360pibot Documentation

Release 1.0

CH

Dec 13, 2022

Contents

1 Introduction **1**
- 1.1 Pictures of the demo implementation . . . 2
- 1.2 Used local coordinate system . . . 5
- 1.3 Cabling of the demo implementation . . . 6
- 1.4 References . . . 7

2 Installation **9**
- 2.1 Getting files . . . 9
- 2.2 Installing needed modules . . . 10
  - 2.2.1 With a virtual environment . . . 10
  - 2.2.2 Without a virtual environment . . . 10
- 2.3 Building/modifying the documentation . . . 11
- 2.4 Used module versions . . . 12
- 2.5 Raspberry Pi . . . 12
  - 2.5.1 Hotspot and remote access . . . 12
- 2.6 References . . . 13

3 Examples **15**
- 3.1 Calibrating 360° servo . . . 15
- 3.2 Emergency stop . . . 27
- 3.3 Moving the robot . . . 27
- 3.4 Moving standard servo . . . 28
- 3.5 Scanning the surrounding . . . 29
- 3.6 Simple collision avoiding algorithm . . . 30
- 3.7 References . . . 31

4 360pibot API **33**
- 4.1 lib_scanner . . . 33
- 4.2 lib_para_360_servo . . . 37
- 4.3 lib_motion . . . 39
- 4.4 References . . . 42

5 Indices and tables **43**

Python Module Index **45**

Index **47**

CHAPTER 1

Introduction

Python 3 implementation for programming a Parallax ActivityBot 360° Robot Kit 360_kit[3] with a Raspberry Pi. The modules (see *360pibot API*) of the implementation are using the pigpio[8] module to control the GPIOs of the Raspberry Pi. No other external module is needed.

At the moment, the following functions are implemented:

- Turning on the spot
- Moving straight - forward and backward
- Scanning the surrounding with an ultrasonic sensor mounted on a servo

The turning and straight movements are controlled by four digital PID controllers. Each wheel is controlled by a cascade control, using a cascade of two PID controllers. The outer loops control the position while the inner loops control the speed of each wheel.

The modules provide simple APIs for turning and straight movements and also for scanning the surrounding or stearing a servo. Have a look at the *Examples* section for some code examples.

Most of the default values in the modules are those which are used while experimenting/developing with the demo implementation. They provide a good starting point for the range of the values. Pictures of the demo implementation can be found further down in this section.

The modules also enable remote controlling the Raspberry Pis GPIOs. This enables use of the modules on a laptop/computer and over e.g. WLAN remote controlling the Raspberry Pi which provides a WLAN hotspot, see remote_pin[2] and pi_hotspot[1] . So, the robot can freely move with a powerbank attached without any peripheral devices while programming/controlling it. The possibillity of remote controlling the Raspberry Pis GPIOs is a big advantage of the used pigpio[8] module. This drastically improves the use of the modules, because then all programming/controlling can be done on a laptop/computer inlcuding using an IDE, having much more system ressources and so on. It is also possible to use the modules on the Raspberry Pi itself and connect to it over VNC, see VNC[9] . For both ways,

[3] https://www.parallax.com/product/32600
[8] https://pypi.org/project/pigpio/
[2] http://gpiozero.readthedocs.io/en/stable/remote_gpio.html
[1] https://www.raspberrypi.org/documentation/configuration/wireless/access-point.md
[9] https://www.raspberrypi.org/documentation/remote-access/vnc/

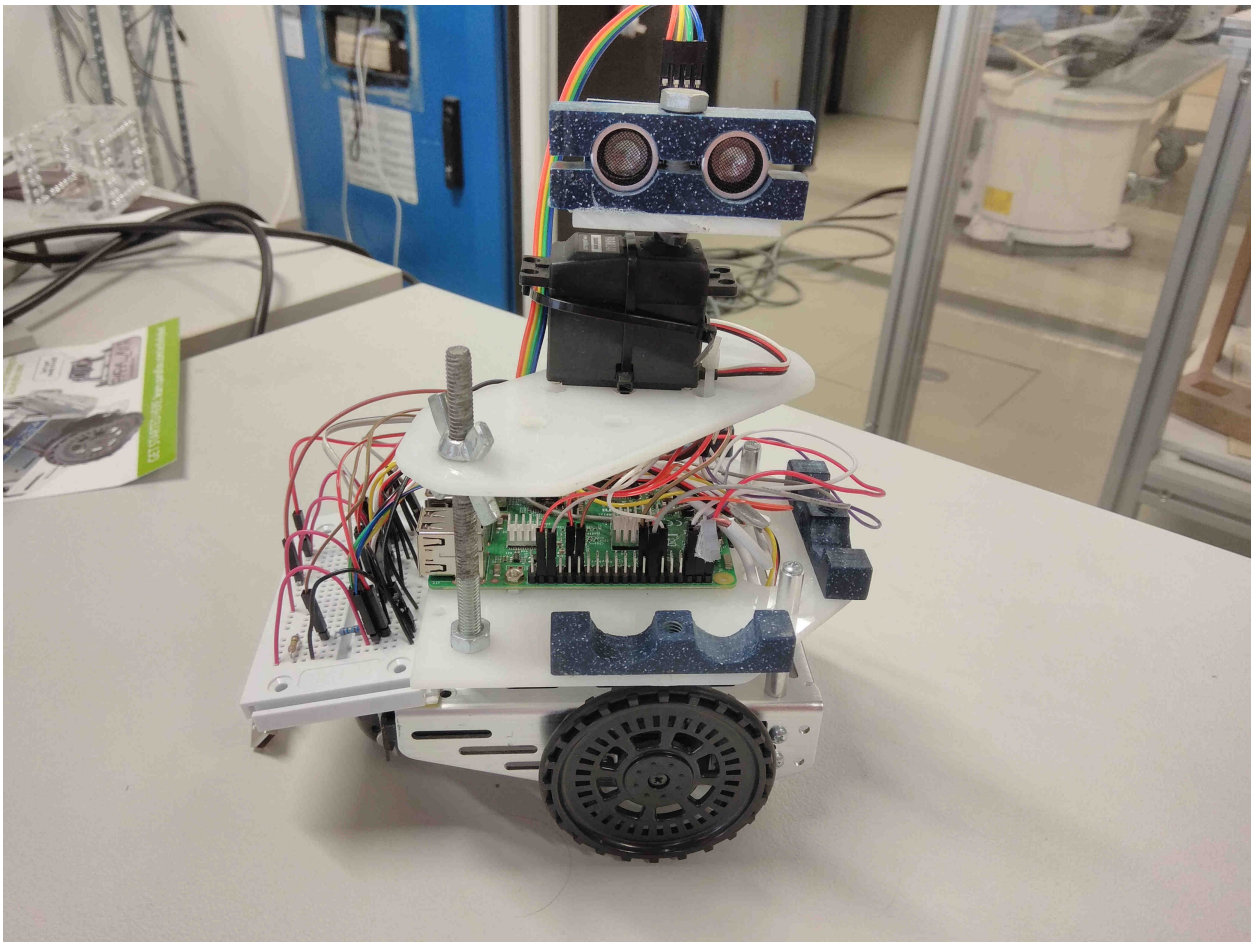
using the modules on the Raspberry Pi itself or remote on a laptop/computer to control the Raspberry Pis GPIOs, no modifications have to be done in the source code of the modules.

The documentation is made with Sphinx[10] and can be extended or modified as needed for e.g. documenting own projects based on this or if extending functionality of the modules and documenting this. The whole documentation is stored in the `docs/` folder.

Instead of buying an ActivityBot 360° Robot Kit 360_kit[3] it is sufficient to buy two Parallax Feedback 360° High-Speed Servos 360_data_sheet[4] , two robot wheels wheel_robot[7], one Parallax Standard Servo stand_data_sheet[5] and a HC-SR04[6] ultrasonic sensor to build your own chassi/robot.

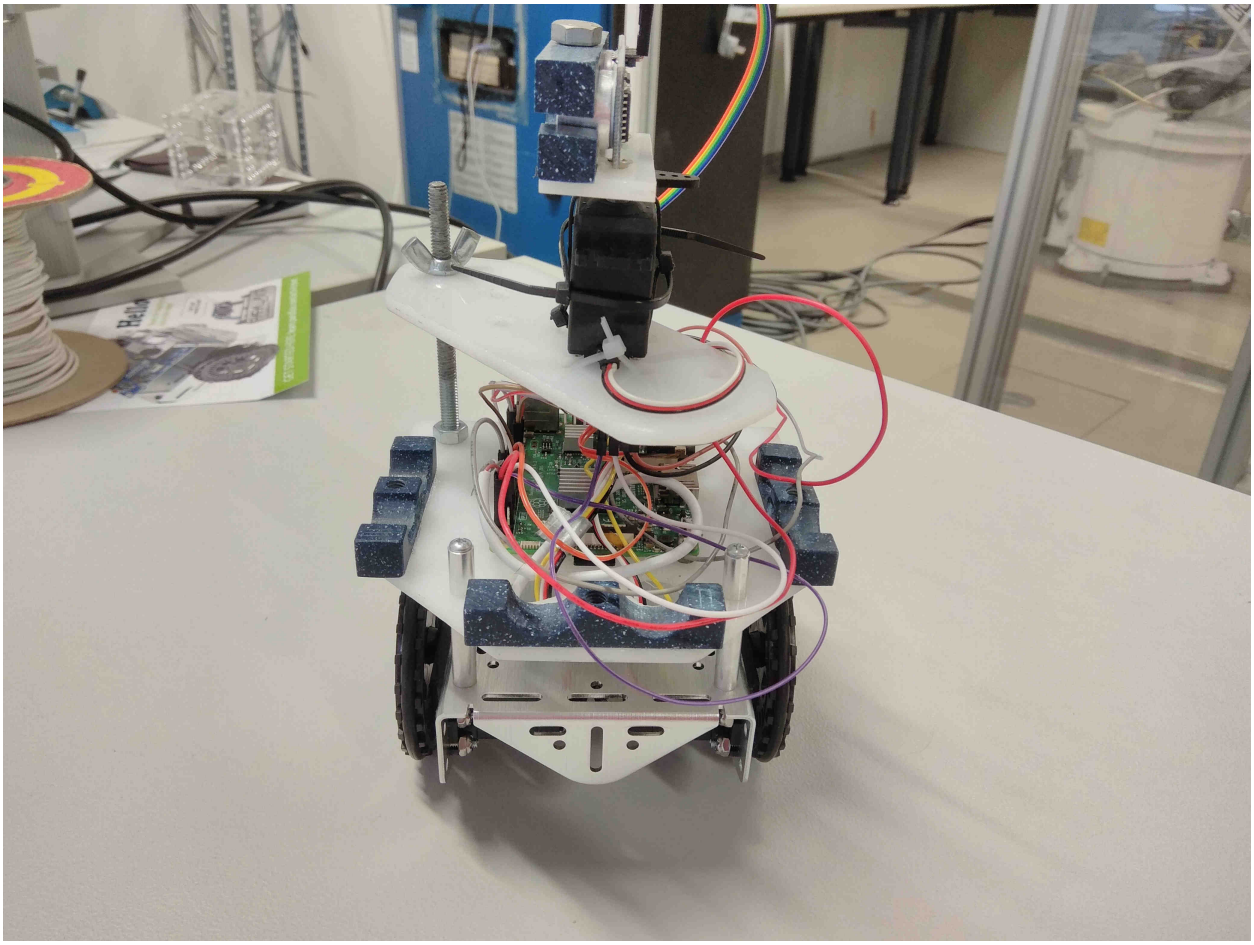
1.1 Pictures of the demo implementation

View from the right side.

