Speech recognition for NAO

Tim Groot, Cornelis Boon, Amir Alnomani, Joris Timmer
University of Amsterdam
Science Park 1098 XH Amsterdam

June 27, 2014

Abstract

A NAO robot is a humanoid robot, mostly used for robot soccer. The emphasis of this research is to let the robot execute certain motions associated with the verbal command accordingly. This was done by differentiating between the different commands by using speech recognition. The Nao-robot is very convenient for tasks related to this one, which is why the Nao was chosen over other robots, as well as that it's relatively easy to program. Another advantage of the NAO is that it can also be controlled remotely using an internet connection, which is very convenient as it's not necessary to update the NAO software, because any updates get uploaded directly to the robot. The application called Choregraphe, provided by Aldebran Robotics, is used to develop complex behaviours for the robots without having to write any code, but its also possible to use python or c++ to develop behaviours. In this experiment a NAO was used to recognize speech and act accordingly to the commands it was given.

Introduction

The NAO is a humanoid robot meant as a fully programmable companion for humans. The NAO has been an important step for the advancement of companion robots. NAO robots can be very useful as they can be made fully autonomous. This way the robot can be programmed to simulate a human being. The aim of this research is to find out how to make NAO react to different commands and execute different behaviours associated with those commands. Therefore, the research question is: How can a NAO efficiently execute different motion commands to its relative speech activation? As this question suggests, this experiment strives to create different tasks that can be executed by certain corresponding commands. The hypothesis states that multiple different commands should be able to work side by side. Starting off with just a single one, we will continue to add more layers gradually, all experimenting with speech- and soundstimuli as input. Once a speech to text algorithm is implemented, it should be easy to add more commands/tasks. Therefore, the main problem we are facing here is making an algorithm which can convert a sound file to a string of text or another form of data that can be analysed easily, such as a numerical value.

Materials and Method

The main tool we used for programming the NAO was the Choregraphe application version 1.14.5 part of NAOqi with which it's possible to create behaviours using a specific graphical interface. Its also possible to use python scripts within Choregraphe for more specific behaviours.

First we implemented the command animations using Choregraphe, the commands include: bow, push-ups, head bang and salute. The bow animation makes the NAO bow, and say ”yes, sir”. See the image below.

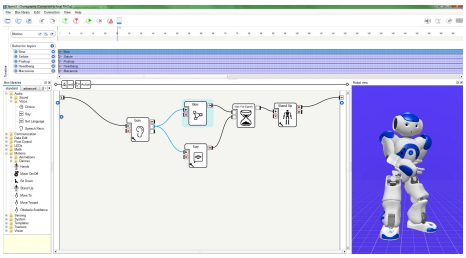

Figure 1: The 'bow' animation as seen in the Choregraphe interface

The push-ups are a combination of animations which first makes the NAO move to the ground with his arms forward, following an arm motion which looks like actual push-ups. The 'head bang' animation moves the right arm of the NAO upward and the head joint move forward and backward eight times. Finally, the salute motion makes the Nao salute like a soldier by raising its arm near the head and then quickly moving it aside. All implemented motions have matching sound files. For example: When the Nao starts doing push-ups, it shouts: 'Yes, Drill Sergeant!'.

As for speech recognition, the first thing that was worked on was sound localization. Taking inspiration from the built-in box called 'sound loc.', we derived a new box that would allow for the NAO to turn towards sound. Sound loc. makes the robot turn its head towards incoming sound, so we edited the box's script in order to try to relocate its body, instead of the head. Of course, turning a whole body is more complicated than just the head, which one can modify using just one variable, called angles. It was soon decided to move on and focus on speech recoginition instead.

The built-in speech recognition box in Choregraphe was used in a manner where every separate motion had its own speech recognition box. All those boxes were simultaneously connected from the input, resulting in moves that would initialize at the same time and the Nao wouldn't listen properly for to all the commands. The problem here was that the built-in speech recognition box isn't actually a check for sounds, like an if-statement. The box always initializes, meaning all commands would play as soon as one of the speech recognition states succeeds. A possible fix to this problem was to store every motion with its recognition box into a layer separately. After finding out that it's possible to have multiple simultaneous layers of behaviours we tried using this together with the build in speech recognition, but to no avail.

