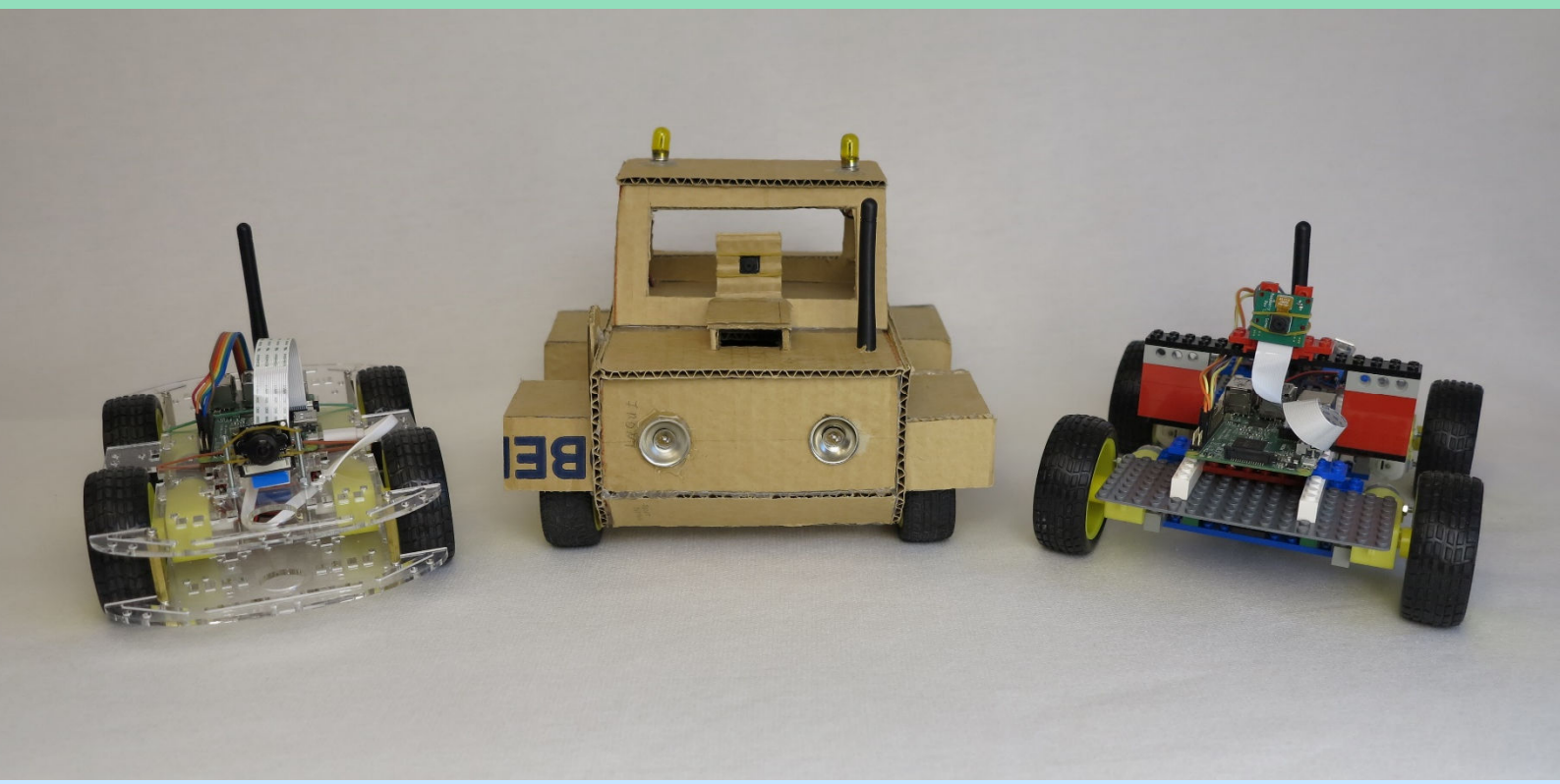
Ingmar Stapel

HOW TO

build a remote controlled car with a Raspberry Pi



Step by step manual to build a remote controlled car with live video streaming

- Learn the basics about **robotics** and the **Raspberry Pi**
- Understand **hardware** and **electronic components**
- Introduction to software programming with **Scratch und Python** to control your remote controlled car

Second edition

Ingmar Stapel studied computer science and is an IT project manager in the automotive and finance industry. He enjoys blogging about current technical trends and his travel experiences. Since 2013 he has been interested in robotics and single board computers and he shares his knowledge and experience at maker fairs and robotic meetups. This eBook is a result of his latest project, a remote controlled car using a Raspberry Pi.

I want to thank my wife Kathrin Stapel for always supporting me during my projects.
Thank you for tolerating the fact that our living room looks like a small workshop from time to time.

Author: Ingmar Stapel

Website: http://custom-build-robots.com
E-Mail: book@custom-build-robots.com

Illustration: Ingmar Stapel
Editorial: Joachim Stapel

Second edition, August 2016.

All rights reserved. No part of this book may be reproduced or transmitted in any form by means, electronic or mechanical, including photocopying, recording, or by any information storage or retrieval system, without the prior written permission of the copyright owner and the publisher.

Where trademarked names appear in this book, we use the names in an editorial fashion and to the benefit of the trademark owner, with no intention of infringement of trademark. This book is distributed globally.

The information in this book is distributed on an “as is” basis, without warranty. Although every precaution has been taken in the preparation of this book, neither the authors or its publisher shall have any liability to any person or entity with respect to any loss or damage caused or alleged to be caused directly or indirectly by the information in this book

All trademarks are property of their respective owners.

Contents

List of abbreviations

CSI	Camera Serial Interface
GPIO	General Purpose Input/Output
MAC	Media-Access-Control
PWM	Pulse Width Modulation
SID	Security Identifier
SSH	Secure Shell (network protocol)
SVN	Subversion
VNC	Virtual Network Computing

1 Introduction

Single board computers like the Raspberry Pi[1] are getting more and more popular and have a lot of supporters who create various projects with them. There is a lot of detailed information about single board computers available in books, websites and online forums. However it is hard to find a step by step manual to accomplish a concrete project. With the following eBook anyone who is interested in computers and robotics will be able to build a remote controlled car with the Raspberry Pi as it guides the maker step by step through each project phase with detailed descriptions, figures and software source code.

