



VAASAN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES

Ziye Zhou

THE NAO ROBOT AS A PERSONAL ASSISTANT

Information Technology

2016

FOREWORD

I would like to take this opportunity to express my gratitude to everyone who have helped me.

Dr. Yang Liu is my supervisor in this thesis, without his help I could not come so far and get to know about Artificial Intelligence, I would not understand the gaps between me and other intelligent students in the world. Thanks for giving me a chance to go aboard and get to know more.

Also I would like to say thank you to all the teachers and stuffs in VAMK. Thanks for your guidance. Thanks for your patient and unselfish dedication.

Finally, thanks to my parents and all my friends. Love you all the time.

VAASAN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES
Degree Program in Information Technology

ABSTRACT

Author	Ziye Zhou
Title	The NAO Robot as a Personal Assistant
Year	2016
Language	English
Pages	55
Name of Supervisor	Yang Liu

Voice recognition and Artificial Intelligence are popular research topics these days, with robots doing physical and complicated labour work instead of humans is the trend in the world in future. This thesis will combine voice recognition and web crawler, let NAO robot help humans check information (train tickets and weather) from websites by voice interaction with human beings as a voice assistance. The main research works are as follows:

1. Voice recognition, resolved by using Google speech recognition API.
2. Web crawler, resolved by using Selenium to simulate the operation of web pages.
3. The connection and use of NAO robot.

This project has achieved to use the NAO robot to search train and weather information from web pages according to the user's voice command, and to read the information out in order to interact with the user. When the robot is speaking, the LED lights which are around NAO's body will twinkle in a turn, this will increase the robot's friendly and interactive. The main contribution I did is that I achieved the whole project in the NAO robot, in a previous thesis the NAO robot was always tied together with the computer, the running and calculation of the main program are carried out on the computer instead of in the NAO robot. But in this paper, all codes are running in the NAO robot. This contribution further shows the development of artificial intelligence. In future, buying train tickets, booking hotels, making a travel plan are all possible to be done with NAO robot.

Python is the programming language which is used in the implementation part of this thesis project. The code testing are done under the Windows 7 operating system (Personal computer), python IDLE (python GUI) and NAOqi 2.1 operating system (NAO robot Vision 5).

Key words: NAO robot, Artificial Intelligence, Voice recognition, web crawler

CONTENTS

FOREWORD

ABSTRACT

LIST OF ABBREVIATIONS

LIST OF FIGURES AND TABLES

1	INTRODUCTION	10
1.1	Purpose	10
1.2	Overview Structure.....	10
1.3	Introduction to Nao Robot.....	10
1.3.1	History.....	10
1.3.2	Technology overview of NAO Robot.....	11
1.4	Introduction of Voice recognition	14
1.4.1	History.....	14
1.4.2	Principle of voice recognition technology	14
1.4.3	The future of voice recognition technology.....	14
1.5	Introduction of web crawler	14
1.5.1	History.....	15
1.5.2	Operational principle	15
1.5.3	The future of web crawler technology	16
2	OVERALL STRUCTURE.....	20
2.1	Structure of the Whole Project	20
2.2	Introduction to Modules	21
2.3	Flowchart of the Project	22
3	COMMUNICATION MODULE.....	23
3.1	NAOqi OS	23
3.2	Accessing NAO over SSH	24
3.2.1	NAOqi Python APIs	26
3.2.2	Configure the environment	27
4	SPEECH RECOGNITION MODULE	28
4.1	Speech Recognition APIs.....	28

4.2	Recording wav file by NAO.....	29
4.3	Correction of recognition result	31
5	WEB CRAWLER MODULE	39
5.1	Train of thought and preparation.....	39
5.2	Tools introduction	42
5.3	Implementation steps.....	43
6	IMPLEMENTATION DETAILS	46
6.1	Environment Configuration.....	46
6.1.1	OpenCV Configuration	46
6.2	Timeline.....	46
6.3	NAO Write Animation	48
6.3.1	Prepare Animation	48
6.3.2	Write Animation	49
7	RESULTS	51
8	REVIEW OF FUTURE RESEARCH	53
8.1	Improving NAO's software and hardware	53
8.2	Improving web crawler technology.....	53
8.3	Improving voice recognition technology	53
9	SUMMARY	53
	REFERENCES	54

LIST OF ABBREVIATIONS

PC	Personal Computer
IDLE	Integrated Development Environment
RoboCup	Robot Soccer World Cup
V5	Version 5

LIST OF FIGURES AND TABLES

Figure 1.	NAO robot	p.12
Figure 2.	NAO Robot joints	p.13
Figure 3.	NAO Robot V5 dimension	p.13
Figure 4.	NAO Robot V5 loudspeakers	p.14
Table 1.	NAO Robot V5 Microphones	p.15
Figure 5.	NAO Robot V5 Microphones	p.15
Table 2.	NAO Robot V5 Microphones location	p.15
Figure 6.	Historical development of speech recognition word error rate in increasing difficulty task	p.16
Figure 7.	Voice waveform example	p.16
Figure 8.	Frames	p.17
Figure 9.	Observation sequence	p.18
Figure 10.	Web crawler operational principles	p.20
Figure 11.	Internet division according to web crawler's perspectives	p.21
Figure 12.	Structure of the whole project	p.23
Figure 13.	Relationship between different modules	p.23
Figure 14.	Flowchart of the whole project	p.25
Figure 15.	NAOqi's framework structure	p.26
Figure 16.	NAOqi's framework structure 2	p.26
Figure 17.	User interface of PuTTY	p.28

Figure 18.	Communication between NAO robot and PC	p.28
Figure 19.	Command line user interface	p.29
Figure 20.	Relationships between different directories	p.29
Figure 21.	Files and folders under home directory	p.29
Figure 22.	Programmed files used in the project	p.30
Figure 23.	Basic commands	p.30
Figure 24.	Switch to root authority	p.30
Figure 25-30.	Sample code	p.31
Figure 31.	Link of phantomjs under /usr/local/bin	p.32
Figure 32.	Link of phantomjs under /usr/local/lib	p.32
Figure 33.	Files and folders under /usr/bin	p.32
Figure 34.	User interface of WinSCP	p.34
Figure 35.	Warning window	p.34
Figure 36.	Operation interface of WinSCP	p.35
Figure 37.	Relationship between user and speech recognition module	p.35
Figure 38.	A piece of code	p.37
Figure 39.	Input box of VR's main page	p.40
Figure 40.	Developer tools	p.40
Figure 41.	Main page in English	p.40
Figure 42.	The use of developer tools	p.42

Figure 43-44.	Network situation	p.42
Figure 45.	JourneySearch.do	p.43
Figure 46-51.	Analysis of the data which needs to be grabbed	p.44
Figure 52.	Selenium IDE icon on Firefox	p.47
Figure 53.	User interface of selenium IDE	p.48
Figure 54.	Exported file from selenium IDE	p.49
Figure 55-58.	Result capture	p.49
Figure 59.	NAO robot	p.51

1 INTRODUCTION

1.1 Purpose

The problem of current voice recognition equipment and software is, they are only the replacer of existing services but not the innovator, and cannot replace the mouse and keyboard. Voice recognition needs higher accuracy and continual exploration, learning and growth. So this thesis introduces the principle and composition of voice recognition, and applies voice recognition technology in practice by doing a web crawler to help users check train tickets and weather. And what is more, the whole procedures manifest the possibility of Artificial Intelligence.

1.2 Overview Structure

This thesis consists of 7 parts. Chapter one is an introduction to the necessary background knowledge, including an introduction to the NAO robot, voice recognition and web crawler, their current status of research and application. The second chapter gives an overview structure of the whole project, which presents the sequence, logical and environment configuration of the project. In chapter three, the technology of voice recognition is introduced and analysed, also it declares how to make use of Google speech recognition. The fourth chapter explains web crawler, which is used to get information from websites, includes the principle of web crawler and the operation steps of how to construct a web crawler. Chapter five describes the connection with NAO robot, how to build your own program on NAO and how to build the runtime environment. Chapter six briefly discusses the difficulties in completing this thesis project and the direction and expectations of the future pursuits. The seventh chapter is the summary and conclusion of the dissertation.

1.3 Introduction to Nao Robot

1.3.1 History

In 2005, Mr. Bruno Maisonnier set up the Aldebaran company and attracted five robot enthusiasts to the allies' positions, this is the formation of "Aldebaran workshop".

The meaning of the company's name - "Aldebaran", is a name of a body in the universe, which is the brightest star in the Taurus. The diameter of Aldebaran is 44 times that of the sun. In 300 BC, ancient Persia addresses Aldebaran, Antares, Regulus and Deneb together as four star kings, used for marine and land navigation. The name of the company conveyed the meaning that the company is the navigation star in the field of robotics.

One year after the company was founded, a small team which consisted of 12 people created the first NAO model. Although the time was not ripe, this small humanoid robot had attracted the interest of professional researchers.

In 2008, NAO was elected as the RoboCup standard platform, instead of Sony's robot dog AIBO in RoboCup Soccer League which is an international robot soccer game. The ultimate goal of this robot soccer game is the formation of a humanoid robot soccer team and compete with the champions in the human world cup in 2050.

NAO has gradually become the standard platform for university research and teaching purposes. In 2010, 20 NAO is presented a wonderful dance at the Shanghai Expo, France Pavilion and Paris District Museum stage. In December 2011, NAO Next Gen was published. In 2013, "Autism Solution for Kids"(a solution for autistic children, referred to as "ASK NAO") was launched, it provides a new tool to the special education teaching staff and to children with autism by using the robot technology.

In June 2014, Softbank mobile and Aldebaran jointly launched the world's first personal robot Pepper which can recognize emotions. In the meantime, Aldebaran launched the latest version of NAO—NAO Evolution. This robot is more powerful, and has a more comprehensive operating system./2/

1.3.2 Technical overview of NAO Robot V5

NAO is a unique product which is an ingenious combination of software and hardware, and consists a large number of sensors, motors and software. All software is controlled by a specially designed operating system to NAOqi.

NAO has:

- 25 motors, it can move freely.
- Two cameras, he can see things around; 1 inertial navigator, he can make sure if he is in an upright position, or falling position; a plurality of touch sensors, he can feel your touch; four directional microphone, he can hear your voice; two sets of receivers and transmitter which helps NAO to easily adapt to home automation applications.
- A variety of communication equipment, including voice synthesizer, LED lights and two high fidelity loudspeakers.
- An Intel ATOM 1.6 GHz processor (in the head), running the LINUX kernel, and supports Aldebaran's exclusive development of Middleware (NAOqi).
- A second CPU (on the trunk).
- A 48.6 KWh battery.

Following is a picture of NAO.



Figure 1. NAO robot /1/

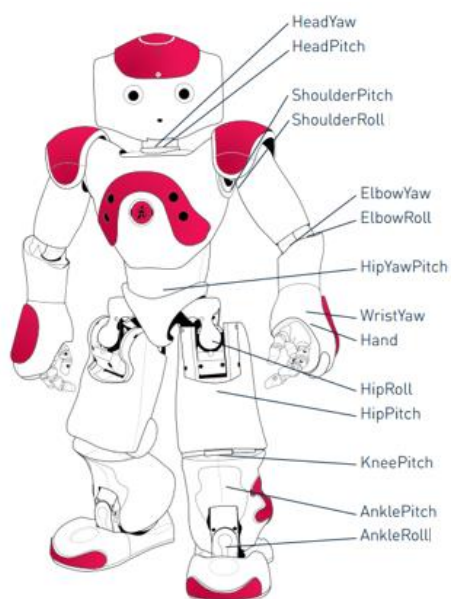
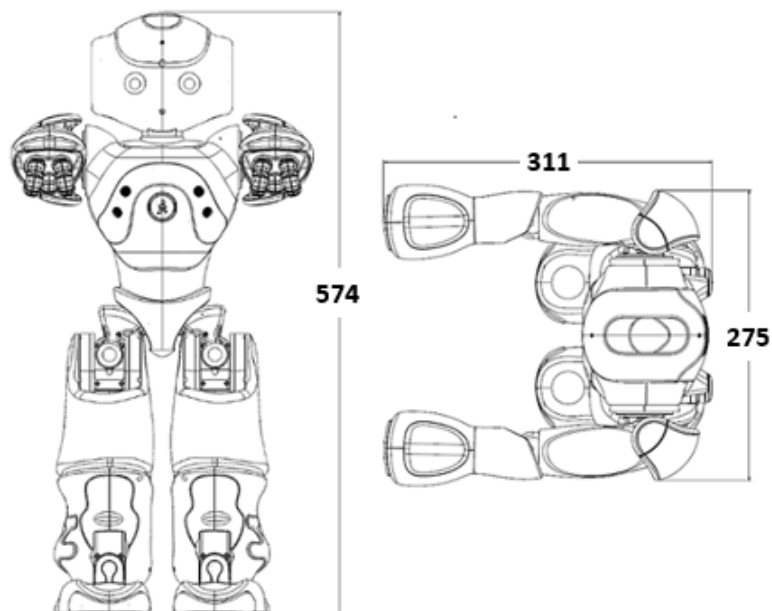


Figure 2. NAO robot /1/

Following are the specifications of NAO V5 related to this project.

