Professional Bachelor Applied Information Science

Active Learning strategies on a Pepper robot

Michiel Gabriels

Promoters:

Brigitte Krenn & Stephanie Gross — Austrian Research Institute for Artificial Intelligence

Bram Heyns — PXL University College

Bachelor paper Academic year 2018-2019

Professional Bachelor Applied Computer Science

Active Learning strategies on a Pepper robot

Michiel Gabriels

Promoters:

Brigitte Krenn & Stephanie Gross — Austrian Research Institute for Artificial Intelligence

Bram Heyns — PXL University College

Bachelor paper Academic year 2018-2019

Acknowledgements

After three months, I can finally say that the end of this road has been reached. It has been an adventurous one, with many challenges along the way, and I could not have done it without the help of others.

First of all, I would like to thank OFAI for giving me this huge opportunity to do an internship abroad. Specifically, I would like to thank Stephanie Gross and Brigitte Krenn, my promoters, who have guided me along the way. Furthermore, I would like to thank Bram Heyns for his guidance and valuable feedback while writing this thesis. In addition to him, I would also like to thank Saskia Schoefs for always being friendly and helpful whenever I had any language specific questions.

I would also like to give a special thanks to Friedrich Neubarth and Matthias Hirschmanner, who have always been very welcoming when I needed any help, and Christiana Tsiourti, who was always very friendly during the sessions with the Pepper robot. Furthermore, I would like to thank Clara Haider for taking the time to explain some very specific and math heavy concepts to me. But I would also like to give a special thanks to Paolo Petta, who has taught me a lot about Vienna and life in general, and would always bring laughter whenever he came into the office.

My parents and sister were always supportive of me throughout my internship, so I would like to give a big thanks to them too. I would also like to thank my friends, whom I could always count on whenever something was not clear.

Abstract

The Austrian Research Institute for Artificial Intelligence (OFAI) does ongoing research on an incremental grounded language learning system that is integrated with a Pepper robot. This system has an active learning extension to help the learning process of the robot by having it ask for feedback in the form of non-verbal communication, like pointing. It will decide for itself to ask for feedback when it thinks it needs more information on certain actions or objects. This is inspired by research on child language acquisition, whereby a child will also use gestures like pointing to ask for new information whenever it is unsure about something. The basic model for this learning system is already established. However, there is still room for improvement on this existing model. By tweaking the parameters, more information could be gathered about whether or not the feedback given by a user should receive more weight than it is given now in the learning process of the robot. Other parameters include the time that is required to learn all the objects present in the scenario and whether or not to use vocalization like “uhm” to make it clearer to the user that the robot needs more feedback.

The thesis describes the methodology used for acquiring the information from these experiments. A hypothesis is formulated, based on the research done on the effects of a specific parameter, after which this hypothesis is tested on the existing system. This system uses the Robot Operating System (ROS) to facilitate the layer between the hardware of the Pepper robot and the software developed for multi-modal word learning. The Pepper robot has its own voice recognition software, but because this is not sufficient, Google’s cloud speech recognition software is used instead. The model uses the Action Verb Corpus made within OFAI, which has been developed using multi-modal data of twelve humans conducting in total 390 simple TAKE, PUT and PUSH actions.

The thesis also briefly touches on the subject of how a student that has no prior research experience integrates in an existing (international) research group working on a subject that is new to the student. This includes the student’s personal findings, how his social and soft skills have improved in an international setting, how this can be used during projects later in life and how prior experience in practical programming can be useful when integrating into a research group. It also includes the general workings of this particular research group and the cooperation with an adjacent research group at a technical university.

Table of contents

Acknowledgements ii
Abstract iii
Table of contents iv
List of figures vi
List of abbreviations vii
Introduction 1
I. Traineeship report 2
1 About the company 2
2 Student integration 2
2.1 Research 2
2.1.1 General 2
2.1.2 International 2
2.1.3 Practical background 3
II. Research topic 3
1 Background info 3
2 Research question (hypothesis) 4
3 Research method 4
3.1 Introduction 4
3.2 Setup 5
3.3 Context 6
3.4 Pepper specifications 6
3.4.1 Cameras 7
3.4.2 Movement (motors) 7
3.5 Observations 9
3.5.1 Pointing mechanism 9
3.5.2 Vibrations 10
3.5.3 Bad lexicon 10
3.5.4 Speech recognition 10
3.5.5 Child vs. Pepper robot 11
3.5.6 Looking while pointing 11
3.6 Ideas (based on observations) 11
3.6.1 Idea 1 12
3.6.2 Idea 2 12
3.6.3 Idea 3 12

3.7 Challenges (new technologies) 13
3.7.1 C++ 13
3.7.2 ROS 14
3.7.3 Python 14
3.7.4 Perl 14
4 Results 15
4.1 Pepper tweaks 15
4.1.1 Solution 1 – Disable object tracking 15
4.1.2 Solution 2 – Two-PC setup 16
4.1.3 Solution 3 – Add a sound 16
4.2 Learning system 16
4.3 Experimental sessions 17
Conclusion 18
Bibliographical references 19
Appendices 21

List of figures

Figure 1: Action Word Corpus: VR setup (left) with instructions that tell the person what action to perform and which sentence to speak out loud (right). 4
Figure 2: Pepper robot setup, with the Google speech recognition (top left), the robot's camera view with the object tracker (bottom left) and the visual simulation (right). 6
Figure 3. The Pepper robot's camera specifications, as seen from the top and from the side. 7
Figure 4: The Pepper robot's possible movement parts. 8
Figure 5: The implemented fixed point in the shoulder (right) is used instead of the existing variable point (left). 8
Figure 6: Top-down view, with the Pepper robot as the circle, the objects as the red shapes and the (imaginary) line from the pointing hand to the objects. 9
Figure 7: The C++ code responsible for the pointing mechanism of the Pepper robot. 13
Figure 8: Simple check to see if the values used in a subtraction are actually initialized. 14
Figure 9: The "stop interaction" signal that has to be send to the Pepper robot's system to have it stop pointing (and looking) at an object. 15

List of abbreviations

AI	Artificial Intelligence
OFAI	Austrian Research Institute for Artificial Intelligence
ROS	Robot Operating System
VR	Virtual Reality

Introduction

Together with colleagues from the Technical University Vienna and Tufts University Boston, the Austrian Research Institute for Artificial Intelligence (OFAI) organized an AAMAS workshop on cognitive models in Montreal together with the Tufts University regarding active learning of a Pepper robot. AAMAS is the International Conference on Autonomous Agents and Multiagent Systems, which focuses on the interaction between a robot learner and a non-expert human teacher using verbal and non-verbal communication. OFAI presented their own work there, inspired by the information seeking strategies found while studying behaviour of children in their early childhood [1]. These strategies were implemented into a learning system that is behind the Pepper robot to simulate the same learning behavior of a child, for example pointing at an object to gain more information about that object [2]. One of the questions regarding these learning strategies was how the system could be improved, so that a user could better understand what the Pepper robot was trying to communicate without it using verbal language, just like a child would in their early childhood. Before implementing those improvements, experiments had to be set up. The main setup was already defined by the research group, more specifically the physical environment in which the Pepper robot would have to operate. However, many parameters had to be tweaked in order to improve certain actions performed by the robot and the interactions with the user, which is what this thesis covers. These parameters were changed throughout this whole improvement process, as new observations came about during each of these experimentation sessions. Many of the observations came from the pointing mechanism of the Pepper robot. Because this was all closely tied together with the research on the behaviour of children in their early childhood, pointing was a very important part of the Pepper robot's function. Those new observations brought with them new ideas, which were either implemented and tested or thrown out depending on the possibilities and the intend of those ideas. The ones that were implemented brought with them new parameters, which in turn also had to be gradually tweaked. This continued until a suitable version of the learning system was developed for the workshop in Montreal.

A number of challenges occurred during this whole process, as a new programming language was learned in order to further increase the ability to adapt the learning system to new ideas and fixes for observed problems. On top of that, the student had to adapt to a completely new environment: from a development environment to a research environment. With only a practical background in programming, this was challenging but not impossible. This was all done internationally, giving the level of adaptation another dimension by the change of the physical environment and language environment. The findings of this process were also recorded, giving helpful tips for future students that might find themselves in a similar situation.

I. Traineeship report

1 About the company

The Austrian Research Institute for Artificial Intelligence (OFAI) is located in the heart of Vienna, Austria. It is part of the Austrian Society for Cybernetic Studies (OSGK) [3]. OFAI uses many different methods to solve problems for their partners (international and national organisations, companies, universities, research institutes, … [4]): machine learning, data mining, language and speech recognition, neural networks, intelligent software agents and other software technologies related to problems in one of their research projects [5]. Many of these software technologies are developed within OFAI, depending on the needs of each partner. OFAI currently has 28 people working for them. Most of them are computer scientists and linguists, but also graduates from different universities. There are also nine scientists employed at OFAI, coming from the universities where they work at as professors [6]. OFAI also houses interns whenever there is an opportunity or a need for it.

