Humanoid robot communication

D. Ódor*, L. Gugolya**
* Óbuda University – Alba Regia Technical Faculty, Székesfehérvár, Hungary
odor.dominika95@gmail.com
gugolya.laszlo@amk.uni-obuda.hu

***Abstract*—This article gives you a insight of how it works to speak with a humanoid robot. And how the speech recognition works in those machines. I am going to show the limits of those technologies.**

I. INTRODUCTION

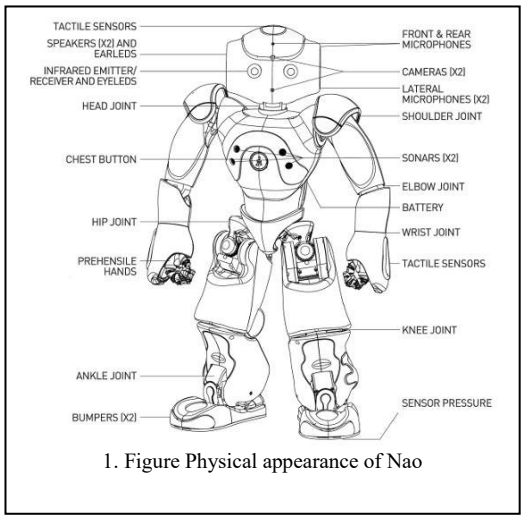
My goal is that the reader of the following lines understands how the speech recognition of the robots works. Which system better for the exercises and what the problem whit the construction of those robots. This theme is all based on my experiences.

A. The humanoid robots

I tested those humanoid robots. One was Nao version V5[2], and the other one was Pepper V1.8a version[3]. Those two robots are manly different in their physique appearances and slightly different in their software parts.

I am not going to introduce all the parts of those robots, because most of them are irrelevant in my theme.

The Nao robot is a bit older than the Pepper robot.



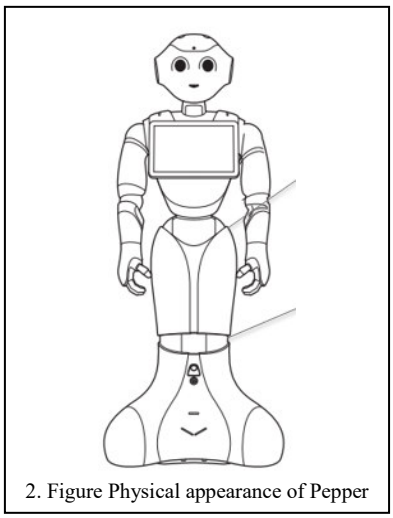
1. Figure Physical appearance of Nao

This was maybe the first humanoid robot that could interact and communicate with people on a normal way.

Pepper, the other robot is a little newer than Nao. It was designed differently, and they used better parts to make it.

The creators of those robots made a special platform to make the programming easier for individuals.

The old software of those robot uses a modular platform, but if somebody wants to make something more complex, then it is possible to do Python programming or use them mixed. Users can modify the modules too.

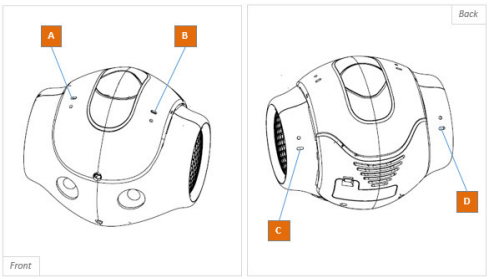


2. Figure Physical appearance of Pepper

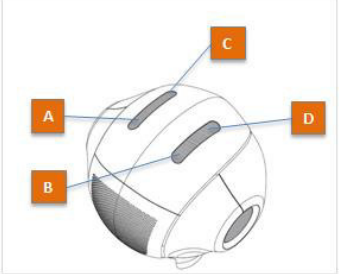
As you can see in the table below, in Nao they used a different TTS – Text To Speech – engine than the Pepper. But in the newer robot they changed it to the world-wide used Nuance.

Table I.
Nao and Pepper compare

Robots	Nao	Pepper
"Age"	9 year	4 year
Used speech recognition system	ACAPELA	NUANCE
Locate of microphones	on the front and back of	on the top of its head



3. Figure Location of the microphones on Nao

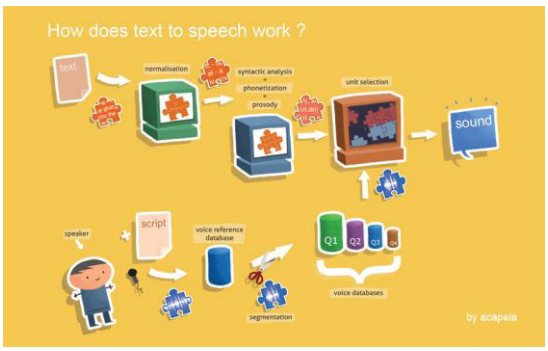


4.Figure Location of microphones on Pepper

II. How speech recognition works

A. The process

For the better understanding you should look at the picture below.