- Dimensions

Height (mm)	Depth (mm)	Width (mm)
574	311	275



Weight

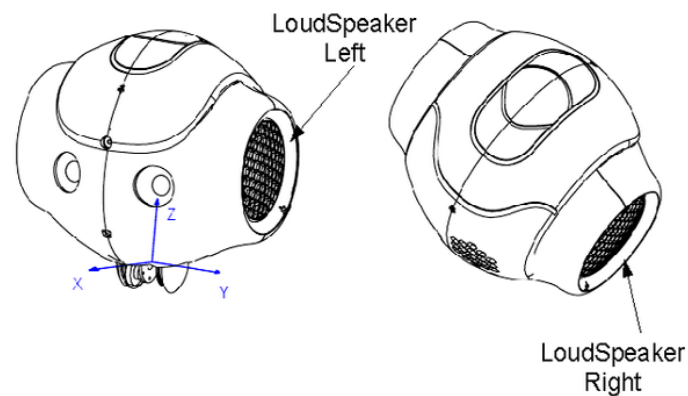
Weight: 5.4kg

Material

ABS-PC/PA-66/XCF-30

Figure 3. NAO Robot V5 dimension/2/

- Motherboard
 - ATOM Z530 1.6 GHz CPU
 - 1 GB RAM
 - 2 GB Flash memory
 - 8 GB Micro SDHC
- Loudspeakers
 - Specification: NAO has two loudspeakers which give him a stereo broadcast system.
 - Location:



	X(m)	Y(m)	Z(m)
Left	0.0038	0.0453	0.0526
Right	0.0038	-0.0453	0.0526

Figure 4. NAO Robot V5 loudspeakers/4/

- Microphones
 - Specification:

Microphones	x4 on the head
Sensitivity	20mV/Pa +/-3dB at 1KHz
Frequency range	150Hz to 12kHz

Table 1. NAO Robot V5 Microphones/6/

- Location:

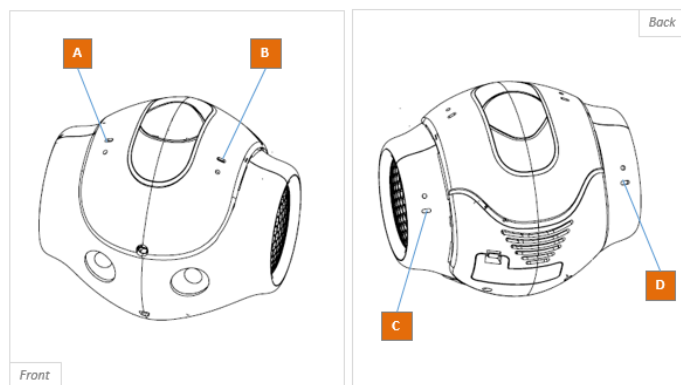


Figure 5. NAO Robot V5 Microphones/6/

Part	Location		Name	X (m)	Y (m)	Z (m)
A	Front	Right	MicroFR	0.0206	-0.0309	0.0986
B		Left	MicroFL	0.0206	0.0309	0.0986
C	Back	Left	MicroRL	-0.0215	0.0558	0.0774
D		Right	MicroRR	-0.0215	-0.0558	0.0774

Table 2. NAO Robot V5 Microphones location/6/

1.4 Introduction to Voice recognition

1.4.1 History

Speech recognition has been a common scenario in science fiction. But in 1976, its actual level was widely divergent with those far fetched functions in the fictional world.

In 1995, it was the first time for Windows95 to equip with Microsoft SAPI, and it enabled developers to create a voice program in Windows. In 1999, the forum that supported IVR phone and VoiceXML was established. In 2001, Bill Gates showed a prototype named MiPad in the U.S. Consumer Electronics Show (CES), 16Mipad showed the vision of voice multimodal mobile device. As Apple, Google and Microsoft have been using voice recognition technology in their products recently, we are witnessing the improvement of equipment handling capacity.

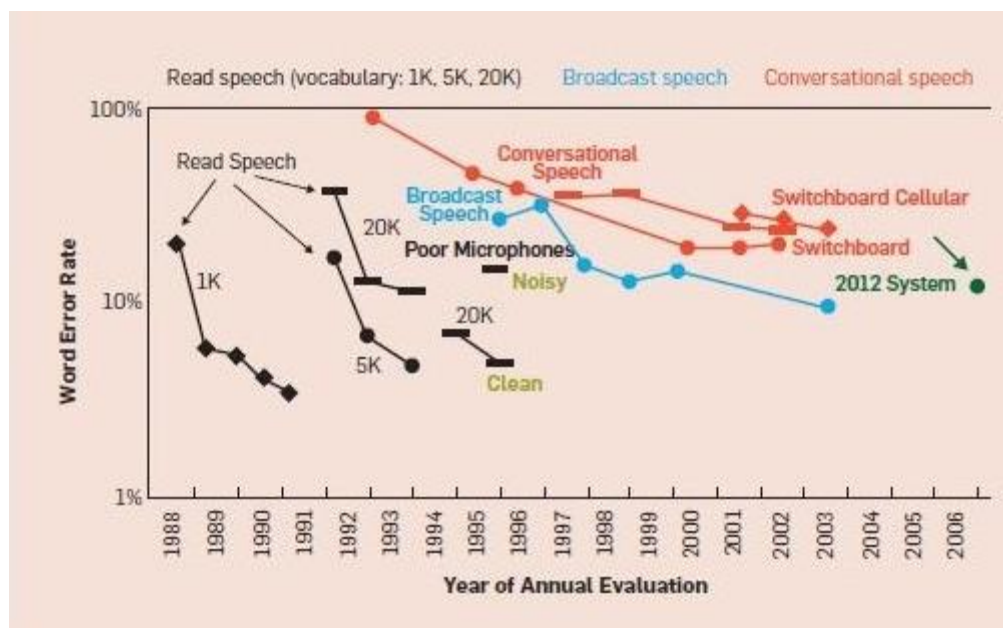


Figure 6. Historical development of speech recognition word error rate in increasing difficulty task/7/

1.4.2 Principle of voice recognition technology

First of all, sound is actually a kind of wave. The common formats, MP3, WMV are all compressed formats. In order to deal with these kind of formats, they must be converted into the unpacked pure waveform files, such as Windows PCM file, which means wav file. In wav file, it stores the points of waveform and a file header. The higher the sampling rate, the greater points contained in every millisecond voice. In addition, voice has single channels, double channel, four channels etc. For speech recognition tasks, a single channel is enough. Thus generally, sound needs to be turned into a single channel before processing.

Below is an example of a voice waveform.



Figure 7. Voice waveform example /8/

Before the start of voice recognition, mute that from end to end need to be cut off (this

operation usually called VAD, it needs some signal processing technology) in order to reduce the interference of the following steps.

After mute resection, the voice waveform is divided into frames. Frames are overlapping each other, just like the image show below:

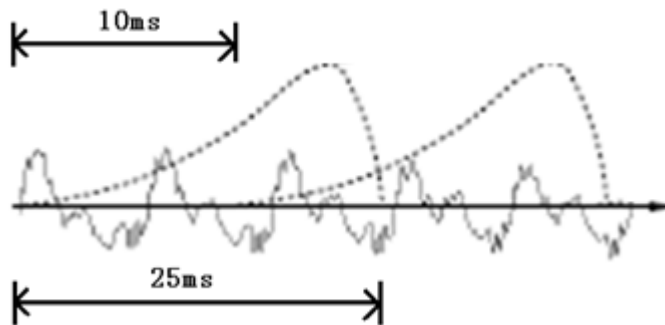


Figure 8. Frames /8/

In figure 6, the length is 25 milliseconds per frame. Between every two frames there is an overlapping which is $25-10=15$ MS. Thus it is said that this frame is divided as the frame length is 25ms, frame shift is 10ms.

However, the waveform almost has no ability to describe in the time domain, it needs to be converted. The most common method is to extract MFCC features, and convert each frame into a 12 dimensional vector. These 12 points are extracted according to the physiological characteristics of the human ear, which means that these 12 points included the frame's content information.

So far, the voice becomes a 12 line, N columns matrix where N is the total number of frames. The observation sequence is as shown, in the figure below, each frame is represented by a 12 dimensional vector, the colour lump's shade indicates the value of the vectors.

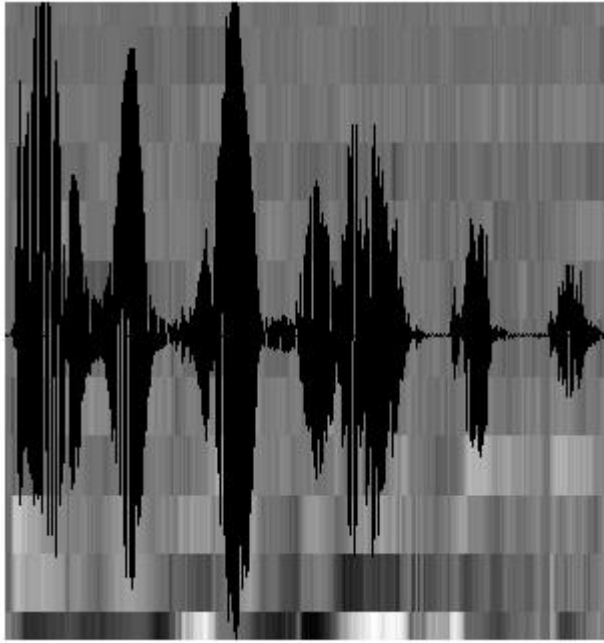


Figure 9. Observation sequence/8/

According to the value of vectors, voice recognition needs to recognize frame to state, combine states into phonemes and combine phonemes to words.

1.4.3 The future of voice recognition technology

The development of speech recognition technology is promoting the transparency of computing, making the interaction between people and computers more easy. Since the beginning of the new century, the world of IT is experiencing a great shift , which is moving the development centre of gravity to the technology of human and computer interaction.

Currently, the market size of global voice recognition technology oversized 30 billion U.S. dollars, and in recent years, the growth rate remained at more than 25%. The future of speech recognition's market is promising, such as telecommunication industry (VoIP), mobile application area(cell phone, learning machine, panel computer and vehicle-mounted system, etc.) all will show exploring growth. Following are some detailed examples:

1. Telecommunication area: telephone banking system.

Telephone banking system is a new and high technology field which was been rising in recent years. It is the basis of realizing the modern management and management of the

bank. It connects customers with the bank through phone which is a modern communication tool so that user do not need go to bank, and the user can always get the services provided by bank and by just a telephone call to the bank. The bank can promote its service quality, and improve customer numbers.

2. Medical field

Due to the strong evolution of the vocabulary database in the medical field, as long as a complete database is established, the name of medicines and diseases can be easily recognized. Doctors can benefit from this invention by using voice search engine to find the past case of the illness in an easier way.

3. Intelligent vehicle

The issues of traffic safety have been given a lot of attention. By using voice recognition technology, a user can use speech to control GPS navigation, information sending and receiving, telephone answering, social network update, etc.

4. Intelligence wearable, smart home, education field, etc.

Speaking is the most natural way for humans to communicate with each other. Artificial intelligence might be pseudo intelligence. The machine will never live like a man, but the machine can understand people more and more.

1.5 Introduction of Web crawler

1.5.1 History

Web crawler is a program or script which captures web information automatically according to certain rules. In the Internet is preliminary development stage, the websites were less, it was easier to find information. However, with the explosive development of the Internet, it is like finding a needle in the ocean for ordinary Internet users to find the required information in the Internet. At this time, the professional search site was born to meet the public demand for information retrieval.

In 1990, Archie was invented by Alan Emtage who is a student of the University of Montreal. Archie is the ancestor of the modern search engine. Inspired by Archie, Nevada System Computing Services University developed another very similar search tool in 1993, in addition to the index file search tool, this time the search tool had been able to

retrieve a web page. At that time, the word “Robot” is a very popular word, Computer Robot refers to software programs that a man is not able to reach the speed of work uninterrupted execution. Due to the fact that the special program for information retrieval seems like spiders crawling across the network, this kind of program is called spider program or web crawler.

By using web crawler people can get the information they need in an easier and more sufficient way.

1.5.2 Operational principle

The web crawler is an important part of the search engine, it is a program which can automatically extract web pages. The main purpose of the web crawler is to download web pages from the internet. Traditional crawler starts with one or several original web pages' URL. It gets the initial web page URL, extracts new URLs into the queue from the current page unceasingly in the process of grabbing web pages until a certain stop condition is satisfied. The general framework of a web crawler is shown as follows:

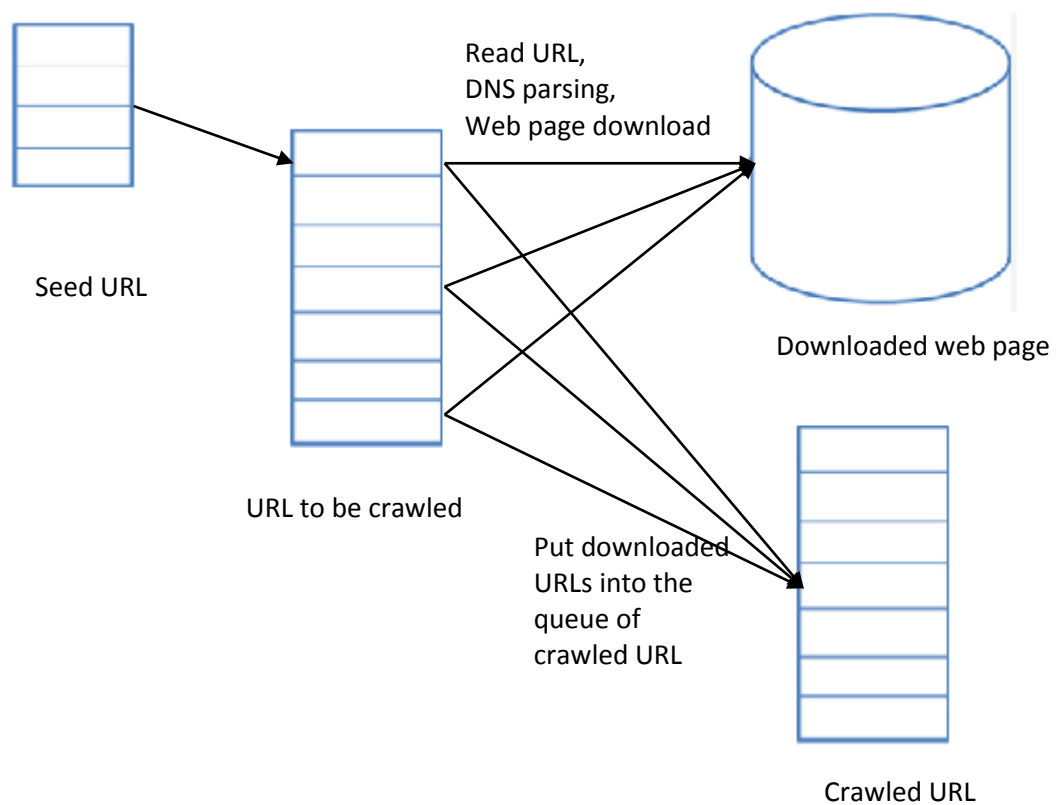


Figure 10. Web crawler operational principles

The basic workflow of web crawler is as following:

1. Carefully select the seed URLs;
2. Put these seed URLs into the URL queue which is going to be crawled.
3. Pop the URLs from the to-be-crawled queue, analysis the DNS and get the host IP, download the corresponding web pages and stored them in the downloaded web page library. In addition, move these URLs to crawled URLs queue.
4. Analysis the URLs in crawled URL queue, and put other URLs into the to-be-crawled URLs queue. Thereby entering the next loop.