The Raspberry Pi was chosen for this project as it has a worldwide active community. This community helps in various questions and is a source for new ideas. The single board computers of the Raspberry Pi Foundation are used in schools and universities and are a good basis for do-it-yourself projects like a remote controlled car.

With this manual you can create step by step, your unique remote controlled car. The interaction of hardware, electronics and software are explained with the help of concrete examples. This makes it easy to understand the more complex parts of the project.

- This eBook starts with the introduction of all necessary electronic components, like the Raspberry Pi and the motor driver, that are necessary to build the remote controlled car.
- Three different body variants, one with LEGO®[2] bricks, another with cardboard and a third with a smart car kit are described.
- The basics of electricity are explained in order to understand the electrical power supply.
- The wiring of all components is illustrated in numerus pictures and illustrations.
- All necessary software as well as the installation and configuration of the software is explained.
- Source code examples for the control program are described on the basis of Scratch and Python and are also available for download in order to control the remote controlled car.
- The Raspberry Pi camera and the live video streaming via WiFi is described in detail.
- To control the remote controlled car within a Web interface, the setup of WebIOPi and services is explained.

[1] Raspberry Pi is a registered trademark of the Raspberry Pi Foundation http://www.raspberry-pi.org.
[2] LEGO® is a trademark of the LEGO Group of companies which does not sponsor, authorize or endorse this book.

For whom is the eBook written?

This eBook is for all students and adults who are interested in electronics and computer hardware and wish to accomplish a project like a remote controlled car with the Raspberry Pi by themselves. Basic knowledge in programming is not necessarily required.

Estimated time:

The described remote controlled car project is intended to be accomplished within one weekend.

2 Electronic components

All necessary components to build a remote controlled car are shown in the following table. The main components such as the Raspberry Pi, the motor driver and the gear motors are explained in detail in the following chapters.

Component	Description
Raspberry Pi 2 Model B	Single board computer as the central processing unit
Raspberry Pi Camera	Enables live video streaming
SD-Card (min. 8 GB)	Storage for programs and operating system
USB WiFi module	Enables remote controlled steering
Gear motors	Propulsion of the remote controlled car
Motor driver	Controls speed and rotation direction of the motors
6x AA batteries	Power supply
Optional: Power supply unit	External power supply for stationary operations
Step-Down Converter	Generates 5V DC
Cable	Female to Female Jumper, 2-core wire, Micro-USB cable

Table 1 Overview electronic components

The following website shows supply possibilities for all named components:

Website: http://custom-build-robots.com/robot-car-components

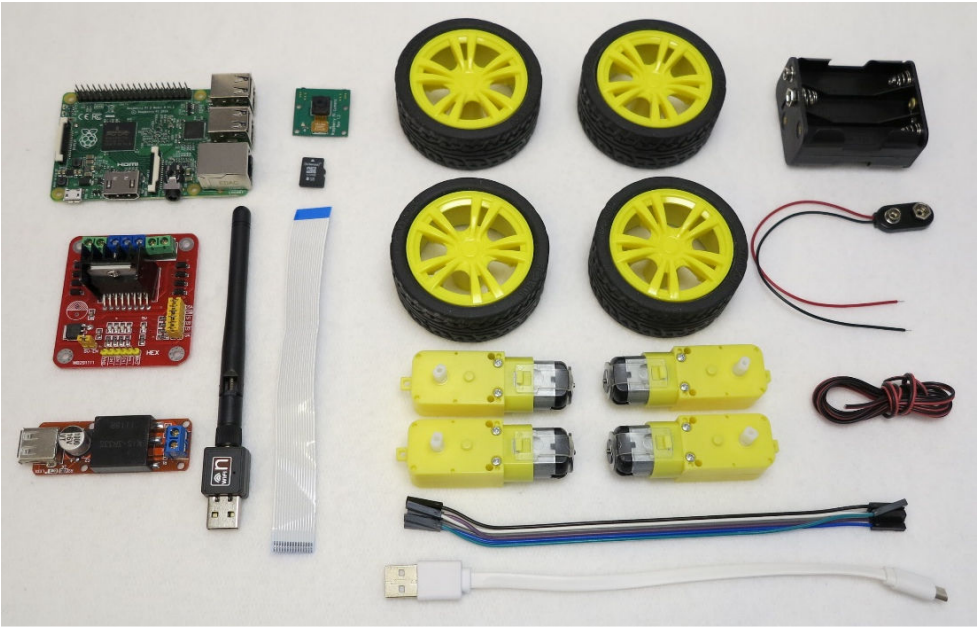


Figure 1 Image showing electronic components for remote controlled car

2.1 Raspberry Pi Single Board Computer

A single board computer is a small but complete and fully functional computer that has all necessary components on a single circuit board. Most of the single board computers use a Linux based operating system and have various input and output options. A major advantage is that these small computers can be operated with batteries as they use low power components.

The Raspberry Pi is developed by the Raspberry Pi Foundation. The idea is to offer a low cost computer that, for example, can be easily used in schools and universities. For this specific remote controlled car project all Raspberry Pi models can be used. Generally speaking, other single board computers like the Odroid C1+ from Hardkernel or a model from the Arduino family can be used.

However, I recommend using the Raspberry Pi 2 Model B as it offers a high performance that is necessary for live video streaming. Additionally the Raspberry Pi community provides a broad knowledge base and is a source for new ideas. For ease of reading Raspberry Pi is used instead of Raspberry Pi 2 Model B in the following eBook.

