 DOI: 10.3217/978-3-85125-663-5-24

Control of Autonomous Mobile Robot using Voice Command

Trung Quoc Nguyen[1], Peter Nauth[2] and Sudeep Sharan[3]

Abstract— **Controlling the machine by voice or speech has always aroused the curiosity of humans. After many pieces of research and developments, the voice recognition system becomes an important and comfortable system to communicate with machines in day today's life. In this paper, a voice control software system is created and integrated with the mapping algorithm available in Robotic Operating Systems (ROS) and implemented with a mobile robot Volksbot [Figure 1]. This paper also expresses the development of a Graphical User Interface (GUI) with different tabs to control the system in different ways, first by clicking voice command to navigate the robot to its destination in the available map, and second by typing the command in words or numbers. If the commands are mistaken, then the user can abort the commands by clicking stop tabs. In order to test the voice system accuracy, the experiment is performed with different voices as well as with different pitches. This work also shows the results of the accuracy of reaching the destination's room in the map.**

I. INTRODUCTION

Service robots have recently become an important part of human society. The number of robots is increasing rapidly in the market. The more user-friendly robot execution becomes, the more easily people can control it. One easy way to control a robot is by means of voice. Human often finds it more suitable to give order by saying something rather than typing the code. Voice communication between robots and users plays a critical part in any applications [1,2]. The user can command the robot to execute the task by speaking, even if the user has no knowledge of machines and computers.

In this article, a voice recognition system is integrated with a mapping algorithm which is implemented on an intelligent mobile robot. Using ROS firmware [3], the system controls the robot to move to different locations.

Section 2 introduces the system structure of a robot control system. Section 3 describes the voice recognition. Section 4 expresses the system implementation of the voice control system on a mobile robot including the mapping algorithm integrated using ROS. The experimental results and testing are discussed in section 5, followed by the conclusion in section 6.

II. DESIGN OF VOICE CONTROL SYSTEM

The mobile robot system is controlled by two computer systems. One is attached to the mobile system and the other is set in a fixed place and used as a virtual system. Both computers are connected to each other via WLAN. Static virtual system (Computer 1) is connected to a microphone, which receives the voice signals from the user and transfers them to the robot system computer (Computer 2). The voice recognition detects the signal here and then the robot executes the action according to the given voice command.

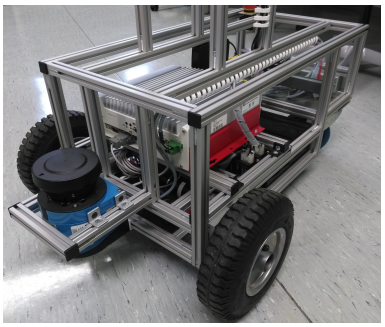

Fig. 1. Mobile Robot "Volksbot"

The mobile system is a Volksbot developed by Fraunhofer Institute. It has two front actuated wheels and two back castor-wheels. The encoders are also connected with the wheel motors to achieve odometry data. A Lenovo Thinkpad laptop is placed on the volksbot which is responsible for all processes of computation. The laser range finder LMS100 is mounted in front of the volksbot, which scans the entire surrounding environment and provides information about the environment, which helps to create a map of an unknown environment. The operating range and angle of view are 0.5m to 20m and 270 degrees respectively. The sensor is used because of its high update rate of 50 Hz. A 3D Kinect-camera is connected to the volksbot in order to generate a 3D cloud map of the environment. Kinect Xbox 360 is used because of its high adaptation and availability. The Kinect camera is connected to the laptop placed on the mobile system and a microphone is connected to a static computer on the desktop (Computer 1), which communicates with the laptop (Computer 2) on the mobile robot via secure shell (SSH). The structure of the mobile voice control system is shown in Figure 2.

The voice control robot system consists of different modules, such as automatic speech recognition (ASR) module and the control module. ASR system analyzes the user's voice and then transfers the command to the robot to execute the user's request. The control module understands the voice command and performs the action accordingly. The process system is expressed in Figure 3.