Another attempt was trying to integrate Googles speech to text to Choregraphe which could be used to return a string of any speech and then use this together with a switch statement box to execute the commands. The crg file we got from Aldebaran's community site claiming to implement this functionality didn't do anything, the NAO would not recognize anything that was said. Whether this was due to the noisy workspace or anything else, remains unclear. After finding out that its possible to have multiple simulataneous layers of behaviours we tried using this together with the build in speech recognition but to no avail.

Lastly, for speech recognition, we have also tried to write some code that would record speech into a .wav file, put the recorded sound from the .wav file through a fourier

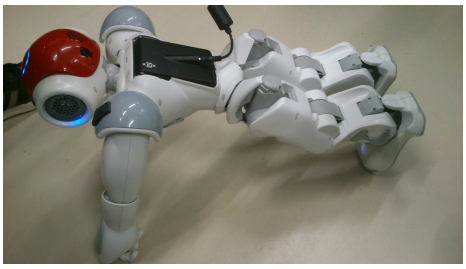

Figure 2: A demonstration of the push-up position in action

transform algorithm called the Fast Fourier TransformCooley and Tukey (1965), and then use the frequencies that would be most occurring to discern between a few commands. However, due to inexperience with the Python language, writing scripts for the Nao and lack of time, we were unable to produce any scripts that wouldn't conflict any other existing boxes.

Results

For as far as this 4-day lasting research goes, it is easy to conclude that the hypothesis has been rejected, since the implementation wasn't able to consider all five motions simultaneously. All the motions, including their activation commands ended up working separately and can be demonstrated by enabling the layer of the desired motion and disabling all the rest. Combined, they caused too many problems to use this final implementation. Every movement/motion separately, including its sound file, works, and can be called upon by enabling the right layer and offering the right keywords in the form of speech to the NAO. The push-up animation stands out from the other motions, because the animation can also be cancelled whilst playing. This is done by, in this case, saying 'stop' or 'enough' to the robot.

Conclusion & Discussion

Returning to the initial research question: How can a NAO efficiently execute different motion commands to its relative speech activation, we can conclude that this can be done by creating the movements and binding them to a speech recognition box relatively easily. However, the real challenge involves being able to take more than one possible command into consideration and this is an issue this research cannot resolve yet.
To make multiple commands work together simultaneously, there needs to be an other way coverting speech to text, in order to work with this text as a string of letters instead. Another improvement would lie in the NAO's wifi connection. We've encountered some practical issues, like the fact that the NAO's connection was not always stable, which delayed our progress immensely. Also, the room we were working in was

very noisy, which caused the NAO's microphone to overload and disfunction, making the speech recognition harder. We could not change rooms due to being restricted to a specific access point whose signal barely reached neighbouring rooms.

Suggestions

Future students that would be interested in the subject of speech and sound recognition, using NAOs, should first get acquainted with Python (or C++), as well as how to work with and write code for the NAO's, prior to starting on this project. Four days are not enough to execute a similar project, without having the needed resources and skills. Working in Choregraphe only, was not enough for this project.

Appendix

- Choregraphe v.1.14.5 (Including Naoqi)
- Toshiba i5 Laptop (Windows 7)
- Dutch Nao Team wifi connection (RoboLab, Science Park, Amsterdam)
- Aldebaran NaO (red)

References

Cooley, J. W., and Tukey, J. W. 1965. An algorithm for the machine calculation of complex fourier series. *Math. of Comp.* 19:297–301.

Dominey, P. F.; Van Der Zant, T.; Lallee, S.; Jouen, A.-l.; Hinaut, X.; Weitzenfeld, A.; Van Hoof, H.; and Chacón, J. D. Cooperative human robot interaction with the nao humanoid: Technical description paper for the radical dudes.

- Aldebaran Robotics, Softbank Group 2014. *NAO animation, 'how to'*. Available from: https://community.aldebaran-robotics.com/resources/tutorial/nao-animation-how-to/