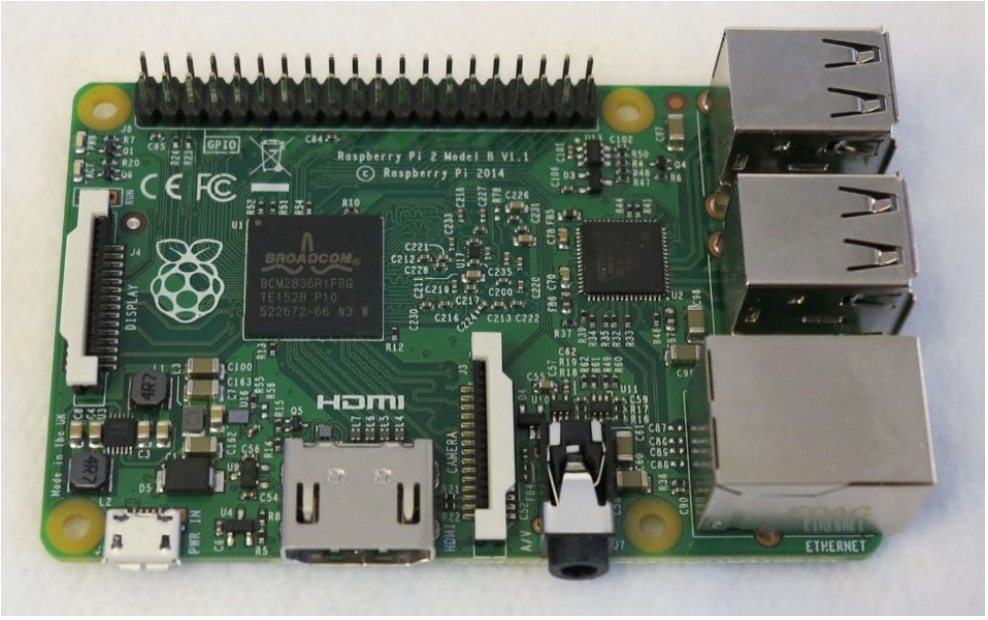


Figure 2 Raspberry Pi 2 Model B

Please find further information about the Raspberry Pi on the Website of the Raspberry Pi foundation.

Website: http://www.raspberrypi.org/

The picture below shows one gear motor with a mounted wheel.

Figure 7 Gear motor with mounted wheel

2.5 Motor driver

With a single board computer it is not possible to directly connect a gear motor, as the outputs are too weak to deliver the required power. To supply the gear motors with energy and to be able to control their speed and rotating direction a motor driver is necessary. Therefore a cost-efficient motor driver that delivers enough power for all four gear motors is chosen for this project. An ideal motor driver is the l298n dual h-bridge as two motors can be controlled separately. Two gear motors for each side of the remote controlled car are connected parallel to the motor driver. Hence the gear motors from one side of the remote controlled car both have the same speed and rotating direction.

The motor driver provides a maximum current of 2A at a maximum voltage of 24V for each channel. This is enough power to accelerate the gear motors smoothly.

The rotation speed of the gear motors can be controlled with the PWM signal of the l298n motor driver. The ability to control the rotation speed of the gear motors is important for cornering, to turn on the spot and the uniform acceleration and deceleration.

Most of the l298n h-bridge motor drivers expect a voltage of 5V at the control input. However the majority of the single board computers are 3.3V modules. That is why it is important to select a motor driver that has a 3.3V control input in order to process the signals from the Raspberry Pi. The motor driver as shown below fulfils this requirement and it can be directly connected to the Raspberry Pi.

On the figure below the l298n h-bridge motor driver is shown. The six yellow control inputs to connect the Raspberry Pi logically are visible in the bottom middle of the picture.

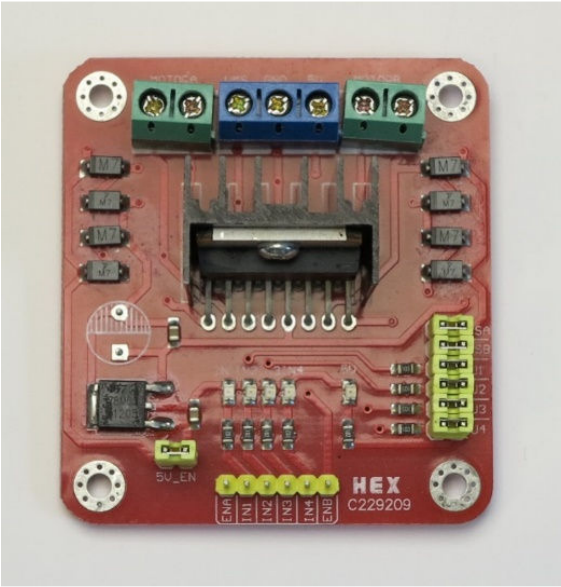


Figure 8 Motor driver l298n h-bridge

2.6 Battery

The used components in the remote controlled car require different voltages in order to work properly. The Raspberry Pi needs 5V supply voltage at 1.2A current. The gear motors need a supply voltage between 3V and 6V but much less current than the Raspberry Pi. If you want to equip your car with LEDs, supply voltages below 3V are required. After several tests I recommend using one power supply source, for example 6 to 8 AA batteries in one battery holder.
A typical AA rechargeable battery has a voltage of 1.2V at 1800mA. Using six of them a voltage supply of 7.2V at 1800mA can be ensured for the remote controlled car.

The picture below shows the battery holder with six AA batteries and the battery clip is shown.

Figure 9 Battery holder with clip

4 Body construction

I started to build a body for the remote controlled car out of cardboard, as this material is available in every supermarket for free. The experience showed that it is very time consuming and takes several attempts until a complete body can be realized. A very comfortable and easy possibility is to build a body out of LEGO® bricks. A further very simple solution is to buy a smart car kit with a body out of acrylic. The following chapter describes three different variants to build a body:

- Body with LEGO® bricks
- Body with acrylic smart car kit
- Body with cardboard

4.1 LEGO® bricks body

To build a body with LEGO® bricks no special tools are required and anyone can build their own individual designed car.

Estimated time:

It takes approximately 20 minutes to build a body with LEGO® bricks.

