iHouse: A Voice-Controlled, Centralized, Retrospective Smart Home

Benjamin Völker(✉), Tobias Schubert, and Bernd Becker

Chair of Computer Architecture, Faculty of Engineering,
Institute of Computer Science, Albert-Ludwigs-University Freiburg,
79110 Freiburg, Germany
{voelkerb,schubert,becker}@informatik.uni-freiburg.de

Abstract. Speech recognition in smart home systems has become popular in both, research and consumer areas. This paper introduces an innovative concept for a modular, customizable, and voice-controlled smart home system. The system combines the advantages of distributed and centralized processing to enable a secure as well as highly modular platform and allows to add existing non-smart components retrospectively into the smart environment. To interact with the system in the most comfortable way - and in particular without additional devices like smartphones - voice-controlling was added as the means of choice. The task of speech recognition is partitioned into decentral Wake-Up-Word (WUW) recognition and central continuous speech recognition to enable flexibility while maintaining security. This is achieved utilizing a novel WUW algorithm suitable to be executed on small microcontrollers which uses Mel Frequency Cepstral Coefficients as well as Dynamic Time Warping. A high rejection rate up to 99.93% was achieved, justifying the use of the algorithm as a voice trigger in the developed smart home system.

Keywords: Smart home · Retrospective home · Offline speech recognition · Wake-up-word recognition · Distributed speech processing

1 Introduction

Smart homes in general are habitats that provide their owners comfort, efficiency, security and convenience even if they are not at home. The provided support is achieved by incorporating common devices into smart objects to be able to control several features of the home like the lighting or heating automatically and more intelligent. Even though, this concept as well as corresponding open and commercial solutions are already available ([1–3] and many more), only minor households have adopted to this hype yet. According to [4], only 14% of all households in Germany for instance used at least one smart home component in 2014, leaving 86% of households not using any type of smart components. As reported, the reason for the actual quite low acceptance of smart homes is the missing compatibility between several providers of smart components as well as a high effort required to install these systems. Furthermore, a (new) smart

M. Magno et al. (Eds.): S-Cube 2016, LNICST 205, pp. 68–80, 2017.
DOI: 10.1007/978-3-319-61563-9_6

home solution requires the purchase of equipment most people typically already own and which is still working. As an example, most aging HiFi systems or televisions can not be controlled with a smartphone or by voice. However, this is not a sufficient reason for most people to buy new systems. It should be possible to retrospectively update the existing systems at low cost to interact as smart components within the smart home system.

Another apprehension related to smart home systems is security. People are afraid that using these systems might help other people to spy on them. Since the own home is a hideaway from the outside world, it should not be vulnerable to any kind of attack. Therefore, smart home systems should be highly secure and should not allow, for example, that the neighbour could turn off or on devices accidentally or intentionally.

An additional issue of existing smart home concepts is the lack of user-friendliness and ease of use. These systems often require a lot of maintenance and are either too complex or do not offer sufficient adjustment options. To control these systems, most providers offer smartphone or web apps. Indeed, the use of smartphones simplifies the tasks of changing settings and controlling several devices. But this simplification is restricted to situations where a person already holds the smartphone in his or her hand. To increase the ease of use, an interface should be used which works everywhere in the house and is carried around by everyone all the time. Thus, the perfect interface is voice. To switch on the light, a user could simply express the command *"House, turn on the light"*. Another benefit of using voice commands is that they can be personalized easily. As an example, the light could be also turned on by the command *"House, it's too dark"*.

The idea of using speech as an input technique for smart home systems is not novel. In [5] a remote speech interaction system to control entertainment devices using beam-forming and speaker-verification techniques has been proposed. More recently, [3] implemented a smart home system using contextual information and the human speech. However, it is not pointed out how the microphone data is acquired and how their systems can be expanded to a multi-room setup. Commercial systems like *Amazon*'s *Echo* [7] or the just announced *Google Home* [6] exist as well. However, the speech interaction is limited to the room in which the system is installed. Furthermore, these devices are continuously listening for fixed, not personalizable WUWs like *"Alexa"* or *"Ok Google"* and require a connection to external servers for continuous speech processing. This implies that a device connected to the internet is always listening regardless of whether a person in the house is speaking towards the system or privately communicating. Due to the "always online state", these systems are of high risk for the leakage of personal information. Regarding a more secure system, it should be guaranteed that voice data is processed completely offline.