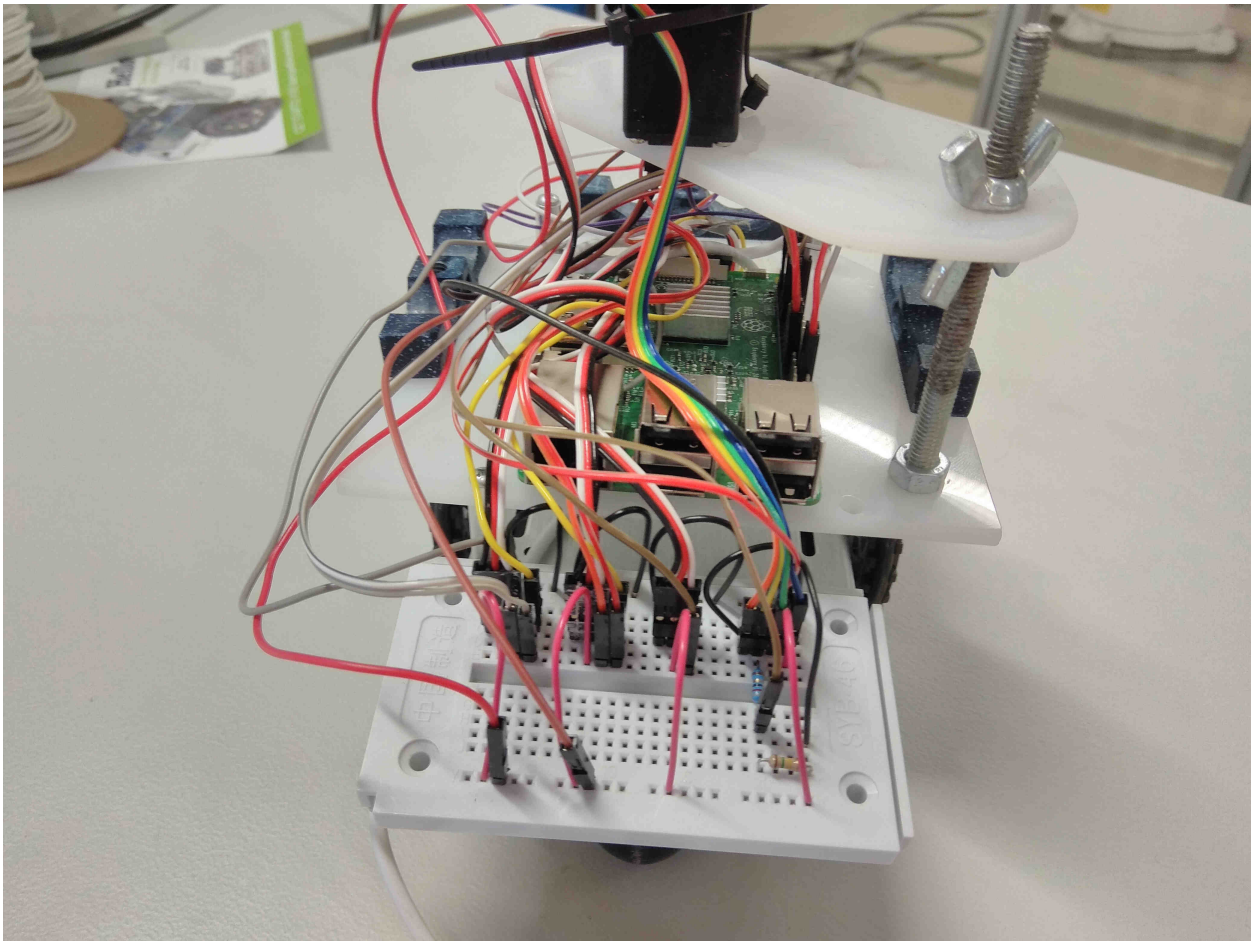


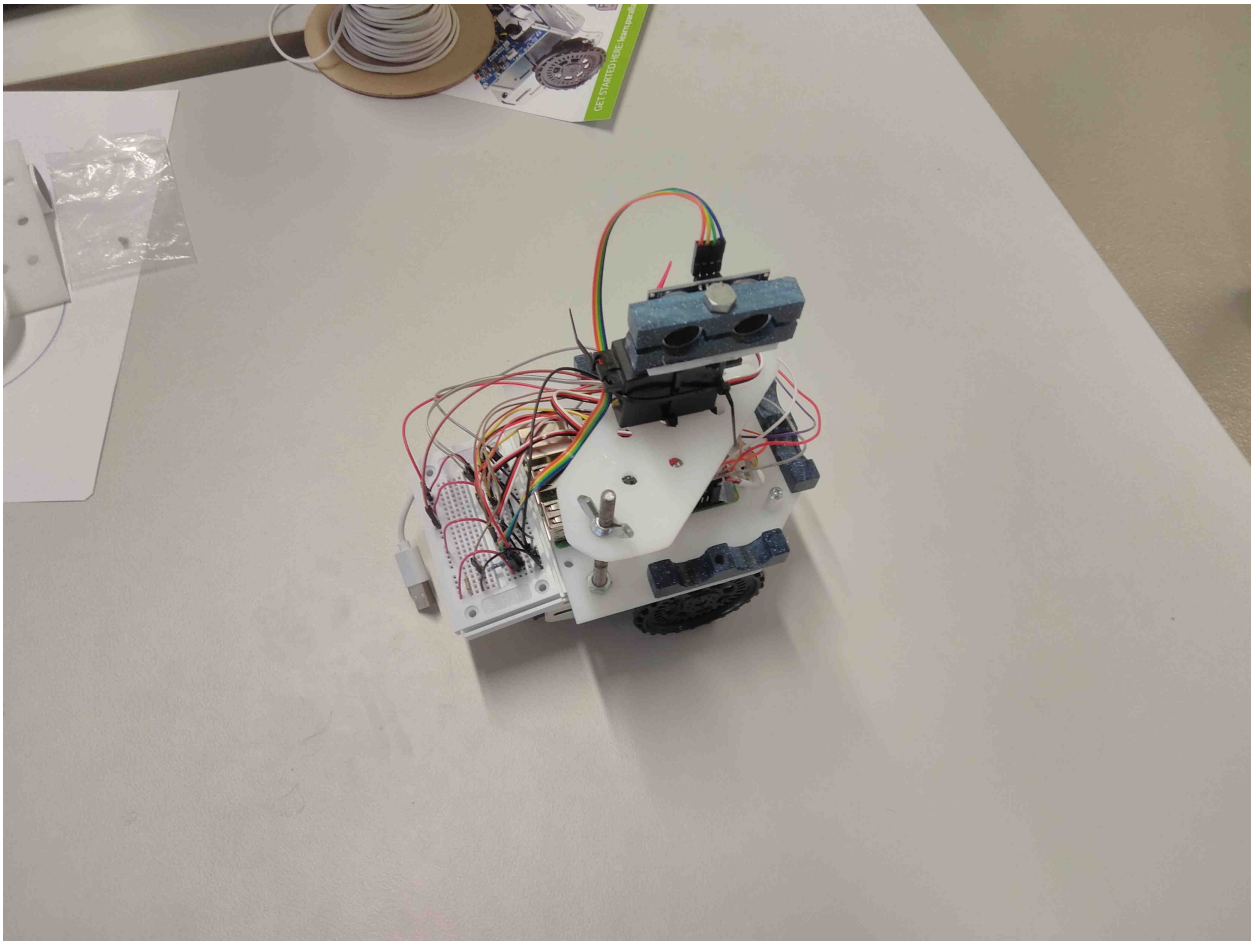
View from the front side.

[10] https://www.sphinx-doc.org/
[4] https://www.parallax.com/sites/default/files/downloads/900-00360-Feedback-360-HS-Servo-v1.1.pdf
[7] https://www.parallax.com/product/28114
[5] https://www.parallax.com/sites/default/files/downloads/900-00005-Standard-Servo-Product-Documentation-v2.2.pdf
[6] https://cdn.sparkfun.com/assets/b/3/0/b/a/DGCH-RED_datasheet.pdf



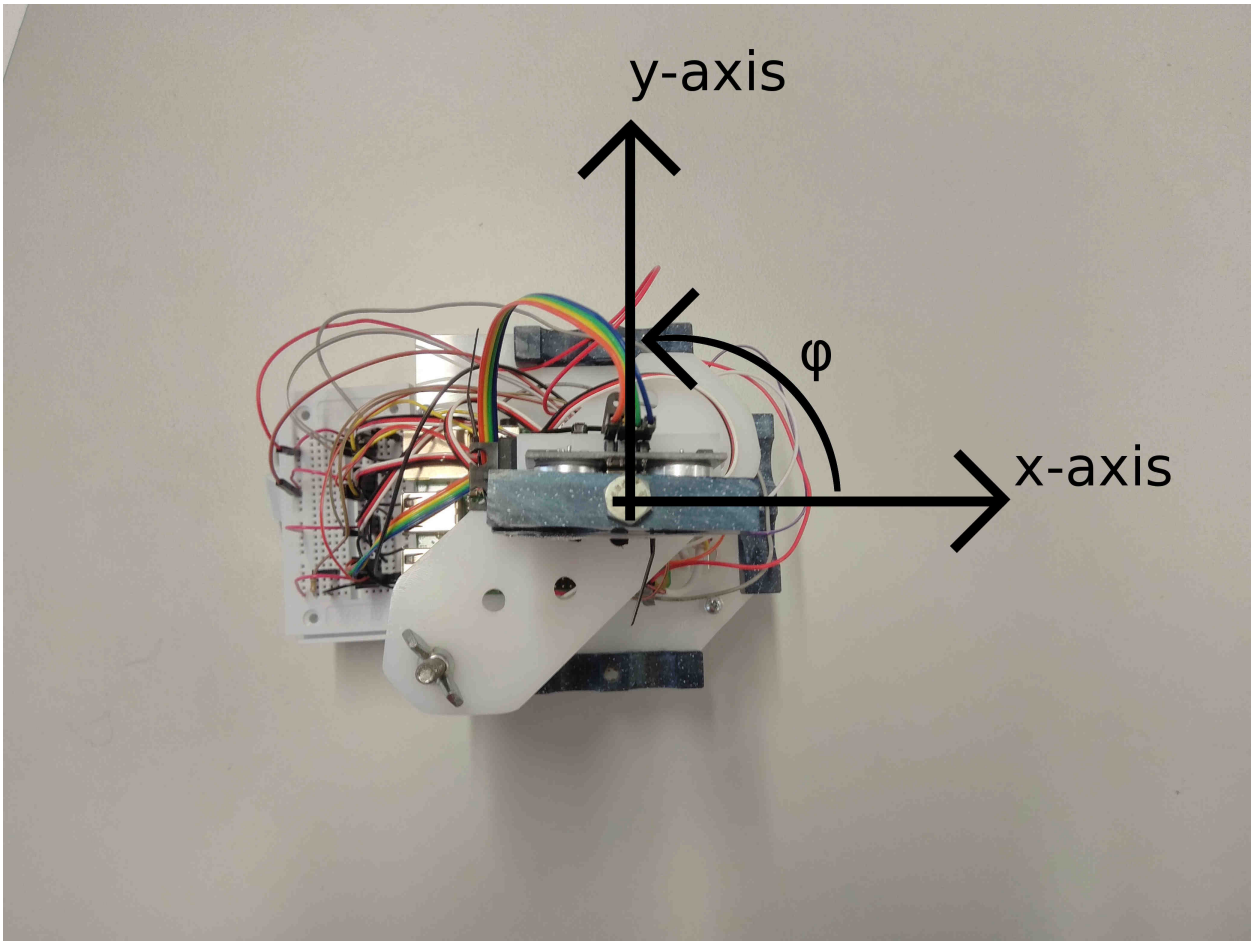
View from the back side.





1.2 Used local coordinate system

The following picture shows the local coordinate system which is used in *lib_motion* .



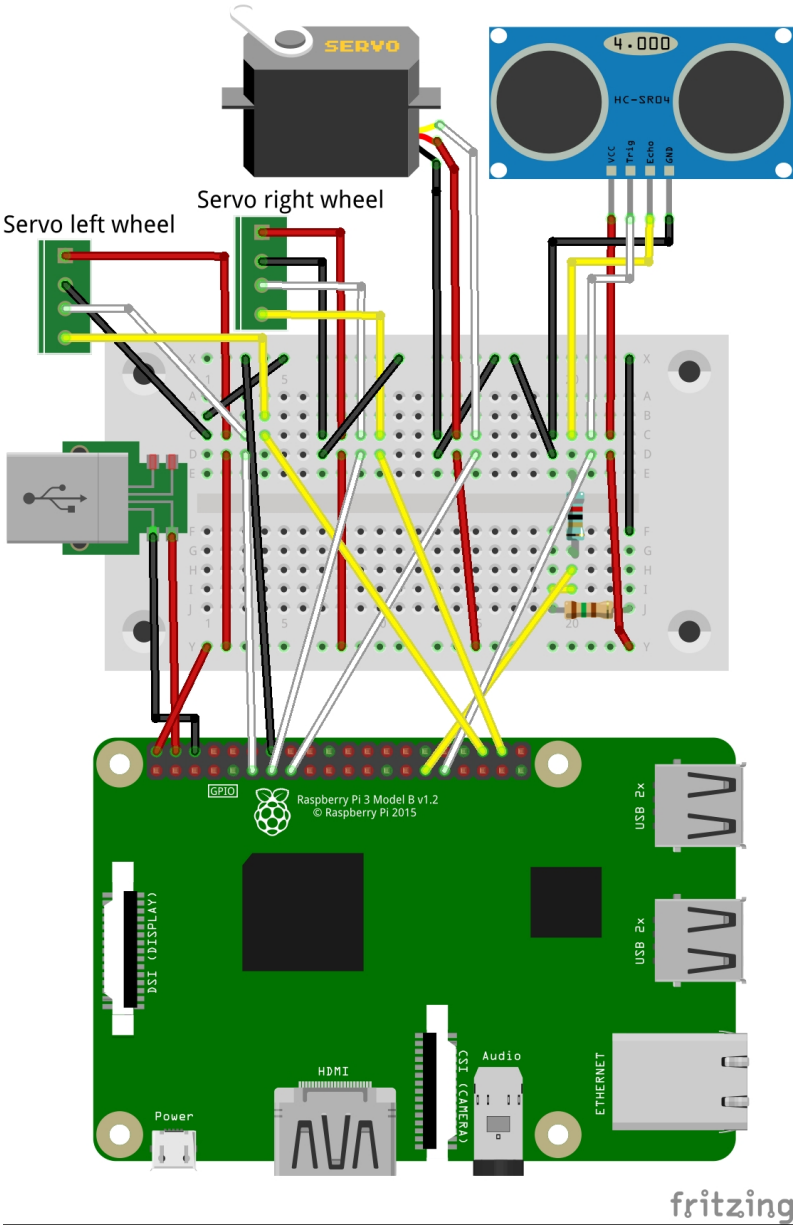
1.3 Cabling of the demo implementation

Below see a Fritzing[11] sheet which illustrates the cabling of the demo implementation. The voltage divider is build out of two resistors, blue one with 82 Ohm and gold one with 150 Ohm.

