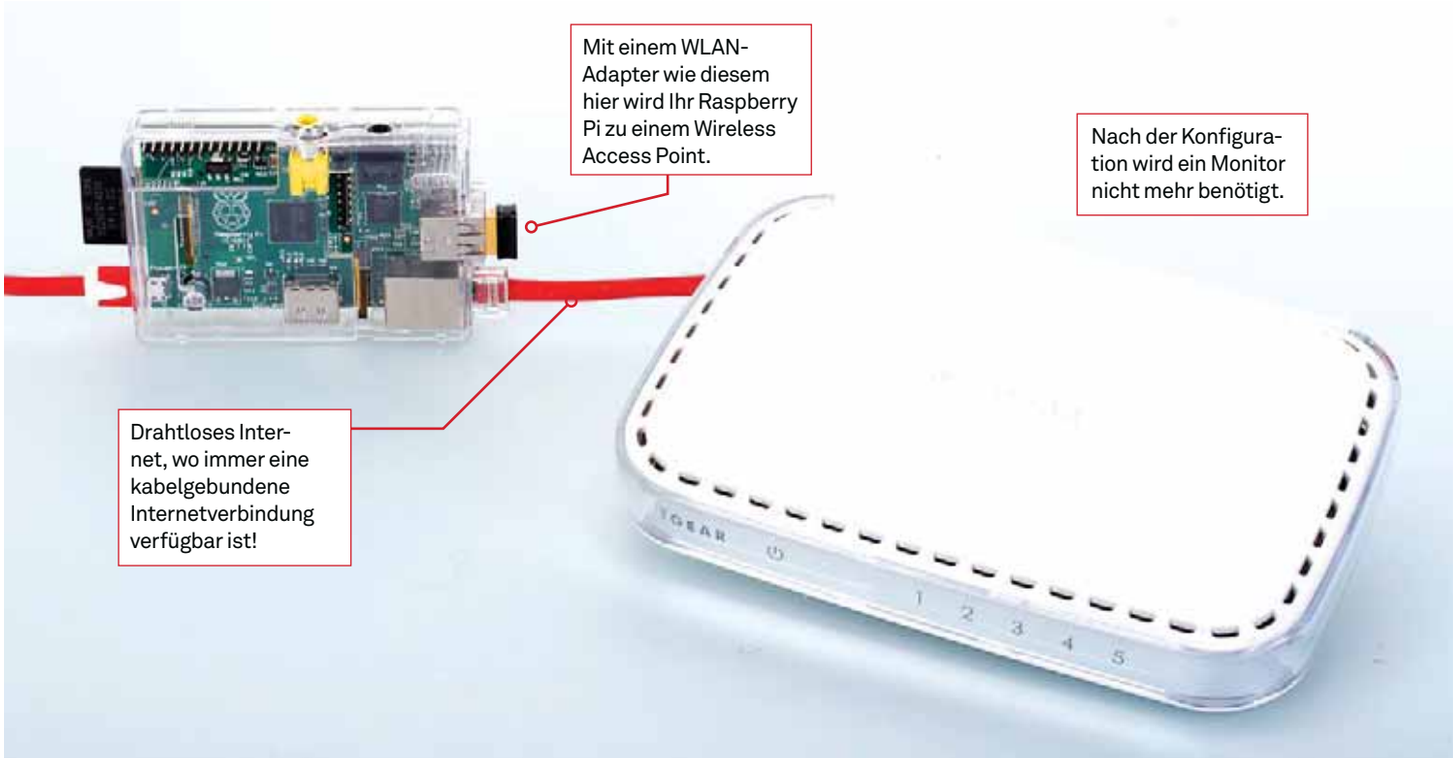
Zurück zum Raspberry Pi. Tippen Sie:

```
$ sudo apt-get install hostapd isc-dhcp-server
```

Damit installieren Sie einen DCHP-Server auf dem Pi. Um ihn einzustellen, editieren Sie die Konfigurationsdatei wie folgt:

```
$ sudo nano /etc/dhcp/dhcpd.conf
```

In der **dhcpd.conf** setzen Sie jeweils ein # vor die zwei Zeilen mit „option domain-name“ (um sie auszukommentieren) und entfernen das # vor dem „authoritative“ einige Zeilen darunter.



04 Server-Adresse

Setzen Sie die nachfolgenden Zeilen an das Ende der Konfigurationsdatei, dann speichern und schließen Sie die Datei.

```
subnet 192.168.42.0 netmask
255.255.255.0 {
range 192.168.42.10 192.168.42.50;
option broadcast-address
192.168.42.255;
option routers 192.168.42.1;
default-lease-time 600;
max-lease-time 7200;
option domain-name “local”;
option domain-name-servers 8.8.8.8,
8.8.4.4;
}
```

05 WLAN deaktivieren

Nun müssen wir noch eine weitere Bearbeitung vornehmen. Öffnen Sie die entsprechende Datei mit:

```
$ sudo nano /etc/default/isc-dhcp-
server
```

Setzen Sie „INTERFACES“ auf „wlan0“ und speichern Sie. Öffnen Sie dann die Datei **interfaces**:

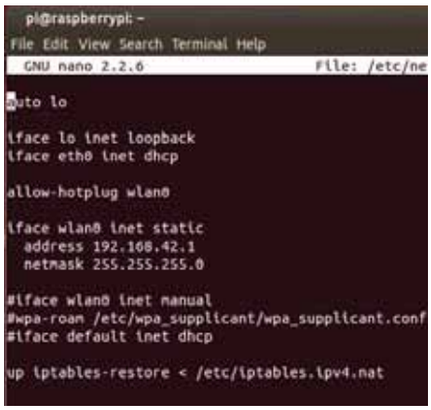
```
$ sudo nano /etc/network/interfaces
```

Vor die Zeile mit „iface wlan0“, die folgenden mit „wpa roam“ und „iface default“ sowie alle anderen, die „wlan0“ betreffen, gehört jeweils ein #.

06 Zugang aktivieren

Geben Sie nach der Zeile „allow-hotplug wlan0“ das Folgende ein:

```
iface wlan0 inet static
```



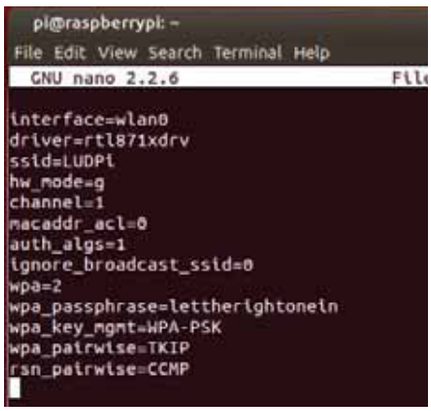

```
address 192.168.42.1
netmask 255.255.255.0
```

Speichern und schließen Sie die Datei, und legen Sie dann die Adresse von wlan0 fest, und zwar mit dem Kommando:

```
$ sudo ifconfig wlan0 192.168.42.1
```

Jetzt brauchen wir eine neue Datei, um das drahtlose Netzwerk anlegen zu können. Erstellen Sie diese Datei wie folgt:

```
$ sudo nano /etc/hostapd/hostapd.conf
```



07 Drahtloses Nzetzwerk

Fügen Sie diesen Inhalt in die neue Datei ein:

```
interface=wlan0
driver=rtl871xdrv
ssid=[access point name]
hw_mode=g
channel=1
macaddr_acl=0
auth_algs=1
ignore_broadcast_ssid=0
wpa=2
wpa_passphrase=[password]
wpa_key_mgmt=WPA-PSK
wpa_pairwise=TKIP
rsn_pairwise=CCMP
```

> **“Dank seiner Abmessungen ist der Pi perfekt, um als portabler Access Point eingesetzt zu werden.”**

Speichern und schließen Sie die Datei. Nun müssen wir die **hostapd** so bearbeiten, dass auf die neue Datei verwiesen wird. Öffnen Sie sie mit dem Befehl

```
$ sudo nano /etc/default/hostapd
```

und fügen Sie hinzu:

```
/etc/hostapd/hostapd.conf to DAEMON_
CONF=””
```

08 Netzwerkadressierung

Geben Sie ins Terminal ein:

```
$ sudo nano /etc/sysctl.conf
```

Fügen Sie `net.ipv4.ip_forward=1` ans Ende der Datei an, speichern Sie sie, und geben Sie diesen Befehl ein:

```
$ sudo sh -c “echo 1 > /proc/sys/
net/ipv4/ip_forward”
```

Dann brauchen wir noch die folgenden drei Kommandos, um sicherzustellen, dass das Internet korrekt weitergeleitet wird:

```
sudo iptables -t nat -A POSTROUTING
-o eth0 -j MASQUERADE
sudo iptables -A FORWARD -i
eth0 -o wlan0 -m state --state
RELATED,ESTABLISHED -j ACCEPT
sudo iptables -A FORWARD -i wlan0 -o
eth0 -j ACCEPT
```

09 Abschluss

Damit nach dem Neustart alles funktioniert, tippen Sie:

```
$ sudo sh -c “iptables-save > /etc/
iptables.ipv4.nat”
```

Hängen Sie `up iptables-restore < /etc/iptables.ipv4.nat` an das Ende der Datei **/etc/network/interfaces** und setzen Sie den Daemon auf:

```
sudo service hostapd start
sudo service isc-dhcp-server start
sudo update-rc.d hostapd enable
sudo update-rc.d isc-dhcp-server enable
```

Sprachsteuerung

Sagen Sie dem Raspberry Pi, was er tun soll.

Seit Captain Jean-Luc Picard zum ersten Mal mit dem Bordcomputer der Enterprise gesprochen hat, faszinierte viele Menschen die Vorstellung, nur mithilfe der eigenen Stimme einen Rechner steuern zu können. Heute ist das Realität, und auch beim Raspberry Pi geht es.

WAS SIE BRAUCHEN:

- ein kompatibles (!) USB-Mikrofon (etwa das Zoom H2)
- einen USB-Hub mit eigener Stromversorgung
- zusätzliche Software github.com/StevenHickson/PiAUISuite

01 Software herunterladen

Zunächst müssen Sie das Softwarepaket von Github herunterladen. Unter dem Link **github.com/StevenHickson/PiAUISuite/archive/master.zip** finden Sie eine Programmsammlung für den Raspberry Pi. Wir benötigen für dieses Projekt allerdings nur das Feature **voicecommand.** Navigieren Sie im Terminal zum Download-Verzeichnis und entpacken Sie das Archiv mit `unzip master.zip.` Heraus kommt ein Verzeichnis namens **PiAUISuite-master.**

02 Abhängigkeiten downloaden

Weiter benötigen Sie noch einige Abhängigkeiten (zusätzliche Programmteile) für das Prrojekt. Tippen Sie ins Terminal:

```
sudo apt-get install libboost1.50-dev libboost-regex1.50-dev youtubedl axel curl xterm libcurl4-gnutlsdev mpg123 flac sox libboost1.46
```

Der Download wird einige Zeit in Anspruch nehmen, Sie können sich also eine Pause gönnen.

03 Datei ausführbar machen

Um die Werkzeuge zu installieren, gehen Sie zu **PiAUISuite-master/Install.** Wir müssen die Datei **InstallAUISuite.sh** für alle Benutzer ausführbar machen, das geht so:

```
chmod 777 InstallAUISuite.sh
```

Jetzt können wir die Installation durchführen:

```
sudo ./ InstallAUISuite.sh
```

```
File  Edit  Tabs  Help
root@raspberrypi:/home/pi/PiAUISuite/Install# ./InstallAUISuite.sh
Installing AUI Suite by Steven Hickson
If you have issues, visit stevenhickson.blogspot.com or email me@stevenhickson.com
Install dependencies? y/n
y
Reading package lists... Done
Building dependency tree
Reading state information... Done
axel is already the newest version.
curl is already the newest version.
flac is already the newest version.
libboost1.46-dev is already the newest version.
libcurl4-gnutls-dev is already the newest version.
mpg123 is already the newest version.
sox is already the newest version.
xfonts-utils is already the newest version.
xfonts-utils set to manually installed.
xterm is already the newest version.
youtube-dl is already the newest version.
Some packages could not be installed. This may mean that you have
requested an impossible situation or if you are using the unstable
distribution that some required packages have not yet been created
or been moved out of Incoming.
The following information may help to resolve the situation:

The following packages have unmet dependencies:
 libboost-regex1.50-dev : Depends: libboost1.50-dev (= 1.50.0-1) but it is not going to be installed
E: Unable to correct problems, you have held broken packages.
Install playvideo? y/n
```

04 Installation

Wenn das Skript durchläuft, werden Sie gebeten, Abhängigkeiten zu installieren. Bejahen Sie dies, und die Installation geschieht automatisch. Falls eine Fehlermeldung bezüglich libboost erscheint, ignorieren Sie diese. Als Nächstes kommt die Nachfrage, welche Tools Sie installieren möchten. Lehnen Sie alle Angebote ab bis auf das letzte, voicecommand.

05 Konfigurierung

Verneinen Sie als Nächstes die Frage, ob Sie eine automatische Konfigurierung von voicecommand wünschen. Damit endet das Skript. Editieren Sie nun selber die Datei mit folgendem Befehl:

```
voicecommand -e
```

Lesen Sie den angezeigten Text und drücken Sie dann die Eingabetaste.

06 Befehle definieren

Sie sehen nun eine leere Nano-Sitzung. Befehle können Sie im Format „[Spracheingabe wort]==[Auszuführender Befehl]“ definieren. Ein Beispiel:

```
Internet==midori &
```

Damit öffnen Sie einen Browser, wenn Sie „Internet“ sagen. Oder, um ein Terminal aufzurufen:

```
Terminal==xterm &
```

Geben Sie diese Beispielzeilen ein, speichern Sie mit Strg+O, und schließen Sie die Editor-Sitzung. Das „&“ am Ende der Zeilen bedeutet übrigens, dass der Befehl im Hintergrund laufen soll.

```
File  Edit  Tabs  Help
  GNU nano 2.2.6

!keyword==computer
!response==Online
bash==xterm &
Internet==midori &
Audio==mplayer /home/pi/linuxoutlaws314.mp3
Status==htop
Radio 1==mplayer -playlist http://bbc.co.uk/radio/listen/live/r1.asx
Radio 6==mplayer -playlist http://bbc.co.uk/radio/listen/live/r6.asx
!continuous==0
```

> **Das Programm wartet auf ein Kennwort.**

07 Mikrofon anschließen

Schließen Sie den USB-Hub mit eigener Stromversorgung an und stecken Sie ein USB-Mikrofon (zum Beispiel das empfehlenswerte Zoom H2) ein. Geben Sie dem Pi ein paar Sekunden Zeit und überprüfen Sie dann mittels `lsusb`, ob das Mikrofon gelistet wird. Das Mikro müsste mit dem Raspberry Pi funktionieren, aber wenn Sie das zuerst testen möchten, installieren Sie Audacity (mit `sudo apt-get install audacity`) und machen Sie eine Probeaufnahme.

08 Konfiguration finalisieren

Als letzten Schritt der Einrichtung müssen wir noch einige Dateien für jeden Benutzer ausführbar machen. Tun Sie dies mit den folgenden Kommandos:

```
sudo chmod 777 /usr/bin/voicecommand
sudo chmod 777 /usr/bin/speech-recog.sh
sudo chmod 777 /usr/bin/X11/speech-recog.sh
```

09 Test

Um den kontinuierlichen Modus von voicecommand zu aktivieren, tippen Sie `voicecommand -c` ins Terminal. Jetzt wartet das Programm auf ein gesprochenes Kennwort, um loszulegen. Standardmäßig lautet das Kennwort „Pi“. Sprechen Sie dies ins Mikrofon, und voicecommand sollte die Meldung „Found audio“ ausgeben und dann „Recording WAVE ‚stdin‘“. Das bedeutet, dass das Programm Ihre Spracheingabe erkannt hat. Sagen Sie nun „Internet“. Daraufhin müsste der Webbrowser Midori anspringen. Falls ja: Glückwunsch, Sie haben jetzt einen Raspberry Pi mit Sprachsteuerung!

10 Ausblick

Das Softwarepaket, das Sie in Schritt 01 heruntergeladen haben, enthält noch weitere Sprachprogramme. Informationen dazu finden Sie unter **stevenhickson.blogspot.co.uk/2013/05/voicecommand-v20-for-raspberry-pi.html**.

Der Raspberry Pi macht Schule

Der Raspberry Pi ist mehr als ein paar Bauteile auf einer Steckplatine. Dieser Artikel beschäftigt sich mit der Geschichte um den größten Hype in der Open-Source-Welt seit Android.

„Informatik wird meistens unter Niveau von fachfremden Lehrkräften unterrichtet, die sich kaum mit Computern auskennen."

Eben Upton sitzt in der Hitze von Korfu und genießt seinen wohlverdienten Urlaub. So relaxed wie er wäre wahrscheinlich jeder nach der erfolgreichen Markteinführung eines Produkts, das zusammen mit dem 3G-iPad von Apple herauskam und mindestens genauso viel, wenn nicht mehr Hype auslöste.

Mit seinem niedrigen Preis und den vielen Möglichkeiten ist der Raspberry Pi ein Computer-Winzling zum Liebhaben. Er ist vorrangig auf den britischen Markt ausgerichtet, Upton wollte damit bei britischen Schulkindern das Interesse am Programmieren und an der Funktionsweise von Computern beleben. Dass der Pi auch bei Hobbyprogrammierern und enthusiastischen Linux-Nutzern gut ankam, war ein willkommener Nebeneffekt.

Als das Interesse schlagartig sehr stark wurde, kam das Raspberry-Pi-Team sehr ins Schwitzen. Upton gesteht, dass sie nie mit einem solchen Andrang gerechnet hatten. „Wir redeten immer davon, dass wir insgesamt vielleicht 10.000 Stück verkaufen", erzählt er. „Das waren so unsere Größenvorstellungen. Die hohe Nachfrage hat uns völlig überrascht."

Die Idee eines kreditkartengroßen Computers, auf dem richtige Anwendungen, Spiele und Videos laufen und der zum geringstmöglichen Preis angeboten wird, war schon lange vor der Bankenkrise entstanden. 2006 war Upton Dozent an der Cambridge University. „Ich ging die Bewerbungen für den Studiengang Informatik durch", erinnert er sich. „Es waren ein paar kluge Kids darunter, aber im Vergleich mit vor zehn Jahren wussten sie praktisch nichts. Einige, die schließlich einen Studienplatz bekamen, hatten am Anfang einen sehr geringen Wissensstand, was Computer betraf. Von der Hacking-Mentalität der 1980er Jahre war nichts mehr zu spüren."

Upton wünschte sich ein bisschen von dieser Kultur zurück. Er sah zwei Probleme: Sie hatten zu wenig gute Informatikstudenten an der Uni, und die Firmen wollten keinen Nachwuchs, den sie erst von Grund auf selber ausbilden mussten. „Leute aus verschiedenen Zusammenhängen kamen zu den gleichen Schlussfolgerungen", sagt er. Upton hatte das Gefühl, dass er und seine Kollegen in ihrer Jugend anders als heute zum Programmieren kamen. Er hatte seinen BBC Micro geliebt und war danach auf einen Commodore Amiga umgestiegen. Keiner der Rechner war neu gewesen – den BBC Micro kaufte er gebraucht, der Amiga war ein Vorführmodell. Er programmierte auf beiden, doch für ihn waren es unterschiedliche Ansätze. „Der Amiga war auf gewisse Weise der Anfang vom Ende. Den BBC Micro schaltete man ein, er machte Piep und los ging's. Der Amiga dagegen brauchte Tools, die nicht beim Rechner dabei waren; es war eher wie ein PC. Wenn man nicht schon ein Programmierer war, dann war es mit dem Amiga unwahrscheinlicher, dass man mit dem Programmieren anfing."

Computer zur Ausbildung

In den 1980ern war der BBC Micro an britischen Schulen weit verbreitet. Für viele Schüler war es der erste Rechner, und wenn sie nur so was wie Maniac Mansion darauf spielten. Eine der großen Ideen hinter dem Micro war die Bereitstellung eines Computers zur Ausbildung. Er sollte Schüler ins Programmieren einführen, sie sollten vertraut werden damit, wie diese neuartigen Geräte funktionierten.

Der Micro wurde von Acorn Computers für das Computer Literacy Project der BBC entwickelt. Er war teuer in der Anschaffung, weshalb sich der Sinclair ZX Spectrum, Commodore 64 und Amstrad CPC als Heimcomputer durchsetzten. Aber er war geeignet für Leute wie David Braben, der während seines Studiums an der Cambridge University zusammen mit einem Freund, Ian Bell, darauf ein Game schrieb. Elite war

Raspberry Pi in Zahlen

700 MHz

Der Pi hat eine 700 MHz-ARM11-CPU

ARMv6 v ARMv7

Ein paar Kritiker bemängelten, dass ARMv6 und nicht ARMv7 benutzt wurde. „Das hatte rein praktische Gründe. Wir schauten uns an, was in unserer Preisklasse in der Broadcom-Roadmap wann zur Verfügung stand, und das war ARMv6“, sagt Eben Upton.

eines der ersten echten 3D-Spiele und extrem erfolgreich. Die beiden verdienten ein Vermögen damit und machten Karriere in der Videospiele-Industrie.

Im Jahr 2004 unterhielt sich Braben mit einem Freund, Jack Lang, der Assistent im Computerlabor der Cambridge University war und bei der Entwicklung des Micro-Betriebssystems beteiligt gewesen war. Sie sprachen über die Probleme in der Ausbildung. „Wir redeten darüber, dass die Zahl der Informatikstudenten dramatisch gesunken war, und spekulierten über mögliche Ursachen. Uns fiel auf, dass Informatik nicht mehr als zeitgenössisch und spannend, sondern im Gegenteil als Streber- und Nerdfach gesehen wird. Noch schlimmer ist, dass Informatik zwar seit Ende der 1990er Jahre Schulfach ist, aber an den Schulen jedes Interesse an IKT einschläft, weil es so ein tödlich langweiliges Fach ist.“

Die Lehrer, so Braben, unterrichteten nicht Programmieren, sondern zeigten den Kindern, wie man mit Word einen Brief schreibt, in Excel eine einfache Tabellenkalkulation anlegt oder mit PowerPoint eine Präsentation erstellt. Braben, der heute die Spieleentwicklungsfirma Frontier Developments leitet, war geschockt, als Kinder in Tests Informatik als ihr langweiligstes Fach bezeichneten und nicht mehr Geschichte oder Mathe wie noch vor zehn Jahren.

Regierungsstatistiken belegten diese Theorien. Seit 2001 war die Zahl der Studienplatzbewerber für Informatik im Vergleich zu den Vorjahren um 54 % gesunken. „Das ist ein dramatischer Einbruch“, sagte Braben, „weil sich insgesamt die Studentenzahlen ja erhöhen.“ Er und Lang machten sich ernsthafte Gedanken.

„Wir dachten erst, am besten ersteigert man den Kids einen BBC Micro von Ebay“, sagte er lachend. „Aber so viele Micros werden auf Ebay gar nicht angeboten. Und was problematischer ist, die Kids würden den Micro nicht cool finden, weil er ziemlich retro daherkommt. Vielen von uns gefällt das, aber verglichen mit modernen Geräten wie dem iPad ist er zu groß und schwerfällig.“ Sie mussten eine andere Lösung finden.

„Für Schuldkinder war Informatik das langweiligste Fach.“

Pete Lomas und ein Raspberry Pi der dritten Generation.

Einer der frühesten Prototypen des Raspberry Pi.

Raspberry Pi in Zahlen
4 USB ports beim Modell B+

Hardware ist wichtig

Braben und Lang dachten zuerst an ein Software-Plattform, mit einer Programmiersprache wie Python. Doch Eben Upton arbeitete schon an Prototypen für ein Hardware-Gerät. Der erste war eine simple Atmel ATmega644-, mikrocontrollerbasierte Platine, die die Nutzer selbst mit einem Lötkolben zusammenbauten. Die Leistung entsprach einem Rechner aus den 1980ern, allerdings lief sie mit 22.1 MHz und hatte 512 K SRAM für Daten- und Framebuffer-Speicherung.

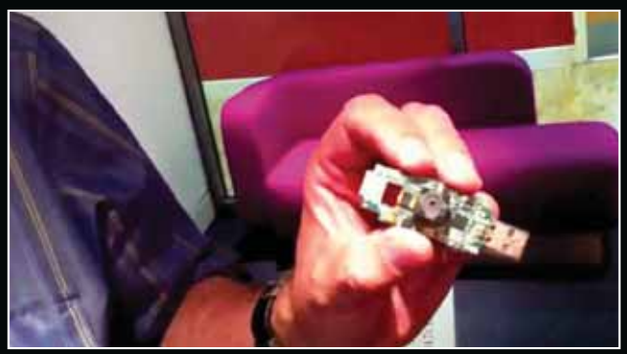
Zusammen mit Rob Mullins, Lang und Alan Mycroft perfektionierte die Gruppe das Gerät, und 2008 war man schon an der zweiten Generation des Pi. Upton arbeitete inzwischen für den Chiphersteller Broadcom, während die Cambridge-Gruppe sich mit Pete Lomas zusammentat, dem Entwicklungsleiter der Hardware-Design-Firma Norcott Technologies.

Lang machte sie auch mit Braben bekannt. „Wir waren zum Lunch verabredet“, erinnert sich Upton an das erste Treffen mit dem Mann hinter Elite. „Wir bestellten Burger in einem ziemlich miesen Laden. Doch was das Problem der mangelnden Computerkenntnisse an den Schulen betraf, hatte David ganz ähnliche Ansichten wie ich. Das war sehr aufregend.“ Die Dinge kamen ins Rollen.

2008 gründete die Gruppe eine Organisation mit dem Ziel der Förderung von Informatik an Schulen und der Entwicklung eines neuartigen Rechners, um Informatik auf Schullevel zu lehren. Daraus wurde die Raspberry Pi Foundation.

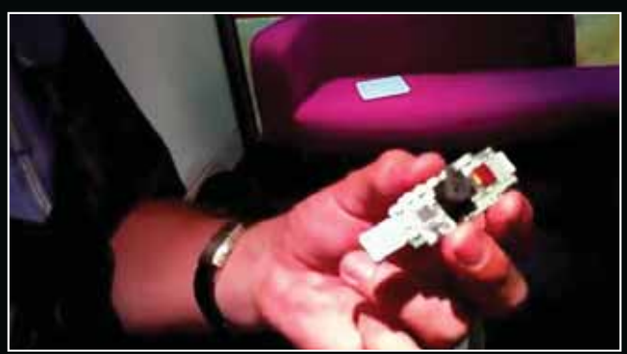
Wie die BBC den Raspberry Pi bekannt machte

Im Mai 2011 besuchten Braben und Upton den Tech-Reporter der BBC, Rory Cellan-Jones. Sie wollten die BBC, die schon dem Micro ihren Namen geliehen hatte, auch vom Pi überzeugen. Cellan-Jones produzierte ein kurzes Video, das sich auf YouTube rasant verbreitete. Ab da gab es kein Halten mehr. Oder, wie Upton es ausdrückt: „Das war der Moment, als es ernst wurde."



„Dieses kleine Gerät ist ein Prototyp des Raspberry-Pi-Computers", sagt Cellan-Jones.

„Im Idealfall kosten sie 15 bis 20 €, sodass sie an Kinder verschenkt werden könnten."



„Es ist ein Computer an einem USB-Stick. Am einen Ende hat er HDMI, am anderen USB."

„Es wird eine Weile dauern, bis jedes Kind einen zu Hause hat. Aber wir hoffen, in zwölf Monaten ist es so weit."

Inzwischen war allen Beteiligten klar, dass sie eine Lösung gefunden hatten für die Probleme, die sie beschäftigten. Hardware, so ihre Vermutung, war wichtig. „Der Pi begeistert die Kids – das sehe ich immer wieder im Klassenzimmer. Sie finden es toll, wenn sie echte Hardware in die Hände bekommen. Eine Platine zum Anfassen, Teile, auf die sie zeigen können und sagen, das da ist der Prozessor, das der Speicher", erzählt Upton. „Vieles könnte man auch mit Software vermitteln, mit einer bootfähigen Lehr-CD. Aber es ist nicht das Gleiche. Keine Software hat je so eine positive Reaktion bei den Schülern ausgelöst wie der Raspberry Pi."

Und die Chips dazu?

Durch Uptons Anstellung bei Broadcom tat sich die Möglichkeit auf, an billige Chips zu kommen. „Wir suchten sehr lange nach einem Partner, um die Herstellung des Pi zu finanzieren", sagt Braben. „Es war mühsam, weil die Leute skeptisch sind. Wenn man Firmen für eine Finanzierung gewinnen will, glauben einem die wenigsten, dass man wirklich etwas Brauchbares herstellen wird. Deshalb war die Verbindung zu Eben und Broadcom für uns sehr wichtig."

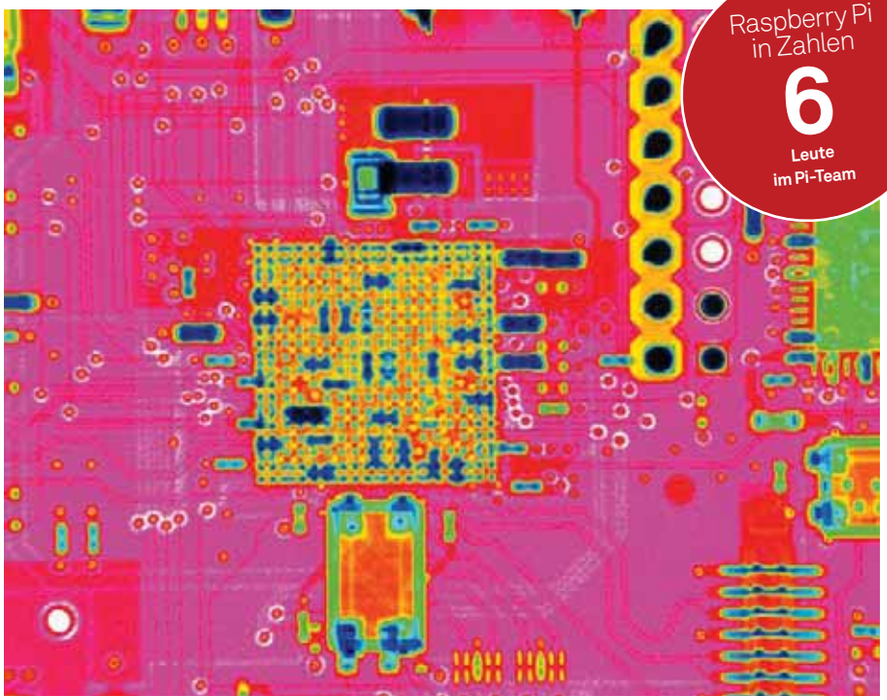
Upton wusste, was Broadcom potentiell bieten konnte. „Was ein Bastler für zehn Euro bei einem Elektronik-Shop erhält, ist nicht zu vergleichen mit dem, was man als Handyfirma für die gleiche Summe bekommt", erklärt er. „Eine Handyfirma bekommt für zehn Euro einen recht beeindruckenden Chip, und Broadcom stellt eine ganze Menge Handychips in dieser Preisgruppe her.

Der Prototyp erfüllte die Leistungsanforderungen, wir hatten am Ende wirklich gute Multimedia-Elemente. Aber es war doch ein recht esoterischer Rechner. Er hatte kein eigenständiges Betriebssystem – man musste es selbst schreiben. Wir programmierten zum Beispiel unsere eigenen SD-Kartentreiber. Einen Standard-Desktop konnte man auch nicht verwenden. Eine Menge Kriterien waren erfüllt, aber er war noch nicht soweit, dass wir ihn auf den Markt bringen wollten."

Das Team machte weiter, und ein Pi der dritten Generation wurde entwickelt. Einige Jahre zuvor, auf einem Forschungstag über neue Computing-Architektur, hatte Mycroft sich

Raspberry Pi in Zahlen

2 USB-Ports beim älteren Modell B



Raspberry Pi in Zahlen

6

Leute im Pi-Team

Die blauen Kleckse auf dem Chip sind die Stützkondensatoren und Lötpunkte, und das große Rechteck darunter ist der Kristall.

mit Lomas über das Fehlen einer vernünftigen Rechner-Plattform für Software-Entwicklung unterhalten. Er erzählte ihm dabei von der Arbeit der Cambridge-Gruppe. Lomas hat ein besonderes Interesse an Hardware, die den Nutzern offenen Zugang für Elektronik-Experimente gewährt, etwas, das seiner Meinung nach durch den PC, die Xbox und eine ganze Generation von spannenden, aber geschlossenen Produkten verdrängt wurde. Er wurde sehr wichtig für die Gruppe. „Pete war das Hardware-Genie hinter dem Pi", erzählt Upton. „Er konnte einem bis ins kleinste Detail erklären, wo die Herausforderungen auf diesem Gebiet lagen."

Eine Stange Geld

Für Lomas hatte absolute Priorität, dass die Funktionsanforderungen zu den geringstmöglichen Kosten erfüllt wurden. Sie mussten das Projekt so kostengünstig wie möglich anbieten, ansonsten würden die Schulen und die Kinder es nicht annehmen. „Wir werden oft gefragt, wie wir es geschafft haben, dass die Computer so billig sind. Die Frage geben wir zurück – warum sind andere Computer so teuer?", sagt Braben. „Wir waren erfolgreich, weil wir die von Broadcom für Handys entwickelte System-on-a-Chip-Technologie benutzten. Das Herzstück ist eine ARM11-CPU. Speicher und CPU haben wir übereinander gebaut, mit einer Technik, die man Package-on-Package (oder PoP) nennt. Das hat die Kosten der Platine und des gesamten Geräts enorm reduziert. Es gab gute Lösungsansätze, Chancen, die wir ergriffen haben, und Broadcom hat uns dabei sehr geholfen."

Der Kostenfaktor schloss viele Techniken aus, die man hätte verwenden können, etwa ultrakleine Komponenten oder eine High-Tech-PCB-Konstruktion. Im frühen Design funktionierte das mit einfachem Silizium, aber man musste zu viel Stützkomponenten hinzufügen, was nicht akzeptabel war. Die Fertigstellung war in greifbarer Nähe, was die Spezifikation betraf, nicht aber die Finanzierung.

„Warum sind andere Computer so teuer?"

Doch es gab auch immer wieder Durchbrüche. „Als Hardware-Designer war mein Anliegen, dass alle Teile auf die Leiterplatte passten", erzählt Lomas. „Weil sich so viel Funktionalität und Input/Output auf so wenig Platz konzentrierte – wir reden hier vom BCM2835 –, waren wir trotz des Anfangserfolgs bald verzweifelt. Fast sah es so aus, als sei der Versuch gescheitert, den BCM2835 auf einer einfachen sechslagigen Platine zu verdrahten. In langen Nächten und mit unterschiedlichen Methoden lösten wir das Problem und entwickelten mit einem kleinen Dreh eine Leiterplatte, die uns die Lösung brachte. Ab da war das Design relativ einfach."

Als der Designprozess begann, gab es eine lange Wunschliste von Geldgebern und Kollaborateuren. „Viele Wünsche fielen auf dem Weg zum endgültigen Design unter den Tisch. Das war nicht immer einfach, und es kam zu hitzigen Diskussionen", sagt Lomas. „Wir versuchten, alles einzubauen, was irgend möglich war. Ich bedaure zum Beispiel sehr, dass wir keine anständigen Audio-Ein- und -Ausgänge aufnehmen konnten. Zusammen mit einer Kamera hätte das eine weitere Ebene an Möglichkeiten eröffnet."

6 an einem Strang

„Es gab Zeiten, da hatte ich die Motivation verloren", sagt Eben Upton über die sechs Jahre, die er in das Raspberry-Pi-Projekt steckte.

Aber weil das Pi-Team aus sechs Leuten bestand, ging die Sache immer weiter. „Wirklich hilfreich war, dass mehrere Leute involviert waren. Immer wieder mal wollte einer von uns aufgeben, aber weil wir zu sechst waren, mit unterschiedlichen Hintergründen, aber alle auf gleicher Augenhöhe, haben wir durchgehalten."

Jeder der sechs hatte noch einen anderen Job, keiner arbeitete ständig an dem Projekt. „Die Motive, warum wir die Probleme lösen wollten, lagen bei jedem etwas anders. Einer oder zwei waren immer voller Elan und Begeisterung, und das hat uns bei der Stange gehalten. Es konnten sechs Monate vergehen, in denen ich nichts beitrug, aber einer der anderen rackerte so lange an dem Projekt."

Nutzer konnten ebenfalls Wünsche äußern. „Gleich am Anfang fragten Nutzer nach Befestigungslöchern, damit der Pi nicht durch das Gewicht des HDMI-Kabels auf dem Tisch herumrutschte“, erzählt Lomas. „Aber Löcher hätten wertvollen Platz auf der Platine verbraucht. Glücklicherweise gibt es Gehäusedesigns zum Anklammern, mit denen wir das Problem lösen können. Viele weitere Wünsche – PoE, mehr Speicher, mehr USB-Anschlüsse, mehr Leistung – hätten zu hohe Mehrkosten bedeutet.“

Wir brauchen eine neue Ausbildung

Der Raspberry Pi ist die Reaktion auf den langweiligen Computerunterricht, der Kinder jeden Spaß an IKT und Informatik nimmt. Doch die Raspberry-Pi-Stiftung weiß auch, dass damit das eigentliche Problem nicht gelöst ist. Nun müssen die Schulen vom Pi überzeugt werden. Und Lehrer brauchen neue Kompetenzen, um die richtige Botschaft zu vermitteln. David Braben sieht da Schwierigkeiten.

„Informatik wird unter Niveau von fachfremden Lehrkräften unterrichtet, die sich kaum mit Computern auskennen“, sagt er. „Sie könnten auch Geschichts- oder Erdkundelehrer sein, oder Französisch unterrichten. Wenn ein Kind ein Problem hat und eine Frage stellt, und der Lehrer kann sie nicht beantworten, dann werden die Schüler im Klassenzimmer unruhig. Manchmal wissen sie mehr als die Lehrer und können zum Beispiel einen PC lahmlegen. Wenn man weiß, was man tut, ist das sehr einfach. Man löscht nur ein paar Batch-Dateien, und schon fährt das Ding nicht mehr hoch.

Wir führen Gespräche mit Regierungsvertretern über Lehrerfortbildung. Da geht es um mehr als ums Geschäft, da geht es nicht nur um den Raspberry Pi. Wir bekommen viel Unterstützung von der British Computing Foundation und der Online-Gruppe für Informatik an der Schule. Die Ankündigung von Bildungsminister Michael Gove, dass ab September 2014 Programmieren an britischen Schulen Pflichtfach wird, ist fantastisch, und wir haben bei der Gestaltung des neuen Lehrplans mitgewirkt.“

Das Ziel war, alles auf dem winzigen Pi zu verschalten.

Raspberry Pi in Zahlen

6

Jahre Entwicklung

Bei der Auswahl der Komponenten des Pi war die weitestgehende Integration der benötigten Funktionen ausschlaggebend. „Dabei bewies der Broadcom BCM2835, was in ihm steckt“, sagte Lomas. „Er ermöglicht eine dreiadrige Chip-Lösung mit Prozessor, Speicher und einem Ethernet USB Combo Chip. Dafür sind nur wenige externe Stützkomponenten nötig, was es leichter machte, die Hardware auf einer kreditkartengroßen Platine unterzubringen.“

Ähnlich wichtig war, dass die Nutzer nie das Gefühl haben sollten, der Raspberry Pi wäre für sie nicht zugänglich. Der Pi basiert auf Linux, und die offizielle Lehrsprache ist Python, wobei die Nutzer nicht auf Python beschränkt sind. Sie werden ermutigt, mit dem Pi zu experimentieren, um zu verstehen, wie er funktioniert. Die populäre Programmiersprache Scratch ist vorinstalliert.

„Python 2.7 ist in jeder Distribution verfügbar“, erklärt Upton. „Das ist sozusagen das High End. Für den Einstieg gibt es Dinge wie Scratch. Mit diesen Tools kann man es bis zum professionellen Software-Entwickler bringen, und alle sind sie standardmäßig auf dem Pi installiert.“

Linux wurde verwendet, um die Kosten niedrig zu halten. Debian wird als Standard-Distribution empfohlen, aber es gab ein paar Probleme mit Ubuntu. „Wir gingen automatisch davon aus, dass Ubuntu auf dem Pi laufen wird“, sagt Upton. „Doch dann fanden wir heraus, dass der Support für ARMv6 eingestellt worden war. Sollten wir in der Zukunft ARMv7 benutzen – zurzeit gibt es keine Pläne in diese Richtung –, dann wird Ubuntu eine Möglichkeit.“

Doch trotz der freien Open-Source-Einstellung von Linux hätte Braben gerne, dass der Pi weniger wie ein Linux-Gerät daherkommt. „Für die Zukunft ist die Herstellung einer Unterrichtsversion des Pi geplant, mit einem Gehäuse und sämtlicher Software vorinstalliert – und die außerdem nicht schon auf den ersten Blick wie ein Linux-PC aussieht. Im Innern bleibt der Pi natürlich ein Linux-Gerät, aber mehr so, wie ein Android-Handy ein Linux-PC ist – mit einer Art Software-Verkleidung. Der Grundgedanke ist, dass der Nutzer wie bei einer Zwiebel mit wachsendem Wissen Schale um Schale löst und am Ende versteht, dass der Pi ein Linux-PC ist. Wir legen nur freundliche Schichten darüber, die

„Braben plant einen Pi, der weniger nach Linux aussieht.“

Umkämpfter Markt

Der Raspberry Pi ist nicht der einzige Billigcomputer. VIA Technologies stellte einen gehäuselosen Desktop-Rechner vor, den APC 8750. Er läuft mit Android, kostet etwa 50 €, Chip, Speicher und I/O-Ports sind auf der Platine untergebracht.

Macht sich Eben Upton Sorgen? „Wenn jemand ein Gerät entwickelt, das zum gleichen Preis so gut wie oder besser als der Pi ist, haben wir ernsthafte Konkurrenz“, sagte er. „Und zwar, was unsere Verkäufe betrifft. Wir haben nichts dagegen, wenn sie auch auf offenen Plattformen laufen – dann wird ja unser Ziel erfüllt, dass offene Hardware allen zur Verfügung steht. Der VIA-Rechner ist fast so preiswert wie der Pi, und die CPU kommt unserer ziemlich nahe. Multimediamäßig sieht es dürftiger aus, die Auflösung ist 720p, nicht 1080p.“

Raspberry Pi in Zahlen

2835

Das SoC ist ein Broadcom BMC2835

es Computeranfängern leichter machen. Wenn man ins Handy Optionen eintippen müsste, nur um anzurufen, dann würde das den Leuten auch nicht besonders gefallen.“

Scharf auf den Raspberry Pi

Ein Jahr, bevor der Pi auf den Markt kam, gab es eine Art Vorankündigung. Braben und Upton besuchten den Technologie-Reporter der BBC, Rory Cellan-Jones. Sie hatten einen Pi dabei, der kaum größer als ein USB-Stick war. Doch die beiden wollten nicht für den Pi werben. Sie wollten die Rundfunkanstalt BBC davon überzeugen, dass sie den Raspberry Pi mit ihrem Namen unterstützte, wie sie es schon beim BBC-Micro getan hatte.

„Wir sprachen über ein mögliches Engagement der BBC, und am Ende sagte Rory: Habt ihr was dagegen, wenn ich ein kleines Video mache? Und er filmte ein kurzes Video von David, der den winzigen Prototypen einfach in die Kamera hielt“, erzählt Upton. „Zu diesem Zeitpunkt arbeiteten wir ohne großes öffentliches Interesse mehr für uns selber am Pi. Und dann bekam dieses Video innerhalb von zwei Tagen 600.000 Hits auf YouTube. In diesem Moment wurde uns klar, dass wir unabsichtlich versprochen hatten, den Pi fertigzustellen. Uns war nicht bewusst gewesen, dass wir aller Welt verkündeten: Wir werden dieses Ding produzieren. Doch dann erwarteten es plötzlich 600.000 Leute von uns. Das war der Moment, als es ernst wurde und wir uns wirklich ins Zeug legten.“

Sie arbeiteten in der Folge an zwei Modellen, von denen das zweite später auf den Markt kam. Modell A hatte 128 MB Arbeitsspeicher, später wurde es für 256 MB umdesignt. Es besaß einen USB-Port und kein Ethernet. Modell B hatte 256 MB Arbeitsspeicher, zwei USB-Ports und einen Ethernet-Port. „Es war immer aufregend, wenn wir neue funktionierende Hardware hatten, egal welche Generation“, erzählt Upton. „Als die Beta-Boards, die Alpha-Boards, und dann die ersten fertigen Boards herauskamen und funktionierten, liefen in China die ersten 10 Pi-Computer von den Produktionsbändern. Ich erinnere mich, als wir die erste große Lieferung erhielten – die ersten 2.000 –, und dieser Typ kommt mit seinem DHL-Laster angefahren, nimmt den Mini-Stapler und holt sie hinten raus. 2.000 Computer auf einer Palette – das war schon ziemlich cool.“

Beim Besuch einer Computermesse in New York war Upton überrascht von den begeisterten Jugendlichen. „Da waren Kids aus Philadelphia hergekommen, weil sie sich eine sehr frühe Version des Raspberry Pi live anschauen wollten“, erinnert er sich. Aber die Entwicklung hatte auch ihre Schattenseite. „Manchmal war es sehr schwierig, Zeit zu finden.“

Abverkauf

Am 29. Februar 2012 kam der Raspberry Pi auf den Markt. Der Zeitpunkt war perfekt, denn der britische Bildungsminister Michael Gove hatte gerade angekündigt, dass der Computerunterricht in Schulen überdacht werden müsse. Gove betonte, dass er einen größeren Schwerpunkt beim Programmieren sehe.

Die Nachfrage nach dem Pi war so groß, dass die Website eines Händlers, Premier Farnell in Leeds, wegen der vielen Bestellungen zusammenbrach. Es kam vereinzelt zu Lieferungsverzögerungen, weil ein falscher Netzwerkstecker auf einige Platinen gelötet worden war.

Die 10.000 Geräte der ersten Produktionscharge waren innerhalb von wenigen Stunden ausverkauft. Es dauerte allerdings noch mehrere Wochen, bis die Geräte tatsächlich ausgeliefert wurden. „Hätte Rory dieses Video nicht gefilmt, dann wären wir später herausgekommen und mit weniger Furore. Davids Name und sein Ruf aus den BBC-Micro-Zeiten verschaffte uns die erste Welle öffentlichen Interesses. Das Video führte dazu, dass der Pi ein Jahr lang meistens deshalb in den Schlagzeilen war, weil wir immer noch nicht fertig waren. Jedes Mal, wenn wir einen Termin ankündigten und es dann doch nicht schafften, bekamen wir viele Kommentare. Andererseits hat uns das den nötigen Druck verschafft, damit wir das Ding fertigmachten.“

Über die Zukunft will Upton nicht viel sagen. „Ich klinge sonst bald wie ein Geschäftsunternehmen“, meint er. „In einem Jahr machen wir vielleicht genügend Umsatz und überlegen uns dann vielleicht etwas Neues. Aber im Moment sind wir glücklich mit dem Pi, so wie er ist, und es geht darum, die Rechner an die Nutzer zu bringen.“

Auf dem Pi in Python programmieren

Erlernen Sie die Grundlagen der Python-Programmierung auf dem Raspberry Pi.

Es empfiehlt sich, am Anfang der Datei den Zweck des Programms zu beschreiben. Bei größeren Projekten mit mehreren Dateien wird das hilfreich sein.

Datentypen müssen bedacht werden. Wir konvertieren diese Zahl in eine dezimale, damit wir bei der Arithmetik keine Nachkommastellen verlieren.

Irgendwo im Code muss die Haltebedingung für eine while-Schleife erfüllt sein, da die Schleife sonst niemals endet!

Die print-Funktion akzeptiert nur den Datentyp String, also müssen wir alle Variablen mit einem Zahlentyp nach String konvertieren, bevor wir sie am Bildschirm ausgeben können.

```
#!/usr/bin/env python

# An advanced Hello World program that will demonstrate the basics of
# programming in Python via a series of examples. Created by Liam Fraser
# for Linux User and Developer - 22/04/2022

# Import the sys library for the sys.exit function
import sys
# Import everything from the decimal library
from decimal import *

# Get the users first name and output a welcome message
firstName = raw_input("Please enter your first name: ")
print("Welcome " + firstName + "\n")

# Ask the user for a number and square it, double it and halve it
number = raw_input("Please enter a number: ")
number = Decimal(number)
numberHalved = number / 2
numberDoubled = number * 2
numberSquared = number * number

# Print out the values we just worked out, converting each float value
# to a string
print("The result of halving that number: " + str(numberHalved))
print("The result of doubling that number: " + str(numberDoubled))
print("The result of squaring that number: " + str(numberSquared))
print("")

# The stopping condition for a while loop
yesOrNo = False

# A while loop that will run until a user enters either 'yes' or 'no'
while yesOrNo == False:
    result = raw_input("Do you want to continue? (yes/no) ")

    if result == "yes" or result == "no":
        yesOrNo = True
    else:
        print("Error, please type yes or no" + "\n")

# Deal with the result
if result == "yes":
    print("\nContinuing")
else:
    print("\nExiting")
    sys.exit()

# Create the count which will be a stopping condition for a while loop
count = 1

# Use a while loop to add 5 to the number and output the value each time
print ("Incrementing the number by 5\n")

while count <= 5:
    number += 1
    print("number = " + str(count) + " - " + str(number))

    # Increment the count
    count += 1

# Finish off by printing that we are exiting
print("\nExiting")
```

Bevor wir mit der Einführung in die Python-Programmierung anfangen, bereiten Sie eine SD-Karte mit dem neuesten NOOBS-Abbild von der Raspberry-Pi-Foundation-Website vor. Das Verwenden eines SD-Karten-Abbilds bedeutet für uns, dass das Betriebssystem und eine Entwicklungsumgebung bereits vorkonfiguriert sind. Wir verwenden eine schlanke Entwicklungsumgebung namens Geany für unsere Python-Entwicklung. Geany bietet ein freundlicheres Interface als Texteditoren wie Nano, und dadurch kommt man einfacher hinein.

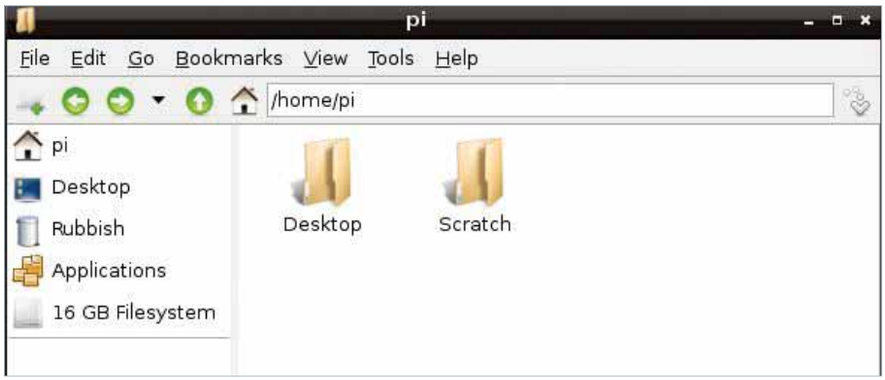
In dieser Übung behandeln wir unter anderem:

- **Grundarithmetik**
- **Vergleichsoperatoren wie „ist gleich“ und „ist nicht gleich“**
- **Kontrollstrukturen wie Schleifen und if-Anweisungen**

Am Ende erhalten wir eine fortgeschrittene Version der „Hello, World“-Anwendung. Legen wir direkt los...

Was Sie brauchen:

- einen Raspberry Pi samt Peripherie
- eine SD-Karte mit einem Abbild des neuesten Debian für den Raspberry Pi
  raspberrypi.org/downloads



01 Organisiert bleiben

Wir wollen die Ordner auf unserem Pi nicht durcheinanderbringen, also blicken wir in den Dateimanager und organisieren uns. Öffnen Sie den Dateimanager mit einem Klick auf das Icon neben dem Menü-Symbol unten links auf dem Bildschirm. Erstellen Sie einen neuen Ordner (Rechtsklick und dann Neu > Ordner), geben Sie einen Namen ein, und klicken Sie auf OK. Wir haben einen Ordner namens Python und darin einen Unterordner namens Hello World erstellt.

02 Geany starten

Starten Sie Geany (im LXDE-Menü unter Programme). Sobald das Interface von Geany erscheint, erstellen Sie eine neue Python-Datei von einer Vorlage (Neu (mit Vorlage) > main.py). Löschen Sie alles aus der Vorlage bis auf die erste Zeile (#!/usr/bin/env python). Diese Zeile ist wichtig, damit der Code von der Kommandozeile aus laufen kann und die Bash-Shell den Python-Interpreter verwendet.

03 Die Arbeit abspeichern

Beim Programmieren ist es immer empfehlenswert, regelmäßig mit Strg+S abzuspeichern. Schließlich wäre es schade, seine Arbeit zu verlieren. Um die Datei zum ersten Mal zu speichern, drücken Sie Strg+S oder wählen Sie im Datei-Menü den Befehl Speichern. Geben Sie der Datei einen aussagekräftigen Namen und legen Sie sie in der ordentlichen Ordnerstruktur ab, die wir erstellt haben. Als Programmierer sollte man sich angewöhnen, die Dateien organisiert zu halten, um nicht durcheinanderzukommen, wenn das Projekt größer und komplexer wird.

```
1  #!/usr/bin/env python
2
3  # An advanced Hello Wo
4  # programming in Pytho
5  # for Linux User and D
6
7  # Import the sys libar
8  import sys
9  # Import everything fr
10 from decimal import *
```

04 Vorbereitungen

Detaillierte Kommentare im Code sind wichtig, um seine Gedanken zu notieren und komplexe Vorgänge zu dokumentieren. Wenn ein anderer Programmierer irgendwann mit Ihrem Code arbeiten muss, wird er es Ihnen danken. Fügen Sie also einen Kommentar ein, der beschreibt, was das Programm macht und von wem es stammt. Alle Kommentarzeilen fangen mit einem Nummernsymbol (#) an und werden vom Python-Interpreter nicht als Code interpretiert. Wir importieren die sys-Bibliothek, damit wir die sys.exit-Funktion nutzen können, um das Programm später zu beenden. Wir importieren außerdem alles aus der decimal-Bibliothek, da wir den Datentyp decimal verwenden wollen.

```
12 # Get the users first name and output a welcome message
13 firstName = raw_input("Please enter your first name: ")
```

05 Variablen

Eine Variable ist ein im Speicher abgelegtes Stück Daten, das über einen Namen abgerufen werden kann. Unser Programm soll den Benutzer am Anfang nach seinem Vornamen fragen, diesen in eine Variable speichern und dann eine Willkommensnachricht ausgeben. Wir schreiben also einen Kommentar, der dies erläutert, und erstellen eine Variable namens firstName (Vorname). Wir haben der Lesbarkeit halber den ersten Buchstaben des zweiten Wortes großgeschrieben. Wir möchten in die Variable firstName das Ergebnis einer Funktion namens raw_input speichern, die den Benutzer zur Eingabe auffordert. Die Frage wird in Klammern an die print-Funktion übergeben, und da sie ein String ist, wird sie in Anführungszeichen gesetzt. Ein String ist eine Abfolge von Schriftzeichen. Beachten Sie das zusätzliche Leerzeichen hinter dem Doppelpunkt. Der Benutzer gibt seine Antwort direkt hinter der Aufforderung ein.

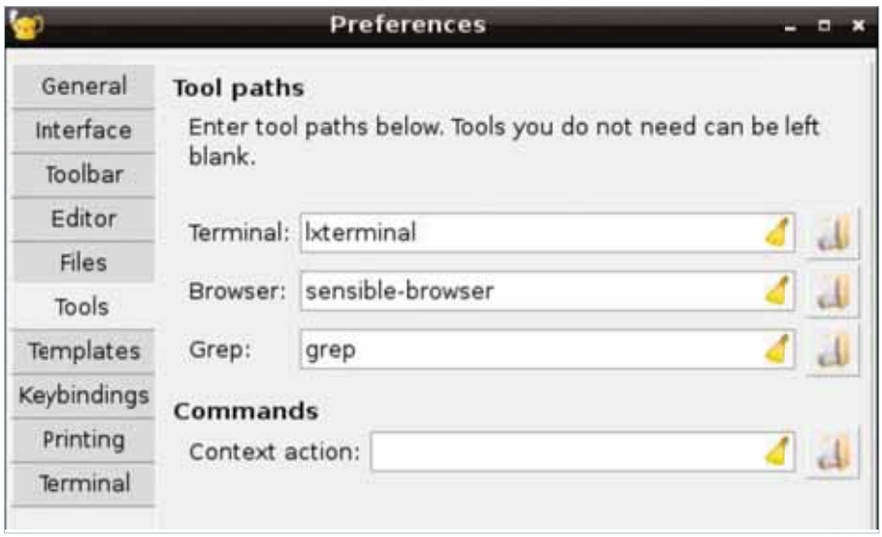
```
14 print("Welcome " + firstName + "\n")
```

06 Eine Nachricht ausgeben

Jetzt haben wir in firstName einen Wert gespeichert. Als Nächstes soll eine Willkommensnachricht am Bildschirm ausgegeben werden. Wir verwenden zur Ausgabe in Python die print-Funktion. Hinter print schreiben wir in Klammern den Wert, der ausgegeben werden soll. Die Verwendung des Additionsoperators bewirkt bei Strings, dass sie aneinandergereiht werden. Beachten Sie, dass firstName nicht in Anführungszeichen steht, weil es ein Variablenname ist. Würde es in Anführungszeichen stehen, würde der Text „firstName“ ausgegeben. Wir schließen mit einem „\n“-Zeichen (neue Zeile) ab, um eine Leerzeile zu hinterlassen, bevor wir mit dem nächsten Beispiel fortfahren.

“Die Verwendung des Additionsoperators bewirkt bei Strings, dass sie aneinandergereiht werden.”

„Das Programm muss vor dem Starten gespeichert werden, da sonst die Änderungen seit dem letzten Speichern von Python nicht interpretiert werden."



07 Ein kleines Problem beheben

Es gibt in dem Debian-Abbild, das wir gerade verwenden, einen Konfigurationsfehler, der Geany betrifft. Drücken Sie die F5-Taste oder wählen Sie im Kompilieren-Menü den Befehl Ausführen. Falls das Programm nicht startet und stattdessen die Fehlermeldung „Terminal kann nicht gefunden werden: xterm" erscheint, liegt das Problem auch bei Ihnen vor. Keine Sorge, das haben wir schnell behoben. Wählen Sie im Menü Bearbeiten den Befehl Einstellungen. Im Kartenreiter Extras suchen Sie den Wert für Terminal und ändern ihn von xterm auf lxterminal.

```
Please enter your first name: Liam
Welcome Liam


------------------
(program exited with code: 0)
Press return to continue
```

08 Programm testen

Jetzt schauen wir doch mal, ob unser Programm einwandfrei läuft. Bedenken Sie, dass die Datei vor dem Starten gespeichert werden muss, da sonst die Änderungen seit dem letzten Speichern von Python nicht interpretiert werden. Starten Sie das Programm mit der Taste F5. Geben Sie Ihren Namen ein und drücken Sie die Eingabetaste. Danach erscheint die Willkommensnachricht. Wenn das Programm mit dem Code 0 abschließt, war alles erfolgreich. Drücken Sie die Eingabetaste, um das Terminal zu schließen.

```
16  # Ask the user for a number and square it, double
17  number = raw_input("Please enter a number: ")
```

09 Mit Zahlen arbeiten

Wir fordern den Benutzer auf, eine Zahl einzugeben, indem wir unsere ersten zwei Zeilen im Prinzip wiederholen. Wir möchten die vom Benutzer eingegebene Zahl halbieren, quadrieren und verdoppeln. Die raw_input-Funktion gibt den vom Benutzer eingegebenen Wert als String zurück. Ein String ist ein Textwert, also kann damit keine Arithmetik vorgenommen werden. Der integer-Typ in Python kann nur ganze Zahlen speichern, während der decimal-Typ auch Nachkommastellen zulässt. Wir nehmen eine sogenannte Typumwandlung vor, die einen Wert von einem Typ in einen anderen Typ konvertiert. Wir konvertieren unseren Zahlenstring in einen dezimalen, weil wir beim Halbieren von Zahlen mit Nachkommastellen rechnen. Hätten wir den Typ integer verwendet, würden die Nachkommastellen abgeschnitten.

```
19  numberHalved = number / 2
20  numberDoubled = number * 2
21  numberSquared = number * number
```

10 Arithmetik durchführen

Die wichtigsten Arithmetikoperatoren in Python sind + (Addition), - (Subtraktion), / (Division) und * (Multiplikation). Wir haben drei neue Variablen namens numberHalved (Zahl halbiert), numberDoubled (Zahl verdoppelt) und numberSquared (Zahl quadriert) erstellt. Wir brauchen nicht anzugeben, dass diese Variablen vom Typ decimal sind, da Python den Typ jeder Variable vom Typ ihres ersten Wertes ableitet. Die Variable number hat den Typ decimal, also sind die Ergebnisse der darauf angewendeten Arithmetik auch vom Typ decimal.

```
23  # Print out the values we just worked o
24  # to a string
25  print("The result of halving that numbe
26  print("The result of doubling that numb
27  print("The result of squaring that numb
28  print("")
```

11 Die Zahlen ausgeben

Jetzt da wir die Arithmetik durchgeführt haben, geben wir die Ergebnisse mithilfe der print-Funktion aus. Die print-Funktion akzeptiert nur Stringwerte, also müssen wir jeden decimal-Wert vor der Ausgabe mithilfe der str()-Funktion in einen String konvertieren. Die print-Anweisung mit leeren Anführungszeichen gibt eine Leerzeile aus. Die print-Funktion startet (für gewöhnlich) nach jeder Ausgabe eine neue Zeile, also gibt ein leerer String einfach eine leere Zeile aus.

„Die print-Funktion akzeptiert nur Stringwerte, also müssen wir jeden Dezimalwert in einen String konvertieren."

```
30  # The stopping condition for a while loop
31  yesOrNo = False
32
33  # A while loop that will run untill a user enters either "yes" or "no"
34  while yesOrNo == False:
35      result = raw_input("Do you want to continue? (yes/no) ")
36
37      if result == "yes" or result == "no":
38          yesOrNo = True
39      else:
40          print("Error, please type yes or no" + "\n")
```

12 while-Schleifen und if-Anweisungen für Gültigkeitsprüfung

Um die while-Schleife und die if-Anweisung zu veranschaulichen, stellen wir dem Benutzer eine Ja/Nein-Frage. Wir fragen, ob der Benutzer fortfahren will, worauf die Antwort entweder „ja“ oder „nein“ (in Kleinbuchstaben) sein muss. Eine while-Schleife ist eine Schleife, die sich so oft wiederholt, bis eine Bedingung erfüllt ist. In diesem Fall erstellen wir eine Variable namens yesOrNo (ja oder nein), und solange yesOrNo den Wert False hat, wiederholt sich die Schleife. Die Variable yesOrNo hat den Typ Boolean, der immer nur den Wert True (wahr) oder False (falsch) haben kann. Wir setzen die Variable am Anfang auf False, damit die while-Schleife überhaupt läuft.

Eine while-Schleife hat das Format „while [Bedingung]:“, und der Code, der den Inhalt der while-Schleife ausmacht (in den dem Doppelpunkt folgenden Zeilen), muss eingerückt werden. Code, der nicht eingerückt ist, ist nicht mehr Teil der Schleife. Das Gleiche gilt für if-Anweisungen. Die Bedingung wird mit dem Vergleichsoperator „==“ (ist gleich) abgefragt. Ein einzelnes „=“ ist ein Zuweisungsoperator, während das doppelte Gleichheitszeichen ein Vergleichsoperator ist. Ein weiterer oft verwendeter Vergleichsoperator ist „!=“ (ist nicht gleich).

Wir erstellen eine Variable namens result (Ergebnis), die die Antwort auf die Frage „Wollen Sie fortfahren?“ enthält. Wir überprüfen dann mit der if-Anweisung die Gültigkeit der Eingabe. Beachten Sie den „or“-Operator, mit dem wir zwei Bedingungen testen. Wenn der Benutzer eine korrekte Eingabe gemacht hat, setzen wir yesOrNo auf True, wodurch die while-Schleife ihr Ende findet. Andernfalls geben wir eine Fehlermeldung aus und die while-Schleife wiederholt sich. Der Benutzer kann jederzeit mit Strg+C im Terminal das Programm abbrechen.

```
42  # Deal with the result
43  if result == "yes":
44      print("\nContinuing")
45  else:
46      print("\nExiting")
```

13 Fortfahren oder beenden?

Als Nächstes kümmern wir uns um das Ergebnis, das während der while-Schleife mit if-Anweisungen gespeichert wurde. Wenn der Benutzer „ja“ eingegeben hat, geben wir „Programm wird fortgesetzt“ aus. Andernfalls geben wir „Programm wird beendet“ aus und rufen die Funktion sys.exit auf. Wir brauchen für das Fortfahren nichts weiter zu tun, da das Programm ja automatisch fortgesetzt wird, wenn sys.exit nicht aufgerufen wird. Der Code zeigt auch, dass „\n“ (neue Zeile) überall im String vorkommen kann, nicht nur innerhalb seiner eigenen Anführungszeichen wie zuvor.

```
# Create the count
count = 1
```

14 Schleifen mit Zahlen

Wir schreiben jetzt eine while-Schleife, die eine Zahl und einen „<=“-Operator (kleiner oder gleich) als Haltebedingung verwendet. Die while-Schleife erhöht die Zahl um 1 und gibt in jedem Durchlauf die Änderung aus, bis die Haltebedingung erfüllt ist. Durch die Variable count (Zähler) wissen wir genau, wie oft die while-Schleife bereits durchlaufen wurde.

```
52  # Use a while loop to add 5 to the
53  print ("Incrementing the number by
54
55  while count <= 5:
56      number += 1
57      print("number + " + str(count)
58
59      # Increment the count
60      count += 1
```

15 Zahlen in der Schleife erhöhen

Die while-Schleife läuft, bis der Zähler bei 6 steht, das heißt es gibt 5 Durchläufe, da der Zähler bei 1 anfängt. In jedem Durchlauf wird die Variable number erhöht und dann der Wert, der zur ursprünglichen Zahl addiert wird, und das Ergebnis ausgegeben. Danach wird der Zähler erhöht.

> **“Durch die Zähler-Variable wissen wir genau, wie oft die while-Schleife bereits durchlaufen wurde.”**

```
62  # Finish off by printing that we are exiting
63  print('\nExiting')
```

16 Und zum Schluss...

Im letzten Schritt geben wir aus, dass das Programm beendet wird. Das ist die letzte Zeile. Wir brauchen nichts weiter zu schreiben, da das Python-Programm automatisch beendet wird, wenn keine weiteren Zeilen zum Interpretieren mehr vorhanden sind.

```
Please enter your first name: Liam
Welcome Liam

Please enter a number: 12.12
The result of halving that number: 6.06
The result of doubling that number: 24.24
The result of squaring that number: 146.8944

Do you want to continue? (yes/no) yes

Continuing
Incrementing the number by 5

number + 1 = 13.12
number + 2 = 14.12
number + 3 = 15.12
number + 4 = 16.12
number + 5 = 17.12

Exiting
```

17 Bewundern Sie Ihr Werk

Wir sind mit der Programmierung jetzt fertig. Speichern Sie alle Änderungen ab und starten Sie das Programm mit der F5-Taste.

Das Kameramodul mit picamera steuern

Dave Jones erklärt uns, wie man seine Python-Kamerabibliothek für den Raspberry Pi einrichtet und verwendet.

Die picamera-Bibliothek ist eine Python-Bibliothek, die eine leichte Steuerung des Kameramoduls des Raspberry Pi erlaubt. Sie wurde ursprünglich entworfen, um eine geschicktere Methode für die Kamerasteuerung zu bieten, als die Kommandozeilen-Anwendungen raspistill und raspivid aus Skripten heraus laufen zu lassen. Seit die Bibliothek auf der Homepage des Raspberry Pi angeboten wird, wurde sie einige tausend Mal heruntergeladen und hat (natürlich) auch einige Fehlerberichte bekommen – wobei diese mit der Version 1.2 ausgebessert sein sollten.

Die Bibliothek hat bereits eine Rolle in mehreren faszinierenden Projekten gespielt, einschließlich Zeitraffervideos, bewegungssensorischen Überwachungskameras, Stop-Motion-Animationen und sogar tragbaren Digitalkameras mit integrierter Dropbox, die per Touchscreen gesteuert werden. In diesem Artikel erhalten Sie Einsicht in die Installation und Verwendung des Raspberry-Pi-Kameramoduls über picamera sowie in die Grundlagen von zwei der oben erwähnten Projekte: Zeitraffervideos und eine bewegungssensorische Überwachungskamera.

Wir gehen davon aus, dass Sie die Distribution Raspbian auf Ihrem Pi verwenden und die X-Window-Umgebung laufen haben (startx nach dem Einloggen). Auch wird angenommen, dass Sie das Kameramodul erfolgreich installiert und bereits mit raspistill oder raspivid getestet haben.

Erste Schritte

Öffnen Sie ein Terminal und führen Sie für die Installation der picamera-Bibliothek für Python 2 die folgenden Befehle aus. Einige Abhängigkeiten für Projekte, die wir später durchführen, werden hier bereits mitinstalliert:

```
pi@raspberrypi ~ $ sudo apt-get update
pi@raspberrypi ~ $ sudo apt-get install python-picamera python-numpy python-rpi.gpio python-opencv ffmpeg
```

Es gibt auch ein Python-3-Paket (python3-picamera), das neben Python 2 besteht. Der gesamte Code in diesem Artikel läuft unter jeder Version von Python (obwohl nicht alle Pakete von Drittanbietern für Python 3 verfügbar sind – insbesondere OpenCV).

Nun da picamera installiert ist, können Sie versuchen, es über eine interaktive Python-Sitzung anzuwenden. Wechseln Sie zu Ihrem Desktop-Verzeichnis und geben Sie Folgendes ein:

```
pi@raspberrypi ~ $ cd ~/Desktop
pi@raspberrypi ~/Desktop $ python
Python 2.7.3 (default, Jan 13 2013, 11:20:46)
[GCC 4.6.3] on Linux2
Type “help”, “copyright”, “credits” or “license” for more information.
>>> import picamera
```

```
>>> camera = picamera.PiCamera()
>>> camera.capture('picture.jpg')
>>> camera.close()
```

Drücken Sie abschließend Strg+D, um den Python-Interpreter zu verlassen. Sie sollten nun eine Datei namens picture.jpg auf Ihrem Desktop sehen.

Beachten Sie, dass wir die Kamera in der oberen Sitzung explizit geschlossen haben. Die Kamera muss vor dem Beenden eines Skripts immer geschlossen werden, aber die Ausführung ist in den meisten Skripts einfacher, wenn man die with-Anweisung von Python verwendet:

```
import picamera

with picamera.PiCamera() as camera:
    camera.capture('picture.jpg')
```

Wenn die with-Anweisung endet, wird die Kamera automatisch geschlossen.

Zeitraffervideos erstellen

Eines der einfachsten, aber lohnendsten Projekte, die Sie mit der Pi-Kamera ausführen können, ist die Erstellung eines Zeitraffervideos. Es gibt viele Prozesse in der realen Welt, die mit solch einer langsamen Geschwindigkeit stattfinden, dass es für einen Menschen schwierig ist zu verstehen, was dabei vor sich geht. Glücklicherweise hat Ihr Pi deutlich mehr Geduld! Wählen Sie ein interessantes Thema für die Aufnahme aus: das Verfaulen eines Obststücks, das Wachstum einer Pflanze, Anstieg und Absinken eines nahegelegenen Flusses oder vielleicht der Bau eines neuen Gebäudes.

Ein entscheidender Aspekt, wenn man ein Zeitraffervideo erstellt, ist, dass die Kamera ganz ruhig bleiben muss. Ein Halter für die Kamera zusammen mit einem dreibeinigen Stativ kann hierbei nützlich sein (siehe „Weiterführende Infos“ auf Seite 43). Versuchen Sie, den Raspberry Pi irgendwo zu platzieren, wo er durch nichts gestört werden kann.

Das Skript (**Abb. 01**) wird die Pi-Kamera veranlassen, fünf Tage lang einmal pro Stunde ein Bild im Format 1280 x 720 aufzunehmen und das Ergebnis in einer Abfolge chronologisch benannter Dateien zusammenzustellen. Ein Bild pro Stunde ist ein ordentlicher Ausgangspunkt für Zeitraffervideos, da bei eine Framerate von 24 fps (üblich bei Videos) der Tagessatz an Bildern (24 Stunden) in einer Sekunde komprimiert wird.

Die Konstanten oben im Skript können beeinflusst werden, um die Auflösung des Videoergebnisses, die Bildrate des Skripts und die Anzahl der insgesamt aufgenommenen Frames zu ändern. Beachten Sie, dass das Skript jedes Mal, wenn es ein Bild macht, die Kamera startet und sie später wieder schließt (denken Sie daran, dass die with-Anweisung das automatisch tut). Die Kamera erhöht deutlich den Stromverbrauch des Raspberry Pi, somit wäre deren permanenter Betrieb ungünstig, zumal wenn Sie Ihren Pi für Zeitraffer-Außenaufnahmen mit einem Akku betreiben möchten. Sie können das Skript wie folgt starten:

```
pi@raspberrypi ~ $ python timelapse.py
```

Am Ende der Aufnahme erzeugt das Skript das Zeitraffervideo durch Aufrufen des Dienstprogramms FFmpeg. Der Raspberry Pi ist bei der Codierung des Ausgabevideos sehr langsam (normalerweise wird mindestens eine halbe Stunde benötigt), doch wenn man bedenkt, dass das Skript zu diesem Zeitpunkt bereits seit fünf Tagen gelaufen ist, ist das auch noch zu verkraften.

Der Code von timelapse.py Abb. 01

```
from __future__ import division

import time
import subprocess
import picamera

VIDEO_RESOLUTION = (1280, 720)
VIDEO_DAYS = 5
FRAMES_PER_HOUR = 1
FRAMES = FRAMES_PER_HOUR * 24 * VIDEO_DAYS

def capture_frame(frame):
    with picamera.PiCamera() as cam:
        cam.resolution = VIDEO_RESOLUTION
        # Wir geben der Kamera ein paar Sekunden Zeit bis
        # zur Aufnahme, um sich einzustellen. Generell ist
        # dies direkt nach dem Initialisieren ratsam.
        time.sleep(2)
        cam.capture('frame%03d.jpg' % frame)

# Bilder aufnehmen
for frame in range(FRAMES):
    # Die Zeit vor der Aufnahme notieren
    start = time.time()
    capture_frame(frame)
    # Auf nächste Aufnahme warten. Wir berücksichtigen
    # die Zeit, die für die Aufnahme des Bildes benötigt
    # wird, während die Verzögerung berechnet wird.
    time.sleep(
        int(60 * 60 / FRAMES_PER_HOUR) - (time.time() - start)
        )

# Video erstellen
subprocess.call([
    'ffmpeg', '-y',
    '-f', 'image2',          # Bilder als Quelle verwenden
    '-i', 'frame%03d.jpg',   # Muster für Name der Inputdatei
    '-r', '24',              # Framerate für die Ausgabe
    '-vcodec', 'libx264',    # Output mit x264 codieren
    '-profile', 'high',      # H.264-Eigenschaften erlauben
    '-preset', 'slow',       # Hochwertiges Preset verwenden
    'timelapse.mp4',         #  Name der Outputdatei
    ])
```

Die Zeitreise-Kamera

Haftungsausschluss: Es wurden keine Gesetze der Physik während des Schreibens dieses Codes verletzt!

Als Nächstes werden wir eine Kamera bauen, die auf Knopfdruck die letzten 30 Sekunden – vor (!) dem Knopfdruck – speichert. Das mag nicht sonderlich plausibel klingen, ist aber in Wahrheit ziemlich simpel zu bewerkstelligen. Sobald das Skript startet, fängt es nämlich sofort an, ein Video mittels einer speziellen Streaming-Methode namens Ringpuffer aufzuzeichnen. Der Ringpuffer hat eine Kapazität für etwa 30 Sekunden Video. Wenn besagter Knopf gedrückt wird, schreibt das Skript die gesamte aktuell gespeicherte Aufzeichnung ab dem ersten Header-Frame im Puffer in eine Datei auf der Festplatte.

Der Ringpuffer wird von der picamera-Bibliothek in Form der PiCameraCircularIO-Klasse unterstützt. Unter der Oberfläche ist es gar kein echter Ringpuffer, vielmehr beginnt er, Bits vom Anfang des Puffers zu entfernen, sobald die Kapazität erreicht wurde. Die Codeauflistung (**Abb. 02**) zeigt, wie es weitergeht. Dieses Skript löst die Zeitreise-Kamera über einen mit dem GPIO-Pin 17 verbundenen Knopf aus (**Abb. 03**).

Welchen Nutzen hat so eine Zeitreise-Kamera? Es gibt einleuchtende Verwendungsmöglichkeiten bei der Überwachung: Die meisten Algorithmen für Bewegungsdetektoren sind nicht vollkommen, daher könnten Letztere, sobald sie eine Bewegung wahrnehmen, Entscheidendes bereits verpasst haben. Auch bei der Aufnahme sportlicher Hochleistungsmomente könnten Sie den verhofften Moment in der Zeit, in der Sie den Auslöser drücken, verpassen. Indem die vorhergehenden paar Sekunden zwischengespeichert werden, können Sie eine Kamera bauen, die Ihnen hilft, die Action nicht zu verpassen.

Sichtfeld

Wie kann ich mir über die Vorschau anzeigen lassen, was aufgenommen wird?
Wenn Sie mit der Kamera experimentieren, bemerken Sie vielleicht, dass sie ein größeres Sichtfeld (FoV) für die Aufnahme von Bildern verwendet als bei der Videoaufzeichnung oder Vorschauanzeige. Sie können die Vorschau dazu bringen, denselben FoV zu verwenden, indem Sie das Auflösungsvermögen der Kamera auf das Maximum stellen:

```
cam.resolution = (2592, 1944)
```

Wenn Sie danach Bilder mit einer niedrigeren Auflösung aufnehmen möchten, verwenden Sie den Parameter für Größenreduktion mit der Aufnahmemethode:

```
cam.capture('foo.jpg', resize=(1024, 768))
```

Das Kapitel zur Kamerahardware in der Dokumentation von picamera erklärt die Ursachen für die Abweichung des FoV.

Oben Der Vorschaubereich zeigt, dass hier ein schlechtes Bild herauskommen würde.

Bewegungsmeldung

Einer der mächtigeren Algorithmen, die Sie mit der Kamera des Raspberry Pi ausführen können, ist die Bewegungsmeldung. Diese Fähigkeit kann auf eine Vielzahl von Projekten angewandt werden: die naheliegende Überwachungskamera, Videoaufnahmen in der Tierwelt, die Konstruktion einer Hochgeschwindigkeitskamera oder (in Kombination mit der am GPIO-Port angeschlossenen Peripherie) die Steuerung von Dingen als Reaktion auf Bewegung.

Die klassische Methode der Bewegungsmeldung ist die „Hintergrundsubtraktion". Zuerst nehmen wir ein Bild auf und nennen es „Hintergrund". Dann halten wir ein anderes fest und ziehen die Pixelwerte des ersten davon ab. Wenn sich nichts geändert hat, wird das Ergebnisbild völlig schwarz sein. In der Praxis verhält es sich natürlich nicht ganz so simpel. Die Kamera erzeugt nicht perfekte Bilder (es ist „Rauschen" vorhanden), also gebrauchen wir einen Erosionsfilter, um das Rauschen zu reduzieren und bestimmen einen Schwellenwert, an dem wir entscheiden, dass Bewegung entdeckt worden ist.

Außerdem können wir nicht dasselbe Hintergrundbild immer wieder verwenden, sonst würde eine Änderung der Szene (wie das Einschalten von Licht in einem Zimmer oder die Bewegung der Sonne bei Außenkameras) bewirken, dass man permanent Bewegungen erfasst. Man könnte das Bild kurz vor dem zu Vergleichenden verwenden, aber das würde auf einen Bewegungsmelder hinauslaufen, den man ziemlich leicht durch langsame Bewegungen austricksen kann (die winzigen Bewegungen werden durch den oben genannten Erosionsfilter weggelöscht). Stattdessen verwenden wir einen gleitenden Mittelwert der letzten paar Frames, sodass sich der Hintergrund langsam an Änderungen anpasst.

In dem Skript (**Abb. 04**) werden wir der Einfachheit halber mit Graustufenbildern arbeiten; speziell mit dem Y-Eingang einer YUV-Rohaufnahme (siehe „Weiterführende Infos"). Sie können mit vollfarbigen Aufnahmen arbeiten, wenn Sie das möchten, aber das wird die benötigte Verarbeitungsdauer erhöhen und erfahrungsgemäß die Empfindlichkeit des Algorithmus beeinträchtigen (denken Sie an einen Schwarzweiß-Film im Gegensatz zu einem farbigen – erkennen Sie Bewegung in Farbe leichter?). Sie können das Skript wie folgt ausführen:

```
pi@raspberrypi ~ $ python motion.py
```

Wenn das Skript läuft, versuchen Sie, Ihre Hand vor der Kamera hin- und herzubewegen. Mit etwas Glück sollten Sie sehen, wie das Terminal eine Nachricht als Antwort ausgibt. Versuchen Sie, den Bewegungsmelder-Algorithmus auszutricksen, etwa durch sehr langsame Bewegungen. Spielen Sie mit den Werten, die die Algorithmus-Empfindlichkeit kontrollieren, vor allem background_delta, erosion und threshold.

Als Faustregel gilt, dass die Auflösung wenig mit der Bewegungsmeldung zu tun hat. Sie können ziemlich niedrige Auflösungen wählen (beispielsweise 160 x 90 wie im obigen Skript) und immer noch angemessene Ergebnisse erzielen, aber nur, wenn das Objekt, dessen Bewegung Sie erfassen wollen (etwa ein Mensch oder ein Tier), einen bedeutenden Teil des Kamerasichtfeldes einnimmt.

Wenn Sie jedoch Bilder oder Videos als Reaktion auf Bewegungsmeldungen aufzeichnen wollen (eine typische Anforderung an eine Überwachungskamera), möchten Sie wahrscheinlich eine höhere Auflösung als 160 x 90. Deshalb stellt das obige Skript die Kamera auf eine hohe

Der Code von timetravel.py

Abb. 02

```
#!/usr/bin/env python

import io
import picamera
import RPi.GPIO as GPIO
import datetime as dt

BUTTON_PIN = 17

def button_callback(channel):
    assert channel == BUTTON_PIN
    print('Button pressed')
    filename = dt.datetime.now().strftime('%Y%m%d-%H%M%S.h264')
    # Ringpuffer schließen und Ausgabe-Stream
    # als Binärdatei öffnen
    with stream.lock, io.open(filename, 'wb') as output:
        # Ersten Header-Frame im Ringpuffer finden
        for frame in stream.frames:
            if frame.header:
                stream.seek(frame.position)
                break
        # Von hier zum Ende des Ringpuffers in die
        # Ausgabedatei effizient kopieren
        while True:
            data = stream.read1()
            if not data:
                break
            output.write(data)
        print('Wrote video to %s' % filename)
        # Inhalt des Ringpuffers löschen, sodass keine
        # Frames wiederholt werden, wenn der Knopf
        # noch einmal schnell gedrückt wird
        stream.seek(0)
        stream.truncate()

#  Knopf als Input setzen
GPIO.setmode(GPIO.BCM)
GPIO.setup(BUTTON_PIN, GPIO.IN, GPIO.PUD_UP)

# Kameraaufnahme starten zum Ringpuffer mit
# Platz für ein 30 Sekunden langes 720p-Video
with picamera.PiCamera() as cam:
    cam.resolution = (1280, 720)
    stream = picamera.PiCameraCircularIO(cam, seconds=30)
    cam.start_recording(stream, format='h264')
    GPIO.add_event_detect(BUTTON_PIN, GPIO.RISING,
            callback=button_callback, bouncetime=200)
    try:
        while True:
            cam.wait_recording(1)
    finally:
        GPIO.remove_event_detect(BUTTON_PIN)
        cam.stop_recording()
```

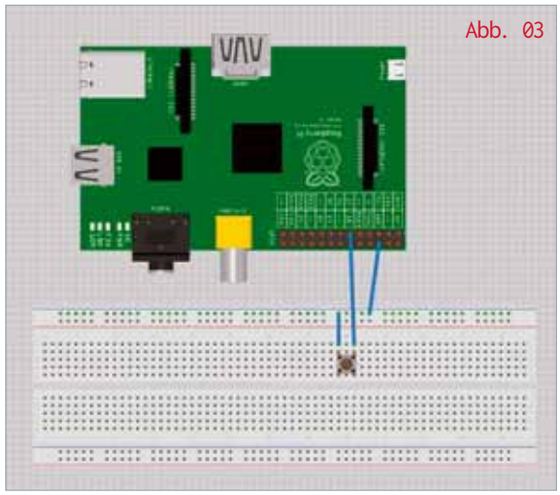


Abb. 03

Links Beispiel einer Verkabelungsschaltung, die einen mit dem GPIO-Pin 17 verbundenen Kameraauslöser zeigt.

Auflösung (1280 x 720) ein und verwendet anschließend die Option zur Größenreduktion der capture-Methode, um die für den Bewegungsmeldungs-Algorithmus erforderlichen Bilder zu verkleinern.

Ein letzter zu beachtender Punkt ist, dass dieses Skript portbasierte Videoaufnahmen verwendet. Während es eine anständige Aufnahmerate bietet (bis hin zur Framerate der Videoaufnahmen), führt es zu einem reduzierten Sichtfeld, weil wir Video aufnehmen, was die Kamera angeht (lesen Sie dazu den Kasten „Sichtfeld“ in diesem Artikel).

Es gibt einige Anleitungen in der Dokumentation von picamera, die nach einer Routine in Bewegungsmeldung verlangen und den oben vorgestellten Ringpuffer anwenden. Versuchen Sie, den hier verzeichneten Code mit jenen Anleitungen zu verbinden, um eine Überwachungskamera herzustellen, die nur dann Filmmaterial aufzeichnet, wenn sie Bewegung erfasst.

Weiterführende Infos

- Das GitHub-Zuhause für picamera samt Infos zu Updates
  github.com/waveform80/picamera
- Die Dokumentation von picamera einschließlich zahlreicher Anleitungen, die eine Reihe von Sachkenntnissen abdecken
  picamera.readthedocs.org
- Installationsanweisungen für das Kameramodul des Pi
  raspberrypi.org/help/camera-module-setup/
- Ein Kamerahalter (etwa für einen Dreibein) ist für viele Kameraprojekte sehr nützlich
  modmypi.com/raspberry-pi-camera/pibow-raspberry-pi-camera-mount
- Einige Pi-Geräte haben integrierte Kamerahalter und sogar verbesserte Linsen
  exp-tech.de/nwazet-pi-camera-box
- GorillaPod ist ein ausgezeichnetes tragbares Dreibein
  joby.com/gorillapod
- YUV-Bildkodierung wird intern von der Kamera des Pi verwendet und bildet die effizienteste Form der Rohaufnahme
  en.wikipedia.org/wiki/YUV

Einer der mächtigeren Algorithmen, die Sie mit der Pi-Kamera ausführen können, ist die Bewegungsmeldung.

Der Code von motion.py

Abb. 04

```
#!/usr/bin/env python

from __future__ import division

import io
import time
import picamera
import cv2
import numpy as np

class MotionDetector(object):
    def __init__(self, camera, resolution=(160, 90), threshold=20, ↵
erosion=10, background_delta=0.3):
        self.camera = camera
        self.resolution = resolution
        self.raw_resolution = (
                (resolution[0] + 31) // 32 * 32,
                (resolution[1] + 15) // 16 * 16,
                )
        self.raw_bytes = self.raw_resolution[0] * self.raw_resolution[1]
        self.threshold = threshold
        self.erosion = erosion
        self.background_delta = background_delta
        self.background = None

    def _get_erosion(self):
        return (self.erosion_filter.shape[0] - 1) // 2
    def _set_erosion(self, value):
        self.erosion_filter = cv2.getStructuringElement(
                cv2.MORPH_RECT, (value * 2 + 1, value * 2 + 1))
    erosion = property(_get_erosion, _set_erosion)

    def poll(self):
        stream = io.BytesIO()
        self.camera.capture(
            stream, format='yuv', resize=self.resolution, use_video_ ↵
port=True)
        data = stream.getvalue()[:self.raw_bytes]
        image = np.fromstring(data, dtype=np.uint8).reshape(
                (self.raw_resolution[1], self.raw_resolution[0])
                )
        cv2.erode(image, self.erosion_filter)
        image = image.astype(np.float)
        if self.background is None:
            self.background = image
            return False
        else:
            diff = cv2.absdiff(image, self.background)
            diff = diff.astype(np.uint8)
            diff = cv2.threshold(
                    diff, self.threshold, 255, cv2.THRESH_BINARY)[1]
            result = diff.any()
            cv2.accumulateWeighted(
                    image, self.background, self.background_delta)
            return result
with picamera.PiCamera() as cam:
    cam.resolution = (1280, 720)
    # Kamera vor dem Beginnen warm laufen lassen
    time.sleep(2)
    detector = MotionDetector(cam)
    while True:
        if detector.poll():
            print("I see you!")
```

KABEL-SALAT
Am Anfang sieht es aus wie ein Chaos aus Drähten, aber wenn die Motoren und Sensoren dazukommen und Sie dabei stets testen und prüfen, blicken Sie bald durch.

Bauen Sie einen Raspberry-Pi-Roboter

Ein Linux-betriebener Roboter für schlappe 200 €.

Für Roboterbegeisterte sind tolle Zeiten angebrochen. Noch bis vor kurzem musste man Tausende von Euros investieren, um einen einfachen Roboter zu bauen, der sich bewegt, Daten sammelt und auf äußere Reize reagiert. Mit Geräten wie dem Raspberry Pi kann man so einen Roboter heute für einen Bruchteil der früheren Kosten bauen. Und wenn Sie schon einen Raspberry Pi besitzen und sich etwas mit Elektronik beschäftigt haben, dann brauchen Sie kaum mehr als 130 €, um Ihr Roboterprojekt zu finanzieren.