If divided the Internet according to web crawler's perspectives,

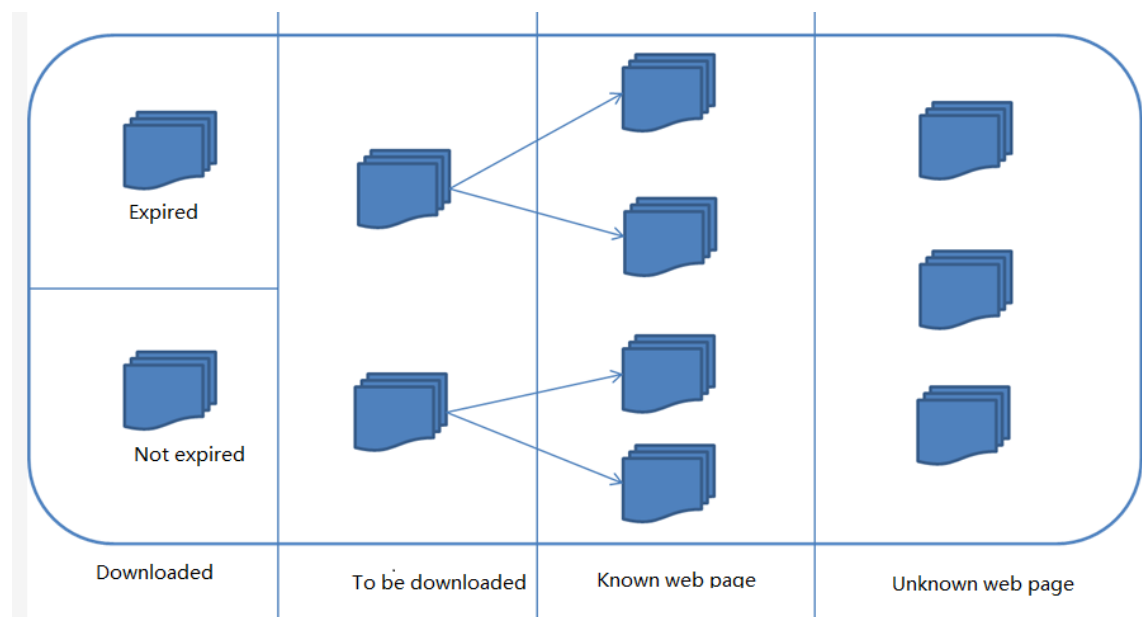


Figure 11. Internet division according to web crawler's perspectives /14/

1.5.3 The future of web crawler technology

The rapid development of the network throws a question to the Internet which carries vast amounts of information— Can we extract the needed information accurately and rapidly? The traditional search engines have Yahoo, Google, Baidu, etc., these search information tools are the only way which must be passed when people visit Internet. But the traditional search engines also have limitations:

- 1) Users with different backgrounds and in different fields have various search purposes and requirements, it cannot be comprehensive and accurate to find the required information, some irrelevant content will also be searched. The information is used less efficiently.

- 2) The goal of universal search engines is to promote the network coverage rate as high as possible. The contradiction between limited search engine server resources and unlimited network data resources will be further deepened.
- 3) Because of the rich data form of the world wide web and the continuous development of the network technology, image, database, audio and video multimedia data, and other various data appear in large numbers, general search engines are often helpless on these data which has intensive information content and a certain structure. These data cannot be found and acquired well.
- 4) Most general search engines provide keyword based retrieval, it is difficult to support query based on semantic information.

So, to improve the speed and quality of information retrieval is the main research content of professional search engine. In order to solve the above problems, the focused crawler which orientation crawl related web resources is bored at the right moment. A focused crawler is a program that can automatically download web pages. It is based on the established grab target to access the World Wide Web page and related links, and get the needed information. Different to the general purpose web crawler, the focused crawler do not pursuit large coverage, its goal is to grab the specific topic which is related to the content of the page and preparing data resources for the subject - oriented user query.

2 OVERALL STRUCTURE

This chapter explains the structure of the whole project, gives brief illustrations for each section and there is a flow chart in the end as a summary.

2.1 Structure of the Whole Project

This project can be divided into three parts, the first part is voice recognition module, the other one is web crawler module, and the last one is communication module between User, computer and NAO robot.

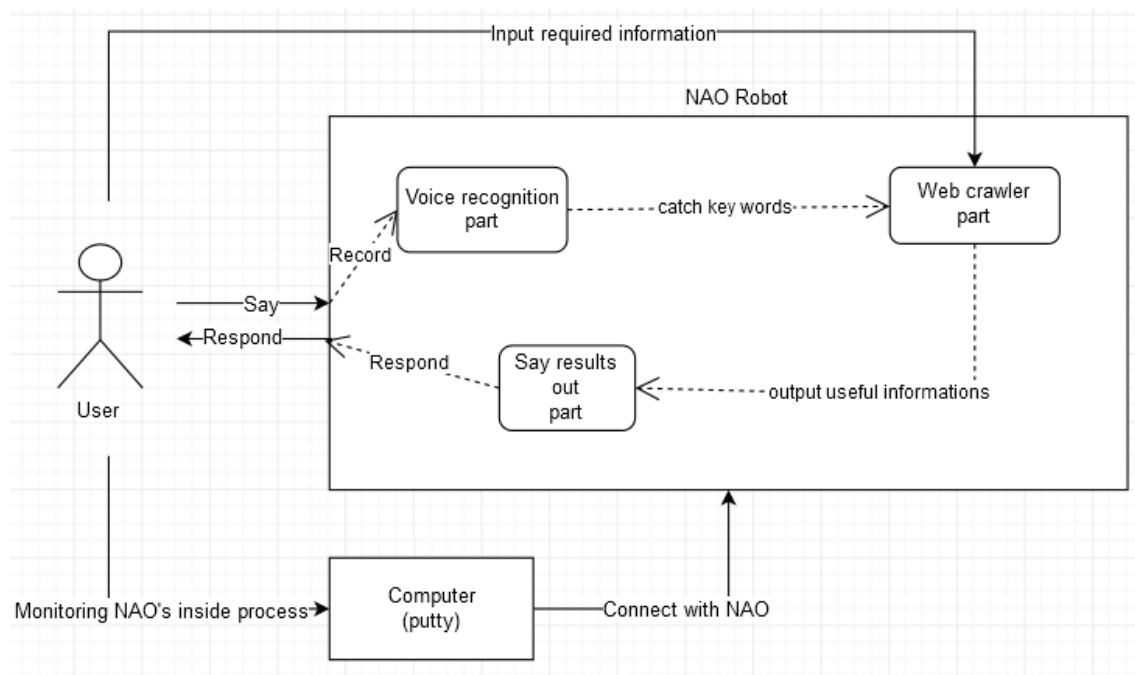


Figure 12. Structure of the whole project

A user can monitor the process which is running inside the NAO robot by a computer, this step is designed for a better debug, otherwise the user can hardly know the reason if there has any problem was occurred.

The voice recognition module and web crawler module belongs to the NAO robot, while the communication module is between user, computer and NAO.

2.2 Description of modules



Figure 13. Relationship between different modules

- Voice recognition module

Voice recognition module is divided into recorder part and recognition part. Recorder part contains microphones in NAO as hardware support, ALProxy module “ALAudioRecorder” is needed for recording the voice in NAO from the user. “ALTextToSpeech” module can be invoked as a friendly reminder to the user. The output of this part is a recorded .war file which contains the user’s speech. Recognition part imported “speech_recognition” package which can be used to retrieve recorded .war file, adjust for ambient noise and do the energy threshold. After processing, Google speech recognition is used for identification of the characters, words or sentences in the .war sound recording file. The recognized result will be repeated so that user can make sure whether the voice recognition module gets the right meaning. The user needs to say “Yes” or “No” to declare his attitude. If the user’s answer is “Yes”, the program will try to match the user’s command and the key words of the web crawler module. If a match is found, the program will enter the web crawler module, otherwise, the user will be asked to do all steps again from the beginning; if the user’s answer is “No”, the program will try to catch the user’s command again until the user’s answer is “Yes”.

- Web crawler module

The web crawler module needs to import three packages: selenium, BeautifulSoup and time. Selenium is used to automate the browser, BeautifulSoup is a python library which can be used to parse the document and grasp data for the user in an easy way. Time module provides a variety of operating time functions, in this project, time module is mainly used to postpone thread to the specified time so that the browser has enough waiting time to load data, or blank pages maybe grasped. The combined use of these packages make the web crawler implemented in a easy and intuitive way.

- Communication module

In this thesis project, all of the code is run on the NAO robot. Thus python and its related packages, environment needs to be configured inside NAO. In order to monitor the process inside NAO, the user can use putty on the personal computer to connect with NAO. Putty is an SSH and telnet client, NAO robot can be regarded as

a computer which operating system is Linux, actually, what putty does is to remotely access to a Linux computer(NAO robot) on a personal computer.

2.3 Flowchart of the Project

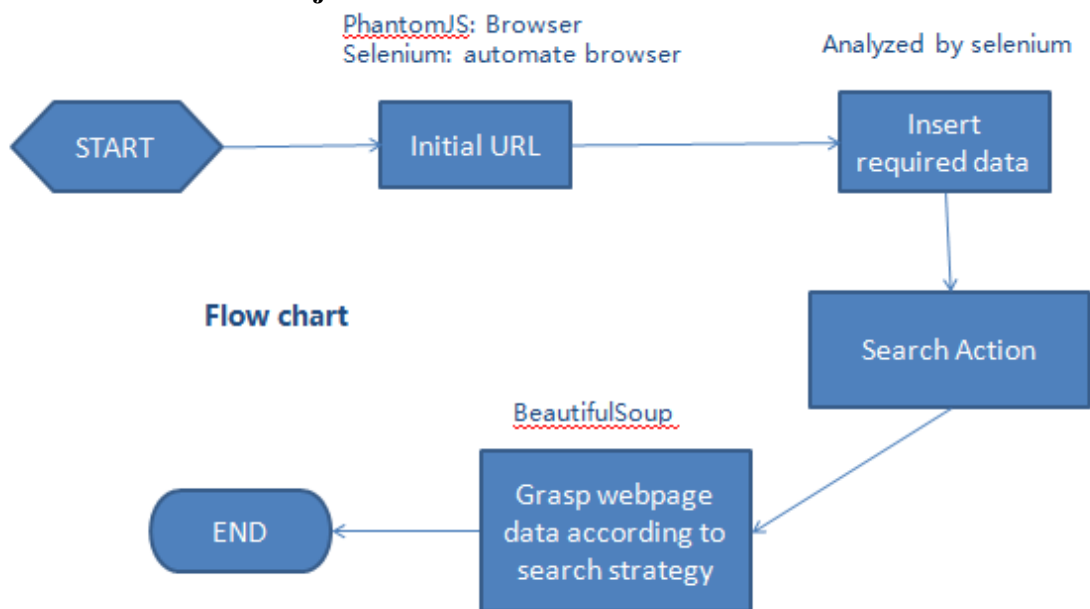


Figure 14. Flowchart of the whole project

Figure 10 is the flowchart of the whole project. First of all the program will inform the user that he can say something now, after the remind message, there will be a few seconds for the user to speak something. During this period, the microphones are recording the voice which is from NAO's microphone. If there is nothing in the recording or user's input speech is too short, the program will return a message that it has not caught it and the user needs to say his command again. After successfully saving the recording file, the program will pass the recording file to Google speech recognition after noise reduction. Then the program will return the recognized word to user and ask user if the word is correct. At that time, the program will invoke the same speech recognition steps as former steps, user has two choices, answer "Yes" or "No". This step is called speech recognition module. If the user's answer is yes, the program check if the user's command matches any key words in web crawler module.

3 COMMUNICATION MODULE

This part describes the communication between the NAO humanoid robot and the personal computer, and the environment configuration.

There are two methods for a personal computer to connect to NAO, one is by plugging an Ethernet cable, and the other one is WIFI connection. The Ethernet cable connection can fix the IP address while the WIFI connection may provide a dynamic IP address. In order to let NAO be better placed, this project chose the WIFI connection. By entering NAO's internet address in the address bar of the web browser, the user can manage the information of NAO in this web page, and the most important thing is that user is able to check NAO's hardware temperature here, if the hardware temperature is higher than 70°C which means NAO need rest now.

3.1 NAOqi OS

NAOqi is the main software which runs on the robot and controls the robot under OpenNAO distribution. NAOqi can also run on a computer so that the code can be tested on a simulated robot.

NAOqi OS is the operating system of NAO robot. It is a GNU/Linux distribution which is based on Gentoo. NAOqi is an embedded Linux distribution which is specially developed to meet the needs of Aldebaran robot development/9/

Following is the release information of NAOqi OS. (lsb_release -a is a linux command which used to prints certain Linux Srandard Base information)

```
VankNaoBeta [0] ~/files_test $ lsb_release -a
LSB Version:      n/a
Distributor ID:   OpenNao
Description:      OpenNao 2.1
Release:          2.1.0.19
Codename:         Murol
```

Figure 15. NAOqi's framework structure

```
VankNaoBeta [0] ~/files_test $ cat /proc/version
Linux version 2.6.33.9-rt31-aldebaran-rt (portage@bnoob-2) (gcc version 4.
5.3 (Gentoo 4.5.3-r1 p1.0, pie-0.4.5) ) #1 SMP PREEMPT RT Mon Jun 23 01:5
2:12 CEST 2014
```

Figure 16. NAOqi's framework structure 2

From above information we can find that OpenNAO is a GNU/Linux distribution which is based on Gentoo and it is specifically developed for NAO robot. Programs and libraries are provided by OpenNAO and NAOqi is the software which gives the robot life.

NAOqi framework works by having Choregraphe, Monitor, Motion module, and Audio module pass information to each other. NAOqi is executed by having Broker deliver information and commands. The following explains the different elements that configure the NAOqi framework.