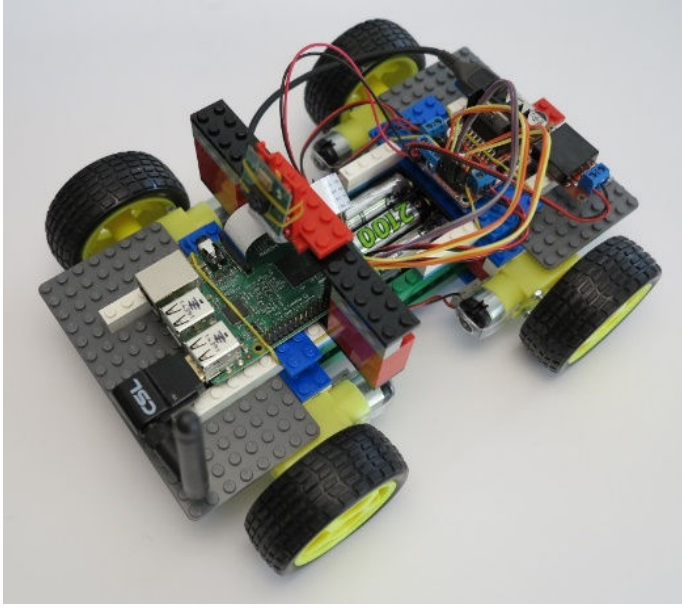


Figure 13 Body build with LEGO® bricks

The specific bricks to fix the gear motors on the body can be ordered in specialised online shops. The focus of this manual is to show one possibility of how to fix the gear motors. The remaining body can be built very individually with available LEGO® bricks.

6 Wiring of the electronic components

The following chapter describes how all electronic components of the remote controlled car need to be connected. For communication between the Raspberry Pi and the motor driver 6 of 40 pins of the Raspberry Pi are needed.

Estimated time:

The wiring with small soldering work requires approximately 90 minutes.

The picture below shows the Raspberry Pi with the multi-pin connector.

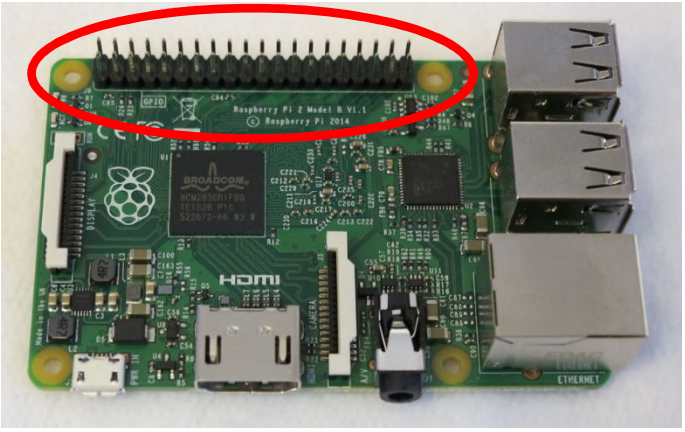


Figure 31 Raspberry Pi with multi-pin connector

For the initial operation and the first tests I recommend using a power supply unit in order to be independent of the limited capacity of rechargeable batteries. This power supply unit with 7V-9V DC and a minimum of 2A needs to be connected to the motor driver that supplies all other components with energy. The power supply shown in the picture below has a switch for selecting different voltages.

Figure 32 Power supply unit

The power supply unit is best used instead of the batteries. However, if no power supply unit is available you can also use rechargeable batteries.

In the following chapter the setup of the battery operated remote controlled car is described.

Required components

- 1 x Raspberry Pi
- 4 x Gear motors
- 1 x micro-USB cable
- 1 x L298N H-bridge
- 1 x Step-Down Converter
- 5 x 20cm 2-core wire
- 1 x Battery holder
- 1 x Battery clip with 2-core wire
- 6 x Female to Female Jumper

The wiring of the components is divided into 4 chapters

- Connecting the electronic components to the power supply
- Connecting the motor driver with the Raspberry Pi
- Wiring of the gear motors
- Connecting the gear motors with the motor driver

6.1 Wiring of the electronic components

The battery supplies the motor driver and the step-down converter with power. The motor driver has three blue lustre terminals that are labelled VMS, GND and 5V respectively. The abbreviations mean:

- VMS = power supply
- GND = ground
- 5V = +5V out (but not stable enough to supply the Raspberry Pi with power)

The inputs VMS and GND are the inputs for the power supply. In both lustre terminals of the motor driver one of the 20cm long 2-core wires and the 2-core wire of the battery clip are connected. Please be aware that VMS is positive and GND (ground) is negative.

The picture below shows the step-down converter and battery both connected to the motor driver.

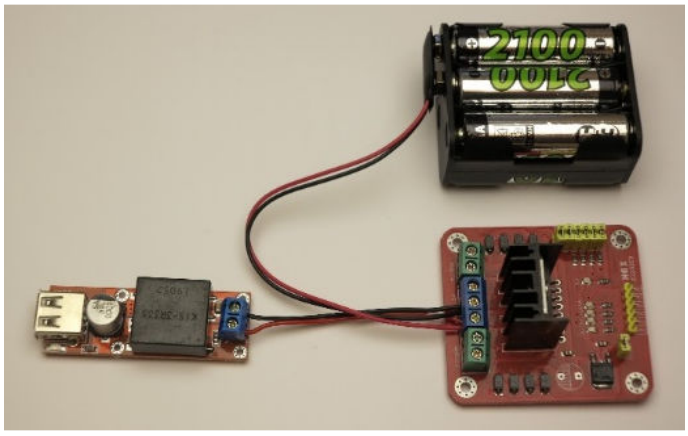


Figure 33 Power supply for the motor driver and step-down converter

The Raspberry Pi is connected with a micro-USB cable to the step-down converter. As there is no SD card with the required operating system in the Raspberry Pi will be unresponsive with the power supply switched on (details for installing the operating system are described in chapter 7).

The picture below shows the Raspberry Pi with the power supply from the step-down converter.