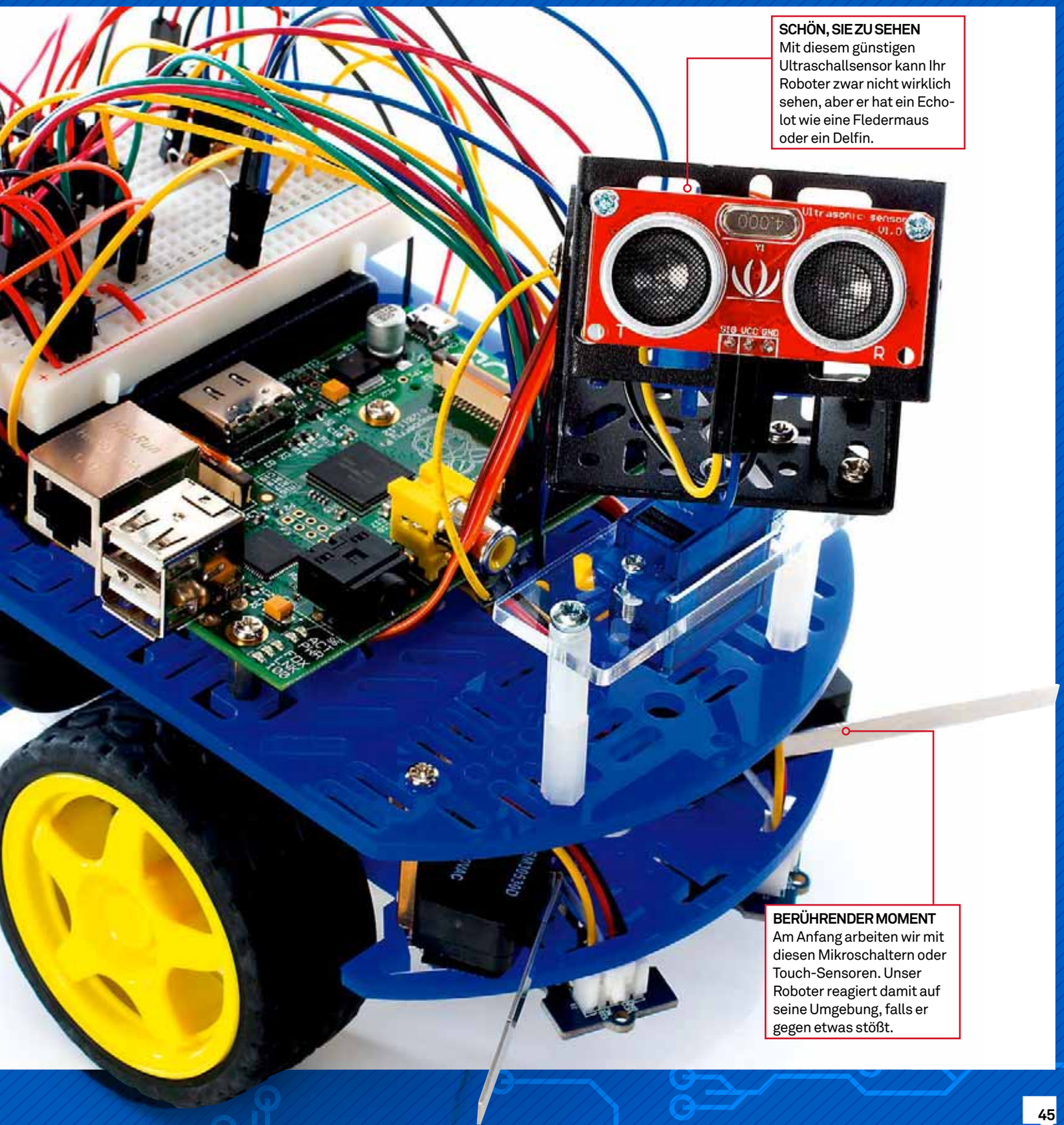
In diesem Artikel beschäftigen wir uns mit Elektronik, Programmierung und den Grundzügen der künstlichen Intelligenz. Sie benötigen kein Vorwissen, aber das Thema inspiriert Sie hoffentlich, mehr zu lernen.

Folgen Sie unserer in Etappen gegliederte Bauanleitung auf den nächsten 16 Seiten, dann steht Ihnen und Ihrem Roboter das Tor zu so spannenden Feldern wie Navigation, Irrgärten und künstlicher Intelligenz weit offen.

ALLES AN BORD
Das preisgünstige Fahrgestell mit Rädern und Motoren ist sehr beliebt. Und es hat schon einen Platz für den USB-kompatiblen Akkupack für den Raspberry Pi.

„Wir beschäftigen uns mit Elektronik, Programmierung und künstlicher Intelligenz."

ACHTUNG
Obwohl bei diesem Projekt auf hohe Sicherheit geachtet wurde, kann beim Nachbau dennoch etwas schiefgehen. Daher können Redaktion und Verlag nicht für etwaige Schäden an Geräten wie dem Raspberry Pi und anderer Hardware haften.



SCHÖN, SIE ZU SEHEN
Mit diesem günstigen Ultraschallsensor kann Ihr Roboter zwar nicht wirklich sehen, aber er hat ein Echolot wie eine Fledermaus oder ein Delfin.

BERÜHRENDER MOMENT
Am Anfang arbeiten wir mit diesen Mikroschaltern oder Touch-Sensoren. Unser Roboter reagiert damit auf seine Umgebung, falls er gegen etwas stößt.

Das alles brauchen Sie

Werkzeug, Bauteile und Know-how – so starten Sie durch!

Wir möchten Sie überzeugen, dass es weder besonders schwierig noch teuer ist, einen Roboter mit einem Raspberry Pi zu bauen. Trotzdem ist es eine technische Herausforderung, bei der Sie auf eine gute Auswahl an elektronischem Zubehör für den Prototyp, Spezial-Chips und diverse Werkzeuge angewiesen sind.

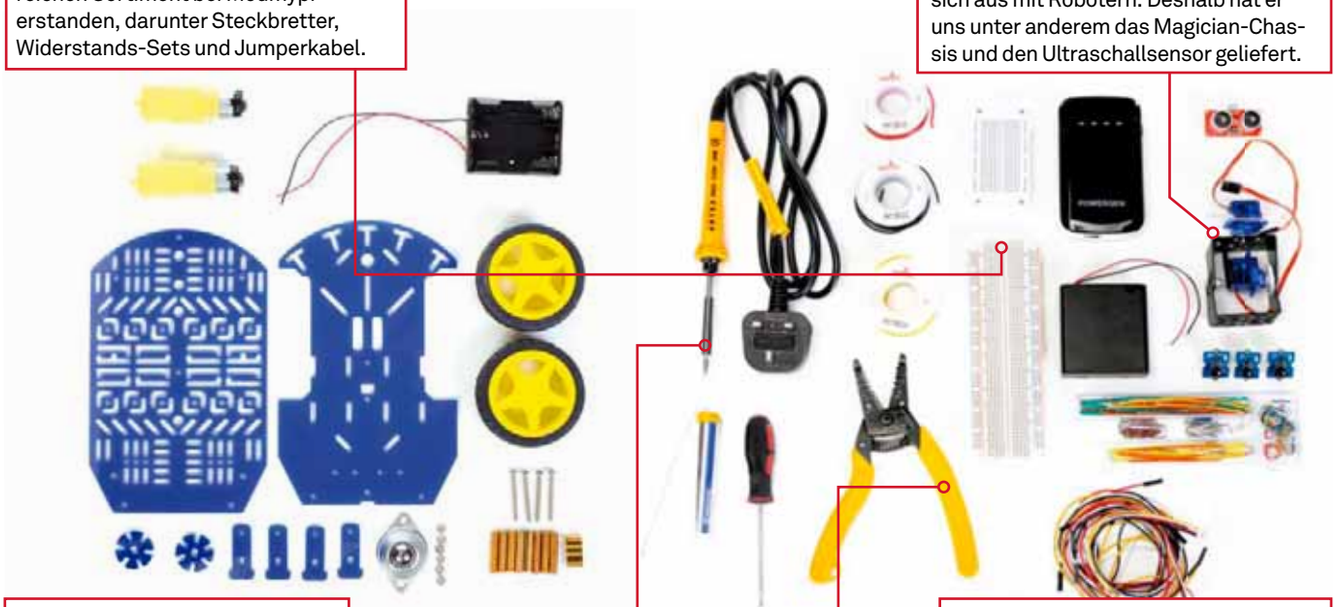
Unten sind viele der Bauteile aufgeführt, aus denen unser Roboter besteht. Lassen Sie sich aber nicht davon abhalten, auch andere Teile zu verwenden. Bei diesem anspruchsvollen Projekt lernen Sie, wie Sie auf Technologie zugreifen und sie kontrollieren. Mit diesem Wissen und dem passenden Code können Sie fast alle anderen digitalen oder analogen Sensoren verwenden.

Sie benötigen einen kleinen Kreuzschlitz-Schraubenzieher, eine anständige Drahtschere und einen Lötkolben. Der Aufbau des Roboters braucht nicht viel Löten, doch später müssen viele Sensoren und Bedienungselemente verlötet werden.

Sie können wochenlang nach dem perfekten Bausatz mit dem besten Service zum günstigsten Preis suchen und im Internet Kundenrezensionen lesen. Oder Sie können einfach dort bestellen, wo wir uns das Zubehör für unseren Bausatz gekauft haben...

MODMYPI
modmypi.com
Wir haben sehr viel aus dem umfangreichen Sortiment bei Modmypi erstanden, darunter Steckbretter, Widerstands-Sets und Jumperkabel.

DAWN ROBOTICS
dawnrobotics.co.uk
Alan Broun von Dawn Robotics Ltd kennt sich aus mit Robotern. Deshalb hat er uns unter anderem das Magician-Chassis und den Ultraschallsensor geliefert.



PIMORONI
shop.pimoroni.com
Was Gehäuse, Kabel und Zubehör betrifft, ist Pimoroni die Nr. 1, und sie haben auch eine große Auswahl an Sensoren.

CPC
cpc.farnell.com
Wir haben den Raspberry Pi, Mikroschalter und einige Werkzeuge von CPC bezogen. Sie haben eine umwerfende Auswahl und unübertroffene Preisangebote.

Python-Code schreiben und ausführen

Schreiben Sie Ihren Python-Code in der Entwicklungsumgebung, die Sie am besten kennen, egal ob IDLE, Geany oder ein anderer Editor. Oder tippen Sie in LeafPad einfach ein Stück Code ein, das Sie als .py-Datei speichern. Das geht schneller, ist bequemer, und wenn Sie neu beim Programmieren sind, bringt es Ihnen mehr für später.

Um Ihre Skripte und unsere Beispiele auszuführen, müssen Sie erweiterte Benutzerrechte verwenden, sonst kann Ihr Code nicht mit den GPIO-Pins interagieren. Gehen Sie deshalb im Terminal zu Ihrer Datei und geben Sie `sudo python file.py` ein (wobei „file“ für den Namen Ihrer Code-Datei steht).

Zugang per SSH

Am besten können Sie Ihren Raspberry-Pi-Roboter programmieren, wenn Sie den Pi mit einem WLAN-Dongle ausstatten und von einem externen Linux-Computer per SSH auf ihn zugreifen. Dazu brauchen Sie die IP-Adresse des Pi. Sie finden diese, indem Sie ein Terminal öffnen und, sobald Sie über WLAN verbunden sind, eintippen:

```
ifconfig
```

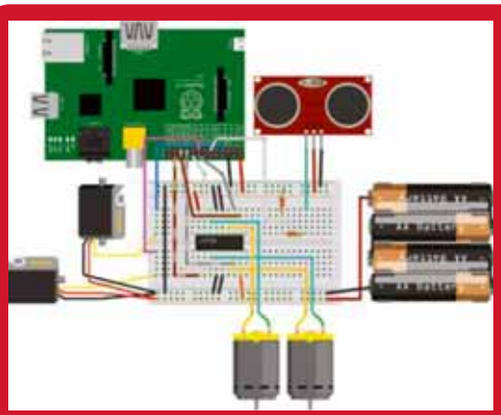
Suchen Sie sich im Output die Stelle, wo es um WLAN geht, und notieren Sie die IP-Nummer. Öffnen Sie ein Terminal-Fenster auf Ihrem Computer und geben Sie ein:

```
ssh pi@xxx.xxx.x.xxx
```

Setzen Sie dabei statt der x'e die IP-Adresse ein. Falls Sie den Standardnamen „pi" geändert haben, tippen Sie den neuen Namen ein. Geben Sie Ihr Passwort (standardmäßig ist „raspberry" eingestellt) ein, und Sie sind mit dem Pi verbunden. Führen Sie nun Ihre Python-Skripte ganz normal durch. Sie können auch

```
nano file.py
```

eingeben, um Ihre Dateien zu bearbeiten, bevor Sie sie ausführen.



Man kann Elektronik für gewöhnlich auf zweierlei Weise darstellen – als komplexe Schaltkreis-Diagramme oder als schwer verständliche Fotos von Steckbrettern. Zum Glück gibt es eine dritte Option, die das Beste aus beiden Welten vereint: Fritzing (**fritzing.org**). Fritzing ist ein Open-Source-Projekt für alle, die mit Elektronik arbeiten. Mit dem Tool können Sie Komponenten auswählen und sie auf ein Dokument ziehen, das Sie dann als Bild ausgeben. Wir hoffen, Sie haben bei Ihrem Projekt genauso viel Spaß mit Fritzing wie wir mit unserem!

Der GPIO-Port

Ohne die GPIO-Pins auf Ihrem Raspberry Pi würden Sie nicht weit kommen.

Die GPIO-Pins (General Purpose Input/Output, auf Deutsch Allzweckeingabe/-ausgabe) auf Ihrem Raspberry Pi bilden die zentrale Schnittstelle des Projekts. Ohne sie können wir nicht mit den Motoren, Sensoren und Bedienungselementen interagieren. Doch mithilfe der Raspberry-Pi-GPIO-Python-Bibliothek ist es sehr einfach, die Pins zu benutzen, vorausgesetzt man verwendet für jede Aufgabe den richtigen Pin. Das ist gar nicht so leicht, denn die Kontaktstifte können verschiedene Bezeichnungen tragen. So heißt GPIO 18 auch Pin 12 und PCM_CLK. Um Verwechslungen zu vermeiden, benutzen wir das Belegungsmuster von Broadcom und nicht die Board-Konvention. Deshalb steht in unseren Code-Listings stets:

```
GPIO.setmode(GPIO.BCM)
```

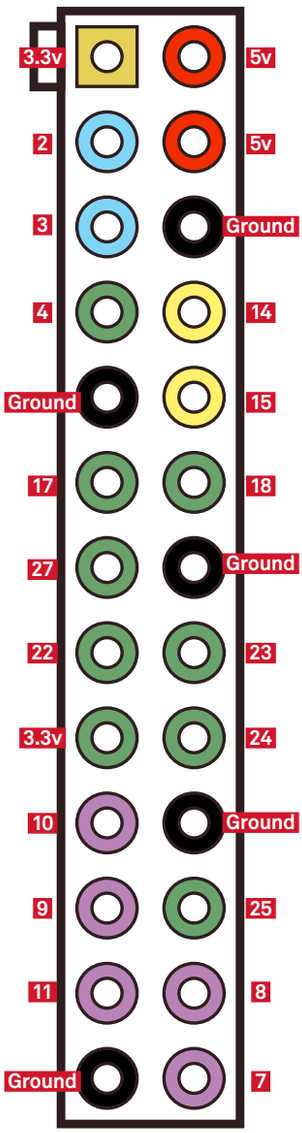
Die Sache wird noch verwirrender, weil sich bestimmte Pin-Nummern zwischen den Boards von Revision 1 und Revision 2 geändert haben. Das nebenstehende Diagramm basiert auf Revision 2 (der Raspberry Pi mit 512 MB Arbeitsspeicher und Befestigungslöchern). Die Version für Revision 1 finden Sie im Netz, wenn Sie nach „Raspberry Pi GPIO" suchen. Das hört sich kompliziert an, doch am besten entscheidet man sich für eine Belegungs-Konvention und bleibt dann dabei!

HIER IST OBEN!
Das obere Ende des GPIO-Ports ist das am nächsten bei der SD-Karte.

BCM, BCM, BCM!
Wir verwenden die Broadcom-Pin-Nummern. Die Belegung ist anders als das Pin-System des Boards, das Sie ebenfalls verwenden können.

DAS IST REV 2
Je nach der Revision Ihres Raspberry Pi können sich Pin-Nummern ändern. Unbedingt prüfen!

LILA PINS
Diese Pins kann man benutzen, doch sie sind für serielle Verbindungen vorgesehen.



Wir empfehlen: Ihr Pi braucht einen WLAN-Dongle und eine SSH-Verbindung.

Aufbau des Motorschaltkreises

Schritt Nr. 1 ist ein einfacher Schaltkreis für den Raspberry Pi.

Die Grundfähigkeit unseres Roboters ist Bewegung – dafür brauchen wir die Motoren, mit denen das Magician-Chassis ausgestattet ist. Motoren gibt es in allen möglichen Formen und Größen, Typen und Modellen. Bei diesem Projekt verbinden wir zwei Gleichstrom-Motoren sicher mit dem Raspberry Pi.

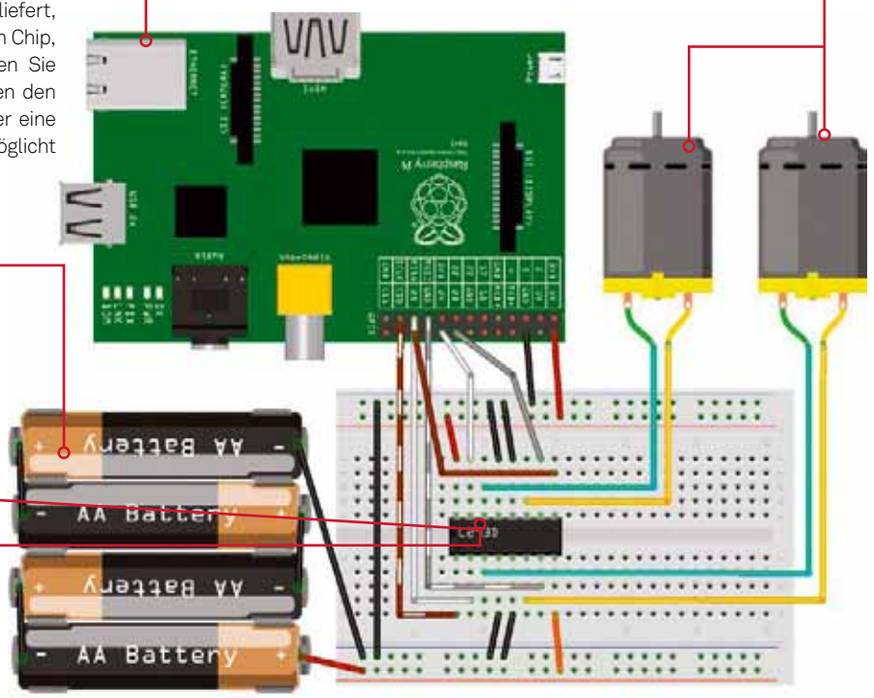
Weil der Pi nur eine begrenzte Menge an Strom liefert, brauchen wir zusätzliche Batterien und einen kleinen Chip, um die Motoren an- und auszuschalten. Versorgen Sie diese niemals vom Pi aus mit Strom! Wir verwenden den Motortreiber L293D, auch H-Brücke genannt. Dieser eine Chip kümmert sich um die Stromregelung und ermöglicht eine bi-direktionale Kontrolle von zwei Motoren.

RASPBERRY PI
Läuft mit Modell B Rev 1 und Rev 2 sowie Modell A des Raspberry Pi.

MEHRERE MOTOREN
Der Motortreiber kann zwei Gleichstrom-Motoren antreiben, was eine unabhängige Kontrolle ermöglicht.

MEHR STROM
Die Motoren laufen mit vier Mignon-Batterien. Sie erzeugen 6 Volt, perfekt für einen Kleinroboter.

MOTORTREIBER
Der L293D sitzt in der Mitte des Boards. Er leistet die meiste Arbeit.



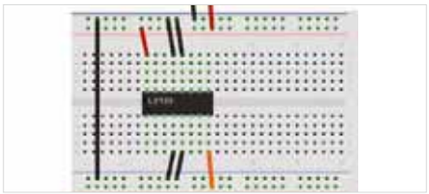
Liste der Bauteile

- Raspberry Pi
- Steckbrett
- 2 x Gleichstrom-Motoren
- L293D-IC
- Jumperkabel
- 4 x Mignon-Batterien
- Batteriehalterung

ACHTUNG
Verbinden Sie niemals (!) die Motoren direkt mit dem Raspberry Pi, denn dadurch könnte der Hauptprozessor beschädigt werden. Dann wäre Ihr Pi nicht mehr als ist ein teurer (wenngleich hübscher) Briefbeschwerer.

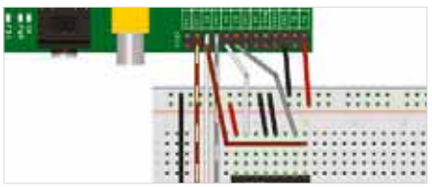
01 Einbau des L293D

Platzieren Sie den L293D-Chip in der Board-Mitte und befestigen Sie die roten und schwarzen Stromkabel. Achten Sie auf die Pins. Das orangefarbene Kabel ist für die Batterien.



02 Datenleitungen konfigurieren

Überprüfen Sie, ob alle Verbindungen an den richtigen Kontaktstiften stecken. Von den GPIO-Pins des Pi sollten sechs Kabel zu den Input-Pins des L293D verlaufen. Sie sind verantwortlich für die Motoren.



03 Schaltkreis schließen

Lassen Sie beim Anschließen der Motoren genügend Spielraum, falls sie sich anders herum drehen als erwartet. Schließen Sie die Batterien an und verbinden Sie sie mit dem Raspberry Pi.

Erster Motorentest

Der Motorschaltkreis steht, jetzt bringen wir Ihre Motoren zum Laufen.

Die Steuerung der Motoren erfolgt mit Python – mit einer Bibliothek namens RPi.GPIO ist das kein Problem. Sie wird in Ihr Skript importiert und erledigt das An- und Ausschalten. Die Skript-Bibliothek kann auch Input von Sensoren und Schaltern verarbeiten. Aber zuerst bringen wir unsere Motoren zum Rotieren, damit Bewegung in die Sache kommt.

Als Erstes importieren wir die benötigte Skript-Bibliothek, RPi.GPIO. Damit das Skript unterbrochen werden kann, brauchen wir die „sleep"-Funktion aus der Bibliothek „time". Dann teilen wir dem Skript mit, welche Pin-Nummerierung wir verwenden. Beim Raspberry Pi gibt es zwei Arten, die GPIO-Pins zu nummerieren: „board" korrespondiert mit den tatsächlichen Steckstellen der Stifte auf dem Board; „BCM" ist das Nummernschema des Prozessors. Wir verwenden das BCM-System.

Um Verwechslungen zu vermeiden, ist es sinnvoll, den Pins Namen zu geben. Wir verwenden die L293D-Pin-Namen. Jeder Motor braucht drei Pins: A und B, um die Drehrichtung zu steuern, und „Enable" als An/Aus-Schalter. Zudem können wir zur Steuerung der Motorengeschwindigkeit auf dem Enable-Pin PWM verwenden. Dazu später mehr.

Jetzt müssen wir noch festlegen, dass die Pins Ausgänge sind, denn wir senden das Signal vom Raspberry Pi aus. Um einen Pin anzuschalten – 1 oder HIGH-Stellung – befehlen wir dem Raspberry Pi, den Pin auf HIGH zu schalten. Um ihn auszuschalten, stellen wir den Pin auf LOW. Sind die Pins eingestellt, unterbrechen wir das Skript mit **time.sleep()**. So können die Motoren ein paar Sekunden laufen und dann erst die Richtung wechseln.

> **Um Verwechslungen zu vermeiden, ist es sinnvoll, den Pins Namen zu geben.**

Code-Listing für den Motorschaltkreis

BEGINN
Dies sind die GPIO-Pin-Nummern, die wir für unsere Motoren benutzen. Wir haben Sie der Übersichtlichkeit halber gemäß der L293D-Konvention benannt.

OUTPUT EINSTELLEN
Da wir möchten, dass die Motoren etwas tun, müssen wir Python mitteilen, dass es sich hier um Output, nicht Input handelt.

PINS FÜR BEWEGUNG
Hier sagen wir dem L293D, welche Pins an sein müssen, um eine Bewegung – vorwärts, rückwärts, anhalten – zu erzeugen.

```
import RPi.GPIO as GPIO
from time import sleep

GPIO.setmode(GPIO.BCM)

Motor1A = 24
Motor1B = 23
Motor1E = 25

Motor2A = 9
Motor2B = 10
Motor2E = 11

GPIO.setup(Motor1A,GPIO.OUT)
GPIO.setup(Motor1B,GPIO.OUT)
GPIO.setup(Motor1E,GPIO.OUT)

GPIO.setup(Motor2A,GPIO.OUT)
GPIO.setup(Motor2B,GPIO.OUT)
GPIO.setup(Motor2E,GPIO.OUT)

print "Going forwards"
GPIO.output(Motor1A,GPIO.HIGH)
GPIO.output(Motor1B,GPIO.LOW)
GPIO.output(Motor1E,GPIO.HIGH)

GPIO.output(Motor2A,GPIO.HIGH)
GPIO.output(Motor2B,GPIO.LOW)
GPIO.output(Motor2E,GPIO.HIGH)

print "... for 2 seconds."
sleep(2)

print "Going backwards"
GPIO.output(Motor1A,GPIO.LOW)
GPIO.output(Motor1B,GPIO.HIGH)
GPIO.output(Motor1E,GPIO.HIGH)

GPIO.output(Motor2A,GPIO.LOW)
GPIO.output(Motor2B,GPIO.HIGH)
GPIO.output(Motor2E,GPIO.HIGH)

print "... for 2 seconds"
sleep(2)

print "And stop before cleaning up"
GPIO.output(Motor1E,GPIO.LOW)
GPIO.output(Motor2E,GPIO.LOW)

GPIO.cleanup()
```

Den vollständigen Programmcode gibt es hier als Zip-Archiv: bit.ly/1mV2FmG

04 Skript-Erstellung

Um das erste Skript zu erstellen, loggen Sie sich in Ihren Raspberry Pi ein (Nutzername „pi", Passwort „raspberry". Tippen Sie `nano motortest.py` ein. Es öffnet sich der nano-Texteditor.

05 Code speichern

Geben Sie den Code ein. Groß- oder Kleinschreibung macht einen Unterschied. Rücken Sie den Code durchgehend mit einer Leerstelle ein. Zum Schluss drücken Sie Strg+X, dann Y, um zu speichern.

06 Motorentest

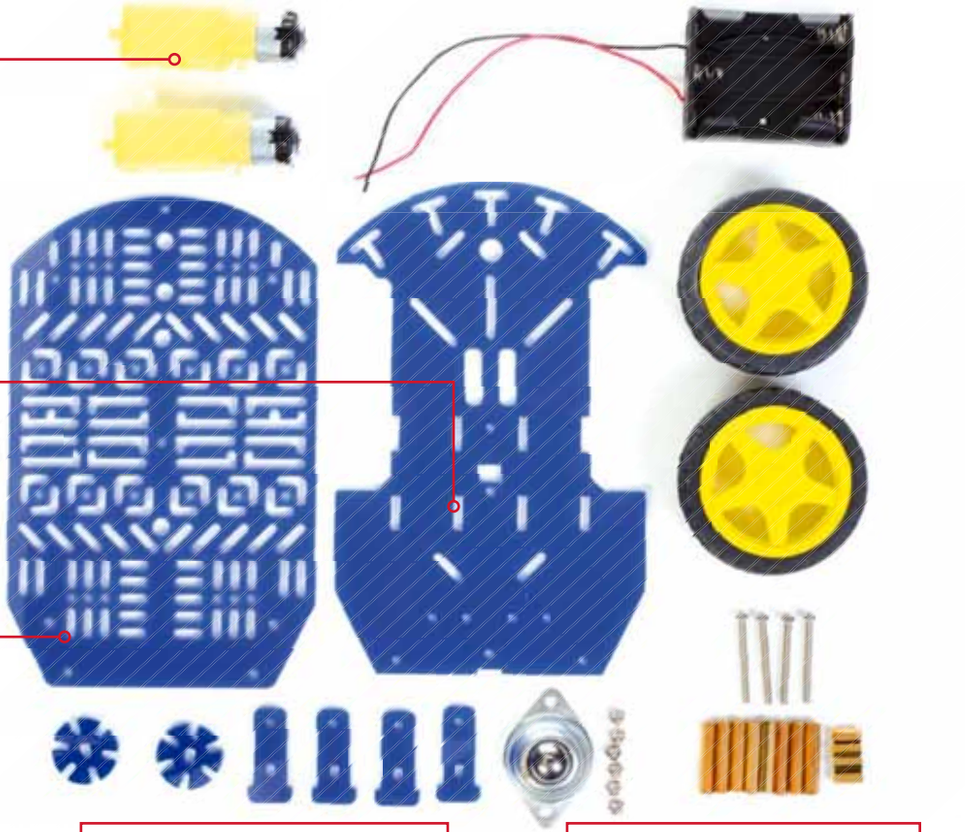
Führen Sie das Skript aus. Tippen Sie `sudo python motortest.py`. Wenn es nicht funktioniert, überprüfen Sie, ob die Kabel in den richtigen Pins stecken und die Batterien aufgeladen sind.

Montage des Roboter-Fahrgestells

Der Motorschaltkreis funktioniert – jetzt bauen wir unseren Raspberry-Pi-Roboter.

Kein Roboter ohne ein Gestell, auf das alle Teile gebaut werden. Deshalb brauchen wir ein Chassis. Chassis sind in verschiedenen Größen und Ausführungen erhältlich. Wir verwenden eins der flexibelsten und am weitesten verbreiteten, das Dagu Magician.

Der Bausatz enthält zwei Montageplatten, zwei Motoren und zwei Räder, dazu eine Batteriebox – die perfekte Grundausstattung für die meisten einfachen Roboter. Sobald unser Roboter „steht", können wir ihn mit neuen Motorfunktionen ausstatten, bevor wir Sensoren hinzufügen.



01 BAUTEILE
Breiten Sie die Bauteile vor sich aus. Können Sie alles zuordnen? Die Montage ist unkompliziert, nur der Einbau der Motorhalterung erfordert etwas Sorgfalt.

02 MOTORHALTERUNG
Stecken Sie eine Klammer in die Platte, befestigen Sie den Motor mit der zweiten Klammer. Die Schraube geht durch die Löcher, die Mutter wird von der Rückseite aufgeschraubt.

03 ZUSAMMENBAU
Führen Sie die Motorkabel durch das Chassis. Befestigen Sie die obere Platte mit Abstandhaltern und Schrauben. Darauf kommt die Laufrolle.

04 VERKABELUNG
Wenn alles montiert ist, stecken Sie den Raspberry Pi wieder ein und fahren Sie ihn hoch. Führen Sie das Test-Skript für den Motorschaltkreis aus.

05 STECKBRETT
Die meisten Boards haben ein Klebepad an der Unterseite. Nehmen Sie Blu-Tack, um das Board nicht dauerhaft anzukleben. Befestigen Sie es an der Vorderseite.

06 Pi-MONTAGE
Der Raspberry Pi Rev 2 und einige Gehäuse besitzen Montagelöcher an der Unterseite. Dort können Sie etwa die Batteriepacks gut befestigen.

Montage-Tipps

Nehmen Sie sich Zeit
Auch wenn Sie sofort loslegen möchten, in der Annahme, Sie wüssten alles, auch ohne einen Blick in die Bauanleitung zu werfen – lesen Sie zuerst den Artikel!

Modifikationen sind willkommen
Lassen Sie sich vom Standarddesign nicht einschränken. Wenn Sie neue Löcher für die Sensoren brauchen, messen Sie zweimal, bevor Sie sägen, aber modifizieren Sie nach Ihren Wünschen.

So viele Möglichkeiten
Es gibt etliche Roboterplattformen. Vier Räder, Fahrachsen und sogar Sechsbeiner sind möglich. Eine tolle Roboterplattform ist frindo – mehr auf **frindo.org**.

Bewegungsfunktionen in Python

Für einen richtigen Roboter reicht unser kleiner Motortest nicht. Deshalb geben wir weitere Bewegungsfunktionen hinzu, die man immer brauchen kann.

Den vollständigen Programmcode gibt es hier als Zip-Archiv: bit.ly/1kj6PKw

Unser Roboter sieht genial aus und ist überall richtig verschaltet (außer die Motoren, die wir vielleicht noch umbauen). Nun schließen wir den Raspberry Pi an und schreiben unser erstes Skript, um den Roboter zu steuern. Mit dem einfachen Motortest kann man wunderbar prüfen, ob die Motoren funktionieren, und das Skript ermöglicht grundsätzliche Bewegung. Aber wir möchten den Roboter vollständig und präzise bewegen können. Dazu erstellen wir unsere eigenen Funktionen.

In Python gruppiert man sich wiederholende Aktionen in eine Definition oder einen **def**-Block. Damit können wir leicht Parameter wie Drehzahl übergeben und Code schreiben, der die Pins steuert. Die PWM-Unterstützung fügen wir hinzu, um einstellen zu können, wie schnell die Motoren laufen.

In den ersten Codeblöcken definieren wir die Pins als Ausgänge. Der nächste Block befiehlt Python, PWM auf den beiden Enable-Pins zu aktivieren.

In den folgenden Blöcken erstellen wir unsere eigenen Funktionen. Wir geben ihnen leicht merkbare Namen wie **forward** und **backward** und erlauben auch eine individuelle Motorsteuerung mit **left** und **right**.

Bis zu diesem Punkt passiert nichts, weil Python nicht weiß, was wir vorhaben. Erst ganz am Schluss befehlen wir den Motoren, mit 100 (volle Kraft) 3 Sekunden lang vorwärts zu rollen, dann 3 Sekunden rückwärts mit voller Kraft.

01 Pin-Einstellung

Zuerst importieren wir die benötigen Klassen, dann stellen wir die Pins ein wie bei dem Motortest.

```
import RPi.GPIO as GPIO
from time import sleep

GPIO.setmode(GPIO.BCM)

GPIO.setup(24,GPIO.OUT)
GPIO.setup(23,GPIO.OUT)
GPIO.setup(25,GPIO.OUT)
GPIO.setup(9,GPIO.OUT)
GPIO.setup(10,GPIO.OUT)
GPIO.setup(11,GPIO.OUT)
```

02 PWM-Unterstützung

Um die Motorendrehzahl zu steuern, benutzen wir Pulsweitenmodulation (PWM). Weil der Enable-Pin PWM unterstützt und in beiden Richtungen funktioniert, belegen wir ihn damit.

```
Motor1 = GPIO.PWM(25, 50)
Motor1.start(0)
Motor2 = GPIO.PWM(11, 50)
Motor2.start(0)
```

03 Bewegungsfunktionen

Mit Python kann man Code vereinfachen und wiederverwenden und so lesbarer machen. Damit wir nicht für jeden Pin seine Funktion neu schreiben müssen, gruppieren wir sie in einen **def**-Block.

```
def forward(speed):
  GPIO.output(24,GPIO.HIGH)
  GPIO.output(23,GPIO.LOW)
  GPIO.output(9,GPIO.HIGH)
  GPIO.output(10,GPIO.LOW)
  Motor1.ChangeDutyCycle(speed)
  Motor2.ChangeDutyCycle(speed)
```

04 Drehzahlveränderung

Mit dem Element **(speed)** können wir eine Zahl in die Funktion eingeben, die sie verarbeitet und das Ergebnis – in unserem Fall die Drehzahl des Motors – zurück zum Skript bringt.

```
def backward(speed):
  GPIO.output(24,GPIO.LOW)
  GPIO.output(23,GPIO.HIGH)
  GPIO.output(9,GPIO.LOW)
  GPIO.output(10,GPIO.HIGH)
  Motor1.ChangeDutyCycle(speed)
  Motor2.ChangeDutyCycle(speed)
```

05 Beweg dich

Bis jetzt sieht man nichts von dem Skript, aber die harte Arbeit ist getan. Damit der Roboter in Bewegung kommt, setzen wir unsere neuen Variablen ein.

```
def left(speed):
  GPIO.output(24,GPIO.HIGH)
  GPIO.output(23,GPIO.LOW)
  Motor1.ChangeDutyCycle(speed)

def right(speed):
  GPIO.output(9,GPIO.HIGH)
  GPIO.output(10,GPIO.LOW)
  Motor2.ChangeDutyCycle(speed)

def stop():
  Motor1.ChangeDutyCycle(0)
  Motor2.ChangeDutyCycle(0)
```

PULSWEITEN-MODULATION

Mit der PWM-Technik ändert man die Stromspannung von beispielsweise LEDs und Motoren durch sehr schnelles An- und Ausschalten.

06 Individuelle Bewegungen

Mit **left()** und **right()** können wir jeden Motor einzeln steuern, sodass sich der Roboter auf der Stelle umdreht. Kombiniert mit **sleep** haben wir einen voll beweglichen Roboter!

```
forward(100)
sleep(3)
backward(100)
sleep(3)
forward(50)
sleep(5)
stop()
left(75)
sleep(2)
right(75)
sleep(2)
stop()
```

WIEDERHOLTER CODE

Python verwendet einen Definitionsblock, um Code-Absätze zu wiederholen. Derselbe Code kann mehrmals benutzt und eine Änderung schnell durchgeführt werden.

“Wir möchten den Roboter vollständig und präzise bewegen können.”

Installation von Mikroschaltern

Geben Sie Ihrem Roboter Tastsinn und Reaktionsvermögen, wenn er gegen etwas stößt.

Unser Roboter kann sich jetzt überall hinbewegen, nun ist die einfachste Form der Interaktion dran: Berührung. Der Tastsinn mag bei ihm nicht so ausgeprägt sein wie beim Menschen, aber sein erster Wahrnehmungssinn hilft unserem Roboter bei der Navigation und stattet ihn mit einer grundsätzlichen Form von Intelligenz aus.

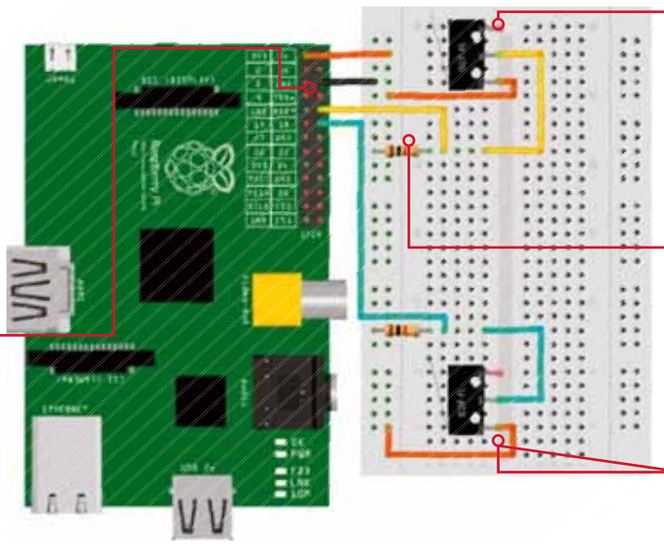
Es gibt verschiedene Methoden, wie ein Roboter „spüren" kann, doch am schnellsten und einfachsten ist es, ihn mit „Antennen" in der Form von Mikroschaltern auszustatten. Trotz ihres Namens sind diese Schalter nicht besonders klein. Sie haben lange hervorstehende „Arme", ideal für die Vorderfront des Roboters. Falls Ihr Schalter keinen oder einen zu kurzen Hebel hat, können Sie ihn mit einen Stück Draht oder einem Trinkhalm verlängern.

Da der Roboter sich meistens vorwärts bewegt, brauchen wir nur vorne Schalter anzubringen. Je mehr Schalter, desto mehr Input bekommt er von seiner Umgebung. Mit einem einfachen Code steuern wir seine Reaktion. Zuerst erstellen wir deshalb den Schaltkreis und testen ihn.

Liste der Bauteile

- Steckbrett
- Jumperkabel
- 2 x 10K-Widerstände
- 2 x CPC-Mikroschalter

NO/NC
Schalter sind oft mit NO oder NC bezeichnet – Normally Open (Schließer, Ruhekontakt) und Normally Closed (Öffner, Arbeitskontakt). Open bedeutet, dass kein Strom fließen kann.



ZUSÄTZLICHE SCHALTER
Um weitere Schalter einzubauen, folgen Sie dem Stromkreis und schließen Sie sie an einen freien Pin an.

PULLDOWN-WIDERSTÄNDE
Es geht um digitale Logik, das heißt der Schalter muss an oder aus sein. Ein Widerstand hilft, indem er den Pin leicht auf Low zieht.

3,3 VOLT SPANNUNG
Der Prozessor des Raspberry Pi ist nicht gegen Überspannung geschützt. Die höchstmögliche Stromeingangsspannung ist 3,3 V.

Der Mikroschalter-Test

Die Schalter sind angeschlossen, jetzt geht es an die Arbeit.

Das Anschließen war leicht, aber wie gesagt toleriert der Raspberry Pi nur eine Eingangsspannung von 3,3 Volt. Deshalb verwenden wir den „3V3"-Pin und nicht (!) den „5V"-Pin.

Der Python-Code, um die Daten zu lesen, ist einfach und logisch. Wir haben einen Schalter pro GPIO-Pin, deshalb braucht Python uns bei einer Abfrage nur zu sagen, in welcher Stellung er ist. Also importieren wir als Erstes unsere üblichen Bibliotheken und setzen die Pins in den BCM-Modus. In **GPIO.setup** befehlen wir Python, Pins 15 und 18 zu Eingangskanälen zu machen.

Durch die Erstellung einer while-True-Schleife entsteht eine Endlosschleife, da die Bedingung immer „True" ist. Während der Schleife speichern wir den Zustand des Inputs in einer Variablen und verwenden dann eine if-Anweisung, um abzufragen, ob er 1 (für gedrückt) oder 0 (für nicht gedrückt) lautet. Wir zeigen auf dem Bildschirm nur, welcher Schalter gedrückt wurde. Dies hilft uns auch zu entscheiden, auf welcher Seite der Mikroschalter platziert wird.

> **„Angezeigt wird, welcher Schalter gedrückt wurde."**

```
import RPi.GPIO as GPIO
from time import sleep

GPIO.setmode(GPIO.BCM)


GPIO.setup(18, GPIO.IN)
GPIO.setup(15, GPIO.IN)


while True:
  inputleft = GPIO.input(18)
  inputright = GPIO.input(15)
  if inputleft:
    print "Left pressed"
  if inputright:
    print "Right pressed"
  sleep(0.1)
```

Fertigstellung des „anstößigen“ Roboters

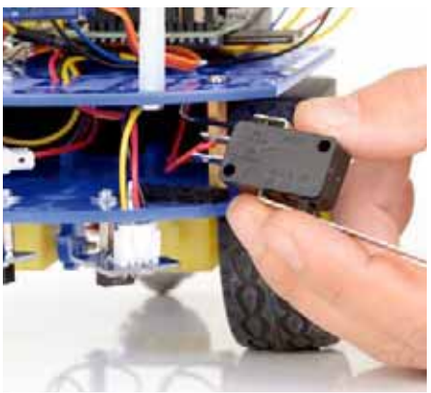
Sind die Schalter angebracht, brauchen wir ein paar Wände, um den Roboter zu testen.

Die Schalter werden vorne am Roboter angebracht, mit doppelseitigem Klebeband oder Blu-Tack. Die Hebel sollen so weit abstehen, dass sie gedrückt werden, wenn der Roboter gegen ein Objekt stößt. Das Skript für die Motorfunktion wird für den Code recycelt, der die Mikroschalter unterstützt. Wenn ein Objekt nun den linken Mikroschalter drückt, erhalten die Motoren den Befehl, für eine Sekunde rückwärts zu fahren und dann anzuhalten. In dieser Zeit hat der Roboter sich hoffentlich wegbewegt, und der linke Motor wird für zwei Sekunden angeschaltet, bevor der Roboter in eine neue Richtung fährt. Das ist ein großer Schritt – wir implementieren AI.

Für noch mehr Beweglichkeit können wir beispielsweise den rechten Motor auf rückwärts schalten und den Roboter auf der Stelle drehen lassen, damit er sich so in eine neue Richtung bewegt.

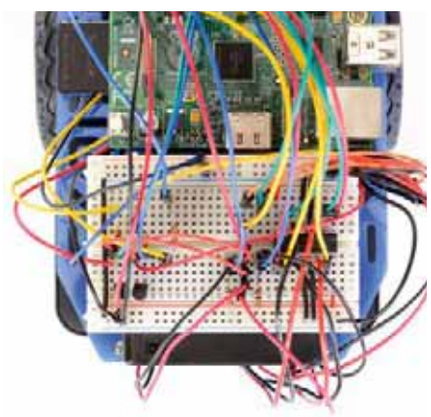
01 Anbringen des Schalters

Platzieren Sie die Mikroschalter möglichst weit vorne am Roboter, doch mit so viel Abstand zueinander, dass klar ist, welche Richtung der Roboter einschlägt, wenn er gegen etwas stößt.



02 Verbindung mit dem Motorschaltkreis

Legen Sie in einigen leeren Lochreihen (vertikalen Spalten) auf dem Steckbrett das GPIO-Jumperkabel an den Pulldown-Widerstand, und schließen Sie den Schalter an wie im Diagramm.



03 Einloggen mit SSH

Wenn der Roboter frei herumläuft, versorgen wir den Raspberry Pi besser mit seiner eigenen Batterie. Über WLAN stellen wir extern eine Verbindung mit SSH her, um das Terminal des Pi zu emulieren.

04 Skripte speichern

Wie bei den Motoren schreiben wir das Skript mit nano. Geben Sie `nano bumpers.py` ein. Speichern Sie verschiedene Skripte, um einzelne Teile zu testen. Sie können die Skripte zum Nachschlagen für längere Programme nutzen.

05 Test an Ort und Stelle

Kopieren Sie das Beispiel-Skript in bumpers.py, speichern Sie per Strg+X und Y. Jetzt können wir es testen, um gegebenenfalls an der Hardware etwas zu ändern. Drücken Sie einen Mikroschalter, während das Skript läuft.

06 Verbesserung am Code

Wenn das Skript startet, drehen sich die Motoren vorwärts. Beim Drücken eines Schalters sollte der Roboter die Richtung ändern und sich im Kreis drehen. Experimentieren Sie mit den Code-Variablen.

Den vollständigen Programmcode gibt es hier als Zip-Archiv: bit.ly/1kj7dZq

Vollständiges Code-Listing für den Roboter

```
import RPi.GPIO as GPIO
from time import sleep

GPIO.setmode(GPIO.BCM)


GPIO.setup(18, GPIO.IN)
GPIO.setup(15, GPIO.IN)

GPIO.setup(24,GPIO.OUT)
GPIO.setup(23,GPIO.OUT)
GPIO.setup(25,GPIO.OUT)
GPIO.setup(9,GPIO.OUT)
GPIO.setup(10,GPIO.OUT)
GPIO.setup(11,GPIO.OUT)

Motor1 = GPIO.PWM(25, 50)
Motor1.start(0)
Motor2 = GPIO.PWM(11, 50)
Motor2.start(0)

def forward(speed):
  GPIO.output(24,GPIO.HIGH)
  GPIO.output(23,GPIO.LOW)
  GPIO.output(9,GPIO.HIGH)
  GPIO.output(10,GPIO.LOW)
  Motor1.ChangeDutyCycle(speed)
  Motor2.ChangeDutyCycle(speed)

def backward(speed):
  GPIO.output(24,GPIO.LOW)
  GPIO.output(23,GPIO.HIGH)
  GPIO.output(9,GPIO.LOW)
  GPIO.output(10,GPIO.HIGH)
  Motor1.ChangeDutyCycle(speed)
  Motor2.ChangeDutyCycle(speed)

def left(speed):
  GPIO.output(24,GPIO.HIGH)
  GPIO.output(23,GPIO.LOW)
  Motor1.ChangeDutyCycle(speed)

def right(speed):
  GPIO.output(9,GPIO.HIGH)
  GPIO.output(10,GPIO.LOW)
  Motor2.ChangeDutyCycle(speed)

def stop():
  Motor1.ChangeDutyCycle(0)
  Motor2.ChangeDutyCycle(0)


while True:
  inputleft = GPIO.input(18)
  inputright = GPIO.input(15)
  if inputleft:
    print “Left pressed”
    backward(100)
    sleep(1)
    stop()
    left(75)
    sleep(2)
  elif inputright:
    print “Right pressed”
    backward(100)
    sleep(1)
    stop()
    right(75)
    sleep(2)
  else:
    forward(75)
  sleep(0.1)
```

SCHROTTEN NICHT SIE DEN PI!

Überprüfen Sie unbedingt die Spezifikationen aller Sensoren. Sie müssen mit einer Netzspannung von 3,3 Volt kompatibel sein.

DIGITALE SCHALTER

Ein Schalter ist ein perfektes digitales Signal, denn er kennt nur zwei Stellungen: an oder aus.

Sensoren für Linienfolger

Leitlinien aus Abdeckband oder farbigem Papier für den Roboter.

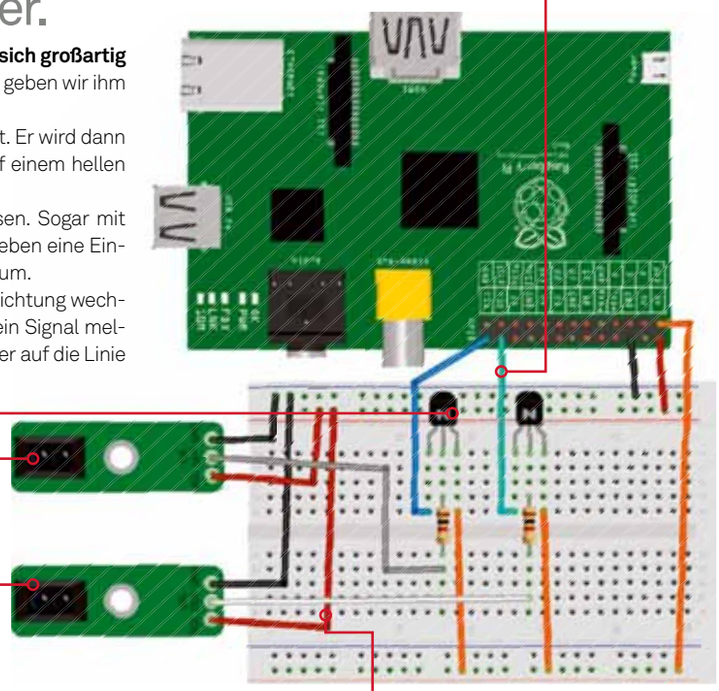
Unser Roboter kann bislang selbst seinen Weg wählen, was an sich großartig ist. Aber er kann sich so auch in Schwierigkeiten bringen. Deshalb geben wir ihm seinen Weg besser vor.

Dazu wird ein Zeilensensor an der Unterseite des Roboters angebracht. Er wird dann über ein Abdeckband auf dem dunklen Boden (oder dunkle Streifen auf einem hellen Boden) gesteuert. Das kann in vielerlei Hinsicht sinnvoll sein.

Wir können den Roboter brav einer Linie auf dem Boden folgen lassen. Sogar mit Kurven sollte er die vorgegebene Strecke navigieren können. Oder wir kleben eine Eingrenzung um den Roboter und beschränken ihn auf einen bestimmten Raum.

Die besten Linienfolger sind zweirädrige Roboter, weil sie schnell die Richtung wechseln können. Das Prinzip funktioniert so, dass immer, wenn ein Sensor ein Signal meldet, der entsprechender Motor gestoppt wird, wodurch der Roboter wieder auf die Linie gelenkt wird.

SICHERHEIT ZUERST
Mit dem Transistor liegt eine viel sicherere Spannung an den GPIO-Pins an, wenn 5-Volt-Geräte verwendet werden.



NIEDRIGERE SPANNUNG
Die Transistoren brauchen nur eine geringe Betriebsspannung. Ein Widerstand glättet den Ausgangsstrom der Sensoren.

ZEILENSENSOREN
Egal, welche Form oder Größe, Sensoren haben meist die gleichen Pins. Am wichtigsten ist der OUT-Pin.

Liste der Bauteile

- Steckbrett
- Jumperkabel
- 2 x 2N3904-Transistoren
- 2 x 1K-Widerstände
- 2 x Zeilensensoren

SICHERE SPANNUNG
Transistoren funktionieren wie Schalter, sie schalten Strom an oder aus. Es ist viel sicherer, die Spannung am GPIO von 3,3 Volt damit zu regeln.

STROM
5 Volt ist die übliche Betriebsspannung bei Sensoren. Ein Transistor reduziert die 5 Volt auf ein sicheres Level für den Raspberry Pi.

Der Zeilensensor-Test

Die Zeilensensoren sind verschaltet, der Raspberry Pi ist an – alles ist bereit für den Test. Dieses Python-Skript ähnelt sehr dem Mikroschalter-Testcode. Wir prüfen den Status des GPIO-Pins – steht er auf HIGH (1 oder an) oder auf LOW (0 oder aus)?

Sensoren funktionieren unterschiedlich, deshalb brauchen wir Hilfe, um den Output zu verstehen. Die angezeigten Daten des Sensors vermitteln uns, wie er auf dunkle und helle Flächen reagiert. Dementsprechend schreiben wir den Code.

01 Projektstart

Loggen Sie sich mit SSH ein oder öffnen Sie auf dem Raspberry Pi ein Terminal. Erstellen Sie mit `nano linefollow.py` ein Testskript für den Linienfolger-Roboter.

02 Sensorentest

Kopieren Sie das Testskript in die Datei. Da jeder Sensor etwas anders ist, müssen wir den Code eventuell anpassen. Machen Sie den Test und ändern Sie das Skript entsprechend.

03 Ausgabeanweisungen

Speichern Sie die Datei wie zuvor. Unser Beispielcode enthält Ausgabeanweisungen, damit angezeigt wird, ob der Sensor Unterschiede zwischen hellen und dunklen Flächen wahrnimmt.

04 Wir haben Daten!

Ist alles korrekt verschaltet, wird auf dem Bildschirm angezeigt, ob der Roboter dunkel und hell erkennt. Legen Sie ein Papier unter die Sensoren, um das auszuprobieren.

```
import RPi.GPIO as GPIO
from time import sleep

GPIO.setmode(GPIO.BCM)

Input1 = 7
Input2 = 8

GPIO.setup(Input1,GPIO.IN)
GPIO.setup(Input2,GPIO.IN)


while True:
  Sensor1 = GPIO.input(Input1)
  Sensor2 = GPIO.input(Input2)

  if Sensor1 == GPIO.HIGH:
    print "Sensor 1 is on White"
  else:
    print "Sensor 1 is on Black"

  if Sensor2 == GPIO.HIGH:
    print "Sensor 2 is on White"
  else:
    print "Sensor 2 is on Black"

  print "------"
  sleep(1)


GPIO.cleanup()
```

Fertigstellung des Linienfolgers

Er kann sehen! Die „Augen“ kommen zum Einsatz.

Inzwischen können Sie Motoren steuern, und wir konzentrieren uns jetzt auf die Sensoren. Die meisten Linienfolger verwenden ähnliche Konventionen wie Mikroschalter. HIGH bedeutet, der Sensor ist über einer schwarzen Fläche, LOW (oder out) bedeutet, er ist über einer weißen Fläche. Folgt der Roboter einer Linie aus hellem Abdeckband, soll der Motor stoppen, sobald der Sensor die Linie berührt. Dann kann der Motor auf der anderen Seite die Fahrtrichtung korrigieren. Da der Code so einfach ist, können Sie ihn leicht an Ihre Situation anpassen.

01 Anbau der Sensoren

Bringen Sie die Sensoren mithilfe der sechskantigen Hilfsstäbe in etwa 10 mm Höhe an, um unebene Flächen auszugleichen. Die meisten Sensoren sind bei diesem Abstand empfindlich genug. Wenn nicht, kann man mit einem Potentiometer die Empfindlichkeit verstärken.

BIG BUSINESS

Manche Fertigungsanlage setzt Linienfolger-Roboter ein, die ganz ähnlich funktionieren wie unserer, nur sind diese viel größer.

02 Anschluss am Steckbrett

Sie sollten genug Platz auf dem Steckbrett haben, doch nutzen Sie die Lochreihen gut aus. Richten Sie Bereiche für verschiedene Sensortypen ein, merken Sie sich Ihre Anordnung. Das erleichtert Ihnen später das Debuggen der Hardware.

03 Sensorschaltkreis

Befestigen Sie die zwei Transistoren und Widerstände auf dem Board. Jeder Pin kommt in seine eigene Spalte. Verbinden Sie die Jumperkabel von Sensoren und Stromzufuhr, danach die zu den GPIO-Pins.

04 Anschalten und Login

Schließen Sie die Motorbatterien an, geben Sie dem Raspberry Pi Strom. Loggen Sie sich mit SSH in Ihren Computer ein, um den Code für den motorkontrollierten Zeilensensor zu schreiben.

05 Das Skript

Der Code für den Linienfolger ähnelt unseren anderen Skripten. Mit „while True“ wiederholt sich das Skript, bis wir es stoppen, und es enthält enthält einige Ausgabeanweisungen zum Debuggen.

06 Skript-Test

Wenn alles stimmt, fährt Ihr Roboter jetzt auf der vorgegebenen Linie. An dem Skript kann vieles verbessert werden, damit der Roboter der Linie folgt. Das Skript fügt sich auch problemlos in den schon existierenden Code ein.

Den vollständigen Programmcode gibt es hier als Zip-Archiv: bit.ly/1nNQyOc

Vollständiges Code-Listing für den Roboter

```
import RPi.GPIO as GPIO
from time import sleep

GPIO.setmode(GPIO.BCM)


GPIO.setup(7, GPIO.IN)
GPIO.setup(8, GPIO.IN)

GPIO.setup(24,GPIO.OUT)
GPIO.setup(23,GPIO.OUT)
GPIO.setup(25,GPIO.OUT)
GPIO.setup(9,GPIO.OUT)
GPIO.setup(10,GPIO.OUT)
GPIO.setup(11,GPIO.OUT)

Motor1 = GPIO.PWM(25, 50)
Motor1.start(0)
Motor2 = GPIO.PWM(11, 50)
Motor2.start(0)

def forward(speed):
  GPIO.output(24,GPIO.HIGH)
  GPIO.output(23,GPIO.LOW)
  GPIO.output(9,GPIO.HIGH)
  GPIO.output(10,GPIO.LOW)
  Motor1.ChangeDutyCycle(speed)
  Motor2.ChangeDutyCycle(speed)

def backward(speed):
  GPIO.output(24,GPIO.LOW)
  GPIO.output(23,GPIO.HIGH)
  GPIO.output(9,GPIO.LOW)
  GPIO.output(10,GPIO.HIGH)
  Motor1.ChangeDutyCycle(speed)
  Motor2.ChangeDutyCycle(speed)

def left(speed):
  GPIO.output(24,GPIO.HIGH)
  GPIO.output(23,GPIO.LOW)
  Motor1.ChangeDutyCycle(speed)

def right(speed):
  GPIO.output(9,GPIO.HIGH)
  GPIO.output(10,GPIO.LOW)
  Motor2.ChangeDutyCycle(speed)

def stop():
  Motor1.ChangeDutyCycle(0)
  Motor2.ChangeDutyCycle(0)


while True:
  sensor1 = GPIO.input(7)
  sensor2 = GPIO.input(8)
  if sensor1 == GPIO.LOW:
    print “Sensor 1 is on white”
    stop()
  else:
    left(60)
  if sensor2 == GPIO.LOW:
    print “Sensor 2 is on white”
    stop()
  else:
    right(60)
  sleep(0.05)
```

SUDO PYTHON?

Stellen Sie sudo voran, um die Benutzerrechte eines Programms auf Superuser einzustellen. Vergessen Sie nicht, GPIO-Pins müssen mit Python gesteuert werden.

Ultraschallsensorik

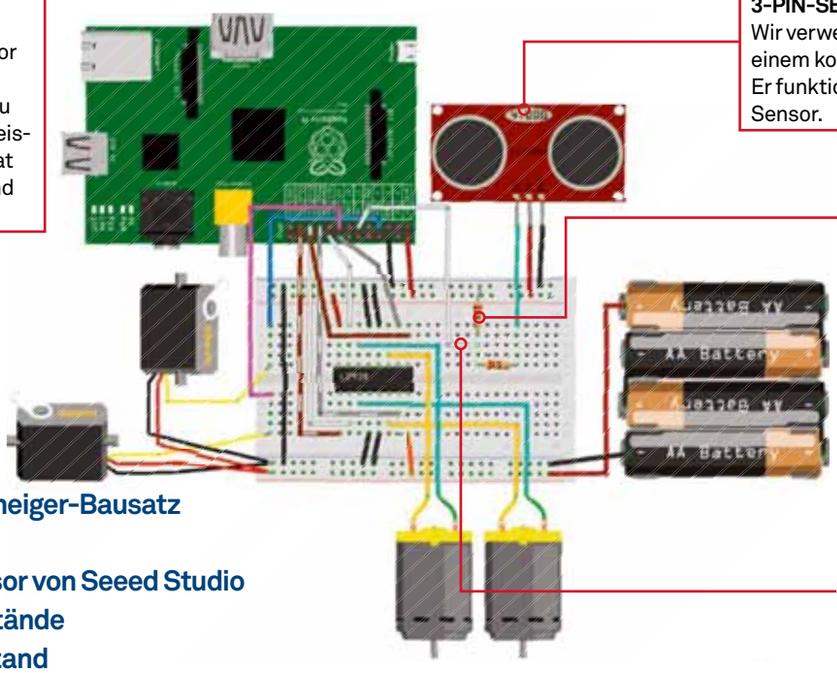
Der Roboter hat Tastsinn und kann unter seine Räder blicken. Wie wäre es nun mit einer neuen Art des Sehens?

Damit die Sache noch etwas komplizierter wird, montieren wir einen Ultraschallsensor auf einer schwenk- und kippbaren Halterung. Ultraschallsensoren werden eingesetzt, um Distanzen zu erfassen. Dazu wird ein Ultraschallsignal entsandt und gezählt, wie lange es dauert, bis es von einem Objekt zurück zum Empfänger kommt. Ultraschallbasierte Einparkhilfen für Autos funktionieren so, dass sie hörbare Signale geben, je nachdem, wie weit ein Objekt noch entfernt ist. Mit einem Ultraschallsensor kann Ihr Roboter etwas unternehmen, wenn er etwa auf eine Wand zusteuert. Er bekommt Zeit, die Lage einzuschätzen und die Richtung zu wechseln.

Es gibt zwei Arten von Ultraschallsensoren, die sich in der Anzahl der Pins unterscheiden (3- oder 4-Pin-Sensoren). Sie funktionieren sehr ähnlich. In diesem Artikel arbeiten wir jedoch vor allem mit dem preisgünstigen 3-Pin-Modell von Dawn Robotics.

Wir brauchen nur einen GPIO-Pin, den wir zunächst als Ausgang festlegen. Dann senden wir einen Puls mit 10 ms, der Sensor beginnt zu zählen. Als Nächstes schalten wir den Pin auf Eingang und warten, bis er auf HIGH springt. In diesem Moment wird die Zeitmessung beendet und die Vorgangsdauer berechnet. Als Letztes konvertieren wir diese Zeit in eine lesbare Messeinheit, in unserem Fall Zentimeter.

4-PIN-SENSOR
Der gebräuchlichste 4-Pin-Sensor HC-SR04 misst Abstände von bis zu 4 Metern. Neben Leistung und Erdung hat er einen Trigger- und einen Echo-Pin.



3-PIN-SENSOR
Wir verwenden einen 3-Pin-Sensor mit einem kombinierten Echo/Trigger-Pin. Er funktioniert genauso wie der 4-Pin-Sensor.

SPANNUNGSTEILER
Wir haben es wieder mit 5-Volt-Sensoren zu tun, deshalb senken wir die Spannung auf 3,3 Volt, um den Raspberry Pi nicht zu gefährden.

NUR EIN GPIO-PIN
Ein GPIO-Pin wechselt sehr schnell zwischen Ausgang und Eingang, er sendet und empfängt Signale. Das spart Pins, auch bei den 4-Pin-Sensoren.

Liste der Bauteile

- Dagu Schwenkneiger-Bausatz
- Jumperkabel
- Ultraschallsensor von Seeed Studio
- 2 x 2K2-Widerstände
- 1 x 10K-Widerstand

Aufbau des Schwenkneigers

Daten aus verschiedenen Blickwinkeln gefällig? Kein Problem!

Schwenkneiger können mit jeder Art von Sensor kombiniert werden. Durch sie erhält der Roboter die Fähigkeit, „den Kopf zu drehen“ und Daten aus seinem ganzen Umfeld zu sammeln, ohne den Körper zu bewegen. Der Schwenkneiger wird von zwei Spezialmotoren, sogenannten Servos, gesteuert. Servomotoren ermöglichen sehr genaue Bewegungen in ihrem Operationsradius, typischerweise zwischen 0 und 180 Grad. Sie reagieren auf genau getimte Impulse. Der Zeitraum zwischen den Impulsen vermittelt dem Servo seinen Drehwinkel.

Der Raspberry Pi hat mit Echtzeit seine Probleme. Manchmal kann er keinen gleichmäßigen Impuls halten und vergisst, was er gerade tut, geht weg und checkt E-Mails. Deshalb programmierte Richard Hirst einen Kernel für Linux mit dem Namen ServoBlaster, der die Zeit perfekt misst, egal was sonst gerade passiert. Der Kernel übernimmt die Kontrolle über einige Timing-Register, um eine akkurate Uhrzeit zu haben. Sie müssen nur den gewünschten Drehwinkel an **/dev/servoblaster** senden, und der Servo erwacht zum Leben!

Den vollständigen Programmcode gibt es hier als Zip-Archiv: bit.ly/1l7lK5u

Vollständiges Code-Listing für den Ultraschall

```
import RPi.GPIO as GPIO
from time import sleep
from time import time
import os

GPIO.setmode(GPIO.BCM)

GPIO.setup(24,GPIO.OUT)
GPIO.setup(23,GPIO.OUT)
GPIO.setup(25,GPIO.OUT)
GPIO.setup(9,GPIO.OUT)
GPIO.setup(10,GPIO.OUT)
GPIO.setup(11,GPIO.OUT)

Motor1 = GPIO.PWM(25, 50)
Motor1.start(0)
Motor2 = GPIO.PWM(11, 50)
Motor2.start(0)

Echo = 17
Pan = 22
Tilt = 4

def forward(speed):
  GPIO.output(24,GPIO.HIGH)
  GPIO.output(23,GPIO.LOW)
  GPIO.output(9,GPIO.HIGH)
  GPIO.output(10,GPIO.LOW)
  Motor1.ChangeDutyCycle(speed)
  Motor2.ChangeDutyCycle(speed)

def backward(speed):
  GPIO.output(24,GPIO.LOW)
  GPIO.output(23,GPIO.HIGH)
  GPIO.output(9,GPIO.LOW)
  GPIO.output(10,GPIO.HIGH)
  Motor1.ChangeDutyCycle(speed)
  Motor2.ChangeDutyCycle(speed)

def left(speed):
  GPIO.output(24,GPIO.HIGH)
  GPIO.output(23,GPIO.LOW)
  Motor1.ChangeDutyCycle(speed)

def right(speed):
  GPIO.output(9,GPIO.HIGH)
  GPIO.output(10,GPIO.LOW)
  Motor2.ChangeDutyCycle(speed)

def stop():
  Motor1.ChangeDutyCycle(0)
  Motor2.ChangeDutyCycle(0)


def get_range():
  GPIO.setup(Echo,GPIO.OUT)
  GPIO.output(Echo, 0)
  sleep(0.1)
  GPIO.output(Echo,1)
  sleep(0.00001)
  GPIO.output(Echo,0)

  GPIO.setup(Echo,GPIO.IN)
  while GPIO.input(Echo) == 0:
    pass
  start = time()
  while GPIO.input(Echo) == 1:
    pass
  stop = time()
  elapsed = stop - start
  distance = elapsed * 17000
  return distance


while True:
  distance = get_range()
  if distance < 30:
    print "Distance %.1f " %
distance
    stop()
    string = "echo 0=10 > /dev/
servoblaster"
    os.system(string)
    sleep(1)
    disleft = get_range()
    print "Left %.1f " % disleft

    string = "echo 0=360 > /dev/
servoblaster"
    os.system(string)
    sleep(1)
    disright = get_range()
    print "Right %.1f " % disright

    if disleft < disright:
      print "Turn right"
      left(100)
      sleep(2)
    else:
      print "Turn left"
      right(100)
      sleep(2)

    os.system("echo 0=160 > /dev/
servoblaster")

  else:
    forward(80)
    print "Distance %.1f " %
distance

  sleep(0.5)

GPIO.cleanup()
```

Installation des Schwenkneigers

Ein kniffliger Job, aber es lohnt sich.

Den Stromkreislauf haben wir geschafft, jetzt installieren wir das Ganze. Zuerst downloaden wir den Kernel. Tippen Sie diesen Befehl in das Terminal des Raspberry Pi:

```
wget https://github.com/Boeeerb/LinuxUser/raw/master/servod
```

Machen Sie das Skript lauffähig:

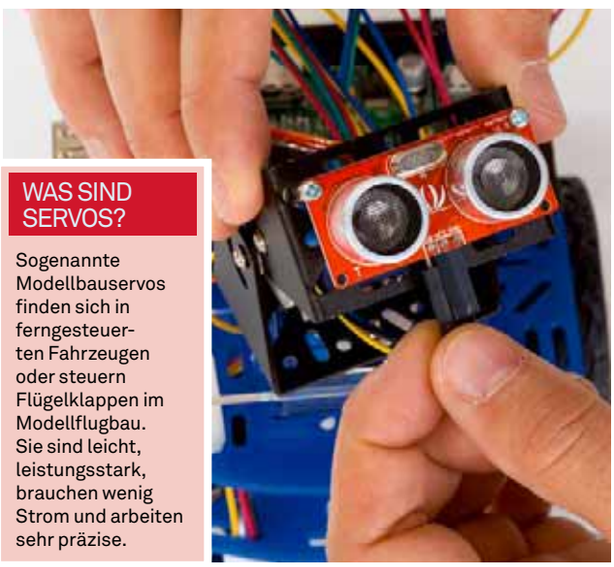
```
chmod +x servod
```

Lassen Sie dann den Kernel laufen. Immer, wenn Sie Ihren Pi rebooten oder anschalten, brauchen Sie nur diese Zeile zu tippen:

```
sudo ./servod
```

Der vorkompilierte Kernel ist so konfiguriert, dass er Pin 4 und 22 als Servos benutzt. Die Servos sollten separat angetrieben werden, denn es sind im Grunde nur Motoren mit ein paar Schaltkreisen.

Unser Code verbindet Radmotoren, Servos und Ultraschallsensoren. Im Endresultat soll der Roboter sich vorwärtsbewegen, bis er sich einem Objekt auf weniger als 30 cm Entfernung nähert. Er soll anhalten, den Schwenkneiger nach links drehen, den Abstand messen, dann den Ultraschallsensor auf dem Schwenkneiger nach rechts drehen und die Richtung mit dem größten Abstand zum nächsten Objekt einschlagen.



WAS SIND SERVOS?

Sogenannte Modellbauservos finden sich in ferngesteuerten Fahrzeugen oder steuern Flügelklappen im Modellflugbau. Sie sind leicht, leistungsstark, brauchen wenig Strom und arbeiten sehr präzise.

01 180-Grad-Blick

Der Schwenkneiger ermöglich einen 180-Grad-Blick von links nach rechts, von oben nach unten – optimal für Ultraschall oder eine Kamera. Mit den Servos lässt es sich perfekt steuern.

02 Anschluss der Servos

Die Servos sind Motoren, deshalb sollten sie nicht über den Raspberry Pi angetrieben werden. Die erforderliche Spannung beträgt bei den meisten bis zu 6 Volt. Sie können mit denselben Batterien wie die Motoren versorgt werden.

03 Der ServoBlaster

Für die Servo-Steuerung muss servod (ServoBlaster) laufen. Laden Sie das Programm herunter. Mit `chmod +x servod` wird es lauffähig, mit `sudo ./servod` führen Sie es aus.

04 Skripterstellung

Jetzt schreiben wir das Testskript. Sie können sich unser Beispielskript unter dem Link oben herunterladen, oder noch besser, Sie tippen es übungshalber ab.

05 Und er läuft...

Wenn Sie das Skript ausführen, sollte sich der Monitor mit Entfernungsangaben füllen. Daran sehen wir, was passiert und welche Richtungen der Roboter einschlägt. Wenn er in einer Ecke steckenbleibt, versuchen Sie, das Skript zu debuggen.

06 Fehlerbeseitigung

Verhält sich der Roboter nicht wie erwartet oder geht der Servo in die falsche Richtung, vertauschen Sie seine Daten-Pins. Überprüfen Sie den Code und versuchen Sie es erneut.

Analoge Sensoren

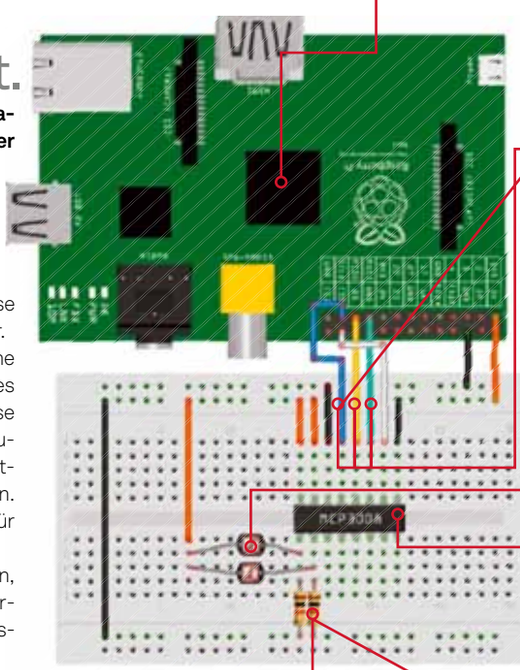
Eröffnen Sie Ihrem Roboter eine neue Welt.

Beim Anbau der Mikroschalter und der Ultraschallsensoren haben wir gesehen, dass der Raspberry Pi in der Lage ist, Input zu verarbeiten und entsprechend auf die Welt zu reagieren. Input kann ein Roboter auf mehreren Wegen bekommen. Am häufigsten sind digitale Sensoren wie Knöpfe und Schalter, aber es gibt auch analoge Sensoren, die Temperatur oder Helligkeit messen. Diese Sensoren geben ihre Daten in Form von Spannung weiter.

Der Raspberry Pi kann ein analoges Signal nicht ohne Weiteres lesen, deshalb braucht er Hilfe in Form eines Mikrochips, dem MCP3008. Der Chip wird üblicherweise als ADC (Analog-Dialog-Wandler) bezeichnet. Er kommuniziert mit dem Raspberry Pi über die serielle Schnittstelle und kann acht analoge Inputs gleichzeitig lesen. Den Spannungswert gibt er als Zahl wieder: 0 steht für die niedrigste, 1023 für die höchste Spannung.

Mit Analogsensoren können wir einen Roboter bauen, der hellem Licht folgen (oder es vermeiden) kann – perfekt, wenn man sich eine Topfpflanze wünscht, die tagsüber der Sonne folgt.

3,3 VOLT SPANNUNG
Schließen Sie den Chip am „3V3“-Pin und nicht am „5V“-Pin des Raspberry Pi an, sonst gefährden Sie dessen Prozessor.



DATENKABEL
Der MCP3008 kommuniziert über das serielle Protokoll SPI (Serial Peripheral Interface). Es können mehrere gleichzeitig verwendet werden.

FOTOWIDERSTÄNDE
Lichtabhängige Widerstände (LDRs oder Fotowiderstände) ändern ihre Spannung je nach der Lichtmenge, die auf sie fällt.

MCP3008
Das Herzstück des Analog-Digital-Wandlers.

PULLDOWN-WIDERSTÄNDE
Um stabile Anzeigen zu erhalten, brauchen wir einen Orientierungswert für die Spannung – deshalb der Pulldown-Widerstand.

Liste der Bauteile

- 1 x MCP3008
- 2 x lichtabhängige Widerstände (LDRs)
- 2 x 10K-Widerstände
- Jumperkabel

Testen, testen, testen...

Gute Informatiker prüfen immer zuerst, ob es funktioniert.

Der A/D-Wandler ist verschaltet, jetzt muss er nur noch funktionieren. Dazu lassen wir uns zuerst mit Python die Daten auf dem Monitor anzeigen. So bekommen wir eine Vorstellung davon, was passiert, wenn die Sensoren sich in hellem Licht befinden, und wie die Zahlenwerte abweichen, wenn sie im Dunkeln sind.

Bevor wir den MCP3008 anschließen, geben wir die seriellen Treiber frei und installieren die Python-Bibliothek spidev. Öffnen Sie ein Terminal oder verbinden Sie sich mit Ihrem Raspberry Pi. Tippen Sie dann die folgenden Befehle:

```
sudo nano /etc/modprobe.d/raspi-blacklist.conf
```

Setzen Sie ein # an den Anfang jeder Zeile in der Datei, und geben Sie dann ein:

```
sudo apt-get install python-pip python-dev
sudo pip install spidev
sudo reboot
```

Und nun können wir endlich die Werte der analogen Sensoren lesen!

In den ersten zwei Code-Zeilen wird Python mitgeteilt, welche Bibliotheken benötigt werden. Danach soll Python eine neue Instanz erstellen – dazu muss es wissen, auf welchem Kanal unser MCP3008-Chip liegt. Das steht in den nächsten beiden Zeilen.

Nun definieren wir eine Funktion, die die Kommunikation übernimmt und sie an das Skript zurückgibt, damit wir auf den Wert, der mit „get_value“ aufgerufen wird, reagieren können.

Die Kanäle auf dem Chip sind von links nach rechts mit 0 bis 7 belegt. Das kombinieren wir mit der get_value-Funktion, um unseren Wert zu erhalten.

```
import spidev
import time

spi = spidev.SpiDev()
spi.open(0,0)


def get_value(channel):
  if ((channel > 7) or (channel < 0)):
    return -1

  r = spi.xfer2([1,(8+channel)<<4,0])

  ret = ((r[1]&3) << 8) + (r[2] >> 2)
  return ret


while True:
  print “Chan 0: “ + str(get_value(0))
  print “Chan 1: “ + str(get_value(1))
  time.sleep(0.3)
```

Den vollständigen Programmcode gibt es hier als Zip-Archiv: bit.ly/1rY12Kl

Lichtmessen mit dem Raspberry Pi

Alles ist verdrahtet. Jetzt jagen wir dem Licht hinterher!

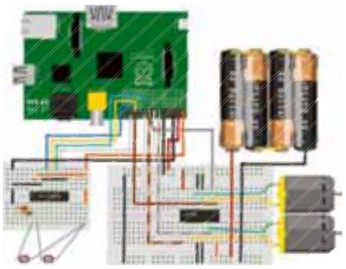
Über Ihren Monitor laufen jetzt hoffentlich Zahlenreihen, und Sie haben getestet, wie sich die Werte ändern, wenn ein Sensor abgedeckt oder mit einer Taschenlampe beleuchtet wird. Wir bringen nun die LDRs vorn am Roboter an, damit er die Lichtstärke misst. Der Roboter soll langsam vorwärtsrollen. Die LDRs prüfen konstant die Lichtstärke. Wenn ein Sensorwert über 600 geht (etwa durch den Strahl einer Taschenlampe), bewegt sich das gegenüberliegende Rad schneller, sodass der Roboter sich auf das Licht zubewegt. Lichtverhältnisse sind immer etwas anders. Führen Sie deshalb das Textskript öfters durch, damit Sie sehen, welche Anzeigen von den LDRs zu erwarten sind. Diese fluktuieren je nach Lichtstärke in der Umgebung.

EXPERIMENTE

Die Lichtstärke fällt unter einen bestimmten Wert: der Roboter stoppt. Ein Licht strahlt auf einen LDR-Sensor: der Roboter dreht sich 5 Sekunden lang. Versuchen Sie sich mal daran.

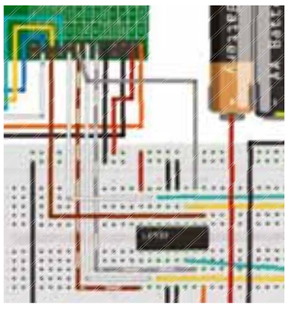
01 Anbau der LDRs

Am besten platziert man die LDRs mit Abstand und richtet sie nach außen, um die Lichtverhältnisse vollständig zu erfassen. Wird die Verschaltung auf dem Steckbrett zu kompliziert, fügen Sie ein weiteres Steckbrett hinzu und trennen Sie die zwei ICs.



02 Motortreiber-Pins ändern

Da wir nun den SPI-Bus für die Kommunikation mit dem MCP3008 verwenden, müssen wir die Motortreiber-Pins in eine neue Reihe von GPIO-Pins umstecken. Wir wechseln Pins 8, 9 und 10 zu 22, 27/21 und 17.



03 Doppelt prüfen

Bei den verschiedenen Spannungen – 3,3 Volt für den MCP3008, 5 Volt für den L293D und die Batterien – überprüfen wir besser zweimal, ob die Bauteile korrekt verschaltet sind. Ist alles okay, schalten Sie den Strom an und loggen Sie sich in den Pi ein.

04 Skripterstellung

Wir schreiben das Python-Skript mit nano. Tippen Sie `nano analog.py` für die Datei, kopieren Sie den Code. Beenden Sie nano mit Strg+X, und speichern Sie die Datei mit Y und Eingabetaste. Inzwischen geht das ja wie aus dem Effeff.

05 Skript ausführen

Geben Sie `sudo python analog.py` ein, um das Skript auszuführen. Der Roboter sollte nun der hellsten Lichtquelle folgen. Wenn nicht, überprüfen Sie Ihren Code und die Verbindungen oder debuggen Sie mit dem Testcode.

06 Etwas stimmt nicht

Wenn Sie keinen Fehler finden, muss vielleicht der Sensorwert geändert werden. Führen Sie das Testskript noch einmal aus, um einen Wert zu erhalten, mit dem Sie die aktuellen 600 ersetzen. Je nach Tageszeit kann es gut sein, dass Sie Variablen anpassen müssen.

Vollständiges Code-Listing für die Analogsensoren

```
import spidev
import time

spi = spidev.SpiDev()
spi.open(0,0)

GPIO.setmode(GPIO.BCM)

GPIO.setup(27,GPIO.OUT)
GPIO.setup(17,GPIO.OUT)
GPIO.setup(22,GPIO.OUT)
GPIO.setup(9,GPIO.OUT)
GPIO.setup(10,GPIO.OUT)
GPIO.setup(11,GPIO.OUT)

Motor1 = GPIO.PWM(22, 50)
Motor1.start(0)
Motor2 = GPIO.PWM(11, 50)
Motor2.start(0)


def forward(speed):
  GPIO.output(27,GPIO.HIGH)
  GPIO.output(17,GPIO.LOW)
  GPIO.output(9,GPIO.HIGH)
  GPIO.output(10,GPIO.LOW)
  Motor1.
ChangeDutyCycle(speed)
  Motor2.
ChangeDutyCycle(speed)

def backward(speed):
  GPIO.output(27,GPIO.LOW)
  GPIO.output(17,GPIO.HIGH)
  GPIO.output(9,GPIO.LOW)
  GPIO.output(10,GPIO.HIGH)
  Motor1.
ChangeDutyCycle(speed)
  Motor2.
ChangeDutyCycle(speed)

def left(speed):
  GPIO.output(27,GPIO.HIGH)
  GPIO.output(17,GPIO.LOW)
  Motor1.
ChangeDutyCycle(speed)

def right(speed):
  GPIO.output(9,GPIO.HIGH)
  GPIO.output(10,GPIO.LOW)
  Motor2.
ChangeDutyCycle(speed)

def stop():
  Motor1.ChangeDutyCycle(0)
  Motor2.ChangeDutyCycle(0)


def get_value(channel):
  if ((channel > 7) or
(channel < 0)):
    return -1

  r = spi.
xfer2([1,(8+channel)<<4,0])

  ret = ((r[1]&3) << 8) +
(r[2] >> 2)
  return ret


while True:
  ldr_left = get_value(0)
  ldr_right = get_value(1)


  if ldr_left > 600:
    print “Turn right”
    right(100)
  elif ldr_right > 600:
    print “Turn left”
    left(100)
  else:
    forward(75)

  time.sleep(0.25)
```

MCP3008/ MCP3004

Es gibt auch einen kleineren A/D-Wandler-Chip, den MCP3004. Er hat nur 4 Analogkanäle und nicht 8 wie der MCP3008.

Was kommt als Nächstes?

Das Projekt Roboterbau ist abgeschlossen – was kann der Roboter noch?

Es gibt so viele Möglichkeiten – das ist eines der tollen Dinge an der Robotik. Sie brauchen Zeit und Fantasie, ansonsten sind Ihnen keine Grenzen gesetzt.

Erweitern Sie die Hardware Ihres Roboters. Wenn Sie mehr wissen und sich mehr zutrauen, können Sie weitere Sensoren anbauen, damit der Roboter mehr über die Welt erfährt. Gase, Licht, Ton – für praktisch jeden Reiz gibt es einen entsprechenden Sensor, mit dem Sie Ihren Roboter ausstatten können. Auf einer größeren Plattform können Sie auch Arme am Roboter anbringen, damit er die Welt nicht nur über die Sinne erfährt. Mit einem Arm kann er Teile aus der Welt aufheben und umherbewegen.

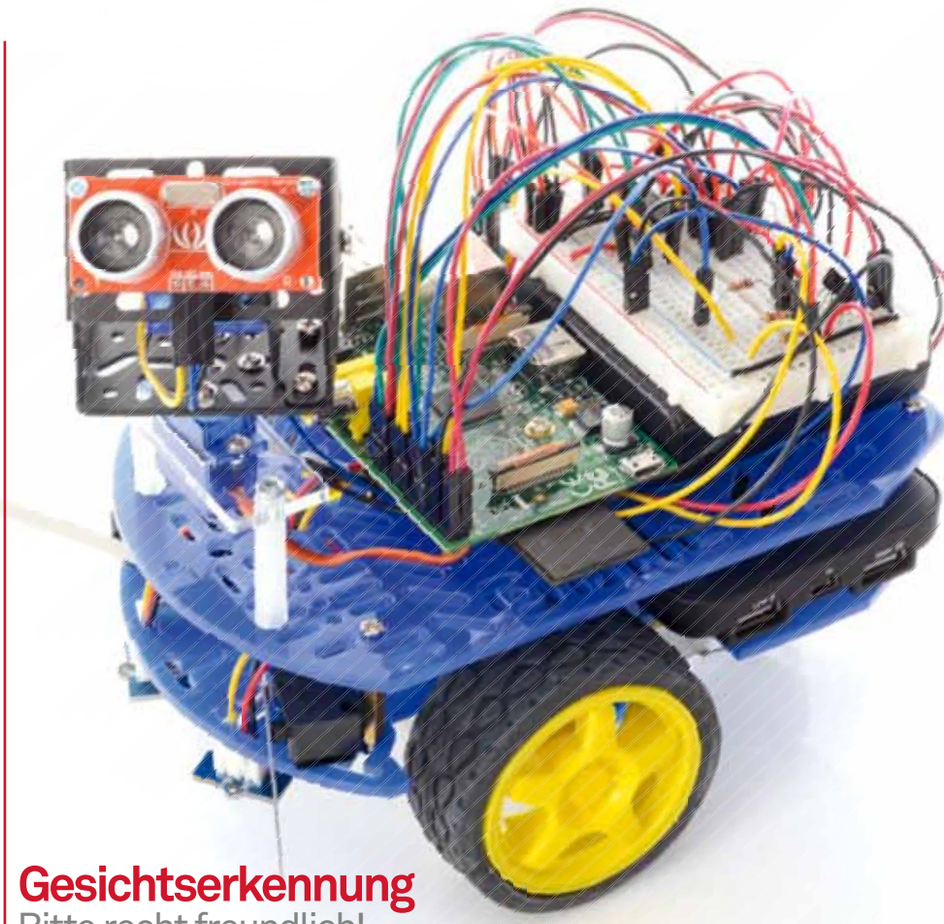
Sie können den Roboter auch erweitern, indem Sie es ihm ermöglichen, in Kommunikation mit den Menschen in seiner Umgebung zu treten. Lautsprecher sind eine Möglichkeit, aber auch mit blinkenden LEDs oder Leuchtstreifen aus Elektrolumineszenz-Kabeln kann der Roboter seinen inneren Zustand vermitteln. Je kreativer, desto besser: Robotik ist nicht nur eine technische Tätigkeit, sondern auch eine künstlerische.

Die Rechenleistung des Raspberry Pi erlaubt es Ihnen, auch die Software Ihres Roboters zu erweitern und so seine Intelligenz zu erhöhen. Wahrscheinlich am populärsten ist eine Webcam oder die Pi-Kamera für Computer Vision. Mit OpenCV, einer sehr umfangreichen Open-Source-Bibliothek für Computer Vision, können Sie schnell loslegen. Damit kann Ihr Roboter Gesichter erkennen, nach interessanten Objekten in seiner Umgebung Ausschau halten und sogar Orte wiedererkennen, an denen er schon einmal war.

Und denken Sie nicht, dass Sie sich auf nur einen Roboter beschränken müssen. Schwarmrobotik ist ein spannendes Feld der Robotik, das von Intelligenz der Insektenschwärme in der Natur inspiriert ist. Es ist faszinierend, welches komplexe Verhalten sich in den Interaktionen von ein paar einfachen Robotern entwickelt. Die Roboter können über WLAN oder über einen zentralen Computer kommunizieren. Oder Sie statten die Roboter mit Infrarot-LEDs und Empfängern aus, für die „lokalere" Kommunikation mit ihren Nachbarn.

Egal, welche Richtung Sie mit Ihrem Raspberry-Pi-Roboter einschlagen, lassen Sie die Raspberry-Pi-Community daran teilhaben! In den Open-Source-Communitys um den Raspberry Pi gibt es viele nette und kluge Leute, die oft selbst Roboter bauen. Orte wie das Raspberry-Pi-Forum sind eine großartige Ressource, wo Sie Unterstützung finden, wenn Sie sich Ihren Traumroboter bauen.

Alan Broun, Geschäftsführer von Dawn Robotics Ltd



Gesichtserkennung

Bitte recht freundlich!

Mit dem Raspberry-Pi-Kameramodul und der OpenCV-Software kann Ihr Roboter Gesichter erkennen. Montieren Sie einfach anstelle der Ultraschallsensoren die Kamera auf der Schwenkneigehalterung. Dann kann sich die Kamera selbst bewegen und anderen Bewegungen folgen.

Er spricht!

Erfahren Sie, was in Ihrem Roboter vorgeht.

Der Raspberry Pi hat eine Audioausgabe. Kombiniert man diese mit einem Reiselautsprecher, öffnet sich für Ihren Roboter eine neue Welt der Kommunikation. Mit einem Python-kompatiblen Sprachmodul wie eSpeak können Sie ihm Sprechen und Singen beibringen. Oder er kann zum Debuggen Statusanzeigen melden. Ein weiteres menschliches Element wie Spracherkennung mit einem USB-Mikrofon bringt den Roboter auf eine ganz neue Ebene.

Raumanalyse

Fertigen Sie Karten Ihrer Umgebung.

Mit dem Ultraschallsensor auf dem Schwenkneiger kann Ihr Roboter jede Wand und jedes Objekt in einem Raum vermessen – eine beliebte Spezialisierung in der Informatik. Nach einer Reihe von Messungen in verschiedenen Richtungen, die von den Servos des Schwenkneigers gesteuert werden, kann man eine Karte anfertigen. Mit etwas mehr Code und Zubehör navigiert der Roboter in Ihrem ganzen Haus. PID ist ein spannendes Feld, das dafür sehr hilfreich sein kann.

Schwarmintelligenz

Ein Roboter ist cool, mehr sind besser.