2 Student integration

2.1 Research

2.1.1 General

Due to the nature of this internship, the student had to adapt to a new work environment. This work environment was mostly focused on research, which is fundamentally different from a regular internship focused on developing a product (optionally with a new programming language). One of the key differences is situated within the day-to-day structure of the internship. While developing a product, there is generally some sort of checklist to work through each day in order to meet certain deadlines, whereas in research there is a lot more leeway to obtain the desired results. Of course, there are also deadlines to be met in research, but the way towards those deadlines is a lot less clear. This is mostly because there is a general hypothesis of what the outcome of a certain research subject will be, but the results can differ significantly depending on how the experiment is set up. There is still some development work to be done (again, depending on the experiments and/or research subject), but this is different from a regular development process in a company. The actual development in a research context is much more focused on solving a single problem, without having to worry about the broader context an implementation might be used in. However, this all depends on the experiments being run and the precise nature of the research subject. Theoretical research involves significantly less development time or even none at all, whereas practical research will have at least some development time to set up an environment for experimentation or to gather data for these experiments.

2.1.2 International

OFAI is located in Vienna, Austria, so the student had to adapt to an international work environment on top of the work environment in research. Because OFAI cooperates with international partners, non-German speaking people can easily speak English to colleagues. This was also the main language used throughout the internship. Most information available on the research subject (online sources, papers both in-house and out-house, …) is in English, as this is the main language in the world of computer programming.

2.1.3 Practical background

Due to the student's practical background, there was a limitation on which research papers could be read relating to the research subject. As the student does not have a university or PhD level background, most math in these research papers was far too advanced. This is a hindrance when going into a work environment revolving around research, especially when the research subject is fairly new. There are not a lot of practical examples around in these subjects, which makes it harder for a student with a practical background to visualize the problems that are being researched. Placing the research subject in a general context can also be hard, depending on the type of subject. Most research papers on these research subjects are focused on a specific hypothesis, building on a lot of other (older) papers that are even harder to understand for a non-university or non-PhD level student. The key points for the student here are to ask whenever something is not clear and to gloss over any difficult math in the research papers. This is not always possible, but in most research papers it is not too difficult to skip over the math and take away the key points presented in it. The most important part here is to always read the abstract and conclusion first if the research paper is heavy on math, because it makes it easier to follow the general outline when reading through it.

II. Research topic

1 Background info

OFAI currently uses an incremental grounded language learning system on a Pepper robot [2]. The Pepper robot is a humanoid robot that is able to recognize faces and basic human emotions, and is developed by SoftBank Robotics [7]. The learning system allows the Pepper robot to learn word to object and word to action mappings by observing a person performing different tasks. These tasks include three actions: TAKE, PUT and PUSH. This is based on how children learn new actions and objects: by observing a person, like their parents, performing different actions and using certain words with those actions [8]. In order to develop such a learning system, a respective data set (the Action Verb Corpus [9]) was collected at OFAI.

Simply put, a Virtual Reality (VR) headset and a microphone are used to record the actions (a set of instructions is visible inside the VR headset) of twelve participants, together with a verbal description of what the person is doing, as seen in Figure 1, quoted from [9]. The objects and the person's hands get tracked using special software (see [10] and [11]). With the help of this object and hand tracking, the actions get identified, e.g., when the hand touches an object and the object is no longer on the table, it is assumed that the action TAKE has been performed. The verbal utterance, that represents the verbal action, is transcribed and aligned with the visual action. The resulting, manually corrected data set has been used for experimenting with different learning models. The most promising one was then adopted for implementation on a Pepper robot.

For the life system (the whole cycle of segmenting speech, pairing it with actions, etc.), the technologies used for object and hand tracking are comparable to the ones used for data collection in the Action Verb Corpus. In addition, the human's utterance is transformed into text by Google's speech recognition software [12]. After the actions have been identified and the text from the speech recognition software has been segmented, the actions are aligned with these utterances. For example: the tracked object and action with the hand from the object tracking software get segmented into the action "TAKE CAN" and the utterance "I take the can" has been aligned with that action. The learning system implemented on the Pepper robot uses this kind of visual action and utterances pairs to incrementally learn mappings between words and the reference to those words (the objects and actions).

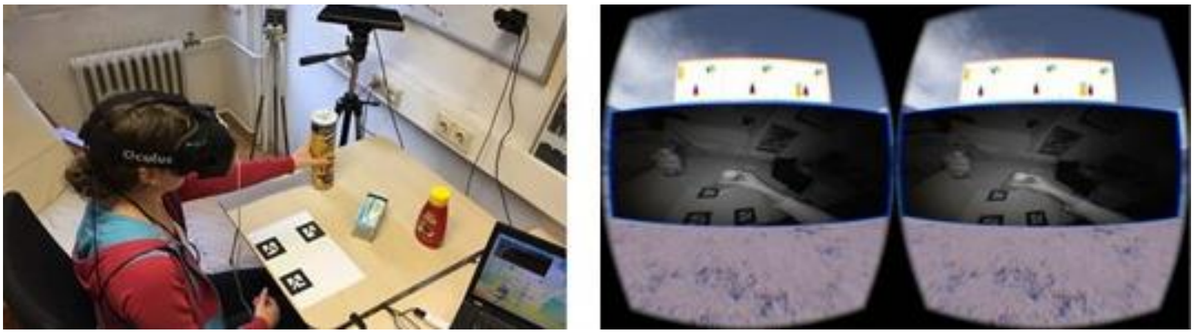

Figure 1: Action Word Corpus: VR setup (left) with instructions that tell the person what action to perform and which sentence to speak out loud (right).

2 Research question (hypothesis)

The idea for an extension to this system is to introduce Active Learning. The Pepper robot would, under certain conditions, interrupt the user and request for more information by pointing at an object. This idea comes from the research on child language acquisition, where it is clear that children use gestures like pointing to try and get more information about things they do not understand or they feel like they are missing information on.

There is some experimentation being done with the model's parameters, such as the duration of time that the model goes into active learning mode. However, there are also other interesting experiments to be done regarding these parameters: what differences are noticed when changing the parameters regarding distance between objects, and how do different communication approaches (looking at the user, saying "uhm", …) for seeking more information compare to each other? In general, how can the current system be improved to make it clearer to the user when and how the robot is asking for feedback?

3 Research method

3.1 Introduction

To find an answer to this question, the Pepper robot and the environment around it have to be set up properly. This includes the real-world environment, but also the background system environment of the robot itself (camera resolution, object tracking, etc.). Next, the observations of working with the Pepper robot have to be documented. This is important to get an idea of how a user that has no prior knowledge or working experience with the robot, would react to certain actions performed and decisions made by the robot during a session. It also helps to have a list to brainstorm for new ideas that could possibly improve these interactions with the user and the different objects. These new ideas would also have to be put into a list and verified or tested to see whether or not they are viable options or roads to consider. This cycle repeats itself, whereby ideas implemented and/or tested could lead to new observations, which in turn could lead to new ideas. Any challenges that come up during these experimentation sessions, brainstorming sessions and/or idea implementations have to be noted down too.

Specifically, a typical experimentation session starts out with setting the whole system up: placing the Pepper robot in front of the table, putting the objects and the object tracking marker on the table, and running all the necessary scripts (object tracker, Google speech recognition software, etc.). Then, a normal session first starts to test whether everything is working correctly. This is usually after the learning system has been altered to fix a bug or when a new feature was added. During this, a new bug is found or the solution for the old bug is not working. At first, it has to be checked whether the new problem is because of a bug in the Pepper robot's system or in the learning system. If it is in the Pepper robot's system, it is usually fixed during the same experimentation session, and the cycle can start over again. If it is in the learning system, however, the code has to be looked over closely to see if the bug can be located. Usually this is the case, but solving it would be too complex to do it during the experimentation session. The experimentation session then continues for a little bit longer to see if any other bugs occur, again going through the whole cycle from the start. Afterwards, the bugs or problems are reported to the person that implements the solutions into the learning system. These implementations are usually done within a day or two, so the next experimentation session can be planned fairly soon after the other experimentation session.