[1]Trung Quoc Nguyen is with Faculty of Computer Science and Engineering, Frankfurt University of Applied Sciences, Nibelungenplatz 1, Frankfurt am Main, Germany trungquo@stud.fra-uas.de

[2]Peter Nauth is with Faculty of Computer Science and Engineering, Frankfurt University of Applied Sciences, Nibelungenplatz 1, Frankfurt am Main, Germany pnauth@fb2.fra-uas.de

[3]Sudeep Sharan is with Faculty of Computer Science and Engineering, Frankfurt University of Applied Sciences, Nibelungenplatz 1, Frankfurt am Main, Germany s.sharan@fb2.fra-uas.de

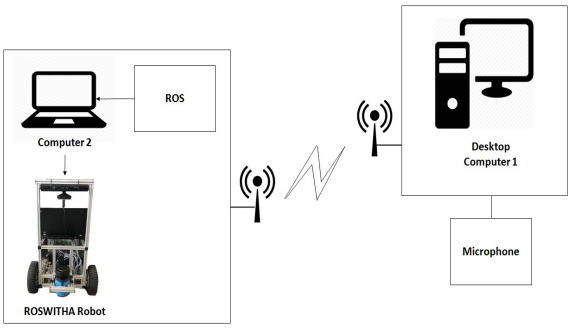

Fig. 2. Hardware connection of the mobile control system

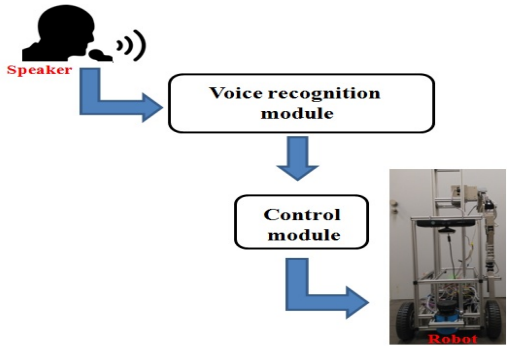

Fig. 3. Command control diagram

The system of the voice command is complicated. When a user commands to the robot, the voice is sent from the microphone to the speech recognition module. Then the module tries to detect the user's voice by extracting features from it. Voice features are analyzed using a trained database, which is given to the system by the user. At the end of the speech recognition process, a text version of the user's command is generated.

The captured commands will be transferred to the control module. Then the control module executes the commands as it gets from the speech recognition module and makes the mobile system move accordingly.

There are few basic trained sentences in the database as follows:
GO TO ROOM TWO ZERO THREE
GO TO ROOM TWO ZERO FOUR
GO TO ROOM TWO ZERO FIVE
GO TO ROOM TWO ZERO SIX
GO TO AUTONOMOUS LAB (Home location)

If the command of the user is any of the above sentences, then the robot executes the command and moves towards that room. The extension of the commands can enable the robot to move to multiple rooms if there are multiple commands such as "GO TO ROOM TWO ZERO THREE then GO TO ROOM TWO ZERO FIVE".

To navigate in an unknown environment, the robot has to create a map and navigate to the proper path to reach the destination. The robot should also have the ability to avoid object. The SLAM algorithm RTAB-MAP [4], which is already available in ROS as a package, is implemented on the mobile system to enable the robot to draw the map. With the help of the navigation stack node, the robot navigates towards the destination by planning the path and avoiding the obstacles in the path. All the names of destination rooms on the map are trained in the voice control module.

III. VOICE RECOGNITION

In order to control the autonomous mobile robot by voice, it is necessary to employ a speech recognition system inside the robot. In this article, the CMU Sphinx recognition system is used. It is a leading toolkit for voice detection which has been developed for a long time at Carnegie Mellon University. CMU Sphinx contains many powerful packages which are used for different situations and environments. In this work, Pocketsphinx, Sphinxbase, and Sphinxtrain are used to develop the voice recognition system.

Pocketsphinx is a lightweight version specialized for mobile devices. Other versions of CMU Sphinx such as Sphinx-II and Sphinx-III are complex, require many memories and often take considerable time to translate the speech, which is favorably used for the static system. On the other hand, Pocketsphinx is aimed for portability, simplicity of implementation and memory efficiency, which is suitable to be installed on the mobile robot. The decoder architecture, which is explained in [5], consists of three consecutive search strategies. Each latter search strategy uses the result of the previous and narrows the search space. Sphinxbase is a support library accompanied by Pocketsphinx. Sphinxtrain is needed during the processing of adapting and training acoustic models, which will be discussed later.