In order to approach the stated issues of existing smart home solutions, the *iHouse* smart home system was developed which is highly customizable, user-friendly and handles sensitive voice data as secure as possible by distributing the task of Wake-Up-Word spotting and continuous speech recognition. Compared to e.g. approaches from Amazon and Google, the system presented here

can be used totally offline, avoiding in particular the "always online state" discussed before. To introduce the iHouse system in more detail, the remainder of this paper is structured as follows: In Sect. 2 an overview of the developed smart home system is given. The developed hardware and how existing devices are controlled (by means of some examples) is explained in Sect. 3. More details about the implementation of the software is depicted in Sect. 4, in which the server application is presented. Details and test results of the novel WUW spotting algorithm are further explained in Sect. 5. Concluding remarks and future work are provided in Sect. 6.

2 iHouse System Overview

The iHouse smart home system was developed to provide its users support and convenience during everyday's tasks. The **i** in the name of the system refers to the fact that the provided support is done as 'intelligent' as possible. Therefore, several devices like the lighting, heating or the television can be controlled and sensor data such as the energy consumption of a device or the room temperature can be monitored in a convenient way. Speech recognition is build on top, so that voice commands can be expressed to trigger certain actions like switching on the ceiling light or getting access to several information like the current temperature in a certain room. An overview of the system's workflow is shown in Fig. 1. If the user expresses a custom voice trigger like *"Ok House"*, the system responds with a certain sound indicating that it is now continuously listening for commands. Each utterance or sentence spoken afterwards, like *"Lights on"* or *"Room temperature"*, will be treated as a command. If such a command is recognized, the system will either execute an action or answer accordingly. Furthermore, the system automatically adds context information by triggering certain sensor measurements or by making use of past events and spatial information. The spatial information is obtained by evaluating in which room the command was triggered. Thus, one does not need to point out, for example, which specific light should be turned on. Furthermore, rules or scenarios can be set, so that for example all lights will be switched off automatically at midnight.

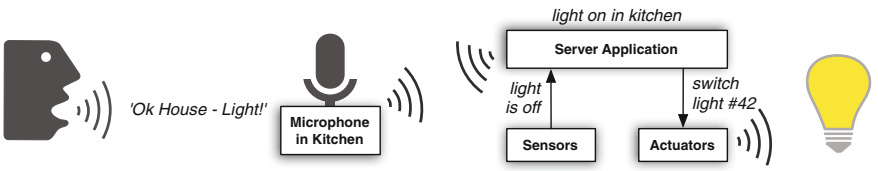

Fig. 1. Working principle of the iHouse smart home system for the example of switching on a light in the kitchen.

Several sensors can be added dynamically to the system to get information about the current temperature, humidity and brightness level. The system is

not limited to switching lights on and off, since other modules can be added as well. For example, one might add thermostats to the system so that the heating can be controlled as well or metering devices to be able to record and monitor the energy consumption of certain devices like the refrigerator. Each of these modules can be added, removed or replaced dynamically in order to provide a highly customizable setup comprising both decentral and central components.