3.2 Setup

The Pepper robot's camera is currently set up with a 640x480 resolution. Given the object tracking, this was the best resolution to work with, without having too low a framerate. It is important that this framerate stays at a constant 15 frames per second, as lower than this would cause delays in the action being performed and the speech, because the camera would not be able to keep up (and along with it the object tracking). The camera is set up in a way that the table, with on it the three objects (a ketchup bottle, a Pringles can and a lunch box), is fully in view, as seen in the bottom left of Figure 2, quoted from [2]. The robot also experiences micro stutters. This element is taken into account when detecting whether or not an object is moving. The safety mechanism that triggers when an object is too close to Pepper, has also been disabled, because it sometimes detects false positives. These false positives would otherwise cause Pepper to not fully point to an object, misleading the user training the robot. The language of the learning system is set to American English, as this gives the best results from the Google speech recognition software in an international setting (in this case, an international research group). The German language was also tested, but this had worse results and was not really useful in an international context. French and Dutch were also tested, whereby the French language recognition was the worst of them all. This is because many French actions and words are a combination of multiple words, as opposed to German and English, where they are only one word. The Dutch language recognition worked pretty well, but was once again not useful in an international context.

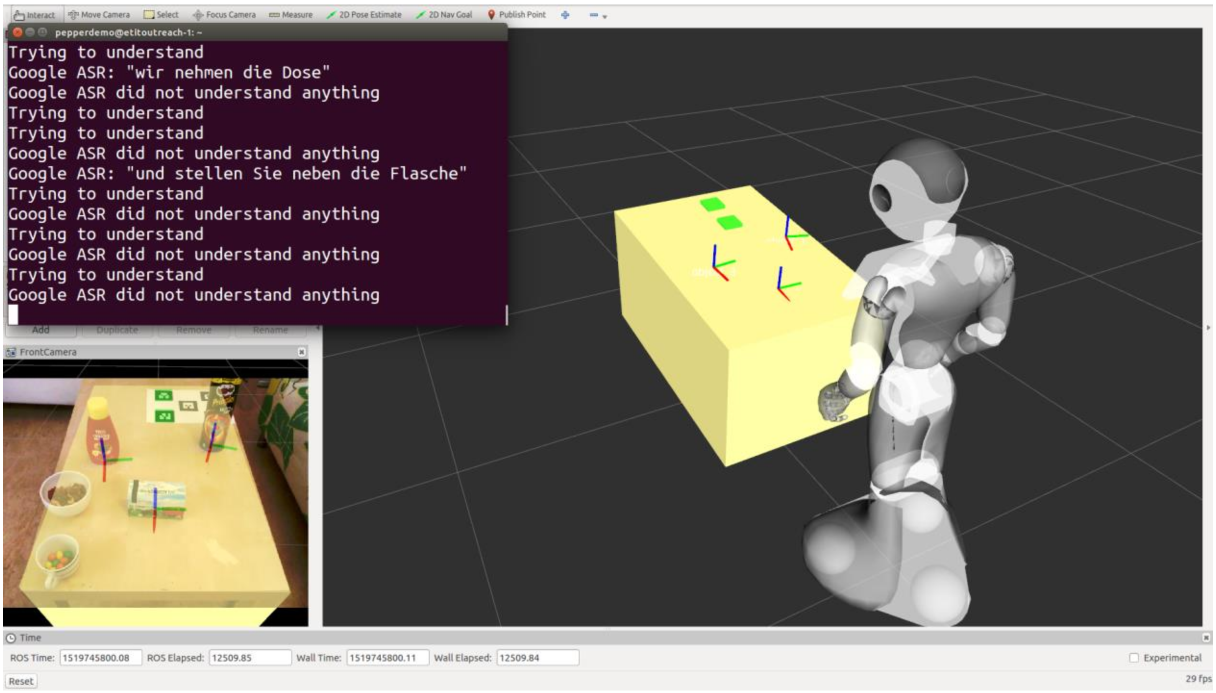

Figure 2: Pepper robot setup, with the Google speech recognition (top left), the robot's camera view with the object tracker (bottom left) and the visual simulation (right).

3.3 Context

As previously mentioned, the Pepper robot has to be seen as a child in their early childhood. This has some major constraints to what is possible. Some ideas that might come up when reading further are:

- Let Pepper say the phrase “What is this?” while pointing at an object.
- If the descriptions of the objects are present in the system (from existing 3D models), let Pepper describe the object it is pointing at.
- When pointing at an object, let Pepper say whether the object is on the left, right or in the middle. As an extension, …
- … let Pepper say whether the object is in the back, in the front or in the middle.

A child in their early childhood will not have developed full sentences yet. As such, the child (and subsequently the Pepper robot) will not ask information on an object by utilizing these full sentences. Instead, it will point and look at an object. This is important to keep in mind, as the whole learning system is based on these constraints.

3.4 Pepper specifications

Extending on the context, it is important to know what specifications the Pepper robot has. These specifications also have a major impact on constraints, as it limits the ability to fully mimic a child in their early childhood. Mainly the body of the Pepper robot is the constraining factor, as it is the most important part for a human to look at when trying to understand or gather information from another human: the way the eyes move, where someone (or something) is looking at, how the shoulders are positioned, what the hands are doing, which finger is pointing at what, etc. These are all examples of ways a human might receive or conceive information.

3.4.1 Cameras

One of these specifications is the camera, multiple cameras in this case. The cameras have a big impact on what objects are visible at the same time and how good the framerate is, which is important for the learning system to get the right actions at the right time. This was briefly mentioned in the setup: the Pepper robot's cameras use a resolution of 640x480 and a framerate of 15 frames per second. They are, however, capable of a resolution up to 2560x1920, but this is only at a framerate of 1 frame per second, which is way too low. At 640x480, it is capable of a framerate of 30 frames per second, but as there is a lot of other software running at the same time, it is only going up to 15 frames per second.

The field of view on the x-axis is 55.20°, which is just enough to fit in a small table in width. However, the field of view on the y-axis is only 44.30°, as the current setup uses the forehead's camera. This means that the Pepper robot's head has to be all the way down for it to see the full table in length. This is one of the constraints, as the Pepper robot would have to look fully up to the person if it would want to provide any information in the way of blinking or squinting the eyes. A child's eyes are more visible to a human when it is looking down, as it has a much bigger field of view on the y-axis. This all can be seen in Figure 3, quoted from [13].

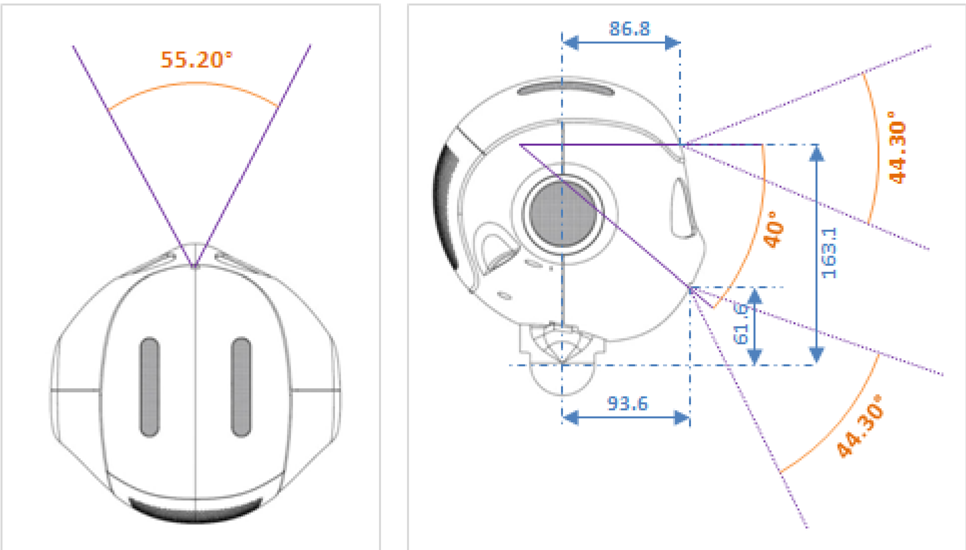

Figure 3. The Pepper robot's camera specifications, as seen from the top and from the side.

3.4.2 Movement (motors)

Another major specification is the movement of the Pepper robot, facilitated by different motors. Many interactions between the robot and the human are constricted by which parts of the robot are able to move, and in what way. The most important parts in this case are the arm and the hand. The movement of the arm and the hand is facilitated by a combination of multiple parts (motors): the shoulder, the elbow, the wrist and then the hand. As seen in Figure 4, quoted from [14], the fingers do not have any individual motors. This makes pointing less accurate, because a human is not able to precisely see what object the Pepper robot is pointing at. Pointing is done by opening the hand and moving it in the direction of the object.