In our system, the process of designing a speech recognition system is divided into four parts:

- Build a dictionary
- Build a language model
- Adapt an existing acoustic model
- Provide decoder models in the previous three steps to start the detection processing

A dictionary contains all words that the robot needs to detect, together with the corresponding sequences of phonemes. An acoustic model describes how likely an acoustic realization is given that the text is known. Finally, a language model illustrates the likelihood of the next word in the sequence when the previous words have been detected [5,6].

IV. SYSTEM IMPLEMENTATION

The implementation of the system is structured in ROS and integrated with the mobile system as shown in figure 4.

The system is structured in ROS by using different packages for controlling hardware including *freenect_launch* driver for the Kinect camera, *lms1xx* for the laser scanner and *volksbot_driver* for driving the volksbot. SLAM algorithm RTAB-Map is implemented in the system, which is available in ROS as *rtabmap*. The RTAB-Map receives the data from odometry, laser scanner and Kinect camera to create a 3D

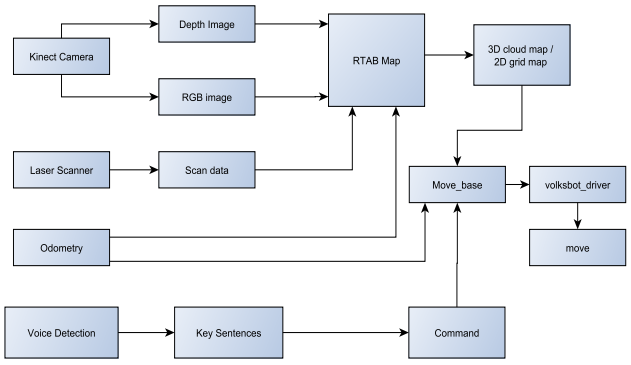

Fig. 4. ROS structure for building autonomous robot system

cloud map/2D grid map of an unknown environment. The 2D grid map is sent to the *move_base* [5] node. The information from the *move_base* node is transferred to the mobile robot so that it navigates according to the user's command.

A. Building voice recognition module

The recognition module is constructed by first building the dictionary, the language model and the acoustic model, then initializing the language decoder with these configurations. The dictionary should include all the words that the system is going to detect, otherwise it cannot detect correctly. First, a corpus file, which is a list of needed sentences, was created. All sentence commands should be included in this file. Next, from sentences in corpus file every word is extracted and stored in another file, together with their corresponding sequence of phonemes. This is the main dictionary file, which should have extension of *.dict*.

During the recognition process, the decoder looks in the dictionary to find the best sequence of phonemes that matches with the acoustic properties of the voice and picks the corresponding words in the dictionary for the next steps. However, providing a dictionary is not enough. A language model must also be generated, which specifies the probability of the next word given existing word. There are three ways to build a language model, which is explained in more details in [6]. The first way is using a keyword list. This list is just like a corpus file, with each sentence linking with a threshold. Long sentences should have a bigger threshold than short sentences. The second way is to use grammar. The third way, also the most complex way, is using a statistical language model. It enables users to use natural language rather than just simple command.

The statistical language model is a file in which probabilities of words and word combination are calculated. These probabilities illustrate how often a phrase or sentence is said by a user. Different words and phrases have different probabilities. The determination of probabilities is based on the input corpus file. One of the largest advantages of statistical language model is that it can detect other combinations than just ones in the corpus. For example, in corpus file, two sentences 'GO TO ROOM TWO ZERO SIX' and 'GO TO ROOM TWO ZERO FIVE' are listed. They have the largest probabilities in the language model. However, probabilities of other sentence combinations still exist, such as GO TO ROOM FIVE ZERO SIX. The user is able to speak more freely. The robot is therefore friendlier to human.

There are three common forms of language model: text ARPA format, a binary BIN format, and a binary DMP format [6]. A text ARPA file has the extension *.lm*. Although it takes more space, it is editable. On the other hand, a binary BIN file is a compressed version of ARPA file, with the extension of *.lm.bin*. It takes less space and loads faster. The last form is obsolete.