5. Figures The process of speech recognition

The first step to reproduce the natural sound of any language, a people records different contents which contain every possible sounds of the chosen language.

Then those recordings are segmented and sorted into an acoustic database. After this the system are creating new databases where it puts in the further sliced recordings, like sentences, phrases, words and etc.

When we reproduce words, the TTS system begins a special analysis that transfers written text into acoustic text. Before the system plays the sound, it sets the proper toning and pronounce.

The last step when we can hear the results.

B. AKAPELA

The Acapela[1] group did this speech recording service that was used at Nao. It is hard to decide if it is a well written program or not, because there can be other issues too that can manipulate the results.

In the description of Nao it is written that people need to speak with it from 1 meter distance and a bit louder than the normal.

The problem that makes things more difficult for Nao to understand the sentences, can be the cooling system in the back of its head. Maybe it makes so much noise that it causes misunderstandings or do not even understand the person who tries to communicate with it.

It is used nowadays too for education robots, toys and for people whit voice damage. They can create a new or similar to the original sound.

C. NUANCE

This platform is more popular than the other. It is used in iPhones for the Siri API and other in smart devices.

In Pepper case it works more reliable than the ACAPELA in Nao. It understands much more precisely. But if we compare the two robots in this aspect, it can be that Pepper understands better because its microphones are better and more sensitive than Nao's. But this can work hectic too, because the creators put there too the cooling system into Pepper's head. So, the cooling fan can bother the recognition here too.

But if we compare the two system, then I think that NUANCE is more advanced than the ACAPELA. We can even choose from more languages.

III. Speak with a humanoid robot

As I already mentioned there is specific software that is used for those human-like robots. Its name is Choregraphe [4]. With this, users can create a unique dialog or smaller speeches.

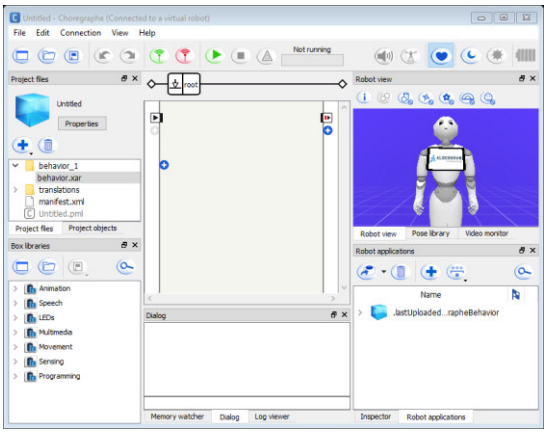
But before a person begins to do something, it is very important that those robots need a network via they can communicate with their TTS system. If it is not possible then we cannot speak with our robots, because those robots make a real time speech recognition.

So, if there is no network signal then we have robot doll that are not able to any interaction with people.

Another important thing if we speak other language we must strive to say the words with more regular pronouncements.

So let's start. On the next picture you we can see the surface of the Choregraphe software.

As we can see, it is a pretty complex but easy to handle program. It has a virtual robot part too, where we can test

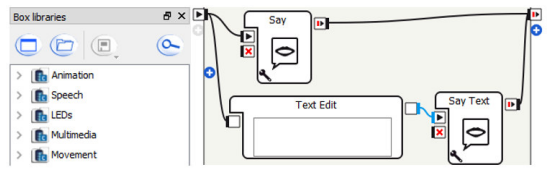


6. Figures Choregraphe

the programmers that the users wrote. So we do not need to wait until we can use the real humanoid robots, because there are many segments we can try at those virtual machines.

This would be the ideal software if we could test every part of the written application we made and then we should not even have to own a humanoid robot. But sadly this is not the case. There are many functions that we cannot test on the virtual robot. Some of those are the speech recognition and the tablet functions. For those actions we need the real ones.

But out of this, it has a very user friendly environment, because this in this application we are capable to make programmers in a modular way. The creators of this program gave the us an environment that the beginners can easily use after they get to know the basic knowledge.