Warning: The voltage divider is very important. The `Echo` Pin of the HC-SR04[6] outputs a PWM with the same voltage as `VCC` Pin (5 V in this case) which needs to be converted to 3.3 V. 3.3 V is the max voltage the Raspberry Pi can handle on a GPIO, otherwise it might get damaged! The chosen resistors for the voltage divider convert 5 V to 3.23 V. **Setup:** The output of the `Echo` Pin is connected to the blue 82 Ohm resistor. At its end, the GPIO is connected and then the golden 150 Ohm resistor at whose end ground is connected.

Warning: The 5 V of the USB power supply should be connceted to the 5 V Pin of the Raspberry Pi directly, as shown in the Fritzing sheet. The USB power supply does not only power the Raspberry Pi itself, also the three servos and maybe more devices, which get added in the future, are powered over it. Therefore, powering all devices over the Raspberry Pis micro-USB port should be avoided, because otherwise all needed current of all devices would be conducted through the Raspberry Pi.

[11] http://fritzing.org/



1.4 References

CHAPTER 2

Installation

The following steps to set up a working environment are valid for both, the Raspberry Pi and a laptop/computer. Remember, the modules can be used on a laptop/computer for remote controlling the Raspberry Pis GPIOs or on the Raspberry Pi itself without any modifications of the source code of the modules. On the Raspberry Pi, the latest Raspbian version and afterwards all available updates should be installed. The latest Raspbian image can be downladed from Raspbian Downloads[1] . Installing all available updates is described here Raspbian Update[2] .

2.1 Getting files

One option is creating a folder named `360pibot` in the Raspberry Pis `home` folder

```
cd ~
mkdir 360pibot
```

and then connect the Raspberry Pi to the internet, open the projects github page, download the repository as a .zip file and unzip it in the created `360pibot` folder.

Another option is to clone the git repository to the `home` folder. This will automatically create the `360pibot` folder and download the project files. First, git will be installed, then the repository will be cloned.

```
sudo apt-get update
sudo apt-get install git
cd ~
git clone https://github.com/choeffer/360pibot
```

Both methods for getting the project files are also working on a laptop/computer.

[1] https://www.raspberrypi.org/downloads/raspbian/
[2] https://www.raspberrypi.org/documentation/raspbian/updating.md

2.2 Installing needed modules

Next step is to install the needed modules. They can be installed in the global Python 3 environment or in a virtual Python 3 environment. The latter has the advantage that the packages are isolated from other projects and also from the system wide installed global once. If things get messed up, the virtual environment can just be deleted and created from scratch again. For more informations about virtual environments in Python 3, see venv1[3] and venv2[4] . First, installing with a virtual environment will be explainend, afterwards with using the global Python 3 environment.

2.2.1 With a virtual environment

On a Raspberry Pi first ensure that the packages `python3-venv` and `python3-pip` are installed. This has also to be checked on Debian based distributions like Ubuntu/Mint.

```
sudo apt-get update
sudo apt-get install python3-venv python3-pip
```

Afterwards, navigate to the created `360pibot` folder and create a virtual environment named `venv` and activate it. An activated virtual environment is indicated by a `(venv)` in the beginning of the terminal prompt.

Note: The created `venv` folder is added to the .gitignore file and will therefore not be tracked by git.

```
cd ~
cd 360pibot
python3 -m venv venv
source venv/bin/activate
```

With the activated virtual environment install the needed `pigpio` module inside.

```
pip3 install pigpio
```

Deactivating the acvtivated virtual environment can be done later by just typing `deactivate` in the terminal where the virtual environment is activated.

```
deactivate
```

Note: For later using the installed module, the virtual environment has to be activated, because the pigpio package is installed inside and is not callable from the global Python 3 environment.

2.2.2 Without a virtual environment

On a Raspberry Pi first ensure that the package `python3-pip` is installed. This has also to be checked on Debian based distributions like Ubuntu/Mint. Then, the `pigpio` module will be installed in the global Python 3 environment.

```
sudo apt-get update
sudo apt-get install python3-pip
pip3 install pigpio
```

[3] https://docs.python.org/3/tutorial/venv.html
[4] https://docs.python.org/3/library/venv.html

2.3 Building/modifying the documentation

The whole documentation is made with Sphinx[5] and can be extended or modified as needed for e.g. documenting own projects based on this or if extending functionality of the modules and documenting this. The whole documentation is stored in the `docs/` folder. The standard docstring format (ReStructuredText[11] (reST)) is used. The used theme is from Read the Docs[6] where also the documentation is hosted. Therefore, two more modules are needed for beeing able to build or extend/modify the documentation. How to use Sphinx is not part of this documentation. But there are good introductions and tutorials available which provide a good starting point, see docs1[12] , docs2[13] , docs3[14] and docs4[15] .

Note: For the creation of the docs `conf.py` , `index.rst` , and folder structure etc. the `sphinx-quickstart` command was used.

Note: The created `docs/_build` folder is added to the .gitignore file and will therefore not be tracked by git. This folder contains the output after building the docs.

If using a virtual environment to install the two modules

```
cd ~
cd 360pibot
source venv/bin/activate
pip3 install sphinx sphinx_rtd_theme
```

or if installing them in the global Python 3 environment.

```
pip3 install sphinx sphinx_rtd_theme
```

After this, the following command `make html` builds the html documentation which will be stored in the `docs/_build/html/` folder. There, open the `index.html` with your preferred web browser.

If using a virtual environment

```
cd ~
cd 360pibot
source venv/bin/activate
cd docs
make html
```

or if using the global Python 3 environment.

```
cd ~
cd 360pibot/docs
make html
```

Sphinx can create the documentation in different formats (e.g. latex, html ,pdf, epub), see sphinx-build[18] for more informations.

[5] https://www.sphinx-doc.org/
[11] http://www.sphinx-doc.org/en/master/usage/restructuredtext/basics.html
[6] https://readthedocs.org/
[12] https://realpython.com/documenting-python-code/
[13] https://docs.python-guide.org/writing/documentation/
[14] https://www.youtube.com/watch?v=0ROZRNZkPS8
[15] https://www.youtube.com/watch?v=hM4I58TA72g
[18] http://www.sphinx-doc.org/en/master/man/sphinx-build.html#cmdoption-sphinx-build-b

2.4 Used module versions

The `requirements.txt` file will install the exact versions of the modules which are used while experimenting/developing with the demo implementation and writing the documentation.

This can be done by using a virtual environment

```
cd ~
cd 360pibot
source venv/bin/activate
pip3 install -r requirements.txt
```

or by installing them in the global Python 3 environment.

```
pip3 install -r requirements.txt
```

The `requirements.txt` file is created with `pip3 freeze > requirements.txt`. The `requirements_rtd.txt` file is used by Read the Docs[6] . The online version of the documentation is auto build/updated each time a `git push` is made to the github repository. For further information, see Read the Docs Webhooks[17] .

2.5 Raspberry Pi

The following steps are specific to the Raspberry Pi. It is necessary to install the `pigpio` package, enable starting the pigpio daemon at boot and then doing a reboot to activate the pigpio daemon. For more information see pigpio_download[7] and remote_pin[10] . For the demo implementation the package from the Raspbian repository is installed. This ensures that the package is good integrated in the system, even if it might be a bit older.

```
sudo apt-get update
sudo apt-get install pigpio
sudo systemctl enable pigpiod
sudo reboot
```

Note: If the Raspberry Pis GPIOs are not responding anymore, it might help to restart the pigpio daemon on the Raspberry Pi. For that, SSH into the Raspberry Pi if remotely working with it, otherwise use the local terminal, and execute the following two commands.

```
sudo systemctl daemon-reload
sudo systemctl restart pigpiod.service
```

2.5.1 Hotspot and remote access

An important step which improves programming/controlling the Raspberry Pi is to make it remotely accessible. This can be done by connecting the Raspberry Pi to a WLAN network or by enabling a hotspot on it, see pi_hotspot[8] . This is recommended before using it. Setting up a hotspot will not be covered here, because the official documentation is good and updated regularly to match the latest Raspbian changes.

[17] https://docs.readthedocs.io/en/latest/webhooks.html
[7] http://abyz.me.uk/rpi/pigpio/download.html
[10] http://gpiozero.readthedocs.io/en/stable/remote_gpio.html
[8] https://www.raspberrypi.org/documentation/configuration/wireless/access-point.md

Also make yourself familiar with using VNC[9] or using remote_pin[10] . Latter will again drastically improve the use of the modules, because then all programming/controlling can be done on a laptop/computer inlcuding using an IDE, having much more system ressources and so on. The latter option is shortly described.

After enabling a hotspot on the Raspberry Pi and beeing connected with your laptop/computer, the following steps are needed to remote control the Raspberry Pis GPIOs. For a more detailed description, see remote_pin[10] .

First, in the Raspberry Pi configuration *Remote GPIO* has to be enabled. This can be done via GUI or `sudo raspi-config`. This will allow remote connections while the pigpio daemon is running.

Then, the environment variable has to be set while or before launching Python 3 or an IDE. This variable will point to the IP address (and optional port) on which the Raspberry Pi is accessible. This can be on its own provided hotspot/network or on a WLAN it is connected to.

```
PIGPIO_ADDR=192.168.1.3 python3 hello.py
PIGPIO_ADDR=192.168.1.3 python3 code .
```

There are also other possibilities available for configuring remote access. They are mentioned in the pigpio documentation, see pigpio_pi[16] . E.g. the IP address and port can be passed as arguments if initializing a pigpio.pi() instance.

2.6 References

[9] https://www.raspberrypi.org/documentation/remote-access/vnc/
[16] http://abyz.me.uk/rpi/pigpio/python.html#pigpio.pi

CHAPTER 3

Examples

In this section some code examples are listed. The intention of this code examples is providing a guideline and a style guide how things should be done in general, and not how each class is working in detail. This is described in section *360pibot API* .

For the Parallax Feedback 360° High-Speed Servo 360_data_sheet[1] also C example code is available sample_360[3] .

Note: In this examples, not all possible or needed arguments are passed to the classes or methods. The examples mostly use the default values of the modules. For more informations about which arguments can be passed to a class or method and which arguments need to be adjusted, see *360pibot API* .

3.1 Calibrating 360° servo

The following code calibrates a Parallax Feedback 360° High-Speed Servo 360_data_sheet[1] . The values `dcMin` and `dcMax` are later needed in *`lib_motion.control`* . For more informations, see *`lib_para_360_servo.calibrate_pwm()`* . This example is included as `calibrate.py` .

Note: As seen in the example code, one pigpio.pi() instance can be passed to all used classes or modules. This reduces the number of parallel threads which gets started. If using one pigpio.pi() instance for each used class or module, so multiple instances in one script, more parallel threads will be started, which is not necessary.

```
1 import time
2
3 import pigpio
4
5 import lib_para_360_servo
```

(continues on next page)

[1] https://www.parallax.com/sites/default/files/downloads/900-00360-Feedback-360-HS-Servo-v1.1.pdf
[3] https://www.parallax.com/downloads/feedback-360%C2%B0-high-speed-servo-propeller-c-example-code

(continued from previous page)

```

#define GPIO for each servo to read from
gpio_l_r = 16
gpio_r_r = 20

#define GPIO for each servo to write to
gpio_l_w = 17
gpio_r_w = 27

pi = pigpio.pi()

#### Calibrate servos, speed  = 0.2 and -0.2
#choose gpio_l_w/gpio_l_r (left wheel), or gpio_r_w/gpio_r_r
#(right wheel) accordingly

servo = lib_para_360_servo.write_pwm(pi = pi, gpio = gpio_r_w)

#buffer time for initializing everything
time.sleep(1)
servo.set_speed(0.2)
wheel = lib_para_360_servo.calibrate_pwm(pi = pi, gpio = gpio_r_r)
servo.set_speed(0)

#http://abyz.me.uk/rpi/pigpio/python.html#stop
pi.stop()
```

