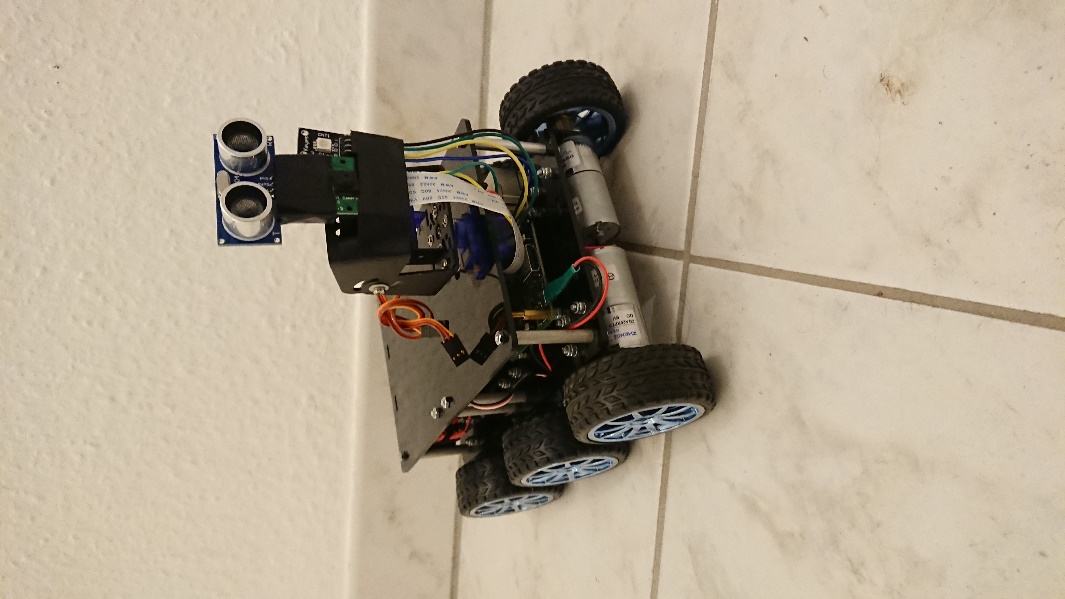
**ASMO**



Documentation

2018-05-08

SRH University Heidelberg  
School of Information, Media and Design  
Degree course "Applied Computer Science"  
Major field "Business Computing"

Reviewers Prof. Dr. Mohammed Yass

**Table of Contents**

[1 Change Log 4](#_Toc509817890)

[2 Introduction 5](#_Toc509817891)

[2.1 Features 5](#_Toc509817892)

[3 Hardware 6](#_Toc509817893)

[3.1 Raspberry Pi 6](#_Toc509817894)

[3.2 Sensors 6](#_Toc509817895)

[3.3 Actuators 7](#_Toc509817896)

[3.4 Wiring 7](#_Toc509817897)

[4 Software 9](#_Toc509817898)

[4.1 Raspbian 9](#_Toc509817899)

[4.2 ASMO Specific Software 10](#_Toc509817900)

[4.2.1 Controller 10](#_Toc509817901)

[4.2.2 Web interface 11](#_Toc509817902)

[5 Usage 12](#_Toc509817903)

[5.1 Installation Guide 12](#_Toc509817904)

[5.2 User guide 13](#_Toc509817905)

[5.2.1 Web interface 13](#_Toc509817906)

[5.2.2 HTTP API 13](#_Toc509817907)

[6 Outlook 18](#_Toc509817908)

[6.1 Driven distance 18](#_Toc509817909)

[6.1.1 Stepper Motors 18](#_Toc509817910)

[6.1.2 Accelerometer 18](#_Toc509817911)

[6.1.3 Sensor on the wheels 18](#_Toc509817912)

[6.2 Indoor positioning 19](#_Toc509817913)

[6.2.1 Wifi based positioning 19](#_Toc509817914)

[6.2.2 Light detection and ranging (LIDAR) 19](#_Toc509817915)

[6.3 Automatic path finding 19](#_Toc509817916)

[6.4 Voice control 19](#_Toc509817917)

[6.5 Camera stream performance 20](#_Toc509817918)

[Index of Figures 21](#_Toc509817919)

[Index of Tables 22](#_Toc509817920)

[Hardware Datasheets 23](#_Toc509817921)

[BattBorg - Pi Battery Power Board PCB Only 23](#_Toc509817922)

[PicoBorg Reverse - Dual 5A Motor Controller 23](#_Toc509817923)

[KY-015 TEMPERATURE AND HUMIDITY SENSOR MODULE 23](#_Toc509817924)

[KY-009 RGB FULL COLOR LED SMD MODULE 23](#_Toc509817925)

[HC-SR04 Ultrasonic range sensor 23](#_Toc509817926)

[7 Bibliography 24](#_Toc509817927)

# Change Log

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Changes | Date | Authors |
| 1.0 | Code reimplementation  Document creation | 26. March 2018 | Martin Vogel |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# Introduction

ASMO is a robotic platform based on the raspberry pi and the diddyborg chassis. It was initially developed with the purpose of creating a robot for caring of elderly people. At this moment in time, ASMO is purely based on python software. This documentation provides and overview over both the software and the hardware of ASMO and provides information about how to implement and modify the existing platform.

## Features

ASMO currently has the following features:

* Driving
* Web interface
* HTTP API
* Camera Stream

# Hardware

## Raspberry Pi

In order to control ASMO, a Raspberry Pi is being used. Raspberry Pi is an open source computer which is commonly used in IoT Applications. A Raspberry Pi can run a number of operating systems, of which Raspbian is the one officially supported by the Raspberry Pi Foundation.

## Sensors

In order to get a starting set of sensors which allow flexibility in the features ASMO has, a kit of 37 Sensors has been ordered from Sunfounder. Documentation of this kit can be found in the file “Sensor kit V1.0 for Rpi B+.pdf”.

Out of these Sensors, the following are being used:

|  |  |  |
| --- | --- | --- |
| Sensor | Purpose | Connection |
| KY-015 Temperature and Humidity | Getting environmental data (temperature and humidity) | VCC: 3.3V SIGNAL: GPIO4 |
| HC-SR04 Ultrasonic range Sensor | Getting the distance that is free in front of ASMO. Used for obstacle detection | VCC: 5V TRIG: GPIO21 ECHO: GPIO20 |
| Raspberry pi camera | Enable ASMO to see | Camera slot |

Table 1 - Sensors

## Actuators

|  |  |  |
| --- | --- | --- |
| Actuator | Purpose | Connection |
| PiBorg Reverse Dual 5A Motor Controller | Controlling the motors in order to enable ASMO to move around. | I2C |
| KY-009 RGB Full Color LED SMD Module | Indicating the status of ASMO | R: GPIO19 B: GPIO13 G: GPIO26 |

Table 2 - Actuators

## Wiring

The wiring diagram was created using fritzing, which is an open source software for hardware documentation (*Fritzing)*. It can be accessed and editied using the file “ASMO\_wiring.fzz”.