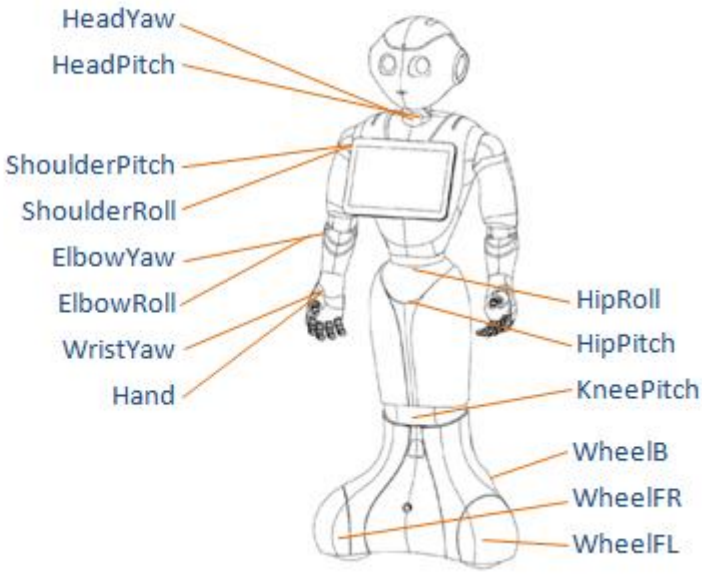

Figure 4: The Pepper robot's possible movement parts.

Because the shoulder moves together with the arm, it creates a problem for pointing towards a specific point. The calculation behind this needs a fixed origin, which the Pepper robot does not have. This was solved by creating a fixed point in the shoulder, that corresponds with the already existing (but moving) point of the shoulder, as seen in Figure 5, quoted from [15].

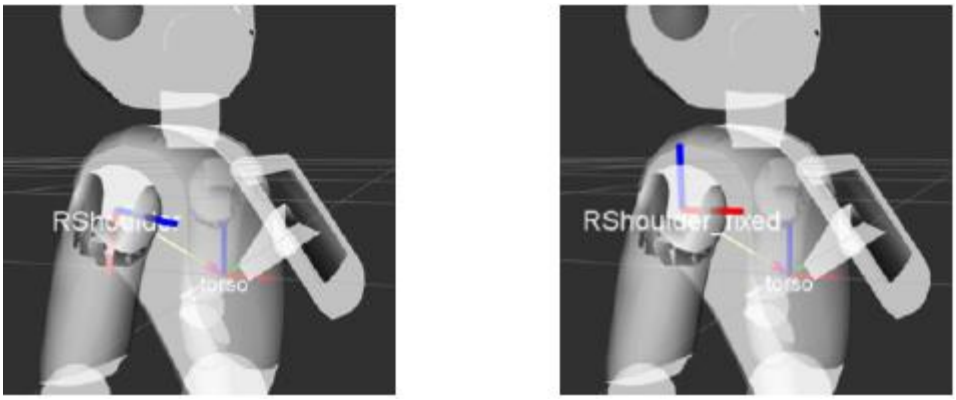

Figure 5: The implemented fixed point in the shoulder (right) is used instead of the existing variable point (left).

The other important moving part is the head. The head can go up and down, but also sideways. This is useful for giving more information to a human, as it can simulate looking at an object in combination with pointing at that object.

Potential other moving parts are the hip and knee, which help simulate the Pepper robot leaning forward or backward. The three wheels (two in front and one in the back) are also able to move, which can help move the Pepper robot around in the physical environment. However, these features are not applicable to the research in this thesis, thus they are not used.

One of the downsides of having these motors move the different parts of the Pepper robot around is that they overheat very easily. Whenever you start a session where the robot has to point and/or look at something or someone, you have to "stiffen the chains". These have to be released after every session, otherwise the Pepper robot starts complaining about overheating. If the loosening of the chains is forgotten, a cooldown period starts where the robot is not able to move any parts for a small period of time. Luckily, the “stiffening” and “loosening” of the chains is as simple as pressing a button, so it is just a matter of not forgetting.

3.5 Observations

3.5.1 Pointing mechanism

One of the main observations found while working with the Pepper robot is that the pointing mechanism is not optimal. Because the robot can only use all fingers at once or none at all, it is hard to know which object it is pointing at. Generally, following the line that the hand is pointing at is the only way of visually figuring this out. This brings up another problem where if the object is close to another one, it is hard to know which of the two the Pepper robot is referring to. A solution for this could be that the robot bends forward along with the pointing of the hand, making it almost touch the object it is referring to. However, this would move the camera around too much, confusing the object tracking system and thus confusing the Pepper robot on the position of the objects. The more optimal solution for this particular situation is to try and keep the objects as far apart as possible. This way, it is a lot clearer which object Pepper is pointing at. This is adjustable by increasing or decreasing a parameter, which defines the minimal distance of an object to the nearest object on the x-axis. Increasing this parameter would mean that objects have to be further apart before they would be considered “pointable”. On the other hand, decreasing this parameter would mean that objects can be a lot closer to each other for them to be considered “pointable”. The optimal distance for this is subjective, as it completely depends on each user whether or not they find it clear enough which object the Pepper robot is pointing at. However, if objects are positioned in a certain way where another object is on the same straight (imaginary) line from the hand to the referring object (see Figure 6), it would still be unclear which of the two objects the Pepper robot is referring to, even if they are far enough apart.

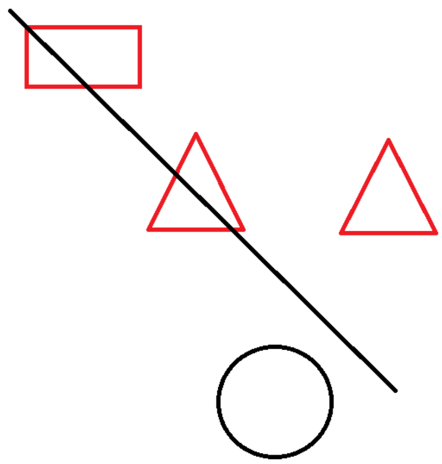

Figure 6: Top-down view, with the Pepper robot as the circle, the objects as the red shapes and the (imaginary) line from the pointing hand to the objects.

Another observation that was found during the sessions with the Pepper robot was that if the user waited too long to say something during pointing, the learning system would time out. The problem with this was that, because the learning system did not send any signal back to the Pepper robot when it timed out, the robot would keep pointing. This made it confusing to the user, as any actions would be discarded: the Pepper robot expects a word or a small sentence to tell it what object is being pointed at, but the learning system expects an action coupled with a sentence (like it normally would). Hence, the robot only sends speech to the learning system, and because the learning system detects that only speech was sent, it thinks it is only a speech chunk and thus disregards it. As a result, the whole system is stuck and has to be restarted. There was a parameter to change this time out threshold, but increasing it did not give the desired effects. Instead, it just delayed the whole system. Luckily, the solution for this was very simple and already present in the current system. The learning system already had a check on whether or not the system timed out, but it did not send any response to the Pepper robot to indicate that it should stop pointing. Hence, the Python code that drives the Pepper robot was not able to stop it from pointing. A simple addition to the learning system made it so that the signal was sent to the Python code, which in turn already had everything ready to stop the Pepper robot from pointing like it normally would.

3.5.2 Vibrations

One of the other observations found was that the vibrations of the Pepper robot (or the table) varied quite substantially. This causes problems with the object tracking, as the Pepper robot thinks the objects are moving when they are not. If people are talking at the same time as this is happening, it would result in a bad lexicon and confuse the whole learning system. The solution for this was to adjust the parameters accordingly: not too large, but also not too small. One of these parameters is the maximum distance from an average point that an object could shift. Increasing this parameter gives more leeway in case the vibrations are high, as it would take a bit more movement of the object for it to be considered in a state of "moving". Decreasing this parameter results in the opposite: it would take a lot less movement for an object to be considered in a state of "moving", which basically means the system is a lot more sensitive to (small) movements. In the same way, the other parameter defines the minimal distance from an average point to disregard a certain data point, aka an object. The more this parameter increases, the more likely small vibrations of the object or the Pepper robot would be disregarded as "noise" and thus ignored. Some days the vibrations are a lot higher, but other days it is quite low. The reason for this is unknown, but probably has to do with the hardware of the Pepper robot itself, which is not adjustable.

3.5.3 Bad lexicon

Another observation found was that in case the Pepper robot was certain it had learned an action and object combination (like "take bottle", which is said out loud by the Pepper robot), there was no way of telling the learning system if what it had learned was actually correct. In case it did learn something wrong, the user teaching the Pepper robot would not know if it could correct this behaviour. A possible solution for this was to have the user teaching the Pepper robot to confirm ("yes", or "no") whether or not what it had learned was correct. This response would have more weight in the lexicon, just like the response after the pointing of the Pepper robot would have more weight. It would speed up the learning process as a whole, because the user would not have to start all over again by trying to get rid of that particular word and action pair by moving the objects around and saying phrases.