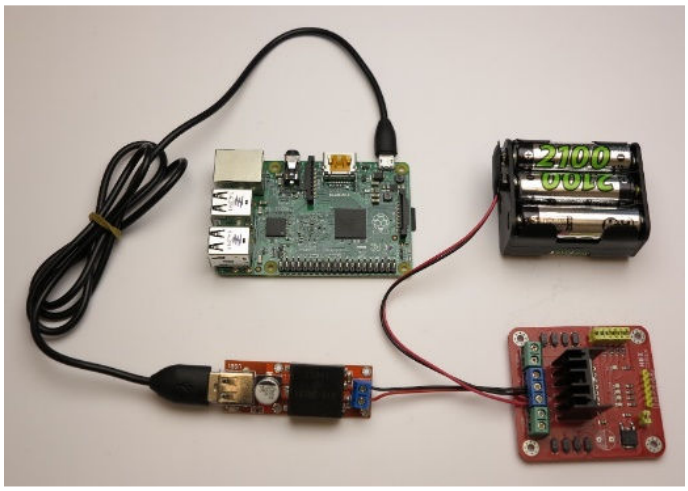


Figure 34 Raspberry Pi power supply

With cabling complete and tested of the power supply of the electronic components, the Raspberry Pi needs to be logically linked to the motor driver.

6.2 Connecting the motor driver with the Raspberry Pi

Female to Female jumpers are special cables required to link the Raspberry Pi with the motor driver. These cables can be easily plugged to the pins of the Raspberry Pi and the motor driver.
Several of the 40 pins of the Raspberry Pi are called general purpose input/output (GPIO) pins (see Table 7). Six of these are needed in the course of this project.

7 Software installation and configuration

This chapter describes which software must be installed on the Raspberry Pi and on a Windows PC. This chapter is based on a Windows operating system. All software is also available for IOS systems. As the remote controlled car is mobile and is not connected to a monitor or a keyboard it will be configured in a way that it is remotely accessible for programing, configuration and controlling. Remote means that the Raspberry Pi will be accessible via WiFi.
The following table lists the programs which need to be installed.

What	Where	Description
Raspbian	Raspberry Pi	Raspberry Pi operation system incl. programs
Midnight Commander	Raspberry Pi	File explorer for the terminal window
TightVNC Server	Raspberry Pi	Remote Desktop Server
TightVNC Viewer	Windows PC	Remote Desktop Viewer
Samba Server	Raspberry Pi	Network share
PuTTY	Windows PC	Access over the remote shell
Python	Raspberry Pi	Programing language
WiFi	Raspberry Pi	Network access over the WiFi
Notepad++	Windows PC	Extended Windows based text editor
Subversion	Raspberry Pi	Central software versioning
Mjpg-streamer	Raspberry Pi	Live video streaming

Table 9 Software required

The following chapter describes in detail the software installation and configuration.

7.1 Prepare the SD-card with the operation system

The Raspberry Pi operation system used by this project is based on Debian, which is a Linux distribution and is published by the Raspberry Pi Foundation with the name Raspbian (Raspberry Pi + Debian = Raspbian).
Firstly the SD-card is configured in a way that the Raspberry Pi is able to boot. To prepare the SD-card an image of Raspbian is needed. Different image files are available for download at the official RaspberryPi.org homepage. Please download the latest version of the Raspbian image (full version):

Website: http://www.raspberrypi.org/downloads/

After the image file is downloaded as a zip file please extract the image. The next step is to install the extracted image on the SD-card. To install the image on the SD-card different tools are available.
Below is a guide describing how to install the image on the SD-card:

8 Programming with Scratch

Scratch[5] is a programming environment which was created and is further developed by the MIT-Scratch-team especially for children and young people. In the Scratch programming environment programs are developed visually by moving blocks into the scripting area. The aim is to introduce programming to children and young people in a very simple and self-explanatory environment.
The Raspberry Pi Foundation has included Scratch within its operating system Raspbian since September 2015 making Scratch easily available for the Raspberry Pi user. For the remote controlled car project Scratch is interesting because it is possible to control GPIOs and use them as PWM out. In this way it is possible to control the motor speed by the PWM signal of the remote controlled car.

The Scratch programming environment is available over the graphical menu on your Raspberry Pi by clicking on “Menu -> Programming –> Scratch”.

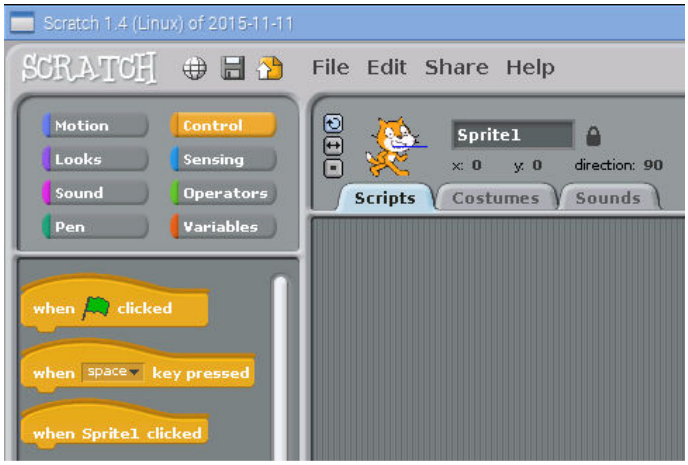


Figure 53: Scratch developing environment

8.1 A basic Scratch program example

Programming in Scratch starts with the initialization of the GPIO pins. In this example the GPIO pin 4 is configured as „out“/ „PWM“. This creates a PWM signal on pin 4 and with this signal a motor driver is able to control the speed of the connected dc motor. With the two GPIO pins 22 and 27 the spinning direction of the motor is set by the motor driver.
In the following list the Scratch blocks which are used for the introduction program are explained to understand the basic functionalities:

1. With the menu item “Control” (yellow Button) the function “When clicked” with the green flag is selected. This initial function is used to start the program and must be clicked with the left mouse button to start the program.
2. Next the function “broadcast” is added. This function has the task of starting the GPIO-server. Enter the command “gpioserveron” in the control button.