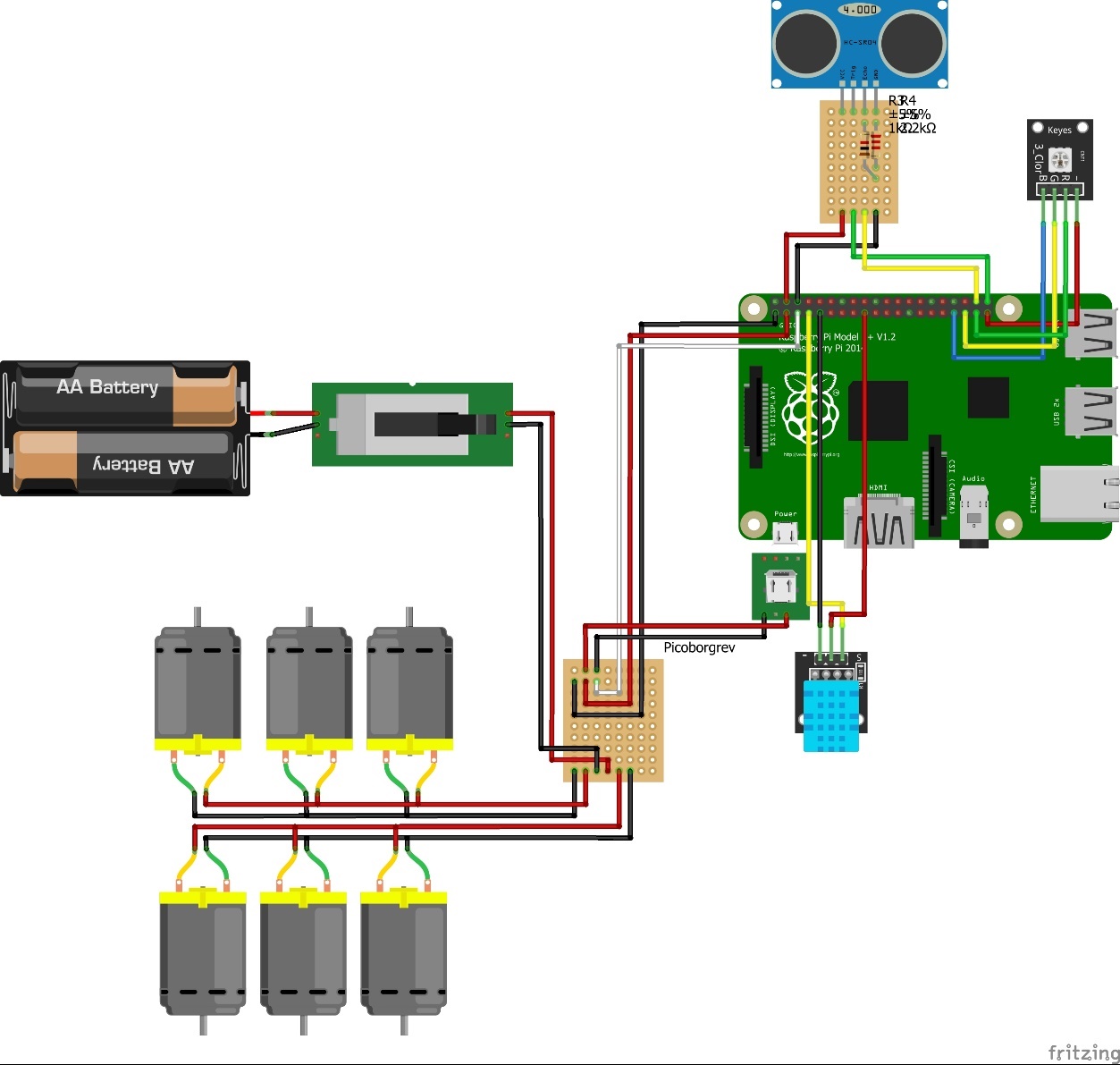


Figure 1 - Wiring diagram

# Software

## Raspbian

The Software developed for ASMO is running on Raspbian. Raspbian is a Linux distribution based on Debian. It is an operating system for the raspberry pi and is officially supported by the raspberry pi foundation. (*Download Raspbian for Raspberry Pi)*.

In addition to Raspbian, the following apt packages are being used:

* build-essential & python-dev  
  These two packages are necessary in order to build and install the Adafruit\_DHT package.
* dnsmasq & hostapd  
  These two packages are needed in order to activate the Wifi Access point. (*Setting up a Raspberry Pi as an access point in a standalone network (NAT) - Raspberry Pi Documentation)*

Raspbian comes with a configuration tool called raspi-config. The ASMO installation script calls this tool using different parameters in order to make the following changes to the raspberry pi specific configuration (*raspi-config command line parameter - Raspberry Pi Forums)*:

* enable camera
* enable i2c
* disable booting into the graphical environment

Raspbian also comes with a predefined python environment which already includes a lot of packages. In addition, the following pip packages are being used:

* web.py  
  This package provides web server functionality (*Welcome to web.py! (web.py),* 28. Februar 2018)
* picamera  
  This package can be used in order to access the raspberry pi camera from python scripts. (*Python picamera - Raspberry Pi Documentation)*

Raspbian uses systemd in order to control, which software is being executed during boot. The ASMO Software is registered as a service for systemd following the guideline from the raspberry pi foundation (*systemd - Raspberry Pi Documentation)*

## ASMO Specific Software

The Software used for controlling ASMO is written entirely in python, which comes pre-installed with Raspbian.

The python script which starts the software is AsmoApi.py which is used in order to start the webserver. In addition, Asmo.conf stores configuration values needed for Asmo. The rest of the software is divided in 4 folders:

* controller  
  This folder contains python files for the interaction with the sensors and actors connected to the raspberry pi.
* routes  
  This folder contains the python files used in order to handle HTTP requests
* static  
  This folder contains a set of static files. Web.py answers requests to the /static/ route by just returning the specified file (*Serving Static Files (such as js, css and images) (web.py),* 28. Februar 2018). Therefore, this folder contains a copy of bootstrap and jQuery.
* views  
  This folder contains the html files used for the web interface.

### Controller

#### Camera

The camera controller makes use of the picamera package. This is based on a tutorial by Miguel Grinberg: (Grinberg) & (Grinberg). The camera will continue to be accessed until no viewer was available for 10 seconds.

#### Distance

The distance controller implements a similar class as the camera controller. When being accesses, it continues to read the distance for 10 seconds until after the last access was done. In case ASMO is moving, it continues to read the distance until the motors stop. In case it recognizes an object closer than a configurable maximum distance, it stops the motors.