Swarming ist ein interessantes Feld der Informatik. Mit nur wenig mehr Code können wir ein Verhalten ähnlich dem eines Bienenschwarms oder Ameisenstaates erzeugen. Ein Roboterschwarm kann schnell ein Gelände erkunden oder zur Simulierung von Verkehrsgeschehen eingesetzt werden. Sie können Synchronisierungsroutinen damit erzeugen oder sogar Ihr eigenes Roboter-Fußballteam zusammenstellen.

> **“Stellen Sie Ihr eigenes Roboter-Fußballteam zusammen.”**

Mit Greifarm

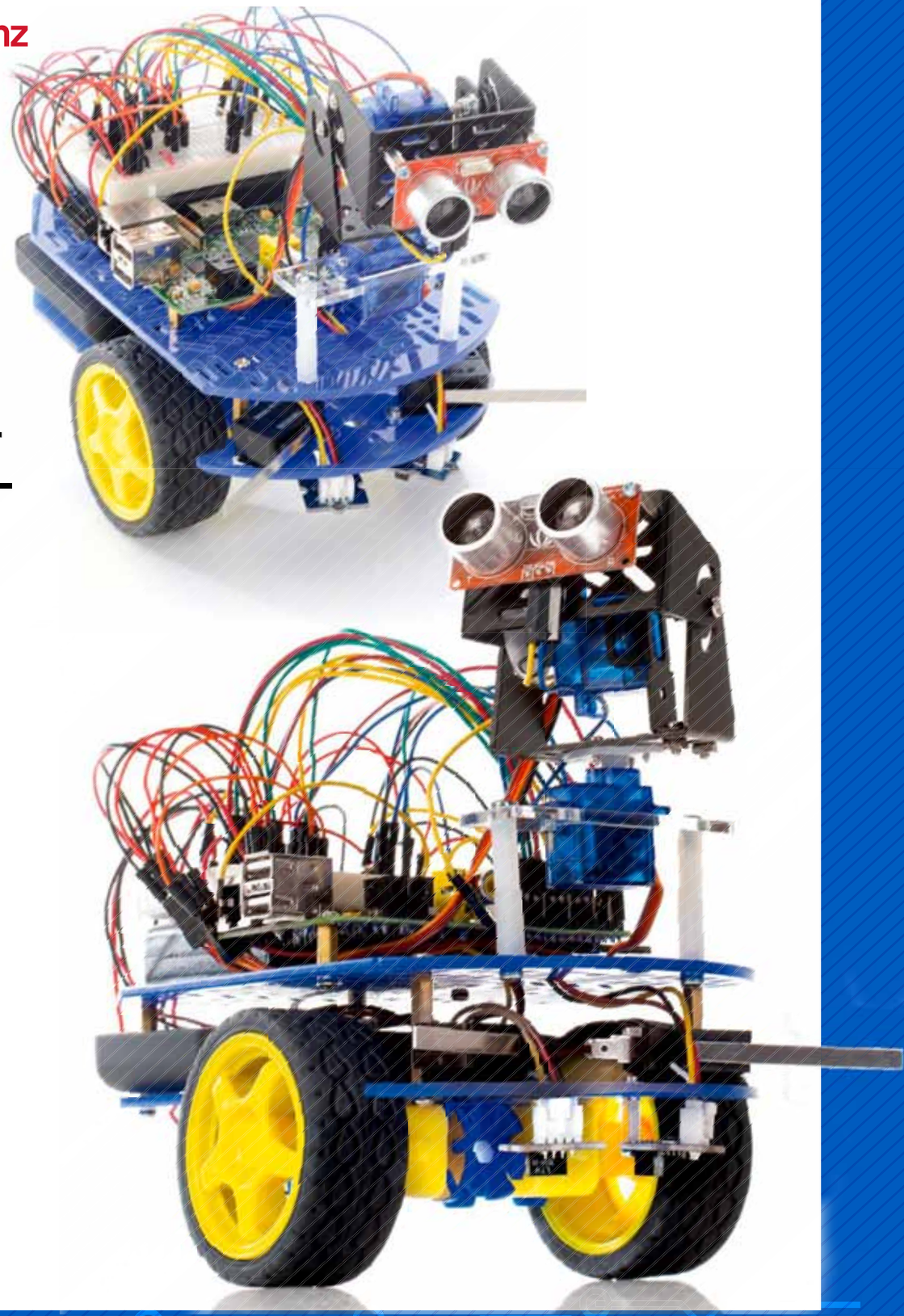
Der Roboter soll es tragen!

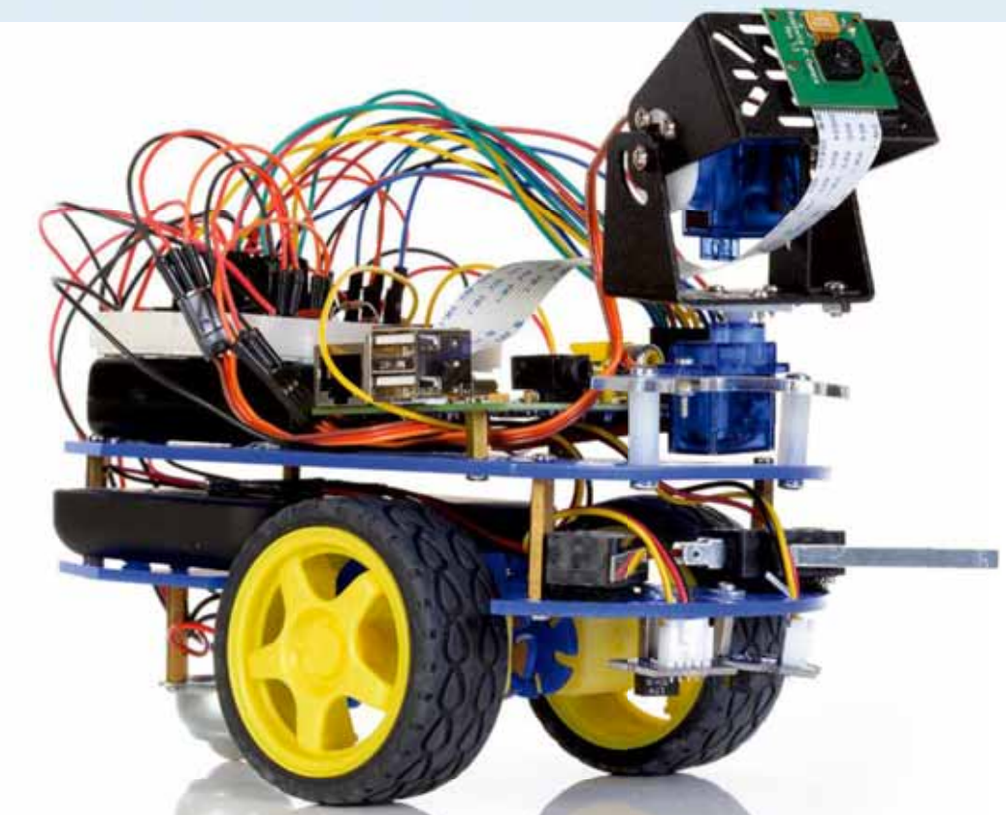
Jeder hätte gerne einen Dienstroboter für die Hausarbeit. An diesem Punkt sind wir leider noch nicht, aber ziemlich nahe dran. Mit einem kleinen Greifarm, installiert an seiner Vorderseite, kann der Roboter leichte Dinge tragen. Und mit dem Pi-Kameramodul oder einem RGB-Farbsensor kann er Legos nach Farben sortieren oder ein Haustier unterhalten.

Wege aus dem Irrgarten

Besser als eine Laborratte.

Pathfinding und Maze Solving sind zwei spannende Felder der Informatik, bei denen Sie Ihren Raspberry-Pi-Roboter einsetzen können. Weltweit finden Maze-Solving-Wettbewerbe statt, in denen Leitlinien oder Ultraschallsensoren eingesetzt werden. Sie brauchen nur einen Mechanismus, der sich vergangene Bewegungen merkt, und eine konzeptionelle Methode, wie der Roboter aus dem Labyrinth findet.





Steuern Sie den Pi-Roboter per Webanwendung

Ihr Raspberry Pi-Roboter ist nun betriebsbereit. Doch mit einer Webanwendung in Python können Sie ihn sogar mit seinen eigenen „Augen“ steuern.

Was Sie brauchen:

- einen Raspberry-Pi-Roboter (siehe Anleitung auf den vorhergehenden Seiten)
- das Raspberry-Pi-Kameramodul
- eine WLAN-Verbindung

Seit wir unseren Raspberry-Pi-Roboter gebaut haben, überlegen wir, welche Fähigkeiten wir ihm noch geben können. Wir kamen auf die Idee, den Roboter von einem mit dem Netz verbundenen Gerät zu steuern, über eine einfache Webanwendung. Um die Sache interessanter zu machen, haben wir das Kameramodul des Raspberry Pi hinzugefügt, sodass wir quasi mit den Augen von Johnny Pi „sehen“ können. Solange er sich in Reichweite des WLAN befindet, kann man den Roboter theoretisch einfach herumfahren lassen und ihn über das Video-Feedback der App steuern.

Sie erfahren hier, wie man MJPG-Streamer verwendet, um ein Video direkt von der Kamera des Raspberry Pi auf den Webbrowser zu streamen. Dann konvertieren wir den Bewegungs-Code aus dem vorherigen Artikel, sodass er von einer Webanwendung ausgeführt werden kann. Diese schreiben wir in Python. Es ist kinderleicht!

01 Anschluss der Kamera

Die blaue Seite der Kamera kommt an die Rückseite der Ethernet-Buchse des Raspberry Pi, und die Seite mit den Pins an die SD-Karte. Ziehen Sie den Steckverbinder hoch und schieben Sie das Flachbandkabel so weit es geht hinein. Drücken Sie den Verschluss nach unten, damit das Kabel fest sitzt.

02 Aktivierung der Kamera

Loggen Sie sich mit dem Nutzernamen „pi“ und dem Passwort „raspberry“ in das Raspbian-System ein. Achtung – alte Versionen unterstützen die Kamera nicht. Aktualisieren Sie Ihr System mit `sudo apt-get update` und dann `sudo apt-get upgrade`. Danach führen Sie `sudo raspi-config` aus und wählen die Option „Kamera aktivieren“. Die Änderungen werden beim nächsten Neustart übernommen.

03 Installation der MJPG-Streamer-Abhängigkeiten

Wir verwenden eine experimentelle Version von MJPG-Streamer, die ein Input-Modul für die Raspberry-Pi-Kamera besitzt. Diese Version liegt für den Raspberry Pi nicht in fertigen Paketen vor, deshalb müssen wir sie selbst kompilieren. Aktualisieren Sie den Paketindex mit sudo apt-get update. Installieren Sie dann Subversion, womit wir den Quellcode herunterladen. Dazu brauchen wir noch libjpeg und imagemagick, ohne die der MJPG-Streamer nicht läuft. Installieren Sie diese Programme mit:

```
sudo apt-get install git cmake
libjpeg8-dev imagemagick
```

04 MJPG-Streamer kompilieren

Laden Sie den MJPG-Streamer herunter und kompilieren Sie ihn mit diesem Code:

```
git clone https://github.com/liamfraser/
mjpg-streamer
cd mjpg-streamer/mjpg-streamer-
experimental
make clean all
```

05 Erster Testlauf

Bevor wir MJPG-Streamer starten, müssen wir das darin enthaltene Verzeichnis exportieren, damit der Loader weiß, von wo er die Input- und Output-Module laden soll. Wir exportieren es als **STREAMER_PATH**, aber der Pfad muss sich in **LD_LIBRARY_PATH** befinden, damit die Module (.so-Dateien) zu finden sind. Diese Variablen stellen wir so ein:

```
export STREAMER_PATH=/home/pi/mjpg-
streamer/mjpg-streamer-experimental

export LD_LIBRARY_PATH=$STREAMER_PATH
```

Starten Sie dann MJPG-Streamer folgendermaßen:

```
$STREAMER_PATH/mjpg_streamer -i „input_
raspicam.so -d 200“ -o „output_http.so
-w $STREAMER_PATH/www“
```

Dabei steht **-d 200** für die Anzahl der Millisekunden zwischen den Aufnahmen. Den Videostream sehen Sie im Browser unter **http://[IP-Adresse des Pi]:8080/stream.html**.

06 Streamer-Start beim Booten

Erstellen Sie eine Sicherheitskopie von **/etc/rc.local** mit `sudo cp /etc/rc.local /etc/rc.local.bak`. Geben Sie (mit `sudo`) die Zeilen aus dem vorherigen Schritt in **/etc/rc.local** ein. Stellen Sie ein Leerzeichen gefolgt von einem &-Zeichen ans Ende der letzten Zeile, wobei **exit 0** am Schluss stehen muss. Starten Sie den Pi neu. Am Ende sollte die **rc.local**-Datei so aussehen:

07 Installation von Apache und nötigen Modulen

Wir verwenden Apache als Webserver, WSGI, um Python vom Webserver auszuführen, und Pyro (Python Remote Objects) für Funktionen, mit denen wir die Kamera und den Roboter bewegen. So braucht der Webserver keine Root-Rechte, um auf das GPIO zuzugreifen. Installieren Sie die Pakete mit:

```
sudo apt-get install apache2
libapache2-mod-wsgi pyro
```

Wir brauchen noch einen Pyro-Nameserver, um später über das Netz auf das Bewegungsmodul zugreifen zu können. Ändern Sie in der **/etc/default/pyro-nsd** das **ENABLED** auf **1**. Starten Sie anschließend pyro-nsd:

```
sudo /etc/init.d/pyro-nsd start
```

08 WSGI-Konfiguration

Um WSGI zu verwenden, legen wir einen Nutzer an, der den Code ausführt, beispielsweise unter dem Namen „robotweb“. Verwenden Sie den Befehl `sudo adduser robotweb` mit einem Passwort Ihrer Wahl. Bearbeiten Sie **/etc/apache2/sites-enabled/000-default** mit `sudo` im Editor. Fügen Sie folgende Zeilen innerhalb der VirtualHost-Tags ein, direkt vor dem Error-Log-Block.

```
WSGIDaemonProcess robotweb
user=robotweb group=robotweb
processes= 1 threads=2

WSGIProcessGroup robotweb

WSGIScriptAliasMatch /action /home/
robotweb/app.py
```

09 WSGI Hallo Welt!

Wechseln Sie mit `su robotweb` zu „robotweb“ als Nutzer. Öffnen Sie **/home/robotweb/app.py** im Editor und fügen Sie den Code aus der Abbildung ein. WSGI ruft zunächst die Anwendungsfunktion der Datei auf, die Sie ihm zuweisen, es durchläuft ein Dictionary mit Umgebungsvariablen und die Callback-Funktion **start_response**, mit der Sie den HTTP-Status samt Antwort-Headern senden. Sobald WSGI diese erhält, wird der Hauptinhalt als Liste zurückgegeben – im Beispiel enthält sie nur einen String.

Dann gehen Sie mit `exit` zurück zum „pi“-Nutzer und tippen `apachectl graceful` ein. Ignorieren Sie Fehlermeldungen, Apache könne den Servernamen nicht bestimmen. Mit dem Aufruf von **http://[Ihr Pi]/action** erscheint „Hello World!“ als Output.

Schritt 06

```
...
fi

export STREAMER_PATH=/home/pi/mjpg-streamer/mjpg-streamer-experimental
export LD_LIBRARY_PATH=$STREAMER_PATH
$STREAMER_PATH/mjpg_streamer -i "input_raspicam.so -d 200" -o "output_http.so -w
$STREAMER_PATH/www" &

exit 0
```

Schritt 09

```
def application(environ, start_response):

    status = '200 OK'
    output = 'Hello World!'

    response_headers = [('Content-type', 'text/plain'),
                        ('Content-Length', str(len(output)))]

    start_response(status, response_headers)

    return [output]
```

10 Erstellung des Bewegungscodes

Das nächste Skript basiert auf den Bewegungs- und Ultraschall-Codes vom vorherigen Artikel. Weil wir den Ultraschall-Teil nicht brauchen, haben wir das Skript umgeschrieben und die Fähigkeit hinzugefügt, den Code ferngesteuert aufzurufen. Die neue Datei **movement_server.py** muss mit `chmod +x` ausführbar gemacht werden. Falls Ihr Roboter sich zu langsam oder zu schnell bewegt, ändern Sie die Drehzahlkonstante.

Die Klasse **movement** muss die Pyro-Objektbasis übernehmen, damit sie von einem Pyro-Client aufgerufen werden kann. Deshalb initialisieren wir die Klasse **base** als Teil der Initialisierungsfunktion der Klasse **movement**. Abgesehen davon ist es eine normale Klasse.

Wir setzen die GPIO-Spannung in beiden Richtungen an den entsprechenden Pins entweder auf HIGH oder LOW, starten die Motoren, lassen sie für die Zeitdauer der Bewegung laufen und rufen dann die **stop**-Funktion auf, die einfach die Drehzahl auf 0 stellt.

Schließlich richten wir einen Pyro-Server ein, indem wir uns als „robotmovement“ beim Pyro-Nameserver anmelden und eine Request-Schleife starten. Sie wartet nur auf Aufforderungen von einem Pyro-Client und führt die entsprechende Funktion aus.

11 Autostart von movement_server.py

Wie beim MJPG-Streamer startet der Bewegungs-Server beim Hochfahren, wenn wir ihn zu **/etc/rc.local** hinzufügen, vor **exit 0**. Fügen Sie dort die folgende Zeile hinzu:

```
/home/robotweb/movement_server.py &
```

12 Das Web-Frontend

Wir verwenden als Web-Frontend Bootstrap 3 von Twitter. Das Framework sieht gut aus und ist leicht zu bedienen. Wechseln Sie ins Verzeichnis **/var/www**. Es wird vom Root-Nutzer kontrolliert, wechseln Sie deshalb mit `sudo su` in den Root-Nutzer. Entfernen Sie zuerst die Standard-Startseite von Apache mit `rm index.html`. Dann laden Sie die Bootstrap-Dateien herunter und extrahieren sie:

```
wget "https://github.com/twbs/bootstrap/releases/download/v3.0.0/bootstrap-3.0.0-dist.zip"

unzip bootstrap-3.0.0-dist.zip

mv dist/* .

rm -r bootstrap-3.0.0-dist.zip dist/
```

Bootstrap erfordert jQuery. Laden Sie es ebenfalls herunter und dazu ein jQuery-Plugin, mit dem wir ein Request abschicken können, ohne dass die Seite neu lädt. Es wäre ziemlich ärgerlich, wenn das Kamerabild bei jedem neuen Befehl an den Roboter verschwände.

```
cd js/

wget "http://malsup.github.com/jquery.form.js"

wget "http://ajax.googleapis.com/ajax/libs/jquery/1.7/jquery.js"
```

13 Erstellung der Webseite

Nun erstellen wir die HTML-Seite **index.html**. Die Datei sollte unter **/var/www/index.html** abgelegt werden und dort die Apache-Standardseite ersetzen, die wir vorhin entfernt haben. Wir geben ihr einen Titel, importieren das Bootstrap-Stylesheet und ändern es ein wenig, um die Seitenbreite zu begrenzen. Danach importieren wir das benötigte JavaScript und schreiben einen kurzen Code, der das **move**-Formular mit AJAX statt mit den üblichen HTTP-Requests handhabt.

14 Body von index.html

Der Body hat einen Header, einen Container für den Videostream sowie ein Formular, über das wir eine Richtung – und die Zeitdauer, wie lange in diese Richtung gefahren wird – an die Python-Webanwendung schicken. Drückt man eine Schaltfläche, wird der Wert für die Richtung und der Zeitwert aus dem Sekunden-Textfeld an das **action**-Skript gesandt.

Die IP-Adresse des Pi ist fest kodiert, damit an Port 8080 auf den Stream zugegriffen werden kann. Mit einer DNS-Adresse für den Pi kann das verbessert werden. Oder der Videostream wird mit Apache via Reverse-Proxy durch Port 80 geleitet.

15 Aufruf des Bewegungs-Codes von der Webanwendung

Unsere Webanwendung ist wirklich einfach, sie benötigt lediglich die Fahrtrichtung und dafür eine Zeitangabe. Der Code oben auf Seite 65 ersetzt das „Hallo Welt!“-Skript, das in **/home/robotweb/app.py** liegt. Wir importieren den Pyro-Kern, um damit den Code auf dem Bewegungs-Server vom Internet aus aufzurufen und die **parse_qs**-Funktion des CGI-Modul zu importieren. Damit können wir Abfrage-Querys (von der Webseite, die wir vorhin erstellt haben) parsen und sie in ein Dictionary umwandeln.

„Unsere Webanwendung ist wirklich einfach, sie benötigt lediglich die Fahrtrichtung und eine Zeitangabe.“

Schritt 15

```
import Pyro.core
from cgi import parse_qs

def application(environ, start_response):

    # Connect to Pyro
    movement = Pyro.core.getProxyForURI("PYRONAME://robotmovement")

    parameters = parse_qs(environ['QUERY_STRING'])

    if 'seconds' in parameters and 'direction' in parameters:
        direction = parameters['direction'][0]
        seconds = int(parameters['seconds'][0])

        # Call the appropriate function
        if direction == 'Stop':
           movement.stop()
        elif direction == 'Forwards':
           movement.forward(seconds)
        elif direction == 'Backwards':
           movement.backward(seconds)
        elif direction == 'Left':
           movement.left(seconds)
        elif direction == 'Right':
           movement.right(seconds)

        status = '200 OK'

    else:
        status = '400 Bad Request'

    response_headers = [('Content-type', 'text/plain'),
                        ('Content-Length', str(len(status)))]
    start_response(status, response_headers)
    return [status]
```

Dann wird eine Verbindung zum Bewegungs-Server hergestellt, den wir beim Pyro-Nameserver als „robotmovement“ angemeldet haben. Steht die Verbindung, erhalten wir unsere Parameter als Dictionary und überprüfen, ob eine gültige Anfrage vorliegt. Wenn ja, erhalten wir die erste Richtung und Sekundenanzahl von der Liste (in jeder Liste ist nur ein Objekt) und rufen die entsprechende Bewegungsfunktion auf.

Hat alles geklappt, wird eine Erfolgsmeldung verschickt, wenn nicht, wird „bad request“ zurückgegeben. Der Nutzer sieht diese Meldungen nicht, weil sie mit jQuery gesendet werden, damit die Kamera nicht immer wieder neu lädt.

16 Noch mehr Tests

Laden Sie Apache mit `apachectl graceful`. Ignorieren Sie Fehlermeldungen, Apache könne den Servernamen nicht bestimmen. Geben Sie die IP-Adresse Ihres Raspberry Pi in einen Browser ein, um die Webseite aufzurufen. Jetzt können Sie die Bewegungen des Roboters mit den Buttons und Sekundenangaben für die Bewegungsdauer steuern. Der Stopp-Button sollte alle anderen Eingaben aufheben.

Dabei stoppen wir den Roboter immer erst nach dem Zeitlimit. Wenn er 5 Sekunden vorwärts und 5 Sekunden rückwärts fährt, wird der Roboter nach 3 Sekunden vorwärts für 2 Sekunden rückwärts bewegt und dann gestoppt. Sie können dieses potenzielle Problem mit einem Queueing-System lösen.

17 Deaktivierung

Die Kamera soll natürlich nicht die ganze Zeit über das Internet senden. Alles, was wir eingerichtet haben, deaktivieren Sie, indem Sie die vier zu **/etc/rc.local** hinzugefügten Zeilen durch ein Hashtag am Anfang jeder Zeile auskommentieren. Apache deaktivieren Sie beim Hochfahren mit `sudo update-rc.d-f apache2 remove` (muss bei jedem Update von Apache neu eingestellt werden). Mit `sudo update-rc.d apache2 defaults` aktivieren Sie Apache, und mit `sudo 0/etc/init.d/apache2` start beziehungsweise `stop` starten und stoppen Sie es manuell.

18 Weitere Verbesserungen

Dieser Artikel vermittelt gutes Basiswissen über ein Web-Interface für Ihren Raspberry-Pi-Roboter. Aber es gibt noch viele weitere Möglichkeiten. Folgendes könnten Sie tun:

- den Schwenkneiger (siehe vorherigen Artikel) ins Web-Interface integrieren
- den Videostream mit Apache über Reverse-Proxy leiten
- eine Authentifizierung einbauen, sodass nur Nutzer mit Passwort den Roboter steuern und den Videostream sehen können
- die Videoverbindung mit SSL sichern
- berechnen, wie lange die Motoren für eine volle Kreisbewegung brauchen, und mit dieser Information den Roboter in bestimmten Winkeln drehen
- ein Queueing-System für Befehle implementieren

Rapiro – der kleine Pi-Roboter

Der über Kickstarter finanzierte Rapiro ist nicht der einzige, aber einer der wichtigsten Raspberry-Pi-Roboter.

Technologie entwickelt sich immer weiter. Zwei der neuesten Errungenschaften sind der extrem erfolgreiche Raspberry Pi sowie Konzepte wie Kickstarter und Crowdfunding. Der Raspberry Pi ist von frühen Heimcomputern inspiriert, doch er besitzt jede Menge moderner Annehmlichkeiten: Handlichkeit, HDMI-Port, geringer Stromverbrauch.

Kickstarter ist ebenfalls für viele kreative Köpfe extrem erfolgreich. Jeden Tag finden Leute für ihre Traumprojekte – in den Bereichen Technologie, Games, Film, Comics, Bücher und vieles mehr – ihre Zielgruppe, die ihnen die nötigen Mittel zuschießt.

Auch wir entdeckten dieses geniale Tech-Projekt auf Kickstarter: den Rapiro. Der in Japan gefertigte Miniroboter wird von Arduino und dem Raspberry Pi angetrieben, der Name steht für „Raspberry Pi Robot“. Durch die Unterstützung der Raspberry Pi Foundation wurde das Finanzierungsziel ziemlich schnell erreicht, der Rapiro wurde sogar zu 375 % überfinanziert. Kickstarter-Kampagnen erhalten die meisten Spenden innerhalb der ersten Tage und dann wieder am Ende. Auch der Entwickler des Rapiro, Shota Ishiwatari, erlebte auf den letzten Metern einen deutlichen Finanzierungsschub.

Der japanische Robotiker erzielte schon mit anderen Kreationen solche Erfolge. Der Rapiro steht in einer Tradition, die, so sagt er, „Niedliches mit Technologie verbindet“. Wir sprachen mit Ishiwatari über die Entwicklung des Rapiro und die Zukunft.

SINNLICH
Mit der optionalen Raspberry-Pi-Kamera und einem Mikrofon-Upgrade kann der Rapiro seinen Besitzer sehen und hören – verstehen kann er ihn auch ein bisschen.

WENDIG
Mit dem Arduino-Controller wird der Rapiro so programmiert, dass er einfache Aufgaben ausführt. Die Übermittlungsstärke auf seine Finger ist so groß, dass er einen Stift halten kann.

„Der Rapiro begann mit zwei Schlüsselworten: 3D-Druck und Raspberry Pi.“

KOMMUNIKATIV
Mit einem Lautsprecher und der richtigen Software spricht der Rapiro mit Menschen. Er kann auch ans Internet der Dinge angeschlossen werden und beispielsweise auf Twitter posten.

Ursprünglich entstand das Rapiro-Projekt aus zwei aktuellen Themen, so Ishiwatari. „Der Rapiro begann mit zwei Schlüsselworten: 3D-Druck und Raspberry Pi. Beide Begriffe waren im vorigen Jahr immer wieder in meinem Umfeld aufgetaucht. Daher beschloss ich, einen Roboter zu entwickeln, der von beidem profitierte."

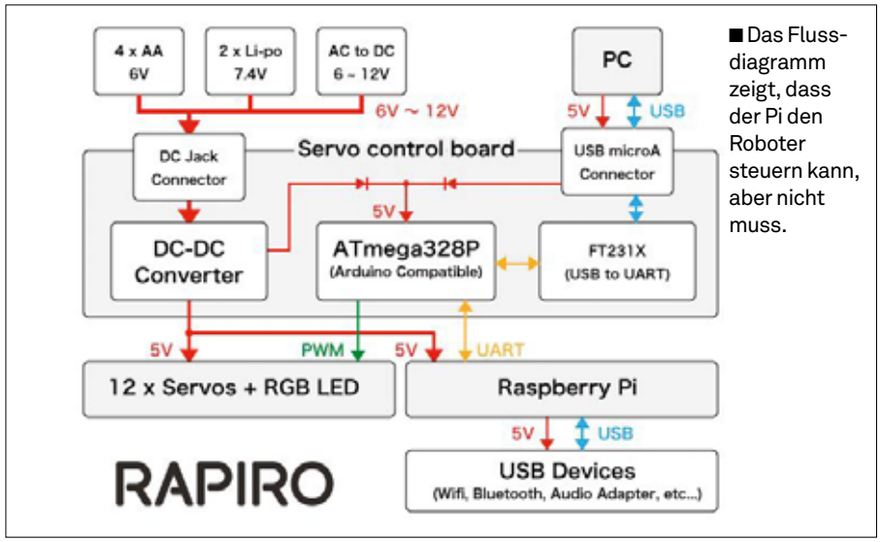
In der Folge wurde aus dem Rapiro fast ein Open-Source-Projekt – er läuft mit Linux auf dem Pi, verwendet Arduino, und der Quellcode für die Demo-Software und sogar die 3D-Daten sollen im STL-Format veröffentlicht werden, sodass jeder einen Rapiro ausdrucken und selbst bauen könnte.

„Ich glaube, dass sowohl der Raspberry Pi als auch Arduino praktisch und weltweit bekannt sind", erklärt Ishiwatari den Schritt zur Freien Software. „Aber ich habe da nicht viel überlegt. Geld damit zu verdienen, war nie mein Antrieb. Die aktuellen Modelle laufen mit Raspbian, und ich glaube, wir verwenden im Moment nur Raspbian und die Arduino-IDE. Für die Spracherkennung benutzen wir Julius."

Der Körper des Rapiro wird vom Raspberry Pi und einem Kontrollpanel gesteuert – wie sieht das im Einzelnen aus?

„Die Servos werden alle vom Kontrollpanel gesteuert, das mit Arduino kompatibel ist", erklärt Ishiwatari. „Der Raspberry Pi steuert die AI (Pi-Kamera, Mikrofon, Spracherkennung und so weiter) über Webanwendungen."

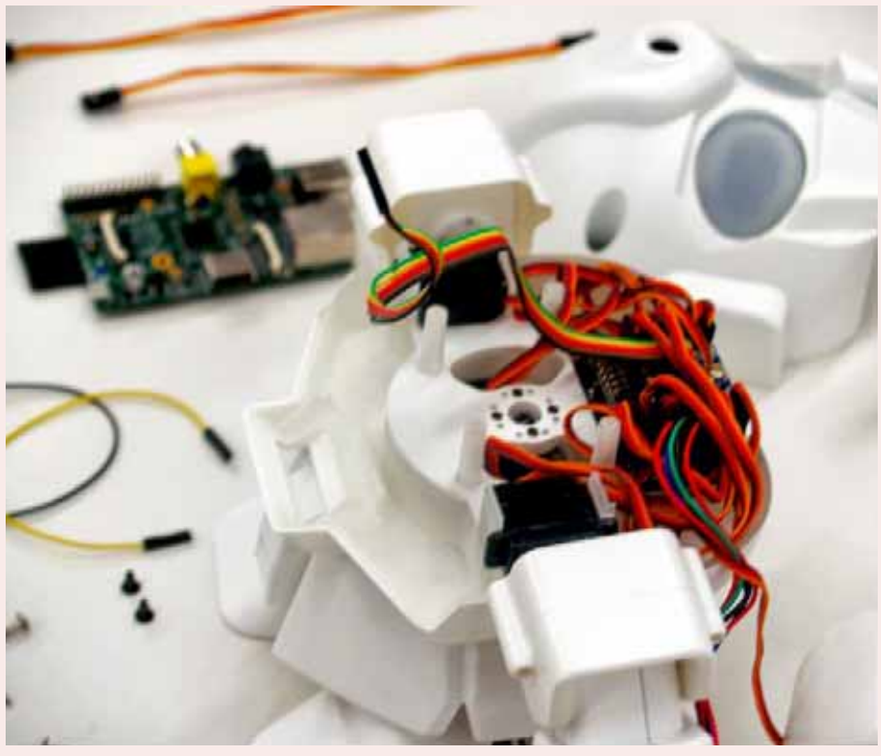
Doch die Hardware ist nur der eine Aspekt. Man braucht Software, um den Roboter zum Laufen zu bringen. Die Software für den Rapiro ist ebenfalls frei, und Ishiwatari erläutert, was man benötigt.



■ Das Flussdiagramm zeigt, dass der Pi den Roboter steuern kann, aber nicht muss.

Rapiro-Daten

Servos 12
Servo-Leistung 6x 2.5kgf-cm, 6x 1.5kgf-cm
Arbeitsgeschwindigkeit der Servos 0.12 sec/60°
Servo-Winkel 180°
Augen vollfarbige LED
Servo-Controller Arduino-kompatibler ATmega328P-Chipsatz
Zusätzliche Rechnerleistung Raspberry Pi (nicht im Bausatz enthalten)
Optik Raspberry-Pi-Kamera (nicht im Bausatz enthalten)
Distribution Raspbian
Sprachen Python, Arduino-IDE
Chassis-Konstruktion 3D-gedrucktes Plastik
Für die Montage erforderlich Schraubenzieher
Vorkenntnisse Keine
Zusätzliche Extras WLAN- oder Bluetooth-Dongle, Mikrofon, IR-LED



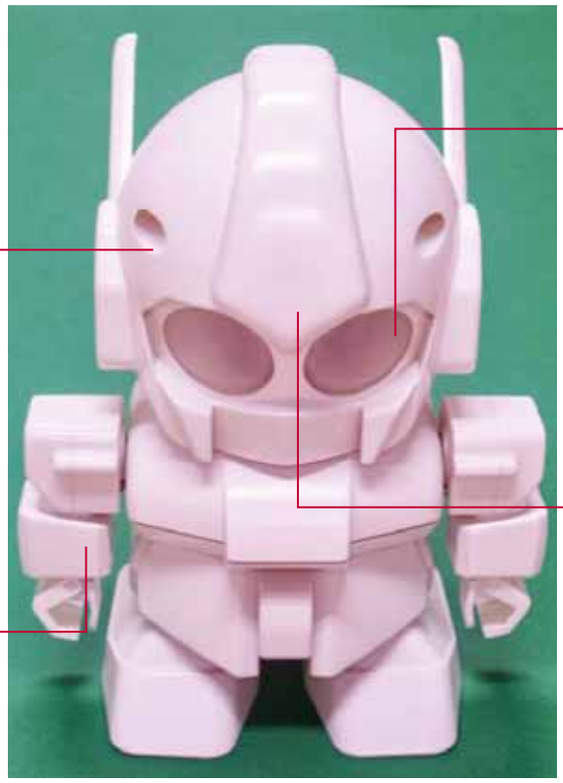
Roboter-Blaupausen

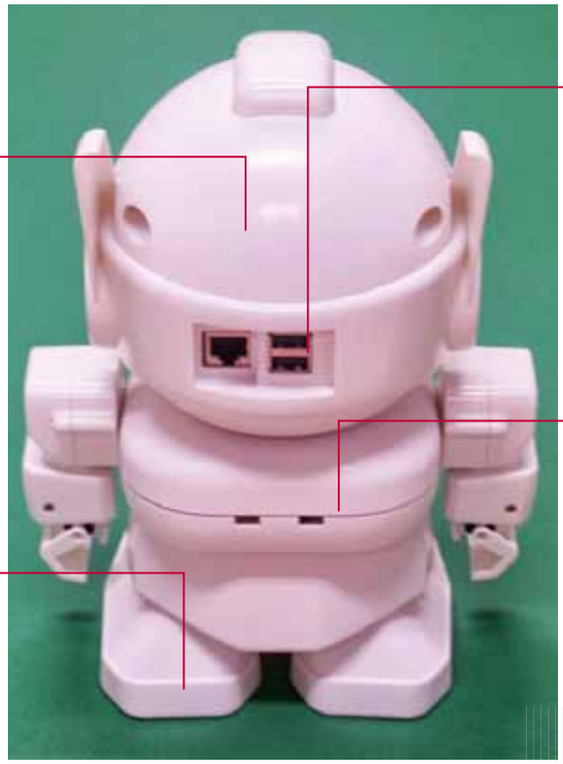
Leichte Montage erforderlich
Die Montage ist leicht, man braucht nur einen Schraubenzieher. Zur Wartung kann der Rapiro auch wieder zerlegt werden.

Eine echte Gelenkpuppe
Die Glieder des Rapiro haben alle Gelenke, er kann gehen und mit den Händen etwa einen Stift greifen.

Hirnleistungen
Im Kopf des Rapiro ist ein Fach für einen Raspberry Pi, das Gehirn des Roboters, der alle Funktionen außer den Bewegungen steuert.

Kontrollübernahme
Ein Arduino-kompatibler Controller steuert 12 Motoren, die strategisch im Körper des Rapiro verteilt sind und ihn bewegen.





Regenbogen der Farben
Die Augen werden von vollfarbigen RGB-LEDs von hinten beleuchtet, sie können eine ganze Reihe von strahlenden Farben annehmen.

Allwissendes Auge
Eine Raspberry-Pi-Kamera kann auf der Stirn platziert werden, damit Rapiro „sehen" und auf seine Umwelt reagieren kann.

Upgrades möglich
Es gibt viele Extras, mit denen man den Rapiro noch ausstatten kann: PSD-Abstandssensoren, Mikrofone, Lautsprecher und mehr.

Ein Replikant
Die Bauteile sind 3D-gedruckt, mit Schlitzen, in die Sie elektronische Komponenten wie die Servos, das Kontrollpanel und andere Extras platzieren können.

Shota Ishiwatari

■ Shota Ishiwatari ist CEO in seinem eigenen Unternehmen, Kiluck.co.jp.

Hinter dem Roboter – der Entwickler von Rapiro

Der Rapiro-Entwickler ist kein Amateur, der sich für Roboter interessiert. Er ist gut ausgebildet für die Herstellung der Miniatur-Automaten. An der Uni habe er Maschinenbau studiert, erzählt Ishiwatari. In Japan hat er offenbar an vielen Robotik-Wettbewerben teilgenommen und war 2005 sogar bei der Rescue Robot League dabei, einem Teil des internationalen RoboCup. Die Rescue League lässt urbane Such- und Rettungsroboter gegeneinander antreten, und 2005 gewann Ishiwataris Team den ersten Platz für Mobilität. Der Rapiro ist nicht sein erster Vorstoß in die Robotik, sondern er entwickelte davor schon den Necomini – ein Wortspiel auf das japanische Wort für Katzenohren. Der Necomini reagiert auf die Stimmungen der Nutzerin, indem er von Anime inspirierte Catgirl-Emotionen nachmacht.

„Der Erfolg von Necomini hat mir Selbstvertrauen gegeben und [einen guten Ruf]“, sagt Ishiwatari. Inzwischen entwickelte er mechanische Fellschwänze, die ähnlich wie der Necomini funktionieren. Für dieses Projekt hatte er auch zum ersten Mal Kickstarter getestet.

Der Raspberry Pi wird über Python angetrieben, die Servo-Teile benutzen die Standard-Arduino-IDE. Zurzeit arbeiten wir an drei Programmen für den Pi: japanische Spracherkennung mit Julius; Android- und iOS-Steuerung über WLAN; Game Controller-Unterstützung für das PS3-Gamepad und für Wii-Fernbedienungen über Bluetooth.

„Wir veröffentlichen den Beispielcode in Python für den Rapiro, sobald die Kickstarter-Kampagne zu Ende ist, allerdings nicht als Pakete für Raspbian oder als Custom Image für den Pi.“

Ein Hauptmerkmal des Rapiro ist, dass er mit lediglich einem Schraubenzieher zusammengebaut werden kann – ohne zusätzliches Löten auf der Platine – und die Servos sich an den Controller klammern lassen.

„Wir haben das so konstruiert, damit selbst Kinder den Roboter bauen können. Das ist der Grund, warum der Rapiro auch ohne den Pi laufen wird. Arduino können sogar Anfänger programmieren, während das beim Pi schwieriger ist. Der Rapiro-Bausatz könnte ein preiswertes Lehrangebot sein, denke ich. Oder man hat mit ihm als Spielzeug oder Hobby seinen Spaß.“

Der Rapiro wird von drei Hardware-Partnern gebaut. Jeder dient einem speziellen Zweck, wie Ishiwatari aufzeigt: „Unser 3D-Druck-Dienstleister ist JMC Inc. Wir lassen bei ihnen die Prototypen 3D-drucken. Sonst produzieren sie 3D-Drucke für Zahnärzte und Kardiologen.

SWITCHSCIENCE ist ein japanischer Online-Shop, der elektronische Bauteile (und auch Raspberry Pis) verkauft. Sie engagieren sich bei der Entwicklung von Freier Hardware in Japan. So haben sie eine Mehrsprachenunterstützung für die Arduino-IDE programmiert und die Anwendung der Community gespendet. Wir haben mit ihnen beim Design für das mit Arduino kompatible Servo-Kontrollpanel zusammengearbeitet.

MIYOSHI Co, Ltd ist eine Kunststoffspritzgießerei, sie werden die Einzelteile des Rapiro für uns anfertigen. Sie haben auch Erfahrung mit dem Gießen von Aluminiumteilen für die Prototypen. Normalerweise fertigen sie Plastikteile für japanische Autos.“

Im Moment ist der Roboter nur ein Selbstbausatz, in den man ein bisschen Arbeit stecken muss, bevor er läuft. Wenn es nach Ishiwatari geht, ändert sich das aber in der Zukunft. „Wenn Nachfrage besteht, möchte ich für den allgemeinen Markt einen Roboter herstellen, der schon zusammengebaut und mit der Software ausgestattet ist.“

Wie sieht die Zukunft von Ishiwatari aus? Er möchte bei weiteren Produkten niedliches Design mit Technologie verbinden und sich vielleicht mit ROS beschäftigen, Middleware für persönliche Roboter.

Doch jetzt ist der Rapiro draußen in der Welt, und Sie können sich noch heute einen auf **rapiro.com** bestellen. Wenn Sie allerdings Zugang zu einem 3D-Drucker haben, brauchen Sie nur ein benutzerdefiniertes Controller Board, mit dem Sie Ihren eigenen Rapiro ausdrucken. Sie können sogar in irgendeiner CAD-Software das Design modifizieren und so ein einzigartiges Produkt herstellen. Denn mehr noch als ein Raspberry-Pi-Roboter ist der Rapiro vor allem ein Open-Source-Roboter.

■ Der Rapiro sieht aus wie der japanische Modellbausatz eines Roboters, doch er bewegt sich wirklich.

Kickstart zum Erfolg

Man braucht Glück auf der Crowdfunding-Plattform Kickstarter, ist Ishiwataris Erfahrung.

Der Rapiro ist nicht das erste Kickstarter-Projekt von Shota Ishiwatari. Davor wollte Kiluck über Crowdfunding den Tailly finanzieren, einen mechanischen Fellschwanz, der auf Emotionen reagiert. Die Kampagne war nicht erfolgreich, aber auch nicht ganz vergebens, sagt Ishiwatari. Kiluck habe viel dazugelernt über Alter und Geschlecht der Zielgruppe, die sich für seine Projekte interessiere. Mit diesen Erfahrungen fand er erfolgreich Kickstarter-Investoren für den Rapiro und hat inzwischen mit der Produktion des Roboter-Bausatzes begonnen. Wir haben Ishiwatari gefragt, ob er es noch einmal bei Kickstarter versuchen werde. Er antwortete mit einem Wort: „Vielleicht."

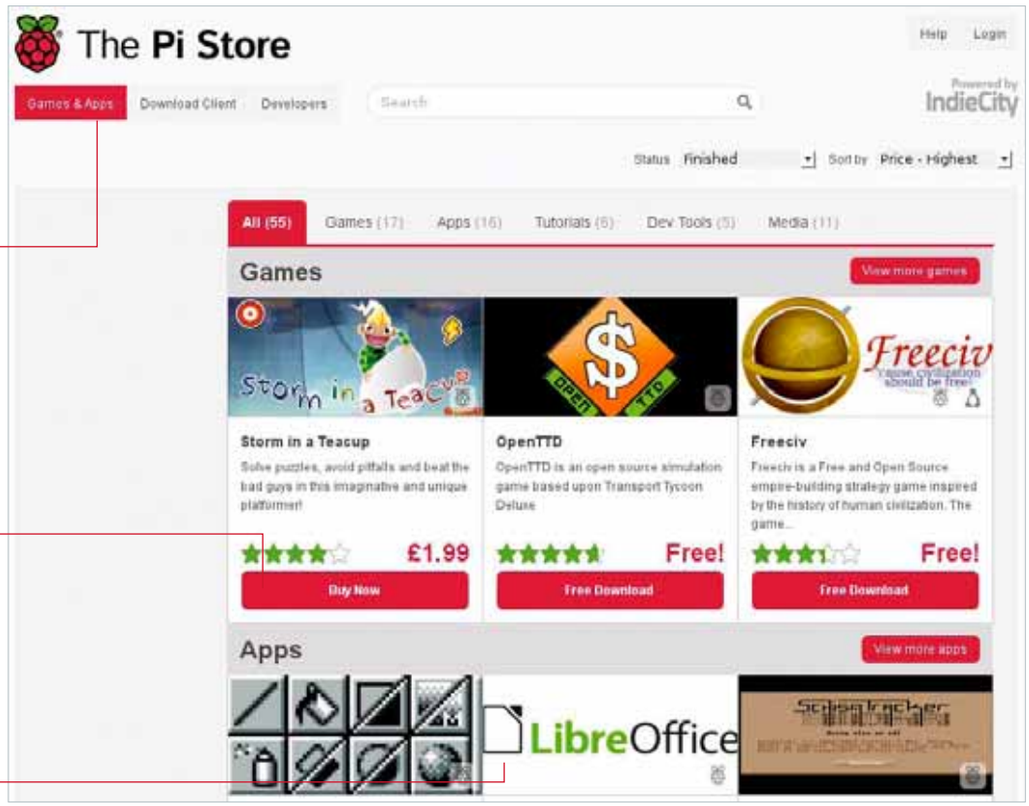
KICKSTARTER

■ Kickstarter ist bei unabhängigen Kreativen sehr beliebt.

Es gibt nicht nur Spiele im Pi Store, sondern auch Tutorials, Entwicklungs-Tools und Anwendungen.

Sobald Sie sich registriert haben, können Sie im Handumdrehen Anwendungen und Spiele installieren.

Auch LibreOffice ist inzwischen für den Raspberry Pi erhältlich und funktioniert sogar ziemlich gut.



Laden Sie Ihre App in den Pi Store

Sie haben ein Spiel oder eine App für den Raspberry Pi entwickelt? Bieten Sie Ihr Werk doch im App-Store an! Wir zeigen Ihnen, wie es geht.

Was Sie brauchen:

- Originalinhalte
  zum Hochladen in den App-Store
- einen Raspberry Pi
  mit dem aktuellen Raspbian-Image und einer Internetverbindung
- eine E-Mailadresse
  die Sie für die Registrierung im App-Store benutzen können

Bevor wir loslegen

Sie können das aktuelle Raspbian-Image unter **raspberrypi.org/downloads** herunterladen. Ziehen Sie das Image von hier wie gewohnt auf Ihre SD-Karte. Eine Anleitung dazu finden Sie unter **linuxuser.co.uk/tutorials/how-to-set-up-raspberry-pi**. Springen Sie direkt zu dem Schritt, bei dem Sie das Image auf die SD-Karte schreiben. Sie müssen die Anleitung leicht anpassen, um das neuere Raspbian-Image (statt das von Debian) zu verwenden.

Die Raspberry Pi Foundation betreibt einen App-Store in Zusammenarbeit mit IndieCity. In den Worten der Macher: „Wir führen den Pi Store ein, um es Entwicklern jeden Alters einfacher zu machen, ihre Spiele, Anwendungen, Tools und Tutorials mit dem Rest der Community zu teilen. Wir hoffen, dass der Pi Store eine zentrale Anlaufstelle für all das wird, was Ihr Raspberry Pi benötigt. Ebenso ist er auch für blutige Anfänger eine einfachere Möglichkeit, Erfahrungen mit dem Raspberry Pi zu sammeln, da sie hier alles Notwendige kostenlos und an ein und derselben Stelle finden."

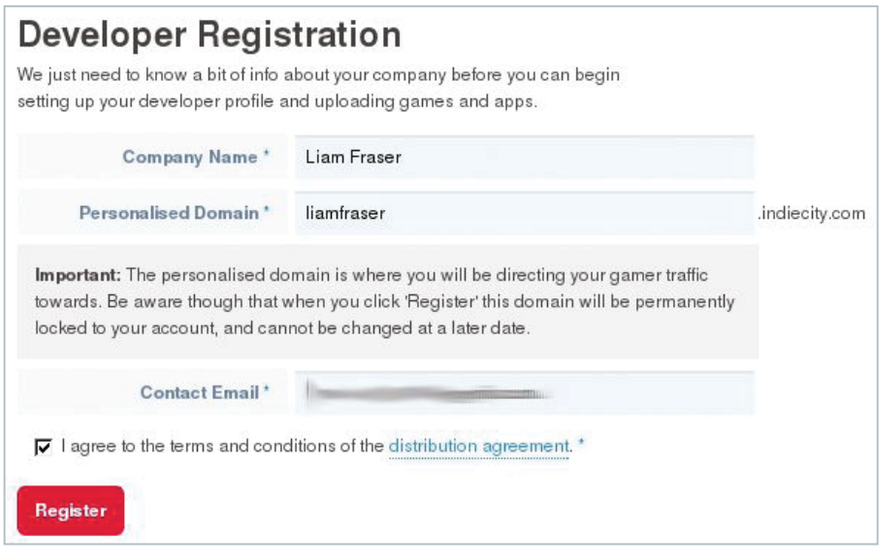
Der App-Store ist eine tolle Sache für Entwickler: Sie können sowohl ihre Arbeiten mit anderen teilen als auch, sofern gewünscht, für ihre Software eine kleine Gebühr verlangen. Beispielhaft werden wir hier ein simples Tic-Tac-Toe-Spiel in den Pi Store hochladen.

01 App-Store aufrufen

Der App-Store funktioniert als X-Windows-Anwendung. Rufen Sie also die LXDE-Desktop-Umgebung mit dem Befehl „startx“ auf und klicken Sie doppelt auf den Pi-Store-Shortcut auf dem Desktop. Möglicherweise braucht der Store etwas Zeit zum Öffnen, weil er seine Paketlisten aktualisieren muss, wenn er das erste Mal gestartet wird.

02 In den Pi einloggen

Sie müssen sich registrieren, bevor Sie im App-Store agieren können. Klicken Sie auf „Login“ oben rechts auf dem Bildschirm und dann auf den erscheinenden „Register“-Button. Die Registrierung ist stressfrei, Sie müssen nur eine E-Mailadresse, ein Passwort und eine Sicherheitsfrage angeben. Sobald Sie die erforderlichen Felder ausgefüllt haben, klicken Sie auf den Anmelde-Button, und schon haben Sie ein Benutzerkonto unter einem Namen à la „IndieGamer17467“. Sie können Ihren Benutzernamen aber ändern und außerdem einen Avatar für Ihr Konto hochladen, indem Sie den blauen „Edit“-Link klicken und die entsprechenden Felder ausfüllen.



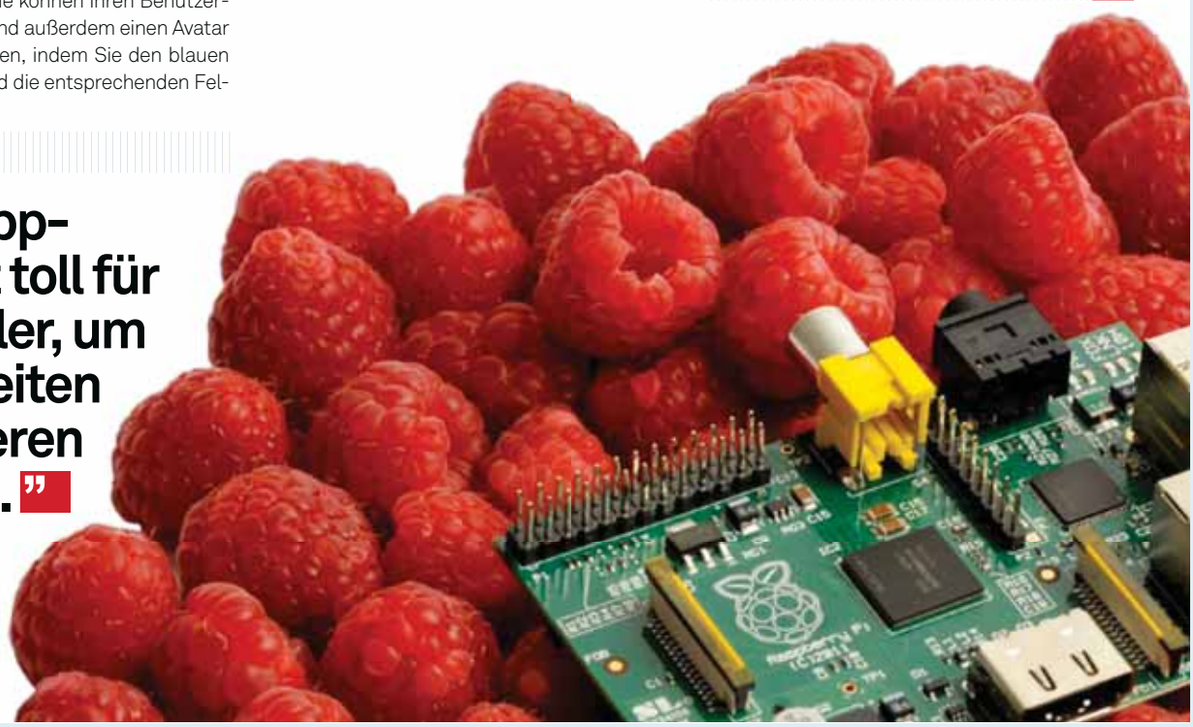
03 Registrierung als Entwickler

Wenn Sie sich als Standardnutzer registriert haben, können Sie auf den Tab „Upload“ klicken und sich als Entwickler anmelden. Wenn Sie keinen Firmennamen besitzen, können Sie Ihren eigenen Namen angeben. Sie müssen weiterhin eine Kontakt-E-Mailadresse eintragen und die Allgemeinen Geschäftsbedingungen der Vertriebsvereinbarung akzeptieren. Wenn Sie mit allem zufrieden sind, klicken Sie auf „Register“.

04 Deaktivieren der Standard-Bluetooth-Plugins

Sobald Sie sich als Entwickler registriert haben, werden Sie auf die Entwickler-Seite geführt. Hier können Sie Informationen über sich selbst oder Ihre Firma eintragen, einschließlich einer öffentlichen E-Mailadresse oder einer eigenen Website. Stöbern Sie ruhig im Entwickler-Bereich herum »

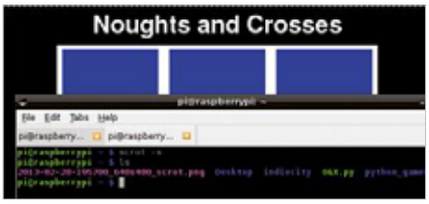
“Der App-Store ist toll für Entwickler, um ihre Arbeiten mit anderen zu teilen.”



und füllen Sie je nach Wunsch die Informationen aus. Wenn Sie auf den Tab „My Company“ gehen, können Sie ein Firmenlogo hochladen und Ihren Account sogar mit Social-Media-Konten wie Twitter oder Facebook verlinken.

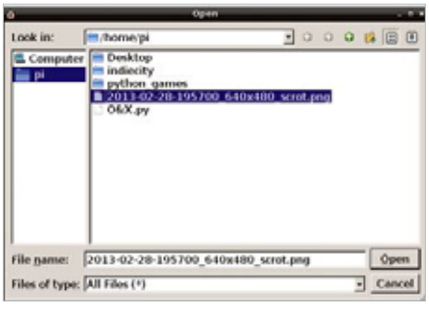
05 Projekt erstellen

Gehen Sie zum Tab „My Games & Apps“ und wählen Sie die Art des Projekts aus, das Sie erstellen möchten. Da wir als Beispiel ein Spiel hochladen möchten, wählen wir die Option „Game“ aus.



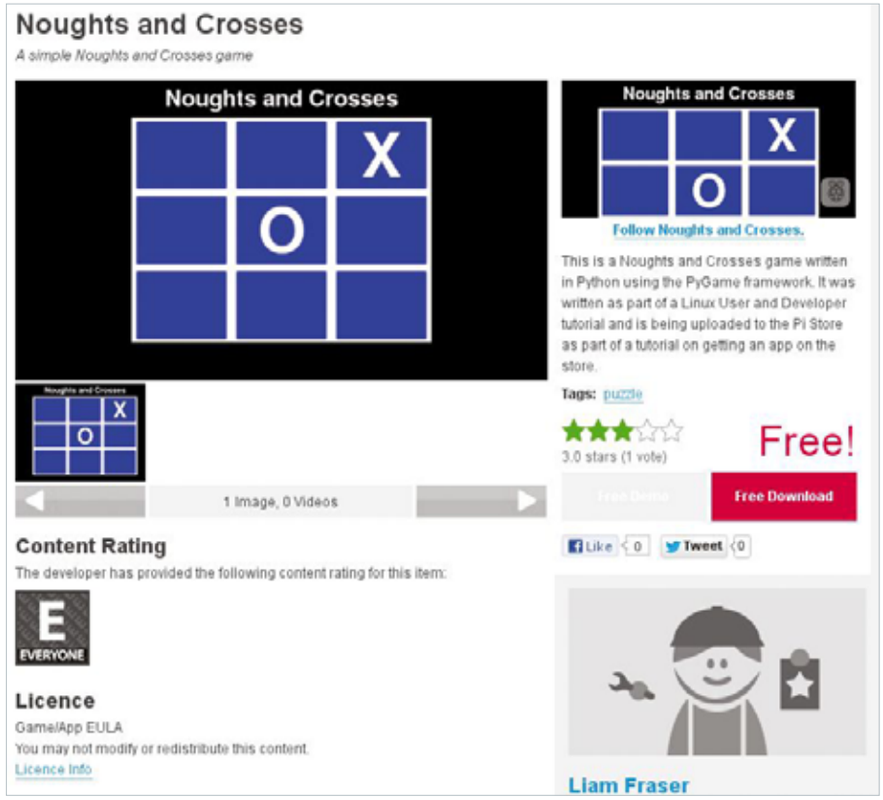
06 Screenshots aufnehmen

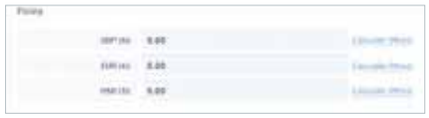
Sie werden Screenshots Ihrer Anwendung benötigen. Diese können Sie zum Beispiel mit dem Programm Scrot aufnehmen, welches Sie per „sudo apt-get install scrot“ installieren können. Sobald Sie Scrot installiert haben, öffnen Sie Ihre Anwendung sowie ein Terminal, in das Sie „scrot -s“ eintippen. Klicken Sie dann auf Ihre Anwendung. Scrot wird einen Screenshot von dem Fenster aufnehmen, auf das Sie geklickt haben, und eine Bilddatei in das aktuelle Verzeichnis Ihres Terminals speichern, benannt nach Datum und Uhrzeit des Screenshots.



07 Ausfüllen der Informationen

Gehen Sie die Projektseite und verschiedene Tabs durch, füllen Sie die Infos über die Anwendung aus, und laden Sie Bilder hoch. Beim Bilder-Upload sollten Sie beachten, dass Sie auf den Pi-Ordner klicken müssen, um auf





Ihre Dateien zugreifen zu können. Beim Ausfüllen des Preises können Sie jeden Wert auf null stellen, wenn Sie Ihre App kostenlos anbieten möchten.

08 Festlegen von URL und Ordnernamen

Gehen Sie zum Tab „Platforms/Installers“. Im ersten Schritt legen Sie eine URL sowie einen Ordnernamen für das Projekt fest, um fortfahren zu können. Sobald Sie die URL und den Ordnernamen bestimmt haben, klicken Sie auf „Save“.

09 Erstellen einer Zip-Datei, die Dateien des Projekts enthält

Projekte werden im Pi Store als Zip-Dateien hochgeladen, weil ein Zip-Archiv in einer einzelnen Datei alles enthalten kann, was Ihr Programm braucht, um zu laufen (etwa auch Sounds und Bilder). In unserem Beispiel

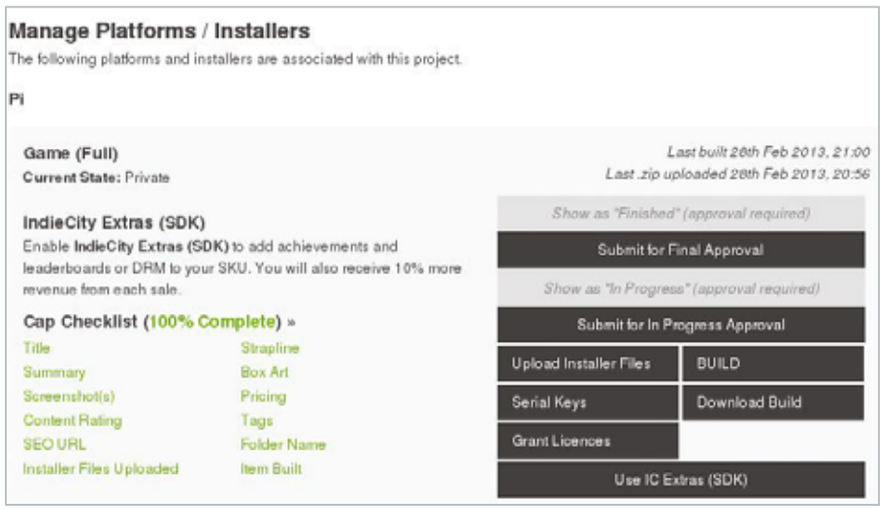
```
pi@raspberrypi ~ $ mkdir NoughtsAndCrosses
pi@raspberrypi ~ $ mv NoughtsAndCrosses.py NoughtsAndCrosses/.
pi@raspberrypi ~ $ zip -r Liam-NoughtsAndCrosses.zip NoughtsAndCrosses/
  adding: NoughtsAndCrosses/ (stored 0%)
  adding: NoughtsAndCrosses/NoughtsAndCrosses.py (deflated 75%)
pi@raspberrypi ~ $ ls
2013-02-28-195700_640x480_scrot.png  Liam-NoughtsAndCrosses.zip
Desktop                              NoughtsAndCrosses
indiecity                            python_games
pi@raspberrypi ~ $
```

erstellen wir für das Projekt ein Verzeichnis mit mehreren Dateien darin für das Zip-Archiv. Da Raspbian allerdings nicht standardmäßig mit Zip kompatibel ist, müssen Sie zuvor die Funktionalität mit „sudo apt-get install zip“ installieren. Dann können Sie eine Zip-Datei erstellen wie unten dargestellt. Der Parameter „-r“ weist den Zip-Befehl an, rekursive Dateien hinzuzufügen, was im Grunde bedeutet, Dateien in Verzeichnissen hinzuzufügen.

```
pi@raspberrypi ~ $ mkdir
NoughtsAndCrosses
pi@raspberrypi ~ $ mv O&X.py
NoughtsAndCrosses/NoughtsAndCrosses.
py
pi@raspberrypi ~ $ zip -r
Liam-NoughtsAndCrosses.zip
NoughtsAndCrosses/
  adding: NoughtsAndCrosses/ (stored
0%)
  adding: NoughtsAndCrosses/
NoughtsAndCrosses.py (deflated 75%)
```

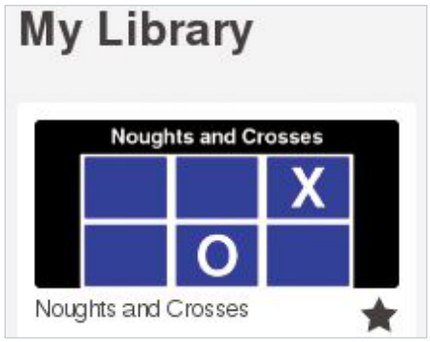
10 Projekt hochladen

Nachdem Sie alles ausgefüllt und die gewünschte Lizenzvariante für Ihr Projekt festgelegt haben, können Sie das Zip-Archiv hochladen. Klicken Sie in der Abteilung „Manage Platforms/Installers“ auf „Pi“ und wählen Sie „Game (Full)“. Sobald Sie das erledigt haben, wählen Sie den Link „Upload Installer Files“ und laden Sie Ihre Zip-Datei hoch. Wenn Ihre Anwendung in Python geschrieben ist, müssen Sie spezifizieren, dass sie mit Python gestartet werden soll. Anschließend müssen Sie den Pfad zu derjenigen ausführbaren Datei festlegen, die den Start der Anwendung ausführen soll. Mit einem Klick in das Feld wird eine Liste der möglichen Dateien angezeigt. Sobald das Formular korrekt ausgefüllt ist, klicken Sie auf den Button „Confirm Settings“.



11 Das Spiel aufbauen

Sobald Sie alle Aspekte auf der Seite mit den Anwendungsinformationen korrekt ausgefüllt haben, können Sie zurück in die Sektion „Manage Platforms/Installers“ gehen und den „BUILD“-Button anwählen, um die Anwendung aufzubauen. Der Aufbauprozess sollte nur ein paar Sekunden dauern.



12 Ihr Ergebnis ausprobieren

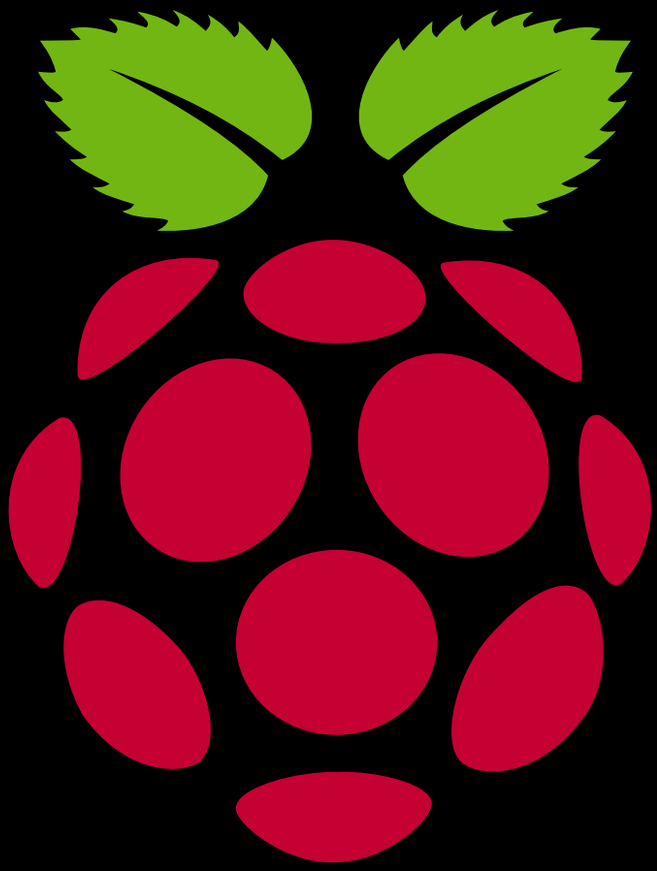
Wählen Sie die Option „Download Build“. Sobald Sie darauf klicken, gelangen Sie in Ihre Bibliothek und das Ergebnis dessen, was Sie hochgeladen haben, wird heruntergeladen. Klicken Sie doppelt auf das Spiel, sobald es installiert ist, um es auszuprobieren.

13 Ihr Spiel zur Genehmigung einreichen

Wenn das Spiel läuft, können Sie zurück zur Sektion „Manage Platforms/Installers“ gehen und das Spiel zur Genehmigung einreichen. Sie können das Spiel sowohl zum „In Progress Approval“ als auch zum „Final Approval“ einreichen. Letzteres bedeutet, dass das Spiel im Store verfügbar gemacht werden soll, während die „In Progress“-Genehmigung bedeutet, dass das Spiel als noch nicht fertig markiert ist, aber trotzdem von Personen, die die Sektion „In Progress“ durchsehen, ausprobiert werden kann. Markieren Sie das Kästchen, um anzuzeigen, dass Sie Ihr Spiel getestet haben. Sobald Sie die Informationen ausgefüllt haben, klicken Sie den „Submit“-Button. Wenn Sie eine verbesserte Version Ihres Spiels hochgeladen haben, können Sie Infos über etwaige reparierte Bugs oder neue Features schreiben. Sie werden eine E-Mail erhalten, die Sie darüber benachrichtigt, dass das Spiel für den Genehmigungsprozess der Community eingereicht wurde.

14 Abwarten

Der Genehmigungsprozess innerhalb der Community kann einige Tage dauern. Unser Spiel war innerhalb von drei Tagen im Store verfügbar. Solange Sie keine urheberrechtlich geschützten Inhalte hochgeladen haben und Ihr Spiel funktioniert, sollte es kein Problem darstellen, Ihre App in den Pi Store zu bekommen. Sie erhalten eine weitere E-Mail, wenn Ihre Anwendung den Genehmigungsprozess der Community erfolgreich durchlaufen hat.



RASPBERRY PI TOOLS & APPS

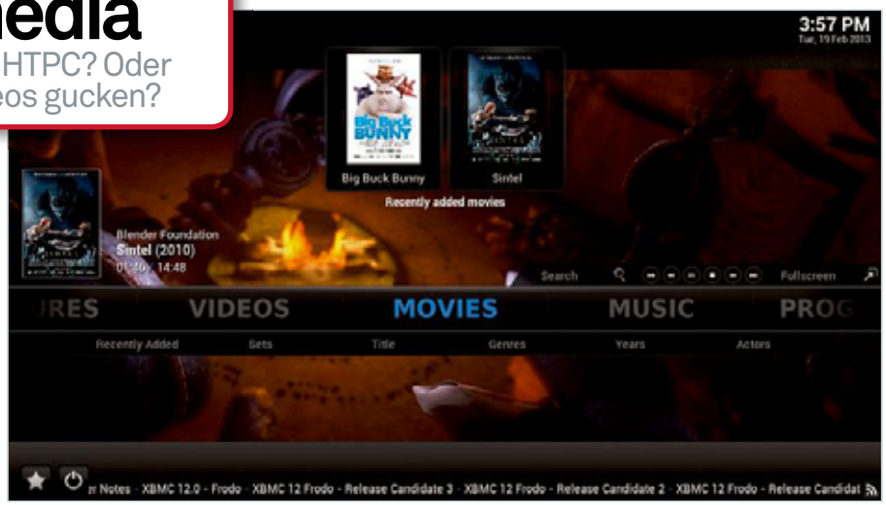
Wir stellen Ihnen 20 tolle Anwendungen, Werkzeuge und Pakete vor, die Ihnen dabei helfen, den Raspberry Pi optimal zu nutzen.

Sie haben nun also Ihren neuen Raspberry Pi zu Hause, Sie haben alles eingerichtet und sich entschieden, welche Distribution Sie als Betriebssystem verwenden möchten. Nun kommt der nächste Schritt: Finden Sie heraus, wie Sie den Minicomputer am besten nutzen, ob zum Entwickeln oder zum Relaxen. Wie wir bereits einige Male zuvor erwähnt haben, ist der Pi ein äußerst flexibles Stück Computertechnik, das sich für beides bestens eignet. Dank seines Formfaktors und geringen Energiebedarfs kann der Raspberry Pi viel einfacher daheim oder im Büro platziert werden als ein klassischer Desktop-PC.

Ob Sie ihn als Arbeitsoberfläche verwenden möchten, als Lern-Tool, als Mediacenter oder als Datei-Server – es stehen Ihnen für jeden Anwendungsfall die passenden Werkzeuge und Hilfsmittel zur Verfügung. Wir enthüllen die unserer Meinung nach 20 besten Programme, die sofort für den Raspberry Pi verfügbar sind – von der Grundausstattung wie Update-Tools und Office-Programme über Softwareentwicklungsumgebungen und Web-Schnittstellen bis hin zu Media-Streaming-Lösungen. Als Open-Source-Software können Sie alle davon leicht online finden und herunterladen. Schauen Sie einfach in Ihrem bevorzugten Paketmanager oder im Pi Store nach.

Multimedia

Ein vollwertiger HTPC? Oder einfach nur Videos gucken?

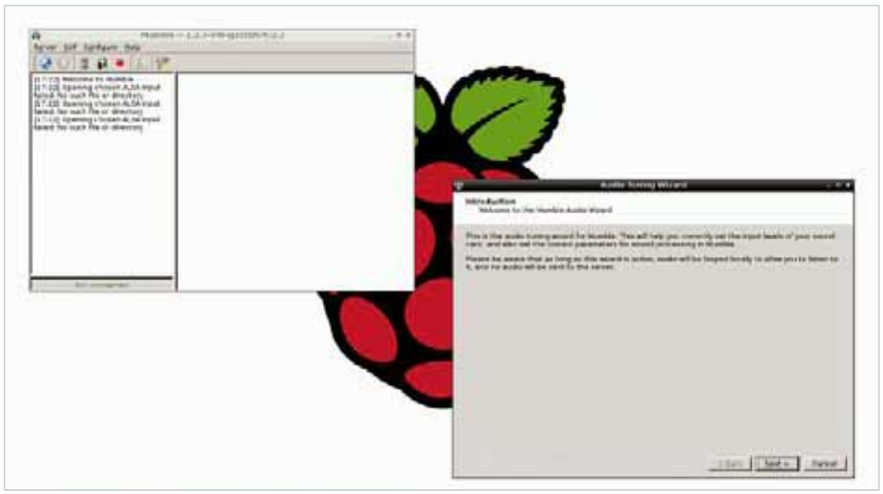


Kodi Media Center (früher XBMC)

Obwohl es so wohl nicht beabsichtigt war, eignet der Raspberry Pi sich aufgrund seines Designs exzellent dazu, als Home Theater PC (HTPC) verwendet zu werden. Er ist klein, hat einen geringen Energiebedarf und verfügt über eine Hardware-Dekodierung, die es ermöglicht, Videos mit 1080p und Audio über HDMI auszugeben. Während Streamen die einfachste Option ist, können Sie mit USB-Festplatten und TV-Tunern Gebrauch von den anderen Eigenschaften des Kodi Media Centers machen. Die aktuelle Version von Kodi beinhaltet eine spezielle Unterstützung für den Raspberry Pi, allerdings stehen schon seit einiger Zeit Builds von Kodi für den Pi zur Verfügung. Während Sie Kodi immer via Raspbian installieren können, gibt es auch dedizierte Distributionen wie RaspBMC und OpenELEC, die ein bisschen glatter laufen.

Mumble

Die Open-Source-Alternative zu TeamSpeak ist eine gute Möglichkeit, sich mit Clan-Kollegen kurzzuschließen und Strategien zu besprechen. Allerdings erfordert Mumble einen dedizierten Server, damit die eigentlichen Gespräche gehostet werden können. Obwohl Sie dafür auch Ihren PC verwenden können, möchten Sie vielleicht Zugriff auf alle verfügbaren Ressourcen haben und die Kommunikation auf ein externes System auslagern. Was aber, wenn man keinen Platz für einen vollständigen Server oder zusätzlichen Computer hat? Hier bietet der Rasberry Pi wieder mal eine tolle Alternative. Dank seiner Größe kann er fast überall untergebracht werden und Ihren Team-Chat managen.



Clementine

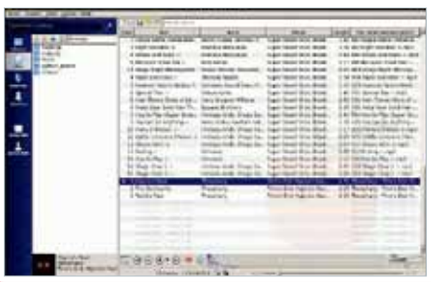
Während Clementine ursprünglich eine Portierung von Amarok war, ist es inzwischen als unabhängige App verfügbar, die Musik abspielt und organisiert. Die Software besteht aus zahlreichen Features, die Ihr Hörerlebnis verbessern können. Clementine ist ein recht schlankes Programm und zieht somit die Leistung des Pi nicht in den Keller, wenn Sie mit ihm Musik hören. Die Anwendung unterstützt das Streamen von Internetradio und kann auch dazu verwendet werden, portable Media-Player zu verwalten, was sie zum perfekten Paket für Audiobegeisterte macht.

VLC

Der Raspberry Pi hat überraschende Qualitäten in puncto Medienwiedergabe, und VLC ist vermutlich der beste Media-Player, um vollen Gebrauch davon zu machen. Einfach, klein und in der Lage, so gut wie jeden Dateityp abzuspielen, ist VLC gut dafür geeignet, die Fähigkeiten Ihres Raspberry Pi über die Funktion einer Arbeitsstation hinaus zu erweitern.

Subsonic

Eine andere Verwendung für einen Raspberry-Pi-Server ist es, Ihre Audiodateien über das Internet an jeden beliebigen Ort zu streamen. Subsonic ist auf dem Pi verfügbar und funktioniert am besten mit großen Musiksammlungen. Es verwendet eine eigene Web-Oberfläche, kann auf mehrere Player gleichzeitig streamen und unterstützt alle gängigen Audio-Dateiformate.



Entwicklung

Mit diesen Tools können Sie Ihre eigene Software entwickeln.

Eric Python IDE

Eric ist eine Alternative zum allgegenwärtigen IDLE für die Python-Entwicklung. Es ist in der Qt-Bibliothek erstellt und nutzt die Scintilla-Editor-Steuerung. Neben den Funktionen, die auch IDLE mitbringt, besitzt Eric einen integrierten Debugger. Es hat sogar ein Plugin-System, um die Software leicht erweiterbar zu machen, und es kann verwendet werden, um in Ruby zu programmieren. Eric funktioniert sowohl mit Python 2 als auch mit Python 3.

Pygame

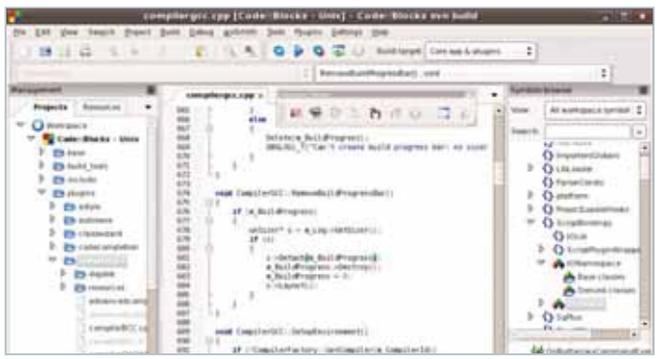
Sie kennen sich bereits bestens mit Scratch aus und wollen immer noch ein paar Spiele erstellen? Python ist eine leicht verständliche Sprache, und das Pygame-Modul erleichtert die Entwicklung von Python-Spielen. Wir haben es in einigen Tutorials verwendet, um ein Tic-Tac-Toe-Spiel zu kreieren, aber es kann für so vieles mehr verwendet werden.

Scratch

Eines der Hauptziele der Raspberry-Pi-Schöpfer war es, Kindern das Programmieren beizubringen, und einer der ersten Schritte dieses Prozesses ist es, ihnen dabei zu helfen, die grundlegende logische Struktur, die allen Programmiersprachen innewohnt, zu verstehen. Scratch bricht Codes auf das Wesentliche herunter, um einem zu ermöglichen, logische Blöcke in einer grafischen Oberfläche zu arrangieren, und damit simple Videospiele zu kreieren. Diese können auf der Scratch-Website mit der Community geteilt werden können. Die Software ist sehr gut dazu geeignet, Kinder für das Coden zu begeistern.

Code::Blocks

Obwohl es in erster Linie als IDE für die C++-Entwicklung geschaffen wurde, kann Code::Blocks auch für die Codierung einer Vielzahl anderer Typen von Apps und Programmen in ARM, D, DirectX, Fortran, GTK+, Qt, wx und noch vielen mehr verwendet werden. Es bietet auch Unterstützung für multiple Compiler und Projekte, die auf allen Plattformen angewendet werden können – sogar FreeBSD. Code::Blocks ist ein fantastischer Editor, der allen, die an seriösem Programmieren interessiert sind, einen guten Einstieg ermöglicht.

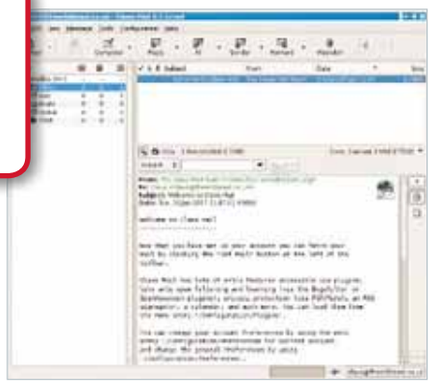


Desktop

Arbeitsoberflächen für jeden Bedarf.

Midori

Midori ist ein sehr schlanker Webbrowser, der in Pi-Distributionen oft standardmäßig verwendet wird und seinen Dienst nicht schlechter verrichtet als Firefox oder Chrome. Er ist zwar weder so erweiterungsfähig wie diese Browser, noch hat er die gleichen Synchronisierungs-Funktionen, aber er ist WebKit-basiert und entspricht so den aktuellen Webstandards. Midori sieht auch vom Design her sehr modern aus.



Claws Mail

Während in puncto E-Mail die für den Raspberry Pi ressourcenfreundlichste Alternative wohl die Verwendung eines Webmail-Dienstes wie Gmail wäre, können aber auch Desktop-Mailprogramme benutzt werden. Claws Mail ist eine Alternative zu Thunderbird, die sehr einfach funktioniert und trotzdem alle Features mitbringt. Es bietet Unterstützung für mehrere Accounts, Adressbücher, alle Arten von Protokollen und verfügt sogar über ein eigenes Plugin-System.

LibreOffice

Mittlerweile steht die wohl beste Open-Source-Office-Anwendung auch für den Raspberry Pi zur Verfügung. Die fantastische Suite aus professioneller Textverarbeitung, Tabellenkalkulation, Präsentationssoftware und Datenbankanwendung läuft ohne Probleme und stellt eine großartige Möglichkeit dar, die Funktionalität eines Desktop-Pi zu erweitern. Alles ist plattformübergreifend und wird noch besser, wenn erst die Online-Zusammenarbeit möglich ist.

Tools

Gestalten Sie Ihr Leben mit dem Pi noch einfacher.

Pi Store

Der Pi Store ist eine Mischung aus Web-Store und ein Software-Center, die sowohl kostenlose als auch kostenpflichtige Programme anbietet. Zur Verfügung gestellt wird die Plattform von der Raspberry Pi Foundation in Kooperation mit IndieCity. Der Pi Store ist nicht nur eine Quelle für Apps und Hilfsprogramme, sondern ermöglicht es Ihnen überdies, Ihre eigenen Werke zu veröffentlichen und mit anderen Pi-Usern zu teilen.

ImageWriter

In Ubuntu gibt es ein Image- und ISO-Schreibwerkzeug namens ImageWriter als Alternative zu dd. Damit können Sie beispielsweise ein Raspbian-Image auf eine SD-Karte überspielen. ImageWriter ist ziemlich einfach in der Benutzung: Laden Sie die Distribution herunter, die Sie verwenden möchten, wählen Sie das gewünschte Image, und geben Sie dann die Zielkarte an. Stellen Sie vorher sicher, dass Sie alle Dateien gesichert haben, denn die SD-Karte wird beim Überspielvorgang komplett überschrieben.

rpi-update

Ab und zu braucht der Raspberry Pi ein Firmware-Update. Dies kann ein kniffliger Prozess sein, wenn man nicht sicher ist, wie es genau funktioniert. Hier kommt das dedizierte Tool rpi-update von Hexxeh ins Spiel. Alles, was Sie tun müssen, ist, es auf einen Pi herunterzuladen, es ausführbar zu machen und dann laufen zu lassen. rpi-update beinhaltet außerdem einige zusätzliche Funktionen für fortgeschrittene Firmware-Updates.

Fedora ARM Installer

Während es ursprünglich für die Installation von Fedora-ARM-Images via Fedora entwickelt wurde, ist das Tool auch für andere Linux-Distributionen verfügbar und wird Ihnen ermöglichen, Images zu installieren, die Sie bereits bezogen haben. Die Oberfläche ist recht einfach und kann Fedora-Images für Sie herunterladen. Sichern Sie alle relevanten Dateien, die noch auf der SD-Karte sind, bevor Sie das Image installieren.

Helfer

Erweitern Sie die Funktionalität des Pi.

Synergy

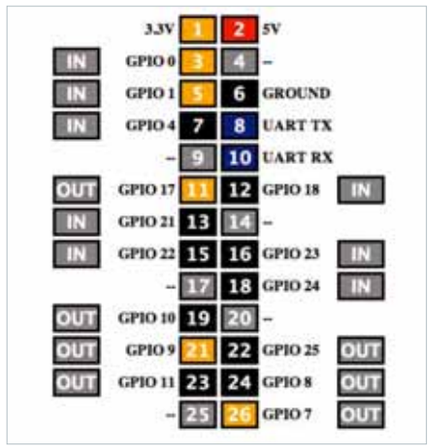
Der Raspberry Pi hat sehr wenige I/O-Ports, das Modell A sogar nur einen USB-Anschluss und nicht einmal irgendeinen Netzwerkanschluss. Das Hinzufügen von unterstützenden USB-Hubs, um die Anzahl der Systemanschlüsse zu erhöhen, verringert jedoch den Größenvorteil, den der Pi hat. Hier springt Synergy ein! Es handelt sich dabei um einen virtuellen KVM-Server, der Ihnen ermöglicht, die Tastatur und die Maus eines anderen Systems über das Netzwerk mitnutzen zu können, um so zum Beispiel die beiden USB-Anschlüsse auf dem Modell B freizuhalten.

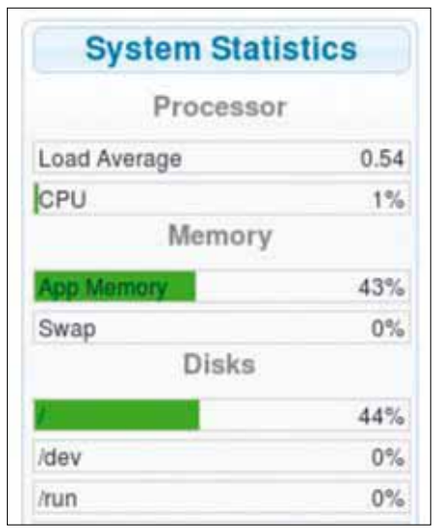
rTorrent

Wenn Sie einen Pi in irgendeiner Weise als Datei-Server fahren, dann möchten Sie vielleicht einen befehlszeilenbasierten Torrent-Client wie rTorrent verwenden, um sich nicht auf die reine Ressourcennutzung zu beschränken, sondern auch mithilfe einer SSH-Verbindung die Verwaltung einfach zu gestalten. Bei rTorrent können Sie sogar eine Auswahl von webbasierten Frontends nutzen. Es gibt jede Menge Automatisierungsoptionen wie das Aufgreifen von Torrent-Dateien aus spezifischen Ordnern oder die Selbstreinigungsfunktion.

WebIOPi

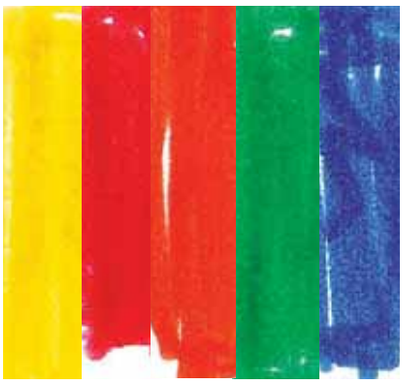
Aufgebaut auf dem REST-Framework, ist WebIOPi eine Web-App, die – einmal auf dem Raspberry Pi installiert – es Ihnen ermöglicht, die GPIO-Pins über einen beliebigen Browser auf irgendeinem Gerät im Netzwerk zu steuern. Die Oberfläche von WebIOPi ist grafisch und gestattet es, den Status eines GPIO-Pins ganz einfach per Mausklick zu ändern.





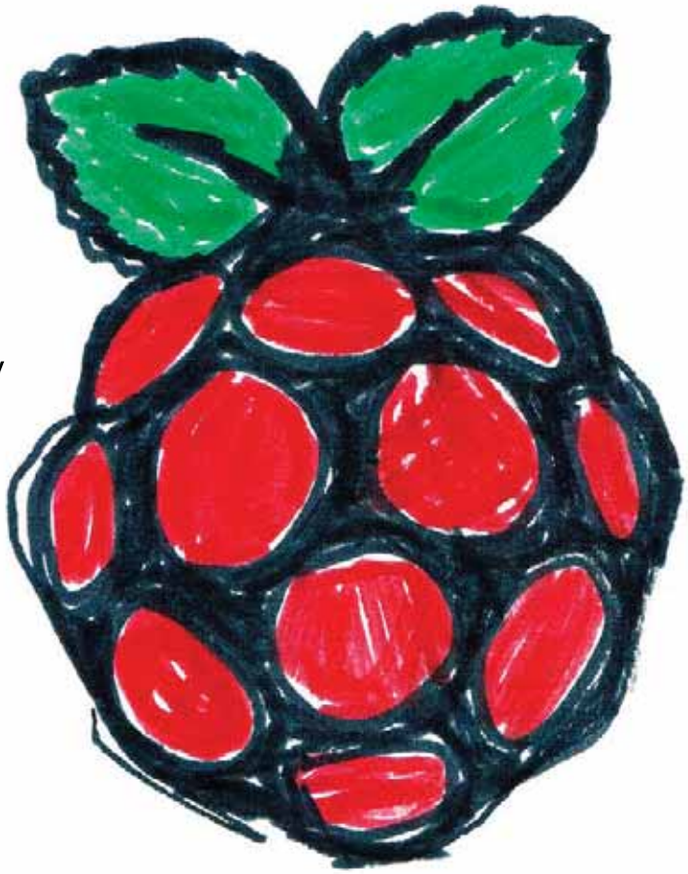
Asterisk

Mit Asterisk verwandeln Sie den Raspberry Pi in einen Kommunikationsserver für VoIP-Telefonanlagen und Konferenzschaltungen. Eine spezielle Portierung für den Pi ist erhältlich. Asterisk läuft allerdings weit besser auf dem Modell B mit 512 MB. Außerdem sollte eine SD-Karte mit 4 GB Speicherplatz verwendet werden.



Raspberry Pi für Kids

Eines der Hauptmotive, den Raspberry Pi überhaupt zu entwickeln, war es, eine Plattform für Bildungszwecke zur Verfügung zu stellen. Wie führen Sie den Raspberry Pi an Kinder heran? Joey Bernard recherchiert...



Ein Grundgedanke hinter der Entwicklung des Raspberry Pi war es, eine zugängliche Plattform zur Verfügung zu stellen, um Schülern zu helfen, mehr über Informatik und Programmierung zu erfahren. Während es sicher sinnvoll ist, Schüler an das Universitäts- und Oberstufenniveau heranzuführen, würden viele Menschen aber wohl auch zustimmen, dass es noch besser wäre, wenn man noch jüngere Leute dafür interessieren könnte zu lernen, wie Computer funktionieren.