[5] Additional information about Scratch is available on the following website: https://scratch.mit.edu

3. A “broadcast” functions follows which configures the GPIO pin 22 as “out”.
4. An additional “broadcast” function follows which configures the GPIO pin 27 as “out”
5. With the next “broadcast” function you set the GPIO pin 22 as „on“.
6. Add a further “broadcast” function and set the GPIO pin 27 to “off”.
7. Another “broadcast” function assigned the GPIO pin 4 with the command “config4outputpwm” as “PWM out”.
8. The next broadcast function is a combination of “broadcast” and “join”. The join function is located in the Scratch menu with the name “Operators”. These combined functions control the speed of the motor. In this example a PWM signal of 50% is created to set 50% of the available power of the motor.
9. With the “wait” function the PWM signal is created for 2 seconds on pin 4. The result is a spinning motor for 2 seconds.
10. With the last “broadcast” function the PWM signal on pin 4 is set to 0%. The result is that the motor stops spinning.
11. Finally the function “wait” causes the program to end after 2 seconds.

Below you see the above described program with the eleven steps:

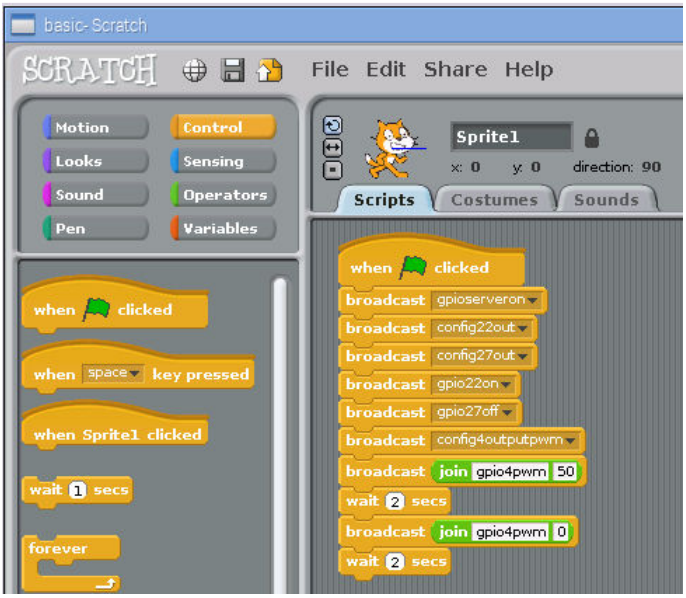


Figure 54 Scratch example

This simple program example shows how it is possible to program a motor to spin for 2 seconds with the available Scratch functions. In the following section a more complex program is described to control the remote controlled car. The basic Scratch program “basic.sb” can be downloaded here:

Website: http://custom-build-robots.com/ebook-download

9 Programming with Python

This chapter explains the programming of the Python program to control the remote controlled car. The aim of this chapter is to write your own Python program to control the remote controlled car with the keyboard over the terminal window. With this understanding, additional projects aside from the remote controlled car may be created such as controlling a remote controlled boat or a flashing LED.
In the next chapter it is described how to control the GPIOs with a Python program. In chapter 12 a program is explained how to control the remote controlled car using a web interface and how to program the web interface.

Check the technical requirements

The first precondition to have all software installed as described in chapter 7.

9.1 Controlling program in Python

Create a new folder in the home directory "/home/pi/" of the user pi with the name „robot".Add the correct permissions in the terminal window to this folder to ensure that the folder is accessible over the Samba network share. The command to change the permissions of the folder is shown as follows.

Command:

```
sudo chmod a+rw robot
```

After executing the command the folder is accessible over the Samba share from the Windows PC and both programs „RobotControl.py" and „L298NHBridge.py" can be saved there.
The two Python programs are created with the text editor Notepad++. The program file for controlling the remote controlled car with the keyboard is called "RobotControl.py". The second program file for controlling the motor driver is called "L298NHBridge.py".

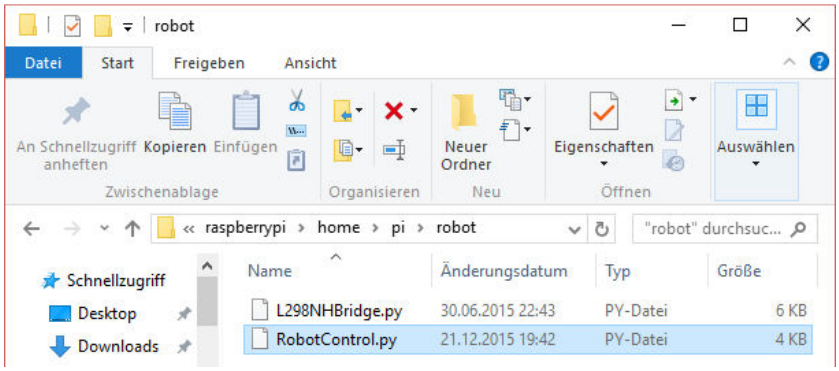


Figure 56: RobotControl.py Python program

9.1.1 Python program – RobotControl.py

With the Python program “RobotControl.py” the remote controlled car can be controlled, either by the terminal window, or by PuTTY. After executing the program a small menu shows up which explains the keyboard layout and how to use the program.
Copy the grey marked source code of the program showing below into the clipboard and paste it into the Notepad++ window. Save the program as “RobotControl.py” in the folder “/home/pi/robot” on your Raspberry Pi.

