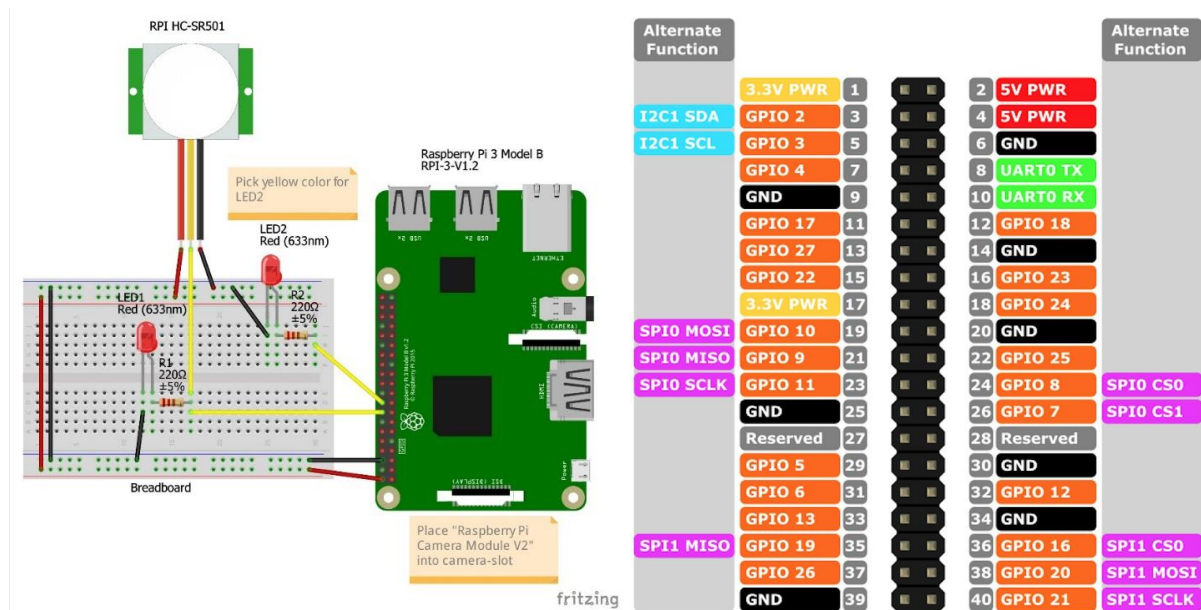# Motion-Camera

## Introduction

This project includes a surveillance camera system with any number of clients that can be switched on and off via a central interface. It should be possible to store recorded images on the network drive and, if necessary, to start a live stream from a targeted camera.

## Hardware-Components

- Raspberry Pi 3 Model B
- RPI HC-SR501 (Motion-Sensor)
- Camera Module V2
- LED (1 x Red & 1 X Yellow)
- Resitors (2 x 220Ohm)
- Cables
- Micro-SD-Card (at least class 10 with UHS-I)

## Hardware-Installation & Pins



In the MitionCamera.pdf is a scetch of the build of the Raspberry-Pi with the Motion-Sensor and the Camera. Do not use other Pins on your own and build this scetch like its shown in the scetch. If you have done the installation correctly and power on the Raspberry-Pi without an image, the red LED should be one, if motion gets tracked. Note that the picture pins must be rotated 180 degrees so that the order of the pins matches the positioning of the Raspberry Pi with the sketch. The two pictures can be viewed enlarged at the following addresses:

- https://www.facing-south.com/img/motionCamera.jpg
- https://roboticsbackend.com/wp-content/uploads/2019/05/raspberry-pi-3-pinout.jpg

# Software

## Operating-System

The operating system that was installed on the Pi is the Raspberry Pi OS (Raspberry Pi OS (32-bit) Lite). The image can be found here: https://www.raspberrypi.org/downloads/raspberry-pi-os/

## Libraries/Updates

Um die in unserem Python-Script verwendeten Bibliotheken zu verwenden, müssen einige zusätzliche Pakete, sowie Updated installiert werden, dafür müssen die folgenden Befehle ausgeführt werden:

- `sudo apt update`
- `sudo apt full-upgrade`
- `sudo apt-get install python-picamera`
- `sudo apt install python3-pip`
- `sudo pip3 install pyTelegramBotAPI`
- `sudo apt-get install rpi.gpio`
- `sudo pip3 install Pillow`
- `sudo apt-get install libopenjp2-7`
- `sudo apt-get install libtiff5`
- `sudo pip3 install systemd`

## Directions

The following folders must be created:

- /home/pi/images
- /home/pi/github

## Download

The entire repository of the MotionCamera project can be loaded directly from Github into the "github" folder. When this step has been completed, the folder structure should look like this:

- /home/pi/github/MotionCamera/MotionCameraClient

## Starting Software

If you are now in the file path specified under Download after downloading the project folder, you can start the bot manually with the following input:

- python3 MotionCameraClient.py

As soon as the software is running, the command / start can be entered in the associated telegram group. Now all captured images are loaded directly into the telegram group.
As soon as the Raspberry-Pi is connected to the power, it also tries to start the monitoring software automatically. As soon as the Raspberry-Pi is connected to the power, it also tries to start the monitoring software automatically. You can tell whether the monitoring software could start itself after a waiting time of approx. 1-2 minutes by the flashing yellow LED, which will flash three times.

# Implementing Auto-Start

## Create Service File

Open a sample unit file using the command as shown below:

```
sudo nano /lib/systemd/system/sample.service
```

Add in the following text:

```
[Unit]

Description = Motion-Camera

After=multi-user.target


[Service]

Type=idle

ExecStart=/usr/bin/python3
/home/pi/github/MotionCamera/MotionCameraClient/MotionCameraClient.py

WorkingDirectory=/home/pi/github/MotionCamera/MotionCameraClient

#StandardOutput=inherit

#StandardError=inherit

Restart=always

User=pi


[Install]

WantedBy=multi-user.target
```

The permission on the unit file needs to be set to 644 :

```
sudo chmod 644 /lib/systemd/system/sample.service
```

## Configure systemd

Now the unit file has been defined we can tell systemd to start it during the boot sequence:

```
sudo systemctl daemon-reload

sudo systemctl enable sample.service
```

Reboot the Pi and your custom service should run:

```
sudo reboot
```

## Usage

The cameras are controlled via the Telegram app. The following commands can be sent to the bot, provided that you are in the same group as the bot:

| Command | Function |
| --- | --- |
| /start | Starts the survaillance |
| /end | Ends the survaillance |

As soon as the motion detector detects movement, the red LED lights up. As soon as a photo has been taken, the yellow LED starts to flash. The motion sensor needs about 6 seconds after each movement to be able to react to the next movement. During this time the yellow LED flashes slowly. As soon as the motion sensor is ready to record the next motion, the yellow LED flashes three times in quick succession.

## Telegram

### Create Bot

To create a Telegram bot I recommend following these instructions: https://core.telegram.org/bots. If you follow the individual steps of these instructions, you will receive a unique token that belongs to the bot created and must be kept secret. Otherwise it could happen that the bot can be abused. This received token must be used in line 15 of the MotionCameraClient.py script (replace the "XXXXX" with your token):

- 15: bot = telebot.TeleBot("XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX")

After that you can Invite the bot into your telegram group.

### Get Group-ID

So that the bot places the recorded surveillance images in the group in which you have invited the bot, the ID of this group has to be determined. To get this ID, I recommend following those instructions: https://stackoverflow.com/questions/32423837/telegram-bot-how-to-get-a-group-chat-id. Once the ID has been determined, it must also be entered in the MotionCameraClient.py script. This happens in line 28 and looks like this:

- 28: bot.send_photo(chat_id=XXXXXX, photo=open("/home/pi/images/image.jpg", "rb"))

At this point, too, the "XXXXX" must be replaced by the determined group ID.

## Finish

Now that all points have been followed, your own surveillance system should work and the captured images should be sent to the Telegram group. Remember that the recordings may only be taken within the framework of the law. Have fun using it.