Below see different example terminal outputs which were generated with the demo installation.

```
#### example output left wheel, speed 0.2

Starting measurements for: 120 seconds.-------------------------------------------------
↪-----------Ascending sorted distinct duty cycle values: [30.939999999999998, 31.
↪849999999999998, 32.76, 35.489999999999995, 36.4, 37.309999999999995, 40.04, 40.
↪949999999999996, 44.589999999999996, 45.5, 46.41, 49.13999999999999, 50.05, 50.
↪959999999999994, 53.69, 54.599999999999994, 55.51, 58.239999999999995, 59.15, 60.
↪059999999999995, 62.79, 63.699999999999996, 64.61, 67.33999999999999, 68.25, 69.16,␣
↪71.89, 72.8, 73.71, 76.44, 77.35, 78.25999999999999, 80.99, 81.89999999999999, 82.
↪80999999999999, 85.53999999999999, 86.44999999999999, 87.36, 90.08999999999999, 91.
↪0, 91.91, 94.64, 95.55, 96.46, 99.19, 100.1, 101.00999999999999, 103.74, 104.
↪64999999999999, 105.55999999999999, 108.28999999999999, 109.19999999999999, 110.11,␣
↪112.83999999999999, 113.74999999999999, 114.66, 117.38999999999999, 118.3, 119.21,␣
↪121.94, 122.85, 123.75999999999999, 126.49, 127.39999999999999, 128.31, 131.04, 131.
↪95, 132.85999999999999, 135.59, 136.5, 137.41, 140.14, 141.04999999999998, 141.
↪95999999999998, 144.69, 145.6, 146.51, 149.23999999999998, 150.14999999999998, 151.
↪06, 153.79, 154.7, 155.60999999999999, 158.33999999999997, 159.25, 160.16, 162.89,␣
↪163.79999999999998, 164.70999999999998, 167.44, 168.35, 169.26, 171.98999999999998,␣
↪172.89999999999998, 173.81, 176.54, 177.45, 178.35999999999999, 181.08999999999997,␣
↪182.0, 182.91, 185.64, 186.54999999999998, 187.45999999999998, 190.19, 191.1, 192.
↪01, 194.73999999999998, 195.64999999999998, 196.55999999999997, 199.29, 200.2, 201.
↪10999999999999, 203.83999999999997, 204.75, 205.66, 208.39, 209.29999999999998, 210.
↪20999999999998, 212.94, 213.85, 214.76, 217.48999999999998, 218.39999999999998, 219.
↪30999999999997, 222.04,222.95, 223.85999999999999, 226.58999999999997, 227.
↪49999999999997, 228.41, 231.14, 232.04999999999998, 232.95999999999998, 235.69, 236.
↪6, 237.51, 240.23999999999998, 241.14999999999998, 242.05999999999997,244.79, 245.7,
↪ 246.60999999999999, 249.33999999999997, 250.24999999999997, 251.16, 253.89, 254.
↪79999999999998, 255.70999999999998, 258.44, 259.34999999999997, 260.26, 262.99, 263.
↪9, 264.81, 267.53999999999996, 268.45, 269.35999999999996, 272.09, 273.0, 273.
↪90999999999997, 276.64, 277.54999999999995, 278.46, 281.19, 282.09999999999997, 283.
↪01, 285.73999999999995, 286.65, 287.56, 290.28999999999996, 291.2, 292.
↪10999999999996, 294.84, 295.75, 296.65999999999997, 299.39, 300.29999999999995, 301.
↪21, 303.94, 304.84999999999997, 305.76, 308.48999999999995, 309.4, 310.31, 313.
↪03999999999996, 313.95, 314.85999999999996, 317.59, 318.5, 319.40999999999997, 322.
↪14, 323.04999999999995, 323.96, 326.69, 327.59999999999997, 328.51, 331.
↪23999999999995, 332.15, 333.06, 335.78999999999996, 336.7, 337.60999999999996, 340.
```

(continues on next page)

(continued from previous page)

```
-----------------------------------------------------------
Ascending counted, sorted and rounded distinct differences between duty cycle values:␣
↪Counter({0.91: 412, 2.73: 205, 3.64: 1})
-----------------------------------------------------------
Smallest 250 values: [30.939999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998,31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
```

(continues on next page)

(continued from previous page)

```
-----------------------------------------------------------
Biggest 250 values: [964.5999999999999, 964.5999999999999, 964.5999999999999, 964.
↪5999999999999, 964.5999999999999, 964.5999999999999, 964.5999999999999, 964.
↪5999999999999, 964.5999999999999, 964.5999999999999, 964.5999999999999, 964.
↪5999999999999, 964.5999999999999, 964.5999999999999, 964.5999999999999, 964.
↪5999999999999, 964.5999999999999, 964.5999999999999, 964.5999999999999, 964.
↪5999999999999, 964.5999999999999, 964.5999999999999, 964.5999999999999, 964.
↪5999999999999, 964.5999999999999, 964.5999999999999, 964.5999999999999, 964.
↪5999999999999, 964.5999999999999, 964.5999999999999, 964.5999999999999, 964.
↪5999999999999, 964.5999999999999, 964.5999999999999, 964.5999999999999, 964.
↪5999999999999, 964.5999999999999, 964.5999999999999, 964.5999999999999, 964.
↪5999999999999, 964.5999999999999, 964.5999999999999, 964.5999999999999, 964.
↪5999999999999, 964.5999999999999, 964.5999999999999, 964.5999999999999, 964.
↪5999999999999, 964.5999999999999, 964.5999999999999, 964.5999999999999, 964.
↪5999999999999, 964.5999999999999, 964.5999999999999, 964.5999999999999, 964.
↪5999999999999, 964.5999999999999, 964.5999999999999, 964.5999999999999, 964.
↪5999999999999, 964.5999999999999, 965.51, 965.51, 965.51, 965.51, 965.51, 965.51,␣
↪965.51, 965.51, 965.51, 965.51, 965.51, 965.51, 965.51, 965.51, 965.51, 968.
↪2399999999999, 968.2399999999999, 968.2399999999999, 968.2399999999999, 969.15, 969.
↪15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15,␣
↪969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.
↪15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15,␣
↪969.15, 969.15,969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.
↪15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15,␣
↪969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.
↪15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15,␣
↪969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.
↪15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15,␣
↪969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.
↪15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15,␣
↪969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.
↪15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15,␣
↪969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.
↪15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15,␣
↪969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 969.15, 970.06, 970.06, 970.06, 970.
↪06]
-----------------------------------------------------------
duty_cycle_min: 31.85
duty_cycle_max: 969.15

#### example output left wheel, speed -0.2

Starting measurements for: 120 seconds.-------------------------------------------------
↪-----------Ascending sorted distinct duty cycle values: [26.389999999999997, 27.
↪299999999999997, 30.939999999999998, 31.849999999999998, 36.4, 37.309999999999995,␣
↪40.04, 40.949999999999996, 41.86, 44.589999999999996, 45.5, 46.41, 49.
↪13999999999999, 50.05, 50.959999999999994, 53.69, 54.599999999999994, 55.51, 58.
↪239999999999995, 59.15, 60.059999999999995, 62.79, 63.699999999999996, 64.61, 67.
↪33999999999999, 68.25, 69.16, 71.89, 72.8, 73.71, 76.44, 77.35, 78.25999999999999,␣
↪80.99, 81.89999999999999, 82.80999999999999, 85.53999999999999, 86.44999999999999,␣
↪87.36, 90.08999999999999, 91.0, 91.91, 94.64, 95.55, 96.46, 99.19, 100.1, 101.
↪00999999999999, 103.74, 104.64999999999999, 105.55999999999999, 108.28999999999999,␣
↪109.19999999999999, 110.11, 112.83999999999999, 113.74999999999999, 114.66, 117.
↪38999999999999, 118.3, 119.21, 121.94, 122.85, 123.75999999999999, 126.49, 127.
↪39999999999999, 128.31, 131.04, 131.95, 132.85999999999999, 135.59, 136.5, 137.41,␣
↪140.14, 141.04999999999998, 141.95999999999998, 144.69, 145.6, 146.51, 149.
↪23999999999998, 150.14999999999998, 151.06, 153.79, 154.7, 155.60999999999999, 158.
↪33999999999997, 159.25, 160.16, 162.89, 163.79999999999998, 164.70999999999998, 167.
↪44, 168.35, 169.26, 171.98999999999998, 172.89999999999998, 173.81, 176.54, 177.45,␣
↪178.35999999999999, 181.08999999999997, 182.0, 182.91, 185.64, 186.54999999999998,␣
↪187.45999999999998, 190.19, 191.1, 192.01, 194.73999999999998, 195.64999999999998,␣
↪196.55999999999997, 199.29, 200.2, 201.10999999999999, 203.83999999999997, 204.75,␣
↪205.66, 208.39, 209.29999999999998, 210.20999999999998, 212.94, 213.85, 214.76, 217.
↪48999999999998, 218.39999999999998, 219.30999999999997, 222.04, 222.95, 223.
```

(continues on next page)

(continued from previous page)

```
----------------------------------------------------------
Ascending counted, sorted and rounded distinct differences between duty cycle values:␣
↪Counter({0.91: 409, 2.73: 203, 3.64: 2, 4.55: 1})
----------------------------------------------------------
Smallest 250 values: [26.389999999999997, 26.389999999999997, 26.389999999999997, 26.
↪389999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997,27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
```

(continues on next page)

(continued from previous page)

```
------------------------------------------------------------------
Biggest 250 values: [960.05, 960.05, 960.05, 960.05, 960.05, 960.05, 960.05, 960.05,␣
↪960.05, 960.05, 960.05, 960.05, 960.05, 960.05, 960.05, 960.05, 960.05, 960.05, 960.
↪05, 960.05, 960.05, 960.05, 960.05, 960.05, 960.05, 960.05, 960.05, 960.
↪9599999999999, 960.9599999999999, 960.9599999999999, 960.9599999999999, 960.
↪9599999999999, 960.9599999999999, 960.9599999999999, 960.9599999999999, 960.
↪9599999999999, 960.9599999999999, 960.9599999999999, 960.9599999999999, 960.
↪9599999999999, 960.9599999999999, 960.9599999999999, 960.9599999999999, 960.
↪9599999999999, 960.9599999999999, 963.6899999999999, 963.6899999999999, 963.
↪6899999999999, 963.6899999999999, 963.6899999999999, 963.6899999999999, 964.
↪5999999999999, 964.5999999999999, 964.5999999999999, 964.5999999999999, 964.
↪5999999999999, 964.5999999999999, 964.5999999999999, 964.5999999999999, 964.
↪5999999999999, 964.5999999999999, 964.5999999999999, 964.5999999999999, 964.
↪5999999999999, 964.5999999999999, 964.5999999999999, 964.5999999999999, 964.
↪5999999999999, 964.5999999999999,964.5999999999999, 964.5999999999999, 964.
↪5999999999999, 964.5999999999999, 964.5999999999999, 964.5999999999999, 964.
↪5999999999999, 964.5999999999999, 964.5999999999999, 964.5999999999999, 964.
↪5999999999999, 964.5999999999999, 964.5999999999999, 964.5999999999999, 964.
↪5999999999999, 964.5999999999999, 964.5999999999999, 964.5999999999999, 964.
↪5999999999999, 964.5999999999999, 964.5999999999999, 964.5999999999999, 964.
↪5999999999999, 964.5999999999999, 964.5999999999999, 964.5999999999999, 964.
↪5999999999999, 964.5999999999999, 964.5999999999999, 964.5999999999999, 964.
↪5999999999999, 964.5999999999999, 964.5999999999999, 964.5999999999999, 964.
↪5999999999999, 964.5999999999999, 964.5999999999999, 964.5999999999999, 964.
↪5999999999999, 964.5999999999999, 964.5999999999999, 964.5999999999999, 964.
↪5999999999999, 964.5999999999999, 964.5999999999999, 964.5999999999999, 964.
↪5999999999999, 964.5999999999999, 964.5999999999999, 964.5999999999999, 964.
↪5999999999999, 964.5999999999999, 964.5999999999999, 964.5999999999999, 964.
↪5999999999999, 964.5999999999999, 964.5999999999999, 964.5999999999999, 964.
↪5999999999999, 964.5999999999999, 964.5999999999999, 964.5999999999999, 964.
↪5999999999999, 964.5999999999999, 964.5999999999999, 964.5999999999999, 964.
↪5999999999999, 964.5999999999999, 964.5999999999999, 964.5999999999999, 964.
↪5999999999999, 964.5999999999999, 964.5999999999999, 964.5999999999999, 964.
↪5999999999999, 964.5999999999999, 964.5999999999999, 964.5999999999999, 964.
↪5999999999999, 964.5999999999999, 964.5999999999999, 964.5999999999999, 964.
↪5999999999999, 964.5999999999999, 964.5999999999999, 964.5999999999999, 964.
↪5999999999999, 964.5999999999999, 964.5999999999999, 964.5999999999999, 964.
↪5999999999999, 964.5999999999999, 964.5999999999999, 964.5999999999999, 964.
↪5999999999999, 964.5999999999999, 964.5999999999999, 964.5999999999999, 964.
↪5999999999999, 964.5999999999999, 964.5999999999999, 964.5999999999999, 964.
↪5999999999999, 964.5999999999999, 964.5999999999999, 964.5999999999999, 964.
↪5999999999999, 964.5999999999999, 964.5999999999999, 964.5999999999999, 964.
↪5999999999999, 964.5999999999999, 964.5999999999999, 964.5999999999999, 964.
↪5999999999999, 964.5999999999999, 964.5999999999999, 964.5999999999999, 964.
↪5999999999999, 964.5999999999999, 964.5999999999999, 964.5999999999999, 964.
↪5999999999999, 964.5999999999999, 964.5999999999999, 964.5999999999999, 964.
↪5999999999999, 964.5999999999999, 964.5999999999999, 964.5999999999999, 964.
↪5999999999999, 964.5999999999999, 964.5999999999999, 964.5999999999999, 964.
↪5999999999999, 964.5999999999999, 964.5999999999999, 964.5999999999999, 964.
↪5999999999999, 964.5999999999999, 964.5999999999999, 964.5999999999999, 964.
↪5999999999999, 964.5999999999999, 964.5999999999999, 964.5999999999999, 964.
↪5999999999999, 964.5999999999999, 964.5999999999999, 964.5999999999999, 964.
↪5999999999999, 964.5999999999999, 964.5999999999999, 964.5999999999999, 964.
↪5999999999999, 964.5999999999999, 964.5999999999999, 964.5999999999999, 964.
↪5999999999999, 964.5999999999999, 965.51, 965.51, 965.51, 965.51, 965.51, 965.51,␣
↪965.51, 965.51, 965.51]
```

(continues on next page)