2.1 Central vs. Decentral Processing

A smart home environment itself is distributed by its nature. It consists of several spatially separated places, each of them with multiple sensors and actuators like temperature sensors or thermostats. To connect with these devices, a centralized approach with a server application was selected. The centralized processing and execution simplifies dependent tasks and the interconnection between different rooms as well as the maintenance of the entire system. The server therefore acts as the central brain of the smart home with knowledge of past and present sensor data and user events. This information is then used to select the most suitable action to support the user. As an interface to the system, a server application by itself is not appropriate since it lacks fast remote access and portability. For this reason, decentral access to the system is provided by either using a smartphone application or voice. However, the arrangement of a home with its multiple distributed rooms hampers the use of speech recognition, since either the user must be equipped with a microphone or every room. If the user is equipped with a microphone, a device with a portable power supply is needed which - like a smartphone - lacks portability and, thus, also accessibility during recharging. If, on the other hand, every room is equipped with a microphone, a portable power supply would not be essential since power sockets are typically available in every room. The decentralization of these devices into multiple rooms leads to different possible approaches. Each microphone module can either independently evaluate and execute a given voice command by itself or has to stream the speech signal to the centralized server which further recognizes the command. Each of the two methods has its benefits and drawbacks. If the devices would recognize commands by their own, each of them needs to have enough computational power to perform speech recognition on a large vocabulary. However, devices with sufficient computational power are typically pretty expensive. The centralized approach requires only one device with enough power to perform speech recognition which is the server. But this approach also requires that the voice data is transmitted to this server over a potential insecure wireless channel. Additionally, receiving and processing voice data from multiple microphone modules is a high overhead even for high-end devices. For the stated reasons, a combination of both approaches is used by distributing the task of voice controlling to decentral WUW recognition on microphone modules and central command detection on a server. The microphone modules recognize any WUW recorded in advance. This recognition is done directly on the module without streaming any audio data to a server. Only if a WUW is recognized, audio data is continuously streamed to the server application. This ensures, that the potentially

insecure wireless channel for streaming is only used if the user wants to express a command and, thus, is aware of it. To minimize eavesdropping, audio data may also be encrypted before it is sent to the server application. Additionally, the outsourcing of the WUW detection to the microphone modules reduces the overhead for the receiving and processing of audio data on the server. Moreover, the microphone modules have to compute less complex speech recognition tasks which reduces the requirements for their hardware.

3 Hardware Components

The main components of the system are the microphone modules placed in every room, the receiver station attached to the server and the server application. The microphone modules serve to recognize spoken WUWs and transmit the speech signals spoken afterwards to the receiver station. The receiver station handles the incoming speech data as well as the transmission of audio back to a certain microphone module for audio feedback. The server application handles the recognition of voice commands and executes them. The overall data flow of the audio data between a microphone module and the server is sketched in Fig. 2.

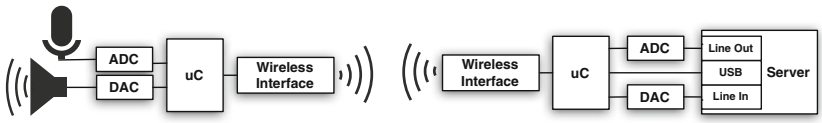

Fig. 2. An overview of the audio data flow between microphone module and server application. The most important hardware components and their connection are shown.

An *ARM Cortex M4* microcontroller is used as the main processing unit for the microphone modules and the receiver station since it offers just enough computational power to sample and process the audio data. Moreover, it features a build in *Analog-to-Digital-Converter*, a *Digital-to-Analog-Converter*, and an *SPI*-bus. SPI is needed for communication with the wireless interface which is a *nRF24L01* from *Nordic Semiconductor*. To be compatible to common PCs, the audio data connection between the receiver station and the server application is achieved over the standard analog soundcard's line in and line out plugs. Furthermore, a USB connection is required for communicating with the server application.

As mentioned, the retrospective use of existing systems in the home like the television or lighting is preferred by most people instead of buying new devices. Therefore, additional modules are used to update these systems so that they can interact as smart components with the iHouse smart home system. To switch and meter lights or generic devices with a power plug, off-the-shelf switchable sockets are used, which communicate via 433 MHz. Existing infrared-light based

systems are controlled with a self-build module featuring an IR-receiver to record commands and an IR-sender to control systems like the TV. The overall connections between the main components and some additional modules is shown in Fig. 3.

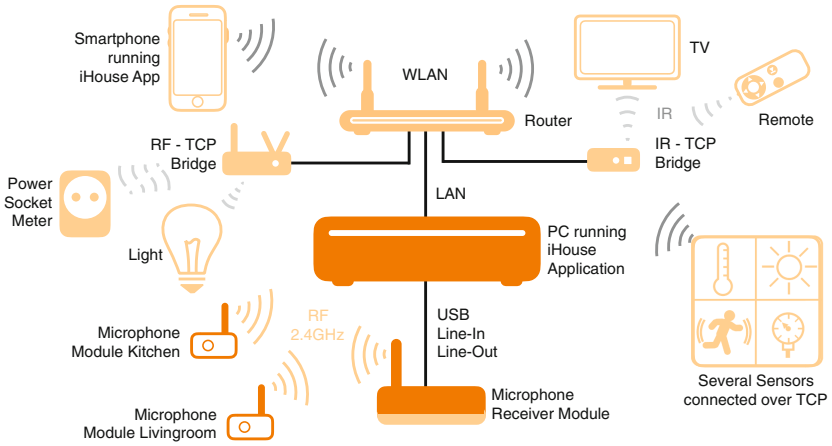

Fig. 3. Schematic overview of the connection of several hardware components in the iHouse smart home system.