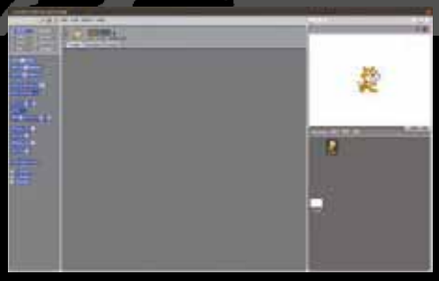
Zu diesem Zweck sind mehrere Projekte verfügbar, um Kindern beim Einstieg behilflich zu sein. Viele von ihnen verwenden Standardsprachen wie Ruby oder Python und packen sie einfach in eine grafische Oberfläche, um den Umgang damit einfacher zu gestalten. Es gibt auch völlig gesonderte Projekte, die ein System bereitstellen, das es Kindern erlaubt, komplette Programme zu basteln, ohne eine einzige Zeile Code selber schreiben zu müssen. In diesem Artikel schauen wir uns zwei dieser Projekte etwas genauer an: Scratch und KidsRuby.

Beginnen wir mit KidsRuby. Eine der Sprachen, die in den letzten Jahren in der Beliebtheit gewachsen sind, ist Ruby. Es wurde ursprünglich als eine Sprache für vernetzte Anwendungen geschrieben, deshalb wurden komplexe Eingabe/Ausgabe-Operationen zu einfach anzuwendenden Funktionen zusammengefasst. Das klassische Beispiel dieser Zusammenfassungen ist die Möglichkeit, einen ganzen Webserver in weniger als 20 Codezeilen zu schreiben. Andere Funktionen wurden geschrieben, um andere komplexe Aktionen auch kurz zusammenzufassen, zusätzlich können Sie jederzeit andere Funktionsbibliotheken (gems) installieren. All das macht Ruby zu einer sehr verlockenden Sprache, die dafür geeignet ist, Kindern das

Scratch für Kinder

Eine vollständige Umgebung für die Spieleentwicklung

Scratch ist eine Programmiersprache, die Kinder verwenden können, um Spiele und interaktive Stories zu erschaffen. Sie ist in zwei Bereiche unterteilt. Sie beginnen mit dem Download eines Clients auf Ihren Desktop, der für Linux, Windows und OS X verfügbar ist. Ist er heruntergeladen, können Sie Ihr Meisterwerk eines Spiels kreieren. Nachdem Sie es fertig codiert haben und bereit sind, können Sie es online über die Scratch-Website teilen. Das Erstellen Ihres Programms ist so einfach wie das Auswählen von Icons, das Darstellen von Handlungen oder Steuerungen und das Zusammenstellen dieser in einer Reihenfolge, sodass eines nach dem anderen ausgeführt wird. Dies lässt eine junge Person die Steuerungs- und Interaktionsgrundlagen erfahren, ohne die Syntax irgendeiner besonderen Sprache lernen zu müssen. Sogar mit solchen Grundstrukturen lassen sich relativ komplexe Programme erstellen. Scratch kann eine solide Einführung in die Ideen hinter der Programmierung zur Verfügung stellen ohne die Last, eine neue Sprache lernen zu müssen.



Vergessen Sie nicht, Ihren Sprites neue und interessante Kostüme zu geben.

Grundelemente, um ein Scratch-Programm zu schreiben.

„Sie können den Code schreiben und ihn sofort testen."

Programmieren beizubringen. Mit nur wenigen Codezeilen können sehr hoch entwickelte Programme relativ schnell geschrieben werden. Das KidsRuby-Projekt versucht, es noch leichter zu machen. Es stellt einen kompletten Arbeitsbereich zur Verfügung, wo Sie den Code schreiben und ihn sofort erproben können.

KidsRuby

Es gibt mehrere Optionen, KidsRuby zu bekommen. Wenn Sie Windows oder OS X verwenden, gibt es verfügbare Installer, die Ihnen beim Start helfen. Es ist außerdem als ISO-Abbild einer Live-CD auf Basis von Ubuntu erhältlich. Das ISO-Abbild hat Ruby und den Editor von KidsRuby startklar zur Verfügung. Wenn Sie den Editor auf Ihrem Raspberry Pi verwenden möchten, können Sie zum Download weitergehen und nur den Editorteil von GitHub herunterladen. Sie werden auch Ruby auf Ihrer SD-Karte installieren müssen. Mit beiden können Sie KidsRuby mit „ruby main.rb" starten. Wir werden die ISO-Version betrachten, um ein Gefühl dafür zu bekommen, wie der Editor arbeitet.

Wenn Sie das erste Mal KidsRuby starten, öffnet sich rechts ein leeres Editorfenster zusammen mit der Hilfeseite von KidsRuby. Die ersten zwei Elemente im Hilfsfenster sind Instruktionen, wie man KidsRuby verwendet, und ein Wörterverzeichnis von Begriffen für Leute, die ganz neu in der Informatik sind. Dazu gibt es Tutorials von Hackety Hack und Ruby4Kids. Diese Tutorials sind umgeschrieben worden, sodass sie besser mit dem von KidsRuby zur Verfügung gestellten Grundgerüst zusammenpassen; sie wurden für eine leichte Navigation geschrieben, und wenn man sich durcharbeitet, erhalten Sie einiges an Informationen. Das Verwenden des Editors wird so leicht und intuitiv wie möglich gemacht. Sie klicken einfach in das Haupteditorfenster und beginnen, Ihren Ruby-Code zu tippen. Das Ausführen Ihres Codes ist einfach, Sie müssen nur den Run-Button unten auf dem Bildschrim drücken. Um Ihr Werk zu sehen, müssen Sie den Output-Reiter im rechten Fenster auswählen. Wenn Sie es nicht tun, wird es KidsRuby in den Vordergrund rücken, sobald es erforderlich ist. Das klassische „Hallo Welt!"-Programm geht so:

```
puts "Hallo Welt!"
```

»

Ziemlich leicht, oder? KidsRuby stellt auch einen grafischen Bereich – genannnt Turtle – zur Verfügung. Mit Turtle können Sie veranlassen, dass Ihr Programm unter Ihrer Kontrolle alles mögliche zeichnet. Angenommen, Sie möchten den Hintergrund einfarbig färben. Sie würden das mit diesem Code tun:

```
Turtle.draw do
 background maroon
end
```

Dieser kleine Codeabschnitt lehrt Konzepte von objektorientierter Programmierung (Turtle.draw), Schleifen (do ... end) und Funktionsaufrufen (background maroon). Somit können Kinder mit einfachen Beispielen anfangen, einige ziemlich komplexen Konzepte zu erlernen und einen hochentwickelten Code zu schreiben. Das Hinzufügen von Zeilen zu Ihrem Output ist so einfach wie das Aufrufen anderer Funktionen, um Ihrem Turtle-Bereich weitere Anweisungen zu erteilen. Zum Beispiel:

```
Turtle.draw do
 background lightslategray
 forward 50
 turnright 90
 forward 50
end
```

Dieses Programm wird einen kleinen rechten Winkel auf Ihrem Bildschirm zeichnen.

Sie können auch Ihre Arbeit in einer Datei speichern, sodass Sie sie später wiederverwenden können. Durch Klicken auf die Speichern-Taste wird ein Dialogfeld geöffnet, in das Sie einen Dateinamen eingeben können, um Ihren Code abzuspeichern. Die Dateiendung .rb wird automatisch in der Eingabezeile hinzugefügt. Sie können die Datei wiederfinden, indem Sie den Öffnen-Button klicken. Dadurch wird ein Dialogfeld geöffnet, das alle .rb- Dateien in Ihrem Arbeitsverzeichnis anzeigt

Scratch

Während KidsRuby eine grafische Umgebung ist, die in eine traditionelle Sprache verpackt ist, in diesem Fall Ruby, liegt Scratch ein völlig anderes Konzept zugrunde. Es ist eine vollständige Entwicklungsumgebung für Spiele. Entwickelt vom Massachusetts Institute of Technology (MIT), versucht Scratch, eine grafische Umgebung zu bieten, innerhalb derer Sie Ihr Programm erstellen. Algorithmuselemente werden als Objekte visualisiert, und Sie erstellen Ihr Programm dadurch, dass Sie diese Elemente in einer bestimmten Anordnung anlegen. Es gibt Installationspakete, die für OS X, Windows sowie Debian-basierte Linux-Systeme verfügbar sind. Das schließt die Standard-Distribution ein, die auf dem Raspberry Pi läuft.

Wenn Sie Scratch das erste Mal öffnen, erhalten Sie einen Arbeitsbereich mit einem Hauptfenster, in dem Sie Ihr Programm erstellen können. Auf der linken Seite finden Sie alle verfügbaren Programmelemente. Diese werden in Kategorien wie Bewegung, Kontrolle und Sensorik eingeteilt. Das Klicken auf die Kategorieüberschrift wird alle Elemente dieses Bereichs aufzeigen. Die Ausgabe Ihres Programms wird dann auf der rechten Seite angezeigt. Da Scratch auf Spieleentwicklung und interaktives Storytelling ausgerichtet ist, hat der Output die Form von Grafiken an.

Um mit dem Schreiben eines Spiels zu beginnen, wählen Sie das erste Element mit Ihrem Mauszeiger aus, klicken darauf und ziehen es in das Hauptfenster. Sie können mehr Funktionalität hinzufügen, indem Sie andere Elemente anklicken und ziehen und sie mit den bereits vorhandenen verbinden. Einige der Elemente haben Parameter, die in Ihrem Programm verändert werden können. Zum Beispiel akzeptiert das Element „Move forward“ eine Anzahl Pixel, um die sich bewegt wird. Sobald Sie also eines in Ihrem Code platzieren, können Sie in die Textbox klicken und die vorgegebenen 10 Pixel abändern. Um Interaktion hinzuzufügen, können Sie ein Element anfügen, um Ihren Code laufen zu lassen, wenn die grüne Fahne gedrückt ist. Wenn Sie das tun und dann rechts auf die grüne Fahne klicken, folgt Ihr Sprite diesen Anweisungen und geht die definierte Anzahl von Pixeln vorwärts. Sie können relativ leicht komplexe Handlungsweisen in Ihr Programm eingeben.

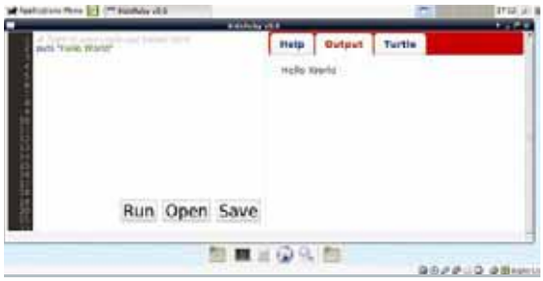
Sie können mehr Sprites hinzufügen oder denjenigen editieren, mit dem Sie starten, indem Sie auf den Reiter Kleidung im Hauptfenster klicken. Sie können sogar ein Bild mit der Kamera

■ Laden Sie KidsRuby von der Website herunter.

■ KidsRuby hat zum Ziel, eine einfachen, aber umfassende Umgebung zur Verfügung zu stellen, damit Kinder ihre eigenen Programme in Ruby codieren können.

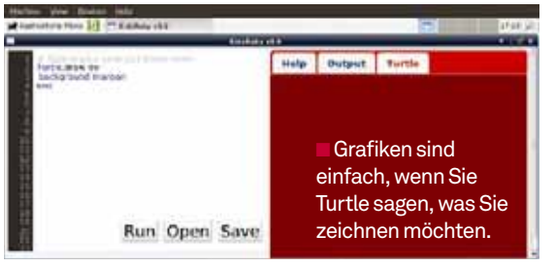
Python

Nach unserer bescheidenen Meinung ist Python eine tolle Programmiersprache für Einsteiger. Python bietet viele Extramodulen, und seine Syntax hilft dabei, einen gut strukturierten Programmcode zu erstellen. Python ist gleichzeitig eine der Sprachen, die am häufigsten auf dem Raspberry Pi verwendet werden. Die bei Python4Kids zur Verfügung gestellten Tutorials sind für Kinder ab acht Jahren ausgelegt, womit sie dazu neigen, kürzer zu sein und nur ein einzelnes Thema auf einmal zu behandeln. Sie sollten in der Lage sein, sie in 10-15 Minuten abzuhandeln. Es gibt auch Links zu anderen guten Informationsquellen von Python sowohl für Kinder als auch für Erwachsene. Auch gibt es Ansätze eines Index zu allen Tutorials, aber Sie werden wahrscheinlich eine Suche starten müssen, um ein spezielles Thema zu finden. Andernfalls fangen Sie am Anfang an und erarbeiten Sie sich Ihren Weg hin zum Python-Guru.



■ "Hallo Welt!" ist in KidsRuby ein simpler Einzeiler.

■ Sie können Turtle sogar eine komplexe Reihe von Richtungen vorgeben und es wird Ihnen folgen.



■ Grafiken sind einfach, wenn Sie Turtle sagen, was Sie zeichnen möchten.

„Sie können Ihr fertiges Projekt hochladen."

Ihres Computers aufnehmen und es als Ihren Sprite verwenden. Um Töne hinzuzufügen, die von Ihrem Code verwendet werden können, klicken Sie einfach auf den Sound-Reiter im Hauptfenster und importieren Sie entweder Sound-Dateien oder nehmen Sie mit dem Mikrofon Ihres Computers neue auf. Das System ist insgesamt darauf ausgelegt, so intuitiv wie möglich zu sein, damit Kinder die Konzepte des Programmierens begreifen können. Es ist ein komplettes Bedienungshandbuch als PDF-Download verfügbar. Es gibt auch einen kompletten Bereich, der Auskunft für Pädagogen gibt, die Scratch im Klassenzimmer verwenden möchten.

Ihr Projekt teilen

Sobald Sie Ihr Spiel oder Ihre interaktive Story fertiggestellt haben, können Sie es auf der Festplatte speichern, indem Sie oben auf das Disketten-Icon klicken. Viel interessanter ist jedoch, dass Sie Ihr fertiges Projekt auf die offizielle Scratch-Website des MIT laden können. Sie müssen dazu nur einen Account auf der Website erstellen, um sich einzuloggen und Ihr Projekt hochzuladen. Sobald es hochgeladen ist, können andere Ihre Sprites und Skripte herunterladen und sie in ihrem eigenen Scratch-Projekt ausführen. Wenn ein anderer Nutzer das Java-Plugin für seinen Browser installiert hat, kann er sogar Ihr Projekt direkt von der Scratch-Website ausführen. Sie können mit anderen über Foren kommunizieren, und jedes Projekt hat auch einen Kommentarbereich.

Wir hoffen, Sie haben entdeckt, wie viele Möglichkeiten bestehen, die dabei helfen, Kinder an das Programmieren heranzuführen. Einer der Hauptgründe für die Entwicklung des Raspberry Pi war es, die Hardware in die Hände von Kindern zu geben, sodass sie etwas haben, womit sie arbeiten können. Mit diesem Artikel sollten Sie eine Vorstellung von der Software bekommen und was alles verfügbar ist, um Kindern diese verrückte Welt der Code-Raufereien nahezubringen. Spornen Sie Ihren Programmierernachwuchs an und bringen Sie ihn dazu, völlig neue Konzepte und Programme zu erstellen. Die neuesten Ideen entstammen sicherlich den jüngsten Köpfen.

Weiterführende Literatur

■ **elinux.org/Rpi_Education**
Die Wiki-Hauptseite für das Raspberry-Pi-Projekt. Mit Links zu Bildungsinformationen und -projekten.

■ **kidsruby.com**
Ein für Kinder entwickelter Ruby-Bereich. Sie werden im Hanumdrehen zu Programmierern.

■ **scratch.mit.edu**
Das Scratch-Projekt des MIT lässt Sie Spiele und mehr entwickeln.

■ **pygame.org**
Eine Reihe von Python-Modulen, die für die Spielentwicklung geschaffen wurden.

■ **computingatschool.org.uk**
Das Projekt „Computer an Schulen nutzen" für die Entwicklung eines Lehrplans.

■ **python4kids.wordpress.com**
Tutorials, die Kindern dabei helfen, Python zu erlernen.

■ **teampython.wordpress.com**
Lernen, wie man in Python programmiert.

Bildungsangebote

Die erste Anlaufstelle für jeden, der an der Nutzung des Raspberry Pi zu Bildungszwecken interessiert ist oder Kinder an die Programmierung heranführen möchte, sollte der Bildungsbereich auf dem Raspberry-Pi-Wiki sein (elinux.org/Rpi_Education). Hier finden Sie Links, die Sie auf Programme an Universitäten und Schulen verweisen, die den Pi verwenden. Sie finden dort auch Links zu einer Vielzahl von Programmiersprachen, Communitys, Softwareplattformen und -bibliotheken. Es gibt Links zu traditionellen Sprachen wie Ruby und Python sowie zu anderen spezialisierten Sprachen wie Scratch und E-Toys.

Coden mit Minecraft Pi

So gelingt der Einstieg in Minecraft Pi – die perfekte Umgebung, um auf dem Raspberry Pi Programmieren in Python zu lernen.

Minecraft ist ein revolutionäres Sandbox-Game, in dem der Protagonist Steve die Aufgabe hat, in einer riesigen, eckigen 3D-Welt – größer noch als die Erde – zu überleben und zu prosperieren. Jeder Meter dieser Welt entspricht einem Block. Ein Block wiederum kann verschiedenste Gegenstände bergen, und je tiefer Sie graben, desto seltenere Dinge finden Sie. Sie müssen also graben, bauen und angesichts einer Armee nachtaktiver Monster Ihre Schätze weise einsetzen, um zu überleben.

Bei Minecraft Pi fällt das Überleben leichter, da es lediglich eine Art Lego-Modus beinhaltet, in dem Sie nahezu unerschöpfliche Rohstoffe und freie Hand beim Bauen haben. Minecraft Pi ist gleichzeitig die einzige Version des Spiels (welches inzwischen auf fast allen aktuellen Spieleplattformen verfügbar ist), in dem Spieler aktiv dazu aufgefordert werden, die Spielwelt mittels einer einfachen Programmierschnittstelle (API) zu hacken. In dieser Anleitung möchten wir deren Grundsätze vermitteln.

01 Minecraft Pi installieren

Zunächst müssen Sie Minecraft Pi auf Ihrem Raspberry Pi installieren – die vollständige Online-Anleitung hierzu finden Sie unter **bit.ly/1mgAj7S**. Wenn Sie erprobt im Umgang mit Tar-Archiven sind, können Sie die Detail-Anleitung überspringen, einfach die Datei von **bit.ly/1jpZ17Z** herunterladen und in Ihr Home-Verzeichnis entpacken.

02 X muss laufen

Bevor Sie Minecraft Pi nutzen können, müssen einige Voraussetzungen erfüllt sein. Ein bekannter Stolperstein ist, dass Minecraft Pi nicht direkt aus dem Terminal startet. Wenn Ihr Pi nicht schon entsprechend konfiguriert ist, müssen Sie zunächst mit **startx** die Desktopumgebung aufrufen. Öffnen Sie dann im Terminal den Ordner mcpi und starten Sie das Spiel mit diesem Befehl:

```
./minecraft-pi
```

Minecraft Pi bietet nicht den Nervenkitzel unterirdischer Höhlen und fieser Mobs, aber dafür dürfen Sie nach Belieben das Spiel hacken.

03 Desktop-Verknüpfung

Das Spiel im Terminal zu starten, behagt nicht jedem. Hinzu kommt noch, dass ein zweites Terminalfenster nötig ist, um Minecraft-Pi-Skripte aufzurufen. Erstellen Sie eine Verknüpfung zum Spiel auf dem Desktop, um diese Terminal-Doppelung zu vermeiden. Öffnen Sie das Terminal und geben Sie ein:

```
cd ~/Desktop
```

Erstellen Sie durch den Befehl **nano minecraft.desktop** eine neue Datei auf dem Desktop, und kopieren Sie Folgendes hinein:

```
[Desktop Entry]
Name=Minecraft Pi
Comment=Minecraft for the Pi
Exec=/home/pi/mcpi/minecraft-pi
Icon=/home/pi/mcpi/logo.png
Terminal=false
Type=Application
Categories=Games;Education;
StartupNotify=true
```

Mit Strg+X, Y und der Eingabetaste speichern und schließen Sie die Datei. Das Minecraft-Symbol fehlt im Ordner mcpi standardmäßig, Sie finden es per Bildersuche im Internet. Benennen Sie es in „icon.png“ um und fügen Sie es in den Ordner ein. Es muss 256 x 256 Pixel groß sein. Eventuell ist ein Neustart des Desktops nötig, bevor das Symbol angezeigt wird.

04 Ausgang per Tab-Taste

Bereits in unserer erwähnten Online-Anleitung haben wir auf das etwas knifflige Zusammenspiel zwischen Minecraft Pi und Python-Skripten hingewiesen. Minecraft Pi muss laufen, bevor Sie ein Python-Skript starten können, das mit dem Spiel interagiert. Dazu müssen Sie in Minecraft Pi die Tabulator-Taste betätigen, um den Mauszeiger zu aktivieren, mit dem Sie dann in das Terminal wechseln können, aus dem Sie Ihre Python-Skripte starten.

Sichern Sie die originalen API-Dateien in einem neuen Ordner.

05 Lesen Sie die Anleitung!

Auf den ersten Blick mag es so aussehen, als gäbe es kaum Hilfestellung beim Programmieren in Minecraft Pi – doch unter Softwarespezialisten gibt es einen Geheimtipp: Der Quellcode eines Programms ist beinahe so gut wie eine vollständige Anleitung. Sie finden beispielsweise Informationen zu allen verwendbaren Blocktypen und deren Namen im Skript block.py, das im Ordner /mcpi/api/python/mcpi liegt. Noch besser: Alle Funktionen, die Sie im Spiel ausführen können, finden Sie in der Datei minecraft.py. Sie sind sogar ausführlich (auf Englisch) kommentiert, sodass Sie direkt erfahren, dass **setPos** „set entity position“ bedeutet.

06 Hinweis zu Skripten

Unter welchem Namen möchten Sie Ihr erstes Minecraft-Pi-Skript speichern? Nun, „minecraft.py“ klingt nicht schlecht, oder? Leider können Sie mit diesem Namen Ihre Minecraft-Pi-Installation ruinieren, weil Sie damit möglicherweise die wichtigste Quellcode-Datei des Spiels überschreiben. Schauen Sie sich alle Dateien Im Ordner /mcpi/api/python/mcpi an – deren Namen sollten für Ihre Skripte tabu bleiben.

07 API in Kopie

Am besten geschützt sind die Originaldateien der Minecraft-Pi-Programmierschnittstelle (API), wenn Sie eine Kopie des gesamten Ordners erstellen und diese nutzen. Erstellen Sie dazu im Home-Verzeichnis einen neuen Ordner, öffnen Sie ihn (etwa mit **mkdir mc_projects && cd mc_projects**) und geben Sie Folgendes ein:

```
cp -r ~/mcpi/api/python/* .
```

Das Leerzeichen und der Punkt am Ende sind notwendig.

Wäre es nicht herrlich, Brücken oder die Säulengänge einer Burg automatisch generieren zu können? Mit Minecraft Pi geht das!

08 Das erste Skript

Erstellen wir also ein Testskript, um sicher zu gehen, dass die API an ihrem neuen Ort auch funktioniert. Geben Sie Folgendes in einen Texteditor ein und speichern Sie es unter „boilerplate.py“:

```
from mcpi.minecraft import Minecraft
from mcpi import block

mc = Minecraft.create()
mc.postToChat(“Minecraft API Connected”)
```

Starten Sie Minecraft Pi, falls es noch nicht läuft. Aktivieren Sie im Spiel per Tab den Mauszeiger, öffnen Sie ein Terminal und navigieren Sie zu Ihrem Minecraft-Projektordner.

Geben Sie darin **python boilerplate.py** ein und bestätigen Sie per Eingabetaste – im Spiel erscheint nun die Chatnachricht „Minecraft API Connected“. Wir empfehlen, alle Ihre Skripte mit diesem Textbaustein – auch Boilerplate genannt – zu beginnen.

09 Es hat nicht geklappt?

Erhalten Sie eine Fehlermeldung von Python, dann sehen Sie den Boilerplate-Code noch einmal nach Tippfehlern durch – in neun von zehn Fällen ist das der Grund. Wird ein Import-Fehler angezeigt, ist eventuell beim Kopieren des API-Ordners etwas schiefgegangen. Wiederholen Sie die vorangegangenen Schritte und versuchen Sie es erneut.

10 Der Spaß beginnt

Der Boilerplate-Code hat die Feuerprobe bestanden, jetzt kann der Spaß mit der Minecraft-Pi-Programmierschnittstelle und Python beginnen. Es gibt viele Möglichkeiten: auf Blöcke zugreifen und sie konfigurieren, die Position des Spielers herausfinden und verändern, Kameraperspektiven anpassen und vieles mehr. Die beliebtesten Funktionen sind sicher diejenigen, die mit der Position des Spielers experimentieren oder neue Blöcke generieren.

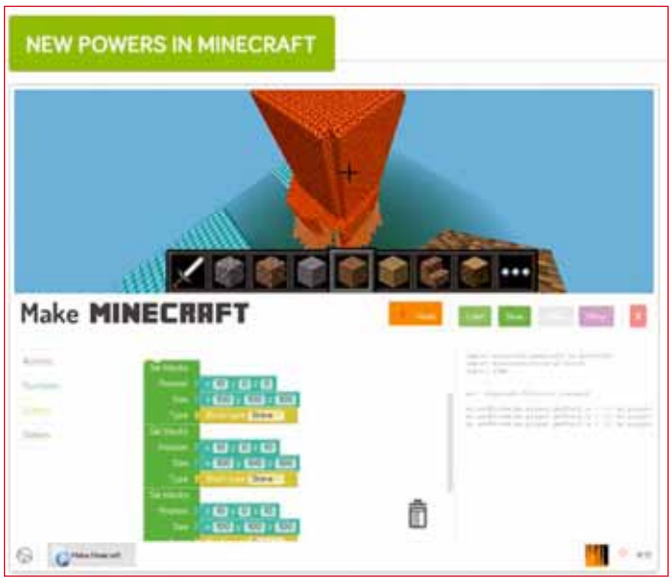
Oben Immer noch keine Schafe in Minecraft Pi, aber Brunnen durchaus.

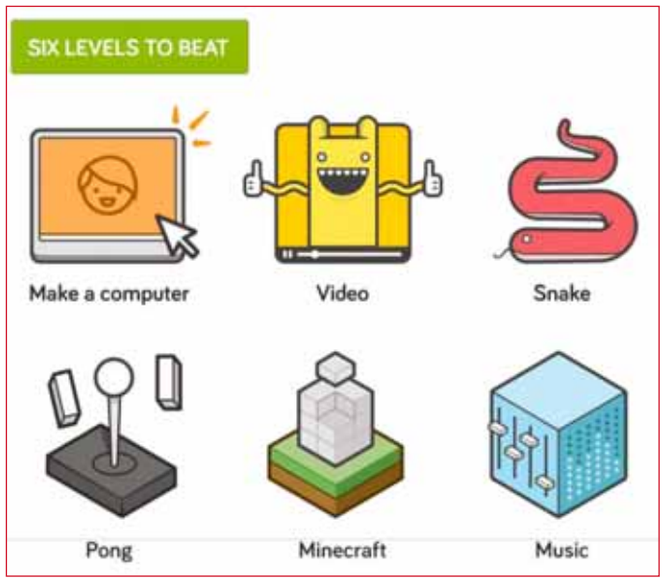
11 Das Koordinatensystem

Ohne ausreichende Kenntnisse im Umgang mit dem Koordinatensystem des Spiels ist es so gut wie unmöglich, mit Minecraft Pi etwas Sinnvolles anzustellen. Noch dazu ist das System etwas unorthodox, indem es die Y- und Z-Koordinaten im dreidimensionalen Raum dreht – bei Minecraft Pi entspricht „X“ der Seitwärtsbewegung, „Z“ der Vorwärtsbewegung und „Y“ der Vertikalbewegung. Links oben auf dem Bildschirm werden immer Ihre aktuellen Koordinaten angezeigt. Investieren Sie ein paar Minuten, um die Koordinaten lesen zu lernen und zu verstehen, wie diese sich verändern, wenn Sie sich im Spiel fortbewegen.

12 Der Startpunkt

Wenn Sie ein neues Minecraft-Spiel starten, beginnen Sie genau im Zentrum der Karte. Ihre Koordinaten sind dort 0, 0, 0. Was bedeutet das? So können auch Anfänger ganz einfach mit der Blockplatzierung beginnen. Am Startpunkt wissen Sie immer genau, wo die umgebenden Blöcke sind – sie befinden sich einfach bei plus oder minus 0.





Oben links Kano OS hat Minecraft Pi vorinstalliert und bietet spielerisches Lernen.

Oben rechts Bei Kano OS coden Sie mit Snake und Pong und machen Musik mit Sonic Pi.

Mit der Funktion setBlocks können Sie viele Blöcke gleichzeitig generieren.

13 getPos

Eine der leichtesten Übungen mit der API ist es, den eigenen Standpunkt herauszufinden. Das Wissen um den Ort, an dem der Spieler sich befindet, ist bei Minecraft Pi eine erstaunlich mächtige Gabe. Sie können damit Blöcke neben oder unter der Spielfigur erzeugen oder Ereignisse auslösen (wie etwas das Finden besonderer Orte). Im untenstehenden Beispiel wird die Position des Spielers bestimmt und im Sinne eines Funktionstests sporadisch mittels der Variablen **my_pos** im Chatfenster ausgegeben.

```
from mcpi.minecraft import Minecraft
from mcpi import block
import time

mc = Minecraft.create()
mc.postToChat("Minecraft API Connected")
while True:
    my_pos = mc.player.getPos()
    mc.postToChat("Meine Position ist:" + str(my_pos))
    time.sleep(1)
```

14 setBlock

Sehr beliebt ist auch das Anpassen von Blocktypen. Eine Liste aller Typen finden Sie im Ordner mcpi in einem Skript namens „blocks.py“. Erweitern wir nun Schritt 13 daingehend, dass die Spielfigur eine Spur eines bestimmten Blocktyps hinterlässt, während Sie umherwandern. Fügen Sie nach der Zeile „mc.postToChat“ im vorherigen Skript **mc.setBlock((my_pos), block.STONE)** hinzu, sodass die while-Schleife des Skripts folgendermaßen aussieht:

```
mc.postToChat("Pflastermodus!")
while True:
    my_pos = mc.player.getPos()
    mc.setBlock((my_pos), block.STONE)
    time.sleep(1)
```

Im Spiel wird Ihnen nun auffallen, dass die Spielfigur jede Sekunde auf ihrem momentanen Standort einen Steinblock generiert. Wenn Sie **setBlocks** nutzen, um durch Eingabe von Start- und Zielkoordinaten viele Blöcke gleichzeitig zu platzieren, werden Sie schon bald gigantische Bauten erschaffen können.

15 Steve teleportieren

Bislang wurde die Position der Spielfigur Steve nur ausgelesen, jetzt wird auch diese Variable manipuliert. Mit der Funktion **setPos** können Sie die Spielfigur zurück zum Startpunkt teleportieren, wenn Sie sich verlaufen haben:

```
from mcpi.minecraft import Minecraft
from mcpi import block
import time

mc = Minecraft.create()
mc.postToChat("Minecraft API
Connected")

mc.player.setPos(0, 10, 0)
```

Wenn Sie stattdessen die Spielfigur aus luftiger Höhe auf die Erde fallen lassen möchten, können Sie als Y-Koordinate eine große (und positive) Zahl wählen. Ein Beispiel:

```
mc.player.setPos(0, 65, 0)
```

Hier kommt Kano

Für junge Minecraft-Fans, denen das Programmieren mit Python noch zu komplex ist, gibt es ein besonderes Betriebssystem: Kano OS für den Raspberry Pi wurde speziell für jüngere Kinder entworfen. Es enthält eine angepasste Version von Minecraft Pi mit einer visuellen Programmieroberfläche. Diese macht es möglich, durch einfache Kombination einzelner bunter Codeblöcke genauso tief in die Minecraft-Welt einzugreifen, wie in dieser Anleitung gezeigt wurde. Mehr unter: **kano.me/downloads**

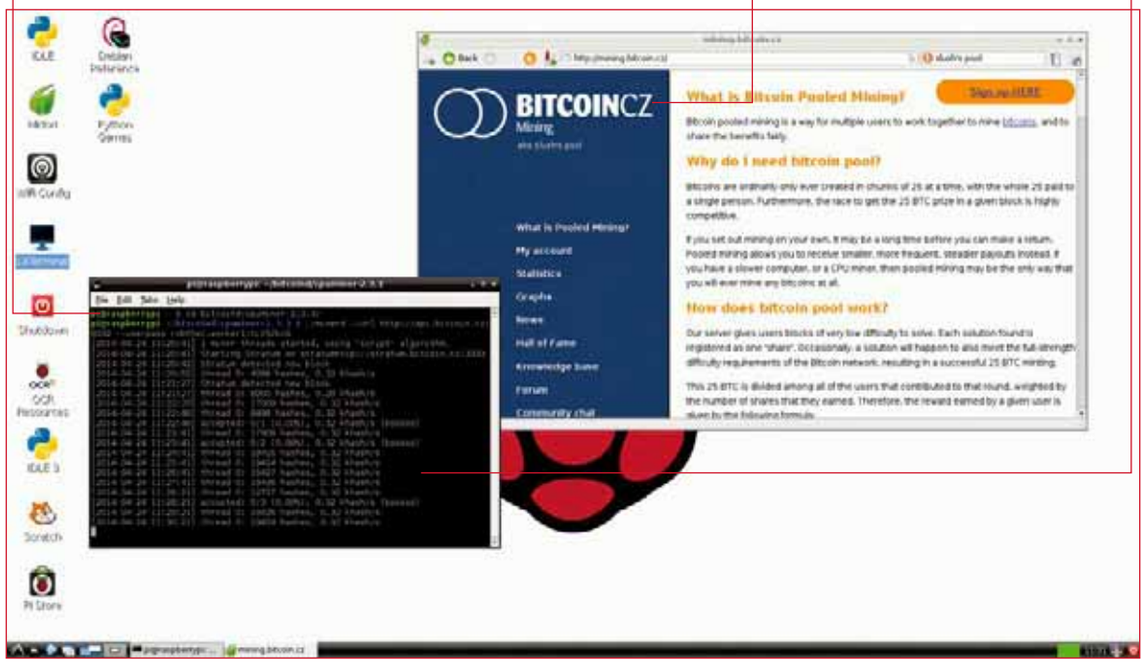
Bitcoin-Mining mit dem Raspberry Pi

Steigen Sie ins Kryptowährungsgeschäft ein: Schürfen Sie Bitcoins oder andere virtuelle Geldeinheiten mit dem Raspberry Pi.

Eigenkreation Kompilieren Sie Ihr eigenes Bitcoin-Mining-Programm, um mehr Kryptogeld zu verdienen.

Zusammenarbeit Arbeiten Sie in Mining-Pools zusammen, um gemeinsam die Chancen auf Geld zu erhöhen.

Fortschritt Mit dem Pi verfolgen Sie den Fortschritt Ihrer Arbeit nahezu in Echtzeit.



Was Sie brauchen:

- aktuelles Raspbian-Image
  raspberrypi.org/downloads
- cpuminer
  sourceforge.net/projects/cpuminer/files
- Bitcoin-Pool

Das Konzept der Digitalwährung entstand im Laufe der letzten Jahre als Reaktion auf das zentral verwaltete Zahlungssystem. Virtuelle Geldeinheiten wie der Bitcoin sind dezentrale Währungen, die nicht der Kontrolle durch Zentralbanken unterliegen. Mit einem Wertanstieg auf umgerechnet über hundert Euro hat sich der Bitcoin zu einem starken Zahlungsmittel entwickelt. Sie können Bitcoins gegen reguläres Geld erwerben oder aber mithilfe Ihres Rechners selbst „minen". Beim „Mining" stellen Sie dem Bitcoin-Netzwerk Ihre Rechenleistung zur Verfügung und tragen so zur Sicherheit des Systems bei. Dafür werden Sie in Form von Bitcoins entlohnt. Hier kommen wir ins Spiel, indem wir Ihren Raspberry Pi in eine Bitcoin-Mine verwandeln.

01 Abhängigkeiten installieren

In diesem Tutorial verwenden wir den cpuminer, um Bitcoins zu schürfen; dafür ist eine Kompilierung aus dem Quellcode nötig. Wir installieren also zunächst einige Abhängigkeiten durch Eingabe von:

```
$ sudo apt-get install gcc gcc-4.5 g++ g++-4.5
libstdc++6-4.5-dev libpcre3-dev libcurl3-dev make less
```

02 Positionen anlegen

Um uns die Arbeit zu erleichtern, erstellen wir nun ein Verzeichnis für unsere eigenen Dateien. Hierfür verwenden wir im Terminal:

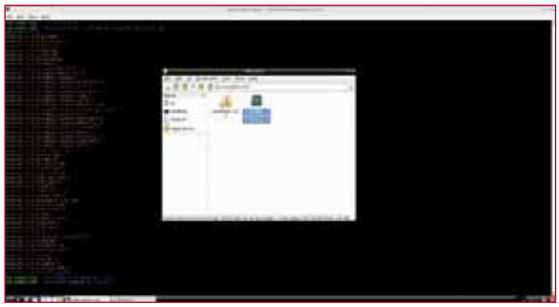
```
mkdir bitcoind
```

Ist das erledigt, so gehen wir zum neuen Verzeichnis über:

```
cd bitcoind
```

03 cpuminer herunterladen

Besuchen Sie den SourceForge-Link, um den aktuellen Quellcode des cpuminer in dieses neue Verzeichnis zu übertragen. Zum Zeitpunkt der Entstehung dieses Artikels heißt die neueste Version pooler-cpuminer-2.3.3.tar.gz.



04 Dateien extrahieren

Am Desktop können Sie die Dateien einfach aus dem Dateimanager extrahieren. Alternativ können Sie sie auch im Terminal entpacken mit:

```
$ tar -zxvf pooler-cpuminer-2.3.3.tar.gz
```

Öffnen Sie nun den neuen cpuminer-Ordner durch Eingabe von cd.

05 cpuminer kompilieren

Um den cpuminer zu kompilieren, benutzen wir die Standardbefehle ./configure und make. Beide Befehle werden etwas Wartezeit erfordern. Wenn Sie die vorherigen Schritte richtig ausgeführt haben, sollte die Programmübersetzung kein Problem darstellen.

06 Einem Pool beitreten

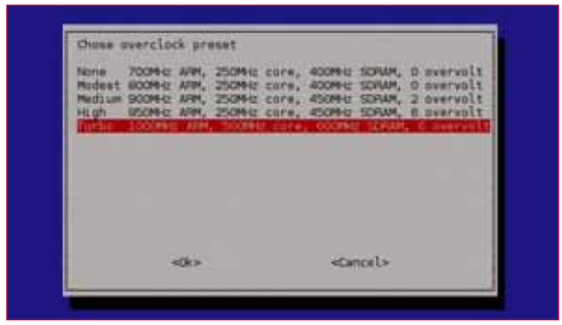
Wenn Sie auf Bitcoins aus sind, dann ist alleine schürfen ein recht aussichtsloses Unterfangen. Um Ihr Ziel zu verfolgen, sollten Sie sich also einem Mining-Pool anschließen. Melden Sie sich an, um Benutzernamen, Password und URL zu erhalten.

07 Mining starten

Kehren Sie mit dem Befehl cd wieder zum cpuminer-Ordner zurück. Geben Sie folgenden Befehl ein, um das Mining zu starten:

```
$ ./minerd --url [pool address] --userpass
[username]:[password]
```

cpuminer wird Ihnen nun Ihre Hashrate anzeigen, welche in diesem Stadium noch etwas langsam sein mag.



08 Mining beschleunigen

Wenn Sie eine höhere Hashrate möchten, dann können Sie Ihren Pi übertakten. Öffnen Sie dazu raspi-config im Terminal und nutzen Sie die Overclock-Funktion. Sie können die Arbeitsgeschwindigkeit beliebig erhöhen, bedenken Sie jedoch, dass zu starkes Übertakten Ihren Pi verschleißt.

09 Aufrüsten

Wenn Sie richtig viele Bitcoins verdienen wollen, dann müssen Sie versuchen, das Maximum aus Ihrem Pi herauszuholen. Steigern Sie Ihre Hashrate exponentiell, indem Sie USB-Bitcoin-Miner an Ihren Raspberry Pi anschließen.

Die Firmware-Dateien auf dem Raspberry Pi sind geschützt, also muss dieses Werkzeug mit Root-Berechtigungen ausgeführt werden.

Statt den Code zu kopieren, durchlaufen wir jede Firmware-Datei und ziehen einen Menüpunkt für sie.

```
#!/usr/bin/env python

# RamPi - a program written by Liam Fraser for a Linux User and
# Developer tutorial - 16/05/2012. This tool allows the user to easily
# change the amount RAM that is used by the Raspberry Pi's GPU.

# Check we have root privelages
if os.geteuid() != 0:
    print ("You need root privelages to use this tool - exiting...")
    sys.exit()

# Create a dictionary with the RAM use in Linux as the key and the
# firmware file path as the value
firmwares = {128:"/boot/arm128_start.elf",
             192:"/boot/arm192_start.elf",
             224:"/boot/arm224_start.elf"}

# Display a menu
print("\nWhat would you like to do?\n")

# Print an option for each firmware
count = 1
for firmware in firmwares.items():
    # Work out the RAM allocation of Linux and GPU
    firmwareLinux = firmware[Key]
    firmwareGPU = 256 - firmwareLinux

    print (str(count) + ". Allocate " + str(firmwareLinux) +
           " MB of RAM to Linux leaving " + str(firmwareGPU) +
           "MB of RAM for the GPU.")

    count += 1

# Add an exit option
print (str(count) + ". Exit.")

# Deal with our options (only in the range of the count as we don't
# need to handle exit)
if option < count:
    # Work out the RAM allocation of Linux and GPU
    newFirmware = firmwares.keys()[option - 1]
    newGPU = 256 - newFirmware

    print ("\nUsing the firmware which allocates " + str(newFirmware)
           + "MB of RAM to Linux and " + str(newGPU) +
             "MB of RAM to the GPU...")

    # Run the cp command with the appropriate values
    copyResult = os.system("cp " + firmwares[newFirmware] + " " +
                           "/boot/start.elf")

    if copyResult != 0:
        print ("\nThere was an error while copying the requested"
               " firmware!")
        print ("Exiting...")
    else:
        print ("\nThe requested firmware was copied successfully.\n")
```

Wir speichern eine Sammlung von Firmware-Dateien mit ihrem jeweiligen RAM-Wert als Schlüssel. Dies macht es einfach, zukünftig weitere hinzuzufügen.

Wir betreiben Systembefehle mit der Funktion „os.system“. So können wir das Ergebnis des Befehls überprüfen und eine angemessene Nachricht für den User ausgeben.

Schreiben Sie Ihre eigenen Hilfsprogramme für den Pi

Erstellen Sie ein Hilfsprogramm, das die Größe des Arbeitsspeichers ändert, der dem Grafikprozessor zur Verfügung steht.

Was Sie brauchen:

- einen Raspberry Pi mit Debian

Im Tutorial ab Seite 36 in diesem Heft ging es um die Grundlagen der Python-Programmierung auf dem Raspberry Pi. Inzwischen sind Sie vermutlich etwas sicherer im Umgang mit der Sprache. Dann wäre doch jetzt ein guter Zeitpunkt, darin ein nützliches Werkzeug zu programmieren.

Der Grafikprozessor – abgekürzt auch die GPU genannt – des Raspberry Pi ist in das Ein-Chip-System eingebaut. Die GPU ist sehr leistungsstark und in der Lage, Videos mit 1080p abzuspielen. Sie teilt RAM (Arbeitsspeicher) mit dem Betriebssystem und benötigt davon verschiedene Mengen, um unterschiedliche Aufgaben durchzuführen. Bei der Wiedergabe von detailreichen 3D-Inhalten oder 1080p-Videos benötigt die GPU zum Beispiel 128 MB von insgesamt 256 MB. Damit würden 128 MB für die Nutzung durch Linux übrig bleiben. Mehr RAM kann beim Programmieren allerdings von Vorteil sein, etwa um parallel einen Webbrowser für die Dokumentationsanzeige und die Entwicklungsumgebung öffnen zu können.

Die RAM-Menge, die die GPU verwendet, wird je nachdem, welche Version der Firmware-Datei „start.elf“ in der „/boot“-Partition vorhanden ist, eingestellt. Derzeit gibt es drei Firmware-Dateien: „arm128_start.elf“, „arm192_start.elf“ und „arm224_start.elf“. Die Zahlen in den Dateinamen repräsentieren die für Linux vergebenen RAM-Größen, wobei der verbleibende RAM der GPU zugeteilt wird. Der RAM für das Betriebssystem wird wie folgt festgelegt:

1. Entscheiden Sie, welche RAM-Größe Sie benötigen.
2. Wählen Sie die entsprechende Firmware-Datei und kopieren Sie sie in „start.elf“, um die momentan bestehende Datei zu überschreiben.
3. Starten Sie den Raspberry Pi neu.

All das ist kein großer Aufwand – aber warum stellen Sie Ihre neuen Fähigkeiten nicht auf die Probe und bauen ein Werkzeug, das Ihnen diese Arbeitsschritte abnimmt? Fangen wir an...

01 Neues Python-Projekt erstellen

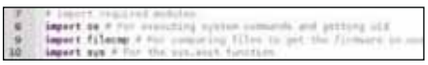
Starten Sie die IDE (integrierte Entwicklungsumgebung) Geany, indem Sie das LXDE-Menü aufrufen und „Programs“ wählen. Von hier aus wählen Sie „Geany“. Sobald Sie in der Geany-Oberfläche sind, erstellen Sie eine neue Python-Datei aus einer Vorlage, indem Sie New (with template) > main.py auswählen. Löschen Sie aus der Vorlage alles außer der ersten Zeile:

```
#!/usr/bin/env python
```

Diese Zeile ist wichtig, weil Sie so den Code aus der Befehlszeile heraus ausführen können. Die Bash-Shell weiß dann, dass er mit dem Python-Interpreter geöffnet werden soll.

02 Ihr Programm dokumentieren

Speichern Sie Ihre Arbeit, bevor Sie fortfahren. Sobald Sie dies erledigt haben, fügen Sie Ihrer Python-Datei eine kurze Erklärung bei, wofür das Werkzeug gut ist, wobei Sie den Zeilen ein Rautensymbol (#) voranstellen. Andere Programmierer werden dies zu schätzen wissen, falls Sie Ihr Werk später teilen.



03 Für Import notwendige Module

Dieses Programm nutzt die Module „os“ (Betriebssystem), „filecmp“ (Dateivergleich) und „sys“ (System). Das Modul „os“ erlaubt uns, die ID des Users zu überprüfen, der Skript und Systembefehle ausführt. Das Modul „filecmp“ ermöglicht es uns, Dateien zu vergleichen und zu ermitteln, welche Firmware-Datei gerade in Benutzung ist. Das Modul „sys“ ist erforderlich, damit wir das Programm verlassen können, bevor das Skript endet.

04 Root-Berechtigungen aktiviert?

Die Firmware-Dateien befinden sich in der „/boot“-Partition des Raspberry Pi. Die Dateien in diesem Bereich verlangen Root-Rechte, wenn Sie verändert werden sollen, damit sie von einem Standardnutzer nicht beschädigt werden können. Aus diesem Grund beginnen wir mit unserem Hilfsprogramm, indem wir sicherstellen, dass wir es als Root-User mit der Funktion „os.geteuid“ (get effective user ID) ausführen. Wenn die User-ID nicht null ist, dann läuft das Programm nicht als Root und beendet sich. Dafür stellen Sie dem Programmnamen den Befehl „sudo“ (superuser do) voran, um es als Root auszuführen.

```
# Check we have root privileges
if os.geteuid() != 0:
   print (“You need root privileges
to use this tool - exiting...”)
   sys.exit()
```

05 Eine Sammlung von Firmware-Dateien definieren

Wir werden ein assoziatives Datenfeld verwenden, um eine Liste der möglichen Firmware-Dateien zu verwahren. Das Datenfeld ist eine Struktur mit Paaren aus Schlüsseln und Werten. Das bedeutet, dass wir den Pfad der Firmware nachverfolgen können, indem der Wert eines bestimmten Schlüssels erfragt wird. Wir erzeugen das Datenfeld, indem wir ein Paar geschweifte Klammern verwenden. Die Schreibweise der einzelnen Elemente ist „key:value“, und mehrere Paare werden durch Komma getrennt. Auf den Wert kann durch Verwendung eckiger Klammern mit der Schreibweise „firmwares[key]“ zugegriffen werden. Es ist gut, eine Sammlung von Firmware-Dateien zu verwenden und nicht die Variable mit dem Pfad jeder Datei, damit später leicht eine hinzugefügt werden kann und dafür nichts mehr im Code verändert werden muss. Das wird weiter unten noch klarer.

```
# We would have exited by now if we
weren’t root, so carry on

# Create a dictionary with the RAM
use in Linux as the key and the
# firmware file path as the value
firmwares = {128:”/boot/arm128_
start.elf”,
         192:”/boot/arm192_start.
elf”,
         224:”/boot/arm224_start.
elf”}
```

06 Aktuelle Firmware finden

Für den User wäre es toll, die gerade im Einsatz befindliche Firmware zu sehen, bevor er sich entscheidet, auf eine andere Firmware zu wechseln. Dazu durchlaufen wir der Reihe nach jede Firmware-Datei und vergleichen sie mit „start.elf“. Wir loopen durch jedes Element in der Sammlung von „firmwares.items“. Der Wert der Firmware-Variablen ist eine Liste bestehend aus zwei Elementen: dem Schlüssel und dem Wert. Auf den Schlüssel kann mit „firmware[0]“ und auf den Wert (den Pfad zur Datei) mit „firmware[1]“ zugegriffen werden. Wir haben Konstanten für diese Werte erzeugt, damit der Code leichter zu lesen ist. Anstatt 0 und 1 zu verwenden, benutzen wir „Key“ und „Path“. Der False-Wert, der an die Funktion „filecmp.cmp“ geleitet wird, sorgt dafür, dass die Datei nicht „oberflächlich“ geprüft wird. Das ist notwendig, weil sich die einzelnen Dateien sehr ähnlich sind.

```
# Constants for easier reading when
getting a key or value from
# an item in the dictionary
Key = 0
Path = 1

# Work out which firmware is
currently in use by comparing each
file
# in the dictionary with start.elf
for firmware in firmwares.items():
   result = filecmp.cmp(“/boot/
start.elf”, firmware[Path], False)
```

> “Zeit, dem User ein Menü anzuzeigen.”

07 Aktuelle Firmware anzeigen

Sobald das Ergebnis eines Dateivergleichs True ist, erhalten wir vom Schlüssel des aktuellen Firmware-Elements den RAM, der Linux zugeteilt wird. Der RAM, welcher der GPU zugeteilt wird, kann durch Abziehen des aktuellen Linux-Wertes von der RAM-Gesamtmenge (256 MB) herausgefunden werden. Diese Information wird dann auf dem Bildschirm ausgegeben. Beachten Sie, wie wir die Funktion „str()“ verwenden, um die ganzzahligen Werte von „currentLinux“ und „currentGPU“ in eine Zeichenfolge zu konvertieren, da die Ausgabefunktion nur String-Typen übernimmt.

```
# When we find the correct file
   if result == True:
      # Work out the RAM
allocation of Linux and GPU
      currentLinux = firmware[Key]
      currentGPU = 256 -
currentLinux

      print (str(currentLinux) +
      “MB of ram is currently
being used by Linux.”)

      print (str(currentGPU) +
      “MB of ram is currently
being used by the GPU.”)
```

08 Ein Menü anzeigen

Inzwischen wurde dem User mitgeteilt, was seine aktuelle Firmware ist, also ist es an der Zeit, ihm ein Menü anzuzeigen, damit er entscheiden kann, was er tun möchte. Wir werden mit dem Ausgeben der Nachricht „Was würden Sie gerne tun?“ beginnen. Das „\n“ wird verwendet, um eine neue Zeile vor und hinter der Nachricht auszugeben.

```
# Display a menu
print(“\nWas würden Sie gerne
tun?\n”)
```

09 Eine Option für jede Firmware

Erinnern Sie sich, dass wir sagten, man werde neue Firmware hinzufügen können, ohne Änderungen am Programm durchführen zu müssen – abgesehen von der Modifizierung des Datenfeldes zu Beginn? Wir werden eine Schleife verwenden, die jeder Firmware im Datenfeld eine Menüoption hinzufügt, ähnlich der Schleife, die zum Vergleich der Firmware-Dateien benutzt wird. Das heißt, dass wir den Code nur einmal ändern müssen, wenn wir etwas im Menü anpassen möchten. Wir starten mit einer Zählung, damit wir unsere Optionen durchnummerieren können. Für jeden Firmware-Eintrag im Datenfeld erarbeiten wir die RAM-Größe, die Linux beziehungsweise der GPU zugeteilt wird, und geben dann diese Informationen aus – mit einer vorangestellten Zahl. Wir erhöhen dann die Zählung, damit der nächste ausgegebene Wert mit einer 2, einer 3 und so weiter beginnt.

```
# Print an option for each firmware
count = 1
for firmware in firmwares.items():
   # Work out the RAM allocation of
Linux and GPU
   firmwareLinux = firmware[Key]
   firmwareGPU = 256 - firmwareLinux

   print (str(count) + ". Allocate "
+ str(firmwareLinux) +
          " MB of RAM to Linux
leaving " + str(firmwareGPU) +
          "MB of RAM for the GPU.")

   count += 1
```

10 Menü fertigstellen

Wir stellen unser Menü fertig, indem wir eine Schließen-Option hinzufügen. Die Zählung wird erhöht, nachdem die letzte Firmware-Option im Menü ausgegeben wurde. Das bedeutet, dass wir nichts mit der Zählung tun müssen, nachdem die Schließen-Option ausgegeben wurde. Dies wird immer die letzte Option sein.

```
# Add an exit option
print (str(count) + ". Exit.")
```

11 Auf eine Antwort warten

Nun da wir unser Menü ausgegeben haben, müssen wir eine Schleife starten, die gewährleistet, dass wir eine gültige Option vom User erhalten. Unser Programm könnte sich aufhängen, wenn der User eine Option einstellt, mit der wir nicht gerechnet haben. Darum kümmern wir uns mittels einer while-Schleife, die ausgeführt wird, bis die Stopp-Kondition „validOption" auf True gestellt wird. Jetzt muss der User entweder eine korrekte Option eingeben oder Strg+C drücken, um das Programm zu beenden.

```
# Loop to make sure we get a valid
reply
validOption = False
while validOption == False:
```

12 try und except

Die while-Schleife „validOption" beginnt, sobald Sie die Funktion „raw_input" verwenden, um eine Frage an den User zu richten, und verwahrt die Antwort in einer Variablen namens „Option". Wir wollen sicherstellen, dass der Wert numerisch und der Typ im Endeffekt ganzzahlig ist. Dafür können wir eine „try"-Anweisung benutzen. Die Fehler des von der „try"-Anweisung umschlossenen Codes werden in der „except"-Anweisung abgehandelt. In diesem Fall bedeutet das, wenn die Antwort, die der Nutzer eingegeben hat, nicht numerisch war, wird sich das Programm nicht aufhängen, indem es versucht, diese in eine ganze Zahl zu konvertieren. Stattdessen behandelt die „except"-Anweisung etwaige Fehler mit dem Typ „ValueError". Wir stellen eine Variable namens „isInt" für jeden dieser Fälle ein, damit wir wissen, ob es sicher ist, später einen numerischen Operator auf den Wert auszuführen. Falls ein numerischer Operator auf einen String ausgeführt wird, wird sich das Programm aufhängen.

```
option = raw_input("\nPlease select
an option from above: ")

   # Try and convert our option to
an integer and set isInt either
   # way... we don't want to crash
on a bad option
   try:
        option = int(option)
        isInt = True
   except ValueError:
        isInt = False
```

13 Schleife stoppen

Falls der Wert an diesem Punkt keine ganze Zahl ist, wird die while-Schleife erneut ausgeführt. Andernfalls müssen wir sicherstellen, dass eine gültige Optionsnummer eingegeben wurde, zum Beispiel weniger oder gleich viele der Zählung, die wir benutzt haben, als wir das Menü ausgegeben haben.

```
if isInt == True:
        if option <= count:
                #Stop the while loop
                validOption = True
```

14 Umgang mit den Optionen

Jetzt müssen wir prüfen, ob tatsächlich etwas ausgeführt werden muss. Die letzte Option ist das Beenden, also müssen wir nur weitermachen, wenn die Option geringer als die Zählung ist. Wir erhalten die Größe der neuen Firmware, indem wir den Schlüssel aus dem Datenfeld mit „(index option - 1)" bekommen. Dies liegt daran, dass unsere Optionen mit 1 beginnen, jedoch verwenden Listen ein nullbasiertes System, wobei null das erste Element ist. Wie zuvor erarbeiten wir die GPU-Größe und geben die Informationen auf dem Bildschirm aus.

Dann müssen wir unter Verwendung der Funktion „os.system" den Befehl „cp" ausführen, um einen externen Befehl aufzurufen. Wir bekommen den Pfad der Firmware mit der Syntax „firmwares[key]", wobei der Schlüssel die RAM-Größe ist, die Linux zugeteilt wird (wir setzen die Schlüsselwerte im Datenfeld an die Spitze). Das Ergebnis der Ausführung dieses Befehls wird in der Variablen „copyResult" gespeichert. Bei der Ausführung beliebiger Befehle in Linux wird der Code 0 zurückgegeben, wenn der Befehl erfolgreich ausgeführt wurde, wobei andere Codes als Indikatoren von Fehlern dienen. Durch Überprüfen dieses Fehlercodes ist es uns möglich, eine Fehlermeldung auszugeben und das Programm zu beenden, falls etwas anderes als 0 zurückgegeben wird. Ansonsten können wir eine Erfolgsmeldung ausgeben und mit dem finalen Teil unseres Programms weitermachen.

```
# Deal with our options (only in the
range of the count as we don't
# need to handle exit)
if option < count:
   # Work out the RAM allocation of
Linux and GPU
   newFirmware = firmwares.keys()
[option - 1]
   newGPU = 256 - newFirmware

   print ("\nUsing the firmware
which allocates " + str(newFirmware)
                + "MB of RAM to
Linux and " + str(newGPU) +
                  "MB of RAM to the
GPU...")

   # Run the cp command with the
appropriate values
   copyResult = os.system("cp " +
firmwares[newFirmware] + " " +
```

```
    "/boot/start.elf")

    if copyResult != 0:
        print ("\nThere was an error
while copying the requested"
                " firmware!")
        print ("Exiting...")
    else:
        print ("\nThe
requested firmware was copied
successfully.\n")
```

15 Schleife neu starten

Das Gerät muss neu gestartet werden, bevor die Firmware-Änderungen wirksam werden, also werden wir, wie wir es bei der Übernahme der Optionen getan haben, eine while-Schleife erstellen, die für den Wert „y" oder „n" anhält. Beachten Sie, dass diese while-Schleife nur dann ausgeführt wird, wenn der Kopiervorgang erfolgreich war, wie bei der „else"-Anweisung. Sehen Sie, wie wir mehrere Stringfolgen in Anführungszeichen in die Funktion „raw_input" setzen. Diese sind implizit verkettet, wir haben das Additionszeichen nicht verwendet. Das sieht in diesem Fall ordentlicher aus. Eine explizite Verkettung ist jedoch dann nötig, wenn außer Strings noch etwas anderes vorkommt – wie etwa eine Variable oder die Funktion „str()".

```
# Loop to make sure we have a proper
response for restarting
        restartLoop = False
        while restartLoop == False:
                restartResult = raw_
input("The device needs to restart "
                "before the changes
take place. Would you like to do "
                "this now? (y/n) ")
```

16 Umgang mit Restart-Input

Wir haben die „if"-Anweisung, die überprüft, ob das Ergebnis „y" war. Wenn dem so ist, wird die Schleife durch Einstellen der Stoppkondition angehalten. Wir werden dann ausgeben, dass das System einen Neustart durchführen soll, und rufen den Systembefehl „shutdown -r now" auf. Die „elif"-Anweisung (else if) wird nur geprüft, wenn die erste Anweisung nicht True ist. Falls das Ergebnis also nicht „y", sondern „n" ist, wird der Code für diese Anweisung ausgeführt. Andernfalls wird die while-Schleife erneut ausgeführt. Wie zuvor kann der Nutzer das Programm an dieser Stelle mit der Tastenkombination Strg+C beenden.

```
if restartResult == "y":
                        # Stop the
while loop
                        restartLoop
= True
                        # We need to
restart
                        print("The
system is going down for a
restart...")

os.system("shutdown -r now")

                elif restartResult
== "n":
                        # Stop the
while loop
                        restartLoop
= True
                        # We just
need to exit (by doing nothing)
                        print ("\
nExiting...")
```

17 Ihr neues Programm zugänglich machen

Stellen Sie durch Abspeichern der Änderungen alles fertig. Jetzt da unser Programm komplett ist, wollen wir es vom Terminal aus zugänglich machen wie jeden anderen Befehl. Öffnen Sie dafür zunächst via LXDE Menu > Accessories > LXTerminal ein Terminal. Verwenden Sie den Befehl „cd", um zum Verzeichnis zu wechseln, wo sich Ihr Python-Skript befindet. Dort verwenden Sie den Befehl „chmod +x [Name des Skripts]", um es ausführbar zu machen. Das Skript muss im Ordner /usr/bin liegen. Doch statt es zu kopieren, sollten Sie lieber einen symbolischen Link erstellen, der im aktuellen Ordner auf das Skript verweist. Das bedeutet, dass alle hier vorgenommenen Änderungen automatisch übernommen werden, wenn das Programm direkt aus dem Terminal ausgeführt wird. Um einen symbolischen Link zu erstellen, verwenden Sie den Befehl „sudo ln -s [Pfad zum Python-Skript] /usr/bin/[gewünschter Befehl für die Kommandozeile]". Hier haben wir zum Beispiel „sudo ln -s /home/pi/Python/RamPi/RamPi.py /usr/bin/RamPi" verwendet.

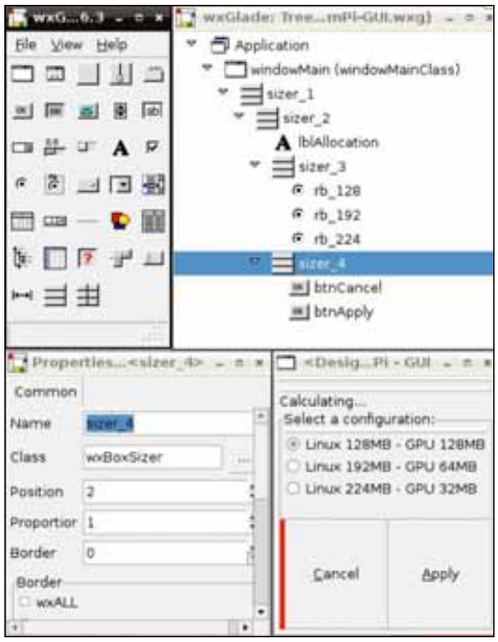
18 Ihr neues Programm ausführen

Da wir nun einen symbolischen Link auf „/usr/bin" erstellt haben, müssen Sie nur noch den Befehlsnamen eingeben, in unserem Fall wäre das „RamPi". Denken Sie daran, das Kommando als Root-User auszuführen, und nichts kann mehr schiefgehen!

```
pi@raspberrypi:~$ ls
Desktop  helloworldv2  output.pdf  Python  Scratch
pi@raspberrypi:~$ cd Python/RamPi/
pi@raspberrypi:~/Python/RamPi$ ls
RamPi.py
pi@raspberrypi:~/Python/RamPi$ chmod +x RamPi.py
pi@raspberrypi:~/Python/RamPi$ ln -s /home/pi/Python/RamPi/RamPi.py
 /usr/bin/RamPi
ln: creating symbolic link '/usr/bin/RamPi': Permission denied
pi@raspberrypi:~/Python/RamPi$ sudo !!
sudo ln -s /home/pi/Python/RamPi/RamPi.py /usr/bin/RamPi
```

> "Jetzt da unser Programm komplett ist, wollen wir es vom Terminal aus zugänglich machen wie jeden anderen Befehl."

Das ist der Controls-Dialog von wxGlade. Jedes Icon vertritt einen Befehl. Um einen Befehl hinzuzufügen, klicken Sie auf das Icon des Befehls und ziehen es auf die Entwurfsansicht.



Dies ist die Tree-Ansicht von wxGlade; sie erzeugt eine hierarchische Ansicht der einzelnen Befehle in der Anwendung.

Das ist das Properties-Fenster von wxGlade, mit dem Sie Einstellungen an Befehlsnamen, an der Rahmengröße, oder am im Befehl angezeigten Text etc. vornehmen können.

Das ist die Design-Ansicht von wxGlade, in der wir einen groben Einblick bekommen, wie unser Programm aussehen wird, wenn es fertig ist.

So programmieren Sie grafische Benutzeroberflächen

Der bisherige Artikel behandelte das Schreiben eines Befehlszeilen-Programms für den Raspberry Pi. Nun lernen wir, eine Version dieses Programms mit einer grafischen Benutzeroberfläche zu erstellen.

Was Sie brauchen:

- einen Raspberry Pi samt Peripherie
- die aktuelle Version von Raspberry-Pi-Debian

Dieses Tutorial knüpft an das vorhergehende von S. 90 an, in dem wir Ihnen zeigten, wie Sie ein Kommandozeilen-Programm schreiben, das es dem User erlaubt, die RAM-Größe, die dem GPU des Raspberry Pi zugewiesen wird, einfach zu ändern. Es ist sinnvoll, die RAM-Zuweisung abhängig davon zu ändern, was Sie mit dem Gerät vorhaben. Zum Beispiel: Größere 3D-Arbeiten würden mehr des für die GPU zugewiesenen RAM benötigen, als wenn Sie das Gerät kopflos ausführen würden.

Wir werden ein Werkzeug namens wxGlade verwenden, das im aktuellsten Debian Image (debian6-19-04-2012 zum Zeitpunkt des Schreibens) enthalten ist. Mit wxGlade können Sie mittels Drag & Drop verschiedener Befehle auf ein Fenster - ähnlich wie Buttons - eine Benutzeroberfläche (GUI) gestalten, um damit eine Python-Datei zu generieren, die dieses Fenster anzeigen wird. Von hier aus werden wir dem Code Ereignisbehandlungen hinzufügen, die es Teilen des Codes erlaubt, als Antwort auf ein durch einen Befehl hervorgerufenes Ereignis ausgeführt zu werden, wie zum Beispiel ein Button-Klick.

Der Code, der die zugewiesene RAM-Größe durch Ändern der geladenen Firmware-Datei im Boot ändert, wird dem im vorherigen Tutorial verwendeten Code ziemlich identisch sein. Stellen Sie also sicher, dass Sie diesen gelesen haben und vertraut mit der bisherigen Vorgehensweise sind, bevor Sie mit diesem Guide fortfahren.

Am Ende dieses Tutorials werden Sie eine Anwendung haben, mit der der User die vom Pi verwendete RAM-Größe mühelos ändern kann und die für Einsteiger simpel und freundlich aussehen wird.

01 wxGlade starten

wxGlade ist der GUI-Designer, den wir nutzen werden. Rufen Sie ihn in der Programming-Kategorie im LXDE-Menü auf. wxGlade arbeitet mit mehreren Fenstern statt eines großen Fensters. Das Fenster mit vielen Icons ist die Controls-Toolbar. Diese Komponenten verwenden wir für den Aufbau unserer Anwendung.

02 Fortschritte speichern

Bevor wir fortfahren, speichern wir unser wxGlade-Projekt – am einfachsten in regelmäßigen Intervallen via Strg + S. So geht keine Arbeit verloren, falls das Gerät abstürzt. Dieses Projekt wird mehrere Dateien beinhalten (die wir „RamPi-GUI“ nennen), erstellen Sie vor dem Speichern also einen Ordner. Gehen Sie in das obige File-Menü, das die Befehle enthält und wählen Sie Save, um den Save-Dialog anzuzeigen. Nutzen Sie den Button „Create Folder“ ganz oben.

03 Einen Rahmen hinzufügen

Klicken Sie auf das Rahmen-Icon im Controls-Fenster. Dies fügt Ihrem wxGlade-Projekt einen Rahmen (ein Fenster) hinzu. In einem Dialogfenster stellen Sie die Werte Ihres neuen Rahmens ein. Wählen Sie wxFrame als Basisklasse und benennen Sie sie angemessen. Wir vergeben den Namen „windowMainClass“. Sobald Sie diese Werte eingegeben haben, klicken Sie OK. Ein neues Fenster wird mit einem Titel erscheinen, der mit <design> beginnt und der Rahmen wird dem Tree-Fenster hinzugefügt.

04 Rahmeneigenschaften einstellen

Um zu gewährleisten, dass der Rahmen ausgewählt ist, klicken Sie in der Tree-Ansicht auf frame_1. Ändern Sie im Properties-Fenster den Namen in „windowMain“ und klicken Sie dann auf den Widget-Tab. Ändern Sie den Titel des Fensters in „RamPi – GUI“. Die Eigenschaft, die Sie bearbeiten, wird geändert, sobald Sie aus dem Properties-Fenster herausklicken, sodass es den Fokus verliert. Vergessen Sie nicht, Ihre Arbeit zu speichern!

05 Anwendungseigenschaften

Jetzt wäre es an der Zeit, die Anwendungseigenschaften einzustellen. Wählen Sie den Application-Eingang in der Tree-Ansicht und prüfen Sie jedes Kontrollkästchen neben dem Namen sowie die Klasseneigenschaften der Anwendung. Setzen Sie beide auf “RamPiGUI”.

06 Sizer verwenden

In wxWidgets und in vielen anderen Toolkits zum Erstellen von GUIs auf Linux ordnen Sie Ihre Befehle in Zeilen, Spalten und/oder Tabellen. Ein Sizer ist ein Befehl, der mehrere Slots in sowohl einer horizontalen als auch einer vertikalen Ausrichtung haben kann. Das ist der Weg, um Zeilen und Spalten zu erzeugen.

Unser windowMain hat bereits einen Sizer namens sizer_1. Um einen weiteren Sizer innerhalb des sizer_1 hinzuzufügen, klicken Sie auf das Sizer-Icon, dann auf sizer_1 in der Tree-Ansicht und klicken Sie dann irgendwo in die Design-Ansicht, um einen Sizer hinzuzufügen. Ein Dialogfeld mit Eigenschaften des neuen Sizers wird angezeigt. Setzen Sie die Ausrichtung auf vertikal, die Anzahl der Slots auf 3 und klicken Sie OK, um den Sizer hinzuzufügen.

07 Ein Label hinzufügen

Im oberen Drittel des Fensters werden wir dem User die aktuelle RAM-Zuweisung erklären; wir werden ein Label verwenden, um diese Information schreibgeschützt anzuzeigen. Klicken Sie auf den Befehl mit einem „A“-Icon, dann in das obere Feld des zweiten Sizers, um diesem das Label hinzuzufügen. Erstellen Sie einen angemessenen Namen für das Label im Properties-Fenster, da uns dieses nur vom Code aus zugänglich ist. Unseres benennen wir mit „lblAllocation“. Wir werden den Text auf „Calculating...“ stellen, weil er sich ändern wird, sobald Codes beim Starten der Anwendung ausgeführt werden. Um dies zu tun, gehen Sie auf den Widget-Tab im Properties-Fenster und ändern Sie den Label-Wert.

08 Ein statisches Feld hinzufügen

Um die dem User zur Verfügung stehenden verschiedenen Speicher-Konfigurationen aufzulisten, fügen wir einen weiteren Sizer im mittleren Slot des ersten Sizers hinzu. Stellen Sie also wie zuvor wieder die Ausrichtung auf vertikal und den Slot auf 3. Lassen Sie diesmal aber die Option „has a static box“ zu und setzen das Label auf „Select a configuration“.

09 Options-Buttons hinzufügen

Jeder Sizer-Slot erhält einen Options-Button, um verschiedene Speicher-Konfigurationen zu vertreten. Klicken Sie dazu auf das Options-Button-Icon und dann in jeden Slot des Sizers. Wiederholen Sie dies für die verbleibenden beiden Slots. Benennen Sie jeden Options-Button mittels des Properties-Fensters, da wir erklären müssen, welcher aus dem Code gewählt wird. Wir haben rb_128, rb_192 und rb_224 für die drei verschiedenen Speicher-Konfigurationen verwendet. Für jeden der Options-Buttons stellen Sie im Widget-Bereich des Properties-Dialogs den Text auf etwas zwischen den Zeilen von „Linux 128MB – GPU 128MB“ ein. Wiederholen Sie das für die verbleibenden zwei RAM- Konfigurationen.

10 Weitere Buttons hinzufügen

Zuletzt fügen wir zweier Buttons hinzu: Apply und Cancel. Zuvor jedoch noch einen Sizer mit einer horizontalen Ausrichtung und zwei Slots. Danach wählen Sie den Control-Button und ziehen einen auf den Slot jedes Sizers. Verwenden Sie das Properties-Fenster, um die Buttons jeweils mit „btnCancel“ und „btnApply“ zu benennen. wxWidgets verfügt über eine Sammlung von Standard-Buttons. Um diese zu verwenden, gehen Sie in den Widgets-Bereich des Properties-Fensters, um ein Stock Item-Feld freizugeben. Wählen Sie dann die Apply- und Cancel-Buttons.

11 Vorschau eines Fensters

Rechts-Klicken Sie in der Tree-Ansicht auf das Element „windowsMain“ und wählen Sie Preview. Beachten Sie, dass die Befehle unschön angeordnet sind und dass es keine Grenzen zwischen den Befehlen gibt. Im Layout-Bereich des Properties-Fensters können Sie mit der angemessenen Anordnung der Befehle herumexperimentieren.

12 Befehlsabstand

Obwohl die angemessene Anordnung der Befehle den Rahmen dieses Tutorials sprengen würde, zeigen wir Ihnen dafür einige Beispiele. Gehen Sie in den Layout-Bereich des Properties-Fensters, wählen Sie den ersten Sizer, den wir in der Tree-Ansicht hinzugefügt haben und stellen Sie den Rahmen auf 5 – stellen Sie ihn dann auf „wxALL“, was bedeutet, dass jeder Seite des Sizers ein Rahmen mit 5 Pixeln hinzugefügt wird. Damit die beiden Buttons am Rand den kompletten Platz ausfüllen, setzen Sie ihre Ausrichtung auf „wxEXPAND“ und ihr Verhältnis auf 1. Das bedeutet, dass beide die gleiche Menge an Platz in ihrem Sizer einnehmen werden.

13 Code-Generation

Klicken Sie auf die Application-Eigenschaft in der Tree-Ansicht und stellen Sie sicher, dass die Sprache der Code-Generierung im Properties-Fenster auf Python eingestellt ist. Legen Sie einen Pfad für den Ausgang einer Datei namens „RamPi-GUI.py“ im Ordner für das Projekt, das wir zuvor erstellt haben, fest. Sobald Sie dies erledigt haben, klicken Sie auf den Button „Generate Code“ und schließen wxGlade.

14 Zurück zu Geany

Starten Sie Geany und öffnen Sie die Datei, die Sie gerade mit wxGlade generiert haben. Schauen Sie sich den Code an, damit Sie verstehen, wie eine Benutzeroberfläche aufgebaut und angezeigt wird.

15 Den RamPi-Code hinzufügen

Nach dem Hinzufügen einiger Kommentare, die beschreiben, was die Anwendung tut, müssen wir die Modulimporte von der Konsolen-Version von RamPi sowie das Firmware-Verzeichnis und die Key/Path-Konstanten herüberkopieren. Nun konvertieren wir die Teile des RamPi, die wir in Funktion sehen wollen. Der Code darunter ist der erste Bereich des „RamPi-GUI“-Codes, bis der „windowMainClass“-Code anfängt.

```
[
#!/usr/bin/env python

# RamPi-GUI - a program written by
Liam Fraser for a Linux User and
# Developer tutorial - 19/06/2012.
This tool allows the user to easily
# change the amount RAM that is used
by the Raspberry Pi's GPU.

# Import required modules
import os # For executing system
commands and getting uid
import filecmp # For comparing files
to get the firmware in use
import sys # For the sys.exit
function
import wx # For the gui toolkit
wxWidgets

# Create a dictionary with the RAM
use in Linux as the key and the
# firmware file path as the value
firmwares = {128:"/boot/arm128_
start.elf",
             192:"/boot/arm192_
start.elf",
             224:"/boot/arm224_
start.elf"}

# Constants for easier reading when
getting a key or value from
# an item in the dictionary
Key = 0
Path = 1

def checkRoot():
   # Check we have root privelages
   if os.geteuid() != 0:
        return False
   else:
        return True

def getCurrentFirmwareKey():
   # Work out which firmware is
currently in use by comparing each
file
   # in the dictionary with start.
elf
   for firmware in firmwares.
items():
        result = filecmp.cmp("/boot/
start.elf", firmware[Path], False)

        # When we find the correct
file
        if result == True:
                # Return the
firmwares key
                return firmware[Key]

def copyFirmware(firmwareKey):
   # Run the cp command with the
appropriate values
   copyResult = os.system("cp " +
firmwares[firmwareKey] + " " +

   "/boot/start.elf")

   if copyResult != 0:
        # Error
        return False
   else:
        # Success
        return True

def restartSystem():
   # Run the restart command
   os.system("shutdown -r now")
]
```

16 Nach Root überprüfen

Wir nutzen unsere Funktion „checkRoot“, um die Anwendung zu beenden, bevor das Hauptfenster anzeigt, ob der User nicht über Root-Berechtigungen verfügt. Dieser Code muss in der Funktion „OnInit“ der Klasse „RamPiGUI“ stehen, nachdem die Funktion „wx.InitAllImageHandlers“ aufgerufen wird. Hat der User keine Root-Berechtigungen, wird ein Nachrichten-Feld angezeigt, indem die Funktion „wx.MessageBox“ mit folgenden Parametern aufgerufen wird: „wx.MessageBox(Message, Window Title, Button Type | Icon)“.

```
[
# Check for root
        if checkRoot() == False:
```

```
                wx.MessageBox("You
need to be root to run this
program",
                "Error", wx.OK |
wx.ICON_ERROR)
                # End the program
here
                sys.exit()
]
```

17 Aktuelle Konfiguration anzeigen

Das Anzeigen der aktuellen Konfiguration muss in der Funktion „__init__“ von „windowMainClass“ erfolgen, bevor „lblAllocatio“ erstellt wurde. Dazu erhalten wir den Schlüssel (die Linux zugeteilte RAM-Größe) der aktuellen Firmware-Datei, um dann die Größe zu erarbeiten, die dem GPU zugeteilt wird. Wir erstellen eine Zahlenfolge formatiert mit diesen Werten, in der die „%d“-Werte mit den Parametern nach dem letzten % ersetzt werden. Letztlich ändern wir “lblAllocation” von “Calculating...” auf “configString”.

```
[
# Get current configuration and
format a string that will be
# set as the value of lblAllocation
linuxRam = getCurrentFirmwareKey()
gpuRam = 256 - linuxRam
configString = ("Current
Configuration: Linux %dMB - GPU
%dMB" %
  (linuxRam, gpuRam))

self.lblAllocation =
wx.StaticText(self, -1,
configString)
]
```

18 Funktionen an ein Ereignis binden

Ereignisse behandeln wir genau wie Button-Klicks, indem wir ein Ereignis an eine Funktion binden. Wir binden Ereignisse in der Funktion „__init__“ der Klasse „windowMainClass“, nachdem die Buttons erstellt wurden. Ereignisse werden mit der Schreibweise „[control name].Bind([event type], [function to call])“ gebunden. Ebenso müssen Sie die Funktionen definieren, die innerhalb der Klasse „windowMainClass“ aufgerufen werden. In unserem Fall müssen wir herausfinden, welcher Options-Button ausgewählt wurde.

Die Funktion muss mittels zweier Parameter definiert werden: „self“ und „e“. Der „self“-Parameter bedeutet einfach, dass auf die Variablen aus der Instanz der Klasse durch Voranstellen des Variablennamens mit „self“ zugegriffen werden kann, und der „e“-Parameter wird alle Ereignisargumente haben, die ihn durchlaufen haben, wenn das Ereignis aufgerufen wird. Wir haben unsere Funktionen am Ende des Blocks „windowMainClass“ des Codes definiert.

```
[
self.btnCancel = wx.Button(self,
wx.ID_CANCEL, "")
self.btnApply = wx.Button(self,
wx.ID_APPLY, "")

# Bind click events for btnCancel
and btnApply
self.btnCancel.Bind(wx.EVT_BUTTON,
self.btnCancel_Click)
self.btnApply.Bind(wx.EVT_BUTTON,
self.btnApply_Click)

   def btnCancel_Click(self, e):
        # Run when the cancel button
is clicked

   def btnApply_Click(self, e):
        # Run when the apply button
is clicked

# end of class windowMain
]
```

19 Der Apply-Button

Nun müssen wir unsere Funktionen mit Codes füllen. Wenn auf den Apply-Button geklickt wird, müssen wir zuerst herausfinden, welche Firmware-Datei ausgewählt wurde, indem Sie überprüfen, ob der Wert des Options-Buttons wahr ist. Sobald wir diesen Wert haben, rufen wir einfach die Funktion „copyfFrmware“ auf, die den Schlüssel der neuen Firmware-Datei durchläuft, die wir benötigen, und ein Meldungsfeld als Indikator dafür anzeigt, ob die Kopie erfolgreich war. Danach wollen wir ein Meldungsfeld anzeigen, das erfragt, ob der User einen Neustart (Klick auf „Yes“) möchte.

```
def btnApply_Click(self, e):
# Run when the apply button is
clicked

   # Find out what firmware file has
been selected
   if self.rb_128.GetValue() ==
True:
        newFirmwareKey = 128
   elif self.rb_192.GetValue() ==
True:
        newFirmwareKey = 192
   elif self.rb_224.GetValue() ==
True:
        newFirmwareKey = 224

   # Replace start.elf with the
firmware file of the new key
   copyResult =
copyFirmware(newFirmwareKey)
   if copyResult == False:
        wx.MessageBox("Error",
"Error", wx.OK | wx.ICON_ERROR)
        sys.exit()
   else:
        wx.MessageBox("Success",
"Success", wx.OK | wx.ICON_
INFORMATION)

   # Ask if the user wants to
restart
   restartResult =
wx.MessageBox("Would you like to
restart now "

"for the changes to take effect?",
"Restart?",

wx.YES_NO | wx.ICON_QUESTION)

   if restartResult == wx.YES:
        restartSystem()
]
```

20 Der letzte Schliff

Um den RamPi abzuschließen, schreiben Sie einen Code, um den Cancel-Button zu aktivieren (durch Aufrufen von „sys.exit()“). Alternativ schreiben Sie „pass“ in den Code für den Cancel-Button, sonst startet das Programm nicht.

21 Unser neues Programm testen

Verlassen Sie Geany. Rufen Sie LXDE-Menü > Accessories > File-Manager auf. Gehen Sie auf den RamPi-GUI-Projektordner und drücken Sie F4, um einen Terminal darin zu öffnen. Die Anwendung benötigt Root-Rechte. Tippen Sie also „sudo python [your python file]“ und genießen Ihr erstes GUI-Programm auf dem Pi!

PiFace: Elektronik für den Raspberry Pi

PiFace bringt die Welt der Automatisierung und Elektronik nach Hause, in die Schule und die Industrie – zu einem zuvor undenkbaren Preis.

Der Raspberry Pi ist extrem günstig und sowohl auf Software- als auch auf Hardwareebene extrem offen. Dadurch ermöglicht er Elektronikprojekte, die bislang sehr teuer und aufwendig zu programmieren gewesen wären.

Der Pi wurde ursprünglich für die Lehre entwickelt. Dr. Andrew Robinson von der University of Manchester sah in ihm das Potenzial, eine neue Generation für Computer und Elektronik zu begeistern. Doch er erkannte auch die Notwendigkeit eines einfachen Einstiegs und einer Möglichkeit, den Benutzern und Lehrern die nötige Sicherheit zum Experimentieren zu geben. Der Raspberry Pi alleine kann direkt an elektronische Geräte angeschlossen werden. Er besitzt eine Allzweck-Stiftleiste, mit der man sein eigenes Interface hacken kann. Robinsons PiFace-Board hingegen ist eine Erweiterung, die das Anschließen schnell und einfach macht. Verbindungen werden mit Schraubklemmen hergestellt, und alle Inputs und Outputs sind zum Schutz vor Fehlern gepuffert.

Bild mit freundlicher Genehmigung von Rosie Slosek (onemanbandaccounting.co.uk)

Dr. Andrew Robinson, der Erfinder der PiFace-Platine.

„Du hast zehn Sekunden, um ihre Aufmerksamkeit zu kriegen und sie zum Staunen zu bringen."

Das PiFace passt genau auf den Pi.

Robinson bemerkte einen Rückgang des Interesses an Informatik und stellte einen Mangel an Fähigkeiten im Embedded Computing fest, besonders seit dem Niedergang des BBC Micro. Elektronischen Bauteile zum Herumbasteln sind für den Normalbürger nicht mehr so leicht zu beschaffen. Spiele wurden in Konsolen und Elektronik in geschlossene Systeme gesperrt. Darum ist er so begeistert vom Potenzial des Pi, den Hobbybastlerbereich zurück in die IT zu bringen. Auch jetzt noch öffnet das vom Raspberry Pi hervorgerufene Wohlwollen Türen und erzeugt Interesse in einem Ausmaß, wie es in IT-Kreisen seit Jahren nicht gesehen wurde.

„Während man beim BBC Micro beobachten konnte, wie der Computer von Büro und Kassensystemen in die Hände der Heimandwender wanderte, sieht man am Beispiel des Raspberry Pi dieselbe Entwicklung bei eingebetteten Systemen: Einst weggesperrt in Handys, Waschmaschinen und MP3-Playern, nun für jedermann zugänglich."

Robinsons Interesse an Elektronik begann im Alter von fünf Jahren mit dem Bau eines Leuchtturmmodells. Seitdem hat er eine Neigung zu allem mit blinkenden Lämpchen und Technik im Allgemeinen. Ein paar Jahre später begann er auf einem Spectrum und einem BBC Micro zu programmieren. Ein großer Reiz des Letzteren lag in dem einfach zugänglichen User-Port, den man verwenden konnte, um Motoren oder Lampen zu steuern. Seinem Interesse für Computer folgend, machte Robinson seine Bachelor- und Master-Abschlüsse an der University of Manchester. Nach einer kurzen Beschäftigung als Lichttechniker in Theatern und auf Tournee vertrieb Robinson computergesteuerte Lichtanlagen und war als Berater für digitale Kunstinstallationen tätig.

An der University of Manchester promovierte er in Prof. Steve Furbers Forschungsgruppe über verbrauchsarme Prozessorarchitekturen (darunter Ideen für effizientere ARM-Prozessoren) und arbeitete als Lehrassistent für Hardware und Software. Während der Arbeit in der Gruppe entwickelte Robinson Interesse daran, sich öffentlich zu engagieren, um mehr Menschen für die Informatik zu begeistern. Der Kreis schloss sich, als Steve Furber, einer der Architekten des BBC Micro, ihn dazu ermutigte, mit öffentlichen Aktionen den Nachwuchs zu inspirieren.

Bei der Veröffentlichung des Raspberry Pi wusste man noch nicht, wie sehr er junge Leute packen würde. Man hatte nicht geahnt, dass seine bei Hobbybastlern ohnehin riesige Popularität auch auf Kinder überspringen würde. Im Dezember 2011 legte Robinson die ersten Spezifikationen fest und designte ein Interface und einen Kerneltreiber. Doch um seine didaktischen Ziele zu erreichen, brauchte er ein Team. So holte er Jim Garside wegen seiner technischen Expertise und Amanda Banks, die für die erzieherische Seite zuständig sein sollte, ins Boot. Über den Sommer wurden die Studenten Tom Heyes, Thomas Macpherson-Pope und Thomas Preston angestellt, um die Schnittstellen zu entwickeln, während Ziyad Gideir und Tomas Cepulis an Lehrmaterialien arbeiteten und Schulen besuchten.

Robinson ist der Meinung, dass man Kindern einen Aufhänger geben muss. „Du hast zehn Sekunden, um ihre Aufmerksamkeit zu kriegen und sie zum Staunen zu bringen. Wenn du sie dann nicht für dich einnimmst, werden sie auch nicht weiter nachforschen." Mit dem PiFace Digital realisierte er einige sehr interessante Elektronikprojekte. Beliebt bei seinen Studenten ist beispielsweise ein Huhn, das Tweets vorliest. Die Stimme des Huhns wird von eSpeak, einem Sprachsynthesizer, erzeugt, während PiFace die Bewegungen des Schnabels steuert. Nach dem Herunterladen des Codes benutzt man einfach folgende Kommandos:

»

PiFace einrichten

Sie benötigen eine SD-Karte mit 4 GB Speicherplatz für Ihren Raspberry Pi. Das jeweils aktuelle Image des Betriebssystems Raspbian für die Karte können Sie unter dem Link downloads.raspberrypi.org/raspbian_latest herunterladen.

Nach dem Download der Distribution können Sie auf einem Linux-PC das Kommando „sudo dd bs=1M if=/path/to/image of=/dev/sdd" verwenden, um das Image auf die Karte zu schreiben. Ubuntu-Nutzer können auch das ImageWriter-Tool aus dem Software-Center installieren. Unter Windows müssen Sie „Image Writer for Windows" von **launchpad.net/win32-image-writer** herunterladen. Seien Sie jedoch sehr vorsichtig mit diesem Werkzeug, bei falscher Benutzung kann es Ihre Festplatte löschen. Vollständige Anleitungen für alle Betriebssysteme sind unter **raspberrypi.org/documentation/installation/installing-images/README.md** zu finden.