(continued from previous page)

```
------------------------------------------------------------
duty_cycle_min: 27.3
duty_cycle_max: 964.6

#### example output right wheel, speed 0.2

Starting measurements for: 120 seconds.------------------------------------------------
 ↪-----------Ascending sorted distinct duty cycle values: [30.939999999999998, 31.
 ↪849999999999998, 35.489999999999995, 36.4, 40.04, 40.949999999999996, 41.86, 44.
 ↪589999999999996, 45.5, 46.41, 49.13999999999999, 50.05, 50.959999999999994, 53.69,␣
 ↪54.599999999999994, 55.51, 58.239999999999995, 59.15, 60.059999999999995, 62.79, 63.
 ↪699999999999996, 64.61, 67.33999999999999, 68.25, 69.16, 71.89, 72.8, 73.71, 76.44,␣
 ↪77.35, 78.25999999999999, 80.99, 81.89999999999999, 82.80999999999999, 85.
 ↪53999999999999, 86.44999999999999, 87.36, 90.08999999999999, 91.0, 91.91, 94.64, 95.
 ↪55, 96.46, 99.19, 100.1, 101.00999999999999, 103.74, 104.64999999999999, 105.
 ↪55999999999999, 108.28999999999999, 109.19999999999999, 110.11, 112.83999999999999,␣
 ↪113.74999999999999, 114.66, 117.38999999999999, 118.3, 119.21, 121.94, 122.85, 123.
 ↪75999999999999, 126.49, 127.39999999999999, 128.31, 131.04, 131.95, 132.
 ↪85999999999999, 135.59, 136.5, 137.41, 140.14, 141.04999999999998, 141.
 ↪95999999999998, 144.69, 145.6, 146.51, 149.23999999999998, 150.14999999999998, 151.
 ↪06, 153.79, 154.7, 155.60999999999999, 158.33999999999997, 159.25, 160.16, 162.89,␣
 ↪163.79999999999998, 164.70999999999998, 167.44, 168.35, 169.26, 171.98999999999998,␣
 ↪172.89999999999998, 173.81, 176.54, 177.45, 178.35999999999999, 181.08999999999997,␣
 ↪182.0, 182.91, 185.64, 186.54999999999998, 187.45999999999998, 190.19, 191.1, 192.
 ↪01, 194.73999999999998, 195.64999999999998, 196.55999999999997, 199.29, 200.2, 201.
 ↪10999999999999, 203.83999999999997, 204.75, 205.66, 208.39, 209.29999999999998, 210.
 ↪20999999999998, 212.94, 213.85, 214.76, 217.48999999999998, 218.39999999999998, 219.
 ↪30999999999997, 222.04, 222.95, 223.85999999999999, 226.58999999999997, 227.
 ↪49999999999997, 228.41, 231.14, 232.04999999999998, 232.95999999999998, 235.69, 236.
 ↪6, 237.51, 240.23999999999998, 241.14999999999998, 242.05999999999997, 244.79, 245.
 ↪7, 246.60999999999999, 249.33999999999997, 250.24999999999997, 251.16, 253.89, 254.
 ↪79999999999998, 255.70999999999998, 258.44, 259.34999999999997, 260.26, 262.99, 263.
 ↪9, 264.81, 267.53999999999996, 268.45, 269.35999999999996, 272.09, 273.0, 273.
 ↪90999999999997, 276.64, 277.54999999999995, 278.46, 281.19, 282.09999999999997, 283.
 ↪01, 285.73999999999995, 286.65, 287.56, 290.28999999999996, 291.2, 292.
 ↪10999999999996, 294.84, 295.75, 296.65999999999997, 299.39, 300.29999999999995, 301.
 ↪21, 303.94, 304.84999999999997, 305.76, 308.48999999999995, 309.4, 310.31, 313.
 ↪03999999999996, 313.95, 314.85999999999996, 317.59, 318.5, 319.40999999999997, 322.
 ↪14, 323.04999999999995, 323.96, 326.69, 327.59999999999997, 328.51, 331.
 ↪23999999999995, 332.15, 333.06, 335.78999999999996, 336.7, 337.60999999999996, 340.
 ↪34, 341.25, 342.15999999999997, 344.89, 345.79999999999995, 346.71, 349.44, 350.
 ↪34999999999997, 351.26, 353.98999999999995, 354.9, 355.81, 358.53999999999996, 359.
 ↪45, 360.35999999999996, 363.09, 364.0, 364.90999999999997, 367.64, 368.
 ↪54999999999995, 369.46, 372.19, 373.09999999999997, 374.01, 376.73999999999995, 377.
 ↪65, 378.56, 381.28999999999996, 382.2, 383.10999999999996, 385.84, 386.75, 387.
 ↪65999999999997, 390.39, 391.29999999999995, 392.21,394.94, 395.84999999999997, 396.
 ↪76, 399.48999999999995, 400.4, 401.30999999999995, 404.03999999999996, 404.95, 405.
 ↪85999999999996, 408.59, 409.5, 410.40999999999997, 413.14, 414.04999999999995, 414.
 ↪96, 417.69, 418.59999999999997, 419.51, 422.23999999999995, 423.15, 424.
 ↪05999999999995, 426.78999999999996, 427.7, 428.60999999999996, 431.34, 432.
 ↪24999999999994, 433.15999999999997, 435.89, 436.79999999999995, 437.71, 440.44, 441.
 ↪34999999999997, 442.26, 444.98999999999995, 445.9, 446.80999999999995, 449.
 ↪53999999999996, 450.45, 451.35999999999996, 454.09, 454.99999999999994, 455.
 ↪90999999999997, 458.64, 459.54999999999995, 460.46, 463.18999999999994, 464.
 ↪09999999999997, 465.01, 467.73999999999995, 468.65, 469.55999999999995, 472.
 ↪28999999999996, 473.2, 474.10999999999996, 476.84, 477.74999999999994, 478.
 ↪65999999999997, 481.39, 482.29999999999995, 485.93999999999994, 486.84999999999997,␣
 ↪487.76, 490.48999999999995, 491.4, 492.30999999999995, 495.03999999999996, 495.95,␣
 ↪496.85999999999996, 499.59, 500.49999999999994, 501.40999999999997, 504.14, 505.
 ↪04999999999995, 505.96, 508.68999999999994, 509.59999999999997, 510.51, 513.24, 514.
 ↪15, 515.06, 517.79, 518.6999999999999, 519.61, 522.3399999999999, 523.25, 524.16,␣
 ↪526.89, 527.8, 528.7099999999999, 531.4399999999999, 532.3499999999999, 533.26, 535.
 ↪99, 536.9, 537.81, 540.54, 541.4499999999999, 542.36, 545.0899999999999, 546.0, 546.
 ↪91, 549.64, 550.55, 551.4599999999999, 554.1899999999999, 555.0999999999999, 556.01,
```

(continues on next page)

(continued from previous page)

```
----------------------------------------------------------
Ascending counted, sorted and rounded distinct differences between duty cycle values:␣
↪Counter({0.91: 412, 2.73: 202, 3.64: 6})
----------------------------------------------------------
Smallest 250 values: [30.939999999999998, 30.939999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998,31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 35.489999999999995, 35.489999999999995, 35.489999999999995, 35.
↪489999999999995, 35.489999999999995, 36.4, 36.4, 36.4, 36.4, 36.4, 36.4, 36.4, 36.4,
↪ 36.4, 36.4, 36.4, 36.4, 36.4, 36.4, 36.4, 36.4, 36.4, 36.4, 36.4, 36.4, 36.4]
```

(continues on next page)

(continued from previous page)

```
-----------------------------------------------------------
Biggest 250 values: [978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999,978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 979.16, 979.16, 979.16, 979.16,
↪ 979.16, 979.16, 979.16, 982.8]
```

(continues on next page)

(continued from previous page)

```
------------------------------------------------------------
duty_cycle_min: 31.85
duty_cycle_max: 978.25

#### example output right wheel, speed -0.2

Starting measurements for: 120 seconds.------------------------------------------------
 ↪-----------Ascending sorted distinct duty cycle values: [26.389999999999997, 27.
 ↪299999999999997, 31.849999999999998, 32.76, 35.489999999999995, 36.4, 37.
 ↪309999999999995, 40.04, 40.949999999999996, 41.86, 44.589999999999996, 45.5, 46.41,␣
 ↪49.13999999999999, 50.05, 50.959999999999994, 53.69, 54.599999999999994, 55.51, 58.
 ↪239999999999995, 59.15, 60.059999999999995, 62.79, 63.699999999999996, 64.61, 67.
 ↪33999999999999, 68.25, 69.16, 71.89, 72.8, 73.71, 76.44, 77.35, 78.25999999999999,␣
 ↪80.99, 81.89999999999999, 82.80999999999999, 85.53999999999999, 86.44999999999999,␣
 ↪87.36, 90.08999999999999, 91.0, 91.91, 94.64, 95.55, 96.46, 99.19, 100.1, 101.
 ↪00999999999999, 103.74, 104.64999999999999, 105.55999999999999, 108.28999999999999,␣
 ↪109.19999999999999, 110.11, 112.83999999999999, 113.74999999999999, 114.66, 117.
 ↪38999999999999, 118.3, 119.21, 121.94, 122.85, 123.75999999999999, 126.49, 127.
 ↪39999999999999, 128.31, 131.04, 131.95, 132.85999999999999, 135.59, 136.5, 137.41,␣
 ↪140.14, 141.04999999999998, 141.95999999999998, 144.69, 145.6, 146.51, 149.
 ↪23999999999998, 150.14999999999998, 151.06, 153.79, 154.7, 155.60999999999999, 158.
 ↪33999999999997, 159.25, 160.16, 162.89, 163.79999999999998, 164.70999999999998, 167.
 ↪44, 168.35, 169.26, 171.98999999999998,172.89999999999998, 173.81, 176.54, 177.45,␣
 ↪178.35999999999999, 181.08999999999997, 182.0, 182.91, 185.64, 186.54999999999998,␣
 ↪187.45999999999998, 190.19, 191.1, 192.01, 194.73999999999998, 195.64999999999998,␣
 ↪196.55999999999997, 199.29, 200.2, 201.10999999999999, 203.83999999999997, 204.75,␣
 ↪205.66, 208.39, 209.29999999999998, 210.20999999999998, 212.94, 213.85, 214.76, 217.
 ↪48999999999998, 218.39999999999998, 219.30999999999997, 222.04, 222.95, 223.
 ↪85999999999999, 226.58999999999997, 227.49999999999997, 228.41, 231.14, 232.
 ↪04999999999998, 232.95999999999998, 235.69, 236.6, 240.23999999999998, 241.
 ↪14999999999998, 242.05999999999997, 244.79, 245.7, 246.60999999999999, 249.
 ↪33999999999997, 250.24999999999997, 251.16, 253.89, 254.79999999999998, 255.
 ↪70999999999998, 258.44, 259.34999999999997, 260.26, 262.99, 263.9, 264.81, 267.
 ↪53999999999996, 268.45, 269.35999999999996, 272.09, 273.0, 273.90999999999997, 276.
 ↪64, 277.54999999999995, 278.46, 281.19, 282.09999999999997, 283.01, 285.
 ↪73999999999995, 286.65, 287.56, 290.28999999999996, 291.2, 292.10999999999996, 294.
 ↪84, 295.75, 296.65999999999997, 299.39, 300.29999999999995, 301.21, 303.94, 304.
 ↪84999999999997, 305.76, 308.48999999999995, 309.4, 310.31, 313.03999999999996, 313.
 ↪95, 314.85999999999996, 317.59, 318.5, 319.40999999999997, 322.14, 323.
 ↪04999999999995, 323.96, 326.69, 327.59999999999997, 328.51, 331.23999999999995, 332.
 ↪15, 333.06, 335.78999999999996, 336.7, 337.60999999999996,340.34, 341.25, 342.
 ↪15999999999997, 344.89, 345.79999999999995, 346.71, 349.44, 350.34999999999997, 351.
 ↪26, 353.98999999999995, 354.9, 355.81, 358.53999999999996, 359.45, 360.
 ↪35999999999996, 363.09, 364.0, 364.90999999999997, 367.64, 368.54999999999995, 369.
 ↪46, 372.19, 373.09999999999997, 374.01, 376.73999999999995, 377.65, 378.56, 381.
 ↪28999999999996, 382.2, 383.10999999999996, 385.84, 386.75, 387.65999999999997, 390.
 ↪39, 391.29999999999995, 392.21, 394.94, 395.84999999999997, 396.76, 399.
 ↪48999999999995, 400.4, 401.30999999999995, 404.03999999999996, 404.95, 405.
 ↪85999999999996, 408.59, 409.5, 410.40999999999997, 414.04999999999995, 414.96, 417.
 ↪69, 418.59999999999997, 419.51, 422.23999999999995, 423.15, 424.05999999999995, 426.
 ↪78999999999996, 427.7, 428.60999999999996, 431.34, 432.24999999999994, 433.
 ↪15999999999997, 435.89, 436.79999999999995, 437.71, 440.44, 441.34999999999997, 442.
 ↪26, 444.98999999999995, 445.9, 446.80999999999995, 449.53999999999996, 450.45, 451.
 ↪35999999999996, 454.09, 454.99999999999994, 455.90999999999997, 458.64, 459.
 ↪54999999999995, 460.46, 463.18999999999994, 464.09999999999997, 465.01, 467.
 ↪73999999999995, 468.65, 469.55999999999995, 472.28999999999996, 473.2, 474.
 ↪10999999999996, 476.84, 477.74999999999994, 478.65999999999997, 481.39, 482.
 ↪29999999999995, 483.21, 485.93999999999994, 486.84999999999997, 487.76, 490.
 ↪48999999999995, 491.4, 492.30999999999995, 495.03999999999996, 495.95, 496.
 ↪85999999999996, 499.59, 500.49999999999994, 501.40999999999997, 504.14, 505.
 ↪04999999999995, 505.96, 508.68999999999994, 509.59999999999997, 510.51, 513.24, 514.
 ↪15, 515.06, 517.79, 518.6999999999999, 519.61, 522.3399999999999, 523.25, 524.16,␣
 ↪526.89, 527.8, 528.7099999999999, 531.4399999999999, 532.3499999999999, 533.26, 535.
 ↪99, 536.9, 537.81, 540.54, 541.4499999999999, 542.36, 545.0899999999999, 546.0, 546.
```