4 Server Application

The server application is the centerpiece of the entire system. Commands are extracted from the voice stream, data of all sensors is collected and actuators are controlled. A graphic user interface makes it possible to add, remove and setup modules as well as to control them. In general, each device has a name, a picture and a device type. While the name and the picture are only for identification and further highlighting, the device type decides which kind of sub-device it is and what smart object it actually represents and, thus, what actions can be performed and what states can be queried. Furthermore, different graphic user interfaces for controlling the device or changing its settings are displayed depending on the device type. One example for such a sub-device is a switchable socket. The available actions for this sub-device are *switching on* and *switching off* the socket, while the available state is whether the socket is currently turned *on* or *off*. The sub-device stores ID information of the physical device in order to establish a physical connection. Such a connection is done either using *TCP* or a more rudimentary protocols via e.g. 433 MHz depending on the smart object. Data collection is achieved using a hierarchic star network structure with the server as the main data sink. If data can not be exchanged with the target platform directly because specialized communication hardware is needed, communication-bridge modules are added to the network which represent the

data sink for all devices communicating with this specialized hardware. These modules further send the data to the server application over standardized protocols such as TCP. As an example, a TCP to infrared-light-bridge is used to be able to control existing television or HiFi-systems.

Voice commands are build on top and can be customized to the personal needs of the user. One can choose the command that needs to be spoken, the corresponding action as well as a text that is read out after the command is executed. Every command has a handler that is executed if the command is recognized. This handler is either a simple function that reads a response text with certain information like the temperature in a room or a more complex function that performs actions of certain devices like turning on the heating.

The graphic user interface of the server application allows to control the smart objects directly within the application. The main setup of the server, running the application is shown in Fig. 4.

Fig. 4. The iHouse smart home server application running on a PC connected to the receiver module over USB, Line in and Line out. A microphone module is shown on the right side. On the left side of the app, the room of interest can be selected. In the center, all available devices in this selected room are shown and can be controlled.

Voice controlling can be activated by either speaking the WUW in a room equipped with a microphone or by triggering a specific keystroke at the desktop computer. The application will automatically display a view showing all recognized commands as well as textual and graphical answers.

In order to add time or action based rules to a device, information and sensor data of other devices need to be accessed, too. If, for example, the heating should automatically adapt to the temperature in the room or to the time of day, those information must be accessible to the heating object. This information propagation is achieved by a *publish* and *subscribe* model. If a new temperature

sensor reading is available, this data is *published* by the sensor object. Other objects interested in the room temperature can now *subscribe* this value and get notified on each new reading. The server application can be seen as a modular platform that can be further optimized and extended according to the preferences of a particular user.

5 The Wake-Up-Word Spotting Algorithm

To enable voice controlling from a specific room of the house, a microphone module needs to be installed as described in Sect. 3. After detecting a pre-recorded WUW, speech data is transmitted to the server for continuous command detection. The detection of the WUW is, however, task of the microphone module itself. To reduce the cost and size of these modules, only inexpensive microcontrollers with limited resources are used, which provide only a small amount of computational power and internal memory (see Sect. 3). Therefore, they are not capable of handling the high load of traditional speech recognition techniques. Optimized algorithms to detect keywords on low-level microcontrollers are proposed in literature but are not suitable for real-time applications. In [8] *Linear Prediction Coefficients* with *Hidden Markov Models* are implemented on an 8 bit, 16 MHz microcontroller resulting in an average recognition time of 15 to 17 s. Another approach using *Linear Prediction Cepstral Coefficients* on an 8 bit MCU, clocked at 40 MHz with a specialized *Dynamic Time Warping* (DTW) algorithm is used in [9]. However, the authors do not give any hint whether their system can be applied to real-time applications or not.