Inside the source code some inline comments are included which explain the parts, functions and configuration parameters of the program.

```
#!/usr/bin/env python
# coding: latin-1
# Autor:Ingmar Stapel
# Datum:        20160731
# Version:      2.0
# Homepage:     http://custom-build-robots.com
# This program is on of the control programs for the robot car.
# With this program the robot car is controlled via the keyboard
# and terminal window.

# Some Python classes are imported to use the functions implemented
# by these programs. The functions offered by these Python classes are
# needed inside the robot control Python program.
import sys, tty, termios, os, readchar

# The program L298NHBridge.py is imported as a module. The module
# enables with his functions the controlling of the motor driver.
import L298NHBridge as HBridge

# The two variables speedleft and speedright are controlling the
# speed of the left and right motor of the robot car.
speedleft = 0
speedright = 0

# This is the menu shown to the user when he started the program.
# The menu explains which keys are used to control the robot car.
print("w/s: direction")
print("a/d: steering")
print("q: stops the motors")
```

```
print("x: exit")

# The funktion getch() reads the keys pressed by the user.
# With the read key values the direction and speed of
# the robot car are controlled.
def getch():
        ch = readchar.readchar()
        return ch

# The function printscreen()  displays the actual menu as well
# the speed of the left and right dc motor.
def printscreen():

        # With the command os.system('clear') the terminal window
        # is cleared before each refresh of the window.
        os.system('clear')
        print("w/s: acceleration")
        print("a/d: steering")
        print("q:stop the mtoren")
        print("x:program exit")
        print("========== Speed display ==========")
        print "Speed of the left motor:  ", speedleft
        print "Speed of the right motor: ", speedright

# The following infinity loop will only quit if the user presses
# the x key. As long as the loop runs the program will read the
# keys pressed by the user to control the robot car.
while True:

        # The function getch() reads the pressed keys by the user.
        # The pressed key values are read and the direction and
        # speed of the robot car is controlled.
        char = getch()

        # The robot car drives forward if the user presses the w key
        if(char == "w"):

                # The robot car accelerate with 10% steps each time the
                # user presses the w key. If 100% are reached the robot
                # car stops to accelerate.
```

```
            speedleft = speedleft + 0.1
            speedright = speedright + 0.1

            if speedleft > 1:
                        speedleft = 1
            if speedright > 1:
                        speedright = 1

            # Now the program L298NHBridge which was imported at the
            # beginning will now be called with the value of the
            # acceleration for the left and right motor.
            HBridge.setMotorLeft(speedleft)
            HBridge.setMotorRight(speedright)
            printscreen()

# The robot car drives backward if the user presses the s key.
if(char == "s"):
            # The robot car brakes with -10% steps each time the
            # user presses the s key. If -100% are reached the robot
            # car stops to de-accelerate at a maximum backward speed.
            speedleft = speedleft - 0.1
            speedright = speedright - 0.1

            if speedleft < -1:
                        speedleft = -1
            if speedright < -1:
                        speedright = -1

            # Now the program L298NHBridge which was imported at the
            # beginning will now be called with the value of the
            # de-acceleration for the left and right motor.
            HBridge.setMotorLeft(speedleft)
            HBridge.setMotorRight(speedright)
            printscreen()

# By pressing the q key the robot car stops.
if(char == "q"):
            speedleft = 0
            speedright = 0
            HBridge.setMotorLeft(speedleft)
```

11 Live Video Stream

For the configuration of the live video streaming some system changes in the installation of the Raspberry Pi operation system are required. The V4L2 driver which was developed by the Raspberry Pi Foundation and Broadcom needs to be installed and the corresponding module loaded.
Some additional small changes that are explained in the subsequent chapters need to be made until the video streaming is fully functional.

Estimated time:
The estimated time required for this chapter is approximately 45 minutes.

11.1 Loading kernel module bcm2835-v412

To load the kernel module “bcm2835-v4l2” the following command needs to be executed.

Command:

```
sudo modprobe bcm2835-v4l2
```

It is important, that the Raspberry Pi camera module is connected and activated in the Raspberry Pi Configuration menu. Otherwise an error will occur during execution of the command.

The following command shows all loaded kernel modules. With this command it is possible to check if the module “bcm2835-v4l2” is loaded.

Command:

```
sudo lsmod
```

At the following website you can learn more about Ubuntu and the kernel modules.

Website: http://wiki.ubuntuusers.de/Kernelmodule

11.2 Mjpg-streamer configuration

Now the final configuration of the mjpg-streamer under Raspbian will be performed.
Open with Midnight Commander the file in the folder “/opt/mjpg-streamer/start.sh” and search for the following line:

```
./mjpg_streamer -i "./input_uvc.so" -o "./output_http.so -w ./www"
```

12 Web-Interface controlling over the WIFI

This chapter describes the creation of a web interface for controlling the remote controlled car with a web browser. The remote controlled car web interface can be opened in a web browser form a cell phone, laptop or tablet. The web interface shows the required functionalities for controlling the remote controlled car as well the live video stream from inside the remote controlled car.
Developed as a simple website with overlays, the web interface shows the buttons necessary for controlling the remote controlled car e. g. forward, reverse, left and right. The fifth button is the stop button located in the centre of the window. The live stream is shown from the remote controlled car as a background of the web interface. The picture below shows the web interface from inside one remote controlled car filming another.

Estimated time:
The estimated time for configuration and developing of the web interface is calculated with 2 hours.

Figure 62: Web-Interface control

12.1 WebIOPi framework installation

For the communication between the website (web interface) and the GPIO pins of the Raspberry Pi WebIOPi is used. The following description is based on the version 0.7.1 of the WebIOPi library. Aside from controlling the GPIOs, the WebIOPi also hosts the website and acts as a server giving direct access to the remote controlled car web interface. The remoted controlled car web interface can be accessed via the IP address of the Raspberry Pi.

On the WebIOPi website many how to guides and program examples are available as well as the WebIOPi package for download.

Website: http://webiopi.trouch.com/

12.3 Running the web interface

After storing all necessary files for the web interface it can be run. To start the web interface execute the following program “RobotControlWeb.py” with the following command.

Command:

```
python RobotControlWeb.py
```

Open the following URL of the Raspberry PI to start the web interface.

URL:
http://<IP remote controlled car>:8000/rapicarweb.html (for the touch interface)
or
http://<IP remote controlled car>:8000/keyweb.html (for controlling with the keyboard and arrow keys)

The picture below shows the web interface for controlling the remote controlled car executed on a cell phone. With the touch screen of the cell phone the remote controlled car can be controlled with the arrow buttons.