7. Figures Choregraphe windows

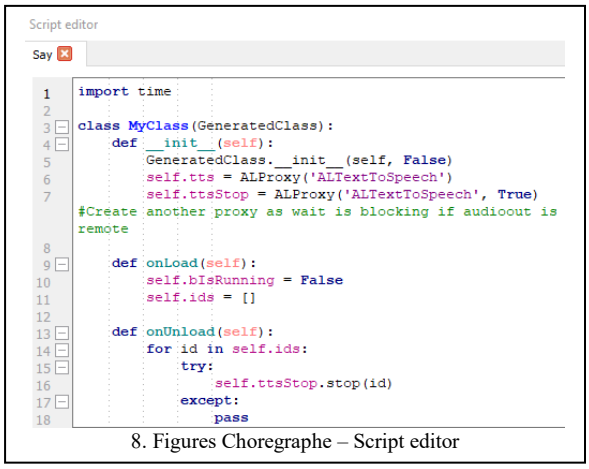
On the left picture we can see the Box libraries menu, where we can find different modules that the users can use for their plans. There are many categories that covers most of the parts and functions of the robots. With those, anybody can make a simple or a little more complex application.

As you can see on the right figure, there are some of these modules, which can make our robots say something. Both variations do the same thing. In the case of the upper one we need to open its setting on the left side and write there the text we want, on the other hand in the second solution there is a Text Edit box where we need to write the text and this box must be connected to the other Say Text box.

There are not any significant difference in the two solution, but for some reason the first one is not as reliable as the second one. There were many times when the robot did not say it. So in many cases I used the other one, because that has always run.

Out of the text we can change many other things too. Some of those the pronunciation, the voice shaping, the speed, the volume and etc. Some of those setting can be modified if we open the settings of the Say boxes, but most of these must be written in the text by the user. Before I show this method I am going to introduce the readers another, a more professional method for the programming.

In this case we will not use the modules, because we will write our codes in Python. The humanoid robots and the Choregraphe too compatible with this language. And if we open the script of a box, we can see that it will show us a Python code.



8. Figures Choregraphe – Script editor

If we do not want to type the whole code ourselves, then we can copy or modify the current code of a box. With those we can create a more complex program than before.

By the way we do not want to use the Choregraphe then we need to make sure to write in the code a connection part with the robot.

```
from naoqi import ALProxy
tts = ALProxy("ALTextToSpeech", "the ip of your robot", 9559)
```

We need to call the ALTextToSpeech module if we want to make our robot to say something. The next thing we need is our robot IP address. The last step is to call the say function.

```
tts.say("Hello world!")
```

After this we play our code, then the real robot are going to say the text. If this is not happening, then check if you have internet connection, the IP address of your robot and your code.

How can we modify the voice of our robot? As I have already mentioned there is a solution for this. The developers made special tags that we need to put in our text in the right place for this. I am going to introduce some of the basic tags.

•\\vct=value\\

With this tag we can change pitch of the voice. We can set the value in percent between 50 and 200. The default value is 100. For example:

```
# Say the sentence with a pitch of +50%
tts.say("\\vct=150\\Hello my friends")
```

•\\rspd=value\\

This will change the speed of the speech. Just like the previous one we use percentage as value, but there we can modify it between 50 and 400. The default there too 100. Example:

```
# Say the sentence 50% slower than normal speed
tts.say("\\rspd=50\\hello my friends")
```

•\\pau=value\\

We can guess what this tag will do. With this we can put pause inside our text. Its value needs to be given in millisecond. Example:

```
# Insert a pause of 1s
tts.say("Hello my friends \\pau=1000\\ how are you ?")
```

•\\vol=value\\

In this case we can modify the volume. The value can be given in percentage between 0 and 100. Exapmle:

```
# Say the sentence with a volume of 50%
tts.say("\\vol=50\\Hello my friends")
```

•\\rst\\

With this we can reset all the settings we did so far.

```
tts.say("\\vct=150\\\\rspd=50\\Hello my friends.\\rst\\ How are you ?")
```

There are many other tags that we can use during our programming, but they can be used just on the Pepper humanoid robot, because that machine uses the Nuance system. And just with this system can we change some options. This can be the reason why the SoftBank Robotics used Nuance for the new released Nao robots.

One of the most important options is that we can change any words pronunciation. With this we can set how the robot will say the selected words. Although the robot uses a library for its speech recognition via the internet but if we change the specifications of the word then that will be stored on the robot so those settings will remain even if the robot is switched off.