For small vocabulary speech recognition systems such as keyword spotting systems, the template matching approach is widely used ([9,10]). The specially developed WUW spotting algorithm adopts this approach but makes it suitable for real-time applications on low-level microcontrollers. An overview of the realized approach is shown in Fig. 5, in which additional as well as optimized stages (compared to the traditional flow) are highlighted in orange.

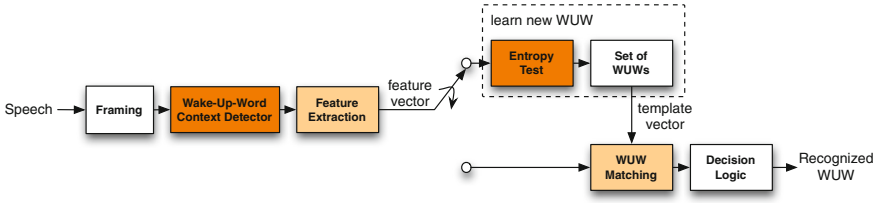

Fig. 5. An overview of the proposed WUW-spotting algorithm. Blocks in orange are added to the standard template matching pipeline while blocks in light orange show highly optimized implementations to maintain real-time behavior for low-level systems. (Color figure online)

To evaluate the dynamic process speech, the simplifying assumption is made that speech is statistically stationary on short time scales. Therefore, features

are extracted and evaluated on a frame by frame basis over a specific time interval called window. For the algorithm, a frame length of 10 ms with a window length of 20 ms and a sampling frequency of 16 kHz is used. Discussion about proper frame and window durations for speech processing can be found in [11]. A WUW Context Detector is developed to distinguish speech directed towards the system from speech that is not. Humans usually get the attention of other people by hyper-articulating their names or by using interjections like *"excuse me"*. This is often coherent with a preceding and succeeding silence to highlight the spoken utterance even more. By applying this human behaviour to a more general model, the WUW context is defined as a preceding voiceless segment followed by a voiced segment in the length of a stored WUW succeeded by an additional voiceless segment. The WUW Context Detector uses a Voice Activity Detector (VAD) [12] to decide whether voice is present or not and thresholds to evaluate the length of each segment.

The feature extraction is optimized for the calculation of *Mel Frequency Cepstral Coefficients* (MFCCs) with 26 filter banks linear distributed in the mel-scale between 100 Hz and 8000 Hz to model the speaker and the spoken WUW. MFCC features have successfully been applied to speech recognition tasks since their introduction in the 70's [14]. An overview of the steps done to compute the MFCCs is shown in Fig. 6.

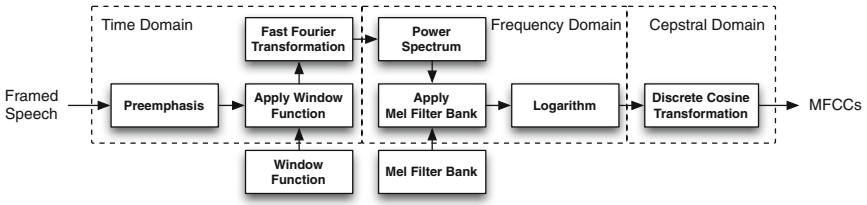

Fig. 6. An overview of the extraction of Mel Frequency Cepstral Coefficients.

Since the major goal is to enable the execution of the algorithm on low-level platforms, the MFCC calculation is optimized as follows. The Hamming window and the independent cosine values used for the Discrete Cosine Transformation calculation are precomputed and stored in a LookUp Table. Furthermore, only the first half of the Hamming window is stored due to its symmetry. An efficient Fast Fourier Transformation for power-of-2 lengths ($N_{FFT} = 512$) is used and the mel filter banks are optimized by only calculating the relevant values for each triangular filter. In the WUW matching block, a sequence of extracted feature vectors is tested consecutively against each stored template. If strong deviation is found between the length of the template and the test sequence, a mismatch is assumed and further matching is canceled. This reduces the computational complexity and improves the systems response time. If no strong deviation is found, further analysis using an optimized DTW algorithm with a *Euclidean Distance* measure is applied to both sequences. DTW has successfully been applied to