3.5.4 Speech recognition

The Google Speech Recognition was found to be the best speech recognition at the time of writing. This was tested and confirmed by the person in charge of the Pepper robot, as they had compared it to Amazon's speech recognition and other more obscure speech recognition software. However, this is still not the most optimal solution. Many words and phrases are not recognized correctly, which is mainly due to the accents of the people using the Pepper robot, none of which are native English speakers. More so, even for someone with a more natural English accent, Google's speech recognition software confuses words that strongly resemble each other. The other language package, German, was previously found to be not as good as the English one, even for a native German speaker. As a result, adjustments had to be made to certain ways of speaking, which would have to be noted down and set as “rules” for people to follow. One of these adjustments is the use of present and future tense. The phrase “I push the bottle to the edge of the table”, results in Google’s speech recognition software to recognize it as “I pushed the bottle to the edge of the table”, which is not quite right and leads to the Pepper robot thinking the action “PUSH” is equal to the word “pushed”. To account for this, people have to say “I will push the bottle to the edge of the table”, so the word “push” is recognized correctly.

3.5.5 Child vs. Pepper robot

After numerous sessions with the Pepper robot and after talking with the people in charge of the robot (and the person that has to give the workshop), it was found that a big problem with teaching a robot like you would teach a child in their early childhood is very hard. A child does not simply say “yes” or “no” to ask for confirmation and also does not explicitly show it needs more information on an object. Many small nuances, like their facial expressions, are very important. These are easily distinguishable by adult humans from other facial expressions, but very hard for a robot to understand and/or generate (as mentioned in the Pepper specifications). This also goes for the person teaching the Pepper robot: an adult human teaching their child in their early childhood would do things like laugh, smile, clap, etc. to show the child that it said or pointed at a correct object. The Pepper robot would have to be able to track all these things at the same time, which is currently not possible.

3.5.6 Looking while pointing

One of the new features of the Pepper robot is its ability to look at the object it is pointing at. This makes it clearer to the user which object the Pepper robot is referring to when needing more information. The key problem with this is that the robot would have to move its head to look at the object, making it seem like all the objects are moving at the same time. This would be problematic if the users (or other people in the room) are talking while this is happening, as the learning system would associate those sentences with the movement actions of all these objects. The first solution for this was to disregard any actions that may occur while the learning system is in interaction mode. Similar solutions were also brought up, but the final solution for this is an out-of-the-box idea: disabling the object tracker as soon as the Pepper robot starts moving its head and enable the object tracker again when the head is back in its normal position (looking down at the table). This was a lot easier to implement than the other solutions.

3.6 Ideas (based on observations)

The following list contains ideas for possible experiments that were thought of throughout the internship. These ideas were presented to the person in charge of the Pepper robot to confirm the possibilities (based on previous experimentations).

These ideas are based on the observation that it is hard to know what Pepper is pointing at:

1. After pointing at an object, let Pepper wait and use the first response it detects.
2. After pointing at an object, let Pepper wait for the user to lift an object, and confirm (say “yes or no”) whether or not that was the object it was referring to when pointing.
3. Have Pepper use either the left or right hand, depending on whether the object is on the left or right of the table.

3.6.1 Idea 1

The first idea (letting Pepper wait and use the first response it detects) was already somewhat integrated into the learning system. The Pepper robot points at an object and then waits for the first speech input it receives. There are, however, some problems with this approach. Because it takes the first speech input it receives, it also takes in noise, which is mainly background chatter that is picked up by the microphone, and thus by the Google speech recognition software. In addition, if the learning system has segmented a spoken sentence into two different sentences (because of a pause that was just long enough for it to be considered a second sentence), it would only take the first sentence when the action has already finished. This is usually an incomplete sentence, and thus with wrong information for the learning system. The solution for this problem was to implement a new parameter that defines how many seconds after the pointing interaction the learning system has to wait before taking in a sentence. This allows for the user to think before telling the Pepper robot what the object is.

3.6.2 Idea 2

The second idea (letting Pepper confirm the object that is being pointed at) is a lot more complex. The learning system behind the Pepper robot only "watches" the object it is pointing at, meaning that it disregards any movement of the other objects while pointing. This, of course, makes it not possible for the Pepper robot to know if the user lifted up an object that is not the object it is pointing at. However, it would be possible as an addition in the future, but requires some reimplementation. The underlying assumption is that the object lifting would be scrapped; the user would only have to say "yes" or "no" to confirm. However, the problem with this is that, as previously mentioned in the context, the Pepper robot should be seen as a child in its early childhood, so one should refrain from hardcoding any responses.

3.6.3 Idea 3

The third idea (letting Pepper use either the left or right hand), after consulting with the person in charge of the Pepper robot, is not possible in its current state. The Pepper robot is able to do it, as the implementation for that is already present, but the current solution with the new pointing mechanism is sufficient enough (as mentioned at the end of this paragraph). A temporary solution was to just increase the parameter at which the learning system would include the objects. This parameter defines how many centimetres apart the objects have to be before the Pepper robot instantiates its pointing mechanism. However, due to the position of the objects, the parameter would have to be increased too much, leading to lesser pointing (the objects would constantly have to be at the edge of the table) and more confusion to the user. The second temporary solution was to come up with a mathematical solution to check whether or not two objects were on the same imaginary line from the hand that is pointing. A simple way is to just get the prediction of the position of the hand when it starts pointing to an object. This, however, is not possible, as the hand moves iteratively until it detects the object it has to point to, so the stopping position of the hand is not known beforehand. A final solution for this was to take the position of the head, add a couple of centimetres along the x-axis (to simulate the actual point from the shoulder instead of the head, which is more accurate in this situation) and then calculate whether or not the object is pointable. The position of the head has to be sent by the Pepper robot itself, as this was not already present in the data being received by the robot in the learning system.

3.7 Challenges (new technologies)

3.7.1 C++

The different mechanics (like pointing) of the Pepper robot are written in C++. The reason for this is that the robot uses ROS (Robot Operating System), which itself is also written in C++, along with Python and Lisp. In general though, C++ was chosen because it is very good for low-level programming and when heavy mathematical calculations have to be performed. An example of this C++ code is seen in Figure 7, which is an implementation of the subject referenced in [15]. This piece of code is responsible for the pointing mechanism of the Pepper robot, specifically the calculation of the movement with certain checks in place.

```cpp
bool move::IKmagic(Eigen::Vector3f objectPose, Eigen::Vector3f handPose){
    //GETING ANGLE CONFIGURATION FROM ROBOT
    updateAngles();

    //CALCULATION FORWARD KINEMATICS AND nablaH
    update();

    //CALCULATING JACOBIAN
    Eigen::MatrixXf J(3, numOfAngles);
    J = Jacobian();

    //CALCULATING THE STEP SIZE
    float stepSizeH = calcStepSize(1);

    //CALCULATING THE NEW ANGLE CONFIGURATION
    float speed = joint_speed;
    Eigen::Vector3f Xpoint = (objectPose - handPose) * speed;
    Eigen::MatrixXf Jcross = J.transpose()*((J*J.transpose()).inverse());

    Eigen::Vector4f newAngleChange;

    //CHECK IF OBJECT IS OUT OF REACH
    //CALCULATE OFFSET TO NEXT ANGLE CONFIGURATION
    if(magnitude(objectPose) > 0.329){
        //OBJECT IS OUT OF REACH
        calcNablaE(objectPose);
        float stepSizeE = calcStepSizeE(1, nablaE, objectPose);

        newAngleChange =  - (stepSizeE * nablaE) - (stepSizeH * (I - (Jcross*J)) * nablaH);
    }
    else{
        //OBJECT IS WITHIN REACH OF THE ARM
        newAngleChange = Jcross * Xpoint - (stepSizeH * (I - (Jcross*J)) * nablaH);
    }

    //CHANGING ANGLES TO NEW POSITION
    for(int i = 0; i < 4; i++){
        newAngle[i] = angles[i] + newAngleChange[i];
        if(newAngle[i] < lowerAngleRange[i] || newAngle[i] > upperAngleRange[i]){
            std::cout << "the new angle " << i+1 << " is out of range" << std::endl;
        }
        else if(
            newAngle[i] != newAngle[i]){ //equation returns false if newAngle[i] is nan(not a number)
        }
        else{
            motion.setAngles(joints[i], newAngle[i], speed);
        }
    }

    if(newAngleChange.squaredNorm() < 0.01f ) {
        return true;
    }
    else{
        return false;
    }
}
```

Figure 7: The C++ code responsible for the pointing mechanism of the Pepper robot.

3.7.2 ROS

ROS is a collection of software tools and libraries, which is used to facilitate an easy abstraction of the hardware of a robot (specifically, the Pepper robot in this case) to help make robot applications. It includes package management and a client library with many implementations for C++, Python and Lisp related applications.