- **Module:** Module is both a class and library that uses the function and API defined in ALModule to obtain information or control regarding each module.
- **Communication:** Communication uses Local Procedure Call or Remote Procedure Call to connect to NAO and exchange information.
- **ALMemory:** ALMemory is the robot's memory. Any module can use or read this data and can monitor events. It can be called when an event occurs. ALMemory is an array of ALValue.
- **Proxy:** All Aldebaran module has been modularized. Rather than directly referencing other module files, the user can request the Proxy to find the corresponding module. If the module does not exist, an exception occurs. The user can tell the corresponding function or module through the Proxy from two independent brokers, mainBroker (local call) and myBroker (remote call).
- **ALValue:** In order to be compatible, some NAOqi modules or methods are saved as a specific data type in ALValue.

3.2 Accessing NAO over SSH

Using PuTTY:

PuTTY is a free and client side of telnet, rlogin and SSH under windows 32 platform. But the function does not inferior to the commercial tools such as telnet. It is very easy to use

PuTTY to manage a remote Linux operating system. Hence this project used PuTTY to configure the required environment and monitor the program running process on NAO. The following picture is the control panel of PuTTY.

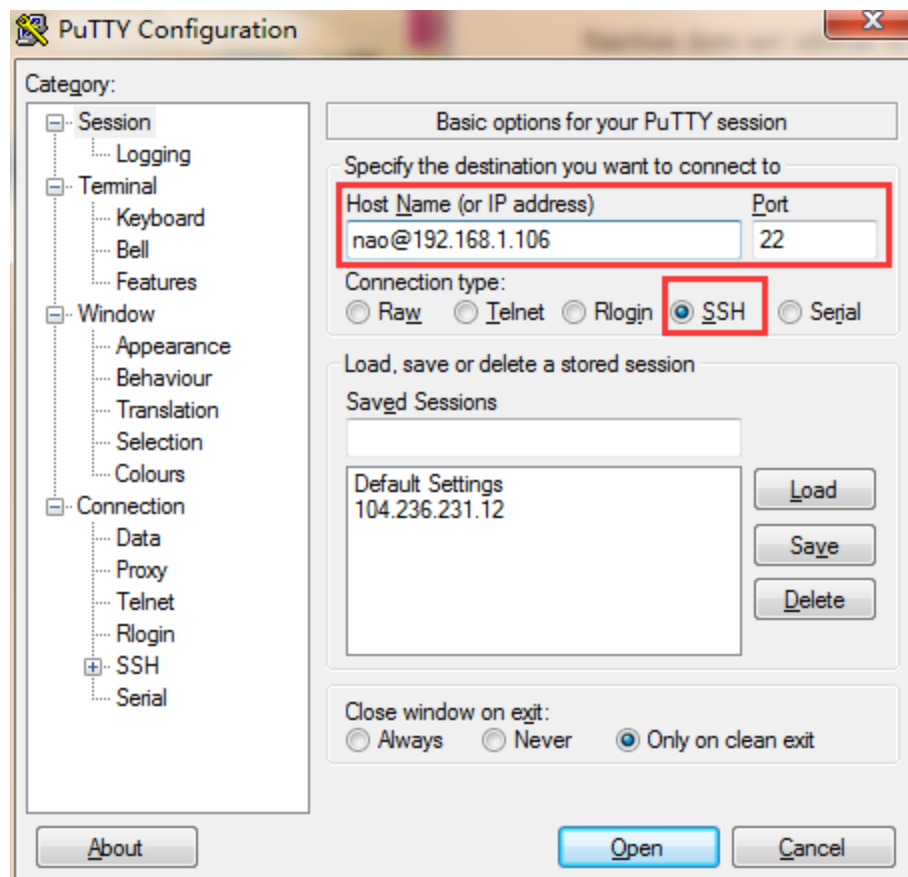
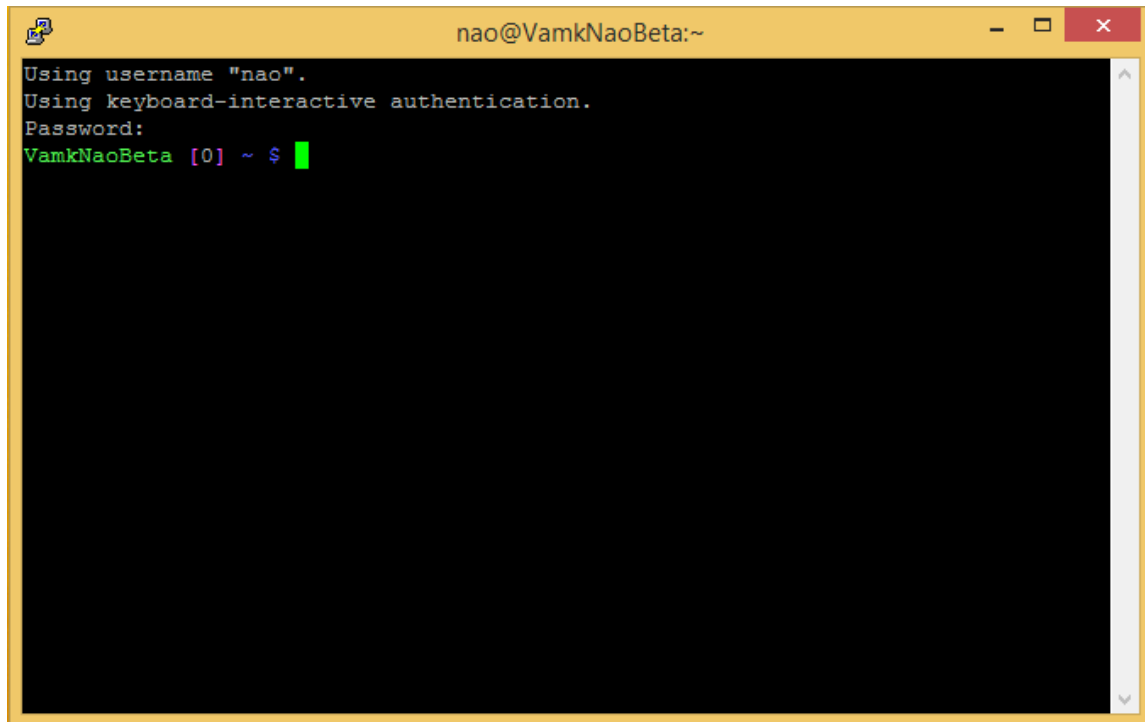


Figure 17. The user interface of PuTTY

Firstly, the user can get NAO's IP address by pressing NAO's chest button once. Then the user can input NAO's IP address into the box in format: [nao@192.xxx.x.xxx](#). "nao" is as the login user name./9/

User	Password	Description
nao	nao	default account
root	root	administrator account

Figure 18. Communication between NAO robot and PC



```

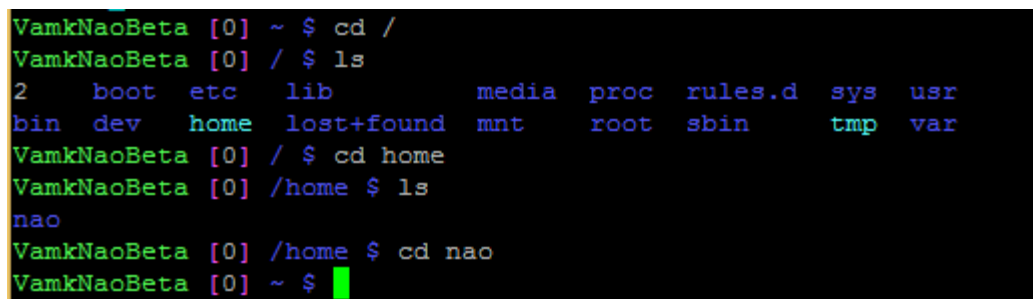
nao@VamkNaoBeta:~
Using username "nao".
Using keyboard-interactive authentication.
Password:
VamkNaoBeta [0] ~ $

```

Figure 19. Command line user interface

After login as “nao”, the interface is shown as above.

For NAO, the directory that the user logged in it is not his home directory. Under the real home directory, the user can do some modification for system configuration.



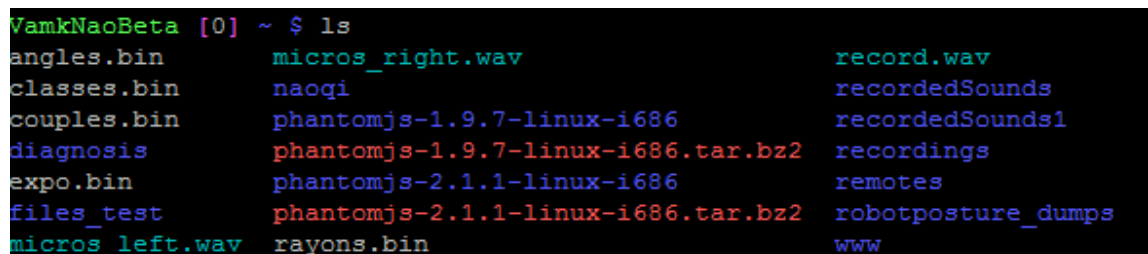
```

VamkNaoBeta [0] ~ $ cd /
VamkNaoBeta [0] / $ ls
2  boot  etc  lib          media  proc  rules.d  sys  usr
bin dev  home lost+found  mnt    root  sbin    tmp  var
VamkNaoBeta [0] / $ cd home
VamkNaoBeta [0] /home $ ls
nao
VamkNaoBeta [0] /home $ cd nao
VamkNaoBeta [0] ~ $

```

Figure 20. Relationships between different directories

Following are the files and folders that an under /home/nao directory.



```

VamkNaoBeta [0] ~ $ ls
angles.bin          micros_right.wav          record.wav
classes.bin         naoqi                     recordedSounds
couples.bin         phantomjs-1.9.7-linux-i686 recordedSounds1
diagnosis           phantomjs-1.9.7-linux-i686.tar.bz2 recordings
expo.bin           phantomjs-2.1.1-linux-i686 remotes
files_test         phantomjs-2.1.1-linux-i686.tar.bz2 robotposture_dumps
micros_left.wav    rayons.bin                www

```

Figure 21. Files and folders under home directory

“files_test” is the folder which I built to store needed python files.

```
VamkNaoBeta [0] ~ $ cd files_test
VamkNaoBeta [0] ~/files_test $ ls
NAO_Record.py          ghostdriver.log      speechRec.py         vr.py
NAO_Record.pyc         main.py              speechRec.pyc        vr.pyc
NAO_Record_correct.py  speechCorrect.py     test.py              weather.py
NAO_Record_correct.pyc speechCorrect.pyc    try.py               weather.pyc
VamkNaoBeta [0] ~/files_test $
```

Figure 22. Programmed files used in the project

The basic commands and programs which are available on NAOqi OS is as following:

Program	Description
htop	monitor process activity (many options are available use F1)
ldd	list library dependencies
gdbserver	start a remote gdb server

Figure 23. Basic commands

Sudo can be used on NAOqi OS, but its usage is only valid on shutting down the robot. su can be used to switch current user to root user.

```
VamkNaoBeta [0] ~ $ su root
Password:
root@VamkNaoBeta [0] nao #
```

Figure 24. Switch to root authority

3.2.1 NAOqi Python APIs

NAOqi APIs for python mainly have the following modules: ALProxy, ALBroker, ALModule, almath. Some of them are used in choregraphe internally, while some of them are generated automatically. This project used ALProxy.

A user can create a proxy for a module by the ALProxy object. There are two constructors to initialize it which depends on whether a Broker instance is available. The format of the first constructor is: ALProxy(name), “name” is the name of module. The second constructor’s format is: ALProxy(name, ip, port), while “name” is the module’s name; “ip” is the broker’s IP in which the module is running; “port” is the broker’s port.

According to the rules, when writing code outside choregraphe, developer should use the second format constructor./10/

Following is a piece of example code:

```

from naoqi import ALProxy
tts = ALProxy("ALTextToSpeech", "<IP of your robot>", 9559)
tts.say("Hello, world!")

```

Figure 25. Sample code

By invoking ALProxy, developers can use the microphones and loudspeakers in an easy way.

3.2.2 Configure the environment

- Install PhantomJS

1. Clear the old version's configuration

```

$ sudo unlink /usr/local/bin/phantomjs
$ sudo unlink /usr/local/share/phantomjs
$ sudo unlink /usr/bin/phantomjs

```

Figure 26. Sample code

2. Make a new folder and download the files inside the folder.

```

$ sudo wget https://bitbucket.org/ariya/phantomjs/downloads/phantomjs-1.9.7-linux-x86_64.tar.bz2

```

Figure 27. Sample code

3. Extract the files.

```

$ tar xjf phantomjs-1.9.7-linux-x86_64.tar.bz2

```

Figure 28. Sample code

4. Built links for phantom files

```

$ sudo ln -s /usr/local/share/phantomjs-1.9.7-linux-x86_64/bin/phantomjs
/usr/local/share/phantomjs;

$ sudo ln -s /usr/local/share/phantomjs-1.9.7-linux-x86_64/bin/phantomjs
/usr/local/bin/phantomjs;

sudo ln -s /usr/local/share/phantomjs-1.9.7-linux-x86_64/bin/phantomjs
/usr/bin/phantomjs

```

Figure 29. Sample code

5. To check if the configuration has completed

```

$ phantomjs --v

```

Figure 30. Sample code

It must appear: 1.9.7 if the configuration is down correctly./15/

The following are two pictures which shows the links of phantom files under /usr/local/bin and /usr/local/bin directory.

```
VamkNaoBeta [0] / $ ls
2   boot  etc    lib          media  proc  rules.d  sys  usr
bin dev  home  lost+found  mnt    root  sbin    tmp  var
VamkNaoBeta [0] / $ cd usr
VamkNaoBeta [0] /usr $ ls
bin  include  lib  libexec  local  sbin  share
VamkNaoBeta [0] /usr $ cd local
VamkNaoBeta [0] /usr/local $ ls
bin  lib  share
VamkNaoBeta [0] /usr/local $ cd bin
VamkNaoBeta [0] /usr/local/bin $ ls
firefox  phantomjs
```

Figure 31. Link of phantomjs under /usr/local/bin

```
VamkNaoBeta [0] /usr/local/lib $ ls
phantomjs
```