Ist Ihre SD-Karte fertig vorbereitet, müssen Sie nur noch booten und sich einloggen:
Benutzername: pi
Passwort: raspberry
Username: pi
Password: raspberry

Technische Daten

ÜBERSICHT

Maße: ..... 56mm x 84.5mm
Digitale Ausgänge: 8
Digitale Eingänge: 8
Raspberry-Pi-Pins: ..... 3v3, 5v, 0v, SPI MOSI, SPI MISO, SPI SCLK, SPI10 CE0 N, SPI10 CE1 N

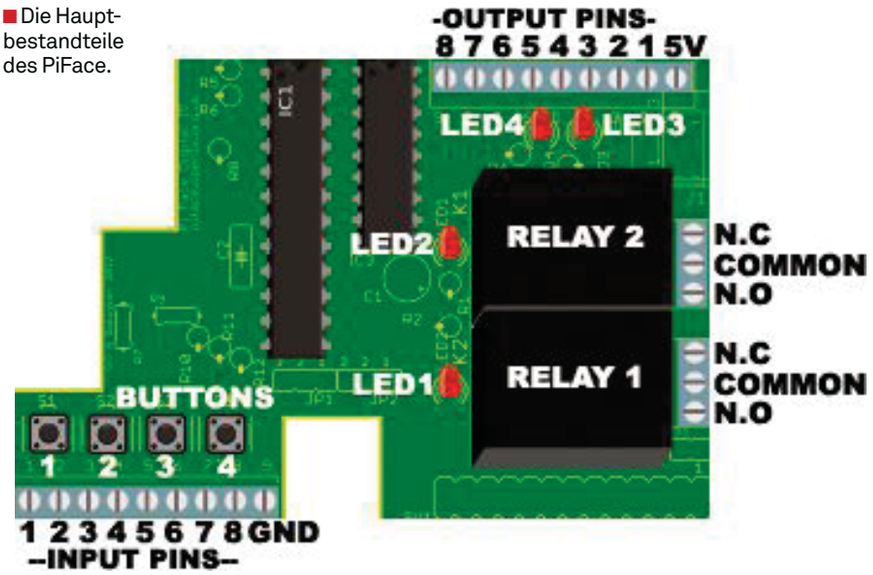
BAUTEILE

8x ..... Schraubklemme 3-polig
1x ..... weiblicher 26-Pin-Anschluss
4x ..... rote LED 3mm
12x ..... Widerstand 330 Ⓟ
4x ..... Druckknopf 6 x 6 mm
1x ..... Keramikkondensator 100 nF
1x ..... Elektrolytkondensator 100 µF
2x ..... Relais OMRON G5LA-1 5VDC
1x ..... MCP23S17-E/SP – (spezialisierter 16-Bit -I/O-SPI-Schnittstellen-Chip)
1x ..... ULN2803A 9921V – (Hochspannungs- und Hochstrom-Darlington-Transistormatrix-Chip)

```
python twitter_listen.py #yourhashtag
python twitter_listen_user.py username
```

Die komplexen Suchanfragen, die sich die Studenten einfallen ließen, um herauszufinden, was man das Huhn alles lesen lassen kann, sprechen für sich. Andere Projekte umfassen eine vom Raspberry Pi gesteuerte Carrera-Bahn und ein Schlag-den-Maulwurf-Spiel mit physischen Knöpfen. Wenn Kinder sehen, was möglich ist, beginnen sie, eigene Ideen zu entwickeln. Sogar nur mit dem Pi und dem PiFace können Schüler schon LEDs leuchten lassen, und es gibt ihnen ein gewisses Machtgefühl, tatsächlich ein physisches Gerät zu kontrollieren. Mit dem PiFace Digital kann der Raspberry Pi auf Eingaben reagieren und Ausgänge ansprechen, die Servo- und andere Motoren, LEDs oder Glühbirnen steuern, oder was man sich sonst so vorstellen kann.

Eben Upton, Gründer der Raspberry-Pi-Stiftung und CEO der Raspberry-Pi-Handelsgesellschaft, stimmt zu: „Wir hoffen, dass die Leute Gertboards und andere Erweiterungskarten verwenden, um alle möglichen Dinge im Gebiet der Heimautomatisierung zu tun. Physical Computing ist ein großes Geschäft – wir sehen, wie viele junge Leute ihre ersten Computererfahrungen mit Plattformen wie Arduino und PiFace sammeln, und hoffen, ähnliche Projekte auf Basis des Raspberry Pi zu sehen."



Die Hauptbestandteile des PiFace.

Die APIs

Es gibt drei Haupt-APIs für PiFace: eine spezielle, modifizierte Version von Scratch, die Python-API und eine HTTP-API. Sie können es aber auch „direkt" in C programmieren.

Für Anfänger ist die Scratch-Oberfläche einfach und intuitiv zu benutzen. Sie liest nicht nur Inputs und bietet die Möglichkeit, Outputs zu verändern, sondern kann auch grafische Ausgaben darstellen. Dies könnte etwa in Schulen für Wetterstationen und andere wissenschaftliche Experimente mit visuellem Feedback benutzt werden.

Die meiste bisher vorliegende Software für PiFace ist in Python geschrieben. Die Python-API ist sowohl flexibel als auch reich an Funktionalität. Die HTTP-API beruht in der Hauptsache auf Python und erlaubt es Ihnen, PiFace über das Internet fernzusteuern. Es gibt eine Android-App, mit der man die an PiFace angeschlossenen Sensoren auslesen und die Ausgänge von einem Smartphone oder Tablet aus bedienen kann.

Die Designer legten großen Wert darauf, die API so einfach wie möglich zu gestalten. So braucht man nicht viel Code, um die Ausgänge des Boards zu kontrollieren.

Um auf PiFace-Funktionen zuzugreifen, öffnen Sie eine Shell und tippen Sie:

```
$ python
import piface.pfio as pfio
pfio.init()
```

Dann können Sie auf die PiFace-API zugriefen. Für alle, die mit Arduino vertraut sind, gibt es aus Konsistenzgründen die Funktion digital_write. Um den Wert eines Ausgangs zu ändern:

```
pfio.digital_write(<output pin number>, <value>)
```

Und um den Wert eines Eingangs zu lesen:

```
pfio.digital_read(<input pin number>)
```

Es gibt Hilfsfunktionen, die es besonders Anfängern noch leichter machen. Um die erste LED einzuschalten:

```
pfio.LED(1).turn_on()
```

Ein einfaches Programm, um die erste LED einzuschalten, wäre demnach:

```
import piface.pfio as pfio
pfio.init()
pfio.LED(1).turn_on()
```

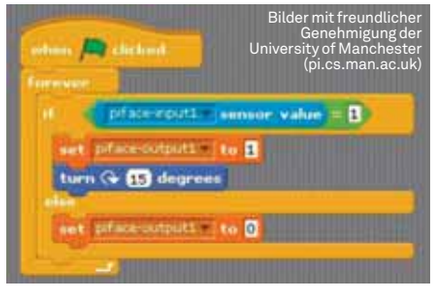
Und um sie wieder auszuschalten, tippen Sie:

```
pfio.LED(1).turn_off()
```

PiFace-Projekte

Außer dem sprechenden Huhn hat das Team hinter PiFace auch ein Schlag-den-Maulwurf-Spiel, ein „Senso"-Spiel und sogar eine Carrera-

> „Wir sehen, wie viele junge Leute ihre ersten Computererfahrungen mit Plattformen wie Arduino und PiFace sammeln, und hoffen, ähnliche Projekte auf Basis des Raspberry Pi zu sehen."



Bilder mit freundlicher Genehmigung der University of Manchester (pi.cs.man.ac.uk)

PiFace-Programmierung in Scratch.

Eine Pi-gesteuerte Carrera-Bahn.

Bahn-Schnittstelle gebaut. All das haben sie mit einfach zu beschaffenden, kostengünstigen Bauteilen gemacht. Was mit etwas Ehrgeiz möglich ist, zeigt ein weiteres Projekt besonders eindrücklich: das Raspberry-Pi-Vogelhäuschen. Es hat zwei Infrarot-Lichtschranken, und abhängig davon, welche zuerst unterbrochen wird, kann es erkennen, ob der Vogel das Vogelhäuschen betritt oder verlässt, und wie schnell er sich bewegt. Eine Kamera, die von den Lichtschranken gesteuert wird, zeichnet Videos auf. Ein modifiziertes Netzwerkkabel liefert den Strom für Pi und PiFace gleich mit, sodass nur ein Kabel benötigt wird. Das Team arbeitet mit dem Londoner Zoo zusammen an Raspberry-Pi-basierten Lösungen, um Geräte im Lebensraum von Tieren zu platzieren. PiFace macht es einfach, Sensoren anzuschließen, sodass neben Fotos und Videos auch andere wertvolle Daten aufgezeichnet werden können.

Ein Vorteil von PiFace sind seine LED-Lampen und Druckknöpfe, dank derer ein fertiges Projekt möglicherweise gar keinen Bildschirm mehr braucht. Es kann sich auch mit dem Internet verbinden – entweder über das Netzwerkkabel oder über eine drahtlose Verbindung. Zum Beispiel wäre es in der Vergangenheit irrsinnig teuer gewesen, ein Gewächshaus zu automatisieren, aber jetzt können der Pi und das PiFace nicht nur die Temperatur messen, sie können auch noch darauf reagieren, indem zum Beispiel ein Motor benutzt wird, um ein Fenster zu öffnen, oder ein Schalter betätigt, um zu heizen. Außerdem kann es Sie per E-Mail oder Twitter benachrichtigen, wenn Ihre preisgekrönten Tomaten besondere Zuwendung brauchen. Es könnte recht leicht ein Gerät zusammengestellt werden, um ein bis zwei Wochen lang eine Katze zu füttern und ihr Kommen und Gehen zu protokollieren – und diese Daten könnten online verfügbar gemacht werden, sodass Sie sie unterwegs abrufen könnten. Es gibt eine Gruppe in Chicago, die Sensoren und einen Raspberry Pi benutzt, um Bier zu brauen: Wegen der präzisen Temperaturkontrolle, die bei der Fermentierung benötigt wird, ist der Pi das ideale Werkzeug zur Steuerung. Dank Lichtsensoren werden Sie möglicherweise nie mehr einen Vorhang öffnen oder schließen müssen, und die Möglichkeiten im Bereich der Heimsicherheit sind schier endlos.

Einige der beeindruckendsten Projekte machen sich die Portabilität des Pi zunutze. Raspberry Pis wurden an ferngesteuerte Fahrzeuge angeschlossen oder mit Wetterballons nach oben geschickt. Dies sind schöne Erfolge, doch mit PiFace gibt es die Möglichkeit, auch wertvolle Daten zurückzuschicken – besonders an Orten, die zu erreichen für Menschen schwierig, gefährlich oder einfach nur unangenehm ist.

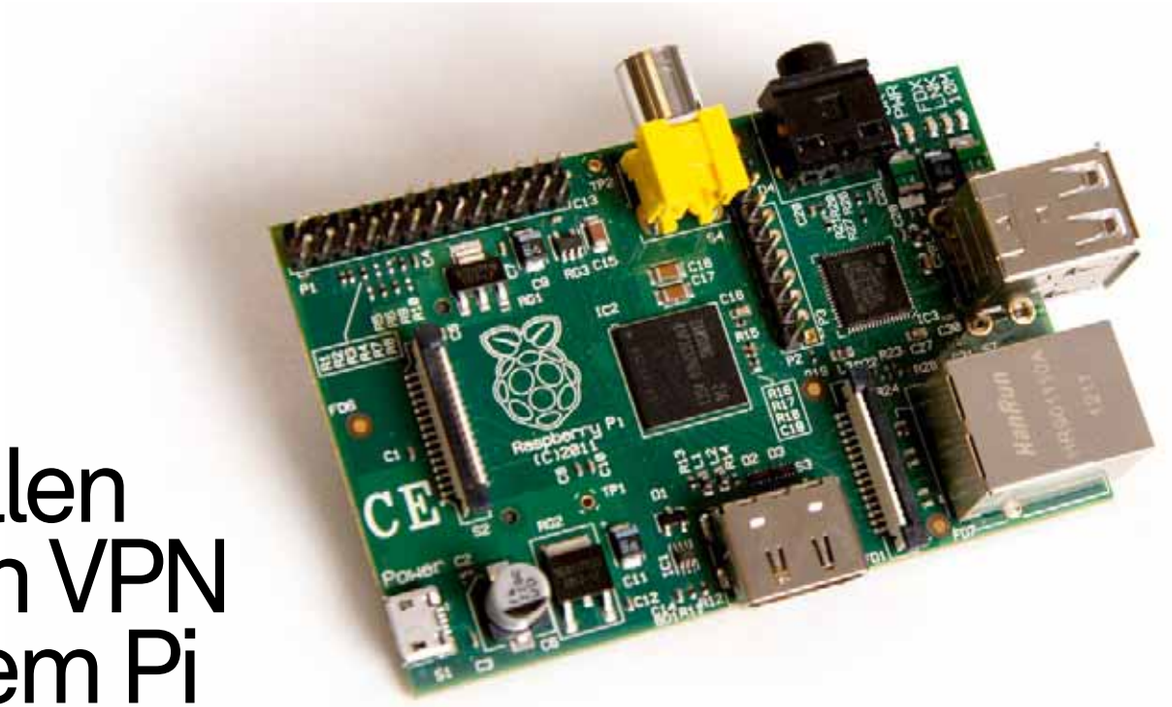
Vom PiFace zum Gertboard

PiFace Digital ist designt, um den Raspberry Pi schnell und einfach mit der Welt zu verbinden. Man kann es überall einsetzen, wo ein Schalter einen Stromkreis unterbrechen würde. Dank der Schraubklemmen können Drähte mithilfe eines Schraubenziehers angeschlossen und getrennt werden – kein Löten notwendig. Mit den Schaltrelais kann es auch als ein Satz vollisolierter, Raspberry-Pi-gesteuerter Schalter gesehen werden. Acht offene Kollektor-Ausgänge (0,5 A, 50 V max.) und acht Eingänge (es ist auch möglich, das in 16 Ausgänge umzukonfigurieren) werden bereitgestellt. PiFace hat die Grundfläche einer Kreditkarte und passt perfekt auf den Raspberry Pi. Manche Raspberry-Pi-Gehäuse können mit nur minimalen Änderungen verwendet werden.

Die acht Benachrichtigungs-LEDs zeigen den Status der Ausgänge, und es gibt vier Druckknöpfe zum Testen der Eingänge, die das Debuggen von Hardware erleichtern. Begleitende Software und Bibliotheken für Python, Scratch und C machen auch das Programmieren einfach. Mit dem Simulator/Emulator kann man Software entwickeln, ohne dass die Hardware angeschlossen ist. Eines der Designziele war, dass es möglich sein sollte, das PiFace mit dem Pi zu verbinden und innerhalb von zehn Sekunden sein erstes Programm zu schreiben, das eine Eingabe liest und einen Ausgang anspricht.

Das Gertboard ist ein Bausatz für fortgeschrittene Bastler, die mit Elektronik experimentieren möchten. Es vermittelt Lötfähigkeiten, denn die Platine muss vor der Benutzung aufgebaut werden. Es hat sechs offene Kollektor-Ausgänge (0,5A, 50V max., gesteuert von demselben Chip wie beim PiFace), 12 digitale Ein-/Ausgänge, zwei analoge Ein-/Ausgänge und eine Motorsteuerung. Es besitzt keine Relais, aber die Anleitung zeigt, wie man einen Schaltkreis bauen kann. Verbindungen werden über Stiftleisten realisiert. Das Gertboard bietet mehr Möglichkeiten beim Experimentieren mit Elektronik, ist aber auch größer als der Pi und wird deshalb nicht aufgesteckt, sondern via Flachbandkabel verbunden.

Das PiFace Digital ist ideal für Anfänger und zum schnellen Steuern digitaler Ein- und Ausgänge. Das Gertboard ist fortgeschrittener und bietet erweiterte Möglichkeiten. Es besteht die Hoffnung, dass Benutzer, die mit dem PiFace Digital anfangen, mehr lernen möchten und deshalb irgendwann zum Gertboard greifen.



Erstellen Sie ein VPN mit dem Pi

Da Anschaffungspreis und Betriebskosten so niedrig sind, eignet sich der Raspberry Pi hervorragend als Server in einem Netzwerk, auf das man aus der Ferne zugreifen möchte.

Was Sie brauchen:

- einen Raspberry Pi samt Peripherie
  raspberrypi.org
- eine SD-Karte mit einem Image des neuesten Arch Linux für den Raspberry Pi
  raspberrypi.org/downloads
- einen zweiten Linux-Rechner als VPN-Client

Ein mögliches Szenario dafür, dass Sie einen kostengünstigen Server haben möchten, den Sie dauerhaft irgendwo hinstellen können, ist, dass Sie sich unter der Woche aus beruflichen Gründen an einem Zweitwohnsitz aufhalten, aber auch von dort aus gerne jederzeit Zugriff auf alle Geräte in Ihrem Heimnetzwerk hätten. So möchten Sie vielleicht Dateien direkt an Ihren PC zu Hause senden oder von dort abrufen, Fehlerdiagnosen durchführen und Ähnliches. Wenn Sie dazu noch einen USB-Hub mit eigenem Netzteil an den Raspberry Pi hängen, kann daheim jemand einen USB-Stick oder eine externe Festplatte einstöpseln, und Sie packen dann aus der Ferne Dateien auf diese Speichermedien. Ein Virtuelles Privates Netzwerk (VPN) also!

Wir benutzen Arch Linux als Betriebssystem für unseren VPN-Server à la Raspberry Pi, da es ressourcenschonend läuft und nur ein Mindestmaß an Paketen benötigt, um zu funktionieren. Der Server selbst umfasst die folgenden Softwarekomponenten:

Arch Linux – Betriebssystem
OpenVPN – Software zum Aufbau eines VPN
Netcfg – Interface für die komfortable Verwaltung mehrerer Netzwerkadapter
bridge-utils – Bridge für VPN- und Ethernet-Adapter
SSH – Protokoll für den sicheren Fernzugriff auf den Raspberry Pi und sein Dateisystem
Dynamischer DNS-Daemon (No-IP) – Software, die im Hintergrund läuft und einen Domainnamen an die IP-Adresse des Routers weiterleitet, sodass der Raspberry Pi von überall mit einer leicht zu merkenden URL angesteuert werden kann

Für die Zwecke dieses Tutorials gehen wir davon aus, dass Sie das aktuelle Arch-Linux-ARM-Image auf eine SD-Karte überspielt haben (eine Anleitung – in englischer Sprache – finden Sie unter **linuxuser.co.uk/tutorials/how-to-set-up-raspberry-pi/**, wobei Sie die Informationen allerdings leicht anpassen müssen, da diese mit Blick auf Debian zusammengestellt wurden).

01 In Arch Linux einloggen

Verkabeln Sie den Raspberry Pi und warten Sie auf den Login-Prompt von Arch Linux. Loggen Sie sich mit dem Standard-Usernamen („root“) und dem Standard-Passwort (ebenfalls „root“) ein. Später werden wir das Passwort ändern.

02 System aktualisieren

Arch Linux wird nicht nach Versionsnummern, sondern als fortlaufend aktualisierte Software veröffentlicht. Der Paketmanager von Arch Linux heißt pacman. Mit dem Befehl „pacman -Syu“ können Sie ein vollständiges System-Update durchführen. Planen Sie dafür allerdings etwas Zeit ein: Erstens sind die Arch-Linux-ARM-Server leider häufig überlastet, zweitens müssen eine Menge Pakete übertragen werden, und drittens sitzt das Betriebssystem beim Raspberry Pi ja auf einer SD-Karte.

03 Nötige Pakete installieren

Auf Seite 102 haben wir die Softwarekomponenten des VPN-Servers aufgelistet. Diese holen Sie sich mit:

```
pacman -S noip netcfg bridge-utils openvpn
```

Beantworten Sie alle Nachfragen mit „y“.

04 Über Subnetze

Eine Sache, auf die hingewiesen werden sollte: Da wir eine Bridge zwischen Client und Site einrichten, verbinden wir den Client mit dem Netzwerk des Servers. Daraus folgt, dass das Subnetz, in dem der Server sich befindet, ein anderes sein muss als das des Clients. Wäre es dasselbe Subnetz, dann gäbe es einen Routing-Konflikt, da die Maschine nicht weiß, ob es sich jeweils um eine lokale Adresse handelt oder auf eine im VPN. Es ist daher eine gute Vorgehensweise, ein Nicht-Standard-Subnetz zu benutzen. In unserem Fall hat der Client ohnehin ein Nicht-Standard-Subnetz, daher spielt das Subnetz des Servers an dieser Stelle keine Rolle. Später sollten wir aber das Subnetz des Servers dennoch ändern, denn Sie könnten sich ja vielleicht mal von einem öffentlichen WLAN aus in Ihr VPN einloggen wollen, und Ersteres könnte sehr wohl ein Standard-Subnetz verwenden. Es empfiehlt sich, ein /24-Subnetz (Subnetzmaske 255.255.255.0) in einem der privaten Adressbereiche zu nehmen:

10.0.0.0 – 10.255.255.255
172.16.0.0 – 172.31.255.255
192.168.0.0 – 192.168.255.255

Die erforderlichen Änderungen nehmen Sie im Einstellungsmenü Ihres Wireless-Routers vor.

05 Netzwerk-Informationen

Wir legen Ihnen sehr ans Herz, dass Sie Ihrem als Server fungierenden Raspberry Pi eine statische IP-Adresse zuweisen, statt dies Ihrem Router zu überlassen. So wissen Sie nämlich immer genau, wo in Ihrem Netzwerk Sie den Pi finden, was sinnvoll ist, wenn Sie von einem anderen Gerät auf ihn zugreifen möchten. Außerdem brauchen Sie eine statische IP-Adresse, wenn Sie den Pi via Internet ansteuern möchten. Um eine solche Adresse festlegen zu können, müssen wir zunächst einige Informationen über Ihr aktuelles Netzwerk ermitteln. Tun Sie dies mit den Kommandos „ifconfig eth0“ und „ip route show“.

```
[root@alarmpi ~]# ifconfig eth0
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.215  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::ba27:ebff:fee2:359f  prefixlen 64  scopeid 0x20<link>
        ether b8:27:eb:e2:35:9f  txqueuelen 1000  (Ethernet)
        RX packets 144025  bytes 97598225 (93.0 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 39428  bytes 3257546 (3.1 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

[root@alarmpi ~]# ip route show
default via 192.168.1.254 dev eth0  metric 202
192.168.1.0/24 dev eth0  proto kernel  scope link  src 192.168.1.215  metric 202
[root@alarmpi ~]#
```

06 Statische IP-Adresse

Nun da Sie die Informationen über Ihr Netzwerk (wie aktuelle IP-Adresse oder Netzwerkmaske) beisammen haben, können Sie eine statische IP-Adresse vergeben. Wir benutzen das Netcfg-Framework für Arch Linux, um die Netzwerkverbindungen zu verwalten, da wir im Endeffekt drei verschiedene davon brauchen: erstens Ethernet, zweitens einen VPN-TAP-Adapter sowie drittens einen Bridge-Adapter, der die ersten beiden verbindet.

Wechseln Sie ins Verzeichnis **/etc/network.d** und erstellen Sie dort mit dem Editor eine neue Datei namens **bridge**:

```
cd /etc/network.d
nano bridge
```

Tragen Sie in die neue Datei die Konfiguration für die Bridge ein (eigene Netzwerkdaten verwenden!):

```
INTERFACE="br0"
CONNECTION="bridge"
DESCRIPTION="VPN Bridge connection"
BRIDGE_INTERFACES="eth0"
IP='static'
ADDR='192.168.1.215'
NETMASK='24'
GATEWAY='192.168.1.254'
DNS=('192.168.1.254')
```

Speichern Sie die Datei mit Strg+O und Enter, dann schließen Sie den Editor. Wir fügen den VPN-Adapter später zu der Bridge hinzu.

Jetzt müssen wir noch festlegen, welche Profile Netcfg laden soll, indem wir die Datei **/etc/conf.d/netcfg** entsprechend editieren. Konfigurieren Sie das Netzwerk-Array wie folgt:

```
NETWORKS=(bridge)
```

Speichern Sie die Änderungen, und geben Sie dann im Terminal die folgenden Kommandos ein, um damit DHCP zu deaktivieren und die Ethernet-Schnittstelle sowie die Bridge mittels einer statischen IP-Adresse permanent zu aktivieren:

```
systemctl disable dhcpcd@eth0.service
systemctl enable netcfg.service
```

Führen Sie einen Neustart des Raspberry Pi durch, damit die Änderungen wirksam werden.

07 Mit SSH einloggen

Öffnen Sie nach erneutem Reboot auf Ihrem Linux-PC ein Terminal und tippen Sie „ssh root@[IP-Adresse des Pi]“. Bestätigen Sie, dass Sie eine Verbindung herstellen möchten, und geben Sie dann das Passwort ein, das nach wie vor „root“ lautet. Daraufhin sind Sie über SSH auf dem Pi eingeloggt.

```
[root@alarmpi ~]# passwd
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
```

08 Root-Passwort ändern

Da Sie vermutlich ein SSH-Login über das Internet zulassen werden, ist es äußerst ratsam, ein eigenes, sicheres Passwort zu vergeben. Tippen Sie „passwd“ und folgen Sie dann den Anweisungen auf dem Bildschirm. Ihre SSH-Session bleibt bestehen, aber bei der nächsten Anmeldung müssen Sie das neue Passwort verwenden. Vielleicht möchten Sie außerdem noch den Inhalt von **/etc/hostname** ändern, indem Sie den Hostnamen etwas aussagekräftiger gestalten. Standardmäßig ist dort „alarmpi“ eingetragen. Sie könnten etwa „vpnserver“ daraus machen. Die Änderung wird erst nach einem Neustart wirksam.

09 Zertifikat-Infrastruktur

Um bei OpenVPN die Authentifizierung vorzunehmen, greifen wir auf eine Zertifikat-Infrastruktur zurück. Für jeden Client werden ein Zertifikat und ein privater Schlüssel (der geheim bleiben muss) generiert und durch die Zertifizierungsstelle signiert. Nur Clients mit signierten Zertifikaten erhalten Zugang. Mithilfe des Schlüssels und des Zertifikats werden die Daten verschlüsselt, die zwischen Client und Server ausgetauscht werden. Dank dieses Sicherheitsansatzes ist eine zusätzliche

»

Authentifizierung mit Benutzernamen und Passwort auf dem Client nicht notwendig. Es gibt eine Reihe von Skripten, die die Einrichtung einer solchen Zertifikat-Infrastruktur erleichtern. Kopieren Sie die Skripte mit folgendem Befehl in den Ordner **/etc/openvpn**:

```
cp -r /usr/share/openvpn/easy-rsa/ /etc/openvpn
```

Wechseln Sie dann in den entsprechenden Ordner.

Wir erstellen ein Template, auf dessen Basis wir die Zertifikate generieren können. Öffnen Sie mit dem Editor die Variablen-Datei **vars**, und ändern Sie die nachfolgenden Einträge. So ungefähr sehen die alten Zeilen aus:

```
export KEY_COUNTRY="US"
export KEY_PROVINCE="CA"
export KEY_CITY="SanFrancisco"
export KEY_ORG="Fort-Funston"
export KEY_EMAIL="me@myhost.mydomain"
export KEY_EMAIL=mail@host.domain
export KEY_CN=changeme
export KEY_NAME=changeme
export KEY_OU=changeme
```

Und dies wären beispielhaft die neuen Zeilen, die stattdessen eingetragen werden müssten:

```
export KEY_COUNTRY="UK"
export KEY_PROVINCE=""
export KEY_CITY="Ormskirk"
export KEY_ORG="Home"
export KEY_EMAIL="liam@vpn.home.org"
export KEY_CN="liamvpn-ca"
export KEY_NAME="liamvpn-ca"
export KEY_OU="None"
```

Speichern Sie die Variablen-Datei, und exportieren Sie sie dann mit:

```
source ./vars
```

Säubern Sie zum Abschluss noch etwaige vorherige Konfigurationen:

```
./clean-all
```

10 Zertifikate erstellen

Fangen Sie mit dem Zertifikat der Zertifizierungsstelle an, mit dem wir später alles andere signieren werden (durch Drücken der Eingabetaste lassen Sie einzelne Felder unverändert):

```
./build-ca
```

Danach machen wir mit der Erstellung eines Server-Zertifikats weiter:

```
./build-key-server [Server-Hostname]
```

Drücken Sie Enter, wenn Sie um irgendwelche Angaben gebeten werden, tragen Sie weder Passwort noch Firmennamen ein, und akzeptieren Sie die Anfrage, das Zertifikat zu signieren.

Nun müssen wir die Parameter generieren, die erforderlich sind, damit zwei Benutzer einen geheimen Schlüssel über ein unsicheres Medium austauschen können – es handelt sich bei dem Protokoll um den sogenannten Diffie-Hellman-Schlüsselaustausch (dies kann einige Minuten dauern):

```
./build-dh
```

Der letzte Schritt besteht darin, ein Zertifikat für jeden Client zu erzeugen, der sich mit dem VPN verbinden können soll. Als Beispiel dient uns hier ein Laptop:

```
./build-key liam-laptop
```

Wiederholen Sie hierbei einfach Ihre Vorgehensweise während des Skripts build-key-server, und dann haben Sie alle Zertifikate, die Sie benötigen.

11 OpenVPN-Server einrichten

Wir stellen unsere Konfigurationsdatei auf Basis der Beispieldatei für die Servereinstellungen zusammen. Kopieren Sie zunächst mit „cp /usr/share/openvpn/examples/server.conf /etc/openvpn/server.conf" die Beispieldatei. Öffnen Sie dann **/etc/openvpn/server.conf** im Editor und ändern Sie als Erstes diese beiden Zeilen:

```
;dev tap
dev tun
```

Ersetzen Sie sie durch:

```
dev tap0
;dev tun
```

Wir verwenden also einen TAP-Adapter, um die Netzwerke per Bridge zu koppeln, statt TUN.

Ersetzen Sie die vorhandenen Zertifikate mit denen, die Sie selbst generiert haben. So sah bei uns die alte Liste aus:

```
ca ca.crt
cert server.crt
key server.key  # This file should be
kept secret
dh dh1024.pem
```

Und so die neue Konfiguration:

```
ca /etc/openvpn/easy-rsa/keys/ca.crt
cert /etc/openvpn/easy-rsa/keys/liamvpn.
crt
key /etc/openvpn/easy-rsa/keys/liamvpn.
key
dh /etc/openvpn/easy-rsa/keys/dh1024.pem
```

Kommentieren Sie außerdem diese Zeile aus, indem Sie ein „;" (Semikolon) an den Anfang setzen:

```
server 10.8.0.0 255.255.255.0
```

Dies ist notwendig, da wir eine Ethernet-Bridge haben möchten und keinen regulären Server. Die Ethernet-Bridge aktivieren wir, indem wir die Auskommentierung folgender Zeile rückgängig machen, also das Semikolon entfernen:

```
server-bridge 10.8.0.4 255.255.255.0
10.8.0.50 10.8.0.100
```

Passen Sie dann die Werte so an, dass sie der Netzwerk-Konfiguration Ihres Servers entsprechen. Der erste Wert bezeichnet die IP-Adresse des Servers, der zweite die Subnetzmaske. Die beiden letzten Werte geben Anfang und Ende des Bereichs an, aus dem IP-Adressen an sich verbindende Clients vergeben werden.

Machen Sie dann noch die Auskommentierung zweier Zeilen rückgängig:

```
;user nobody
;group nobody
```

Dadurch ordnen Sie OpenVPN die geringstmöglichen Benutzerrechte zu. Speichern Sie die geänderte Datei.

12 TAP-Interface konfigurieren

Öffnen Sie im Editor die Datei **/etc/network.d/tap**, fügen Sie die nachfolgenden Zeilen ein, und speichern Sie die Änderung:

```
INTERFACE='tap0'
CONNECTION='tuntap'
MODE='tap'
USER='nobody'
GROUP='nobody'
```

Da wir noch unserer Bridge das tap0-Interface hinzufügen müssen, editieren wir auch **/etc/network.d/bridge**, und zwar so, dass folgende Zeile darin steht:

```
BRIDGE_INTERFACES="eth0 tap0"
```

Ändern Sie zuletzt noch eine Zeile in **/etc/conf.d/netcfg**, und zwar so:

```
NETWORKS=(tap bridge)
```

Beachten Sie, dass zuerst das TAP-Netzwerk gestartet werden muss, damit es erfolgreich der Bridge hinzugefügt werden kann.

13 OpenVPN aktivieren

OpenVPN ist damit konfiguriert, jetzt möchten wir es dauerhaft aktivieren. Geben Sie das Kommando „systemctl enable openvpn@server" ein und starten Sie dann den Raspberry Pi neu. Wir richten nun dynamisches DNS und eine Portweiterleitung ein, um auf das VPN über das Internet zugreifen zu können.v

14 Dynamisches DNS einrichten

Gehen Sie auf die Website **no-ip.com** und legen Sie dort ein Gratiskonto für private Zwecke an. Sie brauchen jedoch nicht den Client von No-IP herunterzuladen, sondern können unseren benutzen. Klicken Sie in der E-Mail, die Sie nach der Registrierung bekommen haben, auf den Aktivierungslink. Nun können Sie sich in Ihr Benutzerkonto einloggen. Wählen Sie dort die Option „Add a host", legen Sie einen Hostnamen

fest, und suchen Sie sich eine der angebotenen Domains aus der Dropdown-Liste aus. Stellen Sie „DNS host“ als Hosttyp ein, und klicken Sie auf „Create host“. Wir haben „liam-ludtest“ mit der Domain no-ip.org ausgewählt, somit läuft der Zugang über **liam-ludtest.no-ip.org**.

15 No-IP konfigurieren

Tippen Sie das Kommando

```
noip2 -C -Y
```

ein, um sich durch die interaktive Konfigurationsroutine des No-IP-Clients führen zu lassen. Wir haben das Update-Intervall bei den vorgegebenen 30 Minuten belassen. Das bedeutet, dass der Client alle 30 Minuten nach einer möglichen Änderung der IP-Adresse Ausschau hält. Wenn Sie fertig sind, starten Sie den Daemon mit:

```
/etc/rc.d/noip start
```

Nach wenigen Minuten wird Ihre IP-Adresse über Ihren No-IP-Hostnamen erreichbar sein. Wenn Sie dies allerdings von innerhalb Ihres Heimnetzwerks versuchen, werden Sie vermutlich lediglich auf der Homepage Ihres Routers landen.

Abb. 1 NAT-Portweiterleitung

16 NAT-Portweiterleitung

Vermutlich hängen mehrere Geräte hinter Ihrem Router, die alle dieselbe externe IP-Adresse haben. Dies liegt einerseits an der Knappheit von IPv4-Adressen, andererseits an Schutzaspekten, da es sicherer ist, das Internet von Ihrem internen Heimnetzwerk zu separieren. Beim NAT-Verfahren (Network Address Translation) wird ein Port von der externen IP-Adresse des Routers an einen Rechner im LAN weitergeleitet. In unserem Fall möchten wir jeglichen Traffic für den TCP-Port 22, der an der externen IP-Adresse des Routers ankommt, an die IP-Adresse des Raspberry Pi weiterleiten. Der TCP-Port 22 ist der Port, der für SSH genutzt wird. SSH bewerkstelligt den Fernzugriff auf Ihren Raspberry Pi – sowie dessen Dateisystem per SCP (Secure Copy Protocol). Sie sollten auch den UDP-Port 1194 forwarden, da dieser von OpenVPN verwendet wird.

Die konkrete Konfiguration der Portweiterleitung hängt stark von Ihrem Routermodell ab, Sie müssen daher vermutlich in dessen Anleitung nachsehen. Das entsprechende Menü finden Sie wahrscheinlich in den erweiterten Einstellungen Ihres Geräts. Sie sollten Ihren Router ansprechen können, indem Sie Ihren No-IP-Hostnamen in die Adresszeile des Browsers eingeben, andernfalls mit der Adresse des Standard-Gateways.

In Abb. 1 sehen Sie beispielhaft, wie das Menü für die Portweiterleitung im Router aussehen kann.

17 OpenVPN-Client installieren

Als Test-Client für das VPN soll uns eine virtuelle Maschine mit Ubuntu 12.04 als Betriebssystem dienen. Die Zahl möglicher Kombinationen ist sehr groß, doch bestimmte Bedingungen müssen auf jedem Client erfüllt sein, und zwar:

- Nutzung als TAP-Gerät
- LZO-Datenkompression
- keine Nutzung des Standard-Gateways im Remote-Netzwerk (dies ist notwendig, da andernfalls die Internetverbindung des Clients nicht funktionieren würde, denn wir haben das VPN nicht für die Internetnutzung konfiguriert)

Ubuntu konfiguriert seine Netzwerke mit dem Network Manager, daher sollten die Instruktionen, die wir hier geben, entsprechend auch für andere Distributionen gelten, die dieses Tool verwenden. Standardmäßig beinhaltet Ubuntu kein OpenVPN-Plugin für den Network Manager, wir müssen dieses also selbst installieren. Geben Sie in die Kommandozeile ein:

```
sudo apt-get update
sudo apt-get install network-manager-
openvpn-gnome
```

18 Erforderliche Zertifikate

Um erfolgreich eine Verbindung herzustellen, benötigen wir diese drei Dateien vom Raspberry Pi:

Zertifikat der Zertifizierungsstelle
Client-Zertifikat
Client-Schlüssel

Per SCP kopieren wir die Dateien in den Ordner **/etc/openvpn/keys**:

```
cd /etc/openvpn
sudo mkdir keys
cd keys
sudo scp root@[Pi's IP address]:/etc/
openvpn/easy-rsa/keys/ca.crt .
sudo scp root@[Pi's IP address]:/etc/
openvpn/easy-rsa/keys/[client].crt .
sudo scp root@[Pi's IP address]:/etc/
openvpn/easy-rsa/keys/[client].key .
sudo chmod +r *
```

Beachten Sie, das wir mithilfe von „chmod“ die Leseberechtigung hinzugefügt haben, da die Dateien für alle Benutzer lesbar sein müssen. Dieses Vorgehen wird dadurch notwendig, dass das grafische Interface des Network Managers nicht als Root-User läuft.

19 VPN-Verbindung herstellen

Um das bereits erwähnte mögliche Problem beim Routing (siehe Schritt 04) zu vermeiden, ist es ratsam, mit dem Client ein anderes Subnetz zu gebrauchen als der Server. Eine Lösung, die wir gefunden haben, beinhaltete die Nutzung einer virtuellen Maschine mit NAT-Verbindung im Subnetz 10.0.2.0/24.

Klicken Sie auf das Netzwerk-Icon in der Menüleiste oben auf dem Bildschirm und wählen Sie „Edit connections“. Es erscheint ein Fenster mit mehreren Tabs, darunter einer für VPN. Klicken Sie dort auf die Hinzufügen-Option, wählen Sie OpenVPN als Verbindungstyp, und klicken Sie auf „Create“. Tragen Sie dann die relevanten Angaben ein.

20 Erweiterte Einstellungen

Bei den erweiterten Einstellungen müssen noch die in Schritt 17 erwähnten Bedingungen erfüllt werden:

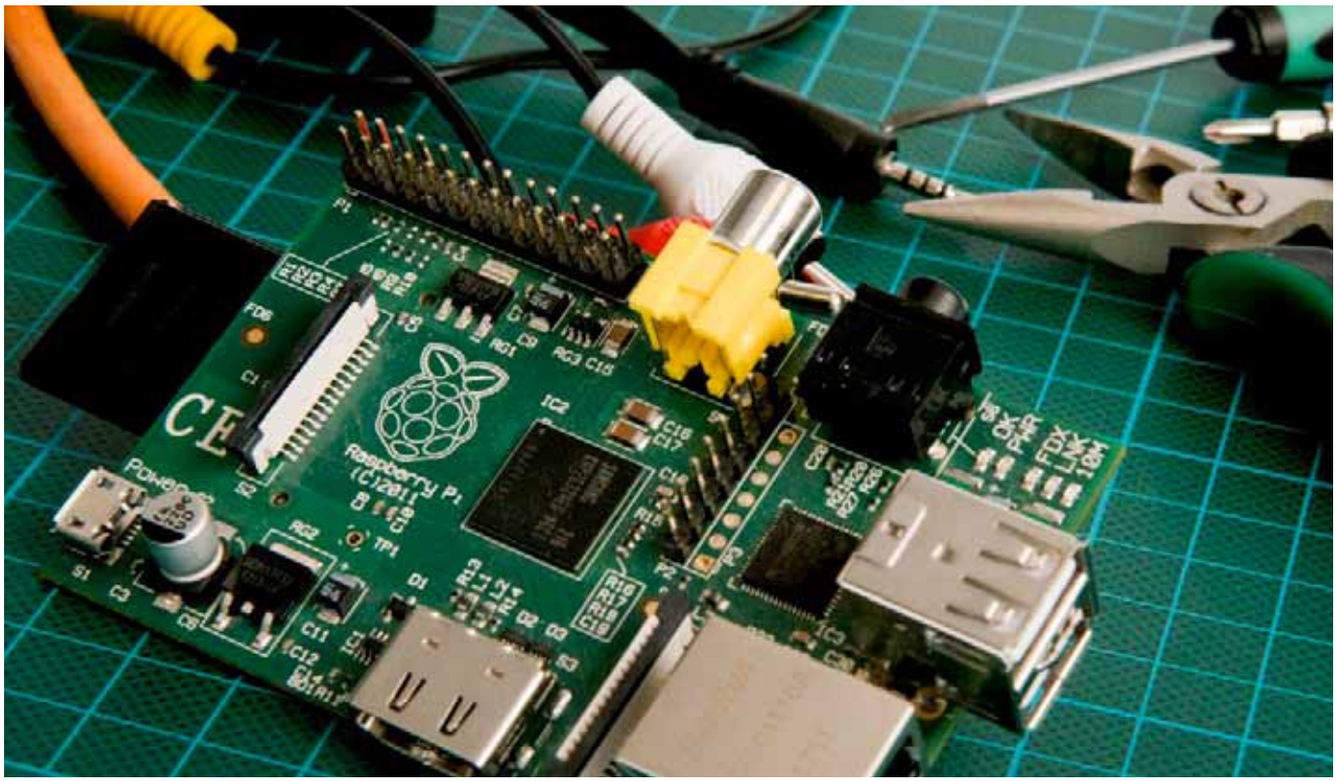
- Nutzung als TAP-Gerät
- LZO-Datenkompression

21 Route-Einstellungen

Schließlich müssen wir noch die Nutzung des Standard-Gateways im Remote-Netzwerk unterbinden. Bei Ubuntu wird dies durch Anhaken der Option „Use this connection only for resources on its network“ im Menü IPv4 Settings > Routes gewährleistet. Danach können Sie Ihre Verbindung speichern und das Fenster mit den Netzwerkverbindungen schließen.

22 Verbindung testen

Klicken Sie im Menü auf das Netzwerk-Icon, fahren Sie mit der Maus über die Option VPN-Verbindungen, und klicken Sie auf das VPN, das sie eben erstellt haben. Sie sollten eine Bestätigungsnachricht angezeigt bekommen, und das Netzwerk-Icon sollte nun mit einem Vorhängeschloss gekennzeichnet sein. Öffnen Sie ein Terminal und führen Sie ifconfig aus, um zu überprüfen, ob das TAP-Gerät eine korrigierte IP-Adresse erhalten hat und ob Sie ein Gerät im VPN anpingen können.



Emulieren Sie eine Bluetooth-Tastatur mithilfe des Pi

Lernen Sie, wie Sie Input von einem USB-Keyboard über den Raspberry Pi an ein Bluetooth-fähiges Gerät senden.

Was Sie brauchen:

- einen Raspberry Pi samt Peripherie
  raspberrypi.org
- einen USB-Bluetooth-Dongle
- einen USB-Hub
  So können Sie parallel zum Bluetooth-Dongle eine Tastatur und eine Maus anschließen
- ein Gerät, an das Sie das virtuelle Keyboard anschließen können
  Wir haben ein Android-Smartphone und einen Linux-Rechner benutzt

In diesem Tutorial erfahren Sie, wie Sie einen Raspberry Pi so präparieren, dass er wie ein Bluetooth-Keyboard funktioniert. Zu diesem Zweck schreiben Sie ein Programm, das Input an ein Client-Gerät weiterleitet. Gleichzeitig bekommen Sie so einen Einblick in die Funktionsweise des Bluetooth-Protokolls und von Sockets und erfahren überdies etwas über Binärdaten und deren unterschiedlichen Erscheinungsformen.

Wir verwenden Raspbian als Betriebssystem für das virtuelle Bluetooth-Keyboard. Das jeweils aktuelle Raspbian-Image bekommen Sie auf **raspberrypi.org/downloads**. Falls Sie noch nicht wissen, wie Sie das Image auf eine SD-Karte überspielen, die dann in den Raspberry Pi eingesteckt wird, hilft Ihnen beispielsweise die Seite **raspberrypiguide.de** weiter.

In Abhängigkeit davon, wie wohl Sie sich bei der Arbeit in einer terminalbasierten Umgebung (Kommandozeile) fühlen, haben Sie verschiedene Möglichkeiten, dieses Tutorial durchzuführen. Wir haben SSH verwendet. Sollten Sie das nicht wollen und auch nicht gerne das Terminal des Raspberry Pi benutzen, dann können Sie ebenso gut die LXDE-Desktop-Umgebung einsetzen. In dem Fall (wie auch beim Pi-Terminal) brauchen Sie allerdings einen USB-Hub, da Sie neben dem Bluetooth-Dongle noch Tastatur und Maus anschließen müssen.

01 Raspberry Pi vorbereiten

Schließen Sie das Stromkabel, das Netzwerkkabel, den Bluetooth-Dongle und eine USB-Tastatur an den Raspberry Pi an. Für die anfängliche Konfiguration benötigen Sie ferner einen Bildschirm, auch wenn Sie danach per SSH weiterarbeiten können. Möchten Sie lieber die LXDE-Oberfläche des Raspberry Pi verwenden, brauchen Sie außer Bildschirm und Tastatur noch eine Maus. Wenn Raspbian gebootet hat, wird ein Konfigurationsmenü angezeigt. Wählen Sie die Option, das Dateisystem zu erweitern, um Platz für Extrapakete zu schaffen. Speichern Sie und booten Sie erneut.

02 In den Raspberry Pi einloggen

Während Raspbian hochfährt, wird die aktuelle IP-Adresse Ihres Raspberry Pi angezeigt. Diese brauchen Sie, um eine SSH-Verbindung herzustellen. Öffnen Sie dazu ein Terminal auf dem Rechner und tippen Sie Folgendes ein:

```
ssh pi@[IP-Adresse des Pi]
```

Bestätigen Sie die Abfrage, ob eine Verbindung aufgebaut werden soll, und geben Sie anschließend das Passwort „raspberry“ ein. Stattdessen können Sie sich auch beim Login den Benutzernamen „pi“ und das Passwort eingeben.

03 Benötigte Pakete installieren

Geben Sie nachfolgende Befehle ein, um die benötigten Pakete zu installieren:

```
sudo apt-get update
sudo apt-get install python-gobject
bluez bluez-tools bluez-firmware
python-bluez python-dev python-pip
sudo pip install evdev
```

Die Abarbeitung der Kommandos kann einige Zeit in Anspruch nehmen.

Der erste Befehl sorgt für ein Update der Liste verfügbarer Pakete. Der zweite installiert die notwendigen Bluetooth-Pakete samt Python-Bindungen. Die übrigen Pakete brauchen wir für die Installation eines Python-Moduls namens evdev, mit dem der Keyboard-Input erfasst wird. Da evdev nicht standardmäßig in den Raspbian-Repositorys vorhanden ist, müssen wir das Python-Entwicklerpaket an den Start bringen und mithilfe des Python-Paketmanagers pip das evdev-Modul aus dem Quellcode kompilieren, bevor wir es installieren können.

04 Unerwünschte Bluetooth-Plugins deaktivieren

Wir haben jetzt die Linux-Bluetooth-Implementierung BlueZ installiert. Standardmäßig sind hier einige Dienste aktiviert, die wir nicht benötigen und die bei einer etwaigen Fehlerbehebung nur stören würden. Folgender Befehl zeigt die Standarddienste an:

```
sudo sdptool browse local
```

Um die unerwünschten Dienste abzuschalten, öffnen Sie in einem Editor wie nano die Datei **/etc/bluetooth/main.conf** und ersetzen Sie die Zeile

```
#DisablePlugins = network,input
```

durch diese hier:

```
DisablePlugins = network,input,audio
,pnat,sap,serial
```

Speichern Sie die Änderungen durch Drücken von Strg+O, Eingabetaste, Strg+X. Damit die Änderungen wirksam werden, müssen Sie den Bluetooth-Daemon erneut starten:

```
sudo /etc/init.d/bluetooth restart
```

Um sicherzugehen, dass die Dienste nicht mehr da sind, können Sie noch Folgendes eingeben:

```
sudo sdptool browse local
```

> **“ Als Erstes die notwendigen Module importieren! ”**

05 Projekt beginnen

Wenden wir uns nun dem eigentlichen Programmieren zu. Zunächst sollten Sie ein Projektverzeichnis anlegen. Gehen Sie in den Ordner **/home/pi** und erzeugen Sie dort mithilfe des Befehls `mkdir PiTooth` ein neues Verzeichnis. Wechseln Sie in das soeben erstellte Verzeichnis **PiTooth** und generieren Sie darin mit dem Kommando `touch PiTooth.py` eine Projektdatei. Diese müssen Sie dann noch mittels `chmod +x PiTooth.py` ausführbar machen. Öffnen Sie im Editor nano die Datei und starten Sie mit folgender Codezeile:

```
#!/usr/bin/python2.7
```

So weiß die Shell, dass sie den Python-2.7-Interpreter verwenden soll.

Sie können auch noch einen kurzen Kommentar über das Programm einfügen:

```
#!/usr/bin/python2.7
#
# Mit PiTooth lässt sich der
Raspberry Pi als Bluetooth-Keyboard
verwenden.
# Tastatureingaben werden von einem
USB-Keyboard zu einem Bluetooth-
Client
# weitergeleitet.
#
```

06 Benötigte Dateien downloaden

Damit das Projekt funktioniert, sind noch weitere Dateien vonnöten. Zwar könnten Sie all diese selber erstellen, doch würde das eine umfassende Einarbeitung in diverse Protokollspezifikationen erfordern. Wir haben Ihnen darum die Arbeit abgenommen und die entsprechenden Dateien bereits für Sie vorbereitet. Speichern Sie Ihren bisherigen Code und laden Sie dann mit dem Kommando `wget http://li-amfraser.co.uk/lud/PiTooth-Resources.zip` besagte Dateien herunter.

Entpacken Sie das Zip-Archiv mit dem Befehl `unzip PiTooth-Resources.zip` und stellen Sie mittels ls sicher, dass die neuen Dateien tatsächlich vorhanden sind. Danach können Sie das Zip-File wieder entfernen:

```
rm PiTooth-Resources.zip
pi@raspberrypi ~/PiTooth $ ls
keymap.py PiTooth.py sdp_record.
xml
```

07 Module importieren

Zurück zu nano und unserem Code. Dieser muss als Erstes die notwendigen Module importieren:

```
import os
import sys
import bluetooth
from bluetooth import *
import dbus
import time
import evdev
from evdev import *
import keymap
```

Die Module haben folgende Funktion: os ruft externe Befehle auf, sys beendet das Skript, dbus (D-Bus) stellt den SDP-Eintrag („Service Discovery Protocol“) auf, time unterbricht den Prozess, evdev erfasst den Keyboard-Input.

08 Bluetooth-Klasse

Wir starten unsere Bluetooth-Klasse wie üblich mit einer Initialisierungsfunktion. Da

»

wir keine elaborierte Entwicklungsumgebung verwenden, sollten Sie Sorge tragen, dass Ihr Code ordentlich formatiert ist. Vier Leerzeichen sind für Einrückungen sicherer als ein Tabulator, da dieser nicht von jedem Editor identisch umgesetzt wird.

Präparieren Sie zuerst den Bluetooth-Dongle für die Verwendung mit einem Bluetooth-Keyboard. Benutzen Sie dafür den Befehl hciconfig, das Bluetooth-Äquivalent von ifconfig.

Nun legen wir die Geräteklasse – sie besteht aus einem Hexadezimalwert für das Keyboard – sowie den Gerätenamen Raspberry Pi fest und machen das Gerät zum Zwecke der Verbindungsaufnahme (Pairing) mit anderen Geräten auffindbar:

```
class Bluetooth:
    def __init__(self):
        # Geräteklasse benennen
und auf Keyboard setzen
        os.system("hciconfig hci0
class 0x002540")
        os.system("hciconfig hci0
name Raspberry\ Pi")
        # Gerät auffindbar machen
        os.system("hciconfig hci0
piscan")
```

09 Bluetooth-Sockets

Bluetooth-HID-Geräte kommunizieren mithilfe zweier Sockets: einem Control- und einem Interrupt-Socket. Der Control-Socket initiiert die Verbindung, sendet Handshakes und Ähnliches. Der Interrupt-Socket sendet Daten geringer Latenz wie Input-Berichte über Tastensignale oder Output-Berichte über beispielsweise (bei einem Gamepad) Vibrationsfeedback. Die Standard-Ports für diese Sockets sind 17 (Control) und 19 (Interrupt).

Bevor wir die Sockets generieren, deklarieren wir noch ein paar Konstanten, die die Portnummern beinhalten. Die Konstanten können außerhalb der Initialisierungsfunktion platziert werden, da sie für jede Instanz der Klasse gleich sind. Der Anfang der Bluetooth-Klasse sollte jetzt folgendermaßen aussehen:

```
class Bluetooth:
    # Ports definieren
    P_CTRL = 17
    P_INTR = 1
```

10 Zertifikate generieren

Wir sind nun wieder in der Initialisierungsfunktion. Nachdem der Bluetooth-Dongle eingerichtet ist, brauchen wir einige Sockets für die Entgegennahme von Client-Verbindungen und müssen sie mit den zugehörigen Ports verbinden. L2CAP ist ein Typ von Bluetooth-Socket. Um die Sockets mit einem Port zu verbinden, schicken wir ein Tupel mit der MAC-Adresse des Dongles und dem Port. Wir greifen auf die Port-Konstanten mit dem Klassennamen zu, statt ihnen ein „self"-Präfix zu geben, denn die Konstanten sind Mitglieder der Klasse und nicht der Instanz.

```
        # Definition der Server-Sockets
für die Kommunikation
        self.scontrol =
BluetoothSocket(L2CAP)
        self.sinterrupt =
BluetoothSocket(L2CAP)

        # Port-Anbindung
        self.scontrol.bind(("",
Bluetooth.P_CTRL))
        self.sinterrupt.bind(("",
Bluetooth.P_INTR))
```

11 D-Bus einsetzen

D-Bus ist ein in Linux häufig verwendetes Framework, das die Kommunikation zwischen Prozessen sowie ferngesteuerte Prozeduraufrufe ermöglicht (indem es eine Funktion in einem anderen Programm aufruft). Die Bluetooth-Implementierung BlueZ beinhaltet eine D-Bus-Schnittstelle, die Zugriff auf einige Dinge bietet, die vom Python-Modul nicht abgedeckt werden. Mithilfe des D-Bus zeigen wir einen Bluetooth-SDP-Eintrag an. SDP steht für „Service Discovery Protocol" und bedeutet, dass Geräte und Dienste im Netzwerk automatisch erkannt werden. Der SDP-Eintrag enthält Informationen über das virtuelle Keyboard wie etwa die Struktur der Input-Berichte (die gesendet werden, wenn eine Taste gedrückt wird) oder die Spracheinstellung des Keyboards.

Wir fügen unseren D-Bus-Code in einen try-except-Block ein, da es gut ist, eine Fehlermeldung parat zu haben, falls etwas schiefgeht. Zunächst holen wir uns eine org.Bluez.Manager-Schnittstelle und den Pfad zum Standard-Adapter. Damit bekommen wir das Service-Interface für den Adapter, wo die SDP-Einträge vorgenommen werden.

```
        # D-Bus zur Anzeige der
Service-Einträge einrichten
        self.bus = dbus.SystemBus()
        try:
          self.manager = dbus.
Interface(self.bus.get_object("org.
bluez", "/"),

"org.bluez.Manager")
          adapter_path = self.
manager.DefaultAdapter()
          self.service = dbus.
Interface(self.bus.get_object("org.
bluez", adapter_path),

"org.bluez.Service")
        except:
          sys.exit("Could not
configure bluetooth. Is bluetoothd
started?")
```

12 SDP-Eintrag lesen

Der D-Bus ist eingerichtet, jetzt muss der Inhalt der XML-Datei, die den SDP-Eintrag beschreibt, in eine Variable eingelesen werden. Auch dies bewerkstelligen wir im Rahmen eines try-except-Blocks, falls das File fehlen sollte. Die Variable sys.path[0] enthält den Pfad zu dem Verzeichnis, wo das Skript liegt.

```
        # Service-Eintrag aus
Datei auslesen
        try:
          fh = open(sys.path[0] +
"/sdp_record.xml", "r")
        except:
          sys.exit("Could not open
the sdp record. Exiting...")
        self.service_record =
fh.read()
        fh.close()
```

"Es ist gut, eine Fehlermeldung parat zu haben, falls was schiefgeht."

13 Nach Verbindung horchen

Der nächste Schritt ist es, eine Horchfunktion zu definieren. Diese fügt den gerade ausgelesenen SDP-Eintrag dem SDP-Server hinzu und wartet dann auf Verbindungen vom Client-Gerät. Jedes Client-Gerät wird sich zuerst mit dem Control-Socket und dann mit dem Interrupt-Socket verbinden. Die accept-Funktion des Sockets gibt einen Socket für diese Verbindung und ein Tupel mit der MAC-Adresse des Clients samt benutztem Port zurück. Die Konstanten dienen dazu, den Code besser lesbar zu machen, wenn auf die Werte im Tupel zugegriffen wird.

```
class Bluetooth:
    HOST = 0 # MAC-Adresse
    PORT = 1 # Portnummer
...
    def listen(self):
        # Service-Eintrag anzeigen
        self.service_handle = self.
service.AddRecord(self.service_record)
        print "Service record added"

        # Beginn Horchen auf
Server-Sockets
        self.scontrol.listen(1) # Limit
of 1 connection
        self.sinterrupt.listen(1)

        print "Waiting for a connection"

        self.ccontrol, self.cinfo =
self.scontrol.accept()
        print "Got a connection on
the control channel from " + self.
cinfo[Bluetooth.HOST]
        self.cinterrupt, self.cinfo =
self.sinterrupt.accept()
        print "Got a connection on
the interrupt channel from " + self.
cinfo[Bluetooth.HOST]
```

14 Hauptfunktion hinzufügen

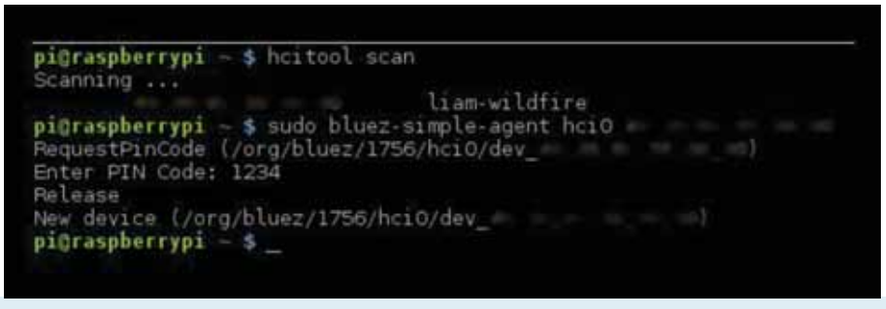
Nun haben wir den Code, um Verbindungen von Geräten anzunehmen. Zwar werden diese Verbindungen gleich wieder fallengelassen, aber das ist zu diesem Zeitpunkt unerheblich. Wir werden am Ende der Datei noch eine Hauptfunktion hinzufügen, um eine Instanz der Bluetooth-Klasse zu erzeugen und die Horchfunktion aufzurufen, die auf Verbindungen wartet. Außerdem machen wir im Rahmen der Hauptfunktion einen Check, ob der Benutzer, der den Code ausführen möchte, root-Rechte besitzt (das ist nötig, da das Skript sonst nicht läuft).

```
if __name__ == "__main__":
    # root-Benutzerrechte notwendig
    if not os.geteuid() == 0:
        sys.exit("Only root can run this
script")

    bt = Bluetooth()
    bt.listen()
```

15 Pairing (Verbindungsaufnahme)

Bevor wir Verbindungen von Client-Geräten annehmen können, muss ein Pairing hergestellt werden. Wir gehen dies beispielhaft für ein Android-Gerät durch. Am besten zeigen wir den SDP-Eintrag des virtuellen Keyboards während der Verbindungsaufnahme an. Hierfür muss eine zweite Shell auf dem Raspberry Pi geöffnet werden, entweder in Form einer weiteren SSH-Sitzung oder – wenn Tastatur und Bildschirm angeschlossen sind – durch Drücken von Strg+Alt+F2, was Sie zur nächsten Konsole bringt (mit Strg+Alt+F1 gelangen Sie zurück zur ersten Konsole).

In der ersten Shell müssen Sie PiTooth starten, tippen Sie dazu `sudo ./PiTooth`. Es wird ausgegeben, dass der Service-Eintrag hinzugefügt wurde und dass auf die Verbindung eines Client-Geräts gewartet wird. Gehen Sie jetzt zu den Bluetooth-Einstellungen Ihres Client-Geräts und machen Sie es auffindbar. Wechseln Sie dann zur zweiten Shell und geben Sie das Kommando hcitool scan ein, um die MAC-Adresse des Clients zu erhalten. Geben Sie mit dieser Information den Befehl `bluez-simple-agent hci0 [MAC-Adresse]`. Geben Sie eine PIN ein und dann noch mal auf dem Client-Gerät. Jetzt sollte das Pairing zwischen dem Raspberry Pi und dem Client geklappt haben. Rückgängig machen können Sie es übrigens mit `bluez-test-device remove [MAC-Adresse]`.



BYTE	D7	D6	D5	D4	D3	D2	D1	D0
0	Report ID = oxo1							
1	OS rechts	Alt rechts	Shift rechts	Strg rechts	OS links	Alt links	Shift links	Strg links
2	Reserved (oxoo)							
3	Taste 1							
4	Taste 2							
5	Taste 3							
6	Taste 4							
7	Taste 5							
8	Taste 6							

16 Bisherigen Code testen

Testen wir einmal den Code, den wir bisher haben. Wechseln Sie zurück zur ersten Shell und scrollen Sie hinunter bis zum Eintrag des Raspberry Pi auf dem Client-Gerät. Wählen Sie die Verbindungsoption. Möglicherweise müssen Sie ein Reset von Bluetooth auf dem Client durchführen (einfach das Gerät aus- und wieder einschalten), falls das sofortige Fallenlassen der Verbindungen diesen durcheinanderbringt. Der Output:

```
pi@raspberrypi ~/PiTooth $ sudo ./
PiTooth.py
Service record added
Waiting for a connection
Got a connection on the control
channel from [Client-MAC]
Got a connection on the interrupt
channel from [Client-MAC]
```

17 Input-Berichte

Bevor wir die Keyboard-Klasse programmieren, sollten Sie etwas über Input-Berichte erfahren. In der Tabelle oben links auf dieser Seite sehen Sie das Schema eines Input-Berichts unter der Bluetooth-HID-Spezifikation. Ein solcher Bericht hat eine Länge von neun Bytes. Davor steht ein Byte, welches anzeigt, dass nun ein Input-Bericht folgt. Das erste Byte des Berichts mit der Report-ID ist immer 0x01, der Wert eines Keyboard-Input-Reports.

Es kann nicht schaden, auch ein bisschen über Binärdaten zu wissen. Ein Byte besteht aus acht Bits, die jeweils einen Wert von 1 oder 0 besitzen. Somit kann ein Byte einen numerischen (Dezimal-)Wert von bis zu 255 haben. Werte in der Form 0x00 sind hexadezimal. Beispielsweise steht 0x01 für 00000001 in Binärcode und repräsentiert die Dezimalzahl 1. Ein Zeichen wie etwa der Buchstabe A umfasst ebenfalls ein Byte. Wir werden dies sogleich anwenden, da wir Datentypen konvertieren müssen.

Das zweite Byte gibt an, welche Hilfstasten (Strg, Alt, Shift, OS-/Betriebssystemtaste) gedrückt werden. Das zu der jeweiligen Taste gehörige Bit hat den Wert 1, wenn sie gedrückt wird, ansonsten 0. Das dritte Byte ist reserviert, und die restlichen Bytes sind bis zu sechs Tasten-Events zugeordnet.

18 Keyboard-Klasse

Wir beginnen mit der Definition der Struktur eines Input-Berichts, wobei standardmäßig alle Tasten ausgeschaltet sind. Dann loopen wir das Ganze, bis ein Keyboard vorhanden ist.

```
class Keyboard():
    def __init__(self):
        # Struktur eines BT-
Keyboard-Input-Berichts
(Größe: 10 Byte)
        self.state = [
          0xA1, # Das ist ein
Input-Bericht
          0x01, # Verwendung
= Keyboard
          # Bit-Array für
Hilfstasten
          [0, # OS rechts
(z.B. Windows-Taste)
          0, # Alt rechts
          0, # Shift rechts
          0, # Strg rechts
          0, # OS links
(z.B. Windows-Taste)
          0, # Alt links
          0, # Shift links
          0], # Strg links
          0x00, # Vendor-
Reservierung
          0x00, # Bereich für
6 Tasten
          0x00,
          0x00,
          0x00,
          0x00,
          0x00 ]

        # Auf ein Keyboard
warten
        have_dev = False
        while have_dev == False:
          try:
#           Versuche, ein
Keyboard zu bekommen - sollte
stets ein event0 sein, da nur ein
            # Gerät angeschlossen
ist
            self.dev =
InputDevice("/dev/input/event0")
            have_dev = True
          except OSError:
            print "Keyboard
not found, waiting 3 seconds and
retrying"
            time.sleep(3)

        print "Found a keyboard"
```

19 Keyboard-Status ändern

Ehe wir weitermachen, werfen Sie einen Blick in die Datei **keymap.py**, die Sie in Schritt 06 heruntergeladen haben. Sie enthält ein Wörterbuch, das evdev-Tasten und Bluetooth-Tasten einander zuordnet, ebenso wie die evdev-Hilfstasten und das Bit, dessen Zustand im Input-Bericht festgelegt werden muss. Die Bluetooth-Werte sind dezimal statt hexadezimal, doch das macht nichts, denn Python wandelt ohnehin Hexadezimal- in Dezimalwerte um.

Diese Funktion benötigt ein evdev-Event und ändert den Status des Keyboards. Zunächst prüfen wir, ob die Taste eine Hilfstaste ist (falls nicht, wird ein negativer Wert zurückgegeben), und falls ja, wird das entsprechende Bit im Array des Input-Berichts umgestellt. Andernfalls erhalten wir den Bluetooth-Code der Taste und loopen dann durch deren Elemente. Wenn die Taste losgelassen wird (Event-Wert 0), suchen wir die Taste und setzen den Wert zurück auf 0. Ansonsten setzen wir die erste leere Stelle auf den Wert der Taste und verlassen die Schleife.

> **Es kann nicht schaden, etwas über Binärdaten zu wissen.**

```
    def change_state(self, event):
        evdev_code = ecodes.KEY[event.
code]
        modkey_element = keymap.
modkey(evdev_code)

        if modkey_element > 0:
        # Bit für Hilfstasten
wird gesetzt
        if self.state[2][modkey_
element] == 0:
        self.state[2][modkey_
element] = 1
        else:
self.state[2][modkey_
element] = 0

    else:
        # Hexadezimal-Tastencode
ermitteln
        hex_key = keymap.
convert(ecodes.KEY[event.code])

        # Durch Elemente 4 bis 9
des Input-Berichts loopen
        for i in range (4, 10):
            if self.state[i] ==
hex_key and event.value == 0:
            # Code ist 0, also
Taste loslassen
            self.state[i] = 0x00
        elif self.state[i] ==
0x00 and event.value == 1:
            # Wenn
Stelle leer ist und Taste gedrückt
wird
            self.state[i] =
hex_key
            break
```

20 Event-Schleife

Die Event-Schleife ist recht simpel. Sie benötigt eine Instanz der Bluetooth-Klasse als Parameter und liest Input des Keyboards in eine Schleife ein. Die evdev-Tasten haben drei Zustände: drücken, loslassen, halten. Da das Bluetooth-Protokoll halten nicht benötigt, lassen wir es bei den ersten beiden bewenden. Es wird vorausgesetzt, dass die Taste gehalten wird, solange ihr Wert nicht auf 0 zurückgesetzt wurde. Wir ignorieren die halten-Events mit der Bedingung „event.value < 2“. Nun können wir einfach das Event an die Funktion „change_state“ weiterleiten und dann die Funktion „send_input“ der Bluetooth-Klasse aufrufen, die wir nun erstellen:

```
def event_loop(self, bt):
    for event in self.dev.read_
loop():
            # Nur, wenn eine Taste
gedrückt wird, ausführen
            if event.type == ecodes.
EV_KEY and event.value < 2:
                self.change_
state(event)
                bt.send_
input(self.state)
```

21 Input senden

Scrollen Sie hoch zur Bluetooth-Klasse und erstellen Sie die Funktion „send_input“. Diese Funktion mag ein bisschen kompliziert aussehen, aber eigentlich tut sie nichts weiter, als den Input-Bericht von der Keyboard-Klasse in einen String umzuwandeln, der mit dem Python-Bluetooth-Modul verschickt werden kann. Wir nehmen also einen Input-Bericht entgegen und definieren einen leeren String, den wir füllen und dann abschicken werden. Daraufhin loopen wir durch alle Elemente des Input-Berichts und prüfen, ob ein Array vorliegt. Falls ja, dann handelt es sich um das Array einzelner Bits, aus denen das Byte für die Hilfstasten besteht. Dieses Array muss in einen String mit acht Zeichen konvertiert werden, indem durch alle Elemente geloopt wird und jedes dem String „bin_str“ hinzugefügt wird. Anschließend werden diese Binärdaten in eine Ganzzahl umgewandelt und diese (die ein Byte groß ist) wiederum in ein Zeichen (ebenfalls ein Byte groß), das dem String „hex_str“ hinzugefügt werden kann. Falls jedoch kein Array vorliegt, dann kann das Element des Input-Berichts direkt in ein Zeichen konvertiert und angefügt werden. Ist der String vollständig, wird er an den Interrupt-Kanal gesendet.

```
    def send_input(self, ir):
        # Array in String
konvertieren
        hex_str = ""
        for element in ir:
            if type(element) is list:
            # Bit-Array
wird in einzelnes Byte
als Zeichen
            # konvertiert
            bin_str = ""
            for bit in element:
            bin_str +=
str(bit)
            hex_str +=
chr(int(bin_str, 2))
        else:
            # Hexadezimalwert,
also direkt in Zeichen umwandeln
            hex_str +=
chr(element)
        # Input-Bericht senden
        self.cinterrupt.send(hex_str)
```

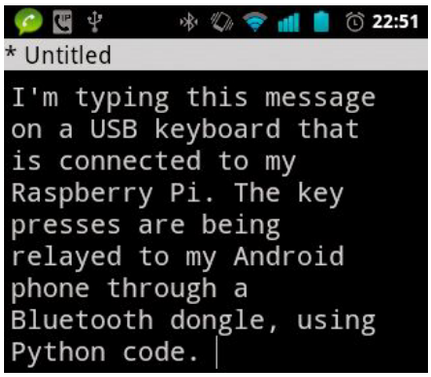
22 Abschluss des Codes

Wir schließen den Code ab, indem wir die Keyboard-Klasse initialisieren und die Event-Schleifen-Funktion aufrufen.

```
kb = Keyboard()
kb.event_loop(bt)
```

23 Fertigen Code testen

Das war's! Insgesamt ist diese Implementierung eher einfacher Natur. Die Performance könnte sicherlich verbessert werden, aber grundsätzlich funktioniert es immerhin. Wir müssen uns nicht einmal mit dem Control-Kanal auseinandersetzen, und wir ignorieren etwaige Output-Berichte auf dem Interrupt-Kanal, die vielleicht Instruktionen bezüglich LEDs auf der Tastatur enthalten könnten. Man braucht nur den Code auszuführen und das Keyboard zu bedienen, um Input an das Client-Gerät zu senden.



XLoBorg-Controller mit Kippsensor

Mithilfe des Beschleunigungsmessers im XLoBorg machen Sie einen Spielecontroller aus dem Raspberry Pi.

Was Sie brauchen:

- Python 2.7
  python.org/doc
- PiBorg XLoBorg
  piborg.org/xloborg
- Pygame
  pygame.org/docs

Wer die Entwicklungen rund um den Raspberry Pi aufmerksam verfolgt, erinnert sich vielleicht an den DoodleBorg, ein massives ferngesteuertes Gefährt von PiBorg, die auch Zusatzmodule für den Pi anbieten. Hier widmen wir uns XLoBorg, einer Platine, die unter anderem Bewegungen und deren Richtung messen kann.

Für unter 15 € ist der XLoBorg zudem ein echtes Schnäppchen, das sowohl einen Beschleunigungsmesser als auch ein Magnetometer (Digitalkompass) mitbringt, die beide auf drei Achsen funktionieren. In dieser Anleitung wird mittels Beschleunigungsmesser ein Spielecontroller mit Kippsensor aus dem Pi, der in einer einfachen Demo gleich auf künftige Pygame-Integration getestet wird.

Oben Der winzige XLoBorg ist kaum größer als die GPIO-Leiste des Raspberry Pi.

Unten Die Platine besitzt einen Beschleunigungsmesser und ein Magnetometer mit drei Achsen, kann also Neigung und Richtung messen.

01 Installation des XLoBorg

Den XLoBorg mit Ihrem Raspberry Pi zu verbinden, ist wirklich einfach: Er wird einfach direkt auf die GPIO-Pins gesteckt, wobei die Platine sich oberhalb des Pi befindet. PiBorg hat auch für eine einfache Installation der Software gesorgt und ein Skript erstellt, das automatisch die passenden I2C-Treiber und -Bibliotheken auf Ihren Pi lädt.

02 Software-Download

Erstellen Sie mit dem Befehl `mkdir ~/xloborg` einen neuen Ordner für die Projektdaten im Home-Verzeichnis, öffnen Sie diesen mit `cd ~/xloborg`, und laden Sie das Paket herunter:

```
wget www.piborg.org/downloads/xloborg/examples.zip
```

Entpacken Sie es mit `unzip examples.zip` und erlauben Sie, install.sh als Programm auszuführen (`chmod +x install.sh`). Starten Sie es mit `./install.sh` und booten Sie den Pi neu, sobald die Installation abgeschlossen ist.

03 Inspektion der Bibliothek

Bei neuer Hardware oder Software, die spezifisch für die Abstraktion eines bestimmten Vorgangs entworfen wurde, ist ein Blick auf die Code-Bibliothek immer hilfreich, um den Prozess zu verstehen. Die XLoBorg-Bibliothek heißt XLoBorg.py, sie enthält gut kommentierten Code, der die Kernfunktionen der Platine erläutert. Sie lässt sich überdies als Programm ausführen, das die Sensormessungen im Terminal ausgibt. Starten Sie es aus dem Terminal mit `python XLoBorg.py`, um seine Funktion zu prüfen. Bewegen Sie den Rechner, um zu verifizieren, dass sich die Messergebnisse ändern. Lesen Sie andernfalls auf der Hilfeseite **piborg.com/xloborg/troubleshoot** nach.

04 Skript starten

Funktioniert alles korrekt, sehen Sie jetzt bewegungsabhängige Sensormessungen von XLoBorg.py. Beginnen Sie

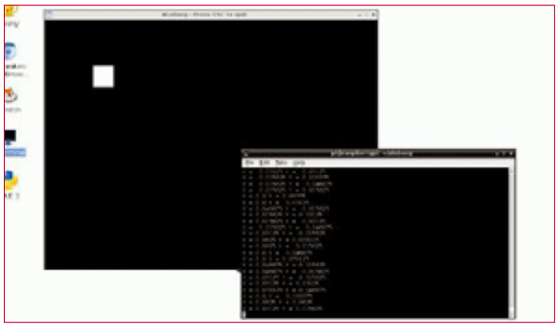
dann mit dem Skript für die Kernfunktionen des XLoBorg. Öffnen Sie eine neue Textdatei und speichern Sie sie als xloborg_app.py. Der XLoBorg muss zunächst im Skript mit XLoBorg.Init() initialisiert werden, danach sollten Sie die Datenausgabe deaktivieren, damit die Terminalausgaben Sie nicht stören:

```
XLoBorg.printFunction = XLoBorg.NoPrint
```

Jetzt benötigen Sie noch eine Variable, um die Ausgabewerte des Beschleunigungsmessers zu speichern. Die folgende Zeile erfüllt diesen Zweck:

```
tilt = XLoBorg.ReadAccelerometer()
```

Damit wird ein Tupel namens tilt erstellt, das die Kräfte x, y und z beinhaltet (in dieser Reihenfolge). Für unser Beispiel ist z nicht relevant, darum können die Messwerte für die Demo einfach in eine praktische kleine Funktion überführt werden, indem tilt[0] und tilt[1] jeweils für die Messungen von x und y aufgerufen werden.



05 Pygame-Demo

Schwer zu glauben, aber mehr XLoBorg-Code ist nicht nötig, um einen Spielecontroller mit Kippsensor aus dem Raspberry Pi zu machen. Erstellen wir nun also eine schlichte Grafik-Demo, aus der ganz leicht etwa ein Murmellabyrinthspiel entstehen könnte. Ganz oben im Skript werden Bildschirm- und Murmelgröße und die Farben festgelegt. Danach wird Pygame gestartet und Anzeige und Fenstertitel erzeugt. Der Großteil des Programms ist einfach eine Code-Klasse, Ball. Die zwei Methoden update(x, y) und collide() sorgen dafür, dass die Position des Balls aktualisiert wird, die letztere legt zudem fest, was bei Kollisionen mit dem Bildschirmrand passiert. Die Methode update() liest zwei Variablen aus – hier landen dann in der Hauptschleife des Skripts die Ausgaben des Beschleunigungsmessers.

06 Die Hauptschleife

Das Spiel läuft abhängig von der Variablen game_over in einer Endlosschleife. Solange game_over nicht zutrifft (= False), wird alles nach der Zeile „while not game_over:“ 30-mal pro Sekunde wiederholt, bis Escape oder Strg+C gedrückt wird. Bevor die Schleife startet, wird geprüft, dass game_ over nicht zutrifft, und ein Ball generiert. Innerhalb der Endlosschleife wird der Beschleunigungsmesser des XLoBorg ausgelesen, der Bildschirm schwarz gefüllt und die Methode ball.update(x, y) mit der Variablen tilt aufgerufen, um die aktuellen Messungen an den Ball zu leiten. Dann wird dessen Position bestimmt, damit er nicht den Bildschirm verlässt, und er wird auf dem Bildschirm dargestellt und mit der Pygame-Funktion flip() erneut gezeichnet.

Speichern Sie das Skript und starten Sie es mit **python xloborg_app.py**. Wenn Sie jetzt den Raspberry Pi kippen, bewegt sich der Ball auf dem Bildschirm – tut er das in die falsche Richtung, stellen Sie den betreffenden Werten einfach ein Minuszeichen voran, um sie zu spiegeln, wie im Code-Beispiel gezeigt.

Der komplette Code

```
#!/bin/bash/env python

import pygame
import XLoBorg

WIDTH = 800
HEIGHT = 600
REF_RATE = 30
BALL_SIZE = (24, 24)
BLACK = (0, 0, 0)
WHITE = (255, 255, 255)

pygame.init()
XLoBorg.Init()
XLoBorg.printprintFunction = XLoBorg.NoPrint

screen = pygame.display.set_mode([WIDTH, HEIGHT])
clock = pygame.time.Clock()
pygame.display.set_caption('XLoBorg test - Press ESC to quit')

class Ball(pygame.sprite.Sprite):
    def __init__(self, width, height):
        pygame.sprite.Sprite.__init__(self)
        self.size = (width, height)
        self.image = pygame.Surface([width, height])
        self.image.fill(WHITE)
        self.rect = self.image.get_rect()
        self.rect.x, self.rect.y = WIDTH / 2, HEIGHT / 2
        self.speed = (50, 50)
        self.tilt = [0, 0]

    def update(self, x, y):
        self.rect.x += x * self.speed[0]
        self.rect.y += y * self.speed[1]
        print 'X =', x, 'Y =', y

    def collide(self):
        if self.rect.x > WIDTH - self.size[0]:
            self.rect.x = WIDTH - self.size[0]
        elif self.rect.x < 0:
            self.rect.x = 0
        if self.rect.y > HEIGHT - self.size[1]:
            self.rect.y = HEIGHT - self.size[1]
        elif self.rect.y < 0:
            self.rect.y = 0

def main_loop():
    game_over = False
    ball = Ball(50, 50)

    while not game_over:
        for event in pygame.event.get():
            if event.type == pygame.QUIT:
                game_over = True
            if event.type == pygame.KEYDOWN:
                if event.key == pygame.K_ESCAPE:
                    game_over = True

        tilt = XLoBorg.ReadAccelerometer()
        screen.fill(BLACK)
        ball.update(-tilt[0], tilt[1])
        ball.collide()
        screen.blit(ball.image, ball.rect)
        pygame.display.flip()
        clock.tick(REF_RATE)
    pygame.quit()

if __name__ == '__main__':
    try:
        main_loop()

    except KeyboardInterrupt:
        pygame.quit()
```

Import und Initialisierung Importieren Sie die Bibliothek und rufen Sie init() auf, um den XLoBorg in Ihren Python-Skripten zu starten. Die Textausgabe ist hier deaktiviert.

Update Die Update-Methode in der Ball-Klasse bewegt den Ball. Sie fragt x- und y-Werte ab, die später in der Hauptschleife aus der Ausgabe des XLoBorg gespeist werden.

Feedback-Schleife In der Funktion main_ loop verrichtet Pygame die meiste Arbeit. Hier werden auch die Sensormessungen des XLoBorg an die Update-Methode geleitet.

Bigtrak-Steuerung à la Raspberry Pi

Man nehme ein Spielzeug, einen Raspberry Pi, einen PS3-Controller, eine Prise Python und etwas Lötzinn… für das perfekte ferngesteuerte Gefährt.

Was Sie brauchen:

- Bigtrak
  bigtrakxtr.co.uk/shop/bigtrak
- Steckplatine und Kabel
- Motortreiber
  bit.ly/1iOnFug
- USB-Akku-Pack
  amzn.to/1h2PBiI
- DualShock 3 (PS3-Controller)

Bigtrak-Steuerung à la Raspberry Pi

Der Raspberry Pi ist ein kleiner und günstiger Computer, der die Leute für die IT und das Programmieren begeistern will – aber deshalb muss es ja nicht bierernst zugehen. In diesem Artikel zeigen wir Ihnen daher, wie man mit dem Raspberry Pi einen Bigtrak in einen Roboter verwandelt. Pädagogisch wertvoll, nicht?

Der Bigtrak ist ein programmierbares Spielzeugauto, das eine Reihe einfacher Kommandos (vorwärts, links, rechts) erhält und diese abarbeitet. Damit die Sache spannend wird, werden wir die bestehenden Schaltkreise durch einen Raspberry Pi ersetzen und mithilfe eines kleinen Motortreibers die Motoren des Bigtrak ansteuern, die wir dann mittels eines DualShock-Controllers dirigieren.

Softwareseitig bringt das aktuelle Raspbian-Image alles nötige mit. Um die Befehle vom Controller zu den Motoren durchzugeben, steuern wir nur ein kleines Python-Skript bei, das die Pygame- und RPI.GPIO-Module verwendet.

01 Bigtrak öffnen – der einfache Teil

Bevor wir den Bigtrak umbauen können, müssen wir ihn öffnen. Drehen Sie ihn dazu um und entfernen Sie die neun Schrauben. An die meisten davon kommt man gut heran, lediglich die Schrauben an der Vorderseite erfordern einen etwas schlankeren Schraubenzieher.

02 Bigtrak öffnen – der fummelige Teil

Die letzten beiden Schrauben befinden sich hinter dem grauen Kühlergrill auf der Rückseite. Der Grill wird von vier Plastikzapfen gehalten, die man drücken muss, während man den Grill abzieht. Das ist nicht ganz einfach, selbst wenn man zusätzliche Hände zum Helfen rekrutiert. Am besten funktioniert es, ein dünnes Stück Plastik (zum Beispiel ein Gitarren-Plektron) zwischenzuschieben, um die oberen Zapfen gedrückt zu halten, während man die unteren mit den Fingern drückt und den Grill nach oben schiebt. Jetzt die restlichen Schrauben lösen.

03 Oberseite entfernen

Drehen Sie den Bigtrak wieder um und entfernen Sie vorsichtig die obere Abdeckung. Diese wird durch ein Flachbandkabel und einen Schalter mit der Unterseite verbunden, also heben Sie die Abdeckung nur etwas an und klappen Sie sie zur Seite, um an das Innenleben zu kommen.

Wenn die Abdeckung offen auf der Seite liegt, lösen Sie die Schraube, die den Schalter fixiert, und entfernen Sie diesen von der Abdeckung. Als Nächstes müssen Sie die beiden Schrauben auf der Leiterplatte rausnehmen, die das Flachbandkabel an Ort und Stelle halten, und dieses ebenfalls lösen.

Sobald Schalter und Kabel getrennt sind, können Sie die Oberseite vollends vom Bigtrak entfernen.

04 Kabel durchtrennen

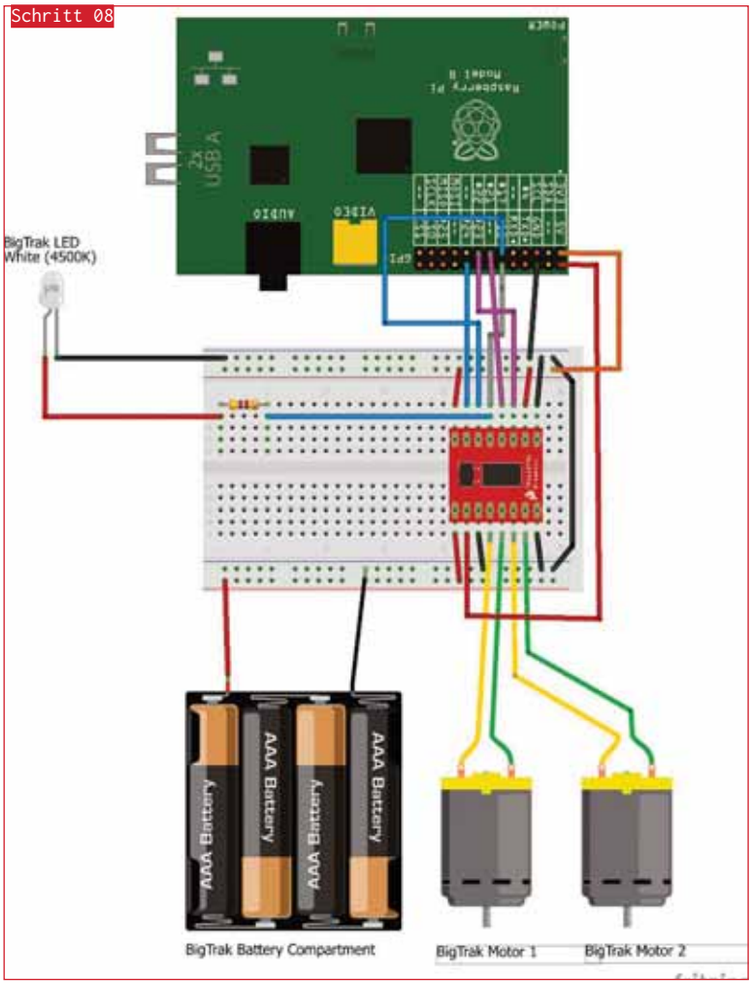
Durchtrennen Sie die Kabel zur Haupt-Leiterplatte. Die für den Schalter und den Strom sollten Sie dicht über der Leiterplatte schneiden (damit man sie später noch verwenden kann), während die zur LED und den Lautsprechern an einer beliebigen Stelle durchtrennt werden können.

05 Motoren entfernen

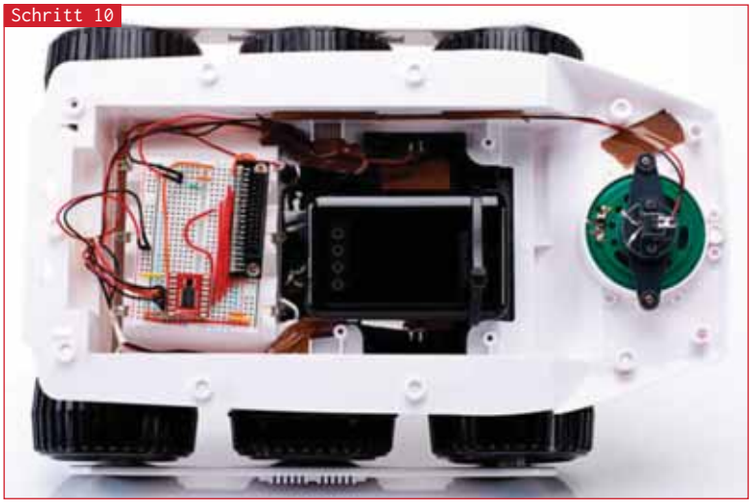
Drehen Sie den Bigtrak wieder herum und entfernen Sie die vier Schrauben, die die Motoren an ihrem Platz halten (damit beim Löten nichts kaputtgeht). Wenn Sie jetzt den Bigtrak vorsichtig drehen, sollten sich die Motoren einfach lösen lassen.

Die Drähte müssen lang genug sein, um bis in den letzten Winkel des Bigtrak zu reichen, also seien Sie großzügig!

Schritt 08



Schritt 10



06 Motoren neu verkabeln

Entfernen Sie das Lötzinn, das die Hauptplatine mit den Motoren verbindet (Entlötlitze ist dabei sehr hilfreich), und sodann die Platine selbst. Jetzt können wir Drähte an den Motoren befestigen, die später die Kommandos des Raspberry Pi weiterleiten. Die Drähte müssen lang genug sein, um bis in den letzten Winkel des Bigtrak zu reichen, also seien Sie ruhig großzügig. Zu lange Drähte lassen sich später immer noch kürzen. Nun können Sie die Motoren wieder in den Bigtrak einbauen.

07 Motortreiber verbinden

Sobald die Motoren wieder an ihrem Platz sind, müssen wir einen Schaltkreis bauen, der an den Raspberry Pi angeschlossen wird. Wir haben ein Flachbandkabel verwendet, um die GPIO-Pins des Raspberry Pi mit einer Steckplatine zu verbinden, auf der wir einen Dual-Motortreiber platzieren (**proto-pic.co.uk/motor-driver-1a-dual-tb6612fng**), welcher dann die eigentliche Ansteuerung der Motoren übernimmt. Auf diese Weise schützen wir die empfindlichen GPIO-Pins vor der höheren Spannung, die zum Betrieb der Motoren nötig ist. Die Verbindungen auf der Steckplatine haben wir in der folgenden Tabelle aufgelistet. Diese Werte sind notwendig, wenn Sie die Software schreiben, und können sich unterscheiden, je nachdem, welches Breakout-Board Sie verwenden und welche Revision des Raspberry Pi zum Einsatz kommt.

Pi-GPIO-Pin	Motortreiber
24	AIN2
17	AIN1
18	STBY
21	BIN1
22	BIN2

Durch die direkte Verbindung der PWMA- und PWMB-Pins mit der 3,3-Volt-Stromschiene werden die Motoren von jetzt an immer mit voller Leistung laufen, solange sie aktiv sind.

08 Steckplatine installieren

Die Steckplatine wird auf dem Batteriefach innerhalb des Bigtrak installiert, weshalb die Verkabelung der Motoren über die Rückseite geführt und mit Isolierklebeband fixiert werden sollte. Auf dieselbe Weise können Sie die Verkabelung der Batterien einsetzen, um die Angelegenheit möglichst kompakt zu halten.

09 Alles verkabeln

Um Motoren und Batterien einfach mit der Steckplatine zu verbinden, haben wir ein paar modulare Verbindungsstecker an die Kabelenden gelötet, die man einfach hineinschieben kann.

Mit der Steckplatine fest an ihrem Platz, können wir nun (nachdem wir alle Verbindungen doppelt überprüft haben) die Motoren und die Energieversorgung anschließen. Damit Sie wissen, wenn die Motoren aktiv sind (und um die Chance ungewollter Bewegungen zu verringern), können Sie die LED mit der Steckplatine verbinden, sodass sie immer aufleuchtet, wenn der Standby-Pin aktiv ist. Schalten Sie einen Widerstand dazwischen, damit die LED nicht zu viel Spannung zieht und durchbrennt.