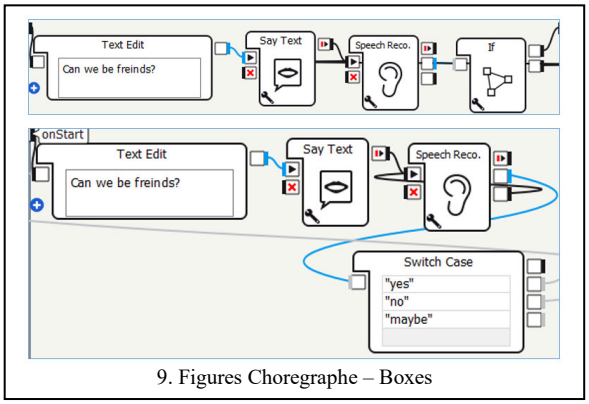
For this we use the addToDictionary function of the ALTextToSpeech modul. Example:

```
tts.setLanguage("English")
# Say Emile in english
tts.say("My name is Emile.")
# Add Emile to dictionary
tts.addToDictionary("Emile", "\\toi=lhp\\E'mil\\toi=orth\\")
# Test it
tts.say("My name is Emile.")
# Delete the word Emile
tts.deleteFromDictionary ("Emile")
# Re-test it
tts.say("My name is Emile.")
```

As you can see on the figure above the first step is to set the language which we want to modify the word. The next step is that we make the robot to say out the current text. Then we change the pronunciation of the word, Emile. After this we test it. If we want that this modification will be temporarily then we need to delete it with the deleteFromDictionary function. At last let's test it if we get back the original word.

It is all good, that we can make our robot to say monologues, but what we want is that we can make conversations with our robot. For this task we have two solutions.

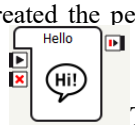
First we can use the Say modules in the Choregraphe and we can complete it with a Speech recognition and an If or Switch case boxes.



9. Figures Choregraphe – Boxes

On the top picture I used the „if" version. In this scenario we must set the condition operator and the value we want our robot to understand. We can add multiple words or sentences if we separate them with „ ; " this sign. But with this we can just make it able to understand our words but after this it is up to us, how it should treat it or what it should do.

But if we use the Switch Case box then this problem would be automatically solved, because each word has its own output that we can connect anywhere we want.

Although we want to make a proper conversation we need from them many copies and many so questions, that we can guess what the answer will be. But for this problem the developers created the perfect module. And this one is the Dialog box. 

This box is already renamed by me.

If we open its script it is a bit different than the other boxes.

```
concept:(greetings) ^rand[hi hello "hey there"]

concept:(drink) [red white] wine
concept:(alcohol) [beer ~drink]

u:(~greetings) ~greetings
u:(do you have _~drink) yes, I have $1
u:(I want to drink something) do you want ~alcohol?
```

There we will not see any code, just the ones we use for making the dialog. The one I used for this example is a simler one. There we can see some lines that I will describe. As we can see on the picture there are 3 concepts where each contains different words. If the words are between in [] those containers, then it means we can say any of them and the robot will understand it. If there is a phrase then we need to put between quotation mark as we can see by the greetings. Buti f there is a word out [] those signs then that means that word is fix.

The real conversation starts under those lines. The wavy sign with the word „greetings" means that when the user say one word from the greetings concept the robot will understand it and says a greeting from the same concept, and it will be a random one. After this if we ask the robot the next question and say a drink from the possible answers then the underline means our robot saved our answer and in its sentence the „$1" means that it will say out our choice.

With this I showed many possible ways that will lead us to having a simple conversation with our robots, but there so many other possibilities that we could use during those tasks that I could write a whole book from them. But this is a preview from what we are capable to do on a humanoid robot. And if it is piqued someone interest then they should take a visit at the website of the Aldebaran.

IV. SUMMARY

My opinion in this theme that we are on the right path to make a humanoid robot that can understand everything that a person says to them. There are multiple TTS and voice recognition systems outside of the mentioned ones. It is good that we have those awesome services, but it is equally important that the hardware part be functional without bothering each other.

I do not know what the developers' solution will be, but in short time we will found out, because they are already making a new type robot, the so-called Romeo. It will be used like a caretaker for old people, so they must solve those problems.

REFERENCES

[1] http://www.acapela-group.com/voices/how-does-it-work/
[2] http://doc.aldebaran.com/24/family/nao_h25/index_h25.html#nao-h25
[3] http://doc.aldebaran.com/25/family/pepper_user_guide/repair/brake_release_keys.html
[4] http://doc.aldebaran.com/2-5/index.html