Figure 32. Link of phantomjs under /usr/local/lib

Following figure shows all the files and folders under /usr/bin.

```
VamkNaoBeta [0] /usr/bin $ ls
2to3          clear          funzip          lesspipe        od              rfcmm           tiffcp
2to3-2.7     cmp            g-ir-annotation-tool  lesspipe.sh     oldfind        rgb2ycbcr       tiffrop
{             code2color    g-ir-compiler    libfdt1-config  omshell        rlfe            tiffdither
acconnect    col           g-ir-generate    libgcrpt-config openssl         rpgen           tiffdump
alfand       colcrt        g-ir-scanner     libpng-config   orc-bugreport  rsync           tiffinfo
alsa-info    colorhead     gatttool        libpng15-config orcc            runcon          tiffmedian
alsaloop     colm          gawk            libtool         pacat          say             tiffset
alsamixer    colum         gdb             libtoolize      pacmd          sbdec           tiffsplit
alsasum      comm          gdbreplay       libusb-config   pactl          sbcenc          time
amidi        compare_octrees  gdbserver       line            padsp          sbcinfo         timeout
amixer       compile_et    gdbtui          lingdbcompiler  pal2rgb        scp             tload
aplay        connectionmanager  gdbus          link            pamon          script          toe
aplaymidi    connman       gencat          linux32         paplay         scriptreplay    top
arch         convert_octree  getconf         linux64         parec          sdiff           touch
arecord      crash-report-upload  getent         lrzcrd          parecord       sdptool        tput
arecordmidi  csplit        getopt          list-bluetooth-devices  passwd         sendmail        tr
aseqdump     curl          gettext         locale          paste          pasuspender    seq            truncate
aseqnet      curl-config   gettext.sh      localdef        patchk         patschk         tset
asever       cut           gettextize     log2graph       pcpfiledump   pcre-config     setargs       tty
asn1Coding   cytune        g10-querymodules  logname         pcre-config    pcregrep        setuid        tzselect
asn1Decoding db4.8.archive  glib-compile-schemas  look            pcretest       setterm        uic
asn1Parser   db4.8.checkpoint  glib-genmarshal  lrelease        peekfd         sftp           ul
audiorecorder-test  db4.8.deadlock  glib-gettextize  lab_release     pg             sha1sum         unexpand
autopoint    db4.8.dump     glib-mkenums    laspu           pgawk          sha224sum       uniq
avahi-browse  db4.8_hotbackup  gmsgfmt         laof            pgrep          sha256sum       unlink
avahi-browse-domains  db4.8_load      gnutls-cli       lscpu           phantomjs      sha384sum       unzip
avahi-publish  db4.8_printlog  gnutls-cli-debug  lsusb           pilconvert.py  sha512sum       unshare
avahi-publish-address  db4.8_recover  gnutls-serv      lctng           pilconvert.py-2.7  shred          unxz
avahi-resolve  db4.8_sql       gobject-query    lctng-gen-tp    pildriver.py   shuf            unzip
avahi-resolve-address  db4.8_stat      openssl          lctng-relayd   shuf            unzip
```

Figure 33. Files and folders under /usr/bin

- Install Selenium

Selenium is an automated testing tool for the web. In comparison with other automated tools, its most important feature is cross platform and cross browser, support windows, Linux, MAC, and support i.e., safari, opera, chrome etc. In addition, there is another important feature that it supports the execution of distributed test cases. The test cases can be distributed to different test machines, this function is equivalent to the distributor's

function. There are two ways to develop with selenium: One is to go to the selenium's website to download the selenium engine in python version; another is to build the robot automation framework, then install the selenium robot plug-in. In this project I used the first way.

Before installing selenium, “pip” need to be installed in advance.

Pip is similar to yum which is inside RedHat. In Ubuntu, it is very convenient to install software with pip. However, the Linux operating system inside the NAO robot do not support apt-get, thus pip can only be installed by “wget”.

1. The download of pip

```
# wget "https://pypi.python.org/packages/source/p/pip/pip-1.5.4.tar.gz#md5=834b2904f92d46aaa333267fb1c922bb"
```

2. Installation of pip

```
# tar -xzf pip-1.5.4.tar.gz
# cd pip-1.5.4
# python setup.py install
```

After the installation of pip, user can directly use pip to install selenium, the command is:

```
#pip install -U selenium.
```

● WinSCP

WinSCP is an open source graphical SFTP client side which used in the environment of Windows. At the same time, it also supports SCP protocol. Its main function is to copy files between local and remote computers safely, it can also be used to edit files directly. By using this software, the programmer can easily transfer files between your personal computer and the NAO robot.

Following is the main graphic interface of WinSCP. Input the robot IP address, username and password, then click “login”.

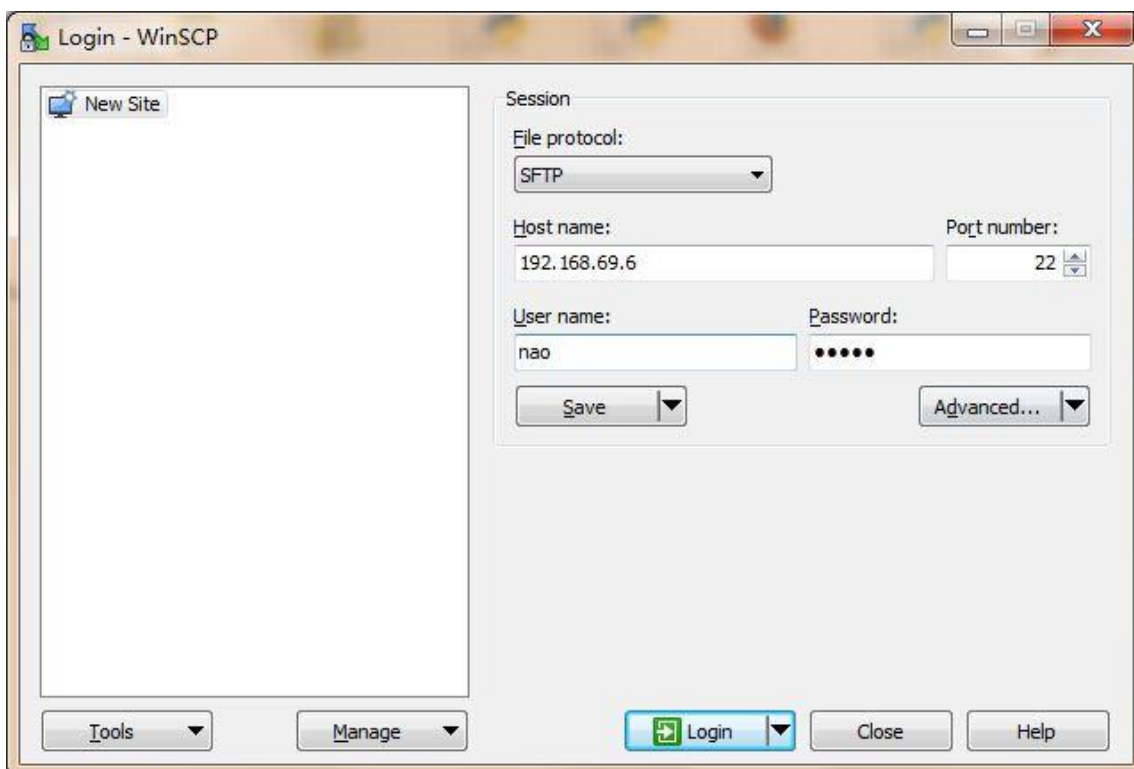


Figure 34. User interface of WinSCP

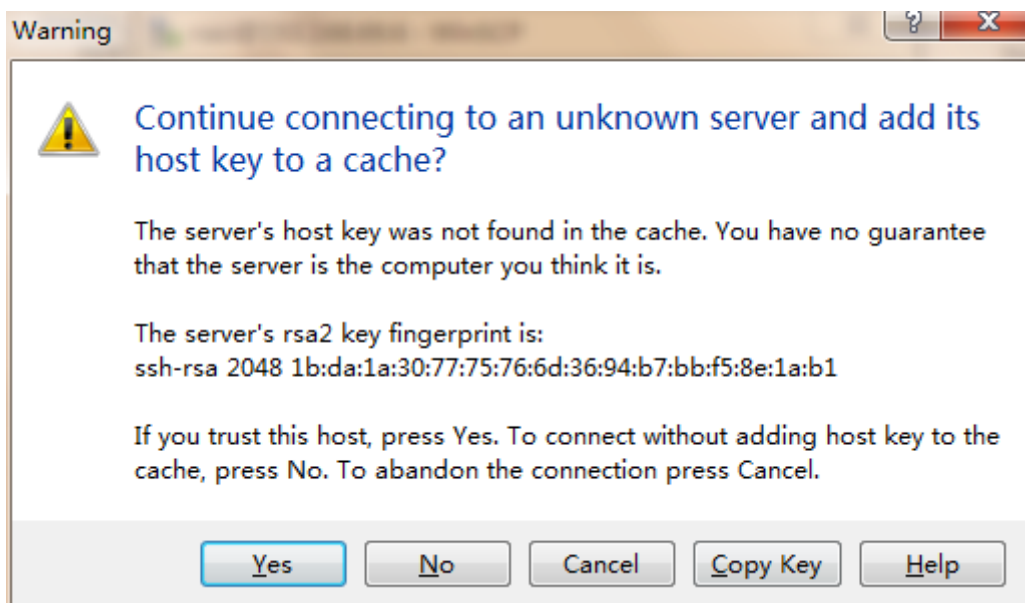


Figure 35. Warning window

By clicking “yes” the user can see the user interface as following. The user can easily transfer files between these two platforms.

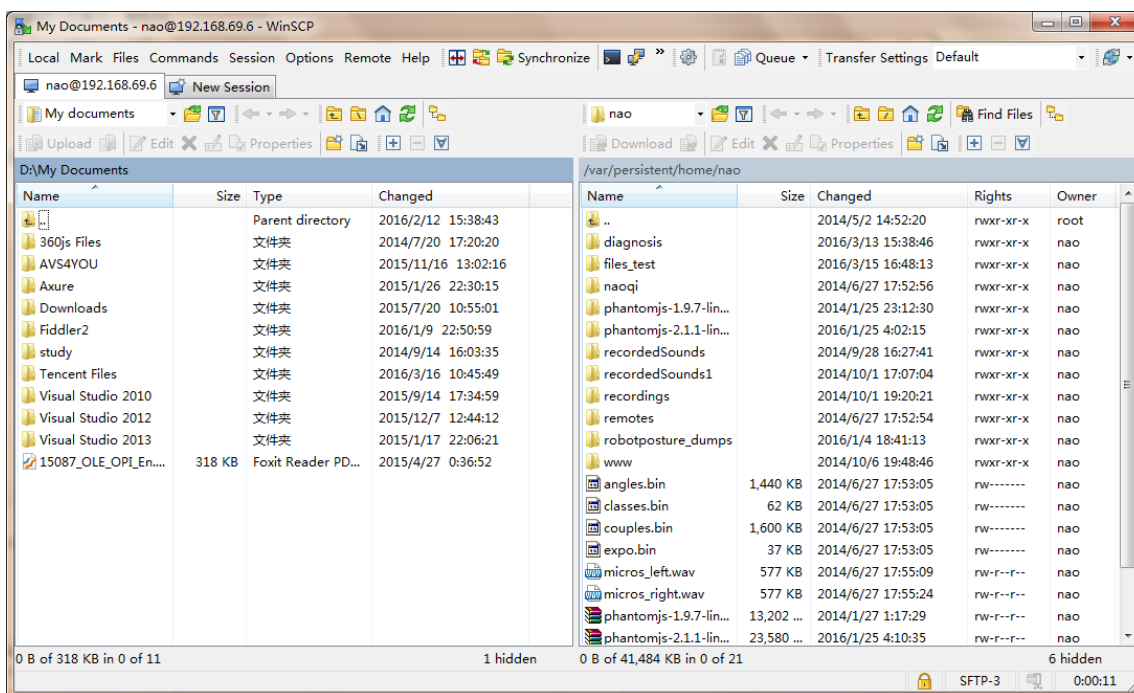


Figure 36. Operation interface of WinSCP

4 Speech Recognition Module

The speech recognition module acts the input device for the whole project. In order to make the NAO robot understand the user's voice command, in other words, the user's voice needs to be understood by the program. Thus the program needs to translate the user's speech to a character command which the program can understand.

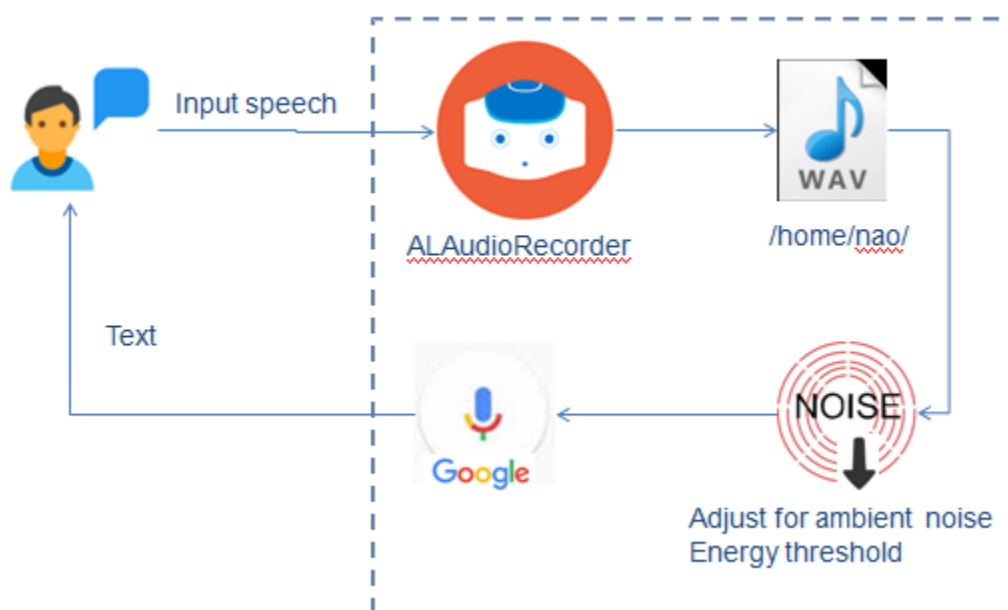


Figure 37. Relationship between user and speech recognition module

As the graph showed above, the NAO robot catches the user's speech input by NAO

robot's module which is named "ALAudioRecorder", this module can save the user's input speech as a wav file. In this project, I saved the wav file under /home/nao directory. After the user's input speech is saved successfully, the wav file will be adjusted for ambient noise and energy threshold. After processing, the wav file will use Google speech recognition API to output the characters and sentences which is inside the input speech, and integrate them into a complete text. Then the NAO robot will respond the text to user by using its "ALTextToSpeech" module to read the text out.