10 Stromversorgung

Die Stromversorgung des Raspberry Pi erfolgt über ein USB-Akku-Pack, das über dem Motor platziert und mit Kabelbindern oder Klettband befestigt wird. Diese externen Akku-

Packs werden oft für Smartphones und Tablets verwendet – die Batterieleistung in unserem Beispiel entspricht 8.000 mAh und sollte den Raspberry Pi acht Stunden lang mit Energie versorgen.

11 Raspberry Pi anschließen – zusätzliche Kabel

Da der Raspberry Pi auf dem Bigtrak montiert werden soll, müssen wir das Flachbandkabel und das Stromkabel durch die obere Abdeckung hindurchführen. Hierzu drehen wir die Abdeckung herum und lösen die beiden Klammern, die das dunkelgraue Plastik fixieren – die dadurch entstehende Öffnung ist groß genug für das Flachbandkabel und das USB-Stromkabel. Stellen Sie sicher, dass die rote Seite des Flachbandkabels mit dem Verbindungsstück auf der Steckplatine übereinstimmt, sodass Sie das Kabel nicht nachträglich innerhalb des Gehäuses drehen müssen.

12 Raspberry Pi anschließen – letzte Schritte

Sobald die Oberseite des Bigtrak wieder befestigt wurde, kann der Raspberry Pi an seinen Platz, mit den GPIO-Pins nach vorn ausgerichtet, damit Sie das Flachbandkabel einfach verbinden können. Das Akku-Pack wird mit Kabelbindern und Klebestreifen an seinem Platz gehalten. Theoretisch kann man den nackten Raspberry Pi auf dem Bigtrak befestigen, allerdings besteht dabei die Gefahr, dass die SD-Karte verbogen wird, weshalb Sie ein Gehäuse für den Raspberry Pi verwenden sollten. Verbinden Sie das Flachbandkabel und das Stromkabel mit dem Pi und schalten Sie ihn ein. Das Gerät wartet jetzt auf Ihre Befehle. Für die nächsten Schritte ist es sinnvoll, eine Tastatur und einen Bildschirm anzuschließen.

13 Controller anschließen

Hier müssen Sie nur den DualShock 3 an den USB-Port anschließen, denn die benötigte Software sollte bereits Bestandteil des aktuellen Raspbian-Images sein und den Controller automatisch erkennen. Um zu bestätigen, dass der DualShock 3 erkannt wurde, starten Sie lsusb und prüfen Sie, ob er in der Ergebnisliste auftaucht.

14 Software starten

Nun da das System eingerichtet ist, müssen Sie noch die Datei bigtrak.py herunterladen, auf den Raspberry Pi kopieren und ausführen. Da das Skript Zugriff auf die GPIO-Pins benötigt, muss es als Superuser laufen, also starten Sie es mit dem Befehl:

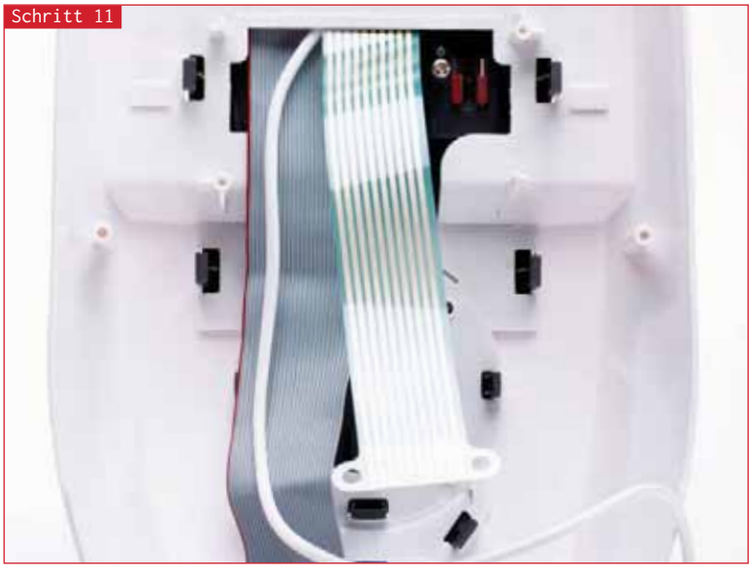
```
sudo python bigtrak.py
```

Jetzt können wir den Bigtrak mit den Analogsticks steuern. Der linke Stick steuert den linken Motor und der rechte Stick den rechten. Um vorwärts zu fahren, drücken Sie beide Sticks nach oben. Rückwärts fahren Sie, indem Sie beide Sticks nach unten ziehen. Drücken Sie einen Stick nach oben und einen nach unten, rotiert das Fahrzeug. Wenn die Analogsticks den Bigtrak nicht wie erwartet steuern, überprüfen Sie die GPIO-Verbindungen.

15 Ausblick

Nachdem Sie jetzt eine solide Basis für Ihren Raspberry-Pi-Roboter haben, könnten Sie weitere Anpassungen vornehmen, etwa einen USB-Bluetooth-Adapter für eine Drahtlosverbindung mit dem Controller einrichten. Oder Sie ersetzen die Steckplatine durch ein PiPlate oder ein „Slice of Pi"-Board, sodass Sie den Pi in das Bigtrak-Gehäuse einbauen können. Man könnte auch die Raspberry-Pi-Kamera und einen USB-WLAN-Adapter hinzufügen, um ein Live-Video zu streamen, während Sie herumfahren. Oder wie wäre es mit einem Roboter-Arm?

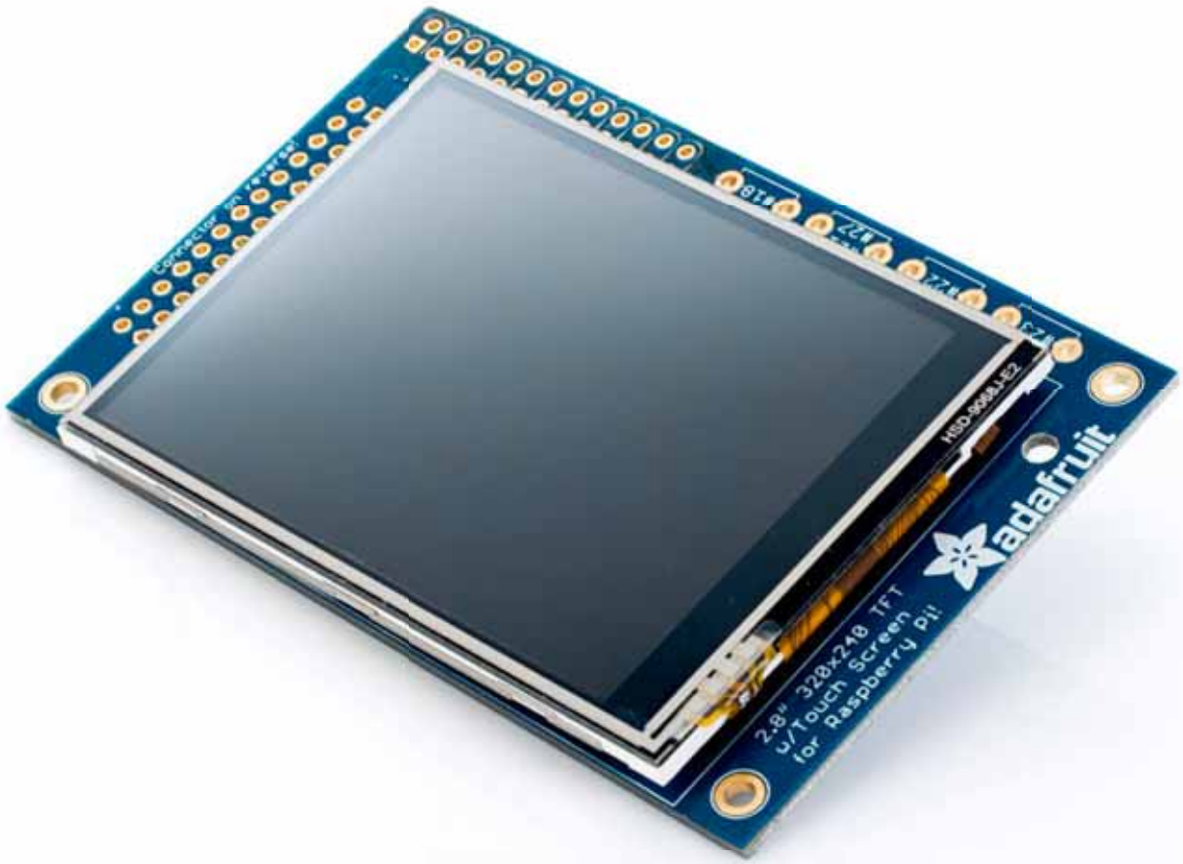
Schritt 11



Schritt 12

PiTFT-Touchscreen: Software installieren

Der kleine Touchscreen von Adafruit ist zwar kein Plug-&-Play-Bauteil, aber hier zeigen wir Ihnen, wie Sie die zugehörige Software einrichten.



Was Sie brauchen:

- SD-Karte mit aktuellem Raspbian-Image
- PiTFT-Touchscreen
  bit.ly/1jHEJT4

Zu den entscheidenden Vorzügen des Raspberry Pi gehören seine geringen Abmessungen und hohe Portabilität. Allerdings bedingt gerade das auch einen der größten Nachteile des Mini-Rechners: das Fehlen eines Displays. Der Anschluss des Pi an ein Mobilgerät per SSH ist umständlich.

Der PiTFT von der New Yorker Hardwareschmiede Adafruit löst dieses Dilemma, denn dieser Touchscreen wurde von der Größe her passend für den Raspberry Pi entwickelt und benötigt überdies keine eigene Stromversorgung. Wir zeigen Ihnen in diesem Tutorial, wie Sie das Display einbauen und aktivieren.

01 Kernel-Dateien von Adafruit

Als Erstes müssen wir die benötigten Kernel-Dateien herunterladen und installieren. Öffnen Sie ein Terminal oder eine SSH-Verbindung und geben Sie ein:

```
$ wget http://adafruit-download.s3.amazonaws.com/libraspberrypi-bin-adafruit.deb
$ wget http://adafruit-download.s3.amazonaws.com/libraspberrypi-dev-adafruit.deb
$ wget http://adafruit-download.s3.amazonaws.com/libraspberrypi-doc-adafruit.deb
$ wget http://adafruit-download.s3.amazonaws.com/libraspberrypi0-adafruit.deb
$ wget http://adafruit-download.s3.amazonaws.com/raspberrypi-bootloader-adafruit-112613.deb

$ sudo dpkg -i -B *.deb
```

Links Holen Sie sich die benötigten Dateien von der Adafruit-Website und übertragen Sie sie auf den Raspberry Pi.

02 Display montieren

Falls Sie ein Raspbian-Image verwendet haben, das nach September 2013 erschienen ist, schalten Sie die Beschleunigung des Framebuffers aus:

```
$ sudo mv /usr/share/X11/xorg.conf.d/99-fbturbo.conf ~
```

Schalten Sie danach den Raspberry Pi komplett aus und stecken Sie das Display auf die GPIO-Stifte der Rechnerplatine.

03 Display-Treiber installieren

Wir können einen kleinen Test durchführen, ob das Display schon funktioniert. Mit folgenden Kommandos schalten Sie es ein (vorher müssen Sie aber neu booten):

```
$sudo modprobe spi-bcm2708
$sudo modprobe fbtft_device name=adafruitts rotate=90
$export FRAMEBUFFER=/dev/fb1
$startx
```

04 Module hinzufügen

Es sind zwei Module vorhanden, die wir so einstellen müssen, dass Sie beim Booten automatisch geladen werden, darunter der eigentliche PiTFT-Treiber. Öffnen Sie die Moduldatei mit folgendem Befehl und fügen Sie dann die beiden unteren Zeilen der Liste hinzu.

```
$ sudo nano /etc/modules
```

```
spi-bcm2708
fbtft_device
```

05 Konfigurationsdateien bearbeiten

Damit der Touchscreen richtig funktioniert, müssen wir noch eine Konfigurationsdatei erstellen bzw. modifizieren. Mit dem folgenden Kommando öffnen Sie das File, und der Text darunter muss dann dort eingetragen werden:

```
$ sudo nano /etc/modprobe.d/adafruit.conf
```

```
options fbtft_device name=adafruitts rotate=90
frequency=32000000
```

06 Optionen „rotate“ und „frequency“

Mit den soeben eingetragenen Optionen hat es folgende Bewandtnis: Mit „rotate“ können Sie die Ausrichtung der Anzeige festlegen (zur Auswahl stehen 0, 90, 180 oder 270 Grad), und „frequency“ gibt die Anzahl der Bilder pro Sekunde (fps) an. Eine Frequenz von 32 MHz entspricht 20 fps. Sollte das Display bei dieser Einstellung herumzicken, wählen Sie stattdessen 16 MHz. Starten Sie den Raspberry Pi neu.

07 Touchscreen einstellen

Wenn Sie wieder im Terminal sind, erstellen Sie noch eine weitere Konfigurationsdatei mit folgendem Befehl und nehmen Sie die darin die darunterstehenden Einträge vor.

```
$ sudo mkdir /etc/X11/xorg.conf.d
$ sudo nano /etc/X11/xorg.conf.d/99-calibration.conf
```

```
Section “InputClass”
        Identifier      “calibration”
        MatchProduct    “stmpe-ts”
        Option  “Calibration”   “3800 200 200 3800”
        Option  “SwapAxes”      “1”
EndSection
```

08 Touchscreen testen

Zuletzt können Sie X neu starten:

```
$ FRAMEBUFFER=/dev/fb1 startx
```

Damit der Start nur mit dem Befehl `startx` funktioniert, fügen Sie **~/.profile** folgende Zeile hinzu:

```
export FRAMEBUFFER=/dev/fb1
```

Sie können in raspi-config einstellen, dass in den Desktop gebootet werden soll. Auf den nächsten Seiten wird beschrieben, wie Sie das Display kalibrieren können und so einen portablen Videoplayer aus Ihrem Raspberry Pi machen.

PiTFT-Touchscreen: Display kalibrieren

Kalibrieren Sie Ihren Touchscreen und machen Sie ihn fit für Eingaben per Finger oder Stift.

Was Sie brauchen:

- SD-Karte mit aktuellem Raspbian-Image
- PiTFT-Touchscreen
  bit.ly/1jHEJT4

Im vorherigen Artikel ging es um die Montage und Aktivierung eines Touchscreens für Ihren Raspberry Pi. Nun wollen wir den Bildschirm aber auch benutzen! Um Eingaben per Fingerdruck oder mit dem Bedienstift möglich zu machen, müssen wir das resistive Display kalibrieren. Doch keine Sorge, die Kalibrierung ist reversibel, das heißt wenn Sie kein optimales Ergebnis erzielt haben, können Sie den Vorgang problemlos wiederholen und dabei gegebenenfalls auch die Methode wechseln.

Der PiTFT-Touchscreen von Adafruit macht Ihren Raspberry Pi unabhängig von einem Monitor und – mit noch einer USB-Powerbank dazu – zu einem echten Mobilgerät.

01 Neue Regel festlegen

Um nicht den Überblick zu verlieren, wie der Touchscreen in der Kommandozeile angesprochen wird, erstellen wir zunächst eine kleine Regel. Verbinden Sie sich per SSH mit dem Pi, tippen Sie zuerst den Befehl, und fügen Sie dann den darunterstehenden Text in die geöffnete Datei ein:

```
$ sudo nano /etc/udev/rules.d/95-stmpe.rules
SUBSYSTEM=="input", ATTRS{name}=="stmpe-ts", ENV{DEVNAME}=="*event*", SYMLINK+="input/touchscreen"
```

02 Touchscreen erneut installieren

Damit die Berührungssteuerung fehlerfrei funktioniert, müssen wir einen Schritt vom letzten Tutorial rückgängig machen, nämlich die Installation der Touchscreen-Unterstützung. Wir stellen die Installation allerdings sogleich wieder her:

```
sudo rmmod stmpe_ts; sudo modprobe stmpe_ts
```

03 Kalibrierungs-Tools

Finden Sie heraus, welches Ereignis dem Touchscreen zugeordnet ist, indem Sie eingeben:

```
$ ls -l /dev/input/touchscreen
```

Nun können wir Tools für die Kalibrierung und die Fehlerbeseitigung installieren und sie auch gleich testen:

```
$ sudo apt-get install evtest tslib libts-bin
$ sudo evtest /dev/input/touchscreen
```

04 Touchscreen kalibrieren

Es folgt die eigentliche Kalibrierung. Geben Sie Folgendes ein, damit der Raspberry Pi eine ungefähre Vorstellung von den Positionen auf dem Bildschirm bekommt:

```
$ sudo TSLIB_FBDEVICE=/dev/fb1 TSLIB_TSDEVICE=/dev/
input/touchscreen ts_calibrate
```

05 Kalibrierung testen

Danach können Sie einen Zeichentest durchführen, um das Verhalten des Displays zu prüfen. Geben Sie hierzu Folgendes ein:

```
$ sudo TSLIB_FBDEVICE=/dev/fb1 TSLIB_TSDEVICE=/
dev/input/touchscreen ts_test
```

Wiederholen Sie Schritt 04 und 05, bis Sie mit dem Ergebnis zufrieden sind.

Der PiTFT-Touchscreen macht Ihren Raspberry Pi unabhängig von einem Monitor.

06 Download von xinput-calibrator

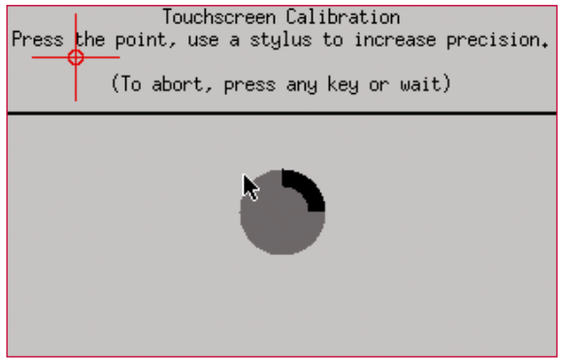
Jetzt muss der Touchscreen noch für das Fenstersystem X kalibriert werden. Laden Sie das dazu benötigte Tool herunter und installieren Sie es:

```
$ wget http://adafruit-download.s3.amazonaws.com/
xinput-calibrator_0.7.5-1_armhf.deb
```

```
$ sudo dpkg -i -B xinput-calibrator_0.7.5-1_armhf.deb
```

Löschen Sie anschließend die nicht mehr benötigten Kalibrierungsdaten:

```
$ sudo rm /etc/X11/xorg.conf.d/99-calibration.conf
```



07 Für X kalibrieren

Tippen Sie startx in der SSH-Shell, um den Bildschirm einzuschalten. Nun können Sie entweder direkt über den Touchscreen xinput_calibrator in die Kommandozeile eingeben oder – falls das nicht gut klappt – das Folgende in die SSH-Shell tippen:

```
FRAMEBUFFER=/dev/fb1 startx & DISPLAY=:0.0 xinput_
calibrator
```

08 Kalibrierung speichern

Nach der Kalibrierung bekommen Sie einen Output, der mit Section „InputClass“ beginnt. Öffnen Sie mit dem folgenden Befehl eine Datei und kopieren Sie den Output zwischen Section und EndSection dort hinein:

```
$ sudo nano /etc/X11/xorg.conf.d/99-calibration.conf
```

09 Ausprobieren

Nun ist Ihr neuer Touchscreen ordnungsgemäß eingerichtet und Sie haben einen schon ziemlich portablen Raspberry Pi zur Verfügung. Im nachfolgenden Tutorial zeigen wir Ihnen, wie Sie einen Videoplayer aus dem Rechner machen.

PiTFT-Touchscreen: Portabler Videoplayer

Unterwegs komfortabel Videos anschauen? Kein Problem mit Ihrem neuen Touchscreen-Raspberry-Pi!

Was Sie brauchen:

- SD-Karte mit aktuellem Raspbian-Image
- PiTFT-Touchscreen
  bit.ly/1jHEJT4

Auf den vorherigen Seiten haben wir gezeigt, wie Sie Ihrem Raspberry Pi ein fest eingebautes Display verpassen können – sogar mit Touchscreen-Funktionalität! Was machen wir jetzt Schönes damit? Wie wäre es mit einem tragbaren Videoplayer? So ein Gerät hätte hinsichtlich seiner Stromversorgung einen wesentlichen Vorteil gegenüber Smartphones und Tablets, die bei dieser Nutzungsart nicht besonders lange durchhalten. Außerdem könnte man es ohne Probleme bei irgendjemandem an einen Fernseher anschließen.

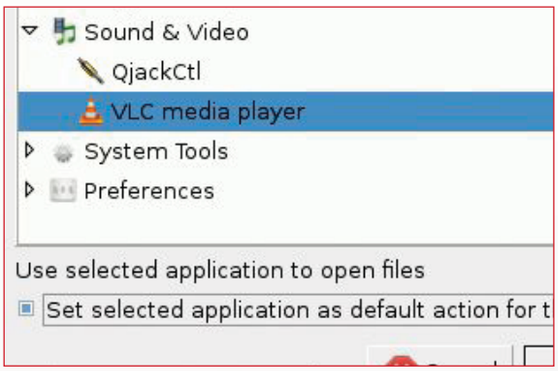
01 VLC installieren

Um auf dem Touchscreen-Pi Videos betrachten zu können, benötigen wir eine Software dafür. Der VLC media player ist eine schlichte, benutzerfreundliche, extrem vielseitige Abspielsoftware. Öffnen Sie ein SSH-Terminal und holen Sie sich den Player:

```
$ sudo apt-get install vlc
```

02 Standardeinstellungen

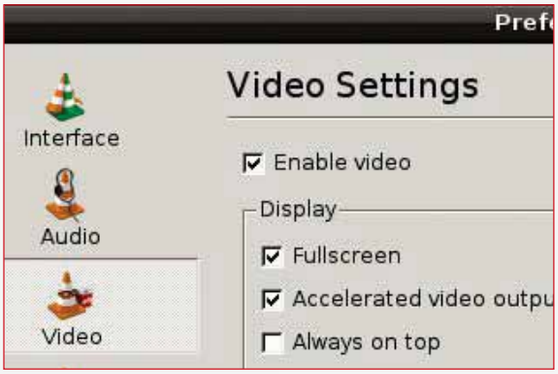
Den folgenden Schritt müssen Sie über den Touchscreen oder an einem zusätzlich an den Raspberry Pi angeschlossenen Monitor ausführen. Außerdem brauchen Sie Tastatur und Maus dazu. Navigieren Sie zu einer AVI- oder MP4-Datei, rechtsklicken Sie auf den Dateinamen und im sich



daraufhin öffnenden Dialogfenster auf „Eigenschaften“ und dann „Öffnen mit...“

03 Immer mit VLC öffnen

Wählen Sie sodann die benutzerdefinierte Einstellung, gehen Sie zu „Sound & Video“, wählen Sie VLC media player aus dem Menü aus, und haken Sie an, dass es als Standardprogramm für diese Medienarten verwendet werden soll. Wenn Sie zukünftig auf eine Audio- oder Videodatei klicken, wird diese automatisch mit dem VLC media player abgespielt.



04 VLC-Optionen

Öffnen Sie nun den VLC media player aus dem Menü heraus und rechtsklicken Sie irgendwo auf das Programmfenster. Gehen Sie im erscheinenden Kontextmenü über Extras > Einstellungen zu der Option „Nur eine Instanz erlauben“ und haken Sie diese an. Legen Sie außerdem im Tab „Videos“ fest, dass die Anzeige im Vollbildmodus erfolgen soll. Speichern Sie alle Änderungen.

05 Videos abspielen

Ab sofort können Sie sich auf dem PiTFT-Touchscreen Ihres Raspberry Pi Videos ansehen, und es wird dabei immer nur eine Player-Instanz laufen. Bei Problemen mit großformatigen Videos können Sie versuchen, diese in 320 x 240 zu konvertieren.

06 Externer Speicher

Was jetzt noch zu tun bleibt, ist, den Raspberry Pi etwas transportfreundlicher zu machen. Als Erstes ist zu bedenken, dass auf der SD-Karte mit dem Betriebssystem nicht allzu viel Speicherplatz verbleibt. Um unterwegs genügend Videomaterial dabeizuhaben, ist es daher sinnvoll, die Mediendateien separat aufzubewahren. Damit dabei nicht die Leistung des Raspberry Pi beeinträchtigt wird, sollten Sie Speichergeräte einsetzen, die keinen zusätzlichen Strom ziehen, etwa eine externe USB-Festplatte oder einen USB-Stick. Sie können auf die darin gelagerten Medien ganz einfach über den Dateimanager zugreifen.

07 Passendes Gehäuse

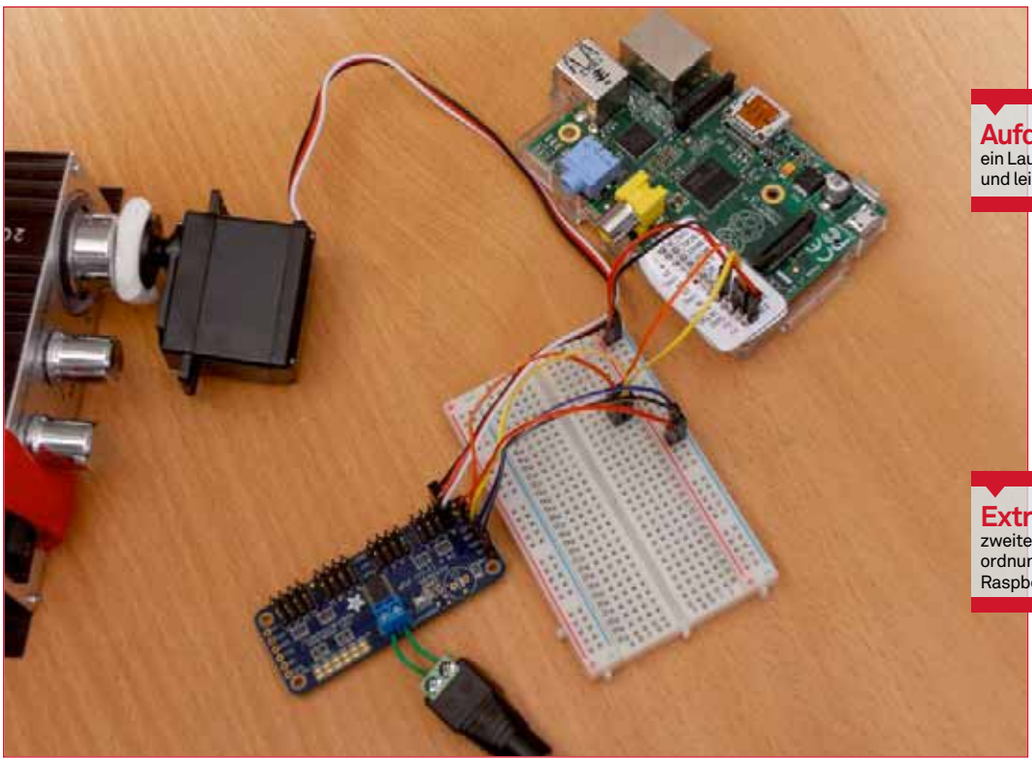
Als Nächstes wäre es eine gute Idee, den Touchscreen-Pi in ein Case zu packen, um ihn zu schützen. Das „PiTFT Pi-Bow“ vom Zubehörhersteller Pimoroni ist eins der wenigen – womöglich gar das einzige – Gehäuse für den Raspberry Pi, worin auch der anmontierte Touchscreen Platz findet (siehe Abbildung oben rechts). Allerdings ist das Case nur für das Modell B Revision 2 des Pi geeignet.

08 Separate Stromversorgung

Und schließlich sollte der Raspberry Pi to go natürlich ein Selbstversorger in Sachen Stromversorgung sein. Eine USB-Powerbank empfiehlt sich hier, wobei Sie allerdings auf hohe Werte bei Ladestrom und Nennleistung achten sollten. Jetzt kann kaum noch was schiefgehen. Viel Spaß mit Ihrem portablen Videoplayer à la Raspberry Pi!

Servos steuern mit dem Raspberry Pi

Nutzen Sie Ihren Pi, um Roboter oder Rotationsgeräte in Betrieb zu nehmen.



Aufdrehen Mit dem Servo wird hier ein Lautstärkeregler bedient, um lauter und leiser zu drehen.

Extra-Power Verwenden Sie ein zweites Netzteil für den Servo, um die ordnungsgemäße Stromversorgung des Raspberry Pi zu gewährleisten.

Was Sie brauchen:

- aktuelles Raspbian-Image
  raspberrypi.org/downloads
- Internetzugang
- Servoregler Adafruit PCA9685
  adafruit.com/product/815
- Servomotoren, je nach Projekt
- Stromversorgung 5 Volt
- Jumperkabel weiblich/weiblich

Servomotoren führen in der Regel keine kontinuierlichen, vollen Drehbewegungen aus, sondern haben einen eingeschränkten Drehwinkel. Anders als sonstige Motoren reagieren sie auf Signale, die sie innerhalb ihres Gesamtwinkels auf bestimmte Positionen (beispielsweise 30 Grad) schwenken lassen. Servos werden verwendet, um Roboter in Bewegung zu setzen, die Landeklappen eines Flugzeugs zu verstellen und Ähnliches. In diesem Tutorial befassen wir uns mit der Inbetriebnahme eines Servos für die Drehung eines Potentiometers. Unser Ziel ist es, den Raspberry Pi als Fernbedienung für den Drehknopf eines einfachen Audio-Verstärkers verwenden zu können. Wir setzen voraus, dass Sie die Stiftleisten bereits an den Adafruit-Servoregler gelötet haben. Ist das der Fall, kann es logehen.

01 Software installieren und I2C konfigurieren

Führen Sie zunächst die folgenden Befehle der Reihe nach aus, um die erforderliche Software herunterzuladen und zu installieren:

```
sudo apt-get update
sudo apt-get install python-smbus i2c-tools git
git clone https://github.com/adafruit/Adafruit-Raspberry-Pi-Python-Code.git
```

Die I2C-Module sind standardmäßig gesperrt. Um dies zu ändern, öffnen Sie die Blacklist-Datei:

```
sudo nano /etc/modprobe.d/raspi-blacklist.conf
```

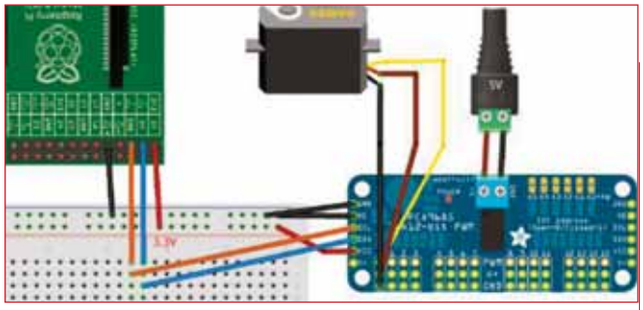
Setzen Sie darin jeweils ein # vor folgende Zeilen:

```
blacklist spi-bcm2708
blacklist i2c-bcm2708
```

Die Änderungen speichern mit Strg+O, dann Enter. Schließen Sie mit Strg+X die Datei. Danach müssen noch zwei Zeilen in die Datei /etc/modules eingefügt werden:

```
i2c-dev
i2c-bcm2708
```

Zuletzt fahren Sie den Pi mit „sudo poweroff“ herunter.



02 Servoregler verdrahten

Orientieren Sie sich am obigen Schaltbild, wenn Sie das Adafruit-Board verdrahten. Kappen Sie zuvor unbedingt die Stromzufuhr und gehen Sie sicher, dass Ihr Pi richtig ausgerichtet ist. Der OE-Pin (Output Enable) des Servoreglers ist Low-aktiv, das heißt der Ausgang ist bei einem Signal von 0 Volt freigegeben. Aufgrund dessen verbinden wir ihn mit der Masse (GND). Der VCC-Pin am Adafruit-Board muss mit dem 3V3-GPIO-Pin des Raspberry Pi verkabelt sein und die Servo-Power in der Mitte des Boards mit dem 5V-Pin. Hier benötigen wir ein zweites Netzteil, da Servos häufig Spannungsabfälle verursachen, der Pi selbst aber eine stabile Stromversorgung braucht.

```
pi@raspbian ~ $ sudo i2cdetect -y 1
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:          -- -- -- -- -- -- -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- UU -- -- -- --
40: 40 -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
70: 70 -- -- -- -- -- -- --
```

03 Verbindung zum Servoregler prüfen

Wir verwenden ein Programm namens i2cdetect, um den Servoregler aufzuspüren. Der I2C-Bus auf älteren Raspberry Pis ist 0, der auf neueren 1. Haben Sie ein neueres Gerät, so geben Sie „sudo i2cdetect -y 1“ ein. Nun sollten Sie eine Ausgabe ungefähr wie auf dem Bild dargestellt erhalten. Ist dies nicht der Fall, dann ist vermutlich irgendetwas falsch verkabelt.

04 Sag „Hallo Welt!“, Servo!

Wechseln Sie in das Verzeichnis, in dem die git-Kopie des Adafruit-Raspberry-Pi-Codes liegt (vermutlich /home/pi), und führen Sie folgende Befehle aus:

```
cd Adafruit-Raspberry-Pi-Python-Code
cd Adafruit_PWM_Servo_Driver
sudo python2 Servo_Example.py
```

Wenn Ihr Servo mit dem Kanal 0 am Adafruit-Board verbunden ist, beginnt er sich zwischen seinen Max- und Min-Positionen zu bewegen.

05 So funktioniert ein Servomotor

Der Servo_Example.py-Code im Nano-Editor fällt recht spärlich aus. Das rührt daher, dass das Adafruit-Board den Großteil der Arbeit erledigt, während der Pi selbst lediglich Anweisungen erteilt. Das Board ist dafür zuständig, dass der Servo möglichst genaue Pulsweitenmodulations-Signale erhält, um seine Position festlegen zu können. Der Pi könnte dafür aufgrund parallel ablaufender Prozesse nicht genügend Leistung aufbringen.

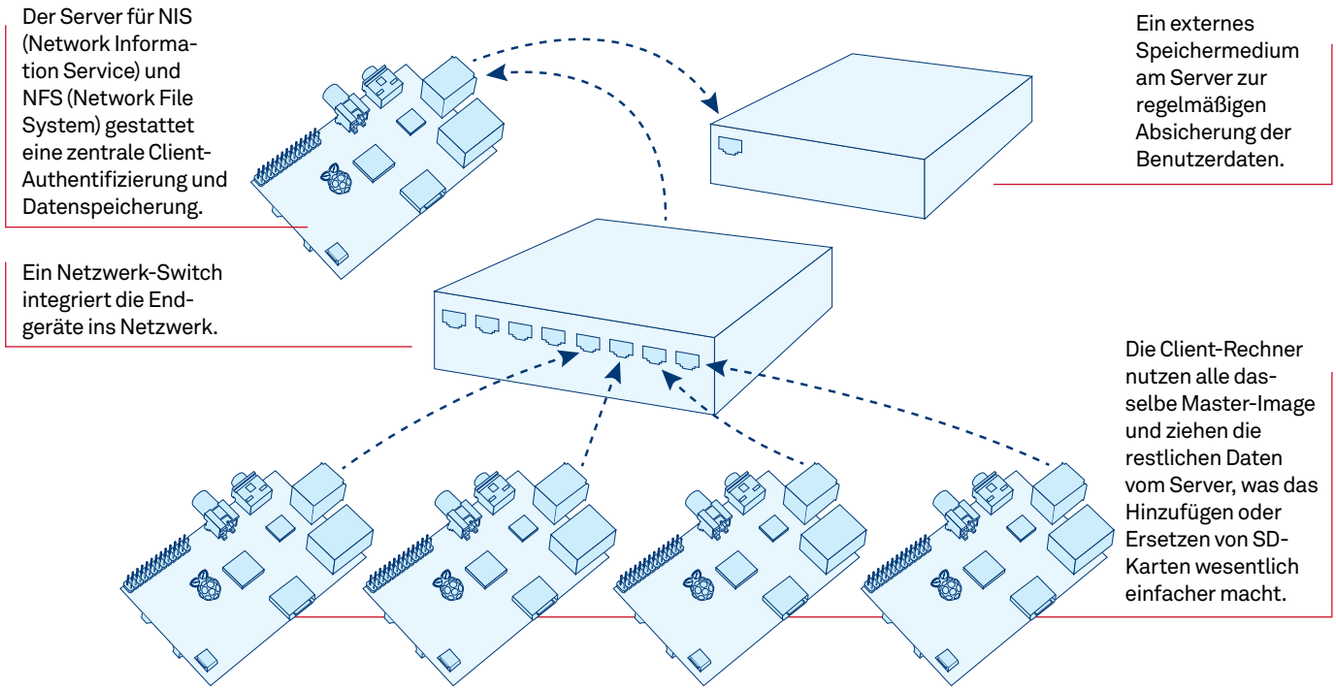
Anhand der zwei Oszilloskop-Graphen können Sie verfolgen, was der Servoregler dem Servomotor sendet. Die Abweichung zwischen Max- und Min-Positionen im Tastgrad (prozentualer Zeitanteil, in dem ein Signal aktiv ist) beträgt grob 10 %.

Der komplette Code

```
import curses
from Adafruit_PWM_Servo_Driver import PWM

# PWM-Gerät mit Standardadresse initialisieren
pwm = PWM(0x40, debug=False)

servoMin = 150  # Minimale Pulslänge aus 4096
servoMax = 600  # Maximale Pulslänge aus 4096
maxDegree = 60 # Drehwinkel des Servomotors
degIncrease = 2 # Schrittgröße der Gradzahlerhöhung

pwm.setPWMFreq(60) # PWM-Frequenz auf 60 Hz setzen

def setDegree(channel, d):
  degreePulse = servoMin
  degreePulse += int((servoMax - servoMin) / maxDegree) * d
  pwm.setPWM(channel, 0, degreePulse)

# Pfeiltastenbelegung
scr = curses.initscr()
curses.cbreak()
scr.keypad(1)
scr.addstr(0, 0, "Servo Volume Control")
scr.addstr(1, 0, "UP to increase volume")
scr.addstr(2, 0, "DOWN to decrease volume")
scr.addstr(3, 0, "q to quit")
scr.refresh()

degree = 60 # Bei geringster Lautstärke beginnen
setDegree(0, degree)

key = ''
while key != ord('q'):
  key = scr.getch()

  if key == curses.KEY_DOWN:
    degree += degIncrease

    if degree > maxDegree:
      degree = maxDegree

    setDegree(0, degree)

  elif key == curses.KEY_UP:
    degree -= degIncrease

    if degree < 0:
      degree = 0

    setDegree(0, degree)

curses.endwin()
```

06 Software laufen lassen

Platzieren Sie den Code im selben Verzeichnis, in dem schon der Adafruit-Beispielcode liegt, und lassen Sie ihn mit „sudo python2 Servo_VolumeControl.py“ laufen. Mit den Pfeiltasten nach oben und unten drehen Sie den Servo nach rechts beziehungsweise links.



Vernetzen Sie die Raspberry Pis

So richten Sie ein Netzwerk mit zentraler Authentifizierung und Datenspeicherung ein.

Was Sie brauchen:

- mindestens 2 Raspberry Pis
  mit entsprechender Peripherie (für den Server-Pi sind lediglich Strom- und Ethernetkabel nötig)
- SD-Karte
- Computer
- Netzwerk-Switch
- Ethernetkabel
- Speichermedium

In diesem Tutorial erfahren Sie, wie man einen NIS-Masterserver (Network Information Service) zur Verteilung von Linux-Konfigurationsdateien auf dem Pi einrichtet. Damit können Sie Benutzerkonten auf dem zentralen Server-Pi anlegen und so die Dateien von jedem einzelnen Client-Pi aus zugänglich machen. Dabei ist es von Vorteil, dass die Daten aller Benutzer an einem zentralen Ort gesichert werden. Ein solches Netzwerk könnte sich zum Beispiel im Schulunterricht als nützlich erweisen. Würde jede/r Schüler/in eigene Zugangsdaten zu einem schulinternen Server erhalten, so könnten mehrere Klassen sich ein paar Raspberry Pis teilen. Weiter unten folgt eine Liste von Linux-Programmen, die gut im Unterricht eingesetzt werden können.

Als Betriebssystem werden wir sowohl für den Server-Pi als auch für die Client-Pis das aktuelle Raspbian-Image verwenden. Es genügt, das Image auf insgesamt zwei SD-Karten zu spielen, da wir für alle potenziellen Clients lediglich eine Master-SD-Karte benötigen. Von dieser aus können wir später weitere SD-Karten beschreiben. Eine Anleitung zum Übertragen des Images auf eine SD-Karte finden Sie unter **linuxuser.co.uk/tutorials/how-to-set-up-raspberry-pi/**. Achten Sie jedoch darauf, dass Sie nicht wie dort beschrieben das Debian-Image kopieren, sondern das neueste Raspbian-Image. In diesem Artikel verwenden wir zur Demonstration das Image 2012-09-18-wheezy-raspbian.zip.

Natürlich setzen wir voraus, dass Ihnen ein bereits existierendes Netzwerk mit DHCP, das heißt mit einem Protokoll zur Verteilung der Netzwerkeinstellungen an angeschlossene Geräte, zur Verfügung steht. Nur so haben Sie mit Ihrem Raspberry Pi Anschluss ans Internet, um nötige Downloads und Installationen durchzuführen.

01 Erste Konfigurationen am Server-Pi

Schließen Sie Tastatur, Maus, Ethernetkabel und Bildschirm an Ihren Server-Pi an und versorgen Sie ihn mit Strom. Wenn Raspbian hochgefahren ist, wählen Sie im Konfigurationsmenü die Option expand_rootfs, um den gesamten Speicherplatz der SD-Karte nutzen zu können. Sie sollten unbedingt das Passwort für den Pi-User ändern, sodass sich niemand mit den Standard-Zugangsdaten (Nutzername „pi“, Passwort „raspberry“) in den Server einloggen kann. Stellen Sie den Memory-Split auf 240 MiB für den ARM und 16 MiB für den VideoCore ein. Aktivieren Sie SSH und wählen dann „Finish“ und „Yes“ für einen Neustart.

02 Linux-Computer anwerfen

Sie können den Server-Pi nun mit SSH konfigurieren, solange Ihr Linux-Computer im selben Netzwerk ist (wenn nicht, schließen Sie ihn jetzt an). Die Konfiguration via SSH funktioniert sogar ohne Bildschirm. Während des Startvorgangs von Raspbian erscheint eine Nachricht mit der IP-Adresse des Raspberry Pi. Öffnen Sie ein Terminal auf Ihrem Linux-Computer und geben Sie `ssh pi@[IP-Addresse des Pi]` ein. Bejahen Sie die Frage, ob Sie eine Verbindung herstellen möchten. Geben Sie dann das von Ihnen festgelegte Passwort ein, um sich in Ihren Server-Pi einzuloggen.

03 Statische IP-Adresse festlegen

Wir empfehlen, Ihrem Server-Pi eine statische IP zuzuteilen, da er so für Sie und vor allem die Client-Pis leicht identifizierbar ist. Hierfür benötigen wir zunächst einige Informationen über Ihre aktuellen Netzwerkeinstellungen. Benutzen Sie die Befehle `ifconfig eth0` und `ip route show`. Lesen Sie an den unten aufgeführten Instruktionen ab, welche Infos aus dem Output der Befehle relevant sind. Wenn wir über die wichtigsten Dinge wie die aktuelle IP-Adresse und die Netzmaske Bescheid wissen, können wir eine statische IP-Adresse bestimmen. Editieren Sie die Netzwerkkonfigurationsdatei mittels:

```
sudo nano /etc/network/interfaces
```

Navigieren Sie mit den Pfeilen und ersetzen Sie zuerst die Zeile `iface eth0 inet dhcp` mit einem Eintrag, der dem folgenden entspricht (eigene Werte einsetzen!):

```
iface eth0 inet static
address 172.17.173.249
netmask 255.255.255.0
gateway 172.17.173.1
```

Versichern Sie sich, dass die IP-Adresse, die Sie Ihrem Server-Pi zuteilen wollen, nicht schon vergeben ist. In der Regel können Sie allerdings davon ausgehen, dass die Adresse, die Ihr Gerät anfangs hatte, nicht vergeben ist. Schreiben Sie sich die IP-Adresse für zukünftige Zwecke auf.

Speichern Sie Ihre Änderungen im Nano-Editor mit der Tastenkombination Strg+O, dann Enter. Verlassen Sie Nano mit Strg+X.

Bislang haben wir uns noch nicht mit dem DNS-Server beschäftigt. DNS wandelt Hostnamen wie etwa google.com in IP-Adressen um. Die IP des DNS-Servers ist unter **/etc/resolv.conf** gespeichert. Diese verändert sich – auch nachdem Sie zu einer statischen IP-Adresse für Ihren Server gewechselt haben – nicht.

```
pi@raspberrypi ~ $ ifconfig eth0
eth0      Link encap:Ethernet
HWaddr xx:xx:xx:xx:xx:xx
          inet addr:172.17.173.249
Bcast:172.17.173.255
Mask:255.255.255.0
pi@raspberrypi ~ $ ip route show
default via 172.17.173.1 dev eth0
172.17.173.0/24 dev eth0
proto kernel  scope link  src
172.17.173.249
```

04 Server-Pi neu starten

Starten Sie den Server-Pi neu, um die soeben durchgeführten Änderungen wirksam zu machen. Loggen Sie sich erneut mit SSH und der neuen IP ein.

05 Benötigte Pakete für NFS-Server installieren

NFS (Network File System) ist für den Export der Benutzer-Homeverzeichnisse zuständig. Aktualisieren Sie die Liste der Softwarepakete mit:

```
sudo apt-get update
```

Die für einen NFS-Server erforderlichen Pakete installieren Sie anschließend mit:

```
sudo apt-get install nfs-kernel-
server-nfs-common rpcbind
```

Die Nachricht „Not starting NFS kernel daemon: no exports“ will nur darauf hinweisen, dass der NFS-Server nicht startet, da bisher noch keine Verzeichnisse exportiert werden können.

06 Verzeichnisse exportieren

Fügen Sie in Nano folgende Zeile ans Ende der Datei **/etc/exports**:

```
/home *(rw,sync)
```

Dadurch wird das Verzeichnis **/home** allgemein freigegeben und werden Lese- und Schreibberechtigungen erteilt. Natürlich ist ein Benutzer nur für sein eigenes Verzeichnis schreibberechtigt. Vor NFS müssen Sie zunächst noch rpcbind starten, das bei Raspbian standardmäßig nicht gebootet wird. Um dies zu ändern, verwenden Sie den Befehl sudo update-rc.d rpcbind enable. Mit folgenden Kommandos können Sie rpcbind und NFS unmittelbar nacheinander starten:

```
sudo /etc/init.d/rpcbind start
sudo /etc/init.d/nfs-kernel-server
start
```

07 NIS installieren

NIS (Network Information Service) war früher als Yellow Pages bekannt, deshalb tragen einige Verzeichnisdienste nach wie vor das Kürzel „yp“ im Namen. Installieren Sie NIS mit dem Befehl `sudo apt-get install nis`. Akzeptieren Sie alle zusätzlichen Pakete. Sie werden daraufhin zum Software-Konfigurationsmenü für NIS weitergeleitet. Hier müssen Sie einen beliebigen NIS-Domainnamen eingeben. Wir wählen mal „raspberrypi“. Wenn das erledigt ist, wird NIS versuchen zu starten. Wundern Sie sich nicht, wenn der Startversuch nach ein paar Minuten fehlschlägt. Es müssen noch weitere Einstellungen vorgenommen werden, um NIS zum Laufen zu bringen.

08 NIS konfigurieren

Um unseren NIS-Server startklar zu machen, müssen wir zuerst **/etc/default/nis** mit Nano öffnen. Sie müssen dem Nano-Befehl sudo voranstellen, wir brauchen nämlich Root-Rechte, um die Daten zu editieren. Ändern Sie die Zeile

```
NISSERVER=false
```

in

```
NISSERVER=master
```

und die Zeile

```
NISCLIENT=true
```

in

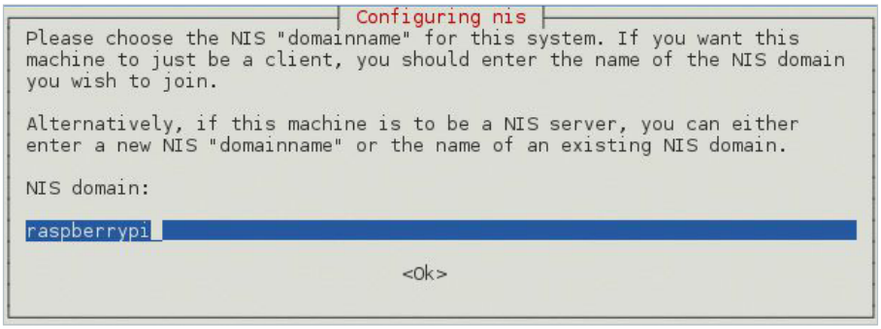
```
NISCLIENT=false
```

Speichern Sie die Änderungen und verlassen Sie Nano. Editieren Sie dann

```
/var/yp/Makefile
```

und ändern Sie dort die Zeile

»




```
ALL = passwd group hosts rpc
services netid protocols netgrp
```
in
```
ALL = passwd shadow group hosts
rpc services netid protocols netgrp
```

Die Passwort-Datei auf Linux beinhaltet eine Liste der Benutzer und deren Daten wie zum Beispiel Homeverzeichnisse und IDs. Wir müssen die Schattendatei zur Liste hinzufügen, denn sie sichert die Passwort-Hashes für jeden Benutzer. Ändern Sie außerdem die Zeile
`MINGID=1000 in MINGID=0`

Schließlich möchten wir unsere NIS-User zu Systemgruppen (wie audio) hinzufügen. Als Nächstes müssen wir den Server-Hostnamen ändern, indem wir die Datei **/etc/hostname** editieren. Unseren Hostnamen raspberrypi ändern wir in nismaster. Um diesen Schritt abzuschließen, müssen wir auch **/etc/hosts** ändern und eine zusätzliche Zeile anfügen. Dies ist nötig, da Server mit NIS in der Regel nicht mit ihrer IP, sondern namentlich eingetragen werden. Zudem sollte man den NIS-Server von den anderen Raspberry Pis unterscheiden können. Ändern Sie die Zeile
`127.0.1.1 raspberrypi`
in
`127.0.1.1 [Ihr neuer Hostname]`
und fügen Sie eine neue Zeile für die IP-Adresse des NIS-Servers wie folgt hinzu:
`[IPAddresse] [Hostname].[NIS-Domain] [Hostname]`

09 NIS-Master-Datenbank initialisieren

Mit `sudo init 6` starten Sie den Pi neu, um die Änderungen wirksam zu machen. Wenn der Rechner hochgefahren ist, verbinden Sie sich erneut und führen Sie folgenden Befehl aus:
`sudo /usr/lib/yp/ypinit -m`

Tippen Sie die Tastenkombination Strg+D, dann Enter und starten Sie den NIS-Server mit:
`sudo /etc/init.d/nis start`

10 Benutzer hinzufügen

Alle Benutzer werden auf dem NIS-Server hinzugefügt. Verwenden Sie dafür den Befehl `sudo adduser [Benutzername]` und tragen Sie die erforderlichen weiteren Informationen ein. Nach dem Hinzufügen eines neuen Benutzers aktualisieren sich die NIS-Datenbanken von selbst. Um User in Systemgruppen einzufügen, verwendet man `sudo usermod -a -G audio,video,plugdev,games,users,netdev,input [Benutzername]`. Da man damit Änderungen am User vornimmt, muss die NIS-Datenbank anschließend manuell überarbeitet werden mit:
```
cd /var/yp
sudo make
```
Als Benutzer können Sie Ihr Passwort mit `sudo passwd [Benutzername]` auf dem NIS-Server ändern. Sie müssen dann allerdings erneut die Datenbank überarbeiten.

11 Client erstellen

Nun ist es an der Zeit, das Image eines Clients zu bearbeiten. Schließen Sie den zweiten Raspberry Pi an, und setzen Sie die SD-Karte mit dem unbenutzten Raspbian-Image ein. Wie immer wird sich raspi-config öffnen. Die expand_rootfs-Option ist die einzige, die wir jetzt brauchen. Wählen Sie „Finish“ und „Yes“ für den Neustart.

12 Einloggen und erforderliche Pakete installieren

Loggen Sie sich mit den Standard-Zugangsdaten (Nutzername „pi“, Passwort „raspberry“) ein und führen Sie `sudo apt-get update` aus. Installieren Sie die erforderlichen Softwarepakete für NFS- und NIS-Clients mit dem Befehl `sudo apt-get install nis rpcbind nfs-common`. Nehmen Sie im NIS-Konfigurationsmenü die gleichen Änderungen vor wie zuvor. Wieder wird NIS nicht starten können, weil noch einige Vorkehrungen getroffen werden müssen, wie zum Beispiel das Booten von rpcbind mittels `sudo update-rc.d rpcbind enable`. Wir werden rpcbind allerdings schon in Kürze benötigen, deshalb starten wir es gleich mit `sudo /etc/init.d/rpcbind start`.

13 Server-Homeverzeichnis mounten

Nun richten wir ein Server-Homeverzeichnis ein. Wir möchten, dass das Verzeichnis bei jedem Systemstart erscheint, also machen wir einen Eintrag in **/etc/fstab**. Öffnen Sie die Datei in Nano und fügen Sie eine Zeile mit folgendem Format ans Ende der Datei an:
`[Server-IP]:/home /home nfs defaults 0 0`

Speichern Sie die Änderungen. Mit `sudo mount -a` können Sie alle Einträge in **/etc/fstab** mounten. Binden Sie hier das Homeverzeichnis des Servers an den Client. Wenn Sie nun den Befehl `ls /home` eingeben, werden sowohl der zuvor erstellte Test-User als auch der Server-User erscheinen.

```
pi@raspberrypi /home $ cat /etc/
fstab
proc            /proc           proc
defaults          0       0
/dev/mmcblk0p1  /boot           vfat
defaults          0       2
/dev/mmcblk0p2  /               ext4
defaults,noatime  0       1
# Da ein Swapfile keine
Swap-Partition ist, nehmen wir
statt swapon/swapoff ab hier
dphys-swapfile swapon/swapoff
172.17.173.249:/home /home nfs
defaults 0 0
pi@raspberrypi /home $ ls /home
pi testuser1
```

14 NIS-Client anmelden

Öffnen Sie **/etc/yp.conf** in Nano (sudo nicht vergessen!) und fügen Sie folgende Zeile hinzu:
```
ypserver [Server-IP]
domain [NIS-Domain] server [Server-
Hostname]
```
Nun müssen Sie noch die Datei **/etc/nsswitch.conf** anpassen, bevor Sie den Client mit `sudo init 6` neu starten.

“Viele Linux-Programme eignen sich sehr gut für den Unterricht.”

15 Login als User des NIS-Servers

Wenn Sie beim Neustart versuchen, sich mit Ihrem Pi-User anzumelden, werden Sie feststellen, dass Ihr altes Kennwort nicht mehr funktioniert. Das liegt daran, dass wir uns nun als Benutzer des Servers einloggen.

Das Schöne an der Sache ist, dass man sich am Pi jederzeit mit den Standard-Zugangsdaten anmelden kann, solange dieser nicht am Netzwerk hängt oder wenn der Server down ist. Das Gerät wird sich allerdings während des Startvorgangs ein wenig beschweren, weil es keinen Zugang zum Homeverzeichnis am Server findet.

16 Pakete auf Client installieren

Das Raspbian-Image enthält bereits eine gute Auswahl an Software, doch wir möchten Ihnen zusätzliche Linux-Lernsoftware nahelegen, die mit dem Master-Image für die Clients kompatibel ist. Folgende Programme eignen sich hervorragend für den Einsatz im Schulunterricht:

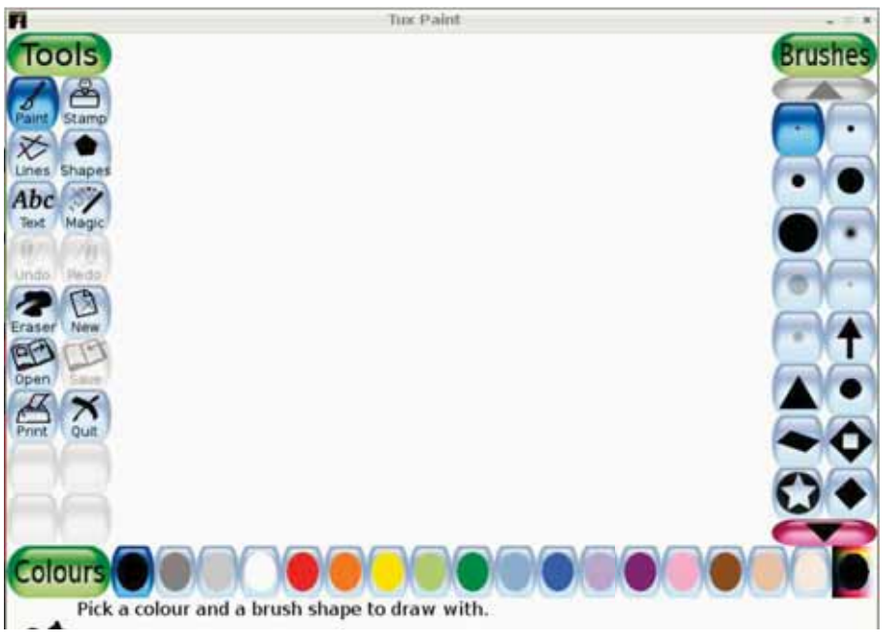
AbiWord – Standard-Textverarbeitung
Gnumeric – Standard-Tabellenkalkulation
GCompris – Lernsoftware-Paket für Kinder
Galculator – Wissenschaftlicher Taschenrechner
TuxMath – Mathematik-Lernhilfe für Kinder
Tux Paint – Kinder-Zeichenprogramm

Sie können mit `sudo apt-get install abiword gnumeric gcompris galculator tuxmath tuxpaint` all diese Programme auf einmal installieren. Ist die Installationen abgeschlossen, tippen Sie `startx`, um zur LXDE-Desktop-Umgebung zu gelangen. Spielen Sie ruhig ein wenig mit der neuen Software herum.

Probieren Sie das Gleiche auch mal als Test-User. Sie werden sehen, dass die Desktop-Symbole zwar nur für den Pi-User verfügbar sind, Sie erreichen die Programme aber dennoch über das Startmenü. Verlassen Sie LXDE, um zum Bedienungsfeld zurückzukehren. Tippen Sie dann `logout`.

17 SD-Karte clonen

Bevor es weitergeht, schließen Sie den Client-Pi mit `su pi` und `sudo init 0`. Die Linux-Inhalte werden als Datei angezeigt. Wenn wir den cat-Befehl in Kombination mit Pipes verwenden, können wir die SD-Karte also einfach duplizieren. Wechseln Sie mit `su root` zum Root-Konto auf Ihrem Linux-Computer. Ermitteln Sie aus dem Output des Befehls `fdisk -l | grep Disk` den Pfad für die SD-Karte. Unserer lautet **/dev/mmcblk0**. Mit `cat /dev/mmcblk0 > nisclient.img` fertigen Sie eine Kopie der Karteninhalte an. Wenn Sie eine neue SD-Karte einsetzen, können Sie die Kopie nun auf diese übertragen, indem Sie `cat nisclient.img > /dev/mmcblk0` ausführen.



18 Externe Festplatte für Backups verwenden

Eine Anleitung zur Einrichtung einer externen Festplatte für die Erstellung von Backups finden Sie auf **linuxuser.co.uk/features/build-a-fileserver-with-the-raspberry-pi.** Siehe Schritte 09 bis 11.

19 Server absichern

Eine zentral gesteuerte Datensicherung bietet den Vorteil, dass alle Inhalte von einem Standort aus abgesichert und wiederhergestellt werden können. Um Sicherheitskopien zu erstellen, müssen wir rsync installieren, führen Sie daher `sudo apt-get install rsync` aus. Von der Annahme ausgehend, dass sich der Aufbewahrungsort Ihrer Backups in **/mnt/data** befindet, nutzen wir den Befehl

```
sudo rsync -avP /home /mnt/data
```

zur Durchführung der Backups. Die einzelnen Switches haben folgende Bedeutungen:

a – Berechtigungen und Eigenschaften der Datei beibehalten
v – Lognachrichten anzeigen
P – Fortschritt anzeigen

Durch den Befehl werden alle Änderungen zwischen **/home** und **/mnt/data** synchronisiert. Dennoch gehen ältere Dateien, die aus **/home** gelöscht wurden, nicht verloren, denn rsync sieht eine geänderte Datei nicht als neu an, sondern kopiert lediglich die Änderungen an der alten Datei. Da wir den Befehl regelmäßig durchführen wollen, müssen wir ihn in cron hinzufügen. Führen Sie dafür crontab mit dem Befehl `crontab -e` aus und setzen Sie folgende Zeile an das Ende der Datei:

```
0 * * * * ‘sudo rsync -avP /home /
mnt/data’
```

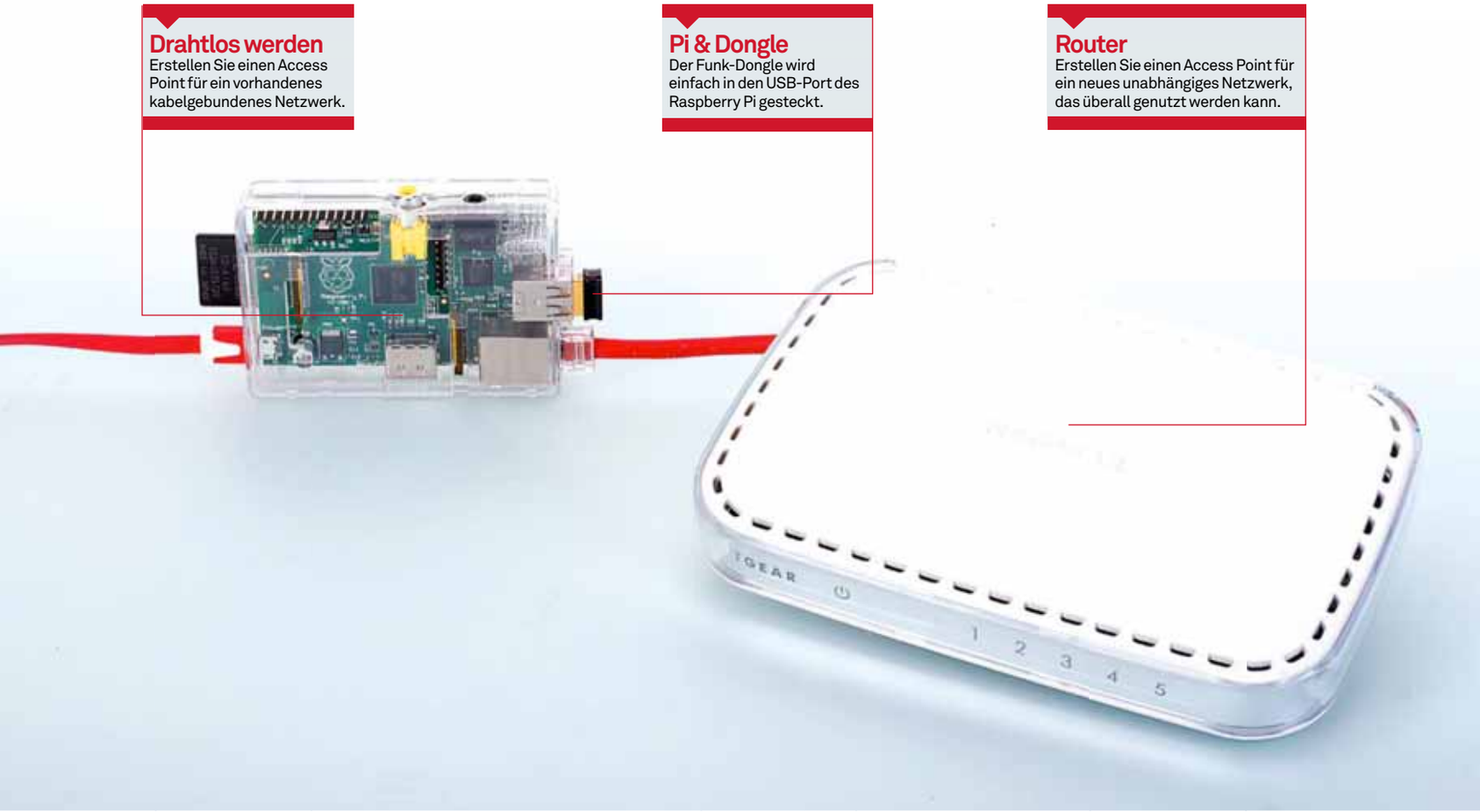
Die crontab-Datei akzeptiert Einträge im Format: Minuten, Stunden, Tag des Monats, Monat, Wochentag (ohne Kommas). Auf Basis obiger Zeile würde cron zu Beginn jeder Stunde ein Backup erstellen. Speichern Sie die Änderungen wie üblich in Nano.

20 Das war's!

Jetzt haben Sie eine zentrale Authentifizierung und Datenspeicherung sowie ein Master-Image mit praktischer Software für die Client-Rechner. Viel Spaß damit!

Der Pi als vielseitiger Wireless-Helfer

Wie man eine drahtlose Verbindung zum Raspberry Pi oder zu einem mit ihm verbundenen Netzwerk herstellt.



Was Sie brauchen:

- aktuelles Raspbian-Image
  raspberrypi.org/downloads
- einen Router oder Switch
- einen Wireless-USB-Dongle, der den Access-Point-Modus unterstützt
- ein WLAN-Gerät zum Testen

Ein Raspberry Pi mit einem Funk-Dongle, der den Access-Point-Modus unterstützt, ist ein sehr vielseitiges Werkzeug. Der Pi kann sich dabei sowohl mit vorhandenen Drahtlosnetzwerken verbinden als auch selber zu einem Access Point für mehrere Geräte werden. Er kann überdies ein existierendes kabelgebundenes Netzwerk mit Wireless-Funktionalität ausstatten.

Eine potenzielle Anwendung ist die Erstellung eines drahtlosen Access Points für Gäste (jedoch mit geeigneten Firewall-Regeln, um sicherzustellen, dass niemand auf Ihr LAN zugreifen kann). Ein weiteres Beispiel wäre eine Verbindung mit dem Raspberry Pi per Funk statt über das Ethernet, was nützlich sein kann, wenn der Ethernet-Port bereits in Gebrauch ist. Zum Beispiel würde ein Pi, der als portabler PXE-Server fungiert (und einen Computer über das Netzwerk startet), über ein Ethernet-Kabel mit dem Computer, den er startet, verbunden sein, während Sie selber sich gleichzeitig separat drahtlos einloggen könnten.

01 Software installieren

Loggen Sie sich mit dem Benutzernamen „pi“ und dem Passwort „raspberry“ in Raspbian ein. Holen Sie sich mit folgendem Befehl die neuesten Paketlisten:

```
sudo apt-get update
```

Wir verwenden hostapd, einen Daemon für Access-Point-Management und -Authentifizierung. Zudem benötigen wir bridge-utils, um das Ethernet per Bridge mit den drahtlosen Schnittstellen zu koppeln. iw wird verwendet, um Informationen über die drahtlose Schnittstelle zu erhalten, und dnsmasq dient als DHCP-Server. Installieren Sie diese Programme mit:

```
sudo apt-get install hostapd bridge-utils iw
dnsmasq
```

02 AP-Modus vorhanden?

Es ist wichtig, dass Sie einen Funk-Dongle benutzen, der den Access-Point-Modus (AP) unterstützt. Um das nachzuprüfen, schließen Sie den Dongle an und tippen Sie **iw list**. In der Liste gibt es einen Bereich, der die unterstützten Schnittstellen-Modi anzeigt. Dort muss „AP“ stehen. Die Liste könnte folgendermaßen aussehen:

```
Supported interface modes:
        * IBSS
        * managed
        * AP
        * AP/VLAN
        * WDS
        * monitor
        * mesh point
```

03 Bridge-Schnittstellen

Wir koppeln die Ethernet-Schnittstelle per Bridge mit der Wireless-Schnittstelle, damit die beiden sich eine IP-Adresse teilen können und der Datenverkehr zwischen ihnen fließen kann. Editieren Sie hierfür **/etc/network/interfaces** (verwenden Sie `auto lo br0`

```
iface lo inet loopback

allow-hotplug eth0
iface eth0 inet manual

allow-hotplug wlan0
iface wlan0 inet manual

iface br0 inet dhcp
    bridge_ports eth0 wlan0
    bridge_waitport 0
```

Einstweilen führt die Bridge zu einem bereits vorhandenen Netzwerk, aber wir richten später ein eigenes ein. Starten Sie mit **sudo reboot** neu, damit die Änderungen wirksam werden.

04 hostapd konfigurieren

Zuerst müssen wir **/etc/default/hostapd** editieren, um den Daemon beim Booten zu starten. Kommentieren Sie die vorhandene Option „DAEMON_CONF“ aus und fügen Sie stattdessen ein:

```
DAEMON_CONF=”/etc/hostapd/hostapd.conf”
```

Anschließend erstellen wir die Konfigurationsdatei für hostpad mit dem obengenannten Pfad. Sie sollte folgenden Inhalt haben:

```
interface=wlan0
bridge=br0
driver=nl80211
country_code=DE
ssid=pinet
hw_mode=g
channel=1
wpa=2
wpa_passphrase=super_secret
wpa_key_mgmt=WPA-PSK
ieee80211n=1
wmm_enabled=1
```

Sollten Sie kein Wireless-N-fähiges Gerät besitzen oder Stabilitätsprobleme beobachten, empfehlen wir, die letzten beiden Zeilen wegzulassen. Sie müssen möglicherweise auch den Funkkanal ändern, wenn Sie Stabilitätsprobleme haben. Sobald Sie das getan haben, können Sie den hostapd-Daemon per **sudo/etc/init.d/hostapd start** starten.

05 Mit dem Netzwerk verbinden

Der SSID in der Konfigurationsdatei ist der Netzwerkname, und die WPA-Passphrase ist das Kennwort. Wenn Sie sich mit dem Netzwerk verbinden, wird die Verbindung per Bridge mit dem vorhandenen kabelgebundenen Netzwerk gekoppelt. Sie sollten eine IP-Adresse von Ihrem vorhandenen Router erhalten, der DHCP unterstützt. Glückwünsch zur Erstellung Ihres Access Points!

06 Unabhängiges Netzwerk erstellen

Wir ändern nun die Netzwerkkonfiguration, um unabhängig von Ihrem vorhandenen Netzwerk zu sein. Zuerst sollten Sie einen privaten Adressbereich wählen, den Sie verwenden möchten. In diesem Beispiel nehmen wir 192.168.0.0/24. Jedoch empfehlen wir, einen zufälligen Adressbereich zu wählen, falls Sie jemals andere Netzwerke ansteuern möchten. Hierfür weisen wir zuerst der Bridge-Schnittstelle eine statische IP-Adresse zu. Ändern Sie den Bridge-Bereich von **/etc/network/interfaces** wie folgt:

```
iface br0 inet static
    bridge_ports eth0 wlan0
    bridge_waitport 0
    address 192.168.0.1
    network 192.168.0.0
    netmask 255.255.255.0
```

07 DHCP-Server konfigurieren

Wir haben im ersten Schritt dnsmasq installiert. Dessen Konfiguration als DHCP-Server ist einfach. Erstellen Sie zuerst eine Kopie der ursprünglichen Konfigurationsdatei:

```
sudo cp /etc/dnsmasq.conf /etc/dnsmasq.conf.orig
```

Dann editieren Sie **/etc/dnsmasq.conf**. Die Datei muss die folgenden Zeilen bekommen:

```
interface=br0
dhcp-range=192.168.0.2,192.168.0.254,255.255.255.0,12h
```

Führen Sie einen Neustart durch, damit die Änderungen wirksam werden. Sie sollten jetzt den Raspberry Pi von Ihrem vorhandenen Netzwerk trennen, sonst sind zwei DHCP-Server im Netzwerk vorhanden.

08 Ausprobieren

Geräte an der Wireless-Schnittstelle können nunmehr mit Geräten an der kabelgebundenen Schnittstelle kommunizieren, und jedes an den Raspberry Pi angeschlossene Gerät erhält eine IP-Adresse, sodass es mit den anderen Geräten kommunizieren kann.

CPU-Leistung spenden mit BOINC

Stellen Sie mithilfe des BOINC-Clients überschüssige Rechenleistung Ihres Raspberry Pi für gute Zwecke zur Verfügung.

Was Sie brauchen:

- SD-Karte mit Raspbian-Image
  raspberrypi.org/downloads
- die BOINC-Client-Software
  boinc.berkeley.edu/download_all.php
- ein BOINC-Projekt
  boinc.berkeley.edu/projects.php

Manche Leute haben rund um die Uhr Verwendung für ihren Raspberry Pi. Ob damit ein Roboter gesteuert, ein Mediacenter betrieben oder die Heimautomatisierung besorgt wird, es gibt viele Arten, den Minicomputer im Dauereinsatz zu halten.

Manch ein Pi wird jedoch weit seltener genutzt. Wenn Sie nicht Ihre eigene Bitcoin-Farm aufziehen möchten (siehe Seite 88 in diesem Heft), könnten Sie seine Rechenleistung auch einem guten Zweck zukommen lassen. Das funktioniert mit dem Dienst BOINC. Die konkreten Ziele sind dabei vielfältig. Der Raspberry Pi könnte zum Beispiel Daten des CERN-Teilchenbeschleunigers Large Hadron Collider durchackern, für die Medizin Proteine falten oder nach extraterrestrischem Leben suchen.

01 Raspberry Pi einrichten

Bevor wir BOINC installieren, müssen wir Raspbian und den Raspberry Pi konfigurieren. Öffnen Sie ein Terminal und geben Sie „sudo raspi-config“ ein, um die Konfigurationsoptionen aufzurufen. Gehen Sie in die „Advanced Options“ und suchen Sie den Punkt „Memory Split“. Ändern Sie den dortigen Wert auf „16“, bestätigen Sie mit „OK“, und rebooten Sie mit „Finish“.

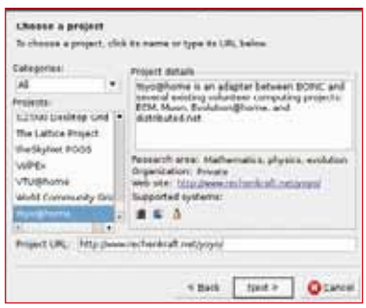
02 BOINC-Software installieren

Nun können wir BOINC über das Raspbian-Repository zusammen mit einigen anderen für die Verwendung nötigen Paketen installieren. Sobald Raspbian neu gestartet ist, geben Sie in ein Terminal Folgendes ein:

```
$ sudo apt-get install boinc-
manager boinc-client
```

03 Erster Start

Falls Raspbian den Desktop noch nicht gestartet hat, holen Sie das mit „startx“ nach. Gehen Sie zum Programm-Menü, darin auf „System Tools“, und suchen Sie den BOINC-Manager. Klicken Sie darauf, um ihn zu öffnen. Damit startet der Prozess, sich ein neues Benutzerkonto anzulegen und neue Projekte hinzuzufügen.



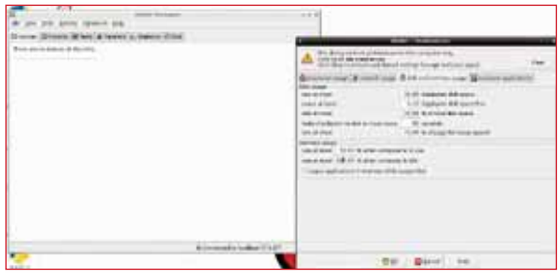
04 Neues Projekt hinzufügen

Hier können wir ein neues BOINC-Projekt hinzufügen. Falls Sie sich noch nicht für eines entschieden haben sollten, können Sie zunächst das Angebot studieren. Stellen Sie sicher, dass „Add Project“ ausgewählt ist, und klicken Sie auf „Next“. Es gibt viele Kategorien, die Sie wählen können, um Ihre Suche einzuschränken. Wählen Sie ein Projekt und klicken Sie abermals auf „Next“.

CERN-Daten durchackern, Proteine falten, nach ET suchen.

05 Identifizieren Sie sich

Nachdem Sie auf „Next“ geklickt haben, müssen Sie einen BOINC-Account anlegen. Sie brauchen für jedes Projekt einen neuen Account. So können Sie auch mehrere Geräte für ein Projekt unter Ihrem Namen rechnen lassen. Legen Sie das Nutzerkonto an oder klicken Sie auf „Login“, dann auf „Finish“, wenn Sie fertig sind. Manche Projekte werden Sie auch bitten, Ihrem Gerät auf einer separaten Webseite einen Namen zu geben.



06 BOINC konfigurieren

Klicken Sie auf „View“ in der Leiste oben und wählen Sie „Advanced View“, um zusätzliche Einstellungen für BOINC zu sehen. Gehen Sie zu Tools > Computing Preferences, und wählen Sie den Tab „Disk and memory usage“. Stellen Sie „Memory usage“ auf „100.00 % when computer is idle“, um den Raspberry Pi bestmöglich auszulasten.

07 Weitere Projekte hinzufügen

Klicken Sie auf View > Simple View, um zum ursprünglichen Fenster zurückzukehren. Im unteren Teil des Fensters gibt es einen „Add Project“-Button, der es Ihnen erlaubt, nicht nur weitere BOINC-zertifizierte Projekte hinzuzufügen, sondern auch andere, die nicht zwingend im offiziellen Verzeichnis gelistet sein müssen.

08 Aus Kommandozeile starten

Falls der Raspberry Pi die Berechnungen außerhalb des normalen Systems erledigen soll, bevorzugen Sie es vielleicht, sich per SSH am Raspberry Pi anzumelden. BOINC hat seine eigene Kommandozeilenversion, die Sie wie folgt aufrufen:

```
$ boinccmd
```

09 Daten durchackern

BOINC startet automatisch, sobald der Raspberry Pi eingeschaltet wird. Suchen Sie also eine Stelle, wo das Gerät angemessen mit Energie versorgt wird und kühl bleibt, und lassen Sie es seine Arbeit verrichten. Sie können weitere Projekte mit der Kommandozeile hinzufügen, oder indem Sie wieder einen Monitor anschließen.

Die Pi-Kamera als Bewegungsmelder

Setzen Sie die Bewegungserkennung der Pi-Kamera für Home-Security-Zwecke ein oder erstellen Sie Zeitrafferaufnahmen.

Was Sie brauchen:

- SD-Karte mit aktuellem Raspbian-Image
- Python-Modul picamera
- Raspberry-Pi-Kameramodul

Die Pi-Kamera ist ein eindrucksvolles und vielseitiges Gerät. Ausgestattet mit anständiger Sensorik, Videofunktionalität und dem Python-Modul „picamera“, bietet sie ein breites Anwendungsspektrum. Die Überwachungs- und Zeitrafferfunktionen der Kamera lassen sich wunderbar kombinieren. Wenn die Kamera nur bei Bewegung aktiviert wird, schont dies einerseits die Ressourcen des Pi bei Überwachungsvorgängen und optimiert andererseits die Zeitrafferfunktion bei menschlichen Motiven. Wie es funktioniert und wie Sie Ihre Bilder auf spezielle Weise speichern können, zeigen wir Ihnen hier.

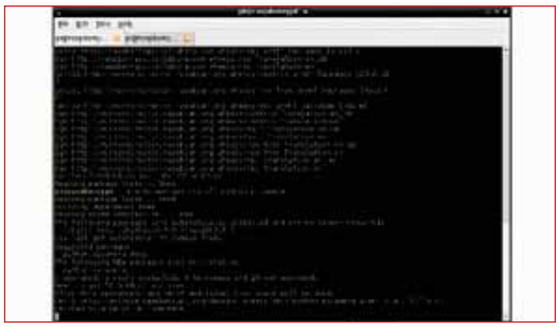
Oben Dank des Baukastenprinzips sind Demontage und benutzerspezifischer Umbau einfach zu bewerkstelligen.

01 Kameramodul aktivieren

Bringen Sie zuerst Ihren Pi und seine Firmware mit **apt-get update**, **apt-get upgrade** und **rpi-update** auf den neuesten Stand. Geben Sie dann im Terminal **raspiconfig** ein, um in die Konfigurationsdatei zu gelangen. Wählen Sie dort „Enable“ in der Kameraoption und „Finish“ für den Neustart.

02 Kamera anschließen

Pi ausschalten, Stromzufuhr kappen und nach der schmalen Schnittstelle zwischen Ethernet- und HDMI-Port suchen. Öffnen Sie die Klemmen und stecken Sie das Flachbandkabel der Kamera mit der silbernen Seite in Richtung HDMI-Anschluss ein.



03 Python-Module installieren

Der Code benötigt noch ein paar zusätzliche Python-Module, nämlich picamera für eine leichtere Bedienung der Kamera sowie die Python Image Library, um Bilder einsehen zu können. Installieren Sie beide mit:

```
$ sudo apt-get install python-picamera python-imaging-tk
```

04 Kamera in Stellung bringen

Hat der Pi noch Zugang zur Stromversorgung? Bedienen Sie Ihre Kamera via SSH, oder sollte sich ein Bildschirm beziehungsweise eine Tastatur oder Maus in Reichweite befinden? Stellen Sie sich solche Fragen, um herauszufinden, wo der richtige Platz für Ihre Kamera ist. Machen Sie ein Testbild mit:

```
$ raspistill -o test.jpg
```

05 Sensitivität einstellen

In unserem Code bestimmen Abweichung und Pixel, wann ein Bild aufgenommen wird. Die Abweichungs-Variable bestimmt den Grad der Farbveränderung eines Pixels, während die Pixel-Variable die Anzahl der veränderten Pixel angibt. Anhand letzterer wird ermittelt, ob genügend Bewegung für eine Bildaufnahme gegeben ist.

06 Stream-Speicher

Wir können ein Testbild machen und es in einem Stream, den wir mit **io.BytesIO** erzeugen, sichern. Beim Vergleich mit dem zuvor aufgenommenen Bild, das nachträglich im Stream gespeichert wurde, wird entschieden, ob ein weiteres Bild aufgenommen wird.

Wenn die Kamera nur bei Bewegung aktiviert wird, schont dies die Ressourcen des Pi.

Der komplette Code

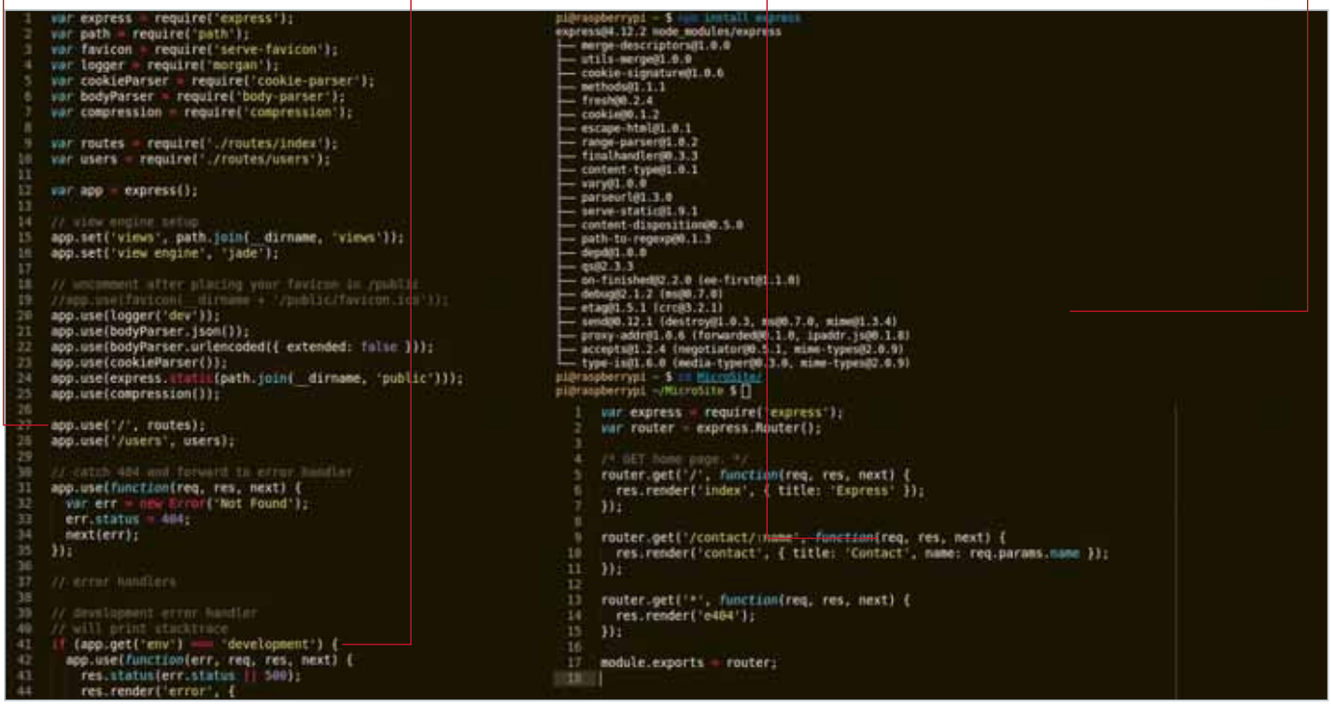
```
import io
import os
import picamera
import time
from datetime import datetime
from PIL import Image

camera = picamera.PiCamera()

difference = 20
pixels = 100

width = 1280
height = 960

def compare():
    camera.resolution = (100, 75)
    stream = io.BytesIO()
    camera.capture(stream, format = 'bmp')
    stream.seek(0)
    im = Image.open(stream)
    buffer = im.load()
    stream.close()
    return im, buffer

def newimage(width, height):
    time = datetime.now()
    filename = "motion-%04d%02d%02d-%02d%02d%02d.jpg" % (time.
year, time.month, time.day, time.hour, time.minute, time.second)
    camera.resolution = (width, height)
    camera.capture(filename)
    print "Captured %s" % filename

image1, buffer1 = compare()

timestamp = time.time()

while (True):

    image2, buffer2 = compare()

    changedpixels = 0
    for x in xrange(0, 100):
        for y in xrange(0, 75):
            pixdiff = abs(buffer1[x,y][1] - buffer2[x,y][1])
            if pixdiff > difference:
                changedpixels += 1

    if changedpixels > pixels:
        timestamp = time.time()
        newimage(width, height)

    image1 = image2
    buffer1 = buffer2
```

Das Aufsetzen von Routen ist wirklich einfach. Wir erklären es Zeile für Zeile.

Es können umgebungsspezifische Einstellungen vorgenommen werden.

Auf speziellen Routen können Variablen an Views weitergegeben werden.

Module installieren, debuggen und Anwendungen starten wird hauptsächlich von der Kommandozeile aus erledigt.



Einen Node.js-Webserver auf dem Pi entwickeln

Haben Sie schon mal daran gedacht, eine Node.js-basierte Website auf Ihrem Pi laufen zu lassen?

Was Sie brauchen:

- Node.js-Quellcode
  nodejs.org/dist/v0.10.28/node-v0.10.28-linux-arm-pi.tar.gz
- OpenSSL-Entwicklerpaket libssl-dev

Seit der Einführung des Raspberry Pi kann man einen leistungsfähigen Webserver über einen einzigen USB-Anschluss betreiben. Dies wird auch durch Node.js, eine aufstrebende neue JavaScript-Plattform für den Server, ermöglicht. Node setzt auf Googles JavaScript-Engine V8 auf und bringt eine eventgesteuerte, asynchrone (non-blocking) I/O-Architektur mit. JavaScript hat seit dem durch V8 gestarteten Wettlauf in Sachen Browser-Performance rasant an Popularität und Geschwindigkeit hinzugewonnen. Bedenkt man zudem die weite Verbreitung von JavaScript, erscheinen Node.js und der Raspberry Pi wie füreinander geschaffen.

Dieses Tutorial beleuchtet das Aufsetzen einer Node-Entwicklungsumgebung auf Ihrem Raspberry Pi und den Bau einer selbst gehosteten Website mit reinem JavaScript und zeigt, wie man Bibliotheken ganz einfach in der Modul-Registry finden und einbinden kann. Es soll Ihnen einen Einstieg bieten und eine Grundlage für eigene Entdeckungsreisen in die Welt von Node.js sein. Worauf warten Sie also noch? Lassen Sie uns beginnen.

```
pi@raspberrypi ~ $ express MicroSite

   create : MicroSite
   create : MicroSite/package.json
   create : MicroSite/app.js
   create : MicroSite/public
   create : MicroSite/public/javascripts
   create : MicroSite/routes
   create : MicroSite/routes/index.js
   create : MicroSite/routes/users.js
   create : MicroSite/public/images
   create : MicroSite/public/stylesheets
   create : MicroSite/public/stylesheets/style.css
   create : MicroSite/views
   create : MicroSite/views/index.jade
   create : MicroSite/views/layout.jade
   create : MicroSite/views/error.jade
   create : MicroSite/bin
   create : MicroSite/bin/www

   install dependencies:
     $ cd MicroSite && npm install

   run the app:
     $ DEBUG=MicroSite:* ./bin/www

pi@raspberrypi ~ $
```

01 Konfiguration

Um Node.js auf dem Raspberry Pi zu installieren, müssen Sie den Quellcode herunterladen und kompilieren. Die empfohlene Distribution (Debian „Squeeze“) enthält nicht die aktuelle Version von Node.js. Für Arch Linux ARM ist ein von der Community gepflegtes Paket über den Pacman-Installer verfügbar.

02 Compiler-Flags setzen

Um sicherzugehen, dass Node.js und V8 für die richtige Architektur (ARMv6) gebaut werden, müssen wir die Compiler-Flags für C und C++ von Hand setzen, indem wir die Umgebungsvariablen „CFLAGS='-march=armv6'“ und „CXXFLAGS='-march=armv6'“ temporär exportieren.

03 Build und Install

Im selben Verzeichnis müssen wir nun die Kommandos „./configure“, „make“ und „sudo make install“ ausführen. Beachten Sie, dass die Ausführung von „make“ etwa 40 bis 90 Minuten dauert! Ob alles geklappt hat, kann dann mit „which node“ und „which npm“ überprüft werden.

04 Module installieren

Es gibt eine Reihe von Node.js-Frameworks für die schnelle Entwicklung von Anwendungen. Das De-facto-Modul für Webentwicklung ist Express (**expressjs.com**). Es kann durch den Befehl „sudo npm install -q -g express express-generator“ global installiert werden.

05 Anwendung aufsetzen

Nach der Installation können wir nun „express MicroSite“ ausführen. Dies legt ein Verzeichnis namens MicroSite mit einer Beispielanwendung darin an. Dann müssen wir in diesen Ordner wechseln und mit „cd MicroSite && npm install“ die Abhängigkeiten installieren.