With the task of building an autonomous mobile robot, the data set is small. Therefore, it is possible to use CMU Tool directly on the website LMTool page of CMU by providing a corpus file. With corpus file of large size or with language other than English, it can be done by SRI Language Modeling Toolkit (SRILM) or CMUCLMTK [6]

Finally, an acoustic model is needed to complete the configuration of the system. Normally, a standard acoustic model such as CMUS Sphinx model for US English is suitable for dealing with a simple recognition task, since it has been developed and optimized for many years. However, in some special cases where users speak with slightly different accents, such as UK English or Indian English, adapting the standard model to user's voice may improve the performance greatly.

The prerequisite for building an acoustic model is an existing corpus file of all necessary sentences. In this step, each sentence is accompanied by a unique ID. These IDs are then stored in an ID file.

A corresponding dictionary for all words in the corpus file is also used to adapt acoustic model. In order to adapt the standard acoustic model, a recording of user speaking each of those sentences is given to the system. Each sentence is recorded independently at sampling rate of 16 kHz in mono with a single channel and stored in a single audio file *.wav*, which is named according to the ID of each sentence. The sound recorder Audacity is used in this project for recording and editing audio files. The third step is extracting acoustic features of audio files for later adapting process. This is done using the Sphinxbase[6]. Each audio generates a feature file with the extension $.mfc$. These feature files, together with standard acoustic model, dictionary, corpus file and ID files are exploited to collect statistics by using the Sphinxtrain command *bw*[6].

In the last step, Maximum Likelihood Linear Regression (MLLR) transform is deployed. Its suitability with a small dataset and online adaptation makes it a good adaptation method for the autonomous system[6]. The result of adapting process is a file called $mllr\_matrix$, which is passed to the decoder in Algorithm 1.

B. Graphical user interface of the system

The graphical user interface (GUI) is responsible for interacting with the entire system. Using GUI, users neither

Algorithm 1: DETECTING SPEECH

```
Result: A Phrase is detected
1  begin
2      initialize configuration for recognition process by
         choosing dictionary, standard acoustic model and
         language model and mllr_matrix;;
3      initialize decoder with existing configuration;
4      using Pyaudio to set up portaudio system;
5      open a stream on input device to record audio;
6      start the stream;
7      set the decoder to start decoding utterance;
8      while true do
9          read frames from stream and store it in buffer;
10         if buffer not None then
11             detect word from buffer and store it; if last
                 feed audio buffer does not contained
                 speech then
12                 stop decoding utterance;
13                 output the hypothesis phrase that is
                     detected;
```

have to type complex command to control the robot, nor they have to understand the mechanism behind the system. Figure 5 shows the window of a GUI.

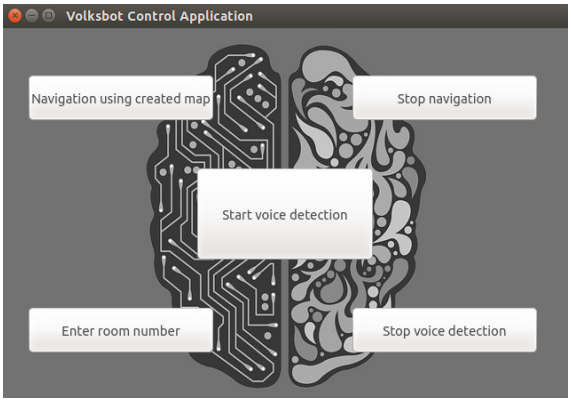

Fig. 5. Main window of a graphical user interface

In the GUI, five tabs are created to control the robot system in different aspects:

- Navigation using a created map: choose a map from the database and make the system ready to navigate in that environment.
- Stop navigation: stop all commands and make the system stop at any point.
- Start voice detection: initialize the voice detection function of the system and make the speaker ready to speak.
- Enter room number: The robot can be controlled by typing numbers, if the robot fails to recognize the voice.
- Stop voice detection: stop the voice command, if the speaker speaks the wrong command.

V. EXPERIMENT & TESTING

In the real experiment and testing, when the user speaks a word or sentence to the microphone, the system checks the voice with the trained database and if the voice matches the robot will execute according to the command.