4.1 Speech recognition API

There are many mature speech recognition applications and most of them provide api for the programmer to use, following are several speech recognition APIs:

- 1) Google API – very accurate but not official, no language model or vocabulary configuration, limited to 50 requests/day, no commercial support whatsoever.
- 2) AT&T Speech API – accurate, vocabulary and grammar configuration, supports many languages.
- 3) Speech Technologies – Microsoft SAPI.
- 4) CMU Sphinx – Speech Recognition Toolkit, offline speech recognition, can be used on mobile./11/

By the way, for Chinese speech dialogue, Turing robot is a good choice.

According to this project requirement, to be recognized speech only contains simple phrases and recognized accuracy is very important, thus Google API is a good choice for this project.

Python has a library for speech recognition which can support Google Speech Recognition, Wit.ar, IBM Speech to Text and AT&T Speech to Text. The latest version is SpeechRecognition 3.3.2. In order to use the library, the version of python must be 2.6, 2.7 or 3.3+. Python 2.7 was used in this project. The most convenient way to install the library is using "pip install SpeechRecognition"/12/

The audio source could be got directly from a microphone if PyAudio is available, and is

undefined otherwise. A wav file could also be used to act as audio source. In order to make the program more intuitive, the audio source is from wav file.

Thus, as long as the program gets a wav file, the program can use python's speech recognition library to get the characters or sentences contains in radio. The next step is to save the user's speech in storage in .wav file format.

4.2 Recording wav file by NAO

In order to record wav file by NAO, "ALAudioRecorder" module from ALProxy is needed. Following is the initial code:

```
import argparse
from naoqi import ALProxy
import time
import os

robot_IP = "192.168.1.138"
tts = audio = record = aup = None

def record_NAO(robot_IP, robot_PORT=9559):
    global tts, audio, record, aup
    # -----> Connect to robot <-----
    tts = ALProxy("ALTextToSpeech", robot_IP, robot_PORT)
    record = ALProxy("ALAudioRecorder", robot_IP, robot_PORT)
    # -----> recording <-----
    record.stopMicrophonesRecording()
    print 'start recording...'
    tts.say("start recording...")
    record_path = '/home/nao/record.wav'
    record.startMicrophonesRecording(record_path, 'wav', 16000, (0,0,1,0))
    time.sleep(4)
    record.stopMicrophonesRecording()
    print 'record over'
    tts.say("record over")
    return
```

Figure 38. A piece of code

In order to make sure that the microphone of NAO has not been occupied, the microphone recording needs to be stopped before use it.

The command "time.sleep(4)" is used to leave 4 seconds for the user to say his voice command to the NAO robot. Four seconds is the appropriate time interval by test results.

4.3 Correction of recognition result

According to ambient noise, incomplete recordings, the limitations of voice recognition technology, and some other factors, it is not guaranteed that the recognized result is correct. Consequently, a confirmation from the user is necessary. Therefore, after getting user's voice and provide feedback to user, the speech recognition system will ask user to say "yes"

or “no” to confirm his directive.

5 WEB CRAWLER MODULE

This module introduced how to extract data from the internet using Python. The web pages can be divided into two categories. The first one is static page, which only runs on client side and do not run on the sever side, such as html, Flash, JavaScript, VBScript. The second one is dynamic page, which runs on the sever side and will different pages to different users in different times, such as ASP, PHP, JSP, ASP.net, CGI.

Static pages and dynamic pages have their own characteristics and advantages, a static page is the foundation of website construction. Static pages and dynamic pages are not contradictory. The web site selects either the dynamic page or the static page mainly depending on the functional requirements and content of the website. If the website function is relatively simple, and its content do not need to update frequently, it will be simpler to use pure static page method. On the contrary, dynamic page is usually selected in usual. In order to let the web site adapt to the needs of the search engine, even if dynamic web site skill is used in background, the page content can be translated into a static page to release.

In this project, the websites that need to be grabbed are both dynamic web pages. One of the websites is “VR”, which used to search train ticket information. Another one is “Yahoo Weather”, which used to search the latest weather forecasts.

Urllib2 is often used to obtain the entire HTML page, and find the corresponding word from the HTML file when using Python to crawl the conventional static web site. But when related to JavaScript, urllib2 is completely inadequate. Thus a simulation browser is required. There are a lot of ways, this project used selenium+PhantomJS, the reason lies in: Selenium2 supports all major browsers and includes non-interface browsers, such as phantomJS.

5.1 The train of thought and preparation

In order to learn how to do a web crawler, firstly, the Internet was used to help me to have

a basic impression of “web crawler”. Then the example which is a simplest web crawler on the Internet was followed in order to deepen my understanding of “web crawler”. Next, the same way was used to catch the information on VR, however the result only contained the structure of the whole web page but nothing about the content, such as the trains’ departure time, duration, prices and so on. The process seemed to be stuck. Therefore, where the trains’ detailed information came from was explored. The first exploring step was to read the source code of the web page. Luckily, a breakthrough was found here. The useful content’s table name is as same as the file name of a js file which is written in the front. It means that the trains’ information is dynamically generated. The former problem is because the method which used to grab static pages cannot be used to grab dynamic pages. Then the search question was changed from “how to grab web page” or “web crawler + how” to “how + grab dynamic pages”. Consequently, the solution for this project was found. Next, the methods and detailed implementation steps will be introduced in detail.

5.2 Tools introduction

- PhantomJS

PhantomJS is a server side JavaScript API Web Kit (open source browser engine). Its support for a variety of Web standards: DOM processing, CSS selector, JSON, Canvas and SVG. PhantomJS can be used for network monitoring, web page automation, and screenshots, no interface test etc.

- Selenium

Selenium is a tool used for testing Web applications. Selenium tests directly run in the browser, just as the real users operation. The supported browsers include IE (7, 8, 9), Firefox Mozilla, Suite Mozilla, PhantomJS etc. The main functions of this tool include: test the compatibility of the browser and system function. It is a acceptance test tool that specially written for web application by ThoughtWorks.

- BeautifulSoup

Except regular expression, BeautifulSoup is even a more powerful tool. With it we can easily extract the contents from HTML or XML Tags. In simple terms, BeautifulSoup is a library of python, its main function is to grab data from web page. It provides some

simple, python style functions which are used to handle navigation, search, and modify the function of the tree, etc. It is a toolbox, and provides users with their needed data through the analysis of documents. Because it is very simple, so a complete function can be written without a lot of codes.

- PIP

PIP is a software package manage system written in Python. It can install and manage software packages. In addition, a lot of software packages can also be found in the "Python package index" (Referred as PyPI). One of the main features of PIP is the ease of its command-line interface. This allows the user to easily install the Python software package through the following text commands:

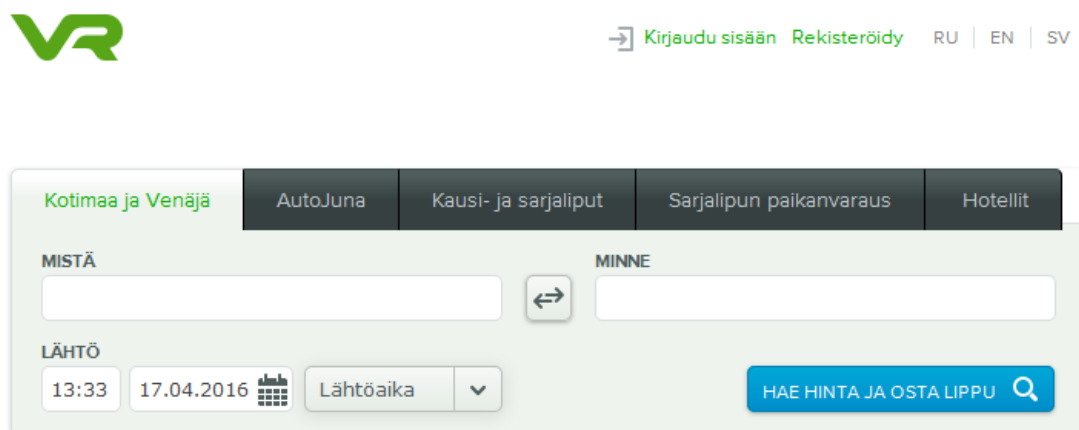
```
pip install some-package-name
```

In addition, user can easily remove the software package by following commands:/13/

```
pip uninstall some-package-name
```

5.3 Implementation Steps

This part used "VR" website as an example to go through the analysis and implementation steps. Following is the main page of the VR website:



The screenshot shows the VR website's main page. At the top left is the VR logo. To the right are links for "Kirjautu sisään" (Login), "Rekisteröidy" (Register), and language options "RU | EN | SV". Below this is a navigation bar with tabs for "Kotimaa ja Venäjä" (Home and Russia), "AutoJuna" (Car and Train), "Kausi- ja sarjaliput" (Season and Ticket), "Sarjalipun paikanvaraus" (Ticket reservation), and "Hotellit" (Hotels). The main content area features a search form with "MISTÄ" (From) and "MINNE" (To) input fields, a date and time selection section for "LÄHTÖ" (Departure) with a calendar icon, and a "HAE HINTA JA OSTA LIPPU" (Search price and buy ticket) button with a magnifying glass icon.

Figure 39. Input box of VR's main page

By pressing "F12" on the web page, the developer can enter developer tools (The part which is circled by red box).

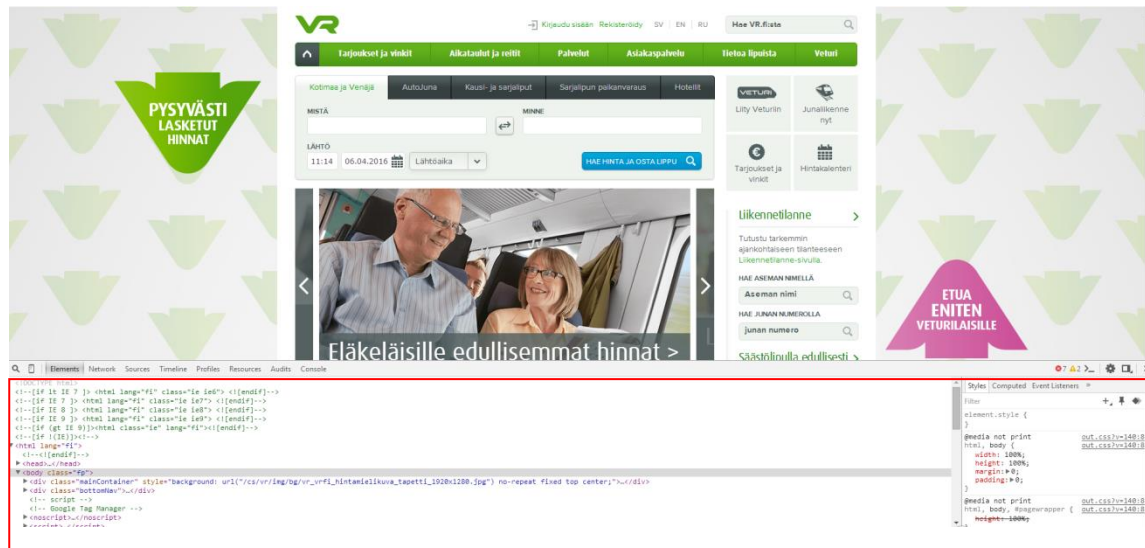


Figure 40. Developer tools

In the developer tools window, the user can see all the communications between the network and all the sound code of the current web page.

URL (Uniform Resource Identifier)

As this project is based on the English language web page, the first step we need to do is to switch the current page's language to "En". The web page will show as below:

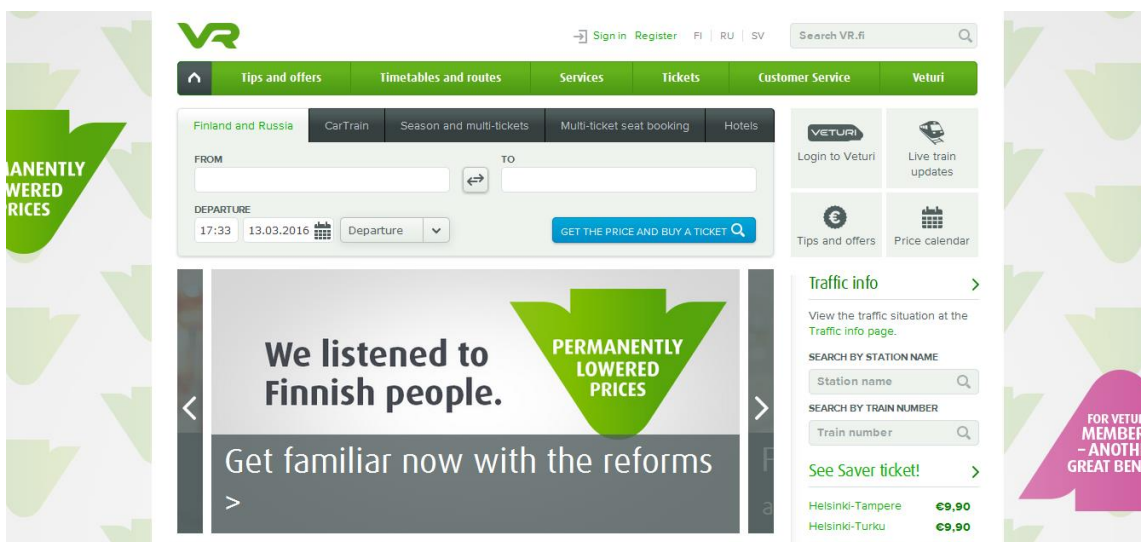


Figure 41. Main page in English

Press F12, developer tools can help the developer better in the overall handling. As shown in figure 42, the developer can quickly know what is the name, id and structure of the web page's components. For example, first choose the magnifier symbol, then click the text or input box you will see the developer tools already located the corresponding code on the

source code.