06 Server starten 1

Damit haben wir aber noch keine komplette Node.js-App, denn es fehlt der Code, um den Server zu starten. Ersetzen Sie daher die letzte Zeile ihrer app.js („module.exports = app;“) durch den folgenden Code:

07 Server starten 2

Node.js spricht von Haus aus HTTP, sodass kein externer Webserver benötigt wird, wie sonst etwa Apache für PHP. Mit „node app.js“ starten Sie einen Webserver unter **http://localhost:3000/**. Wenn alles funktioniert, wird eine Website mit den Worten „Welcome to Express“ angezeigt.

```
<h1>Express</h1>
<p>Welcome to Express</p>
<p style="color:#666;">Node.js running
```

```
h1= title
p Welcome to #{title}
p(style='color:#66;') Node.js running
```

08 Templates verwenden

Express benutzt eine Template-Sprache namens Jade (**jade-lang.com**) anstelle von HTML. Diese Dateien sind im Ordner „/views“ zu finden und können dynamische Inhalte und Variablen ausgeben. Sehen Sie sich den Code an, Sie werden subtile Unterschiede beim Markup erkennen.

```
var express = require('express');
var path = require('path');
var favicon = require('serve-favicon');
var logger = require('morgan');
var cookieParser = require('cookie-parser');
var bodyParser = require('body-parser');

var routes = require('./routes/index');
var users = require('./routes/users');
```

09 Module einbinden

Nun da die Beispielanwendung läuft, sehen wir uns einmal im Quelltext von „app.js“ an, wie Module geladen werden. In dem gezeigten Code sind unter anderem Express und der Ordner „routes“ eingebunden und jeweils Variablen zugewiesen. Diese Variablen befinden sich im lokalen Geltungsbereich der Datei.

```
var app = express();
```

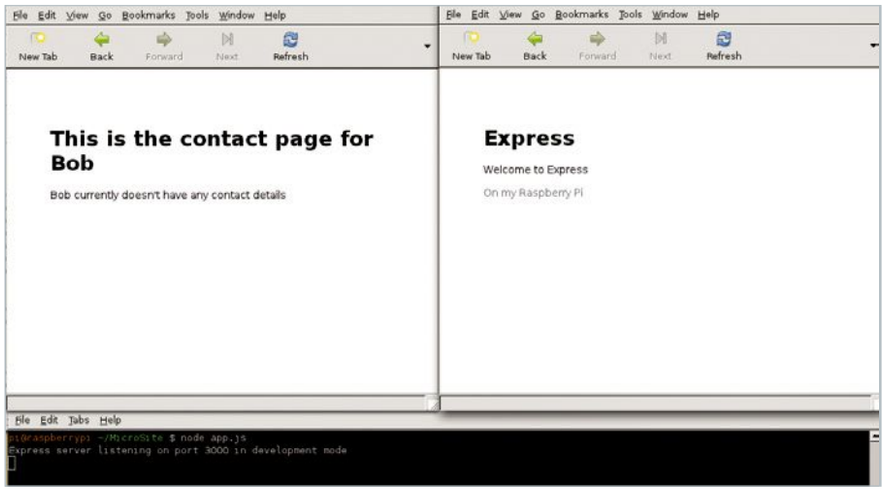
10 Erzeugen der App

Das Herzstück unserer Anwendung ist das App-Objekt, das wir erhalten, indem wir die Methode express() aufrufen und ihren Rückgabewert einer Variablen – in diesem Fall „app“ – zuweisen. Das Express-App-Objekt bietet Methoden zur Konfiguration und zum Hinzufügen von Routen.

```
// view engine setup
app.set('views', path.join(__dirname, 'views'));
app.set('view engine', 'jade');

// uncomment after placing your favicon in /public
//app.use(favicon(__dirname + '/public/favicon.ico'));
app.use(logger('dev'));
app.use(bodyParser.json());
app.use(bodyParser.urlencoded({ extended: false }));
app.use(cookieParser());
app.use(express.static(path.join(__dirname, 'public')));

app.use('/', routes);
app.use('/users', users);
```

11 Globale Konfiguration

Einstellungen können für eine bestimmte Umgebung vorgenommen werden. Hier fügen wir gemeinsame Konfigurationen und Middleware zur Benutzung in allen Umgebungen hinzu, wie zum Beispiel das Setzen der View Engine.




```
// development error handler
// will print stacktrace
if (app.get('env') === 'development') {
  app.use(function(err, req, res, next) {
    res.status(err.status || 500);
    res.render('error', {
      message: err.message,
      error: err
    });
  });
}

// production error handler
// no stacktraces leaked to user
app.use(function(err, req, res, next) {
  res.status(err.status || 500);
  res.render('error', {
    message: err.message,
    error: {}
  });
});
```

12 Umgebungsbezogene Konfiguration

Mithilfe dieses if-Blocks unterbinden wir die Anzeige von Stack Traces auf öffentlichen Websites. „app.get(‚env‘)“ gibt die Umgebung zurück, deren Wert von NODE_ENV kommt und beim Starten der Anwendung durch „NODE_ENV=‘production‘ node app.js“ gesetzt werden kann.

```
/* GET home page. */
router.get('/', function(req, res, next) {
  res.render('index', { title: 'Express' });
});
```

13 Requests routen

Der Router unter „routes/index.js“ antwortet momentan nur auf GET-Requests gegen die „/“-Route und ruft für jeden passenden Request die dort definierte, anonyme Funktion auf. Um dies um eine zusätzliche Kontaktseite zu erweitern, können wir eine neue Route hinzufügen, wie gezeigt:

```
/* GET home page. */
router.get('/', function(req, res, next) {
  res.render('index', { title: 'Express' });
});

router.get('/contact', function(req, res, next) {
  res.render('contact', { title: 'Contact', name: 'Bernd' });
});
```

14 Routing-Methoden

Dieser Code wird als Modul exportiert und dann in app.js geladen. Die dortige Variable „routes“ enthält dann unseren Router. „res.render“ gibt die fertige Seite zurück, die an den Client geschickt wird. Der erste Parameter („index“) bezeichnet das zu benutzende Template aus dem Ordner „/views“, der zweite ist ein Objekt, dessen Schlüssel dem Template zugänglich gemacht werden.

15 Ein View erstellen

Damit die Route auch gerendert wird, erzeugen wir ein neues View: „views/contact.jade“. Dann können wir unseren Inhalt hinzufügen, indem wir das Markup aus „views/index.jade“ kopieren, dabei aber unsere neue Variable „name“ einfügen, sodass deren Inhalt nun auch angezeigt wird.

Nachdem der Prozess auf der Kommandozeile mit Strg+C gestoppt und mit dem Befehl „node app.js“ – hoffentlich fehlerfrei – neugestartet wurde, befindet sich unsere neue Seite nun unter **http://localhost:3000/contact**.

```
router.get('/contact/:name', function(req, res, next) {
  res.render('contact', { title: 'Contact', name: req.params.name });
});
```

16 Parametrisierte Routen

Es ist auch möglich, auf Teile spezieller Routen wie etwa „/contact/:name“ über „req.params.name“ zuzugreifen. Diese können sogar Reguläre Ausdrücke enthalten, um eine begrenzte Menge gültiger Routen zu beschreiben, was in komplexeren Situationen nützlich sein kann.

> Ihr Raspberry Pi sollte nun Ihre eigene Website ausliefern, mit mehreren Routen und einer eigenen Fehler-404-Seite.

17 Port ändern

Unsere Beispielanwendung läuft auf Port 3000, Sie können sie aber an jeden unbenutzten Port Ihres Betriebssystems binden. Bedenken Sie aber, dass der Server für Ports unter 1023 unter dem Benutzer root laufen muss.

18 Andere Module

Module können verwendet werden, um zusätzliche Funktionalität zu installieren, und fungieren als Middleware in den verschiedenen Stufen, die jeder HTTP-Request durchläuft. Um Module zu installieren, verwenden Sie das Kommando „npm install“ des Node Package Managers (NPM). Die Pakete werden dann in einen Unterordner namens „node_modules“ installiert.

```
"dependencies": {
  "body-parser": "~1.12.0",
  "cookie-parser": "~1.3.4",
  "debug": "~2.1.1",
  "express": "~4.12.0",
  "jade": "~1.9.2",
  "morgan": "~1.5.1",
  "serve-favicon": "~2.2.0"
}
```

19 Modulversionen festlegen

Module unterliegen ständigen Updates, was manchmal zu Inkompatibilität führt. Dies kann vermieden werden, indem Modulname und Versionsnummer in die „package.json“-Datei eingetragen werden. Der NPM tut dies automatisch für Sie, wenn Sie bei der Installation mittels „npm install <package>“ das Flag „--save“ verwenden.

20 Module finden

Bei über 9.000 Modulen in der NPM-Registry kann die Suche nach dem Benötigten schnell ausarten. Es gibt hauptsächlich drei Möglichkeiten: das Kommandozeilen-Tool „npm find <suchwort>“, die NPM-Website (**npmjs.com**) und Ihre bevorzugte Suchmaschine.

```
pi@raspberrypi ~/MicroSite $ npm install compression -
compression@1.4.1 node_modules/compression
├── on-headers@1.0.0
├── bytes@1.0.0
├── vary@1.0.0
├── compressible@2.0.2 (mime-db@1.7.0)
└── accepts@1.2.4 (negotiator@0.5.1, mime-types@2.0.9)
pi@raspberrypi ~/MicroSite $
```

21 Module als Middleware

Express selbst bringt kaum Features mit – diese werden nach Bedarf durch in Modulen bereitgestellte Middleware hinzugefügt. Um zum Beispiel von Ihrer Anwendung ausgelieferte Dateien und Seiten mit gzip zu komprimieren, können Sie das Modul „compression“ verwenden. Installieren Sie es mit „npm install compression - -save“.

```
//app.use(favicon(__dirnam
app.use(logger('dev'));
app.use(bodyParser.json())
app.use(bodyParser.urlenco
app.use(cookieParser());
app.use(express.static(pat
app.use(compression());
```

22 Middleware implementieren

Middleware wird mit der Methode „app.use“ übergeben. Je nach Art der Middleware kann die Reihenfolge, in der sie übergeben wird, eine Rolle spielen. In diesem Fall ersetzt das Modul die bereits existierende Funktion „express.static“. Denken Sie daran, das Modul auch zu laden: „var compression = require(‚compression‘);“.

```
/* GET home page. */
router.get('/', function(req,
  res.render('index', { title:
});

router.get('/contact/:name', f
  res.render('contact', { titl
});

router.get('*', function(req,
  res.render('e404');
});
```

23 Eigene Fehlerseiten

Beim Aufruf einer Route, die es nicht gibt, wird eine rudimentäre 404-Seite angezeigt. Für größere Websites ist es wünschenswert, eine eigene 404-Seite zu haben. Das implementieren Sie, indem Sie am Ende alle verbliebenen Routen mit der Wildcard „*“ abfangen. Außerdem müssen Sie natürlich noch das View erstellen. Beachten Sie, dass die „*“-Route die allerletzte sein muss, die definiert wird – in unserem Beispiel kann die Route „/users“ nie mehr erreicht werden!

24 Startklar

Ihr Raspberry Pi sollte nun Ihre eigene Website ausliefern, mit mehreren Routen und einer eigenen Fehler-404-Seite. Und das alles auf einem 35 € günstigen, kreditkartengroßen Computer!

Pimpen Sie Ihren Pi

Diese Gadgets helfen Ihnen, das Maximum aus Ihrem Pi herauszuholen.

Jetzt haben Sie also endlich einen Raspberry Pi und können es gar nicht erwarten, loszulegen. Es gibt allerdings ein Problem – in der Grundausstattung bekommen Sie lediglich die Platine, mehr nicht! Die Folge: Mittlerweile ist ein riesiger Zubehörmarkt für Kabel, Cases und andere Accessoires entstanden.

Pimoroni Pibow
(auch in anderen Farben erhältlich)

pimoroni.com
Preis € 15
Die farbenfrohen Universalgehäuse sind zweifelsohne die Standard-Cases schlechthin. Es besteht aus dünnen Plastikscheiben, die Sie um Ihren Pi herumbauen. Die Platine sitz sicher verschraubt. Das einzigartige Design setzt Ihren Pi toll in Szene.

Pimoroni Stealth Black Case
(auch in anderen Farben erhältlich)

thepihut.com
Preis € 12
Hierbei handelt es sich um das Gehäuse mit dem besten Preis-Leistungs-Verhältnis. Es lässt sich leicht auseinandernehmen, um Pis auszutauschen und an Ports zu gelangen. Die I/O-Ports passen sich genau dem Design an und es gibt Ventilationsöffnungen. Die Statusleuchten sind durchs Gehäuse sichtbar.

USB-Noodle-Kabel und Netzteile

pimoroni.com
Preis € 5 und € 10
Eine schlechte Stromversorgung ist der Hauptgrund für Systeminstabilität, daher sollten Sie den Pi über ein hochwertiges Netzteil mit Strom versorgen. Pimoroni hat außerdem eine riesige Auswahl an Kabeln in verschiedensten Farben im Programm.

SB Components Green Raspberry Pi Case
(auch in anderen Farben erhältlich)

sbcshop.co.uk
Preis € 6
Dieses Gehäuse von SB Components ist extrem Bastler-freundlich. Nicht nur, dass es Ihrem Pi ein lauschiges Plätzchen bietet, Sie haben auch Zugang zu allen Erweiterungs-Anschlüsse. Alle anderen Ports lassen sich dank praktischer Labels leicht voneinander unterscheiden. Beinahe schon genial sind die Lichtröhrchen für die Statusleuchten.

Pimoroni Model A Case

pimoroni.com
Preis € 8
Ein elegantes, maßgeschneidertes Gehäuse für ein Modell-A-Raspberry-Pi, das schlanker als reguläre Pibows ist und besseren Zugang zu den Anschlüssen ermöglicht.

Direkter Zugriff

Sie müssen an Ihrem Raspberry Pi etwas machen? Am schnellsten und einfachsten geht das mit einem Computer und einem UART-Kabel, das den Raspberry Pi sowohl mit Strom versorgt, als auch über den Terminal Zugang zum Distro erlaubt. Dieses UART-Kabel bekommen Sie bei Amazon für 6 Euro. Es findet direkt über die GPIO-Ports Ihres Pis Anschluss. Mit ihm können Sie übrigens sogar Ihre Netzwerk- und Internetverbindung mit dem Gerät teilen.

HDMI-Noodle-Kabel

(auch in anderen Farben erhältlich)

pimoroni.com
Preis € 9

Mit den günstigen HDMI-1.4-Kabeln nutzen Sie die Videoqualitäten Ihres Pis am besten aus.

Pi Series Raspberry Pi Enclosure Black

(Auch in metallic-blau erhältlich)

polycase.com
Preis € 12

Ein einzigartiges Case, das Ihren Raspberry Pi in eine dezente Media-Box für Ihren Fernseher oder in einen Desktop-Mini-Computer verwandelt.

Kühlkörper

thepihut.com
Preis € 5

Wärme abzuleiten funktioniert mit diesen maßgeschneiderten Kühlkörpern ganz fantastisch. Gerade beim Übertakten sorgen sie dafür, dass Ihr Pi cool bleibt.

Proto Armour Case

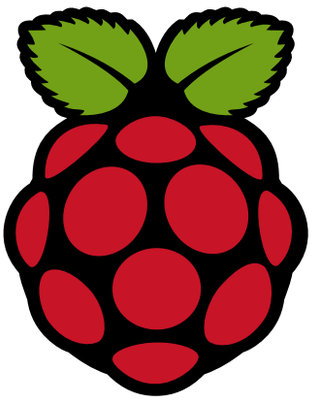
mobileappsystems.com
Preis € 32

Ein robustes Premium-Gehäuse, das aussieht, als ob es einer Explosion standhalten könnte. Zur Ausstattung gehören Kühlkörper, um Ihren Pi vorm Überhitzen zu bewahren.

Pibow VESA Mount

pimoroni.com
Preis € 5

Der Pibow sieht nicht nur klasse aus, sein Stufendesign macht ihn auch äußerst flexibel. Mit einem VESA Mount können Sie ihn praktisch unsichtbar auf der Rückseite eines Fernsehers anbringen.



Eine Fernsteuerung für den Pi

Steuern Sie den Raspberry Pi über das Internet von Ihrem Smartphone, Tablet oder PC aus.

Im Rahmen dieses Projekts erstellen wir eine einfache Internet-Benutzerschnittstelle, die Sie von überall aus erreichen können.

Sie können (und sollten) den Zugang durch eine Sicherheitsabfrage schützen.

Mit dem Raspberry Pi kann man auch im Alltag eine Menge praktische Sachen machen, etwa das Garagentor öffnen, die Beleuchtung und Heizung in der Wohnung regulieren oder Pflanzen bewässern. Für viele solcher Anwendungen ist eine Fernsteuerung des Raspberry Pi vonnöten, und genau mit diesem Thema wollen wir uns hier im Tutorial näher beschäftigen.

Bei diesem Projekt verwenden wir nicht das übliche Raspbian, sondern Arch Linux als Betriebssystem, da Letzteres mit recht wenig Ressourcen auskommt und überdies keine Desktop-Umgebung benötigt.

Wir gehen für dieses Tutorial davon aus, dass Sie Arch Linux auf eine SD-Karte gespielt haben (unter **linuxuser.co.uk/tutorials/how-to-set-up-raspberry-pi/** finden Sie eine Anleitung dafür, wobei dort allerdings das Prozedere Debian beschrieben ist, was Sie aber für Arch Linux nur entsprechend anzupassen brauchen). Alles Weitere in der nachfolgenden Schritt-für-Schritt-Anleitung.

Für die Einrichtung Ihres Fernzugriffs benötigen Sie folgende Software:

- **Betriebssystem Arch Linux**
- **Webserver Apache**
- **Python** (als Skriptsprache)
- **sudo** (um ggf. im CGI-Skript die Benutzerrechte auszuweiten)
- **dynamischer DNS-Daemon noip** – (Software, die im Hintergrund läuft und einen Domainnamen an die IP-Adresse des Routers weiterleitet, sodass der Raspberry Pi von überall mit einer leicht zu merkenden URL angesteuert werden kann)

Was Sie brauchen:

- einen Raspberry Pi
- das Betriebssystem Arch-Linux
  archlinux.org/download
- einen zweiten Computer
  (für SSH und Tests)

01 Arch Linux hochfahren

Verkabeln Sie den Raspberry Pi und warten Sie auf den Login-Prompt von Arch Linux. Loggen Sie sich mit dem Standard-Usernamen („root“) und dem Standard-Passwort (ebenfalls „root“) ein. Später werden wir das Passwort ändern.

02 System-Update durchführen

Arch Linux wird nicht nach Versionsnummern, sondern als fortlaufend aktualisierte Software veröffentlicht. Der Paketmanager von Arch Linux heißt pacman. Mit dem Befehl `pacman -Syu` können Sie ein vollständiges System-Update durchführen. Planen Sie dafür allerdings etwas Zeit ein: Erstens sind die Arch-Linux-ARM-Server leider häufig überlastet, zweitens müssen eine Menge Pakete übertragen werden, und drittens sitzt das Betriebssystem beim Raspberry Pi ja auf einer SD-Karte.

03 Benötigte Pakete installieren

Im Kasten auf Seite 142 unten links ist weitere für das Projekt notwendige Software aufgelistet. Diese holen Sie sich mit `pacman -S noip apache python2 sudo`. Beantworten Sie alle Nachfragen mit „y“.

04 Netzwerk-Informationen

Wir legen Ihnen sehr ans Herz, dass Sie Ihrem als Server fungierenden Raspberry Pi eine statische IP-Adresse zuweisen, statt dies Ihrem Router zu überlassen. So wissen Sie nämlich immer genau, wo in Ihrem Netzwerk Sie den Pi finden, was sinnvoll ist, wenn Sie von einem anderen Gerät auf ihn zugreifen möchten. Außerdem brauchen Sie eine statische IP-Adresse, wenn Sie den Pi via Internet ansteuern möchten. Um eine solche Adresse festlegen zu können, müssen wir zunächst einige Informationen über Ihr aktuelles Netzwerk ermitteln. Tun Sie dies mit den Kommandos `ip addr show dev eth0 | grep inet` und `ip route show | grep default`. Das grep filtert unnötige Angaben aus, indem nur Zeilen angezeigt werden, die „inet“ beziehungsweise „default“ enthalten. So bekommen wir die IP-Adresse des Pi sowie das Standard-Gateway heraus.

```
[root@alarmpi ~]# ip addr show dev eth0 | grep inet
    inet 172.17.173.254/24 brd 172.17.173.255 scope global eth0
    inet6 fe80::ba27:ebff:fef3:9016/64 scope link
[root@alarmpi ~]# ip route show | grep default
default via 172.17.173.1 dev eth0  metric 204
```

> **Sie brauchen eine statische IP, um den Pi via Internet anzusteuern.**

05 Statische IP-Adresse zuweisen

Auf Basis dieser Informationen können wir nun die statische IP-Adresse zuweisen. Hierfür erstellen wir einen neuen systemd-Service. Legen Sie eine neue Datei namens **/etc/systemd/system/static-network.service** an und füllen Sie sie mit folgendem Inhalt (ersetzen Sie dabei die IP-Adresse und das Standard-Gateway mit Ihren eigenen Werten):

```
[Unit]
Description=Static IP Connectivity
Wants=network.target
Before=network.target
BindsTo=sys-subsystem-net-devices-eth0.device
After=sys-subsystem-net-devices-eth0.device

[Service]
Type=oneshot
RemainAfterExit=yes
ExecStart=/sbin/ip link set dev eth0 up
ExecStart=/sbin/ip addr add 172.17.173.254/24 dev eth0
ExecStart=/sbin/ip route add default via 172.17.173.1

ExecStop=/sbin/ip addr flush dev eth0
ExecStop=/sbin/ip link set dev eth0 down

[Install]
WantedBy=multi-user.target
```

Speichern Sie die Datei, beenden Sie den Editor, und führen Sie dann die folgenden Kommandos durch, um DHCP zu deaktivieren sowie die Ethernet-Schnittstelle und die Bridge permanent mit einer statischen IP-Adresse zu aktivieren:

```
systemctl disable netctl-ifplugd@eth0.service
systemctl enable static-network.service
```

Booten Sie den Raspberry Pi neu, um die Änderungen wirksam werden zu lassen. Nach dem Neustart wird die **/etc/resolv.conf** sich verändert haben, denn sie war zuvor durch DHCP konfiguriert, und dieses haben wir ja abgeschaltet. Geben Sie daher den Befehl `echo nameserver 172.17.173.1 > /etc/resolv.conf` – damit können Sie DNS-Adressen wie etwa google.com in IP-Adressen auflösen.

06 Über SSH einloggen

Öffnen Sie nach erneutem Reboot auf Ihrem Linux-PC ein Terminal und tippen Sie `ssh root@[IP-Adresse des Pi]`. Bestätigen Sie, dass Sie eine Verbindung herstellen möchten, und geben Sie dann das Passwort ein, das nach wie vor „root“ lautet. Daraufhin sind Sie über SSH auf dem Pi eingeloggt.

07 Root-Passwort ändern

Da Sie vermutlich ein SSH-Login über das Internet zulassen werden, ist es äußerst ratsam, ein eigenes, sicheres Passwort zu vergeben. Tippen Sie `passwd` und folgen Sie dann den Anweisungen auf dem Bildschirm. Ihre SSH-Session bleibt bestehen, aber bei der nächsten Anmeldung müssen Sie das neue Passwort verwenden. Vielleicht möchten Sie außerdem noch den Inhalt von **/etc/hostname** ändern, indem Sie den Hostnamen etwas aussagekräftiger gestalten. Standardmäßig ist dort „alarmpi“ eingetragen. Sie könnten etwa „remotepi“ daraus machen. Die Änderung wird erst nach einem Neustart wirksam.

08 Apache konfigurieren

Apache gehört weltweit zu den beliebtesten Webservern. Man könnte einwenden, dass ein etwas weniger speicherintensiver Webserver für den Raspberry Pi besser geeignet sein könnte, doch ist Apache eine bewährte Ressource, und auf einem Fernsteuerungssystem wie diesem werden ja nur wenige Benutzer »

sich einloggen. Der Webhoster Mythic Beasts hostete mit Apache einen Spiegelserver auf einem Raspberry Pi. Erst nach über sieben Monaten versagte die SD-Karte ihren Dienst. Bis dahin waren 1,5 TB Datenvolumen über den Pi versendet worden.

Das Common Gateway Interface (CGI) ist eine Standardmethode der Delegierung von Webinhalten von einem Webserver an ausführbare Dateien. In unserem Fall handelt es sich bei der ausführbaren Datei um ein Python-Skript.

Wir müssen das File **/etc/httpd/conf/httpd.conf** so bearbeiten, dass die Ausführung von CGI-Skripten ermöglicht wird und jede Datei mit der Endung .cgi als CGI-Skript behandelt wird. Das File ist recht lang. Etwa bei Zeile 200 finden Sie einen Abschnitt, der mit „<Directory „/srv/http“>“ beginnt. Ergänzen Sie dort die Zeile

```
Options Indexes FollowSymLinks
```

mit

```
Options Indexes FollowSymLinks
ExecCGI
```

und fügen Sie direkt darunter diese Zeile neu hinzu:

```
AddHandler cgi-script .cgi
```

Etwas weiter unten finden Sie die Zeile:

```
DirectoryIndex index.html
```

Ersetzen Sie sie durch:

```
DirectoryIndex index.html index.cgi
```

Dadurch wird automatisch das Skript **index.cgi** ausgeführt, das wir in Kürze erstellen werden, statt eine Liste von Dateien in dem Verzeichnis auszugeben.

```
<IfModule dir_module>
DirectoryIndex index.html index.
cgi
</IfModule>
```

09 Apache starten

Legen Sie mit folgendem Kommando fest, dass Apache während des Bootvorgangs gestartet werden soll:

```
systemctl enable httpd
```

Starten Sie es danach direkt:

```
systemctl start httpd
```

Wenn Sie nun zur Adresse http://[IP-Adresse des Pi] gehen, sehen Sie eine Seite, auf der alle im Verzeichnis **/srv/http** enthaltenen Dateien anzeigt. Das Verzeichnis ist allerdings noch leer, daher ist auch die Liste leer.

10 Erstes CGI-Skript als Test

Wechseln Sie mit dem Befehl `cd /srv/http` in das Verzeichnis **/srv/http** und erstellen Sie mit `touch index.cgi` eine neue Datei namens **index.cgi**. Um diese ausführbar zu machen, tippen Sie `chmod +x index.cgi`. Öffnen Sie die Datei in einem Editor. In der obersten Zeile (mit dem „!#“) wird Apache signalisiert, dass das Skript mit Python interpretiert werden soll. Die erste „print“-Zeile weist Apache darauf hin, dass nun HTML-Inhalt folgt, die zweite erzeugt eine leere Zeile, und die nachfolgenden geben eine Standard-„Hallo Welt!“-HTML-Seite aus. Die drei „“ sorgen auf bequeme Weise für die Ausgabe mehrerer Zeilen.

```
#!/usr/bin/env python2
# Dem Webserver HTML-Inhalt
ankündigen
print “Content-Type: text/html”
print

# Hallo Welt! ausgeben
print “””
<html>
<head>
<title>Hallo Welt!</title>
</head>
<body>
<h1>Hallo Welt!</h1>
</body>
</html>
“””
```

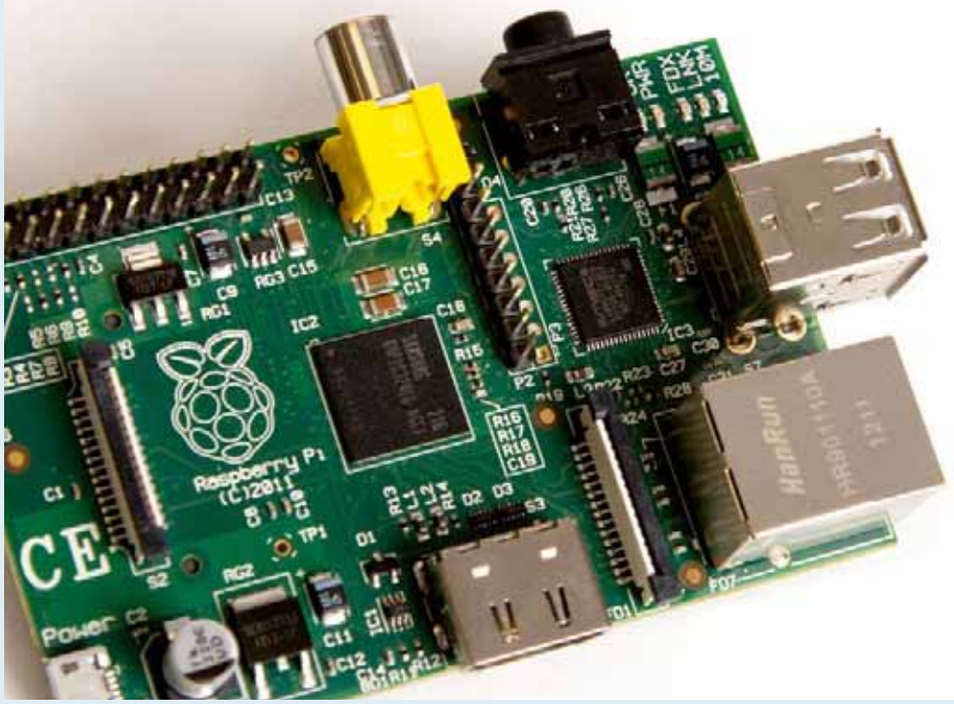
Aktualisieren Sie nun die leere Index-Seite von vorhin, und es sollte „Hallo Welt!“ dort zu lesen sein.

11 Menü erstellen

Um einige Optionsmöglichkeiten zu erhalten, kreieren wir ein paar Links. Als Beispiele nehmen wir einen Link zu einem Systembefehl mit Ausgabe sowie einen zum Öffnen und Schließen des Garagentors. Mit der folgenden Syntax werden Parameter an CGI-Skripte übertragen:

```
script_name.
cgi?var1=value1&var2=value2
```

Wir haben hier einen Dispatcher-Code geschrieben sowie ein Menü, das dafür sorgt, dass der



“Um Optionen zu erhalten, kreieren wir ein paar Links.”

Code ausgeführt wird. Mithilfe einer Funktion wird der CGI-Header ausgegeben und dann die HTML-Seite mit dem von Ihnen festgelegten Body und Title, um doppelten Code zu vermeiden.

Zu diesem Zeitpunkt werden Sie noch einen 500-Fehler vom Webserver angezeigt bekommen, wenn Sie auf einen der Links klicken, denn es sind noch keine Befehle damit verknüpft.

```
#!/usr/bin/env python2

# A CGI dispatcher written in Python by Liam Fraser for a Linux User and
# Developer tutorial.

import cgi

def print_page(title, body):
    # Dem Webserver HTML-Inhalt ankündigen
    print "Content-Type: text/html"
    print

    print """
<html>
<head>
<title>{0}</title>
</head>
<body>
<h1>{0}</h1>
{1}
</body>
</html>
""".format(title, body)

def print_menu():

    # Menü ausgeben
    # Print a menu
    title = "Liam's Raspberry Pi"
    body = """
<p><a href="index.cgi?action=run_command">Run a command</a></p>
<p><a href="index.cgi?action=garage_control">Control the Garage</a></p>
"""
    print_page(title, body)

#
# Start of script
#
if __name__ == "__main__":

    # Parameter holen
    params = cgi.FieldStorage()

    # Variable zur Überprüfung gültigen/ungültigen Inputs
    valid = False

    # Ist ein action in den Parametern enthalten?
    if params.getvalue("action"):
        action = params.getvalue("action")

        if action == "run_command":
            valid = True

        elif action == "garage_control":
            valid = True

    if valid == False:
        print_menu()
```

12 Debugging (Fehlerbeseitigung)

Bei einem unerwarteten 500-Fehler sollte man zunächst überprüfen, ob tatsächlich ein Header ausgegeben wurde. Ansonsten könnte auch ein Syntaxfehler im Python-Code vorliegen, allerdings ist Apache hier von sich aus nicht besonders hilfreich bei der Fehlersuche. Sie können Folgendes tun: Schließen Sie den Editor und führen Sie den Code – mit `./index.cgi` – im Terminal aus, um zu sehen, ob Python mit Syntaxfehlern abbricht. Oder konsultieren Sie das Fehler-Log von Apache. Mit `tail -f /var/log/httpd/error_log` wird Ihnen das gesamte Fehler-Log angezeigt, mit `tail -n 50 /var/log/httpd/error_log` nur die letzten 50 Zeilen (Sie können natürlich auch einen anderen Wert eingeben als 50.

13 Kommando ausführen

Die untenstehende Funktion erzeugt eine Seite, auf der Sie ein Kommando geben können, und stellt dann den zugehörigen Output dar. Wenn Sie sich das HTML-Formular unten ansehen, sehen Sie, dass Name und Wert an die **index.cgi** geschickt werden, wenn der Submit-Button geklickt wird. Möchten wir also beispielsweise den Output von ps haben, sähe die Anfrage so aus:

```
index.cgi?action=run_command&cmd=ps
```

Abgesehen davon ist der Code ziemlich selbsterklärend. Die letzten beiden Dinge, die Sie noch tun müssen, damit alles funktioniert, sind das Hinzufügen der import-Zeile

```
import subprocess
```

sowie das Aufrufen der Funktion aus dem Dispatcher-Teil des Skripts, wobei die Parameter weitergegeben werden müssen:

```
        if action == "run_command":
            valid = True
            run_command(params)
```

Probieren Sie es aus!

```
def run_command(params):
    # Durchführung von Aufgaben im Bereich run_command

    if params.getvalue("cmd"):
        # We have a command to run
        cmd = params.getvalue("cmd")

        # subprocess.check_output benötigt Array von
        # Parametern in Liste, mit Leerzeichen getrennt
        cmd_list = cmd.split()

        try:
            output = subprocess.check_output(cmd_list)
        except:
            output = "Error running command."

        title = "Output of {0}".format(cmd)
        body = "<pre>{0}</pre>".format(output)

        print_page(title, body)

    else:
        # Seite für Kommandoeingabe anzeigen
        title = "Run Command"
        body = """
<form action="index.cgi" method="post">
Enter a command to run: <input type="text" name="cmd">
<input type="hidden" name="action" value="run_command">
<input type="submit" value="Submit">
</form>
"""

        print_page(title, body)
```

14 Benutzerrechte

Apache läuft als HTTP-User, das heißt, es hat nur sehr eingeschränkte Benutzerrechte. Das ist insofern problematisch, als vermutlich Root-Berechtigungen notwendig sind, wenn bei einem Projekt der Wohnungsautomatisierung auf die diversen Datenschnittstellen des Raspberry Pi zugegriffen werden soll. Um das Problem zu lösen, benutzen wir sudo. Wechseln Sie ins Verzeichnis **/etc/sudoers.d**, generieren Sie eine neue Datei namens **http**, und öffnen Sie sie im Editor.

Offensichtlich fehlschlagen würde beispielsweise der reboot-Befehl. Um Abhilfe zu schaffen, lokalisieren Sie zunächst mit whereis reboot das Skript, dann fügen Sie folgende Zeile hinzu, um es für den HTTP-User mit sudo ausführbar zu machen:

```
http ALL=(ALL) NOPASSWD: /usr/sbin/
ip,/sbin/reboot
```

Wie Sie dabei ebenfalls sehen, können mehrere – durch Komma getrennte – Kommandos in einem Rutsch gegeben werden. Wenn Sie jetzt sudo reboot auf der Befehlsseite ausführen, startet der Pi erneut.

15 Garagentoröffner als Beispiel

Der nachfolgende Code soll als Beispiel dienen. Er ist zum Einsatz bei der Steuerung eines Garagentors gedacht. Das Programm ist recht einfach gestrickt und ähnelt dem weiter oben gesehenen, das die Kommandoausführung steuert. Wie dort müssen wir auch hier die Funktion am Ende des Skripts aufrufen, indem wir die Zeile

```
garage_control(params)
```

zum Bereich der if-Anweisung garage_control hinzufügen.

```
def garage_control(params):
    # output hier oben definieren,
damit innerhalb der Funktion
    output = “”

    if params.getvalue(“garage_
action”):
        action = params.
getvalue(“garage_action”)
        if action == “Open Garage”:
            cmd = “sudo /script/to/
open_garage”

        elif action == “Close
Garage”:
            cmd = “sudo /script/to/
close_garage”

        if cmd:

            # Befehl ausführen
            cmd_list = cmd.split()

            try:
                output = subprocess.
check_output(cmd_list)
            except:
                output = “Error
running command.”

    title = “Garage Control”
    # Body mit Output des
Befehls generieren, sofern
    # einer gegeben wurde
    body = “””
<form action=”index.cgi”
method=”post”>
<input type=”hidden” name=”action”
value=”garage_control”>
<pre>{0}</pre>
<input type=”submit” name=garage_
action value=”Open Garage”>
<input type=”submit” name=garage_
action value=”Close Garage”>
</form>
“””.format(output)

    print_page(title, body)
```

16 Authentifizierung

Wenn Sie etwas dem Internet aussetzen, sind Sicherungsmaßnahmen sinnvoll. Verschlüsselte HTTP-Verbindungen würden den Rahmen dieses Artikels sprengen, doch zumindest eine Authentifizierung können Sie relativ einfach einrichten. Erzeugen Sie dazu zunächst eine Authentifizierungsdatei mithilfe des Werkzeugs htpasswd:

```
[root@remotepi httpd]# htdigest -c /
etc/httpd/auth secure liam
Adding password for liam in realm
secure.
New password:
Re-type new password:
```

Entfernen Sie für alle weiteren Benutzer den Parameter -c. Nun müssen wir erneut die Apache-Konfigurationsdatei bearbeiten. Geben Sie direkt unter der Zeile mit „AddHandler“, die wir in Schritt 08 dem Bereich „<Directory "/srv/http">“ hinzugefügt haben, das Folgende ein:

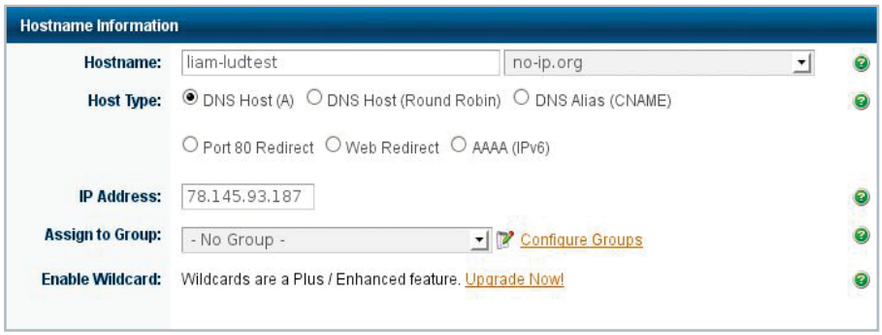
```
# Authentication
AuthType Digest
AuthName “secure”
AuthDigestDomain /
AuthDigestProvider file
AuthUserFile /etc/httpd/auth
Require valid-user
```

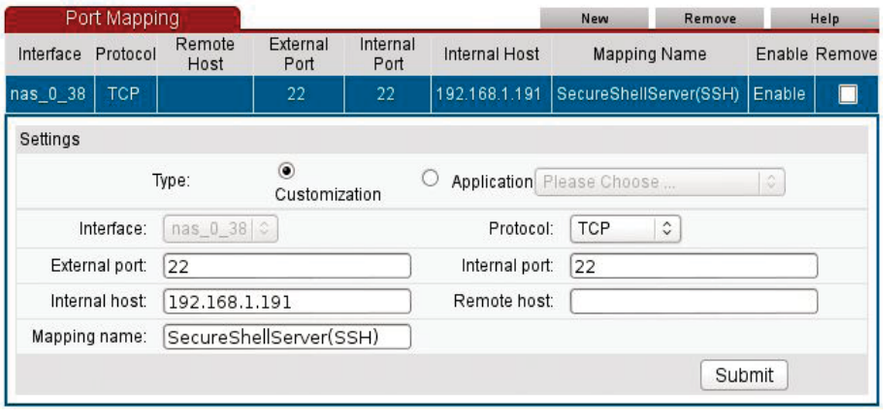
Führen Sie anschließend mit systemctl restart httpd einen Neustart von Apache durch und rufen Sie die Webseite auf. Dort müssten nun Username und Passwort abgefragt werden, ohne die kein Zugriff gewährt wird.

“Wenn Sie etwas dem Internet aussetzen, ist eine Sicherung sinnvoll.”

17 Dynamisches DNS einrichten

Gehen Sie auf die Website **no-ip.com** und legen Sie dort ein Gratiskonto für private Zwecke an. Sie brauchen jedoch nicht den Client von No-IP herunterzuladen, sondern können unseren benutzen. Klicken Sie in der E-Mail, die Sie nach der Registrierung bekommen haben, auf den Aktivierungslink. Nun können Sie sich in Ihr Benutzerkonto einloggen. Wählen Sie dort die Option „Add a host“, legen Sie einen Hostnamen fest, und suchen Sie sich eine der angebotenen Domains aus der Dropdown-Liste aus. Stellen Sie „DNS host“ als Hosttyp ein, und klicken Sie auf „Create host“. Wir haben „liam-ludtest“ mit der Domain no-ip.org ausgewählt, somit läuft der Zugang über **liam-ludtest.no-ip.org**.





18 No-IP konfigurieren

Tippen Sie das Kommando `noip2 -C -Y`, um sich durch die interaktive Konfigurationsroutine des No-IP-Clients führen zu lassen. Wir haben das Update-Intervall bei den vorgegebenen 30 Minuten belassen. Das bedeutet, dass der Client alle 30 Minuten nach einer möglichen Änderung der IP-Adresse Ausschau hält.

Wenn Sie fertig sind, starten Sie den Daemon mit den Befehlen `systemctl enable noip2` und `systemctl start noip2`. Nach wenigen Minuten wird Ihre IP-Adresse über Ihren No-IP-Hostnamen erreichbar sein. Wenn Sie dies allerdings von innerhalb Ihres Heimnetzwerks versuchen, werden Sie vermutlich lediglich auf der Homepage Ihres Routers landen.

19 NAT-Portweiterleitung

Vermutlich hängen mehrere Geräte hinter Ihrem Router, die alle dieselbe externe IP-Adresse haben. Dies liegt einerseits an der Knappheit von IPv4-Adressen, andererseits an Schutzaspekten, da es sicherer ist, das Internet von Ihrem internen Heimnetzwerk zu separieren. Beim NAT-Verfahren (Network Address Translation) wird ein Port von der externen IP-Adresse des Routers an einen Rechner im LAN weitergeleitet. In unserem Fall möchten wir jeglichen Traffic für den TCP-Port 22, der an der externen IP-Adresse des Routers ankommt, an die IP-Adresse des Raspberry Pi weiterleiten. Der TCP-Port 22 ist der Port, der für SSH genutzt wird. Zusätzlich sollten wir den HTTP-Traffic entsprechend weiterleiten, dafür ist der TCP-Port 80 zuständig.

Die konkrete Konfiguration der Portweiterleitung hängt stark von Ihrem Routermodell ab, Sie müssen daher vermutlich in dessen Anleitung nachsehen. Das entsprechende Menü finden Sie wahrscheinlich in den erweiterten Einstellungen Ihres Geräts. Sie sollten Ihren Router ansprechen können, indem Sie Ihren No-IP-Hostnamen in die Adresszeile des Browsers eingeben, andernfalls mit der Adresse des Standard-Gateways (siehe Schritt 04).

In der Abbildung oben sehen Sie ein Beispiel, wie das fragliche Menü aussehen könnte.

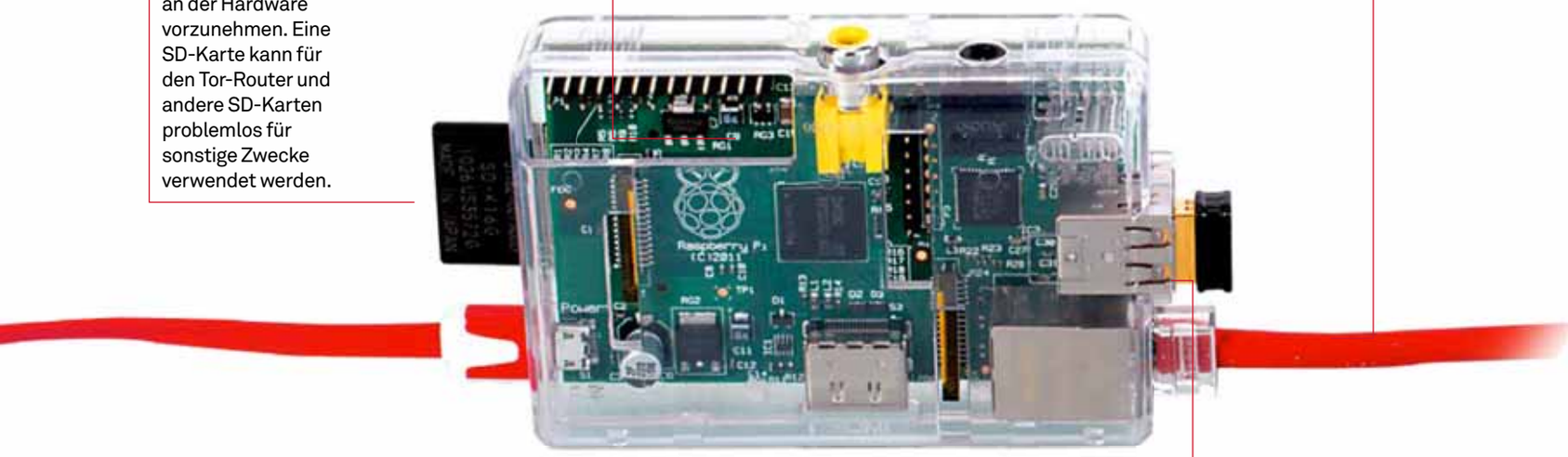
20 Testen

Damit sollten Sie es geschafft haben! Testen Sie aber Ihren No-IP-Hostnamen von außerhalb Ihres Heimnetzwerks, um sicherzugehen, dass er funktioniert.

Bei diesem Projekt brauchen Sie keine Änderungen an der Hardware vorzunehmen. Eine SD-Karte kann für den Tor-Router und andere SD-Karten problemlos für sonstige Zwecke verwendet werden.

Mit nur einem Raspberry Pi samt WLAN-Adapter als Wireless Access Point können Sie einen Tor-Zugang einrichten und so Ihre Anonymität wahren.

So ziemlich überall, wo eine Internetverbindung zur Verfügung steht, können Sie drahtlos surfen: bei Freunden, im Hotel und so weiter.



Dieser kleine Adapter macht es möglich: Verbinden Sie sich ohne Kabel mit dem Raspberry Pi.

Geschützt surfen mit einem Onion Pi

Machen Sie aus dem Raspberry Pi einen hochsicheren und sehr portablen Router, und schützen Sie so Ihre Privatsphäre im Netz.

Was Sie brauchen:

- einen Raspberry Pi
- Raspbian
  raspberrypi.org/downloads
- einen kompatiblen WLAN-Adapter
  adafruit.com/products/814

In dem Tutorial ab Seite 26 in diesem Heft haben wir gezeigt, wie Sie Ihren Raspberry Pi mithilfe eines WLAN-Adapters in einen mobilen, genügsamen Wireless Access Point für andere Geräte verwandeln können. Was aber, wenn Sie nicht einfach nur im Web stöbern möchten, sondern dabei auch rundum geschützt sein wollen? Dann gibt es auch dafür eine Lösung: Lassen Sie Ihren Router mit der Anonymisierungssoftware Tor laufen und schützen Sie so Ihre Privatsphäre.

Die Kombination aus Raspberry Pi, Raspbian und Tor, die von Adafruit „Onion Pi“ getauft wurde, dient hierbei ebenfalls als Wireless Access Point, aber obendrein mit der erwähnten Sicherheitsfunktionalität. Dabei ist die Einrichtung gar nicht schwer: Der vorhandene Wireless Access Point wird einfach mit der Installation der Tor-Software ergänzt, und dann sind noch ein paar Einstellungsarbeiten notwendig.

Auch wenn der Pi gerade nicht am Internet hängt, dient er dabei immer noch als Wireless Access Point für Ihr Heimnetzwerk.

01 Raspbian installieren

Raspbian ist die Raspberry-Pi-Distribution, die wir für den Onion Pi verwenden. Laden Sie das Zip-Archiv herunter, entpacken Sie das Image, und überspielen Sie es mit folgendem Befehl auf eine SD-Karte:

```
$ dd bs=4M if=[version number]-wheezy-raspbian.img of=/dev/[SD card location]
```

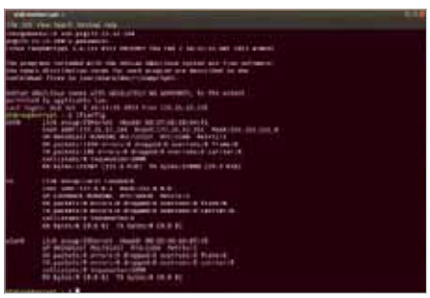
Sie können Raspbian auf Wunsch aber auch mit dem NOOBS-Tool installieren.

02 Raspbian einrichten

Gehen Sie die Setup-Routine des Raspbian-Betriebssystems durch, wobei Sie darauf achten müssen, einen SSH-Server aktiviert zu haben. Deaktivieren Sie außerdem die Autoboot-Funktion für den Desktop, denn diesen brauchen wir bei diesem Projekt nicht und er würde nur unnötig Ressourcen verbrauchen. Sie können weiterhin festlegen, dass der überzählige Speicherplatz der SD-Karte mitverwendet werden soll.

03 IP-Adresse des Pi

Da wir beim Setup per SSH auf den Raspberry Pi zugreifen werden, müssen wir seine IP-Adresse kennen. Geben Sie hierfür das Kommando `ifconfig` ins Terminal ein. Schreiben Sie sich die IP-Adresse auf und schalten Sie den Pi aus.



04 SSH-Verbindung

Schließen Sie den USB-Wireless-Adapter an den Raspberry Pi an und schalten Sie den Pi wieder ein. Öffnen Sie auf einem anderen Computer im selben Netzwerk ein Terminal und geben Sie dort Folgendes ein:

```
$ ssh [user]@[IP address]
```

Wenn Sie aufgefordert werden, für Raspbian Ihr Passwort einzugeben, tun Sie dies.

05 DHCP installieren

Um es anderen Systemen leichter zu machen, sich mit dem Wireless Access Point des Raspberry Pi zu verbinden, installieren wir einen DHCP-Server auf ihm. Das geht so:

```
$ sudo apt-get install hostapd isc-dhcp-server
```

DHCP ordnet den Geräten im Netzwerk automatisch IP-Adressen zu. Sie benötigen also keine statischen IP-Adressen.

06 DHCP einrichten

Jetzt müssen wir den DHCP-Server noch konfigurieren. Öffnen Sie wie folgt die Konfigurationsdatei:

```
$ sudo nano /etc/dhcp/dhcpd.conf
```

Kommentieren Sie die beiden Zeilen aus, die mit option domain-name beginnen, indem Sie jeweils ein # davorsetzen, und entfernen Sie anschließend das # vor authoritative sieben Zeilen weiter unten.

07 Server-Adresse

Fügen Sie folgenden Inhalt am Ende der Konfigurationsdatei ein:

```
subnet 192.168.42.0 netmask 255.255.255.0 {
range 192.168.42.10 192.168.42.50;
option broadcast-address 192.168.42.255;
option routers 192.168.42.1;
default-lease-time 600;
max-lease-time 7200;
option domain-name “local”;
option domain-name-servers 8.8.8.8, 8.8.4.4;
}
```

Speichern und schließen Sie die Konfigurationsdatei.

08 DHCP-Server

Bearbeiten Sie die Dateien für die Server-Konfiguration in der Weise, dass der Server mit dem Wireless-Adapter zusammenarbeitet:

```
$ sudo nano /etc/default/isc-dhcp-server
```

Scrollen Sie runter zu INTERFACES und ändern Sie den Eintrag in:

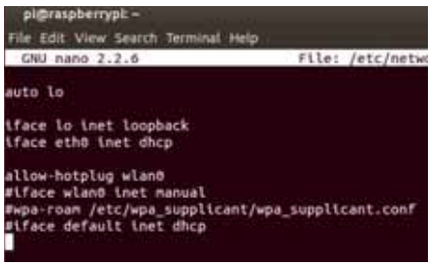
```
INTERFACES=”wlan0”
```

09 Ankommender WLAN-Verkehr

Wir müssen den WLAN-Adapter so einrichten, dass er sowohl statisch ist als auch hereinkommende Signale akzeptiert. Tippen Sie zunächst:

```
$ sudo nano /etc/network/interfaces
```

Setzen Sie ein # vor iface wlan0 und die nachfolgenden Zeilen mit wpa roam, iface default sowie alle anderen Zeilen mit Auswirkungen auf mit wlan0.



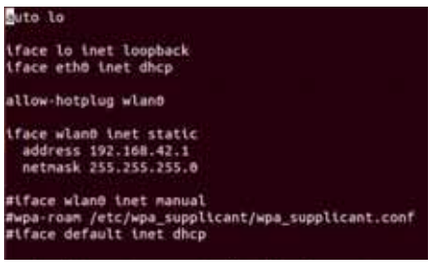
10 Statische IP-Adresse

Vergeben Sie nun eine statische IP-Adresse an die Wireless-Schnittstelle. Geben Sie dazu nach der Zeile allow-hotplug wlan0 das Folgende ein:

```
iface wlan0 inet static
        address 192.168.42.1
        netmask 255.255.255.0
```

Speichern und schließen Sie die Datei, und legen Sie dann die Adresse von wlan0 fest:

```
$ sudo ifconfig wlan0 192.168.42.1
```

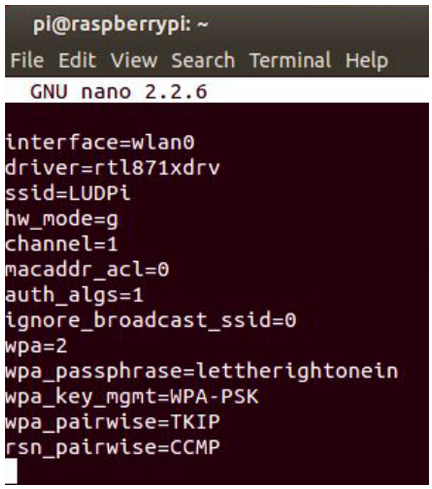


11 WLAN erstellen

Wir brauchen nun eine neue Datei, die sämtliche Informationen betreffend unser Wireless-Netzwerk enthält. Wir werden die Datei mit einem Passwortschutz ausstatten, sodass nur Befugte Zugriff haben. Erstellen Sie die Datei so:

```
$ sudo nano /etc/hostapd/hostapd.conf
```

Geben Sie anschließend den Text aus Schritt 12 in die Datei ein.



12 WLAN-Konfiguration

```
interface=wlan0
driver=rtl871xdrv
ssid=[access point name]
hw_mode=g
channel=1
macaddr_acl=0
auth_algs=1
ignore_broadcast_ssid=0
wpa=2
wpa_passphrase=[password]
wpa_key_mgmt=WPA-PSK
wpa_pairwise=TKIP
```

13 hostapd

Nach dem Speichern und Schließen der Datei müssen Sie das File hostapd so editieren, dass es auf die neue Datei zeigt. Öffnen Sie hostapd mit:

```
$ sudo nano /etc/default/hostapd
```

Suchen Sie die Zeile #DAEMON_CONF="", entfernen Sie das #, und ändern Sie die Zeile wie folgt:

```
DAEMON_CONF="/etc/hostapd/hostapd.conf"
```

14 Netzwerk-Adressierung

Das NAT-Verfahren (Network Address Translation) ermöglicht es, dass mehrere Clients sich mit dem Server verbinden. Aktivieren Sie das Verfahren, indem Sie zunächst diese Datei öffnen:

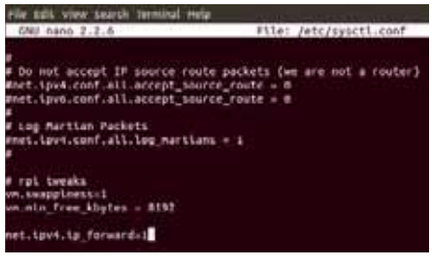
```
$ sudo nano /etc/sysctl.conf
```

Fügen Sie dann am Ende der Datei hinzu:

```
net.ipv4.ip_forward=1
```

Speichern Sie, und schließen Sie das Ganze mit diesem Befehl ab:

```
$ sudo sh -c "echo 1 > /proc/sys/net/ipv4/ip_forward"
```



15 IP-Tabellen

Führen Sie die drei folgenden Kommandos aus, um sicherzustellen, dass die Internetverbindung korrekt weitergeleitet wird:

```
sudo iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
sudo iptables -A FORWARD -i eth0 -o wlan0 -m state --state RELATED,ESTABLISHED -j ACCEPT
sudo iptables -A FORWARD -i wlan0 -o eth0 -j ACCEPT
```

16 Konfiguration anwenden

Damit das Ganze auch nach einem Neustart noch funktioniert, tippen Sie ferner ein:

```
$ sudo sh -c "iptables-save > /etc/iptables.ipv4.nat"
```

Fügen Sie dann am Ende der Datei /etc/network/interfaces das Folgende hinzu:

```
up iptables-restore < /etc/iptables.ipv4.nat
```

17 WLAN fertigstellen

Richten Sie zuletzt einen Daemon ein, damit der Dienst bereits beim Booten gestartet wird. Geben Sie hierfür ein:

```
sudo service hostapd start
sudo service isc-dhcp-server start
sudo update-rc.d hostapd enable
sudo update-rc.d isc-dhcp-server enable
```

Damit ist der Wireless Access Point eingerichtet.

18 Tor installieren

Nach einem Neustart können wir nun Tor installieren. Geben Sie ein:

```
$ sudo apt-get install tor
```

Nach der Installation müssen Sie aber noch die Tor-Konfigurationsdatei anpassen:

```
$ sudo nano /etc/tor/torrc
```

Im nächsten Schritt finden Sie die Informationen, die dort hinzugefügt werden müssen.

19 Tor konfigurieren

Geben Sie den nachfolgenden Inhalt unterhalb des FAQ-Kommentars ein:

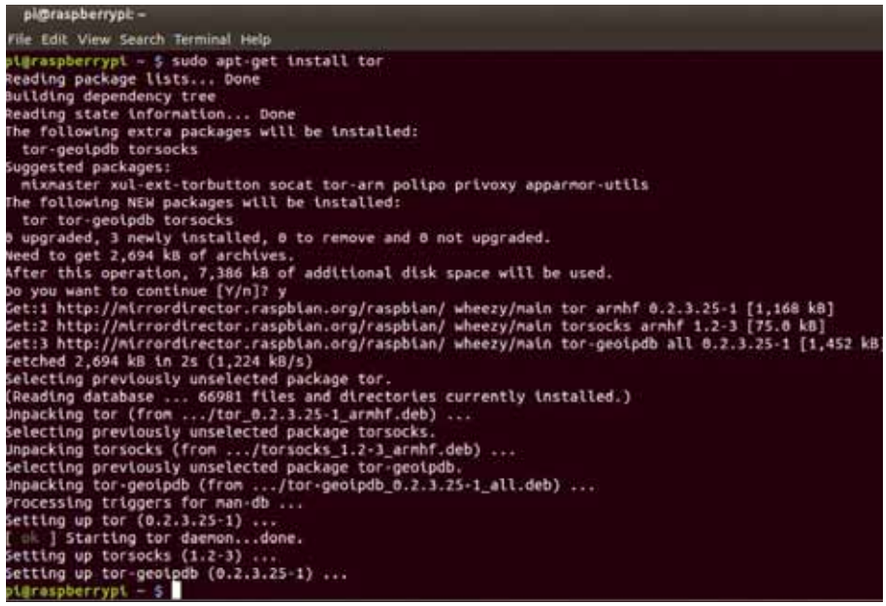
```
Log notice file /var/log/tor/notices.log
VirtualAddrNetwork 10.192.0.0/10
AutomapHostsSuffixes .onion,.exit
AutomapHostsOnResolve 1
TransPort 9040
TransListenAddress 192.168.42.1
DNSPort 53
DNSListenAddress 192.168.42.1
```

20 IP-Tabellen leeren

Wir müssen jetzt die aktuellen IP-Tabellen leeren, um das Routing über Tor laufen lassen zu können. Tippen Sie hierfür zunächst diese Kommandos:

```
$ sudo iptables -F
$ sudo iptables -t nat -F
```

Wenn Sie SSH geöffnet lassen möchten, um die Möglichkeit eines Fernzugriffs zu haben, ist es notwendig, eine entsprechende Ausnahmeregel aufzustellen:




```
$ sudo iptables -t nat -A PREROUTING -i wlan0 -p tcp --dport 22 -j REDIRECT --to-ports 22
```

21 Umleiten

Routen Sie als Erstes sämtlichen DNS-Traffic:

```
$ sudo iptables -t nat -A PREROUTING -i wlan0 -p udp --dport 53 -j REDIRECT --to-ports 53
```

Und dann den TCP-Traffic:

```
$ sudo iptables -t nat -A PREROUTING -i wlan0 -p tcp --syn -j REDIRECT --to-ports 9040
```

22 Prüfen und speichern

Prüfen Sie das Tabellen-Setup mit diesem Befehl:

```
$ sudo iptables -t nat -L
```

Sind Sie zufrieden, dann speichern Sie es wie zuvor in die NAT-Datei:

```
$ sudo sh -c “iptables-save > /etc/iptables.ipv4.nat”
```

23 Logdatei

Wir sollten noch eine Logdatei erstellen, die uns später bei der etwaigen Fehlerbeseitigung gute Dienste leisten wird. Erstellen Sie die Logdatei so:

```
$ sudo touch /var/log/tor/notices.log
$ sudo chown debian-tor /var/log/tor/notices.log
$ sudo chmod 644 /var/log/tor/notices.log
```

Sie können sie wie folgt abrufen:

```
$ ls -l /var/log/tor
```

24 Router sichern

Schließlich können wir nun den Tor-Service aktivieren und ab dann den Wireless Access Point sicher nutzen:

```
$ sudo service tor start
```

Falls gewünscht, können Sie das prüfen:

```
$ sudo service tor status
```

Um Tor beim Booten zu starten, fügen Sie den Dienst mit folgendem Befehl der Datei rc.d hinzu:

```
$ sudo update-rc.d tor enable
```

Space Invaders nachprogrammieren

Schreiben Sie Ihren eigenen Pi-Shooter in 300 Zeilen Python.

Was Sie brauchen:

- Raspbian
  raspberrypi.org/downloads
- Python
  python.org/doc
- Pygame
  pygame.org/docs

Wenn man eine neue Programmiersprache oder ein Modul erlernt, kann man sein Verständnis der Werkzeuge gut erweitern, indem man mit einem vertrauten und relativ simplen Projekt experimentiert. Unser Space-Invaders-Klon, den wir Pivaders getauft haben, ist so ein Beispiel, das sich für Python und das Pygame-Modul hervorragend eignet – ein einfaches Spiel, dessen Regeln und Logik fast jeder kennt. Während die Invasoren sich in die Richtung des Spielers schlängeln, muss dieser sie abschießen und ihren wirren Schüssen ausweichen. Nach jeder abgewehrten Angriffswelle erscheint eine noch schnellere und aggressivere. Wir nutzen die Features von Pygame, welches das Erstellen von Spielen und interaktiven Anwendungen vereinfacht. Die Sprite-Klasse erspart uns zahlreiche Codezeilen zur Kollisionserkennung, und die Bildschirmaktualisierung und viele Aktionen werden zu Einzeilern.

Hoffentlich stimmen Sie zu, dass dieses Spiel aufregend zu spielen und gut geeignet zum Lernen von Python und Pygame ist. (Im nächsten Artikel statten wir unser kleines Game noch mit Animationen und Soundeffekten aus, um dem Anspruch auf Glanz gerecht zu werden, den ein anständiger Space-Invaders-Shooter verlangt.)

01 Abhängigkeiten installieren

Um ein besseres Verständnis des Programmierens von Spielen mit Python und Pygame zu erlangen, empfehlen wir, dass Sie den Pivaders-Code aus dieser Übung in Ihr eigenes Programm kopieren. So können Sie Elemente des Spiels Ihren eigenen Vorstellungen anpassen, etwa die Bilder für die Schiffe, den Schwierigkeitsgrad oder das Verhalten der Aliens. Sie können das Spiel aber natürlich auch einfach nur spielen. Die einzige Abhängigkeit für das Spiel ist Pygame, das (falls nicht bereits vorhanden) wie folgt vom Terminal aus installiert werden kann:

```
sudo apt-get install python-pygame
```

02 Projekt herunterladen

Für Pivaders verwenden wir Git – eine Versionskontrolle, mit der die Programmdateien zusammen mit allen Vorgängerversionen gesichert werden. Normalerweise ist Git auf dem Pi vorinstalliert. Falls nicht, tippen Sie:

```
sudo apt-get install git
```

Git agiert als Verwaltung für den Code, aber ermöglicht auch, Kopien des Codes von anderer Leute Projekte zu klonen, um sie zu verwenden oder weiterzuentwickeln. Um Pivaders zu klonen, wechseln Sie im Terminal in Ihren Heimordner (**cd ~**), erstellen Sie einen Ordner für das Projekt (**mkdir pivaders**),

Der komplette Code (weiter nächste Seite)

Den vollständigen Programmcode gibt es hier als Zip-Archiv: bit.ly/11k5f2x

```
#!/usr/bin/env python2

import pygame, random

BLACK = (0, 0, 0)
BLUE = (0, 0, 255)
WHITE = (255, 255, 255)
RED = (255, 0, 0)
ALIEN_SIZE = (30, 40)
ALIEN_SPACER = 20
BARRIER_ROW = 10
BARRIER_COLUMN = 4
BULLET_SIZE  = (5, 10)
MISSILE_SIZE = (5, 5)
BLOCK_SIZE = (10, 10)
RES = (800, 600)

class Player(pygame.sprite.Sprite):
  def __init__(self):
    pygame.sprite.Sprite.__init__(self)
    self.size = (60, 55)
    self.rect = self.image.get_rect()
    self.rect.x = (RES[0] / 2) - (self.size[0] / 2)
    self.rect.y = 520
    self.travel = 7
    self.speed = 350
    self.time = pygame.time.get_ticks()

  def update(self):
    self.rect.x += GameState.vector * self.travel
    if self.rect.x < 0:
      self.rect.x = 0
    elif self.rect.x > RES[0] - self.size[0]:
      self.rect.x = RES[0] - self.size[0]

class Alien(pygame.sprite.Sprite):
  def __init__(self):
    pygame.sprite.Sprite.__init__(self)
    self.size = (ALIEN_SIZE)
    self.rect = self.image.get_rect()
    self.has_moved = [0, 0]
    self.vector = [1, 1]
    self.travel = [(ALIEN_SIZE[0] - 7), ALIEN_SPACER]
    self.speed = 700
    self.time = pygame.time.get_ticks()

  def update(self):
    if GameState.alien_time - self.time > self.speed:
      if self.has_moved[0] < 12:
        self.rect.x += self.vector[0] * self.travel[0]
        self.has_moved[0] +=1
      else:
        if not self.has_moved[1]:
          self.rect.y += self.vector[1] * self.travel[1]
        self.vector[0] *= -1
        self.has_moved = [0, 0]
        self.speed -= 20
        if self.speed <= 100:
          self.speed = 100
      self.time = GameState.alien_time

class Ammo(pygame.sprite.Sprite):
  def __init__(self, color, (width, height)):
    pygame.sprite.Sprite.__init__(self)
    self.image = pygame.Surface([width, height])
    self.image.fill(color)
    self.rect = self.image.get_rect()
    self.speed = 0
    self.vector = 0

  def update(self):
    self.rect.y += self.vector * self.speed
    if self.rect.y < 0 or self.rect.y > RES[1]:
      self.kill()

class Block(pygame.sprite.Sprite):
  def __init__(self, color, (width, height)):
    pygame.sprite.Sprite.__init__(self)
    self.image = pygame.Surface([width, height])
    self.image.fill(color)
    self.rect = self.image.get_rect()
```

Sauberer Code

Indem wir alle häufig verwendeten globalen Variablen hier deutlich auflisten, ist der spätere Code lesbarer. Außerdem können wir, wenn wir beispielsweise die Größe von etwas ändern möchten, das an einer Stelle tun, und es funktioniert automatisch überall.

Es regnet Geschosse

Die Ammo-Klasse ist kurz und bündig. Wir brauchen nur ein paar Attribute zu initialisieren und in der update-Methode zu überprüfen, ob etwas noch im Spielbereich ist. Wenn nicht, wird es entfernt.

```
class GameState:
  pass

class Game(object):
  def __init__(self):
    pygame.init()
    pygame.font.init()
    self.clock = pygame.time.Clock()
    self.game_font = pygame.font.Font(
    'data/Orbitracer.ttf', 28)
    self.intro_font = pygame.font.Font(
    'data/Orbitracer.ttf', 72)
    self.screen = pygame.display.set_mode([RES[0], RES[1]])
    self.time = pygame.time.get_ticks()
    self.refresh_rate = 20
    self.rounds_won = 0
    self.level_up = 50
    self.score = 0
    self.lives = 2
    self.player_group = pygame.sprite.Group()
    self.alien_group = pygame.sprite.Group()
    self.bullet_group = pygame.sprite.Group()
    self.missile_group = pygame.sprite.Group()
    self.barrier_group = pygame.sprite.Group()
    self.all_sprite_list = pygame.sprite.Group()
    self.intro_screen = pygame.image.load(
    'data/start_screen.jpg').convert()
    self.background = pygame.image.load(
    'data/Space-Background.jpg').convert()
    pygame.display.set_caption('Pivaders - ESC to exit')
    pygame.mouse.set_visible(False)
    Player.image = pygame.image.load(
    'data/ship.png').convert()
    Player.image.set_colorkey(BLACK)
    Alien.image = pygame.image.load(
    'data/Spaceship16.png').convert()
    Alien.image.set_colorkey(WHITE)
    GameState.end_game = False
    GameState.start_screen = True
    GameState.vector = 0
    GameState.shoot_bullet = False

  def control(self):
    for event in pygame.event.get():
      if event.type == pygame.QUIT:
        GameState.start_screen = False
        GameState.end_game = True
      if event.type == pygame.KEYDOWN \
      and event.key == pygame.K_ESCAPE:
        if GameState.start_screen:
          GameState.start_screen = False
          GameState.end_game = True
          self.kill_all()
        else:
          GameState.start_screen = True
    self.keys = pygame.key.get_pressed()
    if self.keys[pygame.K_LEFT]:
      GameState.vector = -1
    elif self.keys[pygame.K_RIGHT]:
      GameState.vector = 1
    else:
      GameState.vector = 0
    if self.keys[pygame.K_SPACE]:
      if GameState.start_screen:
        GameState.start_screen = False
        self.lives = 2
        self.score = 0
        self.make_player()
        self.make_defenses()
        self.alien_wave(0)
      else:
        GameState.shoot_bullet = True

  def splash_screen(self):
    while GameState.start_screen:
      self.kill_all()
      self.screen.blit(self.intro_screen, [0, 0])
      self.screen.blit(self.intro_font.render(
      "PIVADERS", 1, WHITE), (265, 120))
      self.screen.blit(self.game_font.render(
      "PRESS SPACE TO PLAY", 1, WHITE), (274, 191))
```

Gruppen

Jede der Gruppen, die wir hier erstellen, ist im Prinzip eine Menge. Jedes Mal, wenn wir ein solches Objekt generieren, wird es zu der Menge hinzugefügt, sodass Kollisionstests und das Zeichnen einfach durchgeführt werden können.

Tastatur

Die control-Methode übernimmt die Tastatureingaben. Sie reagiert entsprechend auf Tastenereignisse, je nachdem, ob wir uns auf dem Startbildschirm oder im Spiel befinden.

wechseln Sie hinein (**cd pivaders**) und geben Sie ein:

```
git pull https://github.com/
russb78/pivaders.git
```

03 Pivaders testen

Jetzt da Pygame installiert und das Projekt geklont ist, können wir es kurz testen, um zu sehen, ob alles richtig eingestellt ist. Geben Sie dazu einfach **python pivaders.py** im pivaders-Ordner im Terminal ein. Mit der Leertaste können Sie das Spiel starten und dann schießen und mit der Links-/Rechts-Taste das Raumschiff bewegen.

04 Ihren eigenen Klon erstellen

Nachdem Sie einen guten Highscore (alles ab 2.000 Punkten kann sich sehen lassen) erreicht und unsere simple Implementation kennengelernt haben, können Sie tiefer einsteigen, indem Sie den Code erkunden und unseren kurzen Erklärungen folgen. Sie können auch Ihr eigenes Projekt starten, indem Sie einen Ordner anlegen und entweder IDLE oder Leafpad verwenden (oder vielleicht Geany installieren), um Ihre eigenen .py-Dateien zu erstellen und zu speichern.

05 Globale Variablen und Tupel

Nach dem Import der Module, die wir für das Projekt brauchen, haben wir eine lange Liste von Variablen in Großbuchstaben. Die Großschreibung soll anzeigen, dass diese Variablen Konstanten sind, die sich nicht ändern – sie repräsentieren Werte, auf die wir im Code regelmäßig Bezug nehmen, beispielsweise Farben, Größen und Auflösungen. Die Farben und Größen enthalten mehrere Zahlen in Klammern – das sind Tupel. Man kann auch eckige Klammern verwenden (dann sind es Listen), aber wir verwenden Tupel, weil diese unveränderlich sind, das heißt man kann nicht einzelne Elemente darin neu zuweisen. Perfekt für Konstanten, die sich ja auch gar nicht verändern sollen.

06 Klassen

Eine Klasse ist im Prinzip eine Blaupause für ein Objekt. Im Falle von **player** (Spieler) enthält sie alle notwendigen Informationen, von denen wir beliebig viele Kopien machen können (wir erstellen ein player-Objekt in der **make_player()**-Methode in der Mitte des Projekts). Die Klassen in Pivaders erben eine Reihe von Fähigkeiten und Abkürzungen von der Sprite-Klasse aus Pygame, was durch **pygame.sprite.Sprite** in Klammern in der ersten Zeile der Klasse angegeben wird. Erfahren Sie mehr über die Sprite-Klasse in der Dokumentation (**pygame.org/docs/ref/sprite.html**).

Hinsichtlich der Klassen von Pivaders (außer der Block-Klasse) werden Sie feststellen, dass neben den erforderlichen Attributen – das sind einfach Variablen in Klassen – für die Objekte (ob Spieler, Alien oder Geschoss) auch eine **update()**-Methode vorhanden ist. Diese Methode wird in jedem Durchlauf der Spielschleife (unsere heißt **main_loop()**) aufgerufen und bewirkt bei der jeweiligen Klasse eine Bewegung. Im Falle eines Geschosses der Klasse **Ammo** wird eine Bewegung entlang des Bildschirms bewirkt. Wenn es den oberen oder unteren Rand des Bildschirms erreicht, wird es entfernt (es wird ja nicht mehr gebraucht).

08 Geschosse

Das Interessanteste an Klassen ist, dass man mit einer einzelnen Klasse viele verschiedene Dinge schaffen kann. Man könnte zum Beispiel aus einer Haustier-Klasse eine Katze (die miaut) und einen Hund (der bellt) erstellen. Sie sind in vielerlei Hinsicht verschieden, aber beide haben Fell und vier Beine, also können sie von der gleichen übergeordneten Klasse abgeleitet werden. Genau das haben wir mit der Ammo-Klasse gemacht, um sowohl die Geschosse des Spielers als auch die der Aliens abzubilden. Sie haben verschiedene Farben und schießen in entgegengesetzte Richtungen, aber ansonsten sind sie im Grunde gleich. Auf diese Weise sparen wir unnötigen Code und gewährleisten einheitliches Verhalten zwischen den Objekten.

Bei der Gestaltung des Spiels haben wir auf frei benutzbare Open-Source-Kunst und -Fonts zurückgegriffen.

09 Das Spiel

Die letzte Klasse ist **Game** (Spiel). Hier befindet sich die Hauptfunktionalität des eigentlichen Spiels, aber bedenken Sie, bisher ist es nur eine Liste von Zutaten – damit überhaupt etwas passiert, muss ein Game-Objekt generiert werden (ganz am Ende des Codes). In der Game-Klasse befindet sich die meiste Substanz des Spiels, also initialisieren wir Pygame, legen die Bilder für den Spieler und die außerirdischen Gegner fest und erstellen einige GameState-Attribute, mit denen wir wichtige Aspekte der externen Klassen steuern, etwa den Vektor (Richtung) des Spielers ändern oder entscheiden, ob wir zum Startbildschirm zurückkehren müssen.

10 Die Hauptschleife

In der Game-Klasse sind viele Methoden (Klassenfunktionen), die jeweils einen Aspekt des Spielaufbaus oder Spielverlaufs steuern. Die eigentliche Logik, die festlegt, was in einer Runde des Spiels passiert, befindet sich in der **main_loop()**-Methode am Ende der **pivaders.py**-Datei. Damit entschlüsseln Sie, welche Variablen und Funktionen für das Spiel benötigt werden. Lesen Sie vom Anfang der **main_loop()**-Methode Zeile für Zeile bis zum Ende, um genau zu erfahren, was 20-mal pro Sekunde berechnet wird, während man das Spiel spielt.

11 Logik der Hauptschleife

Zuerst prüfen wir, ob das **end_game**-Attribut False ist – wenn es True ist, wird die gesamte **main_loop()**-Schleife übersprungen und wir landen bei **pygame.quit()**, wodurch das Spiel beendet wird. Das Attribut wird nur auf True gesetzt, wenn der Spieler das Spielfenster schließt oder auf dem Startbildschirm (**start_screen**) Escape drückt. Wenn end_game und start_screen beide False sind, startet die eigentliche Hauptschleife mit der **control()**-Methode, die prüft, ob die Position des Spielers geändert werden muss. Danach versuchen wir, ein gegnerisches Geschoss zu erzeugen, aber mit dem **random**-Modul begrenzen wir die Anzahl der Geschosse, die erzeugt werden. Danach rufen wir mithilfe einer for-Schleife für jedes sichtbare Objekt die **update()**-Methode auf, damit alles auf seine neue Position rückt, und schließt prüfen wir auf Kollisionen mit **calc_collisions()**.

Nach der Kollisionserkennung prüfen wir, ob das Spiel noch weiter geht. Dazu rufen wir **is_dead()** (ist tot) und **defenses_breached()** (Verteidigung durchbrochen) auf – gibt eine dieser Methoden True zurück, müssen wir zum Startbildschirm zurückkehren. Andererseits müssen wir auch schauen, ob wir alle Aliens getroffen haben – das macht **win_round()** (Runde gewonnen). Ist der Spieler nicht tot, die Aliens aber schon, dann rufen wir **next_round()** (nächste Runde) auf, um einen neuen Schub Aliens mit höherer Geschwindigkeit zu erzeugen. Zum Schluss zeichnen wir den Bildschirm neu, damit alles, was sich bewegt hat oder gestorben ist, entsprechend angezeigt oder entfernt wird. Vergessen Sie nicht, dass die Schleife 20-mal pro Sekunde durchläuft – die Tatsache, dass die Bildschirmaktualisierung nicht ganz am Ende der Schleife geschieht, fällt also nicht ins Gewicht.

Eine Klasse ist im Prinzip eine Blaupause für ein Objekt.

```
    pygame.display.flip()
    self.control()

def make_player(self):
  self.player = Player()
  self.player_group.add(self.player)
  self.all_sprite_list.add(self.player)

def refresh_screen(self):
  self.all_sprite_list.draw(self.screen)
  self.refresh_scores()
  pygame.display.flip()
  self.screen.blit(self.background, [0, 0])
  self.clock.tick(self.refresh_rate)

def refresh_scores(self):
  self.screen.blit(self.game_font.render(
  "SCORE " + str(self.score), 1, WHITE), (10, 8))
  self.screen.blit(self.game_font.render(
  "LIVES " + str(self.lives + 1), 1, RED), (355, 575))

def alien_wave(self, speed):
  for column in range(BARRIER_COLUMN):
    for row in range(BARRIER_ROW):
      alien = Alien()
      alien.rect.y = 65 + (column * (
      ALIEN_SIZE[1] + ALIEN_SPACER))
      alien.rect.x = ALIEN_SPACER + (
      row * (ALIEN_SIZE[0] + ALIEN_SPACER))
      self.alien_group.add(alien)
      self.all_sprite_list.add(alien)
      alien.speed -= speed

def make_bullet(self):
  if GameState.game_time - self.player.time > self.player.speed:
    bullet = Ammo(BLUE, BULLET_SIZE)
    bullet.vector = -1
    bullet.speed = 26
    bullet.rect.x = self.player.rect.x + 28
    bullet.rect.y = self.player.rect.y
    self.bullet_group.add(bullet)
    self.all_sprite_list.add(bullet)
    self.player.time = GameState.game_time
  GameState.shoot_bullet = False

def make_missile(self):
  if len(self.alien_group):
    shoot = random.random()
    if shoot <= 0.05:
      shooter = random.choice([
      alien for alien in self.alien_group])
      missile = Ammo(RED, MISSILE_SIZE)
      missile.vector = 1
      missile.rect.x = shooter.rect.x + 15
      missile.rect.y = shooter.rect.y + 40
      missile.speed = 10
      self.missile_group.add(missile)
      self.all_sprite_list.add(missile)

def make_barrier(self, columns, rows, spacer):
  for column in range(columns):
    for row in range(rows):
      barrier = Block(WHITE, (BLOCK_SIZE))
      barrier.rect.x = 55 + (200 * spacer) + (row * 10)
      barrier.rect.y = 450 + (column * 10)
      self.barrier_group.add(barrier)
      self.all_sprite_list.add(barrier)

def make_defenses(self):
  for spacing, spacing in enumerate(xrange(4)):
    self.make_barrier(3, 9, spacing)

def kill_all(self):
  for items in [self.bullet_group, self.player_group,
  self.alien_group, self.missile_group, self.barrier_group]:
    for i in items:
      i.kill()
```

Bildschirmaktualisierung

Man muss die Aktualisierung des Bildschirms vorsichtig erwägen. Das Blitting des Hintergrunds zwischen Objektbewegungen ist für eine flüssige Animation unerlässlich.

Geschosse

Geschosse des Spielers und der Aliens verwenden dieselbe übergeordnete Klasse. Wir ändern ein paar wichtige Attribute, um das gewünschte Verhalten zu erzielen. Zum Beispiel zeigt der Vektor jeweils in die entgegengesetzte Richtung.

Tot oder lebendig

Zwei der wohl wichtigsten Fragen werden hier beantwortet: Ist der Spieler tot, oder hat er die Runde gewonnen?

Den vollständigen Programmcode gibt es hier als Zip-Archiv: bit.ly/11k5f2x

```
def is_dead(self):
  if self.lives < 0:
    self.screen.blit(self.game_font.render(
    "The war is lost! You scored: " + str(
    self.score), 1, RED), (250, 15))
    self.rounds_won = 0
    self.refresh_screen()
    pygame.time.delay(3000)
    return True

def win_round(self):
  if len(self.alien_group) < 1:
    self.rounds_won += 1
    self.screen.blit(self.game_font.render(
    "You won round " + str(self.rounds_won) +
    "  but the battle rages on", 1, RED), (200, 15))
    self.refresh_screen()
    pygame.time.delay(3000)
    return True

def defenses_breached(self):
  for alien in self.alien_group:
    if alien.rect.y > 410:
      self.screen.blit(self.game_font.render(
      "The aliens have breached Earth defenses!",
      1, RED), (180, 15))
      self.refresh_screen()
      pygame.time.delay(3000)
      return True

def calc_collisions(self):
  pygame.sprite.groupcollide(
  self.missile_group, self.barrier_group, True, True)
  pygame.sprite.groupcollide(
  self.bullet_group, self.barrier_group, True, True)
  if pygame.sprite.groupcollide(
  self.bullet_group, self.alien_group, True, True):
    self.score += 10
  if pygame.sprite.groupcollide(
  self.player_group, self.missile_group, False, True):
    self.lives -= 1

def next_round(self):
  for actor in [self.missile_group,
  self.barrier_group, self.bullet_group]:
    for i in actor:
      i.kill()
  self.alien_wave(self.level_up)
  self.make_defenses()
  self.level_up += 50

def main_loop(self):
  while not GameState.end_game:
    while not GameState.start_screen:
      GameState.game_time = pygame.time.get_ticks()
      GameState.alien_time = pygame.time.get_ticks()
      self.control()
      self.make_missile()
      for actor in [self.player_group, self.bullet_group,
      self.alien_group, self.missile_group]:
        for i in actor:
          i.update()
      if GameState.shoot_bullet:
        self.make_bullet()
      self.calc_collisions()
      if self.is_dead() or self.defenses_breached():
        GameState.start_screen = True
      if self.win_round():
        self.next_round()
      self.refresh_screen()
    self.splash_screen()
  pygame.quit()

if __name__ == '__main__':
  pv = Game()
  pv.main_loop()
```

Hauptschleife

Hier findet der Betrieb unserer Anwendung statt. Die Schleife wird 20-mal pro Sekunde durchlaufen. Die Schleife sollte für andere Programmierer logisch und nachvollziehbar sein.

Spiel starten

Das Allerletzte im Code ist das Erzeugen eines Game-Objekts und das Aufrufen der Hauptschleife. Abgesehen von den Konstanten ist das der einzige Code außerhalb von Klassen.

Pivaders mit Animation und Sound ausstatten

Unseren Space-Invaders-Klon in 300 Zeilen Python-Code wollen wir jetzt mit Animation und Sound erweitern.

Was Sie brauchen:

- Raspbian
  raspberrypi.org/downloads
- Python
  python.org/doc
- Pygame
  pygame.org/docs
- Bildmaterial
  opengameart.org

In der vorherigen Übung hatten wir Spaß beim Erschaffen eines einfachen Space-Invaders-Klons, den wir Pivaders getauft haben. Eine der wesentlichen Herausforderungen war es dabei, das Projekt überschaubar zu halten – nur 300 Zeilen Python-Code. Ohne die starken Features von Pygame hätten wir dieses Ziel mindestens um das Doppelte überschritten. Dass man mit der Sprite-Klasse von Pygames leicht gruppieren, verwalten und Kollisionen erkennen kann, ermöglicht enorme Einsparungen, nicht nur was die Länge des Quellcodes, sondern auch seine Komplexität angeht. Das Code-Listing v0.1 des ersten Teils des Projekts finden Sie auf GitHub unter git.io/cBVTBg, während die Version v0.2, die alle Bilder, Musik und Soundeffekte enthält, unter git.io/8QsK-w bereitsteht.

Obwohl Pygame ein Framework klar vorgibt, gibt es trotzdem viele Wege, Animation und Sound einzuarbeiten. Es gibt Dutzende Klassen, mit denen sich Sammlungen von Einzelbildern verwalten lassen, oder man kann ein Sprite-Sheet (ein Gesamtbild, das die kleineren Einzelbilder enthält) verwenden, das man dann auf den Bildschirm zeichnen kann (Blitting). Der Einfachheit und der Performance halber haben wir in unserer Game-Klasse einige Animationsmethoden integriert und uns für ein Sprite-Sheet entschieden. Damit ist es nicht nur sehr einfach, auf den Bildschirm zu zeichnen, sondern hält auch die Menge an Zusatzmaterial klein und somit die Performance aufrecht – was auf dem Raspberry Pi natürlich besonders wichtig ist.

01 Abhängigkeiten installieren

Wie auch bei der letzten Übung empfehlen wir, den Code herunterzuladen (**git.io/8QsK-w**) und als Referenz für die Animationen und Soundeffekte in Ihren eigenen Pygame-Projekten zu verwenden. Egal, ob Sie das Spiel erst einmal probespielen oder bereits den Code durchgehen möchten, um den Schaffensprozess besser zu verstehen, Sie brauchen auf jeden Fall einige Abhängigkeiten. Die zwei wesentlichen Voraussetzungen sind Pygame und Git, die beide auf neuesten Raspbian-Installationen schon vorinstalliert sind. Wenn Sie sich unsicher sind, können Sie einfach den folgenden Befehl in die Kommandozeile eingeben:

```
sudo apt-get install python-pygame git
```

Der komplette Code (ab Zeile 86, weiter nächste Seite)

Den vollständigen Programmcode gibt es hier als Zip-Archiv: bit.ly/1xPvY1F

```
class Game(object):
    def __init__(self):
        pygame.init()
        pygame.font.init()
        self.clock = pygame.time.Clock()
        self.game_font = pygame.font.Font(
        'data/Orbitracer.ttf', 28)
        self.intro_font = pygame.font.Font(
        'data/Orbitracer.ttf', 72)
        self.screen = pygame.display.set_mode([RES[0], RES[1]])
        self.time = pygame.time.get_ticks()
        self.refresh_rate = 20; self.rounds_won = 0
        self.level_up = 50; self.score = 0
        self.lives = 2
        self.player_group = pygame.sprite.Group()
        self.alien_group = pygame.sprite.Group()
        self.bullet_group = pygame.sprite.Group()
        self.missile_group = pygame.sprite.Group()
        self.barrier_group = pygame.sprite.Group()
        self.all_sprite_list = pygame.sprite.Group()
        self.intro_screen = pygame.image.load(
        'data/graphics/start_screen.jpg').convert()
        self.background = pygame.image.load(
        'data/graphics/Space-Background.jpg').convert()
        pygame.display.set_caption('Pivaders - ESC to exit')
        pygame.mouse.set_visible(False)
        Alien.image = pygame.image.load(
        'data/graphics/Spaceship16.png').convert()
        Alien.image.set_colorkey(WHITE)
        self.ani_pos = 5 # 11 images of ship
        self.ship_sheet = pygame.image.load(
        'data/graphics/ship_sheet_final.png').convert_alpha()
        Player.image = self.ship_sheet.subsurface(
                    self.ani_pos*64, 0, 64, 61)
        self.animate_right = False
        self.animate_left = False
        self.explosion_sheet = pygame.image.load(
        'data/graphics/explosion_new1.png').convert_alpha()
        self.explosion_image = self.explosion_sheet.subsurface(0, 0, ↵
79, 96)
        self.alien_explosion_sheet = pygame.image.load(
        'data/graphics/alien_explosion.png')
        self.alien_explode_graphics = self.alien_explosion_sheet. ↵
subsurface(0, 0, 94, 96)
        self.explode = False
        self.explode_pos = 0; self.alien_explode = False
        self.alien_explode_pos = 0
        pygame.mixer.music.load('data/sound/10_Arpanauts.ogg')
        pygame.mixer.music.play(-1)
        pygame.mixer.music.set_volume(0.7)
        self.bullet_fx = pygame.mixer.Sound(
        'data/sound/medetix__pc-bitcrushed-lazer-beam.ogg')
        self.explosion_fx = pygame.mixer.Sound(
        'data/sound/timgormly__8-bit-explosion.ogg')
        self.explosion_fx.set_volume(0.5)
        self.explodey_alien = []
        GameState.end_game = False
        GameState.start_screen = True
        GameState.vector = 0
        GameState.shoot_bullet = False

    def control(self):
        for event in pygame.event.get():
            if event.type == pygame.QUIT:
                GameState.start_screen = False
                GameState.end_game = True
            if event.type == pygame.KEYDOWN \
            and event.key == pygame.K_ESCAPE:
                if GameState.start_screen:
                    GameState.start_screen = False
                    GameState.end_game = True
                    self.kill_all()
                else:
                    GameState.start_screen = True
        self.keys = pygame.key.get_pressed()
        if self.keys[pygame.K_LEFT]:
            GameState.vector = -1
            self.animate_left = True
            self.animate_right = False
        elif self.keys[pygame.K_RIGHT]:
            GameState.vector = 1
            self.animate_right = True
            self.animate_left = False
        else:
            GameState.vector = 0
            self.animate_right = False
            self.animate_left = False

        if self.keys[pygame.K_SPACE]:
            if GameState.start_screen:
                GameState.start_screen = False
                self.lives = 2
                self.score = 0
                self.make_player()
                self.make_defenses()
                self.alien_wave(0)
            else:
                GameState.shoot_bullet = True
                self.bullet_fx.play()

    def animate_player(self):
        if self.animate_right:
            if self.ani_pos < 10:
                Player.image = self.ship_sheet.subsurface(
                    self.ani_pos*64, 0, 64, 61)
                self.ani_pos += 1
        else:
            if self.ani_pos > 5:
                self.ani_pos -= 1
                Player.image = self.ship_sheet.subsurface(
                    self.ani_pos*64, 0, 64, 61)

        if self.animate_left:
            if self.ani_pos > 0:
                self.ani_pos -= 1
                Player.image = self.ship_sheet.subsurface(
                    self.ani_pos*64, 0, 64, 61)
        else:
            if self.ani_pos < 5:
                Player.image = self.ship_sheet.subsurface(
                    self.ani_pos*64, 0, 64, 61)
                self.ani_pos += 1

    def player_explosion(self):
        if self.explode:
            if self.explode_pos < 8:
                self.explosion_image = self.explosion_sheet. ↵
subsurface(0, self.explode_pos*96, 79, 96)
                self.explode_pos += 1
                self.screen.blit(self.explosion_image, [self.player. ↵
rect.x -10, self.player.rect.y - 30])
            else:
                self.explode = False
                self.explode_pos = 0

    def alien_explosion(self):
        if self.alien_explode:
            if self.alien_explode_pos < 9:
                self.alien_explode_graphics = self.alien_explosion_ ↵
sheet.subsurface(0, self.alien_explode_pos*96, 94, 96)
                self.alien_explode_pos += 1
                self.screen.blit(self.alien_explode_graphics, ↵
[int(self. explodey_alien[0]) - 50 , int(self.explodey_alien[1]) - 60])
            else:
                self.alien_explode = False
                self.alien_explode_pos = 0
                self.explodey_alien = []

    def splash_screen(self):
        while GameState.start_screen:
            self.kill_all()
            self.screen.blit(self.intro_screen, [0, 0])
            self.screen.blit(self.intro_font.render(
            "PIVADERS", 1, WHITE), (265, 120))
            self.screen.blit(self.game_font.render(
            "PRESS SPACE TO PLAY", 1, WHITE), (274, 191))
            pygame.display.flip()
            self.control()
            self.clock.tick(self.refresh_rate / 2)

    def make_player(self):
        self.player = Player()
```

ship_sheet

Mit der ani_pos-Variablen legen wir fest, welcher kleine Ausschnitt aus dem Sprite-Sheet dem Spielerbild zugewiesen wird. Ändern Sie den Wert, um das Bild zu ändern.