In this work, the voice commands in the system are trained to go to the autonomous lab, go to room 203, go to room 204, go to room 205, go to room 206. In the real scenario, after the user gives the instruction through the microphone, the signal goes to the voice recognition module and then processes to the robot control system. The entire experiment was performed in an environment of the second floor, Building 8 of Frankfurt University of Applied Sciences, Germany. First, the speech was recorded and saved in a database, then the database is generated as explained in section III. Once the voice command is detected, the speech command system recognizes that voice and provides a control signal to the control system of the robot and makes the robot perform the action according to the command received. The robot is moving in real time during the speech recognition process.

A. Testing

1) Testing voice recognition: The test of voice recognition was performed in two parts, the first test was to check the accuracy of detection of trained sentences, where the different commands were spoken, which told the robot to go to different destinations. Each command was spoken 10 times. The result, which was detected sentences, was shown on the computer screen. The word error rate of each detected sentence is calculated. It states that, if both original text and recognition text have a length of N words, I be the number of inserted words, D be the number of deleted words and S be the number of substituted words. The word error rate (WER) is: $WER = (I+D+S)/N$. The second test was performed by making three different people to command the robot to execute the action and this test was performed 5 times for each person and investigate the error rate of the voice detection. The test results of the trained sentences are shown in Table I and the test results of the voice recognition are shown in Table II

Sentences trained	Error rate
GO TO AUTONOMOUS LAB	1%
GO TO ROOM TWO ZERO SIX	0%
GO TO ROOM TWO ZERO FIVE	0%
GO TO ROOM TWO ZERO FOUR	1%
GO TO ROOM TWO ZERO THREE	0%

TABLE I
ERROR RATES OF TRAINED SENTENCES

B. Experimental Results

The experiment was done on the volksbot mobile platform of the robot ROSWITHA (**RO**bot **S**ystem **WITH** **A**utonomy), a self constructed assistive robot system under the roof of Frankfurt University of Applied Sciences, Germany. The robot navigated to different room numbers

Commands	Trials	Speaker 1	Speaker 2	Speaker 3
GO TO AUTONOMOUS LAB	5	5	5	5
GO TO ROOM TWO ZERO SIX	5	5	4	5
GO TO ROOM TWO ZERO FIVE	5	5	5	5
GO TO ROOM TWO ZERO FOUR	5	5	5	5
GO TO ROOM TWO ZERO THREE	5	5	5	5

TABLE II

ERROR RATES OF RECOGNIZING DIFFERENT VOICES

according to the commands. The experiment results are expressed in Figure 6.

Fig. 6. Experiment Results

Finally it is analysed that the trained sentences and the different voice recognition accuracy are almost 100%, which shows that the voice recognition system is robust.

VI. CONCLUSIONS

In this paper, a voice control system is implemented in a mobile robot. The voice system shows better results and recognizes the speech of a user and performs the action according to the user's commands. The system is proved to be efficient in a real-time scenario. This implemented system with GUI shows the potential for being used in a voice related application in automation like HMI (Human Machine Interface). Furthermore, the system is found to be effective in understanding different people's voice commands to execute the actions and navigate to its destination.

ACKNOWLEDGMENT

This work has been funded by the Frankfurt University of Applied Sciences, Germany. The authors would like to thank Mr. Umansky, Mr. Michalik and other colleagues of the Faculty of Computer Science and Engineering, Frankfurt University of Applied Sciences for their guidance and valuable support throughout the project.

REFERENCES

[1] Prof. Dr. Subhash P. Rasal (2014) Voice Controlled Robotic Vehicle, International Journal of New Trends in Electronics and Communication (IJNTECISSN: 2347 - 7334), vol. 2, no. 1, pp. 28-30, 2014.
[2] Peter X. Liu, A.D.C. Chan, R. Chen, K. Wang, Y. Zhu, Voice Based Robot Control,Proceedings of the 2005 IEEE International Conference on Information Acquisition June 27 - July 3, 2005, Hong Kong and Macau, China.
[3] Robotic operating Systems [Available online] http://wiki.ros.org/
[4] RTAB-Map: http://wiki.ros.org/rtabmap
[5] David Huggins Daines, An Architecture for Scalable, Universal Speech Recognition, PhD dissertation (chapter 3), Carnegie Mellon University, 2011.
[6] CMUSphinx, Basic concepts of speech recognition. [Online Available: https://cmusphinx.github.io/wiki/tutorial/]. [Accessed: January 2019]
[7] Navigation Stack: http://wiki.ros.org/move_base