#### LED

The LED controller is rather simple. It provides a method for turning the LED off and a method for toggle the colors red, green or blue.

#### Motor

The Motor controller is based on the PicoBorgRev module provided by PiBorg (*PicoBorg Reverse - Examples)*. When the motors are being activated, it also kicks off the distance reading. This is done in order to stop ASMO again when objects are too close.

#### Temperature

The temperature controller makes use of the Adafruit\_DHT library (*adafruit/Adafruit\_Python\_DHT)*. It provides simple functions for reading the temperature, the humidity or both from a DHT11 Sensor.

### Web interface

The web interface for ASMO is developed based on web.py, which is a lightweight web framework for python (*Welcome to web.py! (web.py),* 28. Februar 2018). Within the AsmoApi.py script, the possible routes are defined and the web.py server gets started. All routes are stored in the routes folder. If a not-specified route gets called, this is getting handled by the route defined in index.py. This route will try to deliver the correspondent file stored in the views folder.

# Usage

## Installation Guide

The easiest way to install the ASMO Software on a new SD-Card it to just burn the image to the SD Card. In order to do this, the tool “etcher” can be used. This is also the recommended tool of the raspberry pi foundation when it comes to installing a clean Raspbian system. (*Installing operating system images - Raspberry Pi Documentation)*.

In case a complete re-installation is preferred, an installation script has been written and the ASMO Software can be installed as follows:

1. Download a recent Version of Raspbian
2. Burn Raspbian to a new SD-Card
3. Enable SSH (*SSH (Secure Shell) - Raspberry Pi Documentation)*
4. Download and Copy the ASMO files onto the SD-Card:  
   <https://github.com/MartinVogelSRH/AsmoApi>
5. Make Install.sh executable: chmod +x Install.sh
6. Execute Install.sh

This will make the following changes to the raspberry pi:

1. Install the python packages web.py and picamera
2. Use the raspi-config tool in order to enable i2c and the camera
3. Use the raspi-config tool in order to disable loading the graphical environment during boot
4. Install the build-essential and python-dev packages
5. Download and install the Adafruit\_DHT package
6. Register and enable the ASMO Software as a service for systemctl
7. Install the build- dnsmasq and hostapd packages
8. Configure dhcpcd, dnsmasq and hostapd to automatically open a Wifi Access Point at boot.

## User guide

There are several possibilities to interact with ASMO. All of them require to be in the same network as ASMO is. The easiest way to achieve this is to log in to the Wifi Network ASMO opens:

SSID: ASMO

Password: 12345678

Within this network, ASMO has the IP-Address 10.0.0.1 and provides the following services:

Port 22: SSH. Username: pi Password: raspberry

Port 8080: Web application to control ASMO, including a HTTP-API

### Web interface

As described above, ASMO provides a Web-interface that can be accessed by just navigating to <http://10.0.0.1>:8080 in any web browser.

In here, multiple tabs exist:

1. Home  
   This is the default page which gets opened when the Web interface is being called without any specific page. It provides a remote control for ASMOs motors.
2. Camera   
   This page displays the camera stream
3. Complete Control  
   This page provides a remote control, displays the camera stream and the output of the distance sensor
4. Shutdown the system  
   Clicking this tab will shutdown the Raspberry Pi. It then has to be rebooted manually.
5. Documentation  
   This tab will navigate to the directory listing of /static/documentation in order to allow the user to download the documentation of Asmo.

### HTTP API

ASMO is being controlled by the usage of an API that can be accessed via HTTP. The following routes have been defined:

#### /api/motor

This route provides the functionality to control ASMOs motors. It requires data to be sent as a POST request with form-data. Within this, it expects the values motor1speed and motor2speed. In Postman, calling this route looks like this:

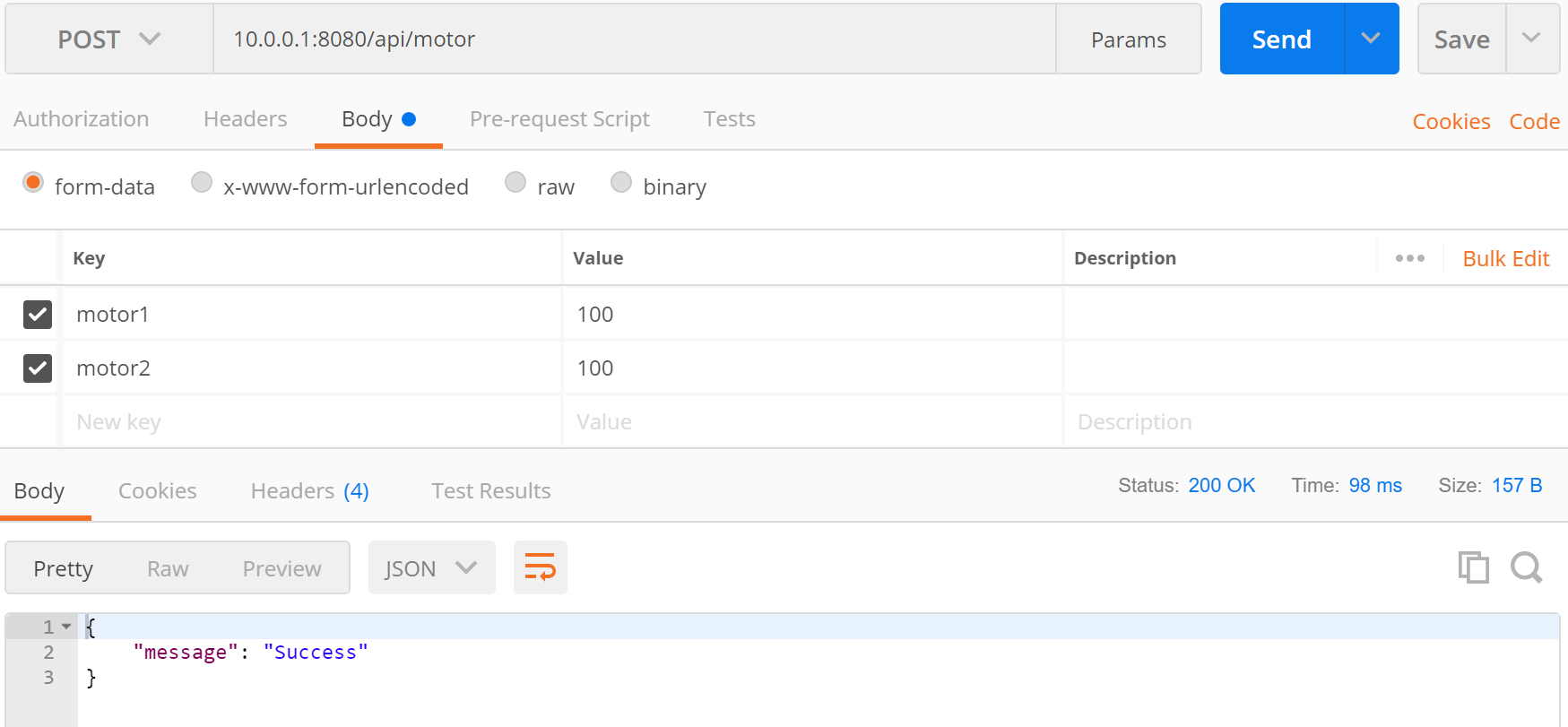


Figure 2 - Postman: /api/motor

#### /api/distance

This route provides access to the distance being read by the ultrasonic sensor in front of ASMO. It has to be used as a GET request and will return a JSON Document containing the distance value. In Postman, calling this route looks like this:

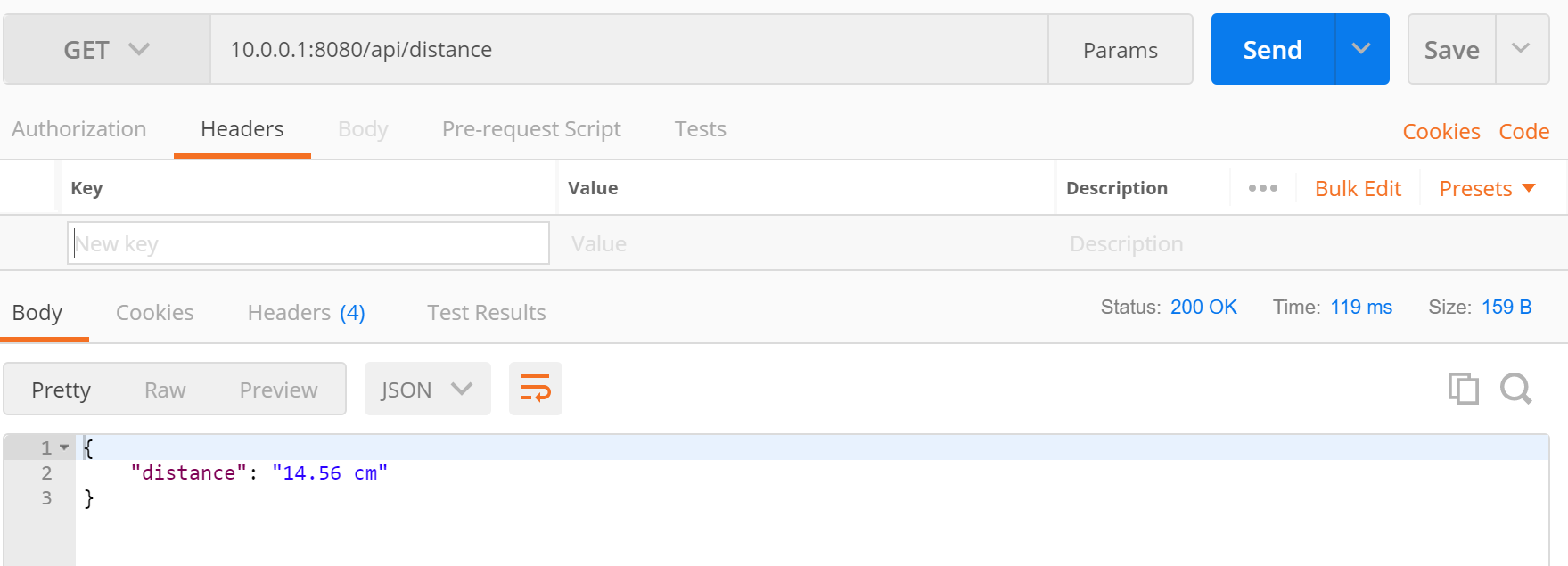


Figure 3 - Postman: /api/distance

#### /api/camera/SinglePicture

This route provides access to the camera of ASMO. It returns an image object containing the current picture from the camera. In Postman, calling this route looks like this:

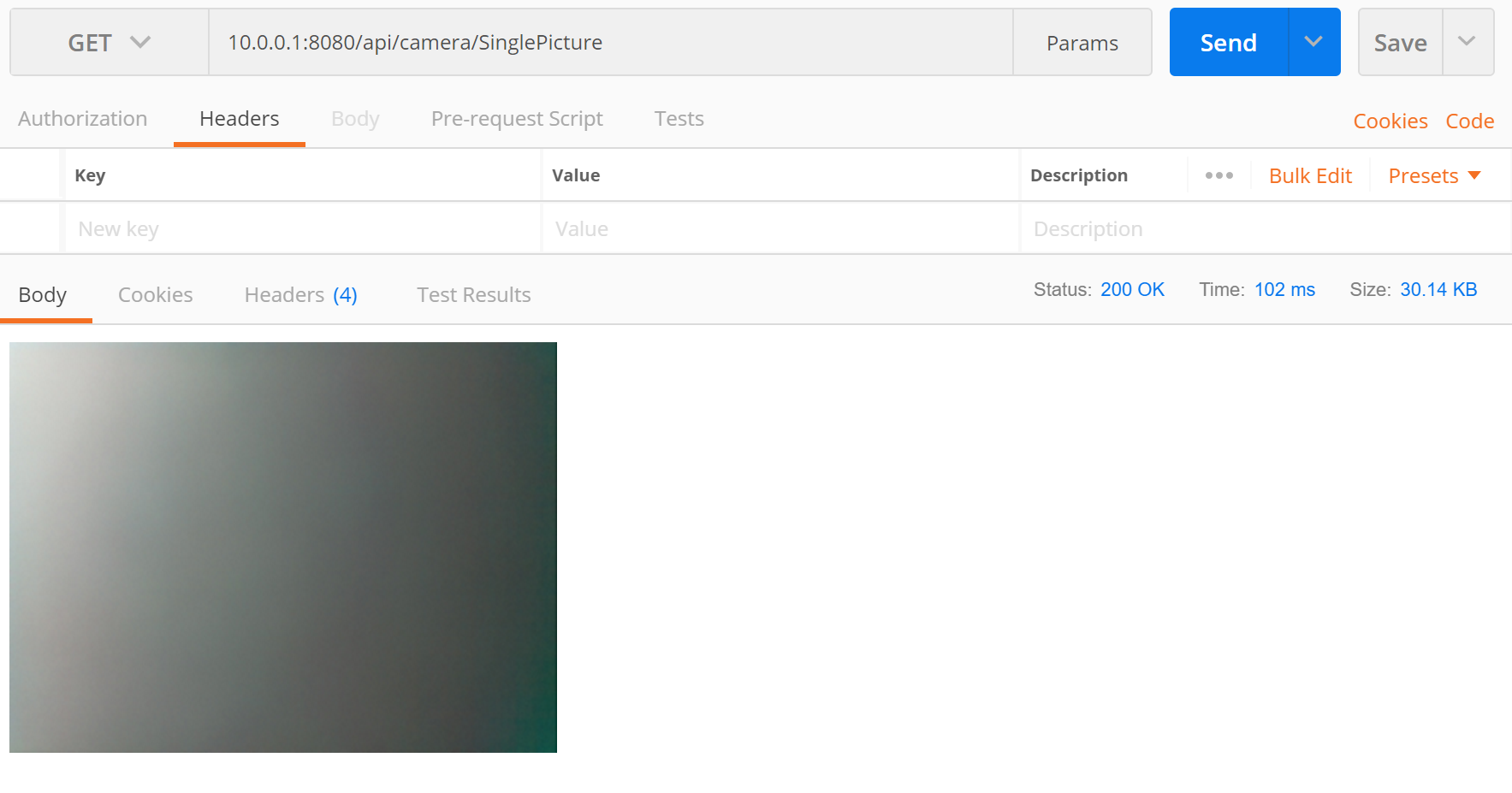


Figure 4 - Postman: /api/camera/SinglePicture

#### /api/camera/Stream

This route provides access to the camera of ASMO. It returns a stream of image objects.

#### /api/led

This route provides access to the LED of ASMO. It can be called both with a GET and a POST request. Calling it with a GET request will turn the LED off and return a JSON document which states that is was successful. With a POST request, the colors of the LED can be toogled. Here, red, green and blue are available. The color provided will be turned on or off dependent on its current status. If multiple colors are turned on, the LED will show the mix of these colors. In Postman, calling this route looks like this:

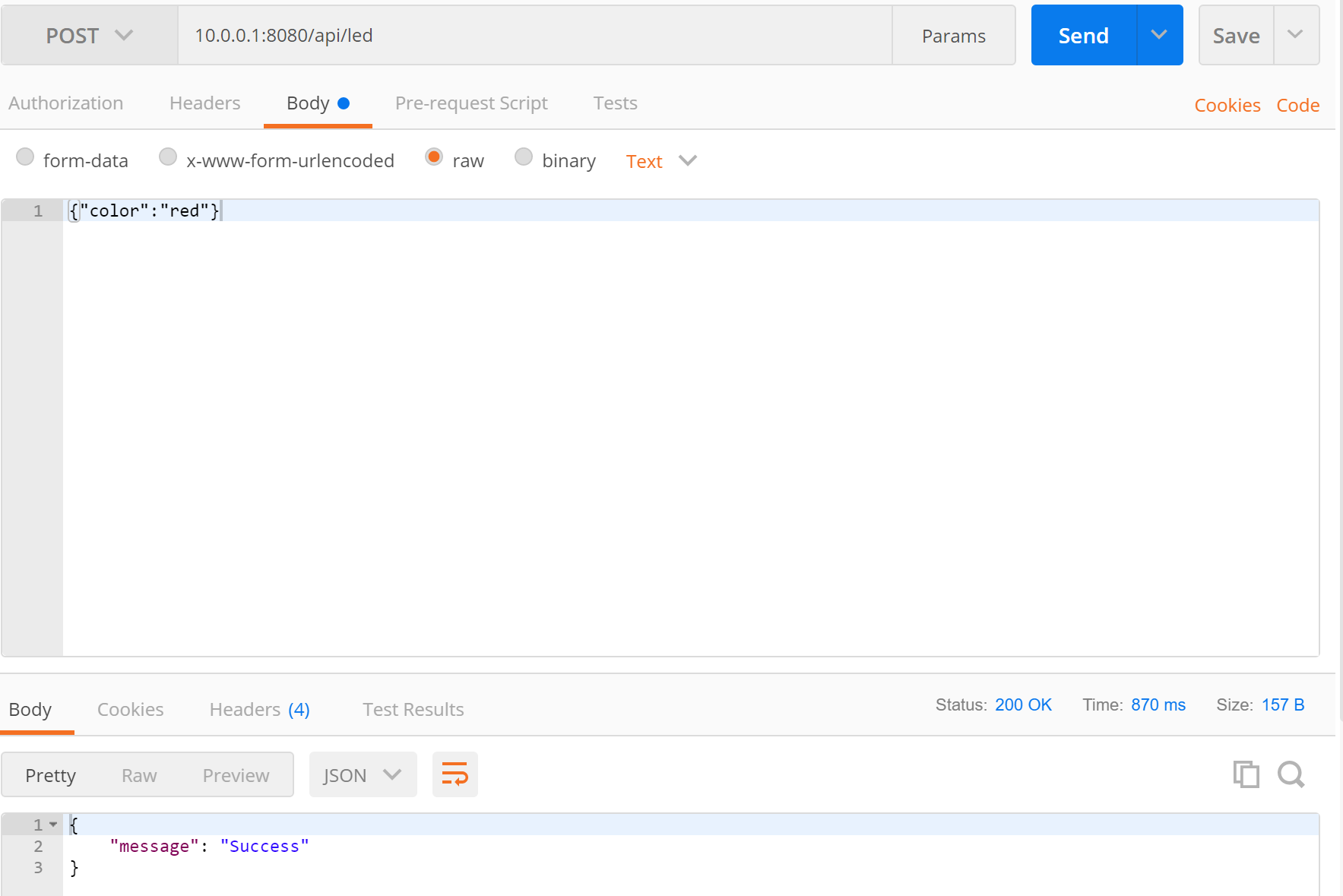


Figure 5 - Postman: /api/led - POST

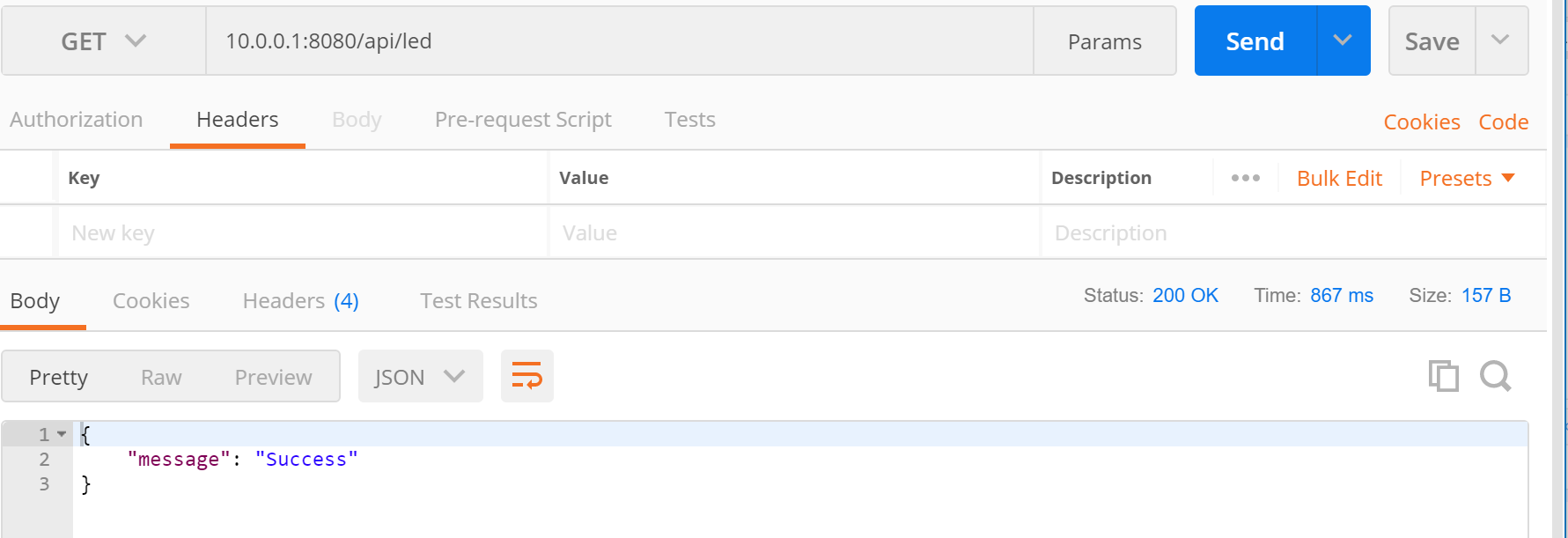


Figure 6 - Postman: /api/led - GET

#### /api/temperature

This route provides access to the temperature sensor of ASMO. It has to be called with a GET request and will return a JSON document containing the current temperature and humidity.

In Postman, calling this route looks like this:

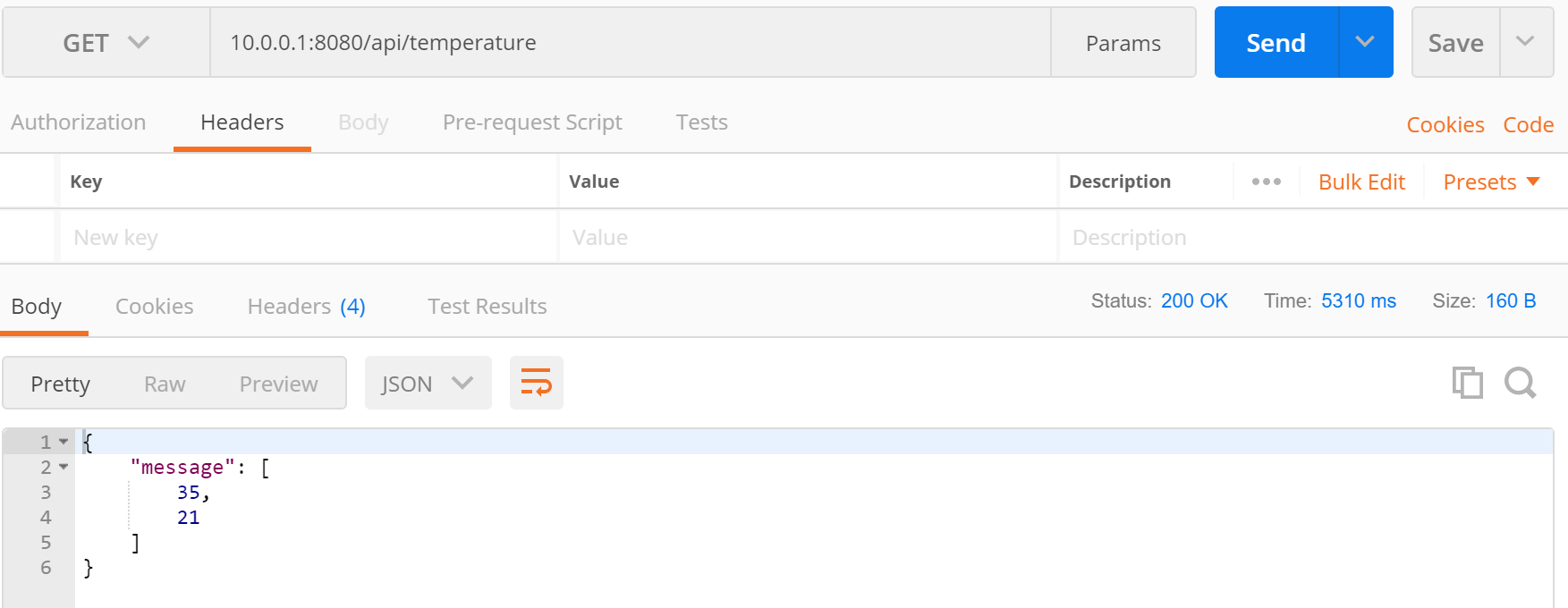


Figure 7 - Postman: /api/temperature

#### /api/settings

This route provides the functionality to get and set the different settings stored in Asmo.conf. It can be called both with a GET and a POST request. Calling it with a GET request will return all settings. In addition, a specific section can be requested within the URL of the request (e.g. calling /api/settings/Server). Calling it with a POST request requires the body to contain a JSON document with all settings that shall be set. In Postman, calling this route looks like this:

#### /api/system/

This route provides the functionality to control the system. It has to get called with a GET request. The specific functionality will be determined with the further URL. Currently, the following functionality is available:

* /api/system/shutdown
* /api/system/reboot
* /api/system/restartServer

Since this will trigger the system or the server to shutdown immediately, no result will be returned:

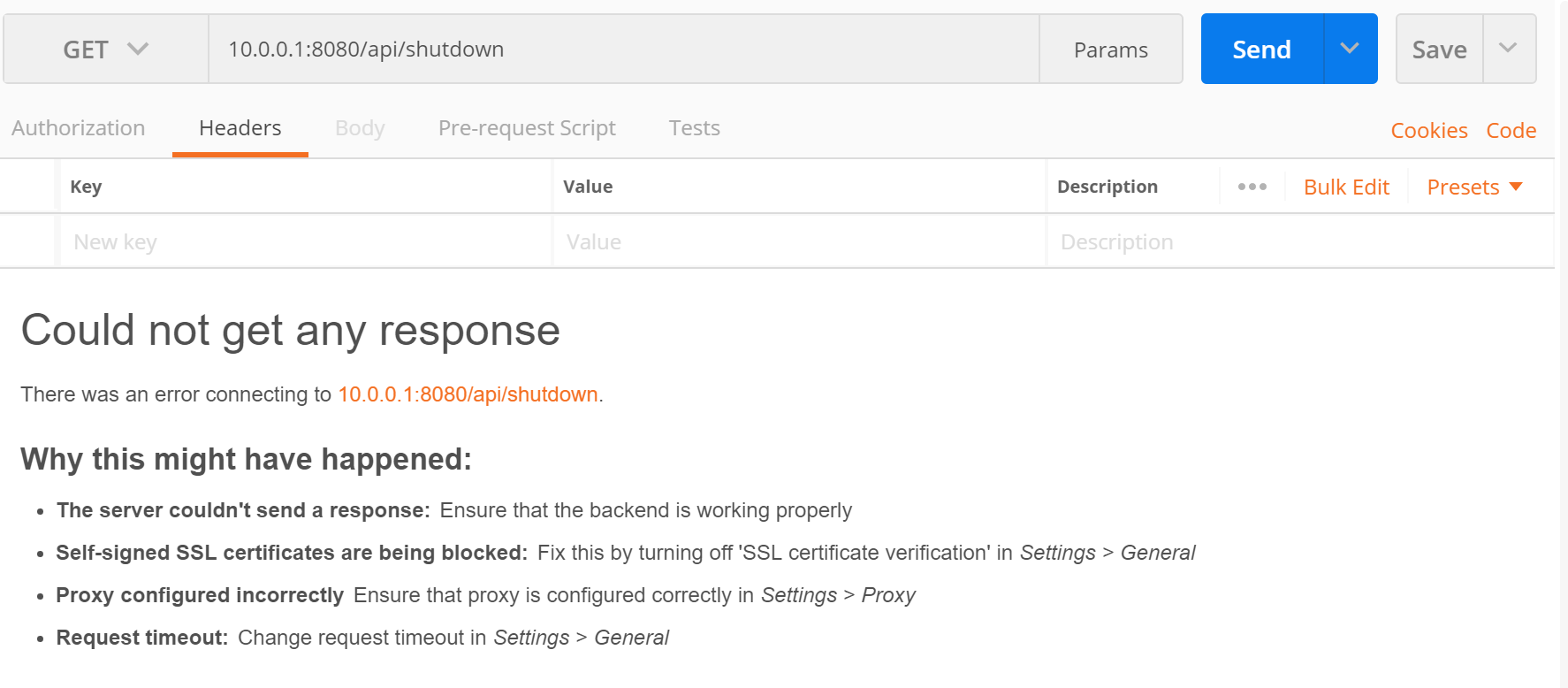


Figure 8 - Postman: /api/system/shutdown

# Outlook

This implementation of ASMO provides only the basic features needed by a robot platform. Therefore, a lot more features can get implemented. This section will try to outline some features and possible ways of implementing them.

## Driven distance

Currently, ASMO is using DC motors in order to drive. While driving is absolutely possible, DC motors have the downside that there is no feedback given about the distance driven. In addition, the speed of the motors depends on the power delivered by the power supply. Therefore, if the batteries are running low, ASMO gets slower.

As of this, there is currently no possibility to determine the distance driven. This could be implemented in multiple ways:

### Stepper Motors

Opposite to DC motors, stepper motors have the advantage that they have defined positions at which they will stop. Therefore, with a stepper motor, the application would always know, how much distance was driven.

According to the documentation of the driver shield used in ASMO (section “stepper motor”), the connection of a stepper motor should be possible. However, since a stepper motor would use both motor connections, an additional one of the picoborgrev shields might be needed (*PicoBorg Reverse - Getting Started)*

### Accelerometer

Data from an accelerometer can be used in order to measure the distance an object has been moved. However, this is a not very precise measurement methodology (Semiconductor und Inc)

### Sensor on the wheels

The motor shield used in ASMO supports the usage of feedback signals when being used with DC motors. This is described in the section “DC (normal) motors with feedback” of the documentation (*PicoBorg Reverse - Getting Started)*. For this, a sensor could be used which sends a “1” at every turn of the wheel. This could possibly be achieved using a magnetic sensor.

## Indoor positioning

Right now, ASMO does not know its position. For indoor positioning, multiple approaches exist with different implementation effort and precision:

### Wifi based positioning

Wifi based positioning makes use of the already existing wireless signals such as wifi signals. This approach could be used by implementing the FIND Software (*schollz/find)*. It has the advantage that no new sensor would be needed.

### Light detection and ranging (LIDAR)

LIDAR sensors are sensors which are being used a lot within the field of autonomous driving. They basically send out laser signals and measure the time the light needs to get back to the sensor. With this, a precise map of the environment can be created. There are existing implementations of LIDAR sensors with the raspberry pi (*How to Use Lidar with the Raspberry Pi)*. Unfortunately, LIDAR sensors are very expensive. However, using LIDAR could replace the ultrasonic distance sensor, which is not very precise.

## Automatic path finding

Once ASMO is aware of its surroundings and the driven distance can be properly controlled, an automatic path finding can be implemented. There was already an implementation of the A\* algorithm done on ASMO. This implementation was controlling the distance by controlling the time the motors have been activated. While this provided good results in a testing environment, it is not useable in a real scenario.

## Voice control

In order to interact with ASMO in a user-friendly way, a voice control system could be implemented. In general, Services like Google Assistant or Amazon Alexa can be used on the raspberry pi. However, they require an active internet connection, which is not given with ASMO. Therefore, an offline system needs to be used. Existing systems are available.

(*Meet Jasper: open-source voice computing - Raspberry Pi)*

(*bishoph/sopare, Step by step: Raspberry Pi offline voice recognition with SOPARE | home of bishoph)*

When implementing voice control, an upgrade to the Raspberry Pi 3 should be considered.

## Camera stream performance

Right now, the camera stream has very poor performance. This is most likely caused by the usage of python and picamera in order to get the stream. In the past, the RPi-Cam-Web-Interface was being used, which provided a stream with low latency. This interface is a wrapper around the raspimjpeg software. It could be checked, how this software can be integrated properly into the ASMO Software. Since it works by writing files into the temporary storage on /dev/shm/, this should be possible (*RPi-Cam-Web-Interface - eLinux.org,* 23. März 2018). For power consumption reasons, focus should be put on starting the camera access only when needed and stopping it again when it is not needed anymore.

# Index of Figures

[Figure 1 - Wiring diagram 7](#_Toc509817302)

[Figure 2 - Postman: /api/motor 13](#_Toc509817303)

[Figure 3 - Postman: /api/distance 13](#_Toc509817304)

[Figure 4 - Postman: /api/camera/SinglePicture 14](#_Toc509817305)

[Figure 5 - Postman: /api/led - POST 15](#_Toc509817306)

[Figure 6 - Postman: /api/led - GET 15](#_Toc509817307)

[Figure 7 - Postman: /api/temperature 16](#_Toc509817308)

[Figure 8 - Postman: /api/shutdown 16](#_Toc509817309)

# Index of Tables

[Table 1 - Sensors 5](#_Toc509817293)

[Table 2 - Actuators 6](#_Toc509817294)

Literatur

*adafruit/Adafruit\_Python\_DHT* [online] [Zugriff am: 26. März 2018]. Verfügbar unter: https://​github.com​/​adafruit/​Adafruit\_​Python\_​DHT