(continues on next page)

(continued from previous page)

```
----------------------------------------------------------
Ascending counted, sorted and rounded distinct differences between duty cycle values:␣
↪Counter({0.91: 414, 2.73: 205, 3.64: 2, 4.55: 1})
----------------------------------------------------------
Smallest 250 values: [26.389999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997,27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 27.
↪299999999999997, 27.299999999999997, 27.299999999999997, 27.299999999999997, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
↪849999999999998, 31.849999999999998, 31.849999999999998, 31.849999999999998, 31.
```

(continues on next page)

(continued from previous page)

```
--------------------------------------------------------
Biggest 250 values: [973.6999999999999, 973.6999999999999, 973.6999999999999, 973.
↪6999999999999, 973.6999999999999, 973.6999999999999, 973.6999999999999, 973.
↪6999999999999, 973.6999999999999, 973.6999999999999, 973.6999999999999, 973.
↪6999999999999, 973.6999999999999, 973.6999999999999, 973.6999999999999, 973.
↪6999999999999, 973.6999999999999, 973.6999999999999, 973.6999999999999, 973.
↪6999999999999, 973.6999999999999, 973.6999999999999, 973.6999999999999, 973.
↪6999999999999, 973.6999999999999, 973.6999999999999, 973.6999999999999, 973.
↪6999999999999, 973.6999999999999, 973.6999999999999, 973.6999999999999, 973.
↪6999999999999, 973.6999999999999, 973.6999999999999, 973.6999999999999, 973.
↪6999999999999, 973.6999999999999, 973.6999999999999, 973.6999999999999, 973.
↪6999999999999, 973.6999999999999, 973.6999999999999, 973.6999999999999, 973.
↪6999999999999, 973.6999999999999, 973.6999999999999, 973.6999999999999, 973.
↪6999999999999, 973.6999999999999, 973.6999999999999, 973.6999999999999, 973.
↪6999999999999, 973.6999999999999, 973.6999999999999, 973.6999999999999, 973.
↪6999999999999, 973.6999999999999, 973.6999999999999, 973.6999999999999, 973.
↪6999999999999, 973.6999999999999, 973.6999999999999, 973.6999999999999, 973.
↪6999999999999, 973.6999999999999, 973.6999999999999, 973.6999999999999, 973.
↪6999999999999, 973.6999999999999, 973.6999999999999, 973.6999999999999, 973.
↪6999999999999, 973.6999999999999, 973.6999999999999, 973.6999999999999, 973.
↪6999999999999, 973.6999999999999, 973.6999999999999, 973.6999999999999, 973.
↪6999999999999, 973.6999999999999, 973.6999999999999, 973.6999999999999, 973.
↪6999999999999, 973.6999999999999, 973.6999999999999, 973.6999999999999, 973.
↪6999999999999, 973.6999999999999, 973.6999999999999, 973.6999999999999, 973.
↪6999999999999, 973.6999999999999, 973.6999999999999, 973.6999999999999, 973.
↪6999999999999, 973.6999999999999, 973.6999999999999, 973.6999999999999, 973.
↪6999999999999, 973.6999999999999, 973.6999999999999, 973.6999999999999, 973.
↪6999999999999, 973.6999999999999, 973.6999999999999, 973.6999999999999, 973.
↪6999999999999, 973.6999999999999, 973.6999999999999, 973.6999999999999, 973.
↪6999999999999, 973.6999999999999, 973.6999999999999, 973.6999999999999, 973.
↪6999999999999, 973.6999999999999, 973.6999999999999, 973.6999999999999, 973.
↪6999999999999, 973.6999999999999, 973.6999999999999, 973.6999999999999, 973.
↪6999999999999, 973.6999999999999, 973.6999999999999, 973.6999999999999, 973.
↪6999999999999, 973.6999999999999, 973.6999999999999, 973.6999999999999, 973.
↪6999999999999, 973.6999999999999, 973.6999999999999, 973.6999999999999, 973.
↪6999999999999, 973.6999999999999, 973.6999999999999, 973.6999999999999, 973.
↪6999999999999, 973.6999999999999, 973.6999999999999, 973.6999999999999, 973.
↪6999999999999, 973.6999999999999, 973.6999999999999, 973.6999999999999, 973.
↪6999999999999, 973.6999999999999, 973.6999999999999, 973.6999999999999, 973.
↪6999999999999, 973.6999999999999, 973.6999999999999, 973.6999999999999, 973.
↪6999999999999, 973.6999999999999, 973.6999999999999, 973.6999999999999, 973.
↪6999999999999, 973.6999999999999, 973.6999999999999, 973.6999999999999, 973.
↪6999999999999, 973.6999999999999, 973.6999999999999, 973.6999999999999, 973.
↪6999999999999, 973.6999999999999, 973.6999999999999, 973.6999999999999, 973.
↪6999999999999, 973.6999999999999, 973.6999999999999, 973.6999999999999, 973.
↪6999999999999, 973.6999999999999, 973.6999999999999, 973.6999999999999, 973.
↪6999999999999, 973.6999999999999, 973.6999999999999, 973.6999999999999, 973.
↪6999999999999, 973.6999999999999, 973.6999999999999, 973.6999999999999, 973.
↪6999999999999, 973.6999999999999, 973.6999999999999, 973.6999999999999, 973.
↪6999999999999, 973.6999999999999, 973.6999999999999, 973.6999999999999,973.
↪6999999999999, 973.6999999999999, 973.6999999999999, 973.6999999999999, 973.
↪6999999999999, 974.6099999999999, 974.6099999999999, 974.6099999999999, 974.
↪6099999999999, 974.6099999999999, 974.6099999999999, 974.6099999999999, 974.
↪6099999999999, 974.6099999999999, 974.6099999999999, 974.6099999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999, 978.2499999999999, 978.
↪2499999999999, 978.2499999999999, 978.2499999999999]
```

(continues on next page)

(continued from previous page)

```
------------------------------------------------------------
duty_cycle_min: 27.3
duty_cycle_max: 978.25
```

In this case, for the left wheel for `duty_cycle_min` / `dcMin` 27.3 should be chosen, so the smallest out of 27.3 and 31.85. For `duty_cycle_max` / `dcMax` 969.15 should be chosen, so the biggest out of 964.6 and 969.15. For the right wheel, for `duty_cycle_min` / `dcMin` 27.3 and for `duty_cycle_max` / `dcMax` 978.25 accordingly.

3.2 Emergency stop

The following code sets the speed of the two used Parallax Feedback 360° High-Speed Servos 360_data_sheet[1] back to zero to stop both wheels. This might be needed e.g. if a script raises an exception and stops executing before setting the speed of the servos back to zero. In this case, the servos will continue rotating with the last set speed. This example is included as `emergency_stop.py` .

```
import time

import pigpio

import lib_para_360_servo

#define GPIO for each servo to write to
gpio_l = 17
gpio_r = 27

pi = pigpio.pi()

servo_l = lib_para_360_servo.write_pwm(pi = pi, gpio = gpio_l)

servo_r = lib_para_360_servo.write_pwm(pi = pi, gpio = gpio_r)

#buffer time for initializing everything
time.sleep(1)

servo_l.set_speed(0)
servo_r.set_speed(0)

#http://abyz.me.uk/rpi/pigpio/python.html#stop
pi.stop()
```

3.3 Moving the robot

The following code lets the robot turning four times 45 degree to the left, then moving 20 cm (200 mm) forwards, then 20 cm backwards and in the end turning two times 90 degree to the right. This example is included as `move_robot.py` .

Note: The PID controller values have a strong influence on the the robots movement. Therefore, try out different values than the default once if the robot is not moving as exspected and also to get a feeling of their influence.

Note: Sometimes the end of a movement takes some time. One reason could be noise in the position measurement of the wheels and therefore not correctly recognizing the reached set-point. So do not be confused if this happens and the robot takes some time and seems not to proper function anmyore. Just be patient. This might be solved with increasing the deadband of the outer PID controllers, which on the other hand would also decrease the accuracy of the movement. At the moment a rather small deadband is chosen. A simple but effective workaround is to give the robot a little poke and let it reach the set-point again. See *lib_motion.control.move()* for more informations.

Note: Sometimes the speed changes abruptly. This might be caused by noise in the position measurement of the wheels from which the rotation speed measurement (ticks/s) is calculated indirectly. Increasing the sliding window size might stabilize it. But this would also increase the delay between calculated and real speed of the wheels and therefore also increase the response time of the speed controllers. At the moment a rather small window size is chosen. See *lib_motion.control.move()* for more informations.

```
import pigpio

import lib_motion

pi = pigpio.pi()

robot = lib_motion.control(pi = pi)

a = 0
while a < 4:
    robot.turn(45)
    a+=1

robot.straight(200)
robot.straight(-200)

a = 0
while a < 2:
    robot.turn(-90)
    a+=1

#http://abyz.me.uk/rpi/pigpio/python.html#callback
robot.cancel()

#http://abyz.me.uk/rpi/pigpio/python.html#stop
pi.stop()
```

3.4 Moving standard servo

The following code stears the standard servo stand_data_sheet[2] . First to the middle position, then to the max right, then to max left and finally to 45 degree (regarding reached max left and max right). For more informations, see *lib_scanner.para_standard_servo* . This example is included as `move_stand_servo.py` .

```
import time

import pigpio
```

(continues on next page)

[2] https://www.parallax.com/sites/default/files/downloads/900-00005-Standard-Servo-Product-Documentation-v2.2.pdf

(continued from previous page)

```

import lib_scanner

pi = pigpio.pi()

servo = lib_scanner.para_standard_servo(pi = pi, gpio = 22)

servo.middle_position()
time.sleep(2)
servo.max_right()
time.sleep(2)
servo.max_left()
time.sleep(2)
servo.set_position(degree = 45)

#http://abyz.me.uk/rpi/pigpio/python.html#stop
pi.stop()
```

3.5 Scanning the surrounding

The following code scans the surrounding of the robot in all five default angles and prints out the result. This example is included as `scanning.py` .

> **Warning:** Make sure that the `min_pw` and `max_pw` values are carefully tested **before** using this example, see **Warning** in *`lib_scanner.para_standard_servo`* . The passed values `min_pw` and `max_pw` for the created ranger object are just valid for the demo implementation!

```
import pigpio

import lib_scanner

pi = pigpio.pi()

"""
.. warning::

    Make sure that the ``min_pw`` and ``max_pw`` values are carefully tested
    **before** using this example, see **Warning** in
    :class:`lib_scanner.para_standard_servo` . The passed values ``min_pw``
    and ``max_pw`` for the created ranger object are just valid for the
    demo implementation!
"""

ranger = lib_scanner.scanner(pi = pi, min_pw=600, max_pw=2350)

distances = ranger.read_all_angles()
print(distances)

#http://abyz.me.uk/rpi/pigpio/python.html#callback
ranger.cancel()

```

(continues on next page)

(continued from previous page)

```
#http://abyz.me.uk/rpi/pigpio/python.html#stop
pi.stop()
```

3.6 Simple collision avoiding algorithm

The following code implements a simple collision avoiding algorithm. The robot will turn 45 degree to the left if there is any obstacle closer than 40 cm at the five default measuring angles. If not, the robot will drive 20 cm forward. This example is included as `no_collision.py` .

Warning: Make sure that the `min_pw` and `max_pw` values are carefully tested **before** using this example, see **Warning** in *lib_scanner.para_standard_servo* . The passed values `min_pw` and `max_pw` for the created ranger object are just valid for the demo implementation!

```
import pigpio

import lib_motion
import lib_scanner

#initialize one pigpio.pi() instance to be used by all lib_*
pi = pigpio.pi()

robot = lib_motion.control(pi = pi)

"""
.. warning::

    Make sure that the ``min_pw`` and ``max_pw`` values are carefully tested
    **before** using this example, see **Warning** in
    :class:`lib_scanner.para_standard_servo` . The passed values ``min_pw``
    and ``max_pw`` for the created ranger object are just valid for the
    demo implementation!
"""

ranger = lib_scanner.scanner(pi = pi, min_pw=600, max_pw=2350)

while True:

    distances = ranger.read_all_angles()
    list_dist = list(distances.values())
    if any(t<0.4 for t in list_dist):
        robot.turn(45)
    else:
        robot.straight(200)

#http://abyz.me.uk/rpi/pigpio/python.html#callback
robot.cancel()
ranger.cancel()

#http://abyz.me.uk/rpi/pigpio/python.html#stop
pi.stop()
```

3.7 References

CHAPTER 4

360pibot API

In this section the use of each module is documented. In the source code it is commented, if necessary, how each part of the code is working and what it is intended to do.

4.1 lib_scanner

Module for making measurements with a HC-SR04[2] ultrasonic sensor and rotating it with a Parallax Standard Servo stand_data_sheet[4] .