The screenshot shows the VR website's search interface. The search form includes fields for 'FROM' (Helsinki), 'TO' (Tampere), and 'DATE' (06.04.2016). The developer tool at the bottom shows the HTML structure of the search input field, highlighting the `<input id="tab1_station" type="text" class="asema sitesFrom sitesFromTab1 matkahakuDrop ui-autocomplete-input" name="sitesFromTab1" autocomplete="off" data-value="Helsinki" value="Helsinki" data-bbox="370 205 840 225">` element. A large green arrow points to the input field with the text "PERMANENTLY LOWERED PRICES".

Figure 42. The use of developer tools

The input boxes are the places where data needs to be filed in and then passed to the background of the web page. So after we found the places' name, we know where to fill in the data. However, we still do not know where to pass the data. So the next step is to find where to pass those data.

By clicking “Network”, the developer can check all the communications between different pages. This showed in the following figure:

The screenshot shows the Network tab in a developer tool. The table below lists the network requests, including the 'frontpage' document and various .js and .png files.

Name	Status	Type	Initiator	Size	Time	Timeline - Start Time
frontpage	200	document	Other	228 KB	3.82 s	
out.css?v=140	200	stylesheet	fontpage59		0 ms	
jquery.min.js	(failed)	script	fontpage58	0 B	19.67 s	
jquery-ui.min.js	(failed)	script	fontpage79	0 B	19.67 s	
out.js?v=140	200	script	fontpage82		0 ms	
jquery.ultimate-smartbanner.css	200	stylesheet	fontpage83		0 ms	
jquery.ultimate-smartbanner.js	200	script	fontpage86		0 ms	
stationlist.js?locale=en	200	script	fontpage147	64.3 KB	2.47 s	
edditthis_widget.js	200	script	fontpage5722		0 ms	
vr.png	200	png	fontpage213		0 ms	
vr_vrt_hinttamellikuva_700x350_8362_0mstakata.jpg?blobno...f...	200	jpeg	fontpage598		0 ms	
vr_happesana.png?blobno...f...	200	png	fontpage1538		0 ms	
vr_juna_ajonmaailmassa.jpg?blobno...f...	200	jpeg	fontpage1576		0 ms	
vr_vrt_hinttamellikuva_emg_700x350_tayvaik_ankata.jpg?blobno...f...	200	jpeg	fontpage2208		0 ms	
Fillakuva_700x350_asema_jappai.jpg?blobno...f...	200	jpeg	fontpage2499		0 ms	
veturi_koron_ajaja.png?blobno...f...	200	png	fontpage2792		0 ms	
junaliikenne_koron_ajaja.png?blobno...f...	200	png	fontpage2793		0 ms	
tr_ajon_ajaja.png?blobno...f...	200	png	fontpage2743		0 ms	
hintakalenteri_koron_ajonilalato.png?blobno...f...	200	png	fontpage2748		0 ms	
akatakalenteri.png?blobno...f...	200	png	fontpage4299		0 ms	
makuupalkki.png?blobno...f...	200	png	fontpage4347		0 ms	
parha.png?blobno...f...	200	png	fontpage4333		0 ms	
haarakka_ja_veturi.png?blobno...f...	200	png	fontpage4361		0 ms	

Figure 43. Network situation (1)

Following is an enlarged view of above situation, we can see that the “frontpage” document and many .js and picture files formed the home page of VR together.

Name	Status	Type
frontpage	200	document
out.css?v=140	200	stylesheet
jquery.min.js	(failed)	
jquery-ui.min.js	(failed)	
out.js?v=140	200	script
jquery.ultimate-smartbanner.css	200	stylesheet
jquery.ultimate-smartbanner.js	200	script
stationlist.js?locale=en	200	script
addthis_widget.js	200	script
vr.png	200	png
vr_vrfi_hintamielikuva_700x350_ENG_ilmantekstia.jpg?blobnocache=f...	200	jpeg
vr_leppavaara.png?blobnocache=false	200	png
vr_juna_syysmaiemassa.jpg?blobnocache=false	200	jpeg
vr_vrfi_hintamielikuva_eng_700x350_taysvalk_eitekstia.jpg?blobnocac...	200	jpeg

Figure 44. Network situation (2)

Then I filled in the input boxes data and sent them to web pages back end by clicking “Get the price and buy a ticket” button. Observe the column of network, as expected, I found the file’s name where data are sent to.

As showed in figure 45, the files name is “JourneySearch.do”.

Name	Status	Type	Initiator	Size	Time	Timeline – Start Tim
JourneySearch.do	302	x-www-form-urlencoded	www.vr.fi/cs/vrweb/jquery-1.9.1.m...	335 B	2.12 s	
243	301	text/html	SearchResultDomestic.do:2929	370 B	629 ms	
analytics.js	(failed)		verkkokauppa_skrptikoonti_bedyb...	0 B	21.00 s	

Figure 45. JourneySearch.do

Next step is to analyse the result page which contains the data we required. Figure 46 shows the results page:

The screenshot shows the VR website interface for a train journey from Helsinki to Vaasa. The page displays the departure time 10:06 and the class name tripStartDomestic. The developer tools are open, showing the HTML structure of the page, with the class name tripStartDomestic highlighted in the code.

Figure 46. Analysis of the data which needs to be grabbed (1)

By using developer tools I found the data and data's code name in a very easy way. Such as the first train in the search result, its departure time is "10:06" and class is "tripStartDomestic". So when I want to catch the data which is "10.06", I just need to find all the data which class name is "tripStartDomestic".

The screenshot shows the VR website interface for a train journey from Helsinki to Vaasa. The page displays the departure time 10:06 and the class name tripStartDomestic. The developer tools are open, showing the HTML structure of the page, with the class name tripStartDomestic highlighted in the code.

Figure 47. Analysis of the data which needs to be grabbed (2)

Similarly, several smaller screening ranges can be added to accurate my grabbed data. All the data I need to grab is under the form “journeyresdomestic” and the table “buyTrip_1”. Under every “tbody” label, there is one choice of the train, and the data that we need to grab is departure time, arrival time, duration, prices and transfer information.

```

<div id="introText"></div>
▼ <form id="journeyresdomestic" name="journeyresdomestic" action="/onlineshop/SearchResultDomestic.do" method="post">
  ▼ <div id="journeyresultdomestic">
    <input type="hidden" name="struts.token.name" value="struts.token">
    <input type="hidden" name="struts.token" value="AEEWM5TELC6ZJ7APW68DOTYKGUY0H36H">
    <h2 class="tripheading">...</h2>
    <p class="tripSubHeading">...</p>
    <div class="buySwitchDay viewTimetable">...</div>
    <input type="hidden" name="prevConnectionsUrl" value="./JourneyBrowsePrevDomestic.do" id="prevConnectionsUrl">
    <div class="btnEarlier viewTimetable">...</div>
    ▼ <table id="buyTrip_1" class="domesticTimetable vrShopTable buyTimetable">
      <thead>...</thead>
      <tbody>
        ▼ <tr class="tripOption tripOptionDomestic tripOptionWide closed" id="option_0_0">
          <td class="paddingColumn"></td>
          <td class="tripStartDomestic">10:06</td>
          <td class="tripEndDomestic">14:36</td>
          <td class="tripDurationDomestic tripTrainTypesDurationFont">04:30</td>
          <td class="tripTrainTypesDomestic tripTrainTypesDurationFont">...</td>
          <td class="tripPriceFrom">...</td>
        </tr>
        <tr class="tripDetailsSeparator">
          <td colspan="5"></td>
        </tr>
        <tr class="tripDetails" id="details_0_0" style="display: none;">...</tr>
      </tbody>
    </table>
    <input type="hidden" name="isNextDayStart0" id="isNextDayStart0" value="true">
  </div>
  <!-- CHG0035433/TMPRD2.3/extVeMa2 - additional action link added for next day journey search from 00:00:00 -->
  <div class="btnLater">...</div>
  <div class="btnarea">...</div>
  <div class="productInfo prod3476" style="display: none;">...</div>
  <div class="productInfo prod3414" style="display: none;">...</div>

```

Figure 48. Analysis of the data which needs to be grabbed (3)

```

▼ <tbody>
▼ <tr class="tripOption tripOptionDomestic tripOptionWide closed" id="option_0_0">
  <td class="paddingColumn"></td>
  <td class="tripStartDomestic">10:06</td>
  <td class="tripEndDomestic">14:36</td>
  <td class="tripDurationDomestic tripTrainTypesDurationFont">04:30</td>
  ▼ <td class="tripTrainTypesDomestic tripTrainTypesDurationFont">
    "
    "
    <br>
    "
    "
    1&nbsp;transfer:"
    "
    "
    InterCity train
    >
    InterCity train
    "
  </td>
  ▶ <td class="tripPriceFrom">...</td>
</tr>
<tr class="tripDetailsSeparator">
  </tr>
▶ <tr class="tripDetails" id="details_0_0" style="display: none;">...</tr>
</tbody>
▼ <tbody>
▼ <tr class="tripOption tripOptionDomestic tripOptionWide opened" id="option_0_1">
  <td class="paddingColumn"></td>
  <td class="tripStartDomestic">13:06</td>
  <td class="tripEndDomestic">17:34</td>
  <td class="tripDurationDomestic tripTrainTypesDurationFont">04:28</td>

```

Figure 49. Analysis of the data which needs to be grabbed (4)

```

▼ <td class="tripPriceFrom">
  ▼ <div class="priceBox " data-price="41,00" data-product="3414">
    ▶ <div class="priceBoxHeader">...</div>
    <div class="priceBoxPrice">
      41,00&nbsp;€
    </div>
    <input type="radio" id="0_1_3414" name="outwardJourneyInput" value="1_3414">
  </div>
</td>

```

Figure 50. Analysis of the data which needs to be grabbed (5)

By analysing the source code, it was found that the whole table which contains the useful data was sent to “onlineshop/SearchResultDomestic.do”. But the source code do not contain anything called “SearchResultDomestic.do”. Then by analysing the web pages’ network communication, a file named “journeysdomestic” was found. However, there is also a javascript file which has the same name, as shown in Figure 51. This means the trains’ information is generated by javascript, in another word, it is dynamic.

```

<!DOCTYPE html>
<html id="fi" xml:lang="fi" xmlns="http://www.w3.org/1999/xhtml" lang="fi">
  <head>
    <script type="text/javascript" async src="https://service.giosg.com/static/giosgclient.app.build.74090d9b4b4e.js"></script>
    <script async src="//www.google-analytics.com/analytics.js"></script>
    <script async src="https://service.giosg.com/live/"></script>
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta http-equiv="content-type" content="text/html; charset=UTF-8">
    <meta name="description" content="VR Verkkokauppa">
    <title>Online Shop</title>
    <style type="text/css"></style>
    <link rel="stylesheet" href="/pages/static/css/styles.css" type="text/css">
    <!--[if lte IE 7]>
      <link rel="stylesheet" href="/pages/static/css/styles-ie7.css" type="text/css" />
    <![endif]-->
    <link rel="SHORTCUT ICON" href="/pages/static/favicon.ico">
    <script id="thxTag" type="text/javascript" async src="https://eu2.thunderhead.com/one/rt/js/one-tag.js?siteKey=ONE-FCIY4RC8WU-7326"></script>
    <script type="text/javascript" src="/pages/static/js/external/jquery-1.4.2.min.js"></script>
    <script type="text/javascript" src="/pages/static/js/external/jquery-ui-1.8.18.custom.min.js"></script>
    <script type="text/javascript" src="/pages/static/js/external/jquery.countdown.js"></script>
    <script type="text/javascript" src="/pages/static/js/external/jquery.form.js"></script>
    <script type="text/javascript" src="/pages/static/js/common/scripts.js"></script>
    <script type="text/javascript" src="/pages/static/js/external/jquery.boxy.js"></script>
    <script type="text/javascript" src="/pages/static/js/external/jquery.datepick.js"></script>
    <script type="text/javascript" src="/pages/static/js/external/jquery.datepick-fi.js"></script>
    <script type="text/javascript" src="/pages/static/js/external/swfobject.js"></script>
    <script type="text/javascript" src="/pages/static/js/external/jquery.autocomplete.js"></script>
    <script type="text/javascript" src="/pages/static/js/domestic/journeysdomestic.js"></script>
    <script></script>
    <script type="text/javascript" src="https://www.vr.fi/cs/vr/verkkokauppa_skriptikoonti_head.js" async="async"></script>
    <script src="//dev.visualwebsiteoptimizer.com/j.php?a=20593&u=https%3A%2F%2Fshop.vr.fi%2Fonlineshop%2FSearchResultDomestic.do&r=0.07372100581414996"></script>
    <link rel="stylesheet" type="text/css" href="https://service.giosg.com/static/visitor/css/giosg/giosg.css?v=ef63470aa377ee5c3f89b5e8ed8ffd39" media
    <style type="text/css" id="giosg-theme-css"></style>

```

Figure 51. Analysis of the data which needs to be grabbed (6)

Following the detailed steps about how to implement the grabbing of dynamic web pages will be introduced.

- Install Selenium IDE and input the web page's URL in "Base URL". This software will help us catch what the current web page has done during the user's operation.



Figure 52. Selenium IDE icon on Firefox.