speech recognition problems ([8,13]). However, its computational complexity is $\mathcal{O}(n \cdot m)$, with n and m being the lengths of two sequences to compare. Depending on the frame duration and the length of the WUW, the extracted sequence of feature vectors can get quite large resulting in a huge matrix that needs to be explored. This exploration is not only computational complex for an embedded device but might also become a memory problem depending on the underlying hardware platform. In order to solve this problem the *Sakoe-Chiba band* [14] with distance r is used to reduce the search space. The complexity is even further reduced by using a sub-word matching technique: Each feature vector sequence is divided into k subsequences of equal length. The corresponding subsequences of the test and template sequence are successively matched with DTW and the distance of each match is stored. The total distance is finally estimated by the mean of the distances of all subsequences. Therefore, the complexity is decreased to

$$n \cdot r + m \cdot r - k \cdot r^2 \rightarrow \mathcal{O}\left(r \cdot \max\left(n, m\right)\right). \tag{1}$$

If the total distance of a test sequence drops below an empirically defined threshold, the sequence and, thus, the utterance said, is classified as a WUW. If the system is in the learning state, an Entropy Test ensures that the template recording is suitable to be used as a WUW by requiring its length to be greater 0.2 s and smaller 2 s and the environment in which it is recorded to be almost noise-free.

With the performed optimizations in the template matching and feature extraction pipeline, the average recognition time is 300 ms for a single stored WUW of 1.3 s length. The recognition time scales according to Eq. 1 with the length of the WUW and linear with the total number of stored WUWs.

5.1 Experimental Setup and Results

To evaluate the proposed system, a database of 11 different speakers - two female and nine male - and two different scenarios (WUW and continuous speech) was recorded for three different distances (1, 3 and 5 m) between the microphone and speaker. Two WUWs were chosen by each subject individually and were recorded in a noiseless environment at a distance of 1 m to the microphone. In the WUW scenario several utterances are spoken successively with a 1 s silence interval between each utterance to ensures that the WUW Context Detector assumes it to be spoken in the WUW context during evaluation. Both WUWs occur at least three times in this scenario. The purpose of this scenario is to evaluate the performance of the feature extraction and matching block of the proposed algorithm. In the continuous speech scenario the subjects were reading texts from a book to simulate a regular conversation. The purpose of this scenario is to test the WUW Context Detector which should reject all speech not spoken in a WUW context. The performance of the system was measured in terms of *correct acceptance* (CA) and *correct rejection* (CR) rates [12], since WUW or keyword spotting systems need a measure of rejection. Rejection is the ability

of the system to detect and reject *Out Of Vocabulary* utterances. CA and CR rates can be obtained by

$$CA = \frac{TP}{TP + FN}, \qquad CR = \frac{\#\,\text{words} - FP}{\#\,\text{words}}, \tag{2}$$

where TP is the number of *true positives*, FN is the number of *false negatives* and FP is the number of *false positives*. As Table 1 shows, the CR rate is as high as 99.69% for a CA rate of 100% in the speech scenario. In the WUW scenario, however, the CR rate drops to 89.66%. Depending on the application of the system, a large CR rate is more important than a high CA rate. Therefore, the system was re-evaluated by omitting the WUW with the largest distance measure to the corresponding template recording. In other words, the worst pronunciation of each WUW was deleted for every person. This leads to an overall CA rate of 67.86%, meaning that approximately two out of three WUW would still be recognized correctly. This reduction causes that the CR rate increases to 99.93% in the speech scenario and to 96.91% in the WUW scenario. 99.93% is equivalent to one erroneously trigger around 1400 words.

Table 1. System performance depending on the scenario evaluated for two CA rates.