3.7.3 Python

Python is used for the high-level operational use of the robot. In general, it is used for two things: the connection between the learning system (the Perl server) and for using the C++ implementations. This way, someone can easily make use of the robot without needing extensive knowledge of C++. The connection with the learning system is used for the research topic in this thesis. The learning system gets the input from the people speaking and moving objects through the Pepper robot. Responses from this learning system will also be handled by the Python code whenever necessary.

3.7.4 Perl

The learning system itself is written in Perl. This is because the person that wrote the learning system was not a programmer originally, and Perl is a good programming language for working with text. This means that in order for someone to implement new ideas into the learning system and/or change existing parameters, they need to have (extensive) knowledge of Perl. This, coupled with complex, non-commented code, is a tough challenge. By following a crash course [16] on Perl, a lot of syntax was more clear, but it also brought up another problem with Perl: the language itself can be written in such a way that it is hard to read for someone who did not write the code themselves, especially if the code is not well-documented. Therefore, the only real benefit of learning Perl in this situation was to be able to locate small bugs more easily and then report them to the person that wrote the code. Luckily, this usually only took a day or two at the most, but it would have been a lot more optimal if the bugs could have been resolved during the experimentation sessions with the Pepper robot.

However, after having more and more sessions with the Pepper robot, a couple of small tweaks were implemented on the spot to resolve some bugs, without causing new bugs to appear. One of these bugs (introduced by the new pointing mechanism in the learning system) was that if an object was lost during object tracking (e.g., it fell off the table or the object tracking just could not detect the object), an uninitialized value was used in a subtraction. A simple check whether or not the values were initialized sufficed to resolve this issue, as seen in the code in Figure 8.

```
   my ($vec1, $vec2) = @_;
   my @outvec = ();
   foreach my $i (0 .. scalar(@$vec1)-1) {
-    $outvec[$i] = $vec1->[$i] - $vec2->[$i];
+    if (length $vec1->[$i] && length $vec2->[$i]) {
+        $outvec[$i] = $vec1->[$i] - $vec2->[$i];
+    }
   }
   return \@outvec;
 }
```

Figure 8: Simple check to see if the values used in a subtraction are actually initialized.

Another bug was found when the new looking while pointing mechanism was implemented. This meant that after the looking and pointing of the Pepper robot was successful, the robot would also have to go back to its original position, whether or not the user actually gave an answer or if the user took too long. When the user takes too long, a timeout is issued in the Python code. This resulted in the system bugging out and causing it to basically lock itself. The solution was for the learning system to send a signal to the Pepper robot to stop the interaction, just as it does when the user gives an answer (which is processed by the learning system). The Python code was already present for dealing with a “stop interaction” signal, so only a small addition was needed in the learning system, which is seen in the code in Figure 9.

```
        # check if interaction is too long ago:
        if ($interact{'on'} == 1 && $interact{'t'} < $jx->{'time'} - $params{'interactl'}) { $interact{'on'} = 0 }
-       if ($params{'interact'} && $interact{'on'}) {$c->send("HTTP/1.0 200 OK\n\n"); next} # ignore movements when in interaction mode!
+       #if ($params{'interact'} && $interact{'on'}) {$c->send("HTTP/1.0 200 OK\n\n"); next} # ignore movements when in interaction mode!
+       if ($params{'interact'} && $interact{'on'}) {$jresp->{'stopinteraction'} = 1}#; print ">>>>><stop pointing>>>>\n"}
```

Figure 9: The "stop interaction" signal that has to be send to the Pepper robot's system to have it stop pointing (and looking) at an object.

4 Results

After numerous tweaks to both the learning system and the system driving the Pepper robot itself, a final result for the AAMAS workshop (as mentioned in the introduction) was achieved. Here, the findings of the current system that were tested on multiple unknown users (people that had no idea about the current system that was being tested), were presented. These tests also brought about a new addition to the Pepper robot (adding a sound) that was previously thought of, but was not implemented, as it is was not deemed as a necessary addition.

4.1 Pepper tweaks

The resulting system that drives the Pepper robot was tweaked numerous times throughout all the experimentation sessions. One of the main additions to the system is the looking while pointing. This introduced a whole new set of problems, but they were eventually solved. The major problems with this were the object tracker, which would see all the objects moving when the Pepper robot would be looking at a specific object and the pointing mechanism itself, which was not as accurate or would sometimes not do anything at all.

During the sessions with the unknown users, it was found that the pointing mechanism was generally ignored. One of the reasons, when asked about, was that it seemed like an idle animation of the Pepper robot. Because it takes a while for the Pepper robot to actually start pointing (due to the different calculations and the Google speech recognition software), the user is already doing other actions with the objects.

Thus, the following problems needed to be solved:

- Objects appear to move when the Pepper robot moves its head
- It takes too long for the Pepper robot to start pointing
- The pointing mechanism of the Pepper robot is ignored

4.1.1 Solution 1 – Disable object tracking

The solution for the first problem, the object tracking while looking, was to simply shut off the object tracker while the Pepper robot is in a pointing state and turn it back on when the robot is back in its original position. The key here is to time it just right, so that the system behind the Pepper robot knows the last known position of the object it has to point and look at. Otherwise, the system would not point at all, instead throwing an error that the Pepper robot is trying to point to an object that does not exist.

4.1.2 Solution 2 – Two-PC setup

The solution for the delay in the whole system was more complicated, as it involved some tweaking in the C++ code of the Pepper robot system. This reduced the delay, but only by a second or so. A host of new problems were also introduced when trying to set-up everything with a two-PC setup. This would increase the framerate of the object tracker, making it easier for the system to track the user's actions. The problem with the two-PC setup was mostly related to the timing, but was not solved in time. The result was just using the normal setup, which is the one laptop running everything.

4.1.3 Solution 3 – Add a sound

The solution for the third problem is a simple sound that the Pepper robot makes before, during and after the pointing mechanism. As soon as the learning system sends a signal to the Pepper robot's system that it should start pointing at a certain object, a curious sound is played, which is just a sound file that makes the noise of a person being curious. This signals the start of the pointing mechanism. When the arm and hand are in the correct position, another curious sound is played to give extra emphasis on the fact that the Pepper robot is confused and needs more information. When the pointing mechanism ends and the Pepper robot goes back to its initial position, another sound is played to emphasize that the robot is using the input to learn. This was found to be really helpful in conveying to the user that the Pepper robot is about to do something (the first sound), but also that the robot is trying to ask something about an object (the second sound).

4.2 Learning system

The learning system was also tweaked numerous times before getting to a result that worked without major faults. The major difference from the initial system is the pointing mechanism, which changed from the previously mentioned x-axis system, where an object had to be further apart to another object on the x-axis before the learning system would consider it pointable. This was tweakable with a parameter. The new system uses a cosine-type function, which checks for the angle from a reference point (the piece of paper with the symbols on to indicate the ground surface, in this case the table, as seen in Figure 2, top right of the table) and then checks whether or not the angle between two objects is greater than a certain value. This value is also tweakable with a parameter, just like in the last pointing system.

During testing with the unknown users, it was found that the learning system did a much better job at learning the words and actions than previously thought. These users were much quicker in their actions than when the experimentation sessions were held. It was first thought that this would be a major problem, as the learning system would not be able to keep up with the quick succession of actions with objects together with the speech. However, this turned out to be perfectly fine. Even though the learning system would get wrongly recognized words by the Google speech recognition software or words associated with a previous action (due to the learning system not being able to keep up), it was able to fix itself when the user continued on and after a while it completely removed those words from the lexicon. This is the intention of the learning system, and it worked flawlessly.

4.3 Experimental sessions

During three sessions with the Pepper robot, six participants were told to move three objects around on the table while telling the robot what they are doing. They were told not to stack the objects and to try and spread out the objects. Other than that, they were not told any other instructions as to see if the system was intuitive enough to make it clear that the robot is asking for feedback and is learning the objects and actions.

During the first session, the participants did not react to the pointing mechanism of the Pepper robot. They said that it looked more like an idle animation. One participant also noted that it looked more as if the Pepper robot was looking at its watch. Afterwards, it was decided that some sounds should be added to accompany the pointing mechanism.

During the second session, the participants reacted to the pointing mechanism with the added sound, but still were not certain what they should do. The idea afterwards was to also enable the face tracking in order to be able to look at the user and have the Pepper robot say the word it has learned for a certain object when pointing at it.

During the third session, one participant did not react to the pointing mechanism, but did react when the Pepper robot was looking at them. However, the participant did not know what to do. Likewise, the other participant also did not know what to do when the Pepper robot started moving, pointing, looking and uttering the word. They both just continued on with what they were doing. A new idea was mentioned where the Pepper robot would also lean forward when pointing at the object. Another idea was to try and force the hand to stay closed, except for the pointing finger, by duct taping it closed.