This module includes three classes. One for making the measurements with an HC-SR04[2] ultrasonic sensor *lib_scanner.hcsr04*, one for stearing a Parallax Standard Servo stand_data_sheet[4] *lib_scanner.para_standard_servo* and one which combines the first two to scan the surrounding *lib_scanner.scanner* .

class lib_scanner.**hcsr04**(*pi*, *trigger*, *echo*, *pulse_len=15*)

Makes measurements with a HC-SR04[2] ultrasonic sensor.

This class allows making measurements with a HC-SR04[2] ultrasonic sensor. A trigger signal will be sent to a defined GPIO Pin `trigger` and a PWM will be recieved on a defined GPIO Pin `echo`. With the recieved PWM the distance to an object is calculated.

Parameters

- **pi** (*pigpio.pi*) – Instance of a pigpio.pi() object.
- **trigger** (*int*) – GPIO identified by their Broadcom number, see elinux.org[1] . To this GPIO the trigger pin of the HC-SR04[2] has to be connected.
- **echo** (*int*) – GPIO identified by their Broadcom number, see elinux.org[1] . To this GPIO the echo pin of the HC-SR04[2] has to be connected.

[2] https://cdn.sparkfun.com/assets/b/3/0/b/a/DGCH-RED_datasheet.pdf
[4] https://www.parallax.com/sites/default/files/downloads/900-00005-Standard-Servo-Product-Documentation-v2.2.pdf
[1] https://elinux.org/RPi_Low-level_peripherals#Model_A.2B.2C_B.2B_and_B2

- **pulse_len** (*int, float*) – Defines in microseconds the length of the pulse which is sent on the trigger GPIO. **Default:** 15, taken from the data sheet (10µs) and added 50%, to have a buffer to surely trigger the measurement.

cancel()
Cancel the started callback function.

This method cancels the started callback function if initializing an object. As written in the pigpio callback[3] documentation, the callback function may be cancelled by calling the cancel function after the created instance is not needed anymore.

read(*temp_air=20*, *upper_limit=4*, *number_of_sonic_bursts=8*, *added_buffer=2*, *debug=False*)
Measures the distance to an object.

This method triggers a measurement, does all the calculations and returns the distance in meters.

Parameters

- **temp_air** (*int, float*) – Temperature of the air in degree celsius. **Default:** 20.
- **upper_limit** (*int, float*) – The upper measurement limit in meters. **Default:** 4, upper limit taken from the data sheet HC-SR04[2] .
- **number_of_sonic_bursts** (*int*) – The number of sonic bursts the sensor will make. **Default:** 8, taken from the data sheet HC-SR04[2] .
- **added_buffer** (*int, float*) – The added safety buffer for waiting for the distance measurement to complete. **Default:** 2, so 100% safety buffer.
- **debug** (*bool*) – Controls if debugging printouts are made or not. For more details, have a look at the source code. **Default:** False, so no printouts are made.

Returns Measured distance in meters.

Return type float

class lib_scanner.**para_standard_servo**(*pi*, *gpio*, *min_pw=1000*, *max_pw=2000*, *min_degree=-90*, *max_degree=90*)
Stears a servo, in this case a Parallax Standard Servo stand_data_sheet[4] .

This class stears a Parallax Standard Servo and should also work with other servos which have a 50Hz PWM for setting the position. The position of the Parallax Standard Servo can be set between -90 (`min_degree`) and +90 (`max_degree`) degree.

Note: `min_pw` and `max_pw` might needed to be interchanged, depending on if `min_pw` is moving the servo to max_right/clockwise or max_left/counter-clockwise, see methods *`max_left()`* and *`max_right()`*. *`max_right()`* -> `min_pw` should let the servo rotate clockwise.

Warning: Be carefull with setting the min and max pulsewidth! Test carefully `min_pw` and `max_pw` before setting them. Wrong values can damage the servo, see set_servo_pulsewidth[5] !!!

Parameters

- **pi** (*pigpio.pi*) – Instance of a pigpio.pi() object.
- **gpio** (*int*) – GPIO identified by their Broadcom number, see elinux.org[1] . To this GPIO the signal wire of the servo has to be connected.

[3] http://abyz.me.uk/rpi/pigpio/python.html#callback
[5] http://abyz.me.uk/rpi/pigpio/python.html#set_servo_pulsewidth

- **min_pw** (*int*) – Min pulsewidth, see **Warning**, carefully test the value before! **Default:** 1000, taken from set_servo_pulsewidth[5] .
- **max_pw** (*int*) – Max pulsewidth, see **Warning**, carefully test the value before! **Default:** 2000, taken from set_servo_pulsewidth[5] .
- **min_degree** (*int*) – Min degree which the servo is able to move. **Default:** -90, taken from stand_data_sheet[4] .
- **max_degree** (*int*) – Max degree which the servo is able to move. **Default:** +90, taken from stand_data_sheet[4] .

max_left()
Sets the position of the servo to -90 degree, so `min_degree` (max left, counter-clockwise).

max_right()
Sets the position of the servo to 90 degree, so `max_degree` (max right, clockwise).

middle_position()
Sets the position of the servo to 0 degree, so middle position.

set_position(*degree*)
Sets position of the servo in degree.

This method sets the servo position in degree. Minus is to the left, plus to the right.

Parameters **degree** (*int*, *float*) – Should be between `min_degree` (max left) and `max_degree` (max right), otherwise the value will be limited to those values.

set_pw(*pulse_width*)
Sets pulsewidth of the PWM.

This method allows setting the pulsewidth of the PWM directly. This can be used to test which `min_pw` and `max_pw` are appropriate. For this the `min_pw` and `max_pw` are needed to be set very small and very big, so that they do not limit the set pulsewidth. Normally they are used to protect the servo by limiting the pulsewidth to a certain range.

> **Warning:** Be carefull with setting the min and max pulsewidth! Test carefully `min_pw` and `max_pw` before setting them. Wrong values can damage the servo, see set_servo_pulsewidth[5] !!!

Parameters **pulsewidth** (*int*, *float*) – Pulsewidth of the PWM signal. Will be limited to `min_pw` and `max_pw`.

class lib_scanner.**scanner**(*pi, trigger=6, echo=5, pulse_len=15, temp_air=20, upper_limit=4, number_of_sonic_bursts=8, added_buffer=2, gpio=22, min_pw=1000, max_pw=2000, min_degree=-90, max_degree=90, angles=[-90, -45, 0, 45, 90], time_servo_reach_position=3, debug=False*)
Scans the surrounding with the help of the *hcsr04* and *para_standard_servo* classes.

This class stears the servo position and triggers measurements with the ultrasonic sensor. With a passed `list`, measurements will be made at the defined positions. A `dict` will be returned with the measured distances at the defined positions.

> **Warning:** Be carefull with setting the min and max pulsewidth! Test carefully `min_pw` and `max_pw` before setting them. Wrong values can damage the servo, see set_servo_pulsewidth[5] !!!

Parameters

- **pi** (*pigpio.pi*) – Instance of a pigpio.pi() object.
- **trigger** (*int*) – GPIO identified by their Broadcom number, see elinux.org[1] . To this GPIO the trigger pin of the HC-SR04[2] has to be connected.
- **echo** (*int*) – GPIO identified by their Broadcom number, see elinux.org[1] . To this GPIO the echo pin of the HC-SR04[2] has to be connected.
- **pulse_len** (*int, float*) – Defines in microseconds the length of the pulse, which is sent on the trigger GPIO. **Default:** 15, taken from the data sheet (10µs) and added 50%, to have a buffer to surely trigger the measurement.
- **temp_air** (*int, float*) – Temperature of the air in degree celsius. **Default:** 20.
- **upper_limit** (*int, float*) – The upper measurement limit in meters. **Default:** 4, upper limit taken from the data sheet HC-SR04[2] .
- **number_of_sonic_bursts** (*int*) – The number of sonic bursts the sensor will make. **Default:** 8, taken from the data sheet HC-SR04[2] .
- **added_buffer** (*int, float*) – The added safety buffer for waiting for the distance measurement. **Default:** 2, so 100% safety buffer.
- **gpio** (*int*) – GPIO identified by their Broadcom number, see elinux.org[1] . To this GPIO the signal wire of the servo has to be connected.
- **min_pw** (*int*) – Min pulsewidth, see **Warning**, carefully test the value before! **Default:** 1000, taken from set_servo_pulsewidth[5] .
- **max_pw** (*int*) – Max pulsewidth, see **Warning**, carefully test the value before! **Default:** 2000, taken from set_servo_pulsewidth[5] .
- **min_degree** (*int*) – Min degree which the servo is able to move. **Default:** -90, taken from stand_data_sheet[4] .
- **max_degree** (*int*) – Max degree which the servo is able to move. **Default:** +90, taken from stand_data_sheet[4] .
- **angles** (*list*) – List of positions where the servo moves to and the ultrasonic sensor will make measurements.
- **time_servo_reach_position** (*int, float*) – Time in seconds to wait until the servo moves from one to another position. This needs to be tested for each servo. **Default:** 3, this should be sufficient to safely (incl. lot of safety buffer) reach each position before the measurement is made. If the servo is not oscillating after reaching each position, even a value of 0.35 was working fine with the demo implementation.
- **debug** (*bool*) – Controls if debugging printouts and measurements are made or not. For more details, have a look at the source code. **Default:** False, so no printouts and measurements are made.

cancel()

Cancel the started callback function.

This method cancels the started callback function if initializing an object. As written in the pigpio callback[3] documentation, the callback function may be cancelled by calling the cancel function after the created instance is not needed anymore.

read_all_angles()

Moves servo and makes measurements at every defined position.

This method moves the servo to every position defined in `list angles` , makes a measurement there and afterwards returns a `dict` with the distance in meter for every position.

Returns Measured distances in meters for each position defined in `angles`.

Return type dict

4.2 lib_para_360_servo

Module for setting the speed and reading the position of a Parallax Feedback 360° High-Speed Servo 360_data_sheet[6] .

This module includes three classes. One for setting the speed *lib_para_360_servo.write_pwm* , one for reading the position *lib_para_360_servo.read_pwm* and one for calibrating a servo to determine the appropriate `dcMin` / `dcMax` values needed in *lib_motion* *lib_para_360_servo.calibrate_pwm* .

class lib_para_360_servo.**calibrate_pwm**(*pi*, *gpio*, *measurement_time=120*)

Calibrates a Parallax Feedback 360° High-Speed Servo with the help of the *read_pwm* class.

This class helps to find out the min and max duty cycle of the feedback signal of a servo. This values (`dcMin` / `dcMax`) are then needed in *lib_motion* to have a more precise measurement of the position. The experience has shown that each servo has slightly different min/max duty cycle values, different than the once provided in the data sheet 360_data_sheet[6] . Values smaller and bigger than the printed out once as "duty_cycle_min/duty_cycle_max" are outliers and should therefore not be considered. This can be seen in the printouts of smallest/biggest 250 values. There are sometimes a few outliers. Compare the printouts of different runs to get a feeling for it.

Note: The robot wheels must be able to rotate free in the air for calibration. Rotating forward or backward might sometimes give slightly different results for min/max duty cycle. Choose the smallest value and the biggest value out of the forward and backward runs. Do both directions three times for each wheel, with speed = 0.2 and -0.2. Then chose the values. The speed has to be set manually, see *Examples*.

Parameters

- **pi** (*pigpio.pi*) – Instance of a pigpio.pi() object.
- **gpio** (*int*) – GPIO identified by their Broadcom number, see elinux.org[1] . To this GPIO the feedback wire of the servo has to be connected.
- **measurement_time** (*int, float*) – Time in seconds for how long duty cycle values will be collected, so for how long the measurement will be made. **Default:** 120.

Returns Printouts of different measurements

At the moment, the period for a 910 Hz signal is hardcoded, as in *read_pwm()* .

Todo: Enable the class to be able to handle different signals, not just 910 Hz.

---

class lib_para_360_servo.**read_pwm**(*pi*, *gpio*)

Reads position of a Parallax Feedback 360° High-Speed Servo 360_data_sheet[6] .

This class reads the position of a Parallax Feedback 360° High-Speed Servo. At the moment, the period for a 910 Hz signal is hardcoded.

Parameters

- **pi** (*pigpio.pi*) – Instance of a pigpio.pi() object.

[6] https://www.parallax.com/sites/default/files/downloads/900-00360-Feedback-360-HS-Servo-v1.1.pdf

- **gpio** (*int*) – GPIO identified by their Broadcom number, see elinux.org[1] . To this GPIO the feedback wire of the servo has to be connected.

Todo: Enable the class to be able to handle different signals, not just 910 Hz.

---

cancel()
Cancel the started callback function.

This method cancels the started callback function if initializing an object. As written in the pigpio callback[3] documentation, the callback function may be cancelled by calling the cancel function after the created instance is not needed anymore.

read()
Returns the recent measured duty cycle.

This method returns the recent measured duty cycle.

Returns Recent measured duty cycle

Return type float

class lib_para_360_servo.**write_pwm**(*pi*, *gpio*, *min_pw=1280*, *max_pw=1720*, *min_speed=-1*, *max_speed=1*)
Steers a Parallax Feedback 360° High-Speed Servo 360_data_sheet[6] .

This class steers a Parallax Feedback 360° High-Speed Servo. Out of the speed range, defined by `min_speed` and `max_speed`, and the range of the pulsewidth, defined by `min_pw` and `max_pw`, the class allows setting the servo speed and automatically calculates the appropriate pulsewidth for the chosen speed value.