Variablen setzen

Wir haben die Boolean-Variablen animate_left und animate_right in der control-Methode eingeführt. Wenn sie True sind, wird der eigentliche Animationscode über eine separate Methode aufgerufen.

fx.play()

Wir haben den Soundeffekt, den man beim Schießen hören soll, bereits geladen. Jetzt müssen wir ihn nur abspielen, wenn die Leertaste gedrückt wird.

02 Pivaders herunterladen

Git ist ein Versionskontrollsystem, mit dem Programmierer ihren Code und zugehörige Dateien sicher ablegen können. Man kann damit sowohl den Verlauf aller Änderungen zurückverfolgen als auch auf Websites wie **github.com** ganze Projekte „klonen“ und weiterentwickeln. Um das Projekt für diese Übung zu klonen, wechseln Sie auf der Kommandozeile in Ihren Heimordner (**cd ~**) und geben Sie ein:

```
git pull https://github.com/russb78/pivaders.git
```

Ein Ordner namens **pivaders** wird dadurch erstellt. Wechseln Sie hinein (**cd pivaders**) und schauen Sie sich um.

03 Projekt navigieren

Das Projekt ist in eine einfache Ordnerstruktur organisiert. In **pivaders** befinden sich eine Lizenz, ein Readme und ein weiterer **pivaders**-Ordner. Darin wiederum liegt die Hauptprogrammdatei namens **pivaders.py**, die die Anwendung startet. Im **data**-Ordner gibt es Unterverzeichnisse für Grafiken und Soundeffekte sowie die Schriftart für den Titelbildschirm und die Punktzahl. Um Pivaders auszuprobieren, wechseln Sie in das **pivaders**-Unterverzeichnis (**cd pivaders/pivaders**) und geben Sie ein:

```
python pivaders.py
```

Lenken Sie mit den Pfeiltasten und schießen Sie mit der Leertaste. Mit Escape können Sie zum Startbildschirm zurückkehren und mit einem erneuten Escape das Spiel ganz beenden.

04 Animation und Sound

Verglichen mit dem Spiel aus der vorherigen Übung ist dieses Projekt viel dynamischer. Das Protagonistenschiff lehnt sich zur Seite, wenn man die Richtung wechselt, und richtet sich wieder auf, wenn man die Taste loslässt. Wenn man ein gegnerisches Schiff abschießt, explodiert es in einer Animation, und wenn man selbst getroffen wird, erscheint eine kleinere Explosion am eigenen Schiff. Musik und Sounds für Laser und Explosionen begleiten die Animationen.

Oben Die Website Freesound hält einen Fundus kostenloser, frei verwendbarer Geräusche bereit.

05 Animierbare Bilder finden

Bevor wir mit dem Programmieren loslegen, ist es sinnvoll, die notwendigen Materialien zusammenzusuchen. Wie bereits erwähnt haben wir uns für die Verwendung von Sprite-Sheets entschieden; diese können Sie online finden oder mit etwas Übung mit GIMP selbst erstellen. Es handelt sich im Prinzip um ein Mosaik aus gleich großen Bildern in gleichem Abstand, die die „Frames“ der Animation ausmachen. Auf **opengameart.org** finden sich fertige Beispiele (so wie jenes, das wir hier verwenden).

06 Material anpassen

Viele der Materialen auf Websites wie **opengameart.org** lassen sich zwar unverändert wiederverwenden, doch manchmal möchte man sie lieber in ein Bildbearbeitungsprogramm wie GIMP laden und an die Anforderungen anpassen – bei dem Sprite-Sheet für unser Schiff haben wir das gemacht, um den Code einfach zu halten. Wir haben zunächst das zentrale Schiff-Sprite in ein neues Fenster zentriert. Wir haben die Größe und Breite der Frames gesetzt und die restlichen Frames auf beide Seiten kopiert. Danach hatten wir 11 Frames der gleichen Größe und Breite in einem Dokument. Pixelgenaue Präzision ist hier wichtig, damit wir für jeden Frame einfache Multiplikation nutzen können.

07 Sprite-Sheet laden

Da die **Player**-Klasse von der **Sprite**-Klasse abgeleitet ist, können wir das Aussehen des Spielers ändern, indem wir **Player.image** modifizieren. Zuerst laden wir das Sprite-Sheet für das Schiff mit **pygame.image.load()**. Da wir das Sprite-Sheet mit einem transparenten Hintergrund erstellt haben, können wir **.convert_alpha()** an das Ende der Zeile anhängen, sodass die Schiff-Frames korrekt (ohne Hintergrund) erscheinen. Dann verwenden wir **subsurface**, um das anfängliche **Player.image** auf das Schiff-Sprite in der Mitte des Sprite-Sheets zu setzen. Das ist durch **self.ani_pos** bestimmt, welches einen Anfangswert von 5 hat. Ein anderer Wert würde das Aussehen des Schiffs verändern: Mit 0 würde es sich ganz nach links lehnen, mit 11 ganz nach rechts.

08 Animations-Optionen

Etwas weiter unten in der Liste im Initialisierungscode der **Game**-Klasse setzen wir zwei Optionen für die Spieleranimation: **self.animate_left** und **self.animate_right**. In der **control**-Methode der **Game**-Klasse sehen Sie, dass wir diese Variablen auf Boolean-Werte (**True** oder **False**) setzen, um anzuzeigen, ob eine Animation vorkommen soll. So können wir „automatisch“ das Spieler-Sprite in seine Ursprungsposition zurückanimieren (sonst würde es sich weiterhin nach links lehnen, selbst wenn es längst zum Stillstand gekommen ist).

09 Die Animationsmethode

Die obengenannten Optionen kommen im Code für die Animation des Spielers wieder vor (**animate_player()** in der **Game**-Klasse). Hier verwenden wir verschachtelte if-Anweisungen, um die Animation zu steuern und das Spielerbild entsprechend zu setzen. Wenn **animate_right** auf **True** steht und die aktuelle Animationsposition nicht die ist, die sie sein sollte, wird die **ani_pos**-Variable inkrementell angepasst und das Spielerbild entsprechend gesetzt. Die else-Anweisung animiert das Schiff-Sprite zurück in die Ursprungsposition, und die gleiche Logik wird für die andere Richtung angewandt.

10 Explosionen animieren

Die Methoden **player_explosion()** (Spieler-Explosion) und **alien_explosion()** (Gegner-Explosion) hinter dem Spieler-Animationsblock in der Game-Klasse sind ähnliche, aber einfachere Ausführungen desselben Prinzips. Wir müssen ja nur eine vorgegebene Anzahl von Frames durchlaufen (diesmal vertikal), also prüfen wir einfach, ob **self.explode** und **self.alien_explode** auf **True** stehen, und inkrementieren dann die Variable, die das angezeigte Bild ändert. Da das Sprite-Sheet vertikal ist, wird diesmal ein anderer Teil von **subsurface** auf die Variablen **alien_explode_pos** und **explosion_image** gesetzt.

11 Projekt mit Musik bereichern

Mit Pygame lässt sich leicht eine Hintergrundmusik zum Projekt hinzufügen. Besorgen Sie sich ein geeignetes Musikstück (wir haben unseres von **freemusicarchive.org**) und laden Sie es mithilfe der **Mixer**-Klasse aus Pygame. Da es mit **pygame.init()** bereits initialisiert wurde, können wir die Musik einfach mit folgendem Code laden:

```
pygame.mixer.music.load('data/sound/10_Arpanauts.ogg')
pygame.mixer.music.play(-1)
pygame.mixer.music.set_volume(0.7)
```

Der zweite Befehl besagt, dass die Musik beim Programmstart anfangen und bis zum Beenden in einer Endlosschleife spielen soll. Ersetzten wir die -1 durch eine 5, würde die Musik fünfmal spielen und dann verstummen. Mehr über die **Mixer**-Klasse erfahren Sie auf **pygame.org/docs/ref/mixer.html**.

12 Soundeffekte

Sounds zu laden und abzuspielen, funktioniert in Pygame so ähnlich wie das Verfahren für Bilder. Zuerst laden wir den Soundeffekt mit einer einfachen Zuweisung. Für den Laserstrahl sieht die Initialisierung so aus:

```
self.bullet_fx = pygame.mixer.Sound(
    'data/sound/medetix__pc-bitcrushed-lazer-beam.ogg')
```

Dann lösen wir den Soundeffekt einfach zur richtigen Zeit aus. Der Laser soll beispielsweise immer beim Drücken der Leertaste (also beim Schießen) abgespielt werden, daher platzieren wir ihn in der **control**-Methode der **Game**-Klasse direkt hinter die Zuweisung auf **shoot_bullet**.

Kostenlose Soundeffekte findet man auf **freesound.org**.

```
        self.player_group.add(self.player)
        self.all_sprite_list.add(self.player)

    def refresh_screen(self):
        self.all_sprite_list.draw(self.screen)
        self.animate_player()
        self.player_explosion()
        self.alien_explosion()
        self.refresh_scores()
        pygame.display.flip()
        self.screen.blit(self.background, [0, 0])
        self.clock.tick(self.refresh_rate)

    def refresh_scores(self):
        self.screen.blit(self.game_font.render(
        "SCORE " + str(self.score), 1, WHITE), (10, 8))
        self.screen.blit(self.game_font.render(
        "LIVES " + str(self.lives + 1), 1, RED), (355, 575))

    def alien_wave(self, speed):
        for column in range(BARRIER_COLUMN):
            for row in range(BARRIER_ROW):
                alien = Alien()
                alien.rect.y = 65 + (column * (
                ALIEN_SIZE[1] + ALIEN_SPACER))
                alien.rect.x = ALIEN_SPACER + (
                row * (ALIEN_SIZE[0] + ALIEN_SPACER))
                self.alien_group.add(alien)
                self.all_sprite_list.add(alien)
                alien.speed -= speed

    def make_bullet(self):
        if GameState.game_time - self.player.time > self.player.speed:
            bullet = Ammo(BLUE, BULLET_SIZE)
            bullet.vector = -1
            bullet.speed = 26
            bullet.rect.x = self.player.rect.x + 28
            bullet.rect.y = self.player.rect.y
            self.bullet_group.add(bullet)
            self.all_sprite_list.add(bullet)
            self.player.time = GameState.game_time
        GameState.shoot_bullet = False

    def make_missile(self):
        if len(self.alien_group):
            shoot = random.random()
            if shoot <= 0.05:
                shooter = random.choice([
                alien for alien in self.alien_group])
                missile = Ammo(RED, MISSILE_SIZE)
                missile.vector = 1
                missile.rect.x = shooter.rect.x + 15
                missile.rect.y = shooter.rect.y + 40
                missile.speed = 10
                self.missile_group.add(missile)
                self.all_sprite_list.add(missile)

    def make_barrier(self, columns, rows, spacer):
        for column in range(columns):
            for row in range(rows):
                barrier = Block(WHITE, (BLOCK_SIZE))
                barrier.rect.x = 55 + (200 * spacer) + (row * 10)
                barrier.rect.y = 450 + (column * 10)
                self.barrier_group.add(barrier)
                self.all_sprite_list.add(barrier)

    def make_defenses(self):
        for spacing, spacing in enumerate(xrange(4)):
            self.make_barrier(3, 9, spacing)

    def kill_all(self):
        for items in [self.bullet_group, self.player_group,
        self.alien_group, self.missile_group, self.barrier_group]:
            for i in items:
                i.kill()

    def is_dead(self):
        if self.lives < 0:
            self.screen.blit(self.game_font.render(
            "The war is lost! You scored: " + str(
            self.score), 1, RED), (250, 15))
            self.rounds_won = 0
            self.refresh_screen()
            self.level_up = 50
            self.explode = False
            self.alien_explode = False
            pygame.time.delay(3000)
            return True

    def defenses_breached(self):
        for alien in self.alien_group:
            if alien.rect.y > 410:
                self.screen.blit(self.game_font.render(
                "The aliens have breached Earth defenses!",
                1, RED), (180, 15))
                self.refresh_screen()
                self.level_up = 50
                self.explode = False
                self.alien_explode = False
                pygame.time.delay(3000)
                return True

    def win_round(self):
        if len(self.alien_group) < 1:
            self.rounds_won += 1
            self.screen.blit(self.game_font.render(
            "You won round " + str(self.rounds_won) +
            "  but the battle rages on", 1, RED), (200, 15))
            self.refresh_screen()
            pygame.time.delay(3000)
            return True

    def next_round(self):
        self.explode = False
        self.alien_explode = False
        for actor in [self.missile_group,
        self.barrier_group, self.bullet_group]:
            for i in actor:
                i.kill()
        self.alien_wave(self.level_up)
        self.make_defenses()
        self.level_up += 50

    def calc_collisions(self):
        pygame.sprite.groupcollide(
        self.missile_group, self.barrier_group, True, True)
        pygame.sprite.groupcollide(
        self.bullet_group, self.barrier_group, True, True)

        for z in pygame.sprite.groupcollide(
        self.bullet_group, self.alien_group, True, True):
            self.alien_explode = True
            self.explodey_alien.append(z.rect.x)
            self.explodey_alien.append(z.rect.y)
            self.score += 10
            self.explosion_fx.play()

        if pygame.sprite.groupcollide(
        self.player_group, self.missile_group, False, True):
            self.lives -= 1
            self.explode = True
            self.explosion_fx.play()

    def main_loop(self):
        while not GameState.end_game:
            while not GameState.start_screen:
                GameState.game_time = pygame.time.get_ticks()
                GameState.alien_time = pygame.time.get_ticks()
                self.control()
                self.make_missile()
                self.calc_collisions()
                self.refresh_screen()
                if self.is_dead() or self.defenses_breached():
                    GameState.start_screen = True
                for actor in [self.player_group, self.bullet_group,
                self.alien_group, self.missile_group]:
                    for i in actor:
                        i.update()
                if GameState.shoot_bullet:
                    self.make_bullet()
                if self.win_round():
                    self.next_round()
            self.splash_screen()
        pygame.quit()

if __name__ == '__main__':
    pv = Game()
    pv.main_loop()
```

Den vollständigen Programmcode gibt es hier als Zip-Archiv: bit.ly/1xPvY1F

Quadrocopter mit Raspberry Pi

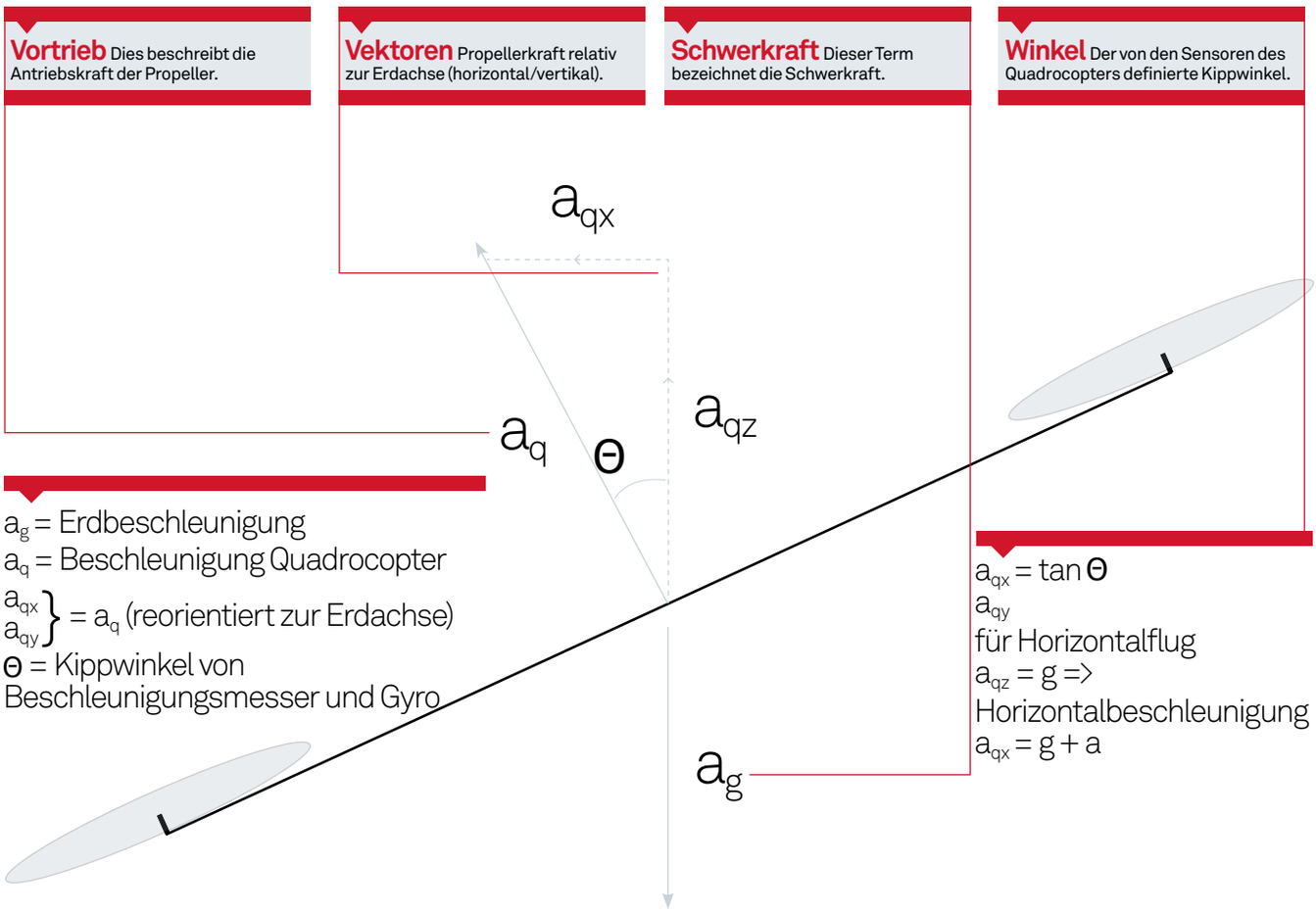
Was ist noch besser als ein Raspberry-Pi-Roboter? Einer, der fliegt! Andy Baker zeigt, wie Sie sich in luftige Höhen coden.

Der Raspberry Pi ist eine fantastische Projektplattform. Auf der Suche nach einer Herausforderung, die den Raspberry-Pi-Roboter von Seite 44 noch übertrifft und uns in Sachen Hacken und Coden an die Grenzen bringt, sind wir auf den Quadrocopter gestoßen.

Man kann ein solches Fluggerät zwar bereits „ready-to-fly“ (RTF) kaufen, wenn man einfach nur damit fliegen und Spaß haben möchte, doch das ist keine große Herausforderung. Unsere Basis ist ein Bausatz, der „almost-ready-to-fly“ (ARF) ist: der DJI Flame Wheel F450. Er enthält sämtliche Hardware, aber keine Steuerelektronik oder Software. Viele Eigenbau-Quadrocopter nutzen den Arduino, es muss also auch mit dem Raspberry Pi funktionieren, wenngleich dieser dafür noch kaum verwendet wurde.

Der Artikel orientiert sich am Python-Code auf **bit.ly/1sHD09w**, sodass Sie Schritt für Schritt alle Teile zusammensetzen können und so am Ende nicht nur den Code verstanden haben, sondern auch seine Interaktion mit der Umgebung – somit gehen Sie gut vorbereitet an Ihr eigenes Quadrocopter-Projekt.

Im Artikel selbst werden Sie durch Tags geleitet, deren Entsprechung Sie im Code finden. Um beispielsweise den Code zum Abschnitt „# Angles“ zu finden, suchen Sie einfach nach dem Tag „# Angles“.



Oben So werden Sensoren aus Sicht des Quadrocopters erdabhängig (horizontal/vertikal) konvertiert, um den Horizontalflug zu ermöglichen.

Interpreter

Der Befehlsinterpreter konvertiert Anweisungsketten, entweder von einer Fernsteuerung oder direkt aus dem Code. Die Anweisungen kombinieren Richtung und Geschwindigkeit in Abhängigkeit vom Pfad, den der Nutzer dem Quadrocopter vorgibt. Der Code konvertiert sie in eine Reihe von Zielvorgaben für Vertikal- und Horizontalgeschwindigkeit und Giergeschwindigkeit – jede Joystickeingabe kann in solche Zielvorgaben aufgeschlüsselt werden.

Inputs

Eingaben bekommt der Quadrocopter von elektronischen Sensoren, die Bewegungsinformationen bereitstellen. Ein Beschleunigungsmesser misst die Beschleunigungskraft (inklusive Schwerkraft) an den drei Achsen des Quadrocopters, ein Kreiselinstrument (Gyroskop) misst die Achsengeschwindigkeit, mit der er nickt (Bug/Heck nach oben oder unten), rollt (links/rechts nach oben oder unten) und giert (Drehung im und gegen den Uhrzeigersinn um seine Hauptachse).

Axes

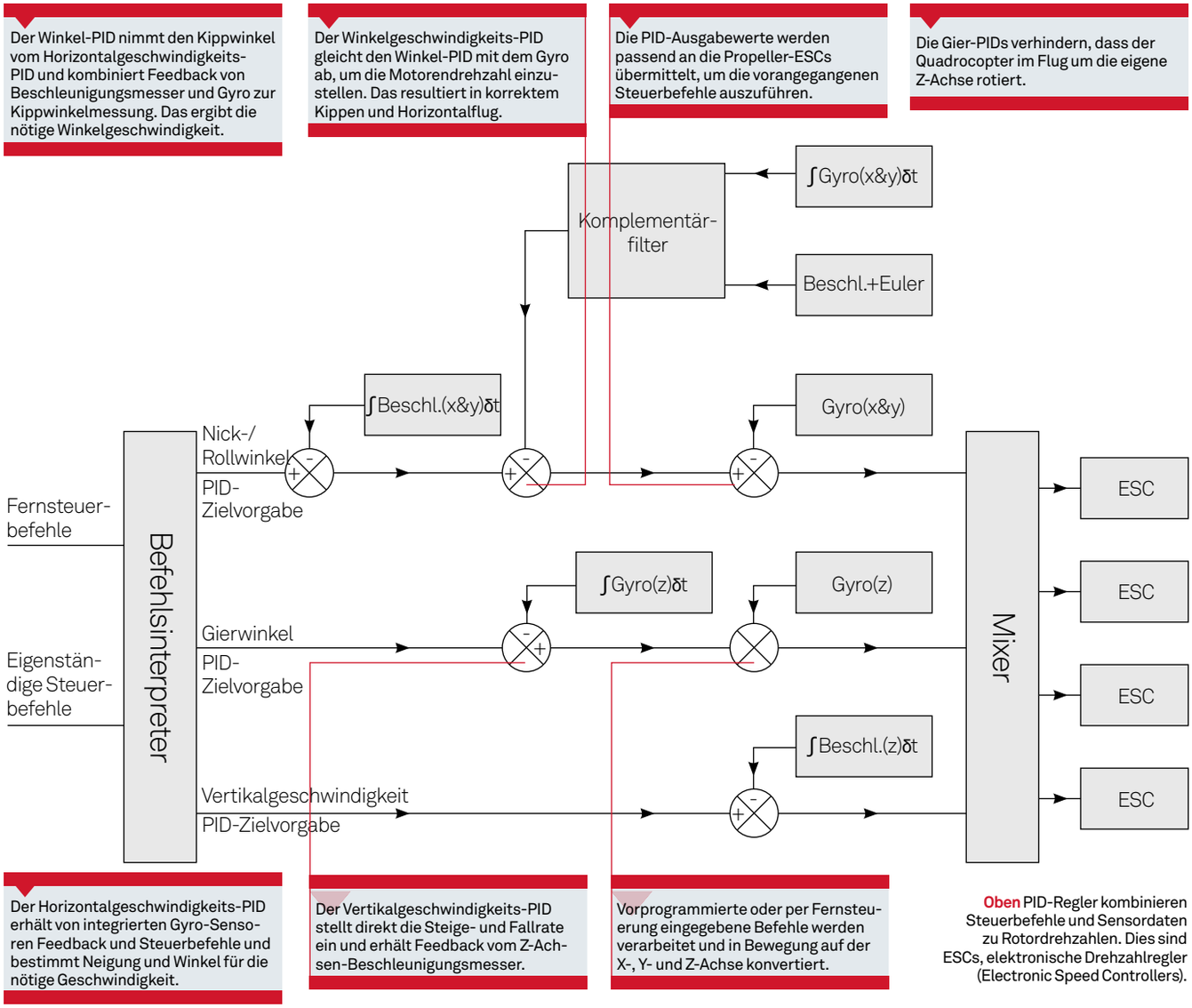
Der Beschleunigungsmesser misst relativ zur Achsrichtung des Quadrocopters, während die Zielvorgaben immer relativ zu den Erdachsen sind – Horizont und Schwerkraft. Um die Ausgabewerte des Sensors mit diesen Achsen abzugleichen, braucht

So funktionieren Quadrocopter

Dieser Artikel konzentriert sich auf Software, aber für den Kontext ist etwas Grundwissen zur Hardware des Bausatzes wichtig.

Ein Quadrocopter hat vier nach oben zeigende Rotoren (daher der Name), die jeweils mit einem bürstenlosen DC-Motor an den Ecken des (meist) quadratischen Rahmens verbunden sind. Zwei Motoren drehen im, zwei gegen den Uhrzeigersinn, um das Impulsmoment des Quadrocopters im Flug zu minimieren. Jeder Motor hat einen eigenen elektronischen Drehzahlregler (ESC). Die Motoren selbst haben drei Spulenpaare (Phasen); die ESCs konvertieren ein pulsweitenmoduliertes (PWM) Software- oder Hardwaresteuersignal in einen dreiphasigen Starkstromimpuls, der die Motoren auf die vorgegebene Geschwindigkeit stellt.

Versorgt werden die ESCs und alle anderen Bauteile von einem Lithium-Polymer-Akku (LiPo) mit einer Nennspannung von 12 V bei 3300 mA und einem Stoßstromgrenzwert von 100 A – hierin liegt die Kraft!



Oben PID-Regler kombinieren Steuerbefehle und Sensordaten zu Rotordrehzahlen. Dies sind ESCs, elektronische Drehzahlregler (Electronic Speed Controllers).

es ein wenig Trigonometrie und Kenntnis der Neigungswinkel der Nick- und Rollachsen des Quadrocopters im Bezug auf die Erde.

Angles

Gyroskop und Beschleunigungsmesser können Winkelinformationen ausgeben, aber beide sind ungenau.

Mit den Daten des Beschleunigungsmessers kann über das Euler-Verfahren der Winkel berechnet werden. Die Messung wird allerdings durch die Motoren/Propeller beeinträchtigt, sodass eine Einzelabfrage extrem ungenau sein kann. Der Langzeit-Messdurchschnitt ist allerdings recht akkurat.

Das Gyroskop wiederum ist nicht störanfällig, misst aber nur die Winkelgeschwindigkeit, die noch mit dem Zeitverlauf integriert werden muss, um den absoluten Winkel des Quadrocopters relativ zum Horizont herauszufinden. Dabei führen Rundungsfehler im Zeitverlauf zu immer größeren Abweichungen, die im Endeffekt auch die maximale Flugzeit beschränken.

Filter

Beide Messwerte sind also problematisch, doch sie können mathematisch kombiniert werden, um ihre Fehler gegenseitig auszugleichen, was eine störungsfreie und akkurate Langzeitmessung ermöglicht. Der beste unter den vielen bekannten Störschutzfiltern ist der Kalman-Filter. Wir verwenden einen etwas weniger genauen, aber dafür leichter verständlichen und programmierbaren: den Komplementärfilter.

Durch genaue Winkelmessung ist es nun möglich, die Daten des Beschleunigungsmessers relativ zu den Erdachsen zu konvertieren, und mit den Zielvorgaben abzugleichen, wie schnell der Quadrocopter sich nach oben, unten, links, rechts, vorn und hinten bewegt.

PIDs

Jetzt haben wir also ein Ziel für den Quadrocopter, Eingaben dafür, wie er es erreichen soll, sowie Motoren, die die Lücke zwischen den beiden schließen. Nun müssen diese Dinge nur noch zusammengebracht werden. Ein konkreter mathematischer Algorithmus ist fast unmöglich – die Gleichung müsste das genaue Gewicht des Quadrocopters, die Rotationskraft jedes Rotors, die Gewichtsverteilung und vieles mehr berücksichtigen. Keiner dieser Faktoren ist stabil: Im Flug (und bei Abstürzen!) werden Rotoren beschädigt, Batterien verrutschen in der Halterung, Gras, Matsch und Nässe verändern das Gewicht des

Beschleunigungsmesser und Gyroskop geben Winkelinformationen aus – beide sind problematisch.

Quadrocopters, Luftfeuchtigkeit und Höhe müssten einbezogen werden. So könnte kein Quadrocopter abheben.

Stattdessen wird eine Schätzmethode eingesetzt, die eine Feinabstimmung mittels Sensordaten erreicht. Der Code alterniert mehr als 100-mal pro Sekunde zwischen Schätzwert und Sensorfeedback und kann so sehr schnell auf „Fehler" reagieren, wenngleich er nichts über all die Faktoren weiß, die er ausgleicht – das Sensorfeedback erledigt all das blindlings. Dies ist ein PID-Algorithmus. Er zieht die Daten des Feedbacks von der Zielvorgabe ab und berechnet so den Fehler. Letzterer wird anschließend durch Proportional-, Integral- und Differentialalgorithmen verarbeitet, die so einen Ausgabewert erzeugen.

Blender

Die Ausgabewerte werden abwechselnd an alle ESCs (elektronische Drehzahlregler) weitergeleitet: Die Vertikalgeschwindigkeit wird gleichmäßig an allen Rotoren geschaltet; der Neigungswert wird 50/50 geteilt, von den vorderen Rotoren abgezogen und zu den hinteren addiert, was den Nickwinkel ergibt. Der Rollwinkel und auch das Gieren funktionieren ähnlich, wobei Letzteres durch Schaltung diagonal entgegengesetzter Rotoren erreicht wird, die in die gleiche Richtung drehen.

Diese ESC-spezifischen Ausgabewerte werden dann pulsweitenmoduliert (PWM) und mit aktualisierten Motorgeschwindigkeiten an die Hardware-ESCs geschickt.

Code und Realität

Der Code enthält insgesamt neun PID-Regler. Der Horizontalgeschwindigkeits-PID konvertiert die gewählte Geschwindigkeit in Horizontalbeschleunigung und Neigungswinkel auf der X- und Y-Achse in Horizontalrichtung; der Winkel-PID rechnet den Neigungswinkel dann in eine Neigungsrate um, die der Drehgeschwindigkeits-PID wiederum in Motorsteuerbefehle für vorn/hinten oder links/rechts konvertiert und so Nicken und Rollen steuert.

In Vertikalrichtung reicht ein einziger PID, der die Steige- und Senkrate für gleichmäßige Beschleunigung an die Motoren leitet.

Unerwünschtes Gieren (denken Sie an einen Kreisel) vermeiden zwei PIDs – einer, um den gewünschten Gierwinkel zu halten, der auf 0 festgelegt ist, ein weiterer, der die Giergeschwindigkeit einstellt. Deren Ausgabewerte steuern die diagonal gegenüberliegenden Motoren, die in die gleiche Richtung drehen.

> Bei Gegenwind wird ein Quadrocopter von der Kraft des Windes zurückgedrängt.

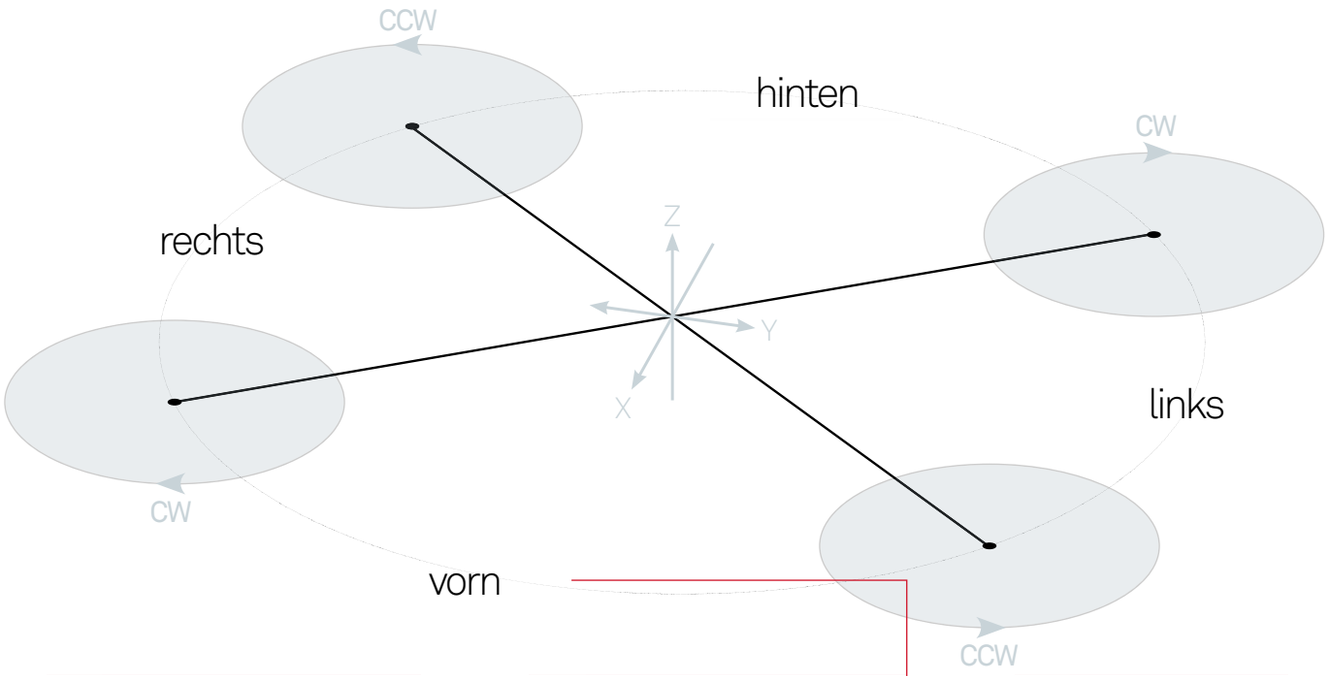
Absolut unverzichtbar unter den neun PIDs ist die Nick-/Roll-/Gier-Stabilisation, die dafür sorgt, dass der Quadrocopter unabhängig von gestellter Aufgabe und externen Bedingungen stabil bleibt. Andernfalls würden die restlichen PIDs nicht mehr funktionieren. Nicken wird durch relative Geschwindigkeitsunterschiede zwischen den vorderen und hinteren Rotoren gesteuert, Rollen analog dazu zwischen links und rechts, Gieren durch die Differenz in den PID-Ausgaben im und gegen den Uhrzeigersinn. Die Rohdaten aller drei PIDs werden dann als individuelle Pulsweiten an die jeweiligen PWM-Kanäle der Motoren gesendet.

Ist alles stabil, kann der Vertikalgeschwindigkeits-PID bereits einfache Start-, Schwebe- und Landemanöver steuern. Stellen Sie den Quadrocopter auf eine ebene Fläche, geben Sie 0,5 m/s als Zielwert ein, und schon hebt er ab, während der Stabilisations-PID die Horizontalorientierung stabilisiert.

Soweit sind die PID-Regler unabhängig voneinander. Nun, was ist mit Horizontalbewegung und Driftkompensation bei Wind? Ein Quadrocopter, der durch Gegenwind zurückgedrängt wird, muss nach vorne nicken, um dem Wind in Horizontalrichtung mehr Motorkraft entgegenzusetzen. Diese fehlt nun natürlich für die Schwebestabilisierung; wenn die Gesamtleistung nicht erhöht wird, beginnt der Quadrocopter zu sinken.

Noch komplexer ist die Horizontalbewegung: Für die Vorwärtsbewegung von 1 m/s muss – analog zur Gegenwindkompensation – durch Nicken und erhöhte Motorleistung beschleunigt werden. Sobald aber die Endgeschwindigkeit erreicht ist, muss der Quadrocopter aufhören, zu beschleunigen, während ihn gleichzeitig der Luftwiderstand ausbremst. Eine dynamische Nickbewegung ist für stabile Geschwindigkeit nötig.

Driftkompensation und kontrollierte Horizontalbewegung nutzen eine Kombination von PIDs; die Ausgabewerte der Horizontalgeschwindigkeits-PIDs der X- und Y-Achse werden als Zielvorgaben für Nick- und Rollwinkel hergenommen. Deren Ausgaben landen wiederum bei den Nick- und Rollgeschwindigkeits-PIDs zur Stabilisation beim Erreichen der Zielwinkel. Das Sensorfeedback senkt dabei die Fehlerzahl beim Erreichen der Horizontalgeschwindigkeit und reduziert die Anzahl der Ziele für den Kippwinkel-PID, was den Quadrocopter horizontal stabilisiert. Darum ist eine akkurate Winkelmessung so wichtig: Das



Rotoren Die Rotoren sind diagonal zur X- und Y-Achse angebracht und vermeiden so bei Rotation Gieren (Rotieren um die Z-Achse).

Ausrichtung Ausrichtung des Quadrocopters nach vorn, hinten, links und rechts, bezogen auf die Anbringung der Rotoren.

Sensoren Die Sensoren übermitteln Daten anhand dieser X-, Y- und Z-Achsen.

Oben Ausrichtung des Quadrocopters im Bezug auf die Flugrichtung, die Drehrichtung der Rotoren und die im Code verwendeten Achsen.

Gegenwind-Beispiel zeigt, wie die Sensoreingaben an den Vertikalgeschwindigkeits-PID durch den Kosinus des gemessenen Kippwinkels relativ zum Horizont abgeglichen werden. Analog dazu benötigen Sensoreingaben an den X- und Y-Geschwindigkeits-PID den Abgleich mit Nick- und Rollwinkel, wenn die Zielgeschwindigkeit in Abhängigkeit vom Beschleunigungsmesser eingestellt wird.

Experimentieren und Verbessern

Im Code spiegelt sich zwar alles hier Beschriebene akkurat wider, aber eine Komponente kann nur unter Einsatzbedingungen erprobt werden: die PID-Verstärkung. Jeder PID-Regler hat unabhängige Proportional-, Integral- und Differential-Verstärkungen, die sich nur durch Schätzen oder Experimentieren ergeben. Jeder Quadrocopter verhält sich hier unterschiedlich, doch zum Glück gibt es eine gute Lösung.

Finden Sie zunächst die PWM-Startgeschwindigkeit heraus, indem Sie den Quadrocopter aufstellen und langsam den PWM-Wert erhöhen, bis er fast abhebt – in unserem Fall war die Pulsweite etwa 1590 µs (oder 1000 µs + 590 µs, wie im Code sichtbar).

Weiter zu den Stabilisations-PIDs. Wenn der Quadrocopter gerade steht und zentral in der Waage ist, dann funktionieren Nick- und Giereinstellung genau gleich. Um das Nicken einzustellen, müssen zwei diagonal gegenüberliegende Motoren ausgeschaltet und auf zwei Kanten abgestellt werden, sodass der Quadrocopter dazwischen hängt. Schalten Sie dann die beiden freien Motoren auf Startgeschwindigkeit (bei uns knapp 1550 µs). Wackelt das Fluggerät wie verrückt umher, fängt es sich gerade noch, richtet es sich wieder auf, wenn es angestoßen wird, oder tut sich gar nichts? Stellen Sie zunächst die P-Verstärkung ein. Sobald diese stimmt, fügen Sie etwas I-Verstärkung hinzu – so garantieren Sie eine Rückkehr zum Wert 0 und Stabilität. D-Verstärkung ist optional, verbessert aber die Handhabung. Ein Quadrocopter mit stabiler D-Verstärkung ist wie ein guter Tisch, er wackelt nicht.

Die Werte für den Vertikalgeschwindigkeits-PID können auch geschätzt werden. 1590 µs bedeuten Abheben; Startgeschwindigkeit soll 0,5 m/s sein, daher ist eine P-Verstärkung von 100 in Ordnung; I- oder D-Verstärkung sind unnötig.

Nun ist sicheres Starten, Schweben und Landen möglich und auch nötig, da nur so die Richtungs-PIDs eingestellt werden können. Hier ist Vorsicht angebracht – zu hohe Verstärkungen resultieren leicht in Kollisionen mit Wänden oder Purzelbäumen mit anschließendem Sturzflug. Große Freiflächen mit Rasen, am besten noch nass vom nächtlichen Regen, empfehlen sich dafür.

An diesem Schritt führt kein Weg vorbei, also rechnen Sie mit Abstürzen und Schäden und genießen Sie das Scharmützel, so gut es geht! Wenn alles nach Plan gelaufen ist, haben Sie jetzt einen Quadrocopter, der bei Wind und Wetter abhebt, schwebt und wieder landet.


USB-Kabel mit Schalter
320 x 240 TFT mit Touchscreen (PiTFT Mini-Kit)
16-Kanal-PWM- und Servomotor-Platine (Adafruit)
PiBorg PicoBorg Reverse
10-bit-Analog-Digital-Wandler (MCP3008)
Raspberry-Pi-Kameramodul
Raspberry Pi Camera
Rev 1.3
Made in the UK
Power

Erweitern Sie den Raspberry Pi

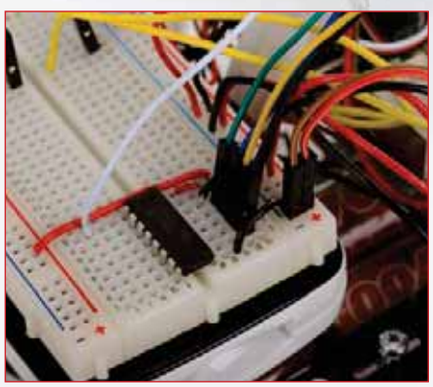
10 äußerst geniale Hardware-Zubehöre

Der Raspberry Pi ist ein fantastisches Produkt britischer Ingenieurskunst. Noch vor fünf Jahren wäre das Konzept eines kreditkartengroßen Computers, der eine komplette Desktop-Umgebung vorhält und fähig zum Abspielen von Full-HD-Videos ist, als Hexerei abgetan worden. Dass das Gerät diese Fähigkeiten zu einem Preis von 35 € bieten würde, hätte das Ganze nicht glaubwürdiger gemacht.

Es ist klar, dass der Raspberry Pi eine Revolution für Rechner mit kleinem Formfaktor darstellt. Dennoch ist er nicht perfekt, schließlich kann nur eine begrenzte Menge an Technik auf eine so kleine Platine gepackt werden. Für Enthusiasten und Bastler gibt es jedoch eine wachsende Menge an Hardware-Erweiterungen, die bei der Realisierung von Projekten, Gadgets und Geräten für das „Internet der Dinge“ helfen. Wir haben zehn der interessantesten Erweiterungen ausgewählt und werden auf den folgenden Seiten erklären, wie Sie sie einsetzen können.

Analoge Eingangssignale verarbeiten

Anders als die meisten Mikrocontroller kann der Raspberry Pi keine analogen Sensoren auslesen. Das Hinzufügen dieser wichtigen Funktion ist aber erstaunlich günstig.



Wie man sieht, verbraucht der MCP3008 sehr wenig Raum auf der Steckplatine.

Was Sie brauchen:

- MCP3008 £2 (ca. 3 €, z.B. conrad.de)
- Steckplatine
- Verbindungskabel (männlich zu weiblich)

Was ist das?

Wenn Sie die Fähigkeit zum Lesen analoger Eingangssignale beim Raspberry Pi nachrüsten möchten, brauchen Sie ein ganz bestimmtes Stück Hardware – einen Analog-Digital-Wandler (ADC). Die einfachste und billigste Art, dieses in den Pi zu integrieren, ist ein ADC-Chip wie der MCP3008, der acht analoge Eingänge auf Kosten von nur vier GPIO-Pins nachrüstet. Mit einer Steckplatine, einigen Drähten und etwas Python-Code lesen Sie Temperaturen oder die Eingangssignale Ihres analogen Joysticks, um damit Ihre eigenen Spiele zu entwerfen.

Wozu brauche ich das?

Die GPIO-Pins sind großartig – sie ermöglichen es Ihnen, mit der realen Welt auf eine tolle Art und Weise zu interagieren, einfach indem Sie die Spannung der einzelnen Pins ein- und ausschalten. Mit einem einzelnen pulsweitenmodulierenden (PWM) Pin bietet der Raspberry Pi eine recht feingranulare Kontrolle über die Helligkeit einer LED oder die Geschwindigkeit eines Gleichspannungs-Motors. Aber für richtige Fans von elektronikbasieren Projekten ist das Messen analoger Eingangsgrößen etwas, das schmerzlich vermisst wird und auf anderen Boards wie dem Arduino zur Verfügung steht. Mit einem ADC können Sie sehr genaues Feedback über die Einstellung eines Potentiometers bekommen (ein drehbarer Knopf, der wie ein Lautstärkeregler aussieht) oder einen Infrarot-Abstandsmesser auslesen, damit Ihr Roboter nicht ungebremst in Hindernisse hineindonnert. Sie können den um einen ADC erweiterten Raspberry Pi auch dafür verwenden, das Umgebungslicht zu überwachen, indem Sie billige lichtabhängige Widerstände (Fotozellen) auslesen. Wenn es dunkel wird, schaltet Ihr Raspberry Pi dann ein Nachtlicht ein, so wie wir das in unserem Beispiel demonstrieren.

▼ Nötige Software installieren

01 Blacklist leeren

Tippen Sie `sudo nano /etc/modprobe.d/raspi-blacklist.conf` in ein Terminal. Fügen Sie am Anfang jeder Zeile ein Rautensymbol (#) ein und drücken Sie anschließend Strg+X und dann Y. Mit Enter speichern Sie die Datei.

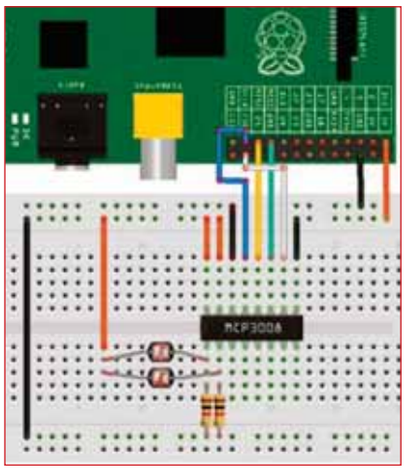
02 Python-Entwicklungswerkzeuge und spidev installieren

Es gibt ein paar Pakete, die wir unbedingt benötigen. Schreiben Sie `sudo apt-get install python-dev python-pip` in ein Terminal. Sobald die Pakete installiert wurden, geben Sie `sudo pip install spidev` und zuletzt `sudo reboot` ein.

03 Sensoren auslesen

Um das Ergebnis irgendeines links am MCP3008 angeschlossenen Sensors auszulesen, müssen Sie nur folgenden Code verwenden. Tippen Sie ihn in Leafpad ab und speichern Sie ihn als adc_test.py. Im Skript lesen wir die ersten beiden Pins aus, wo unsere Fotozellen angeschlossen sind. Zum Testen führen Sie die Datei mit `sudo python adc_test.py` aus.

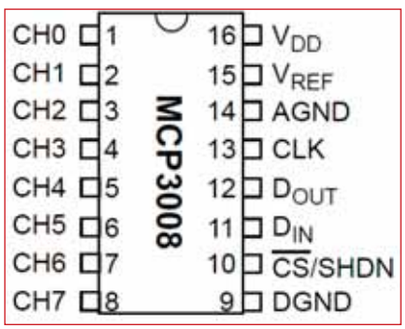
```
import spidev
import RPi.GPIO as GPIO
import time
GPIO.setmode(GPIO.BCM)
spi = spidev.SpiDev()
spi.open(0,0)

def readadc(adcnum):
    if ((adcnum > 7) or (adcnum < 0)):
        return -1
    r = spi.xfer2([1,(8+adcnum)<<4,0])
    adcout = ((r[1]&3) << 8) + r[2]
    return adcout

while True:
    print "Photo Cell 1: " + str(readadc(0))
    print "Photo Cell 2: " + str(readadc(1))
    time.sleep(1)
```



■ Schließen Sie Ihren Chip wie gezeigt an das GPIO-Feld an. Die Stromversorgung ausgenommen gibt es nur vier Verbindungen.

Wie benutze ich es?

Der Analog-Digital-Wandler MCP3008 ist etwas aufwendig einzurichten, sobald das aber erledigt ist, sehr einfach zu verwenden. Dieser kleine Chip kann für nur etwa 3 € beim gut sortierten Elektronikhändler erworben werden. Da Sie vermutlich nicht gleich Ihr erstes Projekt fest verlöten wollen, empfehlen wir die Verwendung einer Steckplatine mit entsprechenden Kabeln, die auf einer Seite mit einem Stecker ausgerüstet sind. Stecken Sie alles zusammen, so wie Sie es auf dem Schema oben sehen. Obwohl der Plan recht kompliziert aussieht, verwenden wir für die Ansteuerung des Chips eigentlich nur vier GPIO-Pins. Zwei weitere Pins dienen der Stromversorgung und werden beim IC rechts oben und unten angeschlossen (der kleine Halbkreis auf dem Chip markiert die Oberseite). Alle Pins auf der linken Seite des Chips sind für analoge Eingänge vorgesehen.



■ Alle Verbindungen auf der Steckplatine mit dem Raspberry Pi finden auf der rechten Seite statt. Die Seite links dient ausschließlich dem Anschluss von Sensoren.

Analoger Entfernungs-sensor

Infrarot-Reflexion zum Erkennen der Umgebung verwenden.

Was Sie brauchen:

IR-Entfernungssensor Sharp GP2D120
(ca. 20 €, z.B. voelkner.de)

Was ist das?

Ein IR-Reflexionssensor wie der Sharp GP2D120 bietet eine großartige Möglichkeit, die Welt zu erfühlen. Er nutzt das Infrarotspektrum, um seine Umgebung zu „pingen“, ähnlich wie Fledermäuse das Echo ihrer Rufe nutzen, um Hindernisse im Dunkeln zu erkennen. Je näher ein Hindernis ist, desto höher ist der vom Sensor ausgelesene Analogwert (zwischen 0 und 1023).

Wozu brauche ich das?

Diese Art von Sensoren ist perfekt für Raspberry-Pi-Projekte, die in irgendeiner Weise das Erkennen von Umgebungen erfordert. Sie können beispielsweise einen Sensor in der Nähe eines Vogelhauses platzieren und die Kamera anweisen, ein Bild zu machen, sobald eine Aktivität im Zielbereich auftritt. Alternativ könnten Sie diverse Sensoren auf einem autonomen Roboter unterbringen – wenn der Sensor links einen hohen Wert ausgibt, ist es logisch, den Roboter nach rechts ausweichen zu lassen.

Wie benutze ich es?

Wir verwenden den Roboter und den Code, den wir für den MCP3008 geschrieben haben, als Beispiel dafür, wie einfach die Eingangssignale gelesen und interpretiert werden können. Der Roboter kann somit intelligent erscheinende Entscheidung treffen:

■ Analoge Infrarot-Entfernungssensoren wie diese sind perfekt für den Bau autonomer Roboter geeignet.

```
import spidev
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BCM)

spi = spidev.SpiDev()

spi.open(0,0)

front = 0

trigger = 450

def readadc(adcnum):
  if ((adcnum > 7) or (adcnum < 0)):
      return -1
  r = spi.xfer2([1,(8+adcnum)<<4,0])
  adcout = ((r[1]&3) << 8) + r[2]
  return adcout

def read_sensor():
  result = readadc(front)
  return result

while True:
  reading1 = read_sensor()
    if reading1 > trigger:
      print “Sensor”, str(reading1),
“triggered: Run!”
    else:
        print "Nothing to see here."
    time.sleep(0.5)
```

Fotos aufnehmen mit dem Pi

Spendieren Sie Ihrem Raspberry Pi die Möglichkeit, zu sehen, und öffnen Sie ihn so für Fotos, Videos und optische Wahrnehmung.

Was Sie brauchen:

Aktuelle Version von Raspbian

Raspberry-Pi-Kameramodul
(ca. 22 €, z.B. reichelt.de)

Was ist das?

Die Raspberry-Pi-Kamera ist eine kleine Platine, auf der eine winzige Kamera mit 1,3 Megapixeln verbaut ist. Diese ist in der Lage, Fotos und Videos zu machen sowie eine Live-Vorschau von dem anzuzeigen, was die Kamera gerade sieht.

Wozu brauche ich das?

Aufgrund seiner Größe und Mobilität ist der Raspberry Pi beinahe so praktisch wie ein modernes Telefon mit Kamera, bietet dank spezialisierter Bibliotheken und Software aber mehr Kontrolle. Bilder könnten zwar auch per Webcam geschossen werden, diese würde jedoch einen USB-Slot verbrauchen. Die Kamera wird an einem speziellen Videoeingang (der bereits für eine etwaige zukünftige Kamera konzipiert wurde) angeschlossen und belegt somit keinen der anderen Stecker.

Wie benutze ich es?

Schalten Sie den Raspberry Pi aus und entfernen Sie die Stromversorgung. Zwischen dem HDMI- und dem Netzwerk-Port liegt ein spezieller Anschluss. Ziehen Sie die Laschen auf beiden Seiten hoch und stecken Sie das Flachbandkabel in die Öffnung, sodass die Kontaktflächen zum HDMI-Anschluss zeigen. Schließen Sie nun vorsichtig die Laschen und verbinden Sie den Raspberry Pi wieder mit dem Strom. Um die Kamera zu verwenden, müssen Sie das Kameramodul im Konfigurationsmenü von Raspbian aktivieren.

▼ Kameramodul aktivieren

01 Raspberry Pi aktualisieren
Stellen Sie sicher, dass alles auf dem aktuellen Stand ist. Starten Sie ein Terminalfenster und aktualisieren Sie das Betriebssystem und die sonstige Software:

```
$ sudo rpi-update
$ sudo apt-get update && sudo apt-get upgrade
```

02 Ab ins Konfigurationsmenü
Sobald die Aktualisierung abgeschlossen ist, können Sie die Konfigurationsdateien updaten. Das verläuft über dasselbe Menü, das Sie schon gesehen haben, nachdem Sie Raspbian installiert haben. Tippen Sie dazu im Terminal:

```
$ raspi-config
```

03 Kameramodul aktivieren
Gehen Sie zur Option „Enable Camera“ und stellen Sie diese auf „Enable“. Wählen Sie dann „Finish“ auf der Hauptseite. Nach dem Login funktioniert die Kamera. Um ein Bild zu machen, geben Sie diesen Befehl ein:

```
$ raspistill -o [imagename].png
```

Es wird eine fünfsekündige Vorschau angezeigt, bevor das Bild gespeichert wird. Videos können wie folgt aufgezeichnet werden:

```
$ raspivid -o [videoname].h264
```

Diese haben eine Länge von fünf Sekunden. Es gibt weitere Parameter, mit denen Sie die Blende, Länge der Vorschau, Bildrate und Auflösung kontrollieren können. Es gibt auch ein Python-Plugin namens „picamera“, mit dem Sie die Kamera aus Skripten heraus ansteuern können.

Schalter nachrüsten

Ein-/ausschalten ohne Ein-/Ausstöpseln

Was Sie brauchen:

USB-Kabel mit Schalter
(ca. 6 €, z.B. pimoroni.com)

Was ist das?

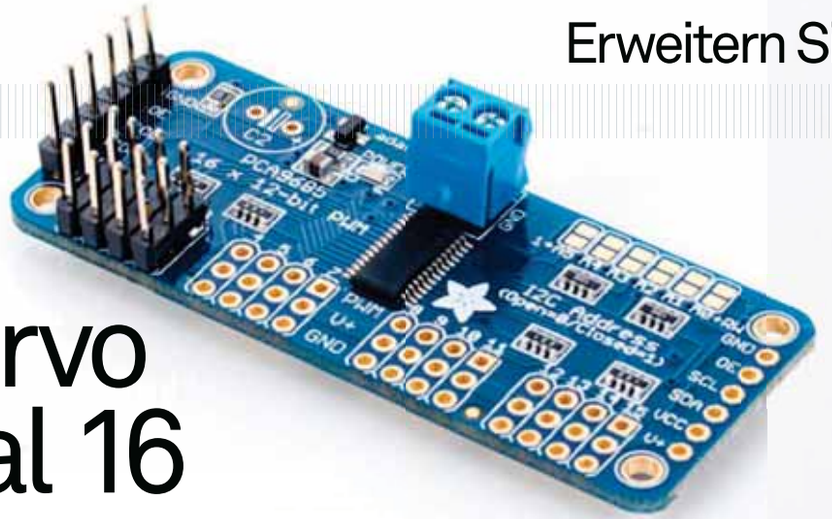
Es handelt sich um einen simplen Schalter mit zwei Positionen, der zwischen einem weiblichen und männlichen USB-Stecker Typ A sitzt: Jener Stecker, der auf fast jedem Kabel zu finden ist. Sie können damit die Stromversorgung ganz einfach trennen und wiederherstellen.

Wozu brauche ich das?

Wie bereits viele Nutzer des Raspberry Pi wissen, muss man, nachdem man den Raspberry Pi heruntergefahren hat, die Stromversorgung trennen und wiederherstellen, bevor das Gerät neu bootet. Dieser Schalter erlaubt es Ihnen, dieses lästige Ritual zu überspringen.

Wie benutze ich es?

Die Hauptkomponente in dieser Erweiterung ist das USB-Kabel mit dem Schalter von Pimoroni. Sie brauchen aber noch ein Mikro-USB-Kabel, um den Schalter zusammen mit dem Raspberry Pi betreiben zu können.



Servo mal 16

Mit nur einem PWM-fähigen Pin ist die Unterstützung für Servomotoren sehr begrenzt.

Was ist das?

Das 16-Kanal-Servo/PWM-Board von Adafruit ist ein beeindruckendes Stück Hardware. Mit diesem sind Sie in der Lage, bis zu 16 kleine Servomotoren über nur zwei Pins zu steuern. Sie können bis zu 62 dieser Boards miteinander verschalten, sodass Sie in der Lage sind, bis zu 992 PWM-Servos oder LED-Lichter gleichzeitig anzusprechen.

Wozu brauche ich das?

PWM (Pulsweitenmodulation) ist eine Herausforderung auf dem Raspberry Pi, da dieser nur einen einzigen dazu fähigen Pin hat. Da der Pi kein Echtzeitgerät wie etwa der Arduino ist, ist die softwaregesteuerte PWM ungenau. Während sie zur Kontrolle von LEDs ausreicht, kann die Steuerung von Servomotoren zu ungewollten Effekten wie Stottern führen.

Was Sie brauchen:

16-Kanal-PWM- und Servomotor-Platine
(ca. 19 €, z.B. pimoroni.com)

Servomotoren
(je ca. 7 €, z.B. conrad.de)

Batteriehalterung
(ca. 3 €, z.B. pimoroni.com)

Wie benutze ich es?

Möchten Sie irgendetwas steuern, das hohe Präzision benötigt, ist eine dedizierte Hardware-Lösung für PWM Pflicht, und genau in diese Bresche springt Adafruits 16-Kanal-PWM-Board. Sobald das Löten erledigt ist, müssen Sie den Raspberry Pi so einrichten, dass er über I2C mit dem Board zusammenarbeitet. Verwenden Sie dazu die Schritt-für-Schritt-Anleitung unten auf dieser Seite.

Wie funktioniert es?

Sobald die Software installiert und das Board angeschlossen und mit einem Batteriepaket versorgt ist, können Sie mit Servomotoren in Ihren Projekten experimentieren. Mit einer Fremdhersteller-Bibliothek programmieren Sie die Servos. Nutzen Sie dazu gegebenenfalls auch die weiterführenden (englischsprachigen) Informationen unter **bit.ly/1nNDaXC.**

Python-Bibliothek verwenden

01 Abhängigkeiten installieren
Um I2C zu installieren, müssen Sie sicherstellen, dass die I2C-Kommunikation nicht blockiert ist. Tippen Sie dazu `sudo nano /etc/modprobe.d/raspi-blacklist.conf` und kommentieren Sie „blacklist i2c-bcm2708“ aus (indem Sie ein „#“ an den Anfang der Zeile stellen). Installieren Sie dann SMBus und die I2C-Tools mithilfe des Kommandos `sudo apt-get install python-smbus i2c-tools` im Terminal.

02 Python-Bibliothek holen
Da git bei der Distribution Raspbian bereits installiert ist, geht es einfach, die offizielle Adafruit-Python-Bibliothek zu installieren. Das geschieht mit folgendem Kommando in einem Terminal: `git clone https://github.com/adafruit/Adafruit-Raspberry-Pi-Python-Code.git`
In den Ordner gelangen Sie mit:
`cd Adafruit-Raspberry-Pi-Python-Code/Adafruit_PWM_Servo_Driver`

03 Dritthersteller-Bibliothek
Obwohl die offizielle Bibliothek schon nicht schlecht ist, bietet sie nur sehr grundlegende Funktionen, welche den Umgang mit Servos erschweren. Glücklicherweise gibt es noch eine kleine Dritthersteller-Bibliothek, die die wichtigen Teile aus der Adafruit-Bibliothek verwendet und auf ihr aufbaut. Sie holen sie sich mit der Eingabe `git clone https://github.com/labatrockwell/raspberrypi-experiments.git` in der Kommandozeile.

Pi mit Ohren

Zeichnen Sie Ton preisgünstig auf.

Was Sie brauchen:

Elektret-Mikrofonverstärker
(ca. 8 €, z.B. exp-tech.de)

Was ist das?

Ein Elektret-Mikrofon ist ein cleveres Stück Technik, das ein permanent geladenes Material nutzt, nämlich – Elektret. So wird die Notwendigkeit einer polarisierenden Energiequelle umgangen. Elektret hat eine eingebaute statische Ladung, die über Hunderte von Jahren nicht abfallen wird – ein sehr elegante Lösung. Das Elektret-Mikrofon von Adafruit (Bild unten) kommt mit einem eingebauten Verstärker und ist bereit, verlötet zu werden.

Wie benutze ich es?

Verbinden Sie es einfach mit der 3,3-Volt-Stromversorgung, Masse und einem analogen Eingang (siehe Seite 168). Schließen Sie den Ausgang des Mikrofons an einem Eingangs-Pin des ADC an, und Sie können das Signal des Mikrofons messen. Wenn Sie es besonders ausgefallen haben möchten, können Sie die Signale, die über das Mikrofon ankommen, dazu nutzen, eine LED anzusteuern und damit die Diode durch Ihre Stimme aufleuchten zu lassen.

Touchscreen passend für den Raspberry Pi

Machen Sie den Pi mit einem Touchscreen wirklich mobil.

Was ist das?

Es handelt sich um einen 2,8 Zoll großen, kapazitiven Touchscreen, der von den Bastel-Gurus bei Adafruit speziell für den Raspberry Pi entworfen wurde. Das Display kann direkt auf den Raspberry Pi aufgesetzt werden und ist etwa so groß wie der Taschencomputer selbst.

Wozu brauche ich das?

Es gibt zahlreiche Gründe, warum Sie so einen Bildschirm am Raspberry Pi angeschlossen haben möchten. Einer der wichtigsten ist, dass der Pi sehr kompakt und mobil ist, ein Monitor in der Regel jedoch nicht. Auch wenn Sie unterwegs den Pi über ein VPN mit einem Smartphone verknüpfen könnten, hat der Touchscreen dagegen den großen Vorteil, dass er direkt am Raspberry Pi anmontiert ist und Sie so jeglichen Problemen mobiler Netzwerke aus dem Weg gehen. Dank der Touch-Funktionalität müssen Sie auch keine weiteren Eingabegeräte mitschleppen.

Das eröffnet eine Vielzahl an Möglichkeiten: Ein tragbarer Computer, ein Kontroll-Pad, eine mobile Filmkamera, all das ist mit dem Pi und dem Touchscreen möglich.

Wie benutze ich es?

Alles, was sie brauchen, um den Bildschirm zum Laufen zu bekommen, ist das LCD selbst. Allerdings müssen Sie etwas löten. Schalten Sie den Raspberry Pi aus und stecken Sie das Display in das GPIO. Stellen Sie sicher, dass der Pi immer noch an einem normalen Bildschirm angeschlossen ist, und fahren Sie ihn wieder hoch. Bleiben Sie nun aber im Kommandozeilenmodus. Noch wird sich der PiTFT nicht einschalten, weil Sie den Raspberry Pi und insbesondere Raspbian noch konfigurieren müssen, um das Display zu unterstützen. Sobald sie eingeloggt sind, laden Sie mit folgenden Befehlen einige Dateien herunter:

■ Die Kamera und der Touchscreen zusammen ermöglichen es mit der richtigen Konfiguration, Bilder mit einer hohen Framerate aufzuzeichnen.

Was Sie brauchen:

320 x 240 TFT 2,8 Zoll mit Touchscreen (PiTFT Mini-Kit)
(ca. 43 €, z.B. flikto.de)

```
$ wget http://adafruit-download.s3.amazonaws.com/libraspberrypi-bin-adafruit.deb
$ wget http://adafruit-download.s3.amazonaws.com/libraspberrypi-dev-adafruit.deb
$ wget http://adafruit-download.s3.amazonaws.com/libraspberrypi-doc-adafruit.deb
$ wget http://adafruit-download.s3.amazonaws.com/libraspberrypi0-adafruit.deb
$ wget http://adafruit-download.s3.amazonaws.com/raspberrypi-bootloader-adafruit-112613.deb
```

Installieren Sie alle mit `sudo dpkg -i -B *.deb` und starten Sie neu. Sobald Sie sich wieder eingeloggt haben, installieren Sie den Treiber des Touchscreens mit:

```
$ sudo modprobe spi-bcm2708
$ sudo modprobe fbtft_device name=adafruitts rotate=90
$ export FRAMEBUFFER=/dev/fb1
$ startx
```

An dieser Stelle sollten Sie ein Bild auf dem Touchscreen sehen. Es verschwindet aber, wenn Sie neu starten. Beenden Sie das Laufende mit Strg+C. Damit der Bildschirm direkt nach dem Start funktioniert, müssen Sie zuerst die Datei /etc/modules öffnen und Folgendes einfügen:

```
spi-bcm2708
fbtft_device
```

Speichern Sie die Datei und öffnen Sie nun die Adafruit-Konfiguration:

```
$ sudo nano /etc/modprobe.d/adafruit.conf
```

Fügen Sie dann folgende Zeile ein:

```
options fbtft_device name=adafruitts rotate=90 frequency=32000000
```

Dies dreht die Bildausgabe um 90 Grad. Das Gleiche müssen wir mit der Touch-Eingabe machen, damit Bild und Eingabemaske deckungsgleich sind. Wir müssen dazu ein Verzeichnis und eine Datei anlegen, die sich darum kümmert:

```
$ sudo mkdir /etc/X11/xorg.conf.d
$ sudo nano /etc/X11/xorg.conf.d/99-
calibration.conf
```

In dieser neuen Datei fügen Sie Folgendes ein:

```
Section "InputClass"
  Identifier "calibration"
  MatchProduct "stmpe-ts"
  Option "Calibration" "3800 200 200 3800"
  Option "SwapAxes" "1"
EndSection
```

Nun sind wir beinahe fertig. Ändern Sie die Datei „~/.profile“ und fügen Sie die Zeile `export FRAMEBUFFER=/dev/fb1` hinzu. Speichern Sie anschließend. Bei jedem Start, bei dem startx ausgeführt wird, wird das Display eingeschaltet. Damit es immer an ist, wenn der Raspberry Pi läuft, installieren Sie:

```
$ sudo apt-get install xserver-xorg-
video-fbdev
```

Anschließend legen Sie eine Datei /usr/share/X11/xorg.conf.d/99-fbdev.conf an und fügen folgende Befehle ein:

```
Section "Device"
  Identifier "myfb"
  Driver "fbdev"
  Option "fbdev" "/dev/fb1"
EndSection
```

Öffnen Sie nun die Raspbian-Konfiguration mit **raspi-config,** aktivieren Sie den Desktop, und bestätigen Sie das mit „Finish“. Nach einem Neustart sollten Sie sich mit einem aktivierten LCD-Bildschirm einloggen können.
Der Bildschirm ist natürlich recht klein und eignet sich vor allem für eigens dafür gestaltete Oberflächen. Das geht mit Python oder jeden anderen Sprache, die Sie bevorzugen. Mit Zusatzsoftware lässt sich der Touchscreen kalibrieren, falls er für Ihre Zwecke nicht ausreichend genau sein sollte.

Das Wetter voraussagen

Verwenden Sie den Raspberry Pi, um sowohl den Luftdruck als auch die Temperatur zu messen.

Was ist das?

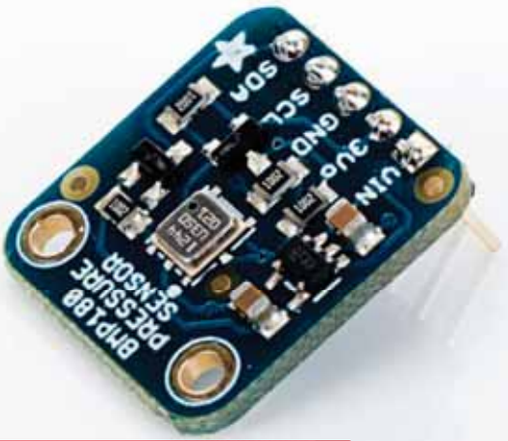
Ein Drucksensor, der es Ihnen ermöglicht, Änderungen im Luftdruck zu messen. Kombiniert mit einem Temperatursensor, kann man damit kurzfristige Änderungen des Wetters voraussagen.

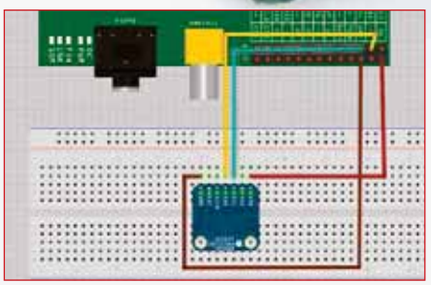
Wozu brauche ich das?

Die Wettervorhersage bringt schon seit Langem naheliegende Vorteile wie etwa das Wissen darum, ob man eine Sonnenbrille oder einen Regenschirm braucht. Sie könnten es aber auch im Bereich der Heimautomation verwenden, etwa um Fenster in Abhängigkeit vom Wetter zu öffnen oder Rollläden zu schließen.

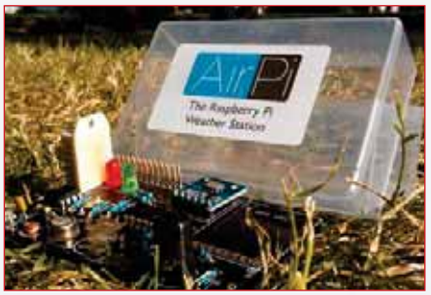
Was Sie brauchen:

Adafruit BMP180
(ca. 11 €, z.B. flikto.de)
Steckplatine
Verbindungsdrähte





■ Wie Sie sehen, lässt sich der BMP180 sehr einfach anschließen.



■ Der „AirPi“ beobachtet nicht nur das Wetter, er misst auch Luftqualität und Verschmutzung.

Einrichtung

01 I2C einschalten

Fügen Sie der modules-Datei im Terminal zwei Zeilen hinzu, um die I2C-Pins zu aktivieren:

```
$ sudo nano /etc/modules
i2c-bcm2708
i2c-dev
```

02 Code holen

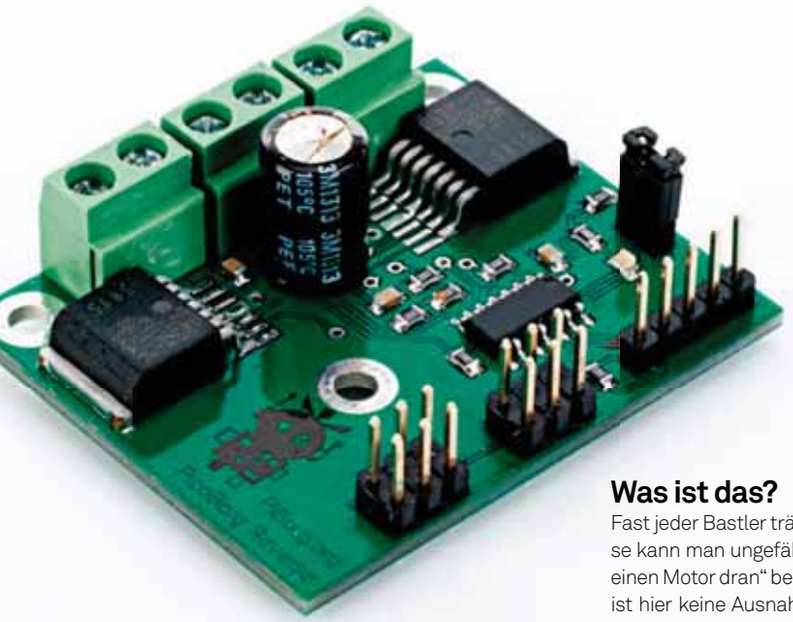
Die Software des BMP finden Sie im Adafruit-Github (github.com/adafruit). Laden Sie sie herunter und navigieren Sie zum richtigen Ordner:

```
$ git clone https://github.com/adafruit/
Adafruit-Raspberry-Pi-Python-Code.git
$ cd Adafruit-Raspberry-Pi-Python-Code
$ cd Adafruit_BMP085
```

03 Sensoren testen

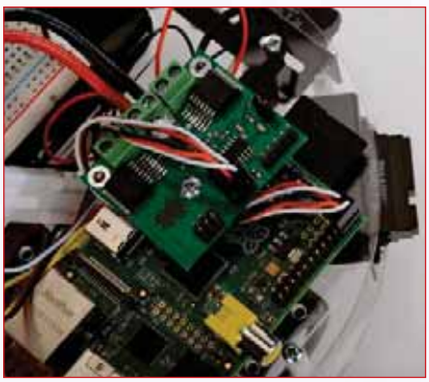
Nun sind Sie fast bereit, erste Daten zu erhalten. Was noch fehlt, ist die Installation eines Pakets. Das erledigen Sie über das Terminal mittels folgend Befehle:

```
$ sudo apt-get install python-smbus
$ sudo python Adafruit_BMP085_example.py
```



(Schritt-) Motoren verwenden

Motoren mit dem Raspberry Pi zu kontrollieren, ist ein Kinderspiel mit PiBorg.



■ Es sind lediglich zwei 3-Pin-Anschlüsse zwischen dem Pi und dem PicoBorg Reverse notwendig.

Was Sie brauchen:

PiBorg PicoBorg Reverse
(ca. 43 €, piborg.org/picoborgrev)

Was ist das?

Fast jeder Bastler trägt eine fixe Idee in sich. Diese kann man ungefähr mit „Wenn es geht, mach einen Motor dran" beschreiben. Der Raspberry Pi ist hier keine Ausnahme, eigentlich hat sich um Motoren für den Raspberry Pi bereits eine eigene Industrie entwickelt, bei der jedes Kickstarter-Projekt seinen Teil vom Kuchen haben will.

Es gibt zahllose Wege, wie Sie Motoren am Raspberry Pi betreiben können, angefangen bei einem simplen IC, dem guten alten L293D, der relativ einfach mit einer Steckplatine genutzt werden kann. Die einfachere Variante ist jedoch eine Erweiterungsplatine, die einen passenden Motor-Antriebs-IC bereits verbaut hat. Aus den Angeboten haben wir das das PicoBorg Reverse ausgesucht. Dieses Board ist so vielseitig, Sie könnten so viele davon zusammenschalten, dass Sie einen Roboter damit bauen könnten, der einen Wohnwagen ziehen kann.

Wozu brauche ich das?

Warum sollte man über den Raspberry Pi Motoren kontrollieren? Es gibt eine riesige Bandbreite von Anwendungsmöglichkeiten, angefangen bei einem motorisierten Kamerawagen, der Ihre Kamera beim Zeitrafferfilmen nach vorne bewegt, bis hin zu einem autonomen und ferngesteuerten Roboter, fähig, selbstständig um Ihr Haus zu preschen. Wir haben das PiBorg Reverse aus mehreren Gründen ausgesucht. Es passt für fast alle gängigen Motorgrößen, nimmt nicht das gesamte GPIO-Feld in Beschlag (ein echtes Problem vieler anderer Lösungen) und bietet eigene Erweiterungspins an (sodass Sie mehrere dieser Boards zusammenschalten und wichtige Pins wie die zur Stromversorgung trotzdem noch erreichen können). Das PiBorg Reverse ist nicht das günstigste Board auf dem Markt, aber es gehört zu den anwendungs- und nutzerfreundlichsten.

Wie benutze ich es?

In Bezug auf die Inbetriebnahme haben sie bei PiBorg reichlich nachgedacht. Das PicoBorg Reverse kommt mit zwei farbkodierten Drei-Pin-Kabeln, die es mit dem GPIO-Feld des Raspberry Pi verbinden. Es enthält überdies ein Montageset, sodass Sie die Platine einfach anschließen können, ohne Zugang zu den Anschlüssen zu verlieren. Mit der montierten Platine und den beiden Kabeln gemäß Anleitung angeschlossen, wechselt die Inbetriebnahme auf den softwareseitigen Teil. Befolgen Sie für die Installation den Schritt-für-Schritt-Wegweiser unten auf dieser Seite.

Die Python-Bibliothek, die mit dem PicoBorg Reverse geliefert wird, ist funktional sehr gut ausgestattet, weshalb es keine Schwierigkeiten geben sollte. Die Bibliothek bietet sogar eine Hilfefunktion an, die alle Möglichkeiten auflistet und kurz erklärt.

▼ Die Software zum PicoBorg Reverse

Wenn Sie dachten, der Einbau der Hardware wäre bereits einfach, lassen Sie sich von der fast vollautomatischen Installation der Software begeistern.

01 Beispiele herunterladen
Erstellen Sie einen Ordner für das Python-Modul und die Beispiel-Skripte. Im Terminal geben Sie dazu folgenden Befehl ein:

```
mkdir ~/picoborgrev
```

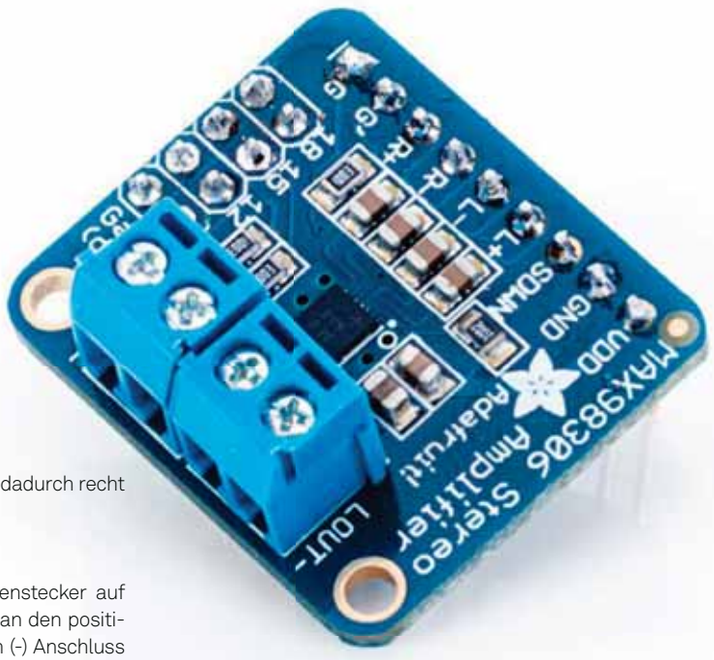
Navigieren Sie dann in das Verzeichnis (cd ~/picoborggrev).

02 Skript ausführen
Das PicoBorg Reverse nutzt I2C zur Kommunikation, aber statt es manuell zu aktivieren, müssen Sie nur den Download auspacken, die Berechtigungen ändern (mit chmod +x install.sh) und das Skript mit „./install.sh" in der Kommandozeile ausführen. Ja, es ist wirklich so einfach.

03 Beispielprogramme ausprobieren
Angenommen, Sie haben Ihre Motoren bereits angeschlossen, sind Sie damit eigentlich schon bereit loszulegen. Sie werden sogar eine neue Desktop-Verknüpfung auf Ihrem Raspberry Pi finden, mit einem kleinen grafischen Programm, das Ihnen beim Start hilft. Sie finden alle Beispielprogramme und Python-Bibliotheksfunktionen unter **piborg.org/picoborgrev/examples.**

Verstärkte Stereoboxen

Auch ohne Kopfhörer hören, was der Raspberry Pi zu melden hat.



■ Sie könnten den Verstärker mit dem Elektret-Mikrofon kombinieren, um einen Lautsprecher zu bauen.

Was Sie brauchen:

Adafruit Stereo-Verstärker MAX98306
(ca. 10 €, z.B. flikto.de)

Was ist das?

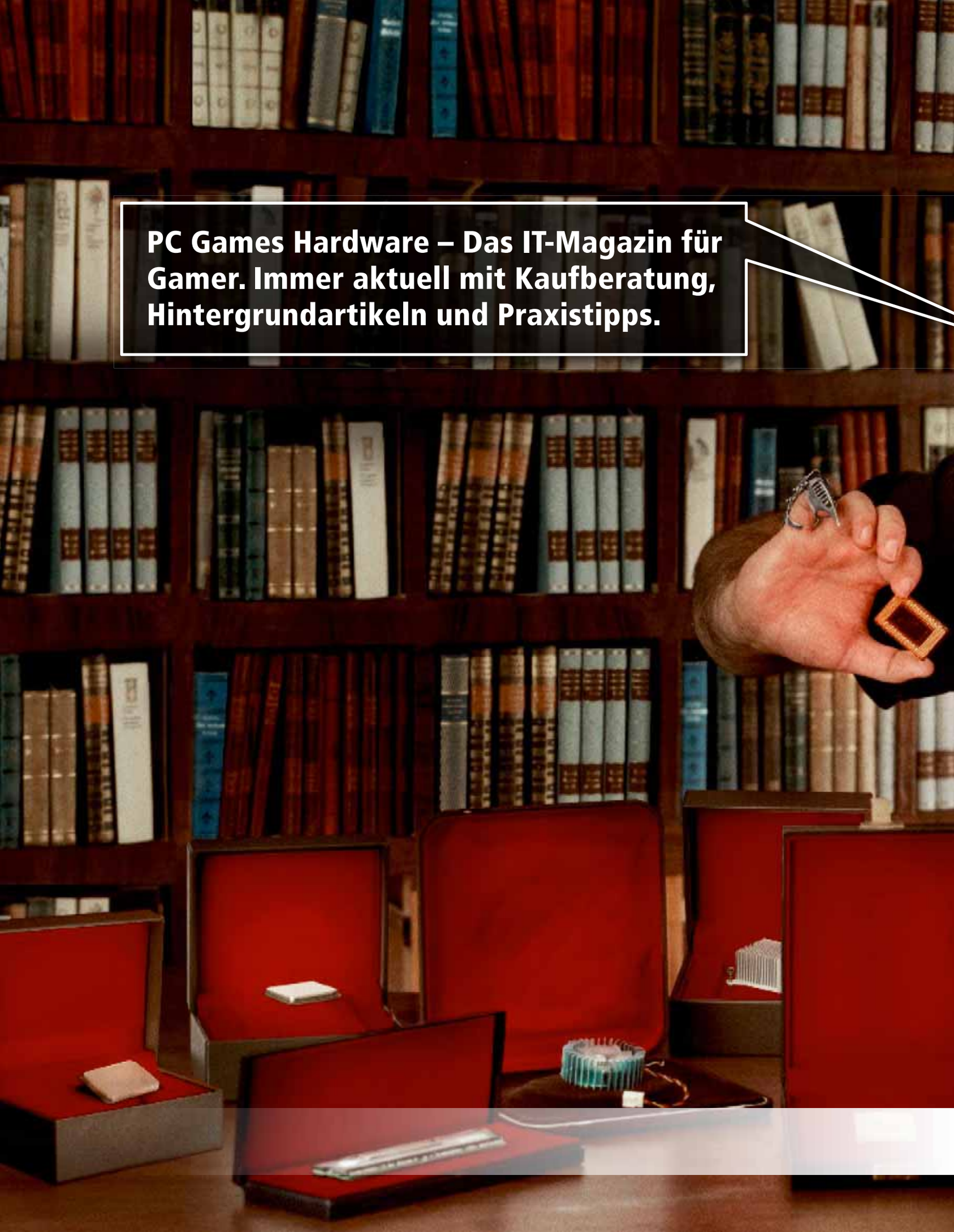
Eine kleine Platine, die auf der einen Seite an den Pi und an der anderen an Stereo-Lautsprecher angeschlossen wird.

Wozu brauche ich das?

Der Raspberry Pi hat keine aufgelöteten Lautsprecher und kann nur über HDMI oder den analogen Audio-Ausgang Ton ausgeben. Dieser ist zudem nicht gut verstärkt und dadurch recht leise.

Wie benutze ich es?

Trennen Sie ein Kabel mit Klinkenstecker auf und schließen Sie ein Drahtende an den positiven (+) und eins an den negativen (-) Anschluss einer der Boxen an. Anschließend machen Sie das Gleiche mit einem weiteren Kabel und der anderen Box. Verbinden Sie die Lautsprecher mit dem Board und versorgen Sie sie über VCC und GND mit Spannung. Suchen Sie sich das Maß der Verstärkung an den entsprechenden Pins aus und stecken Sie abschließend den Klinkenstecker im Raspberry Pi ein.


PC Games Hardware – Das IT-Magazin für Gamer. Immer aktuell mit Kaufberatung, Hintergrundartikeln und Praxistipps.


PC Games
Hardware
www.pcgameshardware.de • www.pcghx.de
DAS HARDWARE-MAGAZIN FÜR PC-SPIELER
HARDCORE
FÜR SCHRAUBER
QR-Code scannen und hinsurfen!
Neue Ausgabe jetzt am Kiosk erhältlich oder einfach online bestellen unter: www.pcgh.de/shop
Auch erhältlich als ePaper bei:
computec MEDIA KIOSK
iKiosk
PresseKatalog Online-Kiosk

SFT MIT FILM-DVD | TV | AUDIO | TABLET | SMARTPHONE | FOTO | INTERNET | WWW.SPIELEFILMETECHNIK.DE

TOP-FILME AUF HEFT-DVD

SFT

SPIELE | FILME | TECHNIK

Rennspaß 2.0!

TEST: Carrera Digital 132

Carreras neue Rennbahn bietet Fahrzeugtuning und Rundenzeitmessung per Smartphone-App.

Seite 59

1. FILM

„Drei Engel für Charlie trifft auf Tarantino“ (Fantasy Film Fest)

2. Film

„Geheimtipp für Genrefans“ (Video.de)

Die besten Tablet-PCs

22 Rechenflundern aller Preis- und Leistungsklassen im großen SFT-Check

Clevere Zeiteisen

TEST Samsungs Gear S und Asus' Zenwatch wetteifern um den Smartwatch-Thron.

Seite 38

Klangkünstler

TEST Die Multiroom-Systeme von Harman Kardon, Geneva und Denon wollen sowohl optisch als auch klanglich begeistern.

Seite 44

TEST: MECHANISCHE TASTATUREN

Was steckt hinter dem Comeback der mechanischen Keyboards? Sechs Tippbretter im Test.

Seite 48

WEITERE TECHNIK-HIGHLIGHTS IM TEST

- Teufel Massive
- New Nintendo 3DS
- Convertible-Notebooks
- Panasonic 55AXW904
- Samsung SSD T1

FSK ab 16 freigegeben

124. AUSGABE
03/15 | März
€ 4,99
Erhältlich auch ohne DVD für € 3,50

DAS TEST-MAGAZIN
FÜR DIGITALE UNTERHALTUNG

DAS BESTE AUS ALLEN TECHNIK-WELTEN:

Flat-TVs | Smartphones | Tablets | Notebooks | Digitalkameras | Video | HiFi

plus: **DIE COOLSTEN SPIELE** und **ALLE FILM-BLOCKBUSTER**

2 TOP-MOVIES AUF HEFT-DVD

Auch als Magazin-Variante ohne DVD für € 3,50

QR-Code scannen und hinsurfen!

WWW.SPIELEFILMETECHNIK.DE

Neue Ausgabe jetzt am Kiosk erhältlich oder einfach online bestellen unter: shop.spielefilmetechnik.de

Auch erhältlich als ePaper bei:

computec MEDIA KIOSK | iKiosk | PresseKatalog Online-Kiosk

Raspberry Pi

PROJEKTE

Der vollwertige PC für 35 € im Kreditkartenformat

Vielseitige Apparaturen

Geräte, Maschinen und Equipment – der Raspberry Pi als Heimtechnikwunder

Faszinierende Story

Wie es dazu kam, dass ein schmaler Computer die weite Welt erobern konnte

Zubehör: Software

Fast alles, was man so braucht, hat schon jemand programmiert...

Werden Sie zum Hacker

...und was fehlt, programmiert man einfach selbst – mit Python

Zubehör: Hardware

Shoppen im Elektronikladen – ja, ich will! Kabel und Gehäuse

Kein Traum zu fern

Ihre Idee ist verrückt genug, dass sie funktionieren könnte? Ans Werk!