BSc Computational and Data Science
CDS-205 Computer Science
Dozentin: Dr. sc. Ana Petrus

Pi light

Let there be light: a voice activated LED for home automation

Sandro Zimmermann[1,*]

[1]*Fachhochschule Graubünden*

[*]*E-Mail Adresse: sandro.zimmermann@stud.fhgr.ch*

December 16, 2025

Abstract

A home automation system will monitor and control a home's electrical components such as climate, appliances, entertainment systems and of course lighting. In commercial buildings automation is commonly found. Light or pressure sensors, motion detectors automatically open doors, turn on lights and activate escalators. In private homes, this features aren't found. This brought to the question, how achievable is it to bring automation to our homes. And since every home has light sources, the goal of this project is to control an LED with voice commands.
The code is run on a Raspberry Pi 4 Model B. A simple offline voice recognition software allows the control of an LED with english words. Though it's simplicity makes it difficult to recognize even "single word" commands. Nonetheless it's possible to achieve home automation. The physical limiting factors of a Raspberry Pi make it difficult to apply this project's proof of concept to a real household. Additional Hardware is required, which may or may not allow integration with a Raspberry Pi.

1 Introduction

This paper explains how to setup your own Raspberry Pi, the necessary hardware and software. The available Python code makes it possible to turn an LED on and off with speech recognition. The recognized voice commands are: *fire* to turn on the LED and *dark* to turn it off.

2 Methods

2.1 Materials

2.1.1 Hardware

The necessary hardware to compleat this project is as follows:

- Raspberry Pi 4 Model B
- Micro SD card, minimum capacity 8 GB
- USB Microphone
- Electronic breadboard
- Jumper wires
- 2.2V LED
- 100 Ohm resistor

Calculations and schematics

For calculating the resistor capacity Ohm's law[1] is applied. The GPIO pin, that will be used, has a source voltage of 3.3V[2]. On the LED Store's online page[3], the forward voltage (2.2V) and forward current (20mA) are displayed. Inputting the values in the formula[4]

$$R = \frac{V_s - V_{LED}}{I_{LED}}$$

results in 55 Ohm. 55 Ohm is the minimum capacity a resistor has to have, for safely operating the circuit. A 100 Ohm resistor was used, because 55 Ohm resistors are not readily available. A higher resistance than necessary, will result in the LED being dimmer.

The circuit's schematic was generated on the Website (*LED Calculator*, accessed: 2025-12-14) and looks like this:

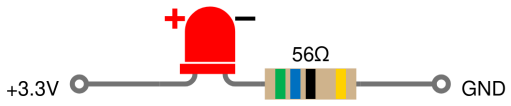

Figure 1: LED circuit schematic

Based on the schematic the Raspberry Pi and the electrical components, need to be plugged in similar to this:

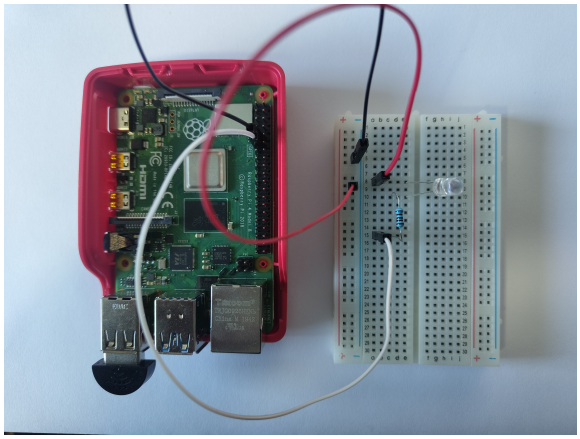

Figure 2: Raspberry Pi and electrical components

The white wire is plugged into the GPIO 17 pin, while the black wire is right above it, into a ground pin.

[1] (Hagmann, 2017)
[2] (*Raspberry Pi 4 GPIO Pinout and Specifications*, accessed: 2025-12-14)
[3] (*The Pi Hut*, accessed: 2025-12-14)
[4] (*LED Resistor Calculator*, accessed: 2025-12-14)

2.1.2 Software

The codebase is available for download on the git repository[5]. The software used is listed in the following table:

OS	Version
Raspberry OS 64bit	6.0
Programming language	
Python	3.13.5
Libreries	
PyAudio	0.2.14
pocketsphinx	5.0.4
RPi.GPIO	0.7.1

2.2 Setup

2.2.1 OS

The Raspberry Pi is going to run with Raspberry Pi OS. It is the official operating system. The OS is installed on the micro SD card. The imager is downloaded from (*Raspberry Pi Imager*, accessed: 2025-12-14) website.

2.3 Python environment

The entire code is written in Python. The Python interpreter is preinstalled on Pi OS.

Virtual environment

To make sure, no compatibility issues arise. The code was run in a virtual environment

```
python3 -m venv .venv
source .venv/bin/activate
```

Installation of portaudio and pyaudio binaries

```
sudo apt install portaudio19-dev
   python3-pyaudio
```

Installation of PyAudio, SpeechRecognition, pocketsphinx and GPIO python libraries

```
pip install PyAudio
   SpeechRecognition pocketsphinx
   RPi.GPIO
```

- *PortAudio* is an open-source, audio I/O library. It lets you write simple audio programs[6]
- *PyAudio* provides Python bindings for PortAudio. With PyAudio, you can easily use Python to play and record audio[7]
- *SpeechRecognition* is a library for performing speech recognition, with support for several engines and APIs, online and offline[8]
- *PocketSphinx* is one of Carnegie Mellon University's open source large vocabulary, speaker-independent continuous speech recognition engines[9]
- *RPi.GPIO* package provides a Python module to control the GPIO on a Raspberry Pi[10]

3 Results

Turning an LED on an off with voice commands was successfully done. A demo video is available on the git repository [5].