Note: `min_pw` and `max_pw` might needed to be interchanged, depending on if `min_pw` is moving the servo max_forward/clockwise or max_backwards/counter-clockwise, see methods *`max_forward()`* and *`max_backward()`*. *`max_forward()`* -> `min_pw` should let the servo rotate clockwise.

Warning: Be carefull with setting the min and max pulsewidth! Test carefully `min_pw` and `max_pw` before setting them. Wrong values can damage the servo, see set_servo_pulsewidth[5] !!!

Parameters

- **pi** (*pigpio.pi*) – Instance of a pigpio.pi() object.
- **gpio** (*int*) – GPIO identified by their Broadcom number, see elinux.org[1] . To this GPIO the control wire of the servo has to be connected.
- **min_pw** (*int*) – Min pulsewidth, see **Warning**, carefully test the value before! **Default:** 1280, taken from the data sheet 360_data_sheet[6] .
- **max_pw** (*int*) – Max pulsewidth, see **Warning**, carefully test the value before! **Default:** 1720, taken from the data sheet 360_data_sheet[6] .
- **min_speed** (*int*) – Min speed which the servo is able to move. **Default:** -1.
- **max_speed** (*int*) – Max speed which the servo is able to move. **Default:** 1.

max_backward()
Sets the speed of the servo to -1, so `min_speed` (max backwards, counter-clockwise)

max_forward()
Sets the speed of the servo to 1, so max_speed (max forward, clockwise)

set_pw(*pulse_width*)
Sets pulsewidth of the PWM.

This method allows setting the pulsewidth of the PWM directly. This can be used to test which min_pw and max_pw are appropriate. For this the min_pw and max_pw are needed to be set very small and very big, so that they do not limit the set pulsewidth. Normally they are used to protect the servo by limiting the pulsewidth to a certain range.

Warning: Be carefull with setting the min and max pulsewidth! Test carefully min_pw and max_pw before setting them. Wrong values can damage the servo, see set_servo_pulsewidth[5] !!!

Parameters **pulsewidth** (*int*, *float*) – Pulsewidth of the PWM signal. Will be limited to min_pw and max_pw.

set_speed(*speed*)
Sets speed of the servo.

This method sets the servos rotation speed. The speed range is defined by by min_speed and max_speed .

Parameters **speed** (*int*, *float*) – Should be between min_speed and max_speed , otherwise the value will be limited to those values.

stop()
Sets the speed of the servo to 0.

4.3 lib_motion

Module for moving the robot.

This module includes the method *lib_motion.control.move()* which is the core of the movement controlling. The module imports *lib_para_360_servo* .

class lib_motion.**control**(*pi*, *width_robot=102*, *diameter_wheels=66*, *unitsFC=360*, *dcMin_l=27.3*, *dcMax_l=969.15*, *dcMin_r=27.3*, *dcMax_r=978.25*, *l_wheel_gpio=16*, *r_wheel_gpio=20*, *servo_l_gpio=17*, *min_pw_l=1280*, *max_pw_l=1720*, *min_speed_l=-1*, *max_speed_l=1*, *servo_r_gpio=27*, *min_pw_r=1280*, *max_pw_r=1720*, *min_speed_r=-1*, *max_speed_r=1*, *sampling_time=0.01*, *Kp_p=0.1*, *Ki_p=0.1*, *Kd_p=0*, *Kp_s=0.5*, *Ki_s=0*, *Kd_s=0*)
Controls the robot movement.

This class controls the robots movement by controlling the two Parallax Feedback 360° High-Speed Servos 360_data_sheet[6] . The robots local coordinate system is defined in the following way. X-axis positive is straight forward, y-axis positive is perpendicular to the x-axis in the direction to the left wheel. The center (0/0) is where the middle of the robots chassi is cutting across a imaginary line through both wheels/servos. Angle phi is the displacement of the local coordinate system to the real world coordinate system. See *Used local coordinate system* for a picture of it.

Warning: Be carefull with setting the min and max pulsewidth! Test carefully min_pw and max_pw before setting them. Wrong values can damage the servo, see set_servo_pulsewidth[5] !!!

Parameters

- **pi** (*pigpio.pi*) – Instance of a pigpio.pi() object.
- **width_robot** (*int*) – Width of the robot in mm, so distance between middle right wheel and middle left wheel. **Default:** 102 mm, measured.
- **diameter_wheels** – Diameter of both wheels. **Default:** 66 mm, measured and taken from the products website wheel_robot[7] .
- **unitsFC** (*int*) – Units in a full circle, so each wheel is divided into X sections/ticks. This value should not be changed. **Default:** 360
- **dcMin_l** (*float*) – Min duty cycle of the left wheel. **Default:** 27.3, measured with method *lib_para_360_servo.calibrate_pwm()* , see *Examples* .
- **dcMax_l** (*float*) – Max duty cycle of the left wheel. **Default:** 969.15, measured with method *lib_para_360_servo.calibrate_pwm()* , see *Examples* .
- **dcMin_r** (*float*) – Min duty cycle of the right wheel. **Default:** 27.3, measured with method *lib_para_360_servo.calibrate_pwm()* , see *Examples* .
- **dcMax_r** (*float*) – Max duty cycle of the left wheel. **Default:** 978.25, measured with method *lib_para_360_servo.calibrate_pwm()* , see *Examples* .
- **l_wheel_gpio** (*int*) – GPIO identified by their Broadcom number, see elinux.org[1] . To this GPIO the feedback wire of the left servo has to be connected.
- **r_wheel_gpio** (*int*) – GPIO identified by their Broadcom number, see elinux.org[1] . To this GPIO the feedback wire of the right servo has to be connected.
- **servo_l_gpio** (*int*) – GPIO identified by their Broadcom number, see elinux.org[1] . To this GPIO the control wire of the left servo has to be connected.
- **min_pw_l** (*int*) – Min pulsewidth, see **Warning**, carefully test the value before! **Default:** 1280, taken from the data sheet 360_data_sheet[6] .
- **max_pw_l** (*int*) – Max pulsewidth, see **Warning**, carefully test the value before! **Default:** 1720, taken from the data sheet 360_data_sheet[6] .
- **min_speed_l** (*int*) – Min speed which the servo is able to move. **Default:** -1, so that the speed range is also scaled between -1 and 1 as the output of the inner control loop.
- **max_speed_l** (*int*) – Max speed which the servo is able to move. **Default:** 1, so that the speed range is also scaled between -1 and 1 as the output of the inner control loop.
- **servo_r_gpio** (*int*) – GPIO identified by their Broadcom number, see elinux.org[1] . To this GPIO the control wire of the right servo has to be connected.
- **min_pw_r** (*int*) – Min pulsewidth, see **Warning**, carefully test the value before! **Default:** 1280, taken from the data sheet 360_data_sheet[6] .
- **max_pw_r** (*int*) – Max pulsewidth, see **Warning**, carefully test the value before! **Default:** 1720, taken from the data sheet 360_data_sheet[6] .
- **min_speed_r** (*int*) – Min speed which the servo is able to move. **Default:** -1, so that the speed range is also scaled between -1 and 1 as the output of the inner control loop.
- **max_speed_r** (*int*) – Max speed which the servo is able to move. **Default:** 1, so that the speed range is also scaled between -1 and 1 as the output of the inner control loop.

[7] https://www.parallax.com/product/28114

- **sampling_time** (*float*) – Sampling time of the four PID controllers in seconds. **Default:** 0.01. 1. PWM of motor feedback is 910Hz (0,001098901 s), so position changes cannot be recognized faster than 1.1 ms. Therefore, it is not needed to run the outer control loop more often and update the speed values which have a 50 Hz (20ms) PWM. 2. Tests of the runtime of the code including the controller part have shown that writing the pulsewidth (pi.set_servo_pulsewidth()) in *lib_para_360_servo.write_pwm.set_pw()* is the bottleneck which drastically slows down the code by the factor ~400 (0,002 seconds vs 0,000005 seconds; runtime with vs without writing pulsewidth). 3. For recognizing the RPMs of the wheels 10ms is needed to have enough changes in the position. This was found out by testing. See method *move()* for more informations.
- **Kp_p** (*int, float*) – Kp value of the outer PID controllers, see method *move()* for more informations. **Default:** 0.1.
- **Ki_p** (*int, float*) – Ki value of the outer PID controllers, see method *move()* for more informations. **Default:** 0.1.
- **Kd_p** (*int, float*) – Kd value of the outer PID controllers, see method *move()* for more informations. **Default:** 0.
- **Kp_s** (*int, float*) – Kp value of the inner PID controllers, see method *move()* for more informations. **Default:** 0.5.
- **Ki_s** (*int, float*) – Ki value of the inner PID controllers, see method *move()* for more informations. **Default:** 0.
- **Kd_s** (*int, float*) – Kd value of the inner PID controllers, see method *move()* for more informations. **Default:** 0.

cancel()

Cancel the started callback function.

This method cancels the started callback function if initializing an object. As written in the pigpio callback[3] documentation, the callback function may be cancelled by calling the cancel function after the created instance is not needed anymore.

move(*number_ticks=0*, *straight=False*, *turn=False*)

Core of motion control.

This method controls the movement of the robot. It is called from *lib_motion.control.turn()* or *lib_motion.control.straight()* and is not ment to be called directly. Four digital PID controllers are used to make two cascade control loops, one cascade control loop for each wheel. Each cascade control loop has the same parameters (P/I/D parameters), so that both wheels are controlled in the same way. Chosen default: Outer control loop is a PI controller, inner control loop is a P controller. The outer loop is a position controller, the inner loop a speed controller. After both wheels have reached their set-point (position), the method waits one second before the movement is marked as finished. This ensures that overshoots/oscillations are possible and that both wheels can independently reach their set-point (position). The I part of each PID controller is limited to -1 and 1, so that the sum of the errors is not integrated till infinity which means to very high or low values which might cause problems. The output value of each inner PID controller is scaled between -1 and 1 and the output value of each outer PID controller is limited to -1 and 1. This ensures that no scaling factors are introduced in the P/I/D parameters and also that the output of each PID controller matches the speed range of the servos, defined in *lib_para_360_servo.write_pwm.set_speed()* . A sliding median window is used to filter out the noise in the rotation speed measurement (ticks/s) which is done indirectly by measuring the position of the servo. Also a dead-band filter after the error calculation of the outer control loop is implemented. This adjustments help to make the controllers more stable, e.g. filter out outliers while calculating the rotation speed and therefore avoid high value changes/jumps or avoid oscillations after reaching the set-point (position). The sample

time of the digital PID controllers can also be freely chosen and does not influence the P/I/D parameters, the rotation speed measurement or the time before the movement is marked as finished.

Parameters

- **number_ticks** (*int, float*) – Number of ticks the wheels have to move.
- **straight** (*bool*) – True or False, if robot should move straight. **Default:** False.
- **turn** (*bool*) – True or False, if robot should turn. **Default:** False.

straight(*distance_in_mm*)

Moves the robot about x mm forward or backward.

This method moves the robot x mm forward or backward. Positive distance values move the robot forward (regarding the local x-axis), negative distance values backward (regarding the local x-axis), see picture in *Used local coordinate system* , where the local coordinate system of the robot is shown. This method calls *lib_motion.control.move()* which controls the movement of the robot.

Parameters **distance_in_mm** (*int, float*) – Distance the robot has to move.

turn(*degree*)

Turns the robot about x degree.

This method turns the robot x degree to the left or to the right. Positive degree values turn the robot to the left, negative degree values to the right, see picture in *Used local coordinate system* , where the local coordinate system of the robot is shown. This method calls *lib_motion.control.move()* which controls the movement of the robot.

Parameters **degree** (*int, float*) – Degree the robot has to turn.

4.4 References

CHAPTER 5

Indices and tables

- genindex
- modindex
- search

Python Module Index

l

lib_motion, 39
lib_para_360_servo, 37
lib_scanner, 33

Index

C

calibrate_pwm (*class in lib_para_360_servo*), 37
cancel() (*lib_motion.control method*), 41
cancel() (*lib_para_360_servo.read_pwm method*), 38
cancel() (*lib_scanner.hcsr04 method*), 34
cancel() (*lib_scanner.scanner method*), 36
control (*class in lib_motion*), 39

H

hcsr04 (*class in lib_scanner*), 33

L

lib_motion (*module*), 39
lib_para_360_servo (*module*), 37
lib_scanner (*module*), 33

M

max_backward() (*lib_para_360_servo.write_pwm method*), 38
max_forward() (*lib_para_360_servo.write_pwm method*), 38
max_left() (*lib_scanner.para_standard_servo method*), 35
max_right() (*lib_scanner.para_standard_servo method*), 35
middle_position() (*lib_scanner.para_standard_servo method*), 35
move() (*lib_motion.control method*), 41

P

para_standard_servo (*class in lib_scanner*), 34

R

read() (*lib_para_360_servo.read_pwm method*), 38
read() (*lib_scanner.hcsr04 method*), 34
read_all_angles() (*lib_scanner.scanner method*), 36
read_pwm (*class in lib_para_360_servo*), 37

S

scanner (*class in lib_scanner*), 35
set_position() (*lib_scanner.para_standard_servo method*), 35
set_pw() (*lib_para_360_servo.write_pwm method*), 39
set_pw() (*lib_scanner.para_standard_servo method*), 35
set_speed() (*lib_para_360_servo.write_pwm method*), 39
stop() (*lib_para_360_servo.write_pwm method*), 39
straight() (*lib_motion.control method*), 42

T

turn() (*lib_motion.control method*), 42

W

write_pwm (*class in lib_para_360_servo*), 38