Conclusion

The question was asked whether or not the system could be improved to make it more clear to the user that the Pepper robot is asking for feedback. As such, many improvements were made to the learning system. Furthermore, a lot of new additions were added to the system behind the Pepper robot. These improvements and additions came as a result of numerous sessions with the robot, both internally and with participants that did not know that it would ask for feedback.

One of these improvements was the new pointing mechanism. This makes it so objects that are too close to each other or placed in niche situations, like the imaginary line problem, are not pointable and the Pepper robot waits for new interactions by the participant. The reason for this is that in these situations, it is not clear which object the robot is pointing at. Another improvement was the addition of sound. The current system has a sound before it starts to point, after the Pepper robot's arm stops moving and is pointing at an object, and after the robot is done pointing. This, accompanied with tracking the participant's face while pointing, made it vastly more clear that the robot is asking for some input.

However, even with these improvements, the participants still found the movement of the Pepper robot to be more of an idle animation than anything meaningful. One of the reasons that this is the case, is that the Pepper robot cannot stretch its arm fully, so the pointing looks odd. Coupled with not having the ability to move individual fingers, for some participants it did not look like pointing at all. Another big bottleneck in the system is the speech recognition. Because of the German accents of the participants, many words were wrongfully recognized and thus the Pepper robot learned the wrong words for some objects. But, even in their natural language (German), it still recognized the wrong words. A second bottleneck, albeit not as big, is that there is a delay in the whole system. Because the participants are going faster than the system can keep up, the pointing mechanism starts too late. The participants would already be doing another action, so they get confused at what is happening. It might also cause words to be linked with the wrong object. This, however, corrects itself the longer the participant is working with the Pepper robot.

In conclusion, the system did improve its ability to make it clear to the participant that the Pepper robot is asking for feedback. However, it is still far from finalized and needs many more improvements. In the future, the open issues will be further investigated and more testing will be done with participants that have no experience within the field of information technology, as this might give different results compared to the ones who do have experience in that field. Some bottlenecks, like the speech recognition, will naturally improve over the course of time, as this is software from a third party that is in continuous development. Likewise, the hardware of the robot also gets improved as time goes on, with more robot models coming out in the future.

Bibliographical references

[1] K. Lucca and M. P. Wilbourn, “The what and the how: Information-seeking pointing gestures facilitate learning labels and functions,” *Journal of Experimental Child Psychology,* no. 178, pp. 417-436, 2019.

[2] M. Hirschmanner, S. Gross, B. Krenn, F. Neubarth, M. Trapp and M. Vincze, “Grounded Word Learning on a Pepper Robot - Figure 3,” 5 November 2018. [Online]. Available: https://dl.acm.org/citation.cfm?id=3267903. [Accessed 29 March 2019].

[3] “ Oesterreichische Studiengesellschaft fuer Kybernetik (OSGK),” 21 February 1996. [Online]. Available: http://www.osgk.ac.at/. [Accessed 1 April 2019].

[4] “OFAI Services,” [Online]. Available: http://www.ofai.at/services/partners.html. [Accessed 1 April 2019].

[5] “OFAI Research,” [Online]. Available: http://www.ofai.at/research/index.html. [Accessed 1 April 2019].

[6] “OFAI People,” [Online]. Available: http://www.ofai.at/people.html. [Accessed 1 April 2019].

[7] “Pepper the humanoid robot | SoftBank Robotics EMEA,” SoftBank Robotics, [Online]. Available: https://www.softbankrobotics.com/emea/en/pepper. [Accessed 5 May 2019].

[8] M. Tomasello, “Do young children have adult syntactic competence?,” *Cognition,* vol. 3, no. 74, pp. 209-253, 2000.

[9] S. Gross, M. Hirschmanner, B. Krenn, F. Neubarth and M. Zillich, “Action Verb Corpus - Figure 3,” 7-12 May 2018. [Online]. Available: http://www.ofai.at/research/interact/resources/avc/GrossEA-LREC2018.pdf. [Accessed 29 March 2019].

[10] “V4R Library – Automation & Control Institute,” ACIN – TU Wien: Institut für Automatisierungs- und Regelungstechnik, [Online]. Available: https://www.acin.tuwien.ac.at/en/vision-for-robotics/software-tools/v4r-library/. [Accessed 6 May 2019].

[11] “Leap Motion,” Leap Motion, Inc., [Online]. Available: https://www.leapmotion.com/. [Accessed 6 May 2019].

[12] "Google Cloud," Google, [Online]. Available: https://cloud.google.com/speech-to-text/. [Accessed 21 March 2019].

[13] “2D Cameras — Aldebaran 2.4.3.28-r2 documentation,” [Online]. Available: http://doc.aldebaran.com/2-4/family/pepper_technical/video_pep.html. [Accessed 3 May 2019].

[14] “Motors — Aldebaran 2.4.3.28-r2 documentation,” Aldebaran, [Online]. Available: http://doc.aldebaran.com/2-4/family/pepper_technical/motors_pep.html. [Accessed 6 May 2019].

[15] C. Haider, *Object Tracking and Inverse Kinematics via ROS with Humanoid Robot Pepper,* Vienna, 2017.

[16] B. Yorgey, “Perl: A crash course,” [Online]. Available: https://wso.williams.edu/wiki/images/d/d7/Perl-crash-course.pdf. [Accessed 11 April 2019].

Appendices

Description Appendix A

Description Appendix B

A. Description Appendix A

Weekly reports

Date:	25/02/2019 – 01/03/2019
Scheduled tasks:	- Figure out the topic of my thesis - Getting to know the people at OFAI
Progress:	- Figured out the rough outlines of the topic for my thesis - Read some papers surrounding the topic - Followed a couple of tutorials to get to know Python
Problems and bottlenecks:	- I had no experience going through research papers, so this was very painful at the beginning (loss of focus, not understanding whole sections, …)
Solutions:	- Sat together for an informal meeting with two people from OFAI, explaining my difficulty going through papers; they informed me not to concentrate too much on the mathematical side of things and just gloss over any sections I did not understand
Personal reflection:	- I learned to only take in the relevant information from papers and not to worry too much about the heavy sections
Schedule next week:	- Meeting with the people in charge of the robot to draw a plan for my internship - Continue reading about AL (Google Scholar, the web, …) - Continue learning Python (YouTube, Pluralsight, …)

Date:	04/03/2019 – 08/03/2019
Scheduled tasks:	- Meeting 05/03/2019 - Further research into AL - Further practice in Python
Progress:	- Got more insight on how I will proceed with writing the Python modules - Further knowledge of AL - Further knowledge of Python
Problems and bottlenecks:	- Due to my limited background regarding AI in general, I was not able to understand the majority of the meeting

Solutions:	- Tried to keep following the meeting as best as I could and soak in as much information as possible
Personal reflection:	- A lot more explanation will be needed regarding how I should proceed with writing the Python modules
Schedule next week:	- (Meeting with the people in charge of the robot to define which Python modules I want to/need to write) - Gradually get a better understanding of the different Python modules that will need to be written - Further research on AL - Further practice in Python

Date:	11/03/2019 – 15/03/2019
Scheduled tasks:	- Gradually get a better understanding of the different Python modules that will need to be written - Further research on AL - Further practice in Python
Progress:	- Mostly focused on setting up Ubuntu 18.04.2 (using Hyper-V) on my laptop, which is needed for ROS - Followed the wiki tutorials of ROS - Research into the general concepts of AI to have a better overview of where AL is situated
Problems and bottlenecks:	- A lot of troubles with Hyper-V and setting up a shared folder in the same network
Solutions:	- After fiddling around and looking online, I was able to get it to work without having to spend the whole week on this one single problem
Personal reflection:	- I am glad that I started writing everything down when going through online sources regarding AI; this helps me remember the concepts and is also good for the people at OFAI to check on my progress and see if I am stuck on anything
Schedule next week:	- (Meeting with person in charge of the robot --> TBD) - Look into simulating Pepper on my laptop (VM) - Further research on AI in general and AL specifically

Date:	18/03/2019 – 22/03/2019
Scheduled tasks:	- Meeting with person in charge of the robot (20/03) - Look into simulating Pepper on my laptop (VM) - Further research on AI in general and AL specifically
Progress:	- Changed the idea of what I will be doing for the practical side of my thesis - Incorporated feedback of OFAI supervisor into thesis - Read three internal papers that contains all the background information needed to understand their project
Problems and bottlenecks:	- In the beginning of the week, I still did not really have an idea of how I would be doing the practical side of my thesis
Solutions:	- Asking feedback of the OFAI supervisor and then organizing a small meeting to arrange what I will be doing instead of jumping into the pond and figuring it out
Personal reflection:	- Due to asking for feedback, I now have a concrete project to work on instead of endlessly trying to learn Python and trying to simulate a Pepper robot
Schedule next week:	- Introduction to the Pepper system at the technical university on Monday (25/03) at 14:00 - Incorporating the information from the non-published papers into my thesis - Figuring out the research methods with OFAI supervisor