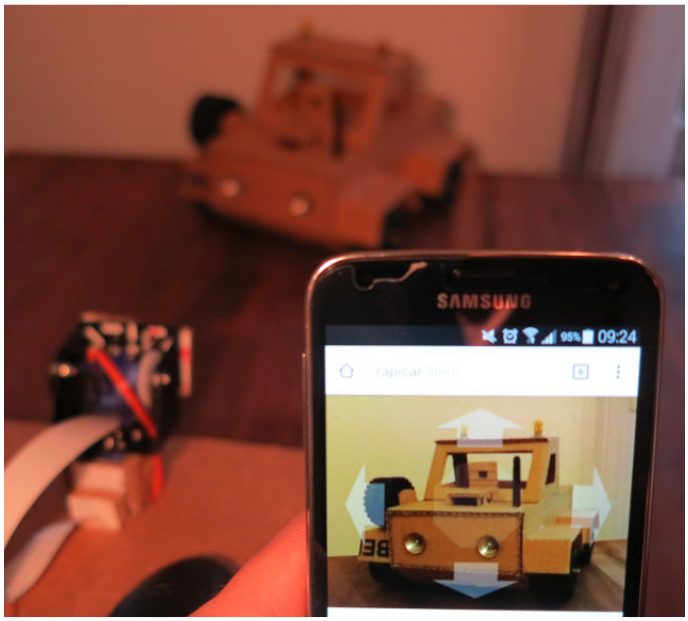


Figure 63: Web-Interface controlling with a cell phone

12.4 Summary – WebIOPi

The web interface described in the above chapter allows for controlling the remote controlled car with a web browser from any device.
With the fully automated start process of the programs described in chapter 13 no steps need to be performed manually to start the web interface. Then it will be sufficient to power up the remote controlled car and to run the URL from the web interface in order to control the car and to see the live video stream.

15 Outlook

The knowledge shared in this book forms the basis for further Do-It-Yourself projects with the Raspberry Pi. The experience of how the Raspberry Pi can interact in the real world allows you to deeply develop more ideas for example in the new and growing field of the Internet of Things. The remote controlled car as it is described in this book has its own IP address and can be reached and controlled via the internet. Many applications are possible such as the prototype of the Discoverer equipped with a GPS receiver and used as metal detector for treasury hunting.

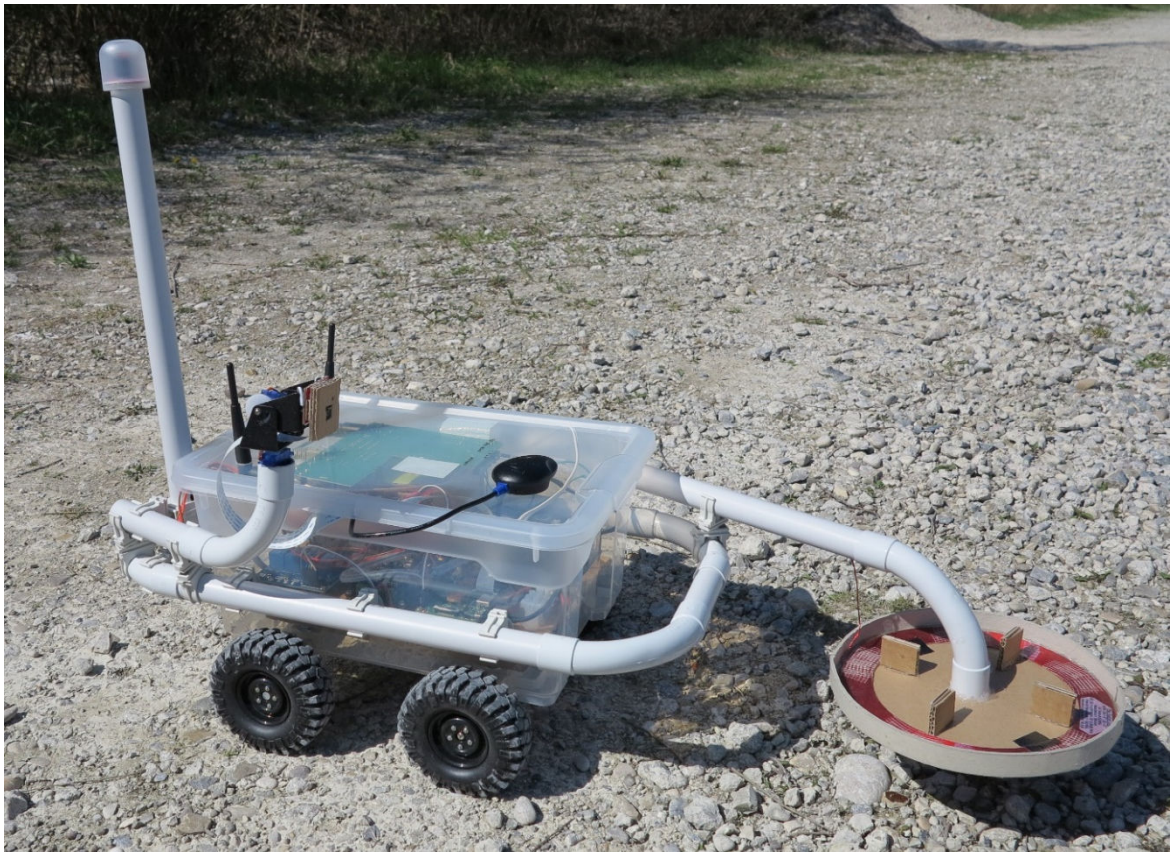


Figure 67: Remote controlled car - Discoverer

The increasing miniaturization of electronic components generates countless new opportunities such as the development of wearables or miniature electronics included into the clothes. This development offers boundless potential, limited only by your creativity in generating new concepts.

Additional ideas for remote controlled cars and pictures are available here:

Website: http://custom-build-robots.com/hall-of-fame

If you would like to show your own remote controlled car on my website please send me a picture to the following e-mail address:

E-Mail: halloffame@custom-build-robots.com