Scenario	# Words	# WUW	CA rate	# TP	# FP	# TN	# FN	CR rate
Speech	4149	0	100%	0	13	4136	0	99.69%
			67.86%	0	3	4146	0	**99.93%**
WUW	1392	282	100%	282	144	1248	0	89.66%
			67.86%	192	43	1067	90	**96.91%**
Total	5541	282	100%	282	157	5102	0	97.17%
			67.86%	192	46	5213	90	**99.17%**

6 Conclusion

This paper proposes a modular and expandable smart home solution, which is user-friendly in installation and usage. Additionally, it is secure by processing all data offline. The system consists of external microphones and a central server application which interacts with different smart components like light switches, the heating, or the TV. To improve the ease of use, voice controlling was added to the smart home system as the main interface. This enables the interaction with the system through voice commands that, unlike existing systems, are processed completely offline and can be customized to the personal needs of the user. To be able to interact with this system from every room of the house, microphone modules were designed which can be installed easily into existing households since all communication to the server is done wireless. To separate speech directed to the system from speech not directed to the system, a novel WUW algorithm

was developed which recognizes predefined WUWs spoken in an alerting context of getting attention. Thus, all voice commands directed to the system must be spoken with a leading WUW like *"Hey House [pause] light on!"*. Speech data is only transmitted to the server if the leading WUW is recognized by the embedded platform. The WUW algorithm was improved in terms of computational complexity to enable the execution on embedded platforms with less computational power. An evaluation was done on a speech database of 11 different speakers. The algorithm achieved an overall correct rejection ratio of 99.17% for a correct acceptance rate of 67.86%. The rejection of speech not expressed in the WUW context is up to 99.93%. Future work for the smart home system will focus on extending its scope and smartness. This will be achieved by utilizing the implemented publish and subscribe model to make the system even more context aware and, thus, to provide additional convenience and aid for the user. Nevertheless, first feedback obtained from test users and other participants indicates that voice controlling is a key feature and will enhance smart homes to bring them into more households any time soon.

References

1. Nest Labs - Nest. http://www.nest.com
2. FHEM: A perl server for house automation. http://www.fhem.de
3. Han, Y., Hyun, J., Jeong, T., Yoo, J., Hong, J.: A smart home control system based on context and human speech. In: 18th IEEE International Conference on Advanced Communication Technology, pp. 165–169. IEEE Press, PyeongChang (2016)
4. Illek, C.P.: Smart home in Deutschland. Survey, Bitkom Research GmbH (2014). http://www.bitkom.org/Publikationen/2014/Studien/Smart-Home-in-Deutschland-Praesentation/Praesentation-Smart-Home.pdf
5. Potamitis, I., Georgila, K., Fakotakis, N., Kokkinakis, G.K.: An integrated system for smart-home control of appliances based on remote speech interaction. In: 8th International Conference on Speech Communication and Technology, pp. 2197–2200. ISCA, Geneva (2003)
6. GoogleHome. http://home.google.com
7. Amazon Echo. http://www.amazon.com
8. Thiang, D.W.: Limited speech recognition for controlling movement of mobile robot implemented on atmega162 microcontroller. In: 1st IEEE International Conference on Computer and Automation Engineering, pp. 347–350. IEEE Press, Bangkok (2009)
9. Yuanyuan, S., Jia, L., Runsheng, L.: IdentificationSingle-chip speech recognition system based on 8051 microcontroller core. IEEE Trans. Consum. Electron. **47**, 149–153 (2001)
10. Barakat, M.S., Ritz, C.H., Stirling, D.A.: Keyword spotting based on the analysis of template matching distances. In: 5th IEEE International Conference on Signal Processing and Communication Systems, pp. 1–6. IEEE Press, Honolulu (2011)
11. Picone, J.W.: Signal modeling techniques in speech recognition. Proc. IEEE **81**, 1215–1247 (1993)
12. Këpuska, V.Z., Klein, T.B.: A novel wake-up-word speech recognition system, wake-up-word recognition task, technology and evaluation. Nonlinear Anal. Theor. Methods Appl. **71**, e2772–e2789 (2009)

13. Zehetner, A., Hagmuller, M., Pernkopf, F.: Wake-up-word spotting for mobile systems. In: 22nd IEEE European Conference on Signal Processing Conference, pp. 1472–1476. IEEE Press, Lisbon (2014)
14. Sakoe, H., Chiba, S.: Dynamic programming algorithm optimization for spoken word recognition. IEEE Trans. Acoust. Speech Signal Process. **26**, 43–49 (1978)