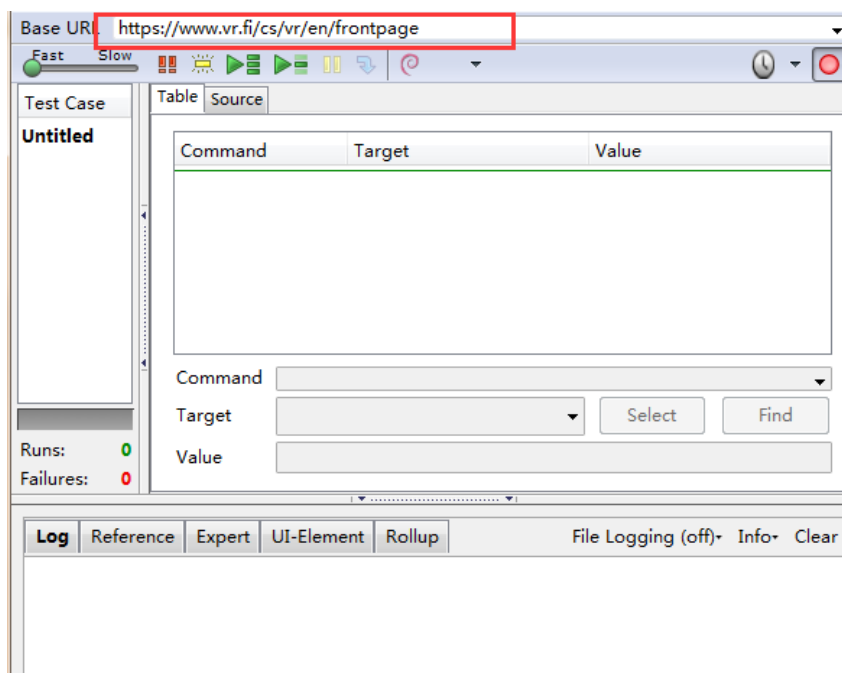


Figure 53. User interface of selenium IDE

Following is the exported file from selenium IDE. This file gives us an example of how to simulate the operation of a browser. In my personal computer, Firefox was used as my browser driver. But in order to get the robot away from the personal computer, I used “phantomJS” as a browser which has no interface to instead Firefox. Through a single layout, I got the result I would like to see.

```

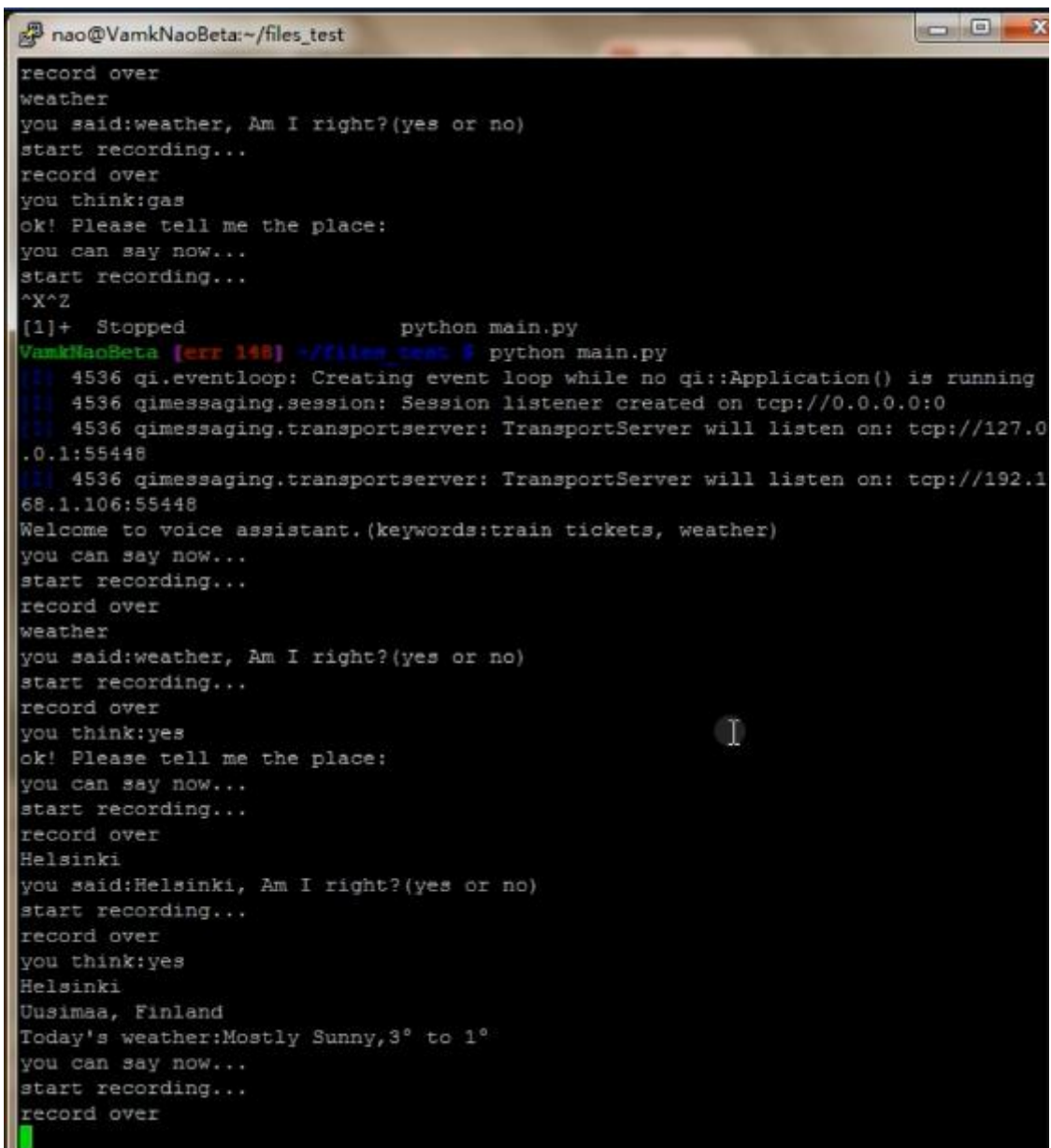
1  # -*- coding: utf-8 -*-
2  from selenium import webdriver
3  from selenium.webdriver.common.by import By
4  from selenium.webdriver.common.keys import Keys
5  from selenium.webdriver.support.ui import Select
6  from selenium.common.exceptions import NoSuchElementException
7  from selenium.common.exceptions import NoAlertPresentException
8  import unittest, time, re
9
10 class Reference(unittest.TestCase):
11     def setUp(self):
12         self.driver = webdriver.Firefox()
13         self.driver.implicitly_wait(30)
14         self.base_url = "https://www.vr.fi/cs/vr/en/frontpage"
15         self.verificationErrors = []
16         self.accept_next_alert = True
17
18     def test_reference(self):
19         driver = self.driver
20         driver.get(self.base_url + "/cs/vr/en/frontpage")
21         driver.find_element_by_id("tabs1_submitbutton").click()
22         driver.find_element_by_id("tabs1_submitbutton1").click()
23
24     def is_element_present(self, how, what):
25         try: self.driver.find_element(by=how, value=what)
26         except NoSuchElementException as e: return False
27         return True
28
29     def is_alert_present(self):
30         try: self.driver.switch_to_alert()
31         except NoAlertPresentException as e: return False
32         return True

```

Figure 54. Exported file from selenium IDE

7 RESULTS

Following are the output captures of the whole project.



```

nao@VamkNaoBeta:~/files_test
record over
weather
you said:weather, Am I right?(yes or no)
start recording...
record over
you think:gas
ok! Please tell me the place:
you can say now...
start recording...
^X^Z
[1]+  Stopped                  python main.py
VamkNaoBeta [err 148] ~/files_test 3 python main.py
[1] 4536 qi.eventloop: Creating event loop while no qi::Application() is running
[1] 4536 qimessaging.session: Session listener created on tcp://0.0.0.0:0
[1] 4536 qimessaging.transportserver: TransportServer will listen on: tcp://127.0
.0.1:55448
[1] 4536 qimessaging.transportserver: TransportServer will listen on: tcp://192.1
68.1.106:55448
Welcome to voice assistant. (keywords:train tickets, weather)
you can say now...
start recording...
record over
weather
you said:weather, Am I right?(yes or no)
start recording...
record over
you think:yes
ok! Please tell me the place:
you can say now...
start recording...
record over
Helsinki
you said:Helsinki, Am I right?(yes or no)
start recording...
record over
you think:yes
Helsinki
Uusimaa, Finland
Today's weather:Mostly Sunny,3° to 1°
you can say now...
start recording...
record over

```

Figure 55. Result capture (1)

```

nao@VamkNaoBeta:~/files_test
train tickets
you said:train tickets, Am I right?(yes or no)
start recording...
record over
you think:yes
ok! Please tell me departure date of the train:
----Year month date(ex,2016 March 15)-----
you can say now...
start recording...
record over
2016 March 26
you said:2016 March 26, Am I right?(yes or no)
start recording...
record over
you think:yes
03
---Departure station(ex,Vaasa)-----
you can say now...
start recording...
record over
Helsinki
you said:Helsinki, Am I right?(yes or no)
start recording...
record over
you think:yes
---Destination station(ex,Helsinki)---
you can say now...
start recording...
record over
rovaniemi
you said:rovaniemi, Am I right?(yes or no)
start recording...
record over
you think:yes

https://shop.vr.fi/onlineshop/SearchResultDomestic.do 2
End1
End2
End3
End4
End5
From Helsinki to rovaniemi, the date is 26.03.2016

```

Figure 56. Result capture (2)

```

From Helsinki to rovaniemi, the date is 26.03.2016
I found 7 trains
Choice 1
Departure_time:10:06 Arrival_time:19:50
InterCity train
Following are prices info:
Basic Ticket 68,00 €

Choice 2
Departure_time:18:52 Arrival_time:07:49
InterCity train
Following are prices info:
Basic Ticket 68,00 €

Choice 3
Departure_time:19:19 Arrival_time:07:49
1 transfer: Commuter train R > InterCity train
Following are prices info:
Basic Ticket 67,00 €

Choice 4
Departure_time:20:06 Arrival_time:07:49
1 transfer: InterCity train > InterCity train
Following are prices info:
Basic Ticket 68,00 €

Choice 5
Departure_time:21:52 Arrival_time:10:47
InterCity train
Following are prices info:
Basic Ticket 80,00 €

```

Figure 57. Result capture (3)

```
Choice 6
Departure_time:22:19 Arrival_time:10:47
1 transfer: Commuter train R > InterCity train
Following are prices info:
Basic Ticket 79,00 €

Choice 7
Departure_time:23:06 Arrival_time:10:47
1 transfer: InterCity train > InterCity train
Following are prices info:
Basic Ticket 80,00 €

you can say now...
start recording...
record over
no thank you
VankNaoBeta [0] ~files_sans_6 █
```

Figure 58. Result capture (4)



Figure 59. NAO robot

Figure 59 shows the NAO robot's reaction during the experiment. When NAO robot is talking, he will rotate his LED around his body at the same time in order to make NAO looks more friendly and cute.

8 RECOMMENDATION FOR FUTURE RESEARCH

8.1 Improving NAO's software and hardware

In this project, one of the difficulties was that NAO's software is not easy to use for a beginner like me. If the NAO robot's software can be updated into a more graphical and commonly used one, more excellent applications will be developed.

8.2 Improving web crawler technology

In this project the NAO robot does not really buy the train tickets, he just needs to check train ticket information. But in the use of reality, after checking the train tickets information, the user may would maybe like to buy the tickets directly. So it is necessary to add the function of buying train tickets via the Internet by NAO. But the user's bank card and payment information are very confidential, so the security work still needs to be much developed and improved.

8.3 Improving voice recognition technology

In this project the NAO robot used Google speech recognition API to do the voice recognition work, it is easy to use, but because the database of speech recognition is widespread, it lacks pertinence. In the future, if the user can set some restrictions to speech recognition's database by some keywords such as "time", "place", "price" and so on. It could reduce a lot of recognition errors and enhance the speed and accuracy of recognition.

9 SUMMARY

This thesis combines web crawler technology with voice recognition technology and artificial intelligence. It is a meaningful attempt to show the bright future of artificial intelligence, although this thesis still has a lot of things which can be improved, such as the network security issues which mentioned in the previous chapter.

The idea of this paper is to show how much can be done with artificial intelligence. The results of this thesis show the great potential of the future of artificial intelligence's future. Robots are gradually permeating people's lives. They can do a lot of things instead of human beings, from teaching our children to taking care of old people.

By doing this thesis, I also learned a lot of things during the process. Many times I think I had arrived and received the right answer, but the practical results told me I had not. This experience has taught me that you should not take anything for granted, try it and you will get the answer. And when you face problems, you should always face them and find a way to solve them. It is normal that you know nothing at the beginning, however you should always try to learn and try to do.

REFERENCES

- /1/Active8 Robots official webpage. Accessed 1 April 2016.
<http://www.active8robots.com/wp-content/uploads/nao-robot1.jpg>
- /2/History of NAO Robot. Accessed 1 April 2016.
<https://www.aldebaran.com/en/humanoid-robot/nao-robot>
- /3/Hardware Platform of NAO Robot. Accessed 1 April 2016.
<https://www.aldebaran.com/en/more-about>
- /4/NAO V5 loudspeakers. Accessed 1 April 2016.
http://doc.aldebaran.com/2-1/family/robots/loudspeaker_robot.html
- /5/Dimension of NAO V5. Accessed 4 April 2016.
http://doc.aldebaran.com/2-1/family/robots/dimensions_robot.html
- /6/ Microphones of NAO V5. Accessed 2 April 2016
http://doc.aldebaran.com/2-1/family/robots/microphone_robot.html
- /7/History of voice recognition technology.
<http://www.almosthuman.cn/2016/01/16/kglr1/>
- /8/Principle of voice recognition. Accessed 2 April 2016.
<https://www.zhihu.com/question/20398418>
- /9/NAOqi OS. Accessed 2 April 2016.
<http://doc.aldebaran.com/2-1/dev/tools/opennao.html>
- /10/Choregraphe Panels. Accessed 2 April 2016.
<http://doc.aldebaran.com/2-1/ref/python-api.html#naoqi-python-api>
- /11/Speech Recognition APIs. Accessed 3 April 2016.
<https://www.quora.com/What-are-the-top-ten-speech-recognition-APIs>.
- /12/Python library Speech Recognition. Accessed 22 April 2016.
<https://pypi.python.org/pypi/SpeechRecognition/>
- /13/PIP, package manager. Accessed 5 April 2016.
https://en.wikipedia.org/wiki/Pip_%28package_manager%29
- /14/Web crawler. Accessed 5 April 2016.
<http://www.cnblogs.com/wawlian/archive/2012/06/18/2553061.html>

/15/PhantomJS. Accessed 6 April 2016.

<https://medium.com/@hengkiardo/installing-phantomjs-1-9-7-on-ubuntu-12-xx-x64-x86-7e772b4796f4#.ao8bxz92m>

/16/Web crawler with dynamic web pages. Accessed 26 April 2016.

<http://blog.csdn.net/lambert310/article/details/49248109>

/17/Python, selenium and javascript. Accessed 26 April 2016.

<http://www.coder4.com/archives/4426>