*BattBorg - Pi Battery Power Board PCB Only (Soldered)* [online] [Zugriff am: 26. März 2018]. Verfügbar unter: https://​www.piborg.org​/​power-​1137/​battborg

*bishoph/sopare* [online] [Zugriff am: 21. März 2018]. Verfügbar unter: https://​github.com​/​bishoph/​sopare

*Download Raspbian for Raspberry Pi* [online] [Zugriff am: 26. März 2018]. Verfügbar unter: https://​www.raspberrypi.org​/​downloads/​raspbian/​

*Fritzing* [online] [Zugriff am: 26. März 2018]. Verfügbar unter: http://​fritzing.org​/​home/​

Grinberg, M. *Flask Video Streaming Revisited - miguelgrinberg.com* [online] [Zugriff am: 26. März 2018]. Verfügbar unter: https://​blog.miguelgrinberg.com​/​post/​flask-​video-​streaming-​revisited

Grinberg, M. *Video Streaming with Flask - miguelgrinberg.com* [online] [Zugriff am: 17. März 2018]. Verfügbar unter: https://​blog.miguelgrinberg.com​/​post/​video-​streaming-​with-​flask

*How to Use Lidar with the Raspberry Pi* [online] [Zugriff am: 26. März 2018]. Verfügbar unter: https://​hackaday.com​/​2016/​01/​22/​how-​to-​use-​lidar-​with-​the-​raspberry-​pi/​

*Installing operating system images - Raspberry Pi Documentation* [online] [Zugriff am: 28. Februar 2018]. Verfügbar unter: https://​www.raspberrypi.org​/​documentation/​installation/​installing-​images/​README.md

*KY-015 Temperature and Humidity Sensor Module - ArduinoModulesInfo* [online] [Zugriff am: 26. März 2018]. Verfügbar unter: https://​arduinomodules.info​/​ky-​015-​temperature-​humidity-​sensor-​module/​

*Meet Jasper: open-source voice computing - Raspberry Pi* [online] [Zugriff am: 20. März 2018]. Verfügbar unter: https://​www.raspberrypi.org​/​blog/​meet-​jasper-​open-​source-​voice-​computing/​

*ModMyPi | HC-SR04 Ultrasonic Range Sensor on the Raspberry Pi* [online] [Zugriff am: 16. März 2018]. Verfügbar unter: https://​www.modmypi.com​/​blog/​hc-​sr04-​ultrasonic-​range-​sensor-​on-​the-​raspberry-​pi

*PicoBorg Reverse - Dual 5A Motor Controller* [online] [Zugriff am: 26. März 2018]. Verfügbar unter: https://​www.piborg.org​/​motor-​control-​1135/​picoborgrev

*PicoBorg Reverse - Examples* [online] [Zugriff am: 26. März 2018]. Verfügbar unter: https://​www.piborg.org​/​blog/​build/​picoborg-​reverse-​build/​picoborg-​reverse-​examples

*PicoBorg Reverse - Getting Started* [online] [Zugriff am: 26. März 2018]. Verfügbar unter: https://​www.piborg.org​/​blog/​build/​picoborg-​reverse-​build/​picoborg-​reverse-​getting-​started

*Python picamera - Raspberry Pi Documentation* [online] [Zugriff am: 22. März 2018]. Verfügbar unter: https://​www.raspberrypi.org​/​documentation/​usage/​camera/​python/​README.md

*raspi-config command line parameter - Raspberry Pi Forums* [online] [Zugriff am: 15. März 2018]. Verfügbar unter: https://​www.raspberrypi.org​/​forums/​viewtopic.php​?​t=​21632

*RGB LED Module - Wiki* [online], 7 März 2017 [Zugriff am: 26. März 2018]. Verfügbar unter: http://​wiki.sunfounder.cc​/​index.php​?​title=​RGB\_LED\_Module

*RPi-Cam-Web-Interface - eLinux.org* [online], 23 März 2018 [Zugriff am: 26. März 2018]. Verfügbar unter: https://​elinux.org​/​RPi-​Cam-​Web-​Interface

*schollz/find* [online] [Zugriff am: 22. März 2018]. Verfügbar unter: https://​github.com​/​schollz/​find

Semiconductor, F. und Inc. AN3397, Implementing Positioning Algorithms Using Accelerometers - Application Notes [online] [Zugriff am: 26. März 2018]. Verfügbar unter: https://​www.nxp.com​/​docs/​en/​application-​note/​AN3397.pdf

*Serving Static Files (such as js, css and images) (web.py)* [online], 28 Februar 2018 [Zugriff am: 26. März 2018]. Verfügbar unter: http://​webpy.org​/​cookbook/​staticfiles

*Setting up a Raspberry Pi as an access point in a standalone network (NAT) - Raspberry Pi Documentation* [online] [Zugriff am: 15. März 2018]. Verfügbar unter: https://​www.raspberrypi.org​/​documentation/​configuration/​wireless/​access-​point.md

*SSH (Secure Shell) - Raspberry Pi Documentation* [online] [Zugriff am: 28. Februar 2018]. Verfügbar unter: https://​www.raspberrypi.org​/​documentation/​remote-​access/​ssh/​

*Step by step: Raspberry Pi offline voice recognition with SOPARE | home of bishoph* [online] [Zugriff am: 21. März 2018]. Verfügbar unter: https://​www.bishoph.org​/​step-​by-​step-​raspberry-​pi-​offline-​voice-​recognition-​with-​sopare/​

*systemd - Raspberry Pi Documentation* [online] [Zugriff am: 15. März 2018]. Verfügbar unter: https://​www.raspberrypi.org​/​documentation/​linux/​usage/​systemd.md

*Ultrasonic Module - Wiki* [online], 20 März 2017 [Zugriff am: 26. März 2018]. Verfügbar unter: http://​wiki.sunfounder.cc​/​index.php​?​title=​Ultrasonic\_Module

*Welcome to web.py! (web.py)* [online], 28 Februar 2018 [Zugriff am: 26. März 2018]. Verfügbar unter: http://​webpy.org​/​

# Hardware Documentation

## BattBorg - Pi Battery Power Board PCB Only

(*BattBorg - Pi Battery Power Board PCB Only (Soldered))*

## PicoBorg Reverse - Dual 5A Motor Controller

(*PicoBorg Reverse - Dual 5A Motor Controller)*

## KY-015 TEMPERATURE AND HUMIDITY SENSOR MODULE

(*KY-015 Temperature and Humidity Sensor Module - ArduinoModulesInfo)*

## KY-009 RGB FULL COLOR LED SMD MODULE

(*RGB LED Module - Wiki,* 7. März 2017)

## HC-SR04 Ultrasonic range sensor

(*Ultrasonic Module - Wiki,* 20. März 2017)

(*ModMyPi | HC-SR04 Ultrasonic Range Sensor on the Raspberry Pi)*