Date:	25/03/2019 – 29/03/2019
Scheduled tasks:	- Acquaintance with Pepper robot - Play around with Pepper robot - Meeting with PXL international coordinator
Progress:	- Getting a lot more information about the Pepper robot due to actually being able to physically work with it
Problems and bottlenecks:	- Hard to go through the code behind the robot because it is not commented
Solutions:	- /
Personal reflection:	- Due to only being able to make experiments on certain days (because of Pepper robot schedule), I have to keep myself busy the other days with writing my thesis
Schedule next week:	- Experiment with Pepper - Come up with some ideas for experiments with Pepper

	- Schedule day for physical access to Pepper

Date:	01/04/2019 – 05/04/2019
Scheduled tasks:	- Experiment with Pepper - Come up with more ideas for experiments with Pepper
Progress:	- Did a lot of experimentation with Pepper, after another person implemented the remarks we had last week - Went through a crash course in Perl to be able to find bugs quicker while experimenting with the physical Pepper robot
Problems and bottlenecks:	- Still had a couple of bugs in the code that we could not figure out, causing some problems with the Pepper robot's pointing mechanism (it did not stop pointing)
Solutions:	- Talk about the remarks with the person who wrote the code - Start learning the code myself so that smaller bugs could be fixed by me during the physical meetings with the Pepper robot
Personal reflection:	- After not being able to do much, I decided to go through a crash course in Perl to be able to start programming again - This also ties into how it would be easier if I myself could fix smaller bugs in the code, so that we can go quickly continue with the experiments
Schedule next week:	- Potential meeting for experimenting with Pepper - Start learning the learning system's code for potential implementation and bug fixing by myself

Date:	11/04/2019 – 12/04/2019
Scheduled tasks:	- Start learning the learning system's code for potential implementation and bug fixing by myself
Progress:	- Played around with the Perl code of the learning system
Problems and bottlenecks:	- Short work week due to taking three days off for family visit
Solutions:	- /
Personal reflection:	- /
Schedule next week:	- Experiment with Pepper on Monday - Keep playing around with the Perl code of the learning system to get a better understanding and think of potential new implementations

Date:	15/04/2019 – 19/04/2019
Scheduled tasks:	- Experimentation with Pepper on Monday
Progress:	- Did a lot of experimentation on Monday with Pepper, but also had a session on Tuesday - A general outline was written for and by the person that has to present the workshop - Some brainstorming for more ideas on improving the system with the person that has to present the workshop
Problems and bottlenecks:	- Brainstorming session was rough, as it was found that a lot of ideas were not possible due to the limitation set by the research paper regarding the workshop
Solutions:	- We had to draw the line and agree that any more difficult implementation would have to be considered after the workshop
Personal reflection:	- Had a really easy time adding lots of new text to my thesis, as we had two experimentation sessions and the person who had to present the workshop joined us during these sessions
Schedule next week:	- Keep adding new text to my thesis - Try to come up with some ideas for after the workshop

Date:	24/04/2019 – 26/04/2019
Scheduled tasks:	- Keep adding new text to my thesis - Try to come up with some ideas for after the workshop
Progress:	- Added a lot of text to prepare my thesis for the language feedback - Incorporated feedback of abstract (= project description) - Tried to solve a pointing problem with the Pepper robot (3.3.1) - Had another meeting with the person in charge of the Pepper robot on Friday (26/04) to test out the solution for the pointing problem
Problems and bottlenecks:	- Could not quite figure out how I would be able to solve the pointing problem on my own
Solutions:	- Asked the person behind the Perl code for help and we both came up with a solution (he implemented it)
Personal reflection:	- Did a good job of asking someone for help when I could not figure something out
Schedule next week:	- Meeting on Friday (03/05) for a change in supervisor - Potential meeting on either Monday or Tuesday with the Pepper robot

Date:	29/04/2019 – 03/05/2019
Scheduled tasks:	- Finish up and send the project description and thesis for evaluation - Meeting on Friday (03/05) with the two supervisors
Progress:	- Had a session with the Pepper robot on Monday, Tuesday and Friday - Meeting with the two supervisors on Friday (see Appendix B)
Problems and bottlenecks:	- Some bugs with the learning system that had to be resolved (found during the sessions with the Pepper robot)
Solutions:	- The person that developed the learning system solved the bugs
Personal reflection:	- It was a good idea to have a meeting on Friday for a general outline on what happens during the last month of the internship
Schedule next week:	- Meeting on Thursday (09/05) with the two supervisors for some feedback on my thesis - Session with the Pepper robot, where internal people will try out the system for us to check whether some things need to be changed (unexpected actions by users that are not used to working with the system)

Date:	06/05/2019 – 10/05/2019
Scheduled tasks:	- Meeting on Thursday (09/05) with the two supervisors for some feedback on my thesis - Session with the Pepper robot, where internal people will try out the system for us to check whether some things need to be changed (unexpected actions by users that are not used to working with the system)
Progress:	- Went over my thesis with the two supervisors and incorporated their feedback - A lot of sessions with the Pepper robot, both with internal people and not
Problems and bottlenecks:	- We were not sure what I could do next now that the workshop is going on and the project might be put on hold (on the Pepper robot's part)
Solutions:	- We will continue to work on a certain section of the Pepper robot if time permits - I will be looking into the pointing mechanism together with the person that wrote that system (in C++)

Personal reflection:	- It was good of me to bring up what I could do next, as I did not want to only be finishing up my thesis for the next couple of weeks; I needed something practical to do
Schedule next week:	- Meeting with person that wrote the pointing mechanism (probably 16/05)

Date:	13/05/2019 – 17/05/2019
Scheduled tasks:	- Meeting with person that wrote the pointing mechanism
Progress:	- Went over the thesis of the person that wrote the pointing mechanism - Went over the C++ code of the person that wrote the pointing mechanism
Problems and bottlenecks:	- /
Solutions:	- /
Personal reflection:	- /
Schedule next week:	- Incorporate language feedback into thesis - Have another session with the Pepper robot

Date:	20/05/2019 – 24/05/2019
Scheduled tasks:	- Incorporate language feedback into thesis - Have another session with the Pepper robot
Progress:	- Incorporated language feedback into the thesis - The Pepper robot now also has face tracking, along with it being able to say the word if it thinks it knows the object
Problems and bottlenecks:	- Had an off-day because I was ill
Solutions:	- /
Personal reflection:	- /
Schedule next week:	- Have another session with the Pepper robot - Finalize thesis

Date:	27/05/2019 – 31/05/2019
Scheduled tasks:	- Have another session with the Pepper robot - Finalize thesis

Progress:	- Had another session with the robot and other participants to see if they reacted to the new improvements of the system - Finalized the thesis
Problems and bottlenecks:	- /
Solutions:	- /
Personal reflection:	- /
Schedule next week:	- /

B. Description Appendix B

Internship discussion

Tracking number:	Internship2019-01
Date:	26/02/2019
Discussion points:	- General topic thesis - Research topic thesis - Rules provided by PXL University of Applied Sciences and Arts - General outline of thesis
Conclusions: Actions: Arrangements:	- Found the two topics for the thesis: AL and Python modules for a robot - Explained the general outline of the thesis - / - Meeting on 05/03/2019
Approval:	Aligned the topics of the thesis with the PXL supervisor.

Tracking number:	Internship2019-02
Date:	25/03/2019
Discussion points:	- New topic for thesis

Conclusions: Actions: Arrangements:	- Discussed the new topic for the thesis. This topic will be more viable, as the current code base would take too much time to understand and implement something into. - / - /
Approval:	Discussed with both a person working with the Pepper robot as with my supervisor.

Tracking number:	Internship2019-03
Date:	03/05/2019
Discussion points:	- Thesis - Last month of the internship
Conclusions: Actions: Arrangements:	- The supervisors will start checking my thesis and adding some feedback what I could change/add - One of the major things to add to the thesis is flesh out the context of the work, as the Pepper robot has a lot of physical constraints (body, add specification figures to thesis). I have to be more specific about the field of view, … - Move the ideas (based on observations) that are not part of the project, e.g. saying “what is this?” to the background information - / - Try to meet up every week for any questions/remarks after feedback is given to the thesis - Meeting on the 9th of May, 2019
Approval:	Discussed with the supervisors from OFAI.