[5](Zimmermann, accessed: 2025-12-14)
[6](*Portaudio*, accessed: 2025-12-15)
[7](*PyAudio*, accessed: 2025-12-15)
[8](*SpeechRecognition*, accessed: 2025-12-15)
[9](*Pocketsphinx*, accessed: 2025-12-15)
[10](*RPi.GPIO*, accessed: 2025-12-15)

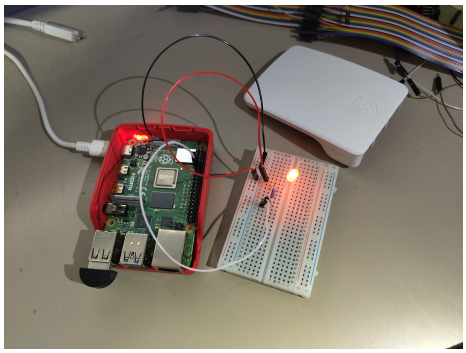

Figure 3: Raspberry Pi with turned on LED

To test how accurate the speech recognition library *PocketSphinx* is, typical commands, were spoken into the microphone, while the software was listening. The commands *fire* and *night* were also tested. Each command word was spoken 10 times. It resulted in this accuracy table:

Command	**Accuracy**
fire	50%
night	80%
turn	0%
add	0%
adjust	0%
rise	80%
set	10%
play	20%
lower	0%
show	80%
start	50%
stop	80%

The mean accuracy is 37.5% and the median is 35%.

4 Discussion

Setting up both LED control and speech recognition was surprisingly easy. The Raspberry Pi offers great Hardware support, while python has many software libraries, which cover a variety of use cases. For this project the two most useful tutorial were (*Control an LED with Raspberry Pi 4 and Python 3*, accessed: 2025-12-06) by *The Robotics Back-End* and (*Raspberry Pi Speech Recognition: The Complete 2025 Guide for Voice Control & Automation*, accessed: 2025-12-06) by *videosdk*. This two guides taught me what Hardware is required, how to determine compatibility, the base code for controlling LEDs and how to access the speech recognition features.

The most disappointing result is the accuracy of PocketSphinx. Active development has largely ceased and it has become very far from the state of the art. Perhaps a more recently developed Speech Model like Vosk[11] or even an online Model like Google Speech API [12] would have had better results.

With this results I can confidently answer the question on how achievable it is, to bring automation to private homes. Yes, it is achievable, a Raspberry Pi offers the tools to do so. But I don't believe one alone can manage an entire household. The physical limitations hold it back. The GPIO are great for small low voltage components, but can't directly be integrated into a 220V grid. The placement of microphones is crucial for voice control. In this project, it was plugged into the Raspberry and 50cm away. In any houshold almost every room would have to be equipped. One way this issue could be solved, is by using smart devices, which communicate via TCP/IP. The Pi could be used as central control. Since it has both Ethernet and WiFi capabilities, both wired and wireless devices could be integrated if they allow it.

[11](*Vosk*, accessed: 2025-12-16)

[12](*Google Speech API*, accessed: 2025-12-16)

References

Control an led with raspberry pi 4 and python 3. (accessed: 2025-12-06). Retrieved from https://roboticsbackend.com/raspberry-pi-control-led-python-3/

Google speech api. (accessed: 2025-12-16). Retrieved from https://ai.google.dev/gemini-api/docs/speech-generation

Hagmann, G. (2017). *Grundlagen der elektrotechnik*. AULA-Verlag GmbH. Retrieved from https://de.book-info.com/isbn/3-89104-804-1.htm

Led calculator. (accessed: 2025-12-14). Retrieved from https://ledcalculator.net/#p=3.3&v=2.2&c=20&n=1&o=w

Led resistor calculator. (accessed: 2025-12-14). Retrieved from https://ohmslawcalculator.com/led-resistor-calculator

The pi hut. (accessed: 2025-12-14). Retrieved from https://thepihut.com/products/ultimate-5mm-led-kit

Pocketsphinx. (accessed: 2025-12-15). Retrieved from https://pypi.org/project/pocketsphinx/

Portaudio. (accessed: 2025-12-15). Retrieved from https://www.portaudio.com/

Pyaudio. (accessed: 2025-12-15). Retrieved from https://people.csail.mit.edu/hubert/pyaudio/docs/

Raspberry pi 4 gpio pinout and specifications. (accessed: 2025-12-14). Retrieved from https://www.etechnophiles.com/raspberry-pi-4-gpio-pinout-specifications-and-schematic/#power-pins

Raspberry pi imager. (accessed: 2025-12-14). Retrieved from https://www.raspberrypi.com/software/operating-systems/

Raspberry pi speech recognition: The complete 2025 guide for voice control & automation. (accessed: 2025-12-06). Retrieved from https://www.videosdk.live/developer-hub/stt/raspberry-pi-speech-recognition

Rpi.gpio. (accessed: 2025-12-15). Retrieved from https://pypi.org/project/RPi.GPIO/

Speechrecognition. (accessed: 2025-12-15). Retrieved from https://pypi.org/project/SpeechRecognition/

Vosk. (accessed: 2025-12-16). Retrieved from https://pypi.org/project/vosk/

Zimmermann, S. (accessed: 2025-12-14). *Git pilight.* Retrieved from https://gitea.fhgr.ch/